

Universidad Nacional Autónoma de Nicaragua León

Área de conocimiento: Ciencias y Tecnología

Área específica: Ingeniería en Sistemas de Información



Monografía para optar al título de Ingeniero en Sistemas de Información e Ingeniero en Telemática

Implementación de contenedores Docker como herramienta para el uso de Sistema de Gestión de Contenidos (CMS), Área específica: Ingeniería en Sistemas de Información, UNAN – León, 2024

Autor(es):

Br. Jason Antonio Flores Romero 17-00347-0

Br. Roberquis Alberto Torrez Picado 17-01726-0

Br. Junior Antonio Flores Romero 17-00346-0

Tutor(es):

MSc. Alvaro Altamirano

León, Nicaragua, Febrero 2024

“A la Libertad por la Universidad”

Resumen

Dentro de la carrera Ingeniería en Sistemas e Información del Departamento de Computación de la UNAN – León, dentro del componente curricular “Introducción a los CMS” es común utilizar el software XAMPP, esto con el fin de realizar las actividades, guías y prácticas asignadas por el docente en este componente, en lo cual este nos puede generar inconvenientes y en muchas ocasiones es tedioso en relación con los diferentes CMS buscando las versiones adecuadas para empezar a realizar las prácticas.

Por estos motivos, la presente investigación nos plantea Implementación de contenedores Docker como herramienta para el uso de Sistema de Gestión de Contenidos (CMS), en la carrera de Ingeniería en Sistema, Departamento de Computación, UNAN – León, 2023.

Para solucionar esta problemática en este proyecto se realizó lo siguiente:

- Se brindó información teórica relevante del uso de contenedores Docker, ya que, es importante saber que es Docker, su arquitectura y los usos que se le pueden dar. De igual forma se plasmó información básica e intermedia sobre su uso permitiendo fortalecer conocimientos sobre su configuración e implementación.
- Se proporcionó información práctica de Docker mediante las guías prácticas del curso “Labs – Docker for the Absolute Begginer – Hands On” de la plataforma web KodeKloud, las cuales fueron resueltas y explicadas.
- Se facilitaron archivos “docker-compose.yml” esto con la finalidad de crear ejemplos de contenedores, con el objetivo de facilitar la resolución de las prácticas del componente de Sistema de Gestión de Contenidos (CMS). Estos contenedores se implementaron a partir de imágenes de Docker Hub, por lo que tomando como como apoyo esta investigación y analizando su contenido, es posible realizar y elaborar contenedores para otros componentes y áreas en general.

Dedicatoria

Dedico este trabajo primeramente a Dios por la fuerza y sabiduría que me ha brindado para culminar mis estudios universitarios. También se lo dedico a mis padres por el sacrificio que hicieron y porque siempre me brindaron su apoyo incondicional para cumplir mis metas.

Junior A. Flores

Dedico este trabajo a Dios por darme la fuerza y la persistencia para seguir adelante, también se lo dedico a mis padres por siempre apoyarme en cada paso que he dado hacia un mejor futuro

Roberquis A. Torrez

El presente trabajo se lo dedico a Dios, porque me ha dado las fuerzas necesarias para poder culminar mis estudios universitarios y cumplir una meta más en mi vida. Dedicado también a mi familia y amistades, especialmente a mis padres, que me han apoyado y motivado desde el inicio de mis estudios para culminar esta etapa de mi vida.

Jason A. Flores

Agradecimiento

Agradezco a Dios por la vida, salud y sabiduría que me brindo en todo momento.

Agradezco a mis padres que siempre estuvieron para mí, el sacrificio y esfuerzo que hicieron para pudiera finalizar mi carrera universitaria.

Agradezco a mi tutor por la paciencia, ayuda y apoyo que me brindo a lo largo del desarrollo del presente trabajo.

Junior A. Flores

Agradezco a Dios por regalarme salud y vida, también se lo agradezco a mis padres que lo han dado todo por mí y gracias a ellos estoy alcanzando esta meta tan importante

Roberquis A. Torrez

Primeramente, agradezco a Dios por regalarme vida, salud, paciencia y sabiduría en todo momento.

Agradezco a mis padres, que en todo momento estuvieron pendiente de mí y todo el sacrificio que tuvieron que hacer para que yo pudiera salir adelante, gracias a ellos puedo decir que he culminado con esta meta de mi vida y toda esta satisfacción se la debo a ellos.

Agradeciendo a nuestro tutor por toda la enseñanza, valores y ayudarnos a llevar a cabo para culminar este trabajo.

Jason A. Flores



Índice General

Introducción	1
Planteamiento del problema	2
Pregunta General:	3
Preguntas Específicas:	3
Antecedentes	4
Justificación	6
Objetivos.....	7
Objetivo General.....	7
Objetivos Específicos	7
Marco Teórico	8
Docker.....	8
Arquitectura de Docker	8
¿Para qué se puede utilizar Docker?	9
Facilidades de los contenedores de Docker.....	10
Casos de uso de los contenedores	11
Imágenes	13
Definición	13
Listar Imágenes.....	13
Descargar imágenes	15
Copia de seguridad	16
Restaurar	16
Eliminar	17
Etiquetar	17
Contenedores	18
Definición	18
Ejecutar contenedor	19
Crear contenedor	19
Listar contenedores.....	20
Inspeccionar.....	21
Pausar	22
Detener.....	23
Iniciar.....	23
Exponer puertos.....	23
Eliminar	24
Exportar.....	24
Importar	25
Ejecutar comando dentro de un contenedor en ejecución	25
Dockerfile	26
Definición	26
FROM.....	27
MAINTAINER	27
RUN.....	27



CMD	27
EXPOSE	28
ENTRYPOINT	28
VOLUME	29
ENV	29
USER	29
WORKDIR	30
Creación de una imagen	30
Docker Compose	32
Definición	32
Características que posee Docker-compose	33
Contenido de Docker-compose	33
Estructura de un archivo docker-compose.yml	33
Comandos básicos para Docker Compose	40
Almacenamiento de Docker	41
Elegir el tipo de montaje adecuado	42
Mas detalles sobre los tipos de montaje	43
Redes Docker	46
Controladores de red	46
Docker Hub	48
Definición	48
Características de Docker Hub	49
Comandos básicos para trabajar con Docker Hub	50
Inicio de sesión en Docker Hub	50
Cerrar sesión en Docker Hub	50
Buscar imágenes	50
Descargar imágenes	51
Cargar imágenes	52
Content Management System (CMS)	53
WordPress	53
PrestaShop	54
Drupal	54
Joomla	55
Moodle	55
Diseño metodológico	57
Etapas de recolección de datos	57
Etapas de selección de herramientas a utilizar	57
Recursos hardware	57
Recursos Software	58
Etapas de elaboración y desarrollo	59
Organización del desarrollo	59
Formato de guías "Labs- Docker for the Absolute Begginer - Hands On"	59
Formato de la documentación de los contenedores	60



Etapa de prueba y funcionamiento.....	61
Desarrollo	61
Desarrollo de las guías "Labs - Docker for the Absolute Beginner – Hands On" de la plataforma KodeKloud	61
Guía de laboratorio 1 - Comandos básicos de Docker.....	62
Guía de laboratorio 2 – Comandos de ejecución de Docker.....	70
Guía de laboratorio 3 – Imágenes en Docker.....	73
Guía de laboratorio 4 – variables de entorno	81
Guía de laboratorio 5 – CMD y puntos de entrada.....	85
Guía de laboratorio 6 – Docker compose	88
Guía de laboratorio 7 – Almacenamiento de Docker.....	94
Guía de laboratorio 8 – Redes Docker	99
Creación de contenedores para CMS	105
WordPress	106
Joomla.....	107
Drupal.....	109
PrestaShop	110
Ejemplos de guías prácticas.....	114
WordPress	114
Enunciados:.....	128
Soluciones:.....	128
Joomla.....	151
Enunciados.....	157
Soluciones:.....	157
Drupal.....	174
Enunciados.....	182
Soluciones.....	183
PrestaShop	194
Enunciados:.....	200
Soluciones:.....	200
Moodle.....	209
Enunciados.....	215
Soluciones.....	216
Programa "Menú de CMS"	233
Conclusiones.....	244
Recomendaciones.....	245
Referencias bibliográficas	246
Anexos.....	248
Electiva VI: Comercio Electrónico	248
Glosario.....	257
Cronograma de actividades.....	261



Índice de ilustraciones

<i>Ilustración 1: Arquitectura de Docker</i>	8
<i>Ilustración 2: Estructura de Docker</i>	9
<i>Ilustración 3: Comparación entre máquina virtual y contenedor</i>	12
<i>Ilustración 4: Listar imágenes descargadas de Docker</i>	14
<i>Ilustración 5: Buscar imágenes en repositorio de Docker Hub</i>	14
<i>Ilustración 6: Descargar imagen de Docker</i>	15
<i>Ilustración 7: Backup de imagen</i>	16
<i>Ilustración 8: Restaurar imagen</i>	16
<i>Ilustración 9: Eliminar una imagen</i>	17
<i>Ilustración 10: Etiquetado de una imagen</i>	18
<i>Ilustración 11: Ejecución de un contenedor</i>	19
<i>Ilustración 12: Creación de un contenedor</i>	20
<i>Ilustración 13: Listar contenedores</i>	21
<i>Ilustración 14: Inspección de un contenedor</i>	22
<i>Ilustración 15: Pausar un contenedor</i>	22
<i>Ilustración 16: Detener un contenedor</i>	23
<i>Ilustración 17: Iniciar un contenedor</i>	23
<i>Ilustración 18: Especificar un puerto</i>	24
<i>Ilustración 19: Eliminar contenedor (por nombre)</i>	24
<i>Ilustración 20: Exportación de un contenedor</i>	25
<i>Ilustración 21: Importación de un contenedor</i>	25
<i>Ilustración 22: Ejecución de un comando interno en contenedor</i>	26
<i>Ilustración 23: Dockerfile</i>	26
<i>Ilustración 24: Creación de una imagen con docker build</i>	31
<i>Ilustración 25: Demostración de la imagen creada</i>	32
<i>Ilustración 26: Docker compose</i>	32
<i>Ilustración 27: Estructura de Docker-compose.yml</i>	34
<i>Ilustración 28: Nombre de servicios en docker-compose.yml</i>	35
<i>Ilustración 29: Opción imagen en docker-compose.yml</i>	35
<i>Ilustración 30: Opción build en docker-compose.yml</i>	36
<i>Ilustración 31: Opciones context y dockerfile en docker-compose.yml</i>	36
<i>Ilustración 32: Opción command en docker-compose.yml</i>	36
<i>Ilustración 33: Opción ports en docker-compose.yml</i>	37
<i>Ilustración 34: Opción expose en docker-compose.yml</i>	37
<i>Ilustración 35: Opción depends_on en docker-compose.yml</i>	38
<i>Ilustración 36: Opción environment en docker-compose.yml</i>	38
<i>Ilustración 37: Opción env_file en docker-compose.yml</i>	39
<i>Ilustración 38: Opción volumes en docker-compose.yml</i>	39
<i>Ilustración 39: Opción restart en docker-compose.yml</i>	40
<i>Ilustración 40: Docker Hub</i>	48
<i>Ilustración 41: Login de Docker hub</i>	50
<i>Ilustración 42: Logout de Docker hub</i>	50



Ilustración 43: Buscar imágenes en Docker hub	51
Ilustración 44: Descargar imágenes de Docker hub	51
Ilustración 45: Insertar tag a una imagen en Docker hub	52
Ilustración 46: Cargar una imagen en Docker hub	53
Ilustración 47: Verificación de la versión de Docker	63
Ilustración 48: Verificación de la cantidad de contenedores activos	63
Ilustración 49: Verificación de la cantidad de imágenes disponibles.....	64
Ilustración 50: Ejecución de un contenedor	64
Ilustración 51: Deteniendo un contenedor	64
Ilustración 52: Cantidad de contenedores en ejecución	65
Ilustración 53: Verificación de los contenedores existentes	65
Ilustración 54: Verificación de la imagen que utiliza un contenedor.....	65
Ilustración 55: Verificación del nombre de un contenedor	66
Ilustración 56: Verificación del ID de un contenedor	66
Ilustración 57: Verificación del estado de un contenedor	66
Ilustración 58: Detener y eliminar todos los contenedores	67
Ilustración 59: Eliminar una imagen.....	67
Ilustración 60: Descargar una imagen	68
Ilustración 61: Ejecución y asignación de nombre a un contenedor	68
Ilustración 62: Detención de contenedores y eliminación de imágenes.....	69
Ilustración 63: Eliminación de todas las imágenes	69
Ilustración 64: Contenedores en ejecución.....	70
Ilustración 65: Imagen que usa un contenedor.....	70
Ilustración 66: Puertos publicados por el contenedor.....	71
Ilustración 67: Número de puertos expuestos en el contenedor	71
Ilustración 68: Numero de puertos expuestos en el host.....	71
Ilustración 69: Búsqueda de imagen KodeKloud/simple-webapp	72
Ilustración 70: Instancia de KodeKloud/simple-webapp con un tag	72
Ilustración 71: Verificación de la cantidad de imágenes disponibles.....	73
Ilustración 72: Verificación del tamaño de una imagen	74
Ilustración 73: Verificación de la etiqueta de una imagen	74
Ilustración 74: Verificación de imagen usada a través del Dockerfile	75
Ilustración 75: Verificación de imagen usada a través de la terminal.....	75
Ilustración 76: Uso del comando COPY en un Dockerfile	75
Ilustración 77: Uso del comando ENTRYPOINT en un Dockerfile	76
Ilustración 78: Uso del comando EXPOSE en un Dockerfile.....	76
Ilustración 79: Creación de la imagen webapp-color	77
Ilustración 80: Ejecución de una instancia de la imagen webapp-color	77
Ilustración 81: Funcionamiento de la aplicación webapp-color	78
Ilustración 82: Verificación del S.O utilizado por una imagen.....	78
Ilustración 83: Tamaño de la imagen webapp-color	79
Ilustración 84: Modificación del Dockerfile de webapp-color	79
Ilustración 85: Construyendo la imagen webapp-color:lite	80



Ilustración 86: Verificación del tamaño de la imagen webapp-color:lite	80
Ilustración 87: Ejecución de contenedor webapp-color:lite con redirección de puertos.....	81
Ilustración 88: Listar contenedores en ejecución.....	81
Ilustración 89: Valor de la variable App_color en un contenedor	82
Ilustración 90: Creación del contenedor blue_app con valor en la variable App_color	82
Ilustración 91: Verificación del nuevo valor de App_color	83
Ilustración 92: Verificación del color en la aplicación	83
Ilustración 93: Documentación de las variables de entorno de imagen mysql en dockerhub.....	84
Ilustración 94: Creación de contenedor asignando valor a MYSQL_ROOT_PASSWORD	84
Ilustración 95: Conociendo las variables de entorno de un contenedor.....	85
Ilustración 96: Localización del Dockerfile de la imagen mysql.....	86
Ilustración 97: Verificación del valor entrypoint en la imagen mysql	86
Ilustración 98: Encontrar Dockerfile de wordpress	86
Ilustración 99: CMD del Dockerfile de la imagen wordpress	87
Ilustración 100: Comando ejecutado al lanzar la imagen de "Ubuntu".....	87
Ilustración 101: Ejecución de una instancia de la imagen ubuntu + "sleep 1000" .	88
Ilustración 102: Creación de contenedor redis	89
Ilustración 103: Creación de contenedor llamado clickcounter vinculando el contenedor redis.....	90
Ilustración 104: Aplicación clickcounter	91
Ilustración 105: Aumentando el número de clicks	91
Ilustración 106: Identificar y detener contenedores	91
Ilustración 107: Deteniendo contenedor redis	91
Ilustración 108: Identificando y eliminando contenedores redis y clickcounter	92
Ilustración 109: Verificación del directorio actual.....	92
Ilustración 110: Creación de docker-compose de redis y clickcounter.....	93
Ilustración 111: Ejecución del docker-compose.....	93
Ilustración 112: Verificación de la creación de los contenedores.....	93
Ilustración 113: Ubicación de contenedores e imagenes	94
Ilustración 114: Identificación de contenedor.....	95
Ilustración 115: Establecer contraseña en la base de datos	95
Ilustración 116: Ejecución de sh get-data.sh con datos	96
Ilustración 117: Ejecución de sh get-data.sh sin datos.....	96
Ilustración 118: Asignación de volumen	97
Ilustración 119: Ejecución de sh get-data.sh con datos	97
Ilustración 120: Ejecución de una nueva instancia de mysql	98
Ilustración 121: Verificación de persistencia de datos.....	98
Ilustración 122: Comprobación de redes existentes.....	99
Ilustración 123: Verificación del ID asociado a la red bridge	100



Ilustración 124: Verificación de creación y ejecución del contenedor alpine-1	100
Ilustración 125: Identificación de la red del contenedor alpine-1	100
Ilustración 126: Inspeccionando la subred configurada en la red bridge	101
Ilustración 127; Creación de contenedor alpine-2 con imagen alpine red none	101
Ilustración 128: Verificación de red none	101
Ilustración 129: Creación de red wp-mysql-network	102
Ilustración 130: Verificación de creación de red wp-mysql-network	102
Ilustración 131: Creación de un nuevo contenedor mysql-db	103
Ilustración 132: Inspección de red del contenedor	103
Ilustración 133: Creación de contenedor webapp vinculado a mysql-db	104
Ilustración 134: Verificación de creación de contenedor webapp y su red	104
Ilustración 135: Verificación de funcionamiento.....	104
Ilustración 136: docker-compose.yml de WordPress	106
Ilustración 137: docker-compose.yml de Joomla.....	107
Ilustración 138: docker-compose.yml de Drupal.....	109
Ilustración 139: docker-compose.yml de PrestaShop	110
Ilustración 140: docker-compose.yml de Moodle.....	112
Ilustración 141: Selección de lenguaje	115
Ilustración 142: Definición de datos	115
Ilustración 143: Sitio creado.....	116
Ilustración 144: Login	116
Ilustración 145: Administración	117
Ilustración 146: Index	117
Ilustración 147: Actualización.....	118
Ilustración 148: Instalar actualizaciones	119
Ilustración 149: Login phpMyAdmin.....	119
Ilustración 150: Página de inicio phpMyAdmin	120
Ilustración 151: Selección base de datos WordPress	120
Ilustración 152: Tablas de base de datos WordPress	121
Ilustración 153: Exportar base de datos	121
Ilustración 154: Backup exportado.....	122
Ilustración 155: Ubicación de backup	122
Ilustración 156: Selección de base de datos WorPress a eliminar.....	122
Ilustración 157: Proceso de eliminación	123
Ilustración 158: Eliminación de base de datos.....	123
Ilustración 159: Error de página	124
Ilustración 160: Asignación de nombre BD	124
Ilustración 161: Creación BD	124
Ilustración 162: Importar BD	125
Ilustración 163: Selección BD	125
Ilustración 164: Abrir BD	126
Ilustración 165: Importar BD	126



Ilustración 166: Importación correcta	127
Ilustración 167: Index	127
Ilustración 168: Añadir entrada	128
Ilustración 169: Imagen destacada	129
Ilustración 170: Seleccionar imagen	129
Ilustración 171: Imagen destacada	130
Ilustración 172: Ingresar título	130
Ilustración 173: Menú principal	131
Ilustración 174: Sitio web cliente	131
Ilustración 175: Título e imagen	132
Ilustración 176: Apariencia	132
Ilustración 177: Opciones de apariencia	133
Ilustración 178: Selección de tema	133
Ilustración 179: Verificar tema	134
Ilustración 180: Personalizar tema	134
Ilustración 181: Estilos	135
Ilustración 182: Selección de estilo	135
Ilustración 183: Guardar estilo	136
Ilustración 184: Verificar estilo	136
Ilustración 185: Ajustes en pantalla principal de WordPress	137
Ilustración 186: Ajustes generales (1)	137
Ilustración 187: Ajustes generales (2)	138
Ilustración 188: Ajustes generales (3)	138
Ilustración 189: Ajustes generales (4)	138
Ilustración 190: Guardar cambios hechos en ajustes generales	139
Ilustración 191: Seleccionando menú de entradas	139
Ilustración 192: Menú de entradas	140
Ilustración 193: Modificando una entrada	140
Ilustración 194: Verificando el cambio realizado a la entrada	141
Ilustración 195: Ubicando comentarios en la pantalla principal de WordPress	141
Ilustración 196: Menú principal de comentarios	142
Ilustración 197: Editando un comentario	142
Ilustración 198: Más detalles de una entrada	143
Ilustración 199: Comentarios de una entrada	143
Ilustración 200: Agregando una nueva entrada	144
Ilustración 201: Nueva entrada	144
Ilustración 202: Verificando la nueva entrada	145
Ilustración 203: Agregando otra entrada	145
Ilustración 204: Verificando la nueva entrada creada	146
Ilustración 205: Ubicando el menú de páginas en la Pantalla principal de WordPress	146
Ilustración 206: Menú de páginas	147
Ilustración 207: Creando una nueva página	147



Ilustración 208: Opción de permitir comentarios.....	148
Ilustración 209: Guardando los cambios.....	148
Ilustración 210: Verificación de la página nueva agregada	149
Ilustración 211: Página nueva (1)	149
Ilustración 212: Página nueva (2)	150
Ilustración 213: Página nueva (3)	150
Ilustración 214: Muestra de si la opción de Permitir comentarios estuviera habilitada	151
Ilustración 215: Configuración del nombre del sitio	152
Ilustración 216: Información vital para el superusuario	153
Ilustración 217: Información requerida de la base de datos del sitio.....	154
Ilustración 218: Configuración de la base de datos del sitio	155
Ilustración 219: Configuración realizada para la instalación.....	156
Ilustración 220: Panel principal de Joomla	157
Ilustración 221: Selección del menú "Artículos".....	158
Ilustración 222: Menú de "Artículos"	158
Ilustración 223: Editor de texto de un artículo.....	159
Ilustración 224: Artículo creado	159
Ilustración 225: Visualización del artículo	160
Ilustración 226: Panel de control del menú sistema	160
Ilustración 227: Menú de extensiones	161
Ilustración 228: Implementar galería desde la web	161
Ilustración 229: Extension "Simple_Image_Gallery "	162
Ilustración 230: Opciones en la sección "contenido"	163
Ilustración 231: Creación de la carpeta galería	163
Ilustración 232: Contenido de la carpeta galería	164
Ilustración 233: Creación de un artículo para la galería	164
Ilustración 234: verificación de la extensión de galería	165
Ilustración 235: panel izquierdo del menú principal de Joomla	165
Ilustración 236: Panel de menús.....	166
Ilustración 237: Agregando un nuevo menú	166
Ilustración 238: Verificación del nuevo menú	167
Ilustración 239: Visualización de la galería	167
Ilustración 240: Agregando "URL embebida" al menú principal	168
Ilustración 241: Visualización del menú de noticias	168
Ilustración 242: Código fuente de un video de YouTube	168
Ilustración 243: Configuración del plugin "Editor-TinyMCE "	169
Ilustración 244: Nuevo artículo con video de YouTube	169
Ilustración 245: Asignación del enlace del video	170
Ilustración 246: Menú de usuarios	170
Ilustración 247: Usuarios existentes	171
Ilustración 248: Nuevos usuarios agregados.....	171
Ilustración 249: Agregando código de un mapa a un articulo.....	172



Ilustración 250: Menú de contactos	173
Ilustración 251: Información para un nuevo contacto	173
Ilustración 252: Agregando elemento de contactos al menú principal	174
Ilustración 253: Verificación del menú contactos.....	174
Ilustración 254: Localización de la carpeta de Drupal	175
Ilustración 255: Selección de lenguaje	175
Ilustración 256: Selección del tipo de perfil de instalación	176
Ilustración 257: Errores al momento de la instalación	176
Ilustración 258: Creación de la carpeta files y el archivo settings.php	177
Ilustración 259: Permisos a la carpeta "sites"	177
Ilustración 260: Configuración de la base de datos.....	178
Ilustración 261: Configuración del sitio (1).....	178
Ilustración 262: Configuración del sitio (2).....	179
Ilustración 263: Menú principal de Drupal.....	179
Ilustración 264: Identificar ID del contenedor de mysql.....	180
Ilustración 265: Accediendo al contenedor de mysql	180
Ilustración 266: Creando backup de drupal	181
Ilustración 267: Artículo de prueba.....	181
Ilustración 268: Cargando backup de drupal	182
Ilustración 269: Menú principal sin el artículo de prueba.....	182
Ilustración 270: Panel de "content"	184
Ilustración 271: Agregando contenido	184
Ilustración 272: Nuevo artículo y nueva página.....	184
Ilustración 273: Verificación de artículo nuevo creado	185
Ilustración 274: Verificación de la nueva página creada	185
Ilustración 275: Menú principal de Drupal con los nuevos elementos creados....	186
Ilustración 276: Agregando roles	186
Ilustración 277: Nuevo rol creado	187
Ilustración 278: Accesos del rol "minieditor"	187
Ilustración 279: Agregando usuarios.....	188
Ilustración 280: Datos del nuevo usuario.....	188
Ilustración 281: Rol del nuevo usuario.....	189
Ilustración 282: Usuarios existentes	189
Ilustración 283: Sesión iniciada con el nuevo usuario	190
Ilustración 284: Descarga del módulo IMCE.....	190
Ilustración 285: Ubicación del módulo IMCE en el administrador de archivos....	191
Ilustración 286: Ubicación del módulo IMCE en Drupal	191
Ilustración 287: Instalación del módulo IMCE	192
Ilustración 288: Ubicación del formato "HTML básico"	192
Ilustración 289: Botones disponibles al momento crear un articulo	192
Ilustración 290: IMCE disponible para crear artículos/páginas.....	193
Ilustración 291: IMCE listo para agregar imágenes al momento de crear artículos/páginas	193



Ilustración 292: Selección de idioma	194
Ilustración 293: Aceptar terminos y condiciones.....	194
Ilustración 294: Datos de cuenta.....	195
Ilustración 295: Contenido de prueba	196
Ilustración 296: Conexión a base de datos	196
Ilustración 297: Creando sitio.....	197
Ilustración 298: Cambiar nombre de carpeta admin	197
Ilustración 299: Eliminar carpeta Install	198
Ilustración 300: Login	198
Ilustración 301: Tienda Online	199
Ilustración 302: Administración	199
Ilustración 303: Agregar producto	200
Ilustración 304: Productos predeterminados	201
Ilustración 305: Agregar producto	201
Ilustración 306: Clientes	202
Ilustración 307: Listado clientes	203
Ilustración 308: Agregar cliente.....	203
Ilustración 309: Información de pedidos	204
Ilustración 310: Realizar pedido.....	204
Ilustración 311: Carrito de cliente.....	205
Ilustración 312: Pedidos realizados	205
Ilustración 313: Inicio de sesión como cliente.....	206
Ilustración 314: Agregar productos	206
Ilustración 315: Finalizar compra	207
Ilustración 316: Datos personales.....	207
Ilustración 317: Selección de logística	208
Ilustración 318: Realizar pago.....	208
Ilustración 319: Habilitar pago.....	209
Ilustración 320: Pedido confirmado.....	209
Ilustración 321: Index	210
Ilustración 322: Login	210
Ilustración 323: Página de inicio como admin.....	211
Ilustración 324: Cambiar idioma.....	211
Ilustración 325: Cambiar credenciales	212
Ilustración 326: Id contenedor.....	212
Ilustración 327: Ingresar a contenedor mariadb	213
Ilustración 328: Ejecutar backup	214
Ilustración 329: Añadir foro de prueba	214
Ilustración 330: Restaurar base de datos	215
Ilustración 331: Index	215
Ilustración 332: Subir plugin.....	216
Ilustración 333: Instalar plugin	216
Ilustración 334: Validación	217



Ilustración 335: Listado de paquetes	217
Ilustración 336: Comprobación de plugin.....	218
Ilustración 337: Cambiar tema	218
Ilustración 338: Aplicar cambios	219
Ilustración 339: Plugin File	219
Ilustración 340: Sección file	220
Ilustración 341: Carrusel y logo.....	221
Ilustración 342: Imagen logo	221
Ilustración 343: Imágenes carrusel	222
Ilustración 344: Verificación	222
Ilustración 345: Creación de curso.....	223
Ilustración 346: Añadir recurso	223
Ilustración 347: Actividades y recursos.....	224
Ilustración 348: Foro y recurso ZIP	225
Ilustración 349: Actividad tarea	225
Ilustración 350: Definir parámetros	226
Ilustración 351: Visualización.....	226
Ilustración 352: Actividad cuestionario.....	227
Ilustración 353: Banco de preguntas.....	228
Ilustración 354: Crear preguntas.....	228
Ilustración 355: Agregar clave de curso.....	229
Ilustración 356: Listado de métodos de inscripción	229
Ilustración 357: Usuario estudiante.....	230
Ilustración 358: Ingresar como estudiante	230
Ilustración 359: Curso restringido	231
Ilustración 360: Material de curso	231
Ilustración 361: Entrega de tarea	232
Ilustración 362: Cuestionario finalizado	232
Ilustración 363: Funciones principales del programa	233
Ilustración 364: Opción de "Levantar un CMS".....	234
Ilustración 365: Levantando un CMS	235
Ilustración 366: Información de los contenedores habilitados.....	236
Ilustración 367: Opciones sobre la detención de un contenedor.....	238
Ilustración 368: Información sobre los contenedores activos y detenidos	239
Ilustración 369: Deteniendo un contenedor	240
Ilustración 370: Opción de crear un respaldo	241
Ilustración 371: Carpeta de respaldos en el CMS	242
Ilustración 372: Respaldo creado.....	242
Ilustración 373: Eliminar datos de un contenedor.....	243



Introducción

Docker es una plataforma abierta utilizada para desarrollar, enviar y ejecutar aplicaciones que dan la posibilidad de contener un proyecto, aislándolo del resto de aplicaciones que estén en el mismo entorno. Hoy en día se ha convertido en la plataforma de contenedores software más populares. Actualmente no se ha implementado directamente en ninguno de los componentes de la carrera de Ingeniería en Sistemas del Departamento de Computación de la UNAN-León, por lo que los estudiantes cada vez que necesiten el uso de una aplicación en concreto se requiere de la preparación del entorno de donde cada estudiante debe darse la tarea de buscar todas las dependencias, archivos de configuración, instalación de paquetes entre otras cosas; sin mencionar los posibles errores de compatibilidad que puede presentar a los estudiantes por no contar con una computadora que posea componentes Hardware medianamente buenos.

El presente trabajo plantea como objetivo general, implementar contenedores Docker como herramienta de entorno y preparación liviana como propuesta de apoyo al proceso de enseñanza-aprendizaje del componente curricular “Introducción a los CMS” en la carrera de Ingeniería en Sistemas, Dpto. Computación, UNAN-León.

Para la realización del objetivo general el trabajo se encuentra orientado en 3 diferentes enfoques, el primero se centra en documentar toda la parte teórica sobre los contenedores Docker, tales como son los conceptos básicos y técnicos, en los cuales se encuentran imágenes y contenedores Docker, un Dockerfile y un Docker compose. El segundo enfoque se orienta en solucionar las guías del curso “Labs – Docker for the Absolute Beginner – Hands On” que se encuentra en la plataforma KodeKloud, estas guías abordan temas interesantes como son los comandos básicos, imágenes, variables de entorno, que es Docker Compose, Almacenamiento y redes de Docker, como tercer enfoque se elaboraran las prácticas, soluciones y ejemplos de contenedores Docker que pueden ser utilizados en el componente de Introducción a los CMS de la carrera de Ingeniería en Sistemas de la UNAN-León.



Planteamiento del problema

En esta sección se abordará el motivo por el cual se está realizando el presente trabajo investigativo, debido a que en un componente en particular que se imparte en la carrera de Ingeniería en Sistemas del Departamento de Computación UNAN-León se utilizan sistemas de gestión de contenido o también llamados CMS, los cuales permiten crear un entorno de trabajo para la creación y administración de contenidos, principalmente páginas web, por parte de los administradores, editores, participantes y demás usuarios, una vez seleccionado el CMS, el estudiante debe realizar las diferentes prácticas que se delegan a lo largo de este componente (distinta prácticas dependiendo del CMS escogido por el maestro) por lo cual debe estar instalado, configurado y contar con los servicios necesarios para su correcto funcionamiento en nuestras máquinas.

El problema radica en que estos CMS cuentan con diferentes requisitos de sistemas, además de que suelen llegar a ocupar mucho espacio de almacenamiento, si bien alguno de ellos funciona bastante bien con sistemas actuales como es el caso de WordPress, hay otros que funcionan con requisitos un poco más antiguos como es en el caso de Drupal, Prestashop, Joomla o algunas versiones de Moodle, que incluso aunque se cuente con los requerimientos necesarios muchas veces estos CMS arrojan problemas de compatibilidad con el sistema, muchas veces se puede solucionar utilizando un servidor local (como XAMPP para Windows o LAMP para Linux) pero aun así no evita que se sigan presentando problemas para los cuales se necesita una configuración mucho más minuciosa y muchas veces no estamos seguros de donde este el problema lo cual provoca que las practicas se entreguen con retrasos o que el estudiante opte por no entregar la practica en cuestión, todo debido a que en la mayoría de los casos no se toma en cuenta el uso de tecnologías como contenedores Docker que está orientado a la virtualización liviana y agiliza la creación del entorno apta para la instalación de cualquier programa.



Pregunta General:

- ¿Cómo implementar contenedores Docker como herramienta para el uso de Sistema de Gestión de Contenidos (CMS), como propuesta de apoyo al proceso enseñanza-aprendizaje en la carrera de Ingeniería en Sistemas, Dpto. Computación, UNAN-León?

Preguntas Específicas:

- ¿Qué conocimientos se deben tener acerca de los contenedores Docker?
- ¿Qué guías se podrían elaborar para facilitar al aprendizaje práctico de contenedores Docker en estudiantes de la carrera de Ingeniería en Sistemas?



Antecedentes

A continuación, se describirán los trabajos monográficos que han sido realizados en relación con la tecnología de contenedores Docker y que servirán como apoyo en el desarrollo del presente trabajo, de los cuales se pueden mencionar:

Delgado y Mendoza (2017) desarrollaron el siguiente trabajo: "Distribución Linux Ubuntu 16.04 que incluya paquetes de software pre-instalados utilizados en las asignaturas de las áreas de Redes y Programación de las carreras que ofrece el Departamento de Computación de la UNAN-León" (Nicaragua). El objetivo que abarca esta investigación es la creación de una distribución de Linux personalizada, con las herramientas pre-instaladas que serán usadas en las asignaturas de las áreas de redes y programación de las carreras que ofrece el Departamento de Computación, UNAN – León. Llegando a la conclusión de que la forma más fácil de crear una distribución Linux es utilizar una distribución previa como base, a la vez utilizar una herramienta que permita la remasterización de distribuciones, tales como, UCK, Remasterys y Ubuntu Builder.

Mazariego y Mora (2022) realizaron el siguiente trabajo: "Implementación de contenedores Docker como herramienta de virtualización liviana para el apoyo del proceso enseñanza-aprendizaje en la carrera de Ingeniería en Telemática, Dpto. Computación, UNAN-León " (Nicaragua). Esta investigación se centró en la Implementación de contenedores Docker como herramienta de virtualización liviana para la realización de las prácticas asignadas por los docentes en muchos de los componentes curriculares: Software como un Servicio, Gestión de Red y Comercio Electrónico. de la carrera de Ingeniería en Telemática. Llegando a la conclusión de que La elaboración de los ejemplos de contenedores Docker que pueden ser utilizados en componentes curriculares como Software como un Servicio, Gestión de Red y Comercio Electrónico constituyen una ayuda a los estudiantes que cuentan con equipos con especificaciones técnicas limitadas a desarrollar sus prácticas utilizando dichos contenedores, además de servir como referencia para que ellos puedan desarrollar sus propios contenedores Docker.



Rivera et.al. (2022) desarrollaron el siguiente trabajo: “CREACIÓN DE AMBIENTES AISLADOS DE PRUEBAS PARA LA EJECUCIÓN DE APLICACIONES EN PHPV8, LARAVELV8, .NETV5 Y RUBY ON RAILS V6 EMPLEANDO DOCKER-COMPOSE VERSIÓN 2” (Nicaragua). Este proyecto plantea como objetivo implementar contenedores Docker en lugar de máquinas virtuales, ya que algunos de los problemas que los estudiantes del Dpto. de Computación de la UNAN-León han enfrentado a lo largo de la carrera es no contar con computadoras con grandes características, así podrán elaborar sus tareas asignadas, ya que estos contenedores no utilizan muchos recursos de cómputo, usando solo lo mínimo para correr cualquier aplicación y montar un entorno más fluido y fácil. Se llegó a la conclusión de que los ambientes de desarrollo creados con Docker Compose pueden ser usados como material de apoyo para la realización de tareas asignadas en algunos componentes de las carreras que ofrece el Dpto. de Computación, logrando también implementarse en desarrollo de aplicaciones web y obteniendo una mejor comprensión acerca de esta tecnología.



Justificación

El enfoque del presente trabajo es hacer uso de contenedores Docker livianos y aislados del sistema que cuenten con el software y los servicios necesarios para la creación del entorno donde se trabaje las practicas del componente “Introducción a los CMS”, así haremos que el consumo de recursos sea mínimo y evitaremos los problemas de compatibilidad con el sistema, esto favorecerá al proceso de enseñanza en la carrera de Ingeniería en sistemas permitiendo que los estudiantes puedan centrarse en el uso de los CMS y menos en buscar solución de problemas relacionados a la compatibilidad.

Aunque actualmente existen otros motores que permiten la contenerización, esta investigación se enfoca en Docker debido a que es el motor de contenedores que más ha llamado la atención de los autores del presente, debido a que es fácil de utilizar, existe mucha información en internet respecto al tema y provee recursos muy útiles como es el caso del Docker Hub que almacena muchas imágenes las cuales podemos descargar según sea nuestra necesidad.



Objetivos

Objetivo General

- Implementar contenedores Docker como herramienta para el uso de Sistemas de Gestión de Contenido (CMS), como propuesta de apoyo al proceso enseñanza-aprendizaje en la carrera de Ingeniería en Sistemas, Dpto. Computación, UNAN-León.

Objetivos Específicos

- Analizar los aspectos técnicos necesarios para la implementación de contenedores Docker en cada uno de los CMS, que se abordan en el componente de Introducción a los CMS.
- Elaborar las guías del curso “Labs – Docker for the absolute Beginner – Hands On” de la Plataforma KodeKlode como un medio de aprendizaje práctico de Docker.
- Documentar todos los procedimientos que debe realizar el estudiante para la implementación de los contenedores Docker que contienen los diferentes CMS.
- Elaborar una serie de guías con contenedores Docker que pueden ser utilizados en el componente curricular “Introducción a los CMS”.



Marco Teórico

Docker

Es un motor de contenerización de código abierto, que automatiza el empaquetado, el envío y la implementación de cualquier aplicación de software que se presenta como contenedores livianos, portátiles y autosuficientes, que se ejecutaran prácticamente en cualquier computadora. (Raj Pethuru, 2015)

Arquitectura de Docker

Docker es una aplicación de cliente-servidor, donde el cliente se comunica con el servidor o demonio de Docker, que, a su vez, hace todo el trabajo. Al demonio de Docker también se le es llamado como Docker Engine. Docker se envía con un cliente de línea de comandos binarios, Docker, así como una API RESTFULL (interfaz que dos sistemas de computación utilizan para intercambiar información de manera segura a través de internet) para interactuar con el Daemon. Puede ejecutar el demonio y el cliente de Docker en el mismo host o conectar su cliente de Docker local a un demonio remoto que se ejecuta en otro host. (Turnbull, 2017).

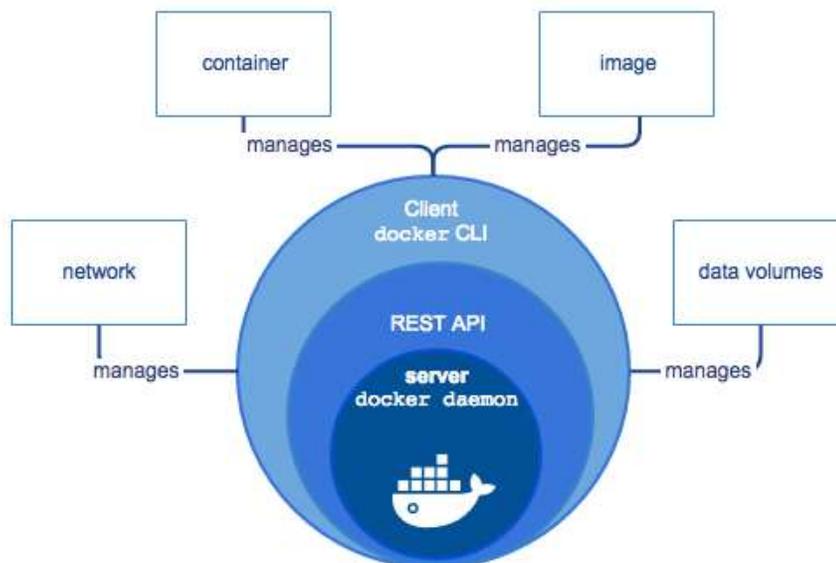


Ilustración 1: Arquitectura de Docker

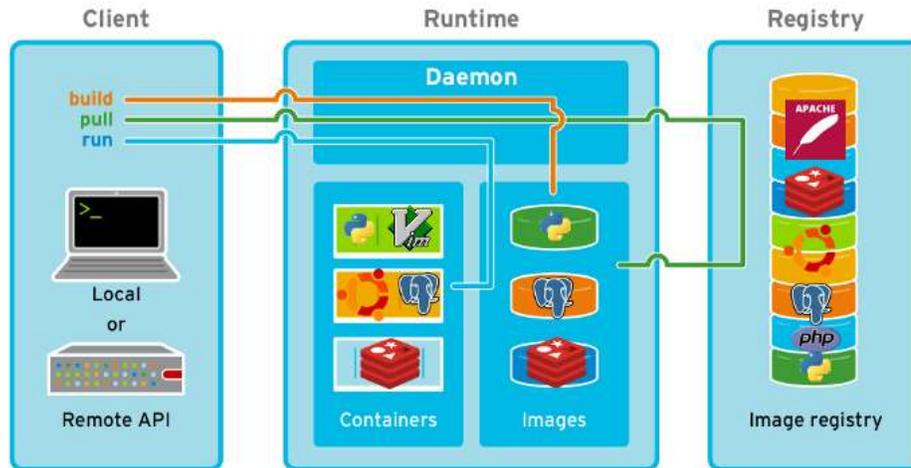


Ilustración 2: Estructura de Docker

¿Para qué se puede utilizar Docker?

- Ayudar a que su flujo de trabajo de desarrollo y construcción local sea más rápido, más eficiente y ligero. Los desarrolladores locales pueden crear, ejecutar y compartir contenedores Docker. Los contenedores pueden construirse en desarrollo y promoverse a entornos de prueba y, a su vez, a producción.
- Ejecución de servicios y aplicaciones independientes de forma coherente en varios entornos, un concepto especialmente útil en arquitecturas e implementaciones orientadas a servicios que dependen en gran medida de microservicios.
- Uso de Docker para crear instancias aisladas para ejecutar pruebas como, por ejemplo, las lanzadas por una suite de integración continua (CI) como Jenkins CI.
- Creación y prueba de aplicaciones y arquitecturas complejas en un host local antes de la implementación en un entorno de producción. • Creación de una infraestructura de plataforma como servicio (PAAS) multiusuario.
- Proporcionar entornos, sandbox autónomos y livianos para desarrollar, probar y enseñar tecnologías, como el shell de Unix o un lenguaje de programación.



- Aplicaciones de Software como un Servicio.
- Despliegues de hosts a hiperscala de alto rendimiento. (Turnbull, 2017)

Facilidades de los contenedores de Docker

Los contenedores son una forma de virtualización del sistema operativo. Un solo contenedor se puede usar para ejecutar cualquier cosa, desde un microservicio o un proceso de software a una aplicación de mayor tamaño. Dentro de un contenedor se encuentran todos los ejecutables, el código binario, las bibliotecas y los archivos de configuración necesarios. (Turnbull, 2017)

Los contenedores no contienen imágenes del sistema operativo. Esto los hace más ligeros y portátiles, con una sobrecarga significativamente menor. En implementaciones de aplicaciones de mayor tamaño, se pueden poner en marcha varios contenedores como uno o varios clústeres de contenedores. Estos clústeres se pueden gestionar mediante un orquestador de contenedores, como Kubernetes.

Algunos de los beneficios de los contenedores son:

- **Menos sobrecarga**

Los contenedores requieren menos recursos del sistema que los entornos de máquinas virtuales tradicionales o de hardware porque no incluyen imágenes del sistema operativo.

- **Mayor portabilidad**

Las aplicaciones que se ejecutan en contenedores se pueden poner en marcha fácilmente en sistemas operativos y plataformas de hardware diferentes.

- **Funcionamiento más constante**

Los equipos de DevOps saben que las aplicaciones en contenedores van a ejecutarse igual, independientemente de dónde se pongan en marcha.



- **Mayor eficiencia**
Los contenedores permiten poner en marcha, aplicar parches o escalar las aplicaciones con mayor rapidez.
- **Mejor desarrollo de aplicaciones**
Los contenedores respaldan los esfuerzos ágiles y de DevOps para acelerar los ciclos de desarrollo, prueba y producción

Casos de uso de los contenedores

A continuación, se mencionan algunas de las formas más habituales en las que diversas organizaciones usan los contenedores:

- **El rehospedaje de las aplicaciones existentes en arquitecturas de nube modernas**
Algunas organizaciones utilizan contenedores para migrar las aplicaciones existentes a entornos más modernos. Aunque esta práctica ofrece algunos de los beneficios básicos de la virtualización de sistemas operativos, no ofrece todas las ventajas de una arquitectura de aplicaciones modular basada en contenedores.
- **Refactorización de las aplicaciones existentes para contenedores**
Aunque la refactorización requiere mucho más que la migración del rehospedaje, ofrece todos los beneficios de un entorno de contenedores.
- **Desarrollo de nuevas aplicaciones nativas del contenedor**
Al igual que la refactorización, este método permite disfrutar de todos los beneficios de los contenedores.
- **Más compatibilidad con las arquitecturas de microservicios**
Las aplicaciones distribuidas y los microservicios se pueden aislar, poner



en marcha y escalar más fácilmente utilizando elementos básicos de contenedores individuales.

- **Soporte de DevOps para la integración y la puesta en marcha continuas (CI/CD)**

La tecnología de contenedores permite la creación, la prueba y la puesta en marcha optimizadas a partir de las mismas imágenes de contenedores.

- **Una puesta en marcha más sencilla de tareas y trabajos repetitivos**

Los contenedores se ponen en marcha para dar soporte a uno o varios procesos parecidos que, a menudo, se ejecutan en segundo plano, como las funciones ETL o los lotes de tareas.

Los contenedores son similares a las máquinas virtuales, excepto que estos no son sistemas operativos completos. Los contenedores generalmente solo incluyen los paquetes y aplicaciones del sistema operativo necesarios. Por lo general, no contienen un sistema operativo completo o virtualización de hardware, es por eso que estos son "ligeros". (Charge, 2020).

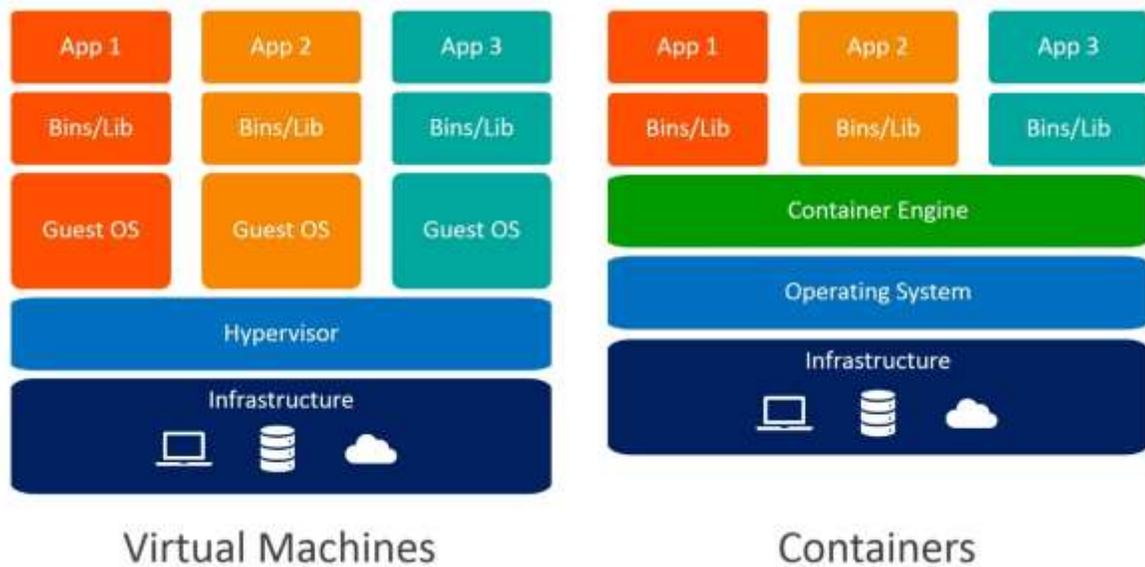


Ilustración 3: Comparación entre máquina virtual y contenedor



Imágenes

Definición

Las imágenes son utilizadas como plantillas para crear contenedores. Las imágenes contendrán todo lo que requiere nuestro proceso o procesos para funcionar correctamente. Estos componentes pueden ser archivos binarios, bibliotecas, archivos de configuración, etc., que pueden ser parte de los archivos del sistema operativo o simplemente componentes creados por usted mismo para la aplicación.

Las imágenes, como las plantillas, son inmutables. Esto significa que no cambian entre ejecuciones. Cada vez que utilicemos una imagen obtendremos los mismos resultados, solo cambiaremos configuración y entorno para gestionar el comportamiento de diferentes procesos entre entornos. (Ramírez, 2020).

Listar Imágenes

Para listar imágenes que tiene en su servidor Docker utilice el siguiente comando:

- `docker images`

Opciones para la acción "images":

-a: muestra todas las imágenes.

-f: filtra la salida con un filtro específico.

-q: muestra solo los identificadores. [OBJ]

Ejemplo:



```
vector@Inspiron-3542: ~$ sudo docker images -a
[sudo] contraseña para vector:
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
ubuntu              latest         58db3edaf2be   2 weeks ago    77.8MB
alpine              latest         042a816809aa   4 weeks ago    7.05MB
hello-world        latest         feb5d9fea6a5   16 months ago  13.3kB
vector@Inspiron-3542: ~$
```

Ilustración 4: Listar imágenes descargadas de Docker

La información es la siguiente:

REPOSITORY: nombre de la imagen.

TAG: etiqueta o versión de la imagen.

IMAGE ID: identificador de la imagen, por defecto es un identificador corto.

CREATED: El tiempo que ha pasado desde la creación de la imagen.

SIZE: tamaño de la imagen comprimida en el sistema. (González, 2017)

Si prefiere buscar una imagen desde el repositorio a través de la terminal utilice el siguiente comando.

`docker search [nombre-imagen]` Opciones para la acción "search":

`-f`: filtra la salida con un filtro especificado

`--limit`: limita el número de resultados obtenidos.

Ejemplo:

```
vector@Inspiron-3542: ~$ docker search ubuntu --limit 5
NAME                DESCRIPTION                               STARS   OFFICIAL   AUTOMATED
ubuntu              Ubuntu is a Debian-based Linux operating sys.. 15573   [OK]
websphere-liberty   WebSphere Liberty multi-architecture images .. 291     [OK]
neurodebian         NeuroDebian provides neuroscience research s.. 98      [OK]
open-liberty        Open Liberty multi-architecture images based.. 56      [OK]
ubuntu-debootstrap  DEPRECATED; use "ubuntu" instead          50      [OK]
vector@Inspiron-3542: ~$
```

Ilustración 5: Buscar imágenes en repositorio de Docker Hub



La información es la siguiente:

NAME nombre de la imagen, si no es oficial mostrará el usuario que la creo.

DESCRIPTION: breve descripción de la imagen.

STARS: valoraciones de la imagen.

OFICIAL: indica si es oficial o no.

AUTOMATED: indica si es una imagen automatizada.

Descargar imágenes

Para descargar una imagen utilice el siguiente comando:

- `docker pull [nombre-imagen]`

Ejemplo:

Proceso de descarga de la imagen oficial del servidor web Ubuntu:

```
Jason@Inspiron-3542: ~  
Jason@Inspiron-3542:~$ sudo docker pull ubuntu  
[sudo] contraseña para Jason:  
Using default tag: latest  
latest: Pulling from library/ubuntu  
677076032cca: Pull complete  
Digest: sha256:9a0bdde4188b896a372804be2384015e90e3f84906b750c1a53539b585fbbe7f  
Status: Downloaded newer image for ubuntu:latest  
docker.io/library/ubuntu:latest  
Jason@Inspiron-3542:~$ sudo docker pull alpine  
Using default tag: latest  
latest: Pulling from library/alpine  
8921db27df28: Pull complete  
Digest: sha256:c8deccde20bcc319a17c28ce674b80b9b7e945a8b460a11272b7f0a3e0d6a746  
Status: Downloaded newer image for alpine:latest  
docker.io/library/alpine:latest  
Jason@Inspiron-3542:~$
```

Ilustración 6: Descargar imagen de Docker



Copia de seguridad

Para guardar una imagen utilice el siguiente comando:

- `docker save [nombre-de-la-imagen] > archivo.tar`

Ejemplo:

Proceso de guardado de una imagen de Ubuntu:

```
vector@Inspiron-3542: ~/Documentos
┌───┐
│ jason@Inspiron-3542: ~/Documentos | Q | ≡ | - | □ | × |
└───┘
jason@Inspiron-3542:~/Documentos$ ls
C# 'instalacion de docker y docker compose'
jason@Inspiron-3542:~/Documentos$ docker save ubuntu > cs_ubuntu.tar
jason@Inspiron-3542:~/Documentos$ ls
C# cs_ubuntu.tar 'instalacion de docker y docker compose'
jason@Inspiron-3542:~/Documentos$
```

Ilustración 7: Backup de imagen

Restaurar

Para restaurar una copia de seguridad de una imagen utilice el siguiente comando:

- `docker import [archivo.tar][nombre-de-la-imagen]`

Ejemplo:

Proceso de importado de la imagen previamente guardada:

```
vector@Inspiron-3542: ~/Documentos
┌───┐
│ jason@Inspiron-3542: ~/Documentos | Q | ≡ | - | □ | × |
└───┘
jason@Inspiron-3542:~/Documentos$ docker images -a
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
alpine        latest   042a816809aa   4 weeks ago   7.05MB
hello-world   latest   feb5d9fea6a5   16 months ago 13.3kB
jason@Inspiron-3542:~/Documentos$ docker import cs_ubuntu.tar ubuntu
sha256:a6c760dce44b889911ebc464cda55a769cba4644d7835779231d91acffb2d5cd
jason@Inspiron-3542:~/Documentos$ docker images -a
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
ubuntu        latest   a6c760dce44b   3 seconds ago  80.3MB
alpine        latest   042a816809aa   4 weeks ago   7.05MB
hello-world   latest   feb5d9fea6a5   16 months ago 13.3kB
jason@Inspiron-3542:~/Documentos$
```

Ilustración 8: Restaurar imagen



Eliminar

Para eliminar una imagen utilice el siguiente comando:

➤ `docker rmi [nombre-de-la-imagen/ID-de-la-imagen]`

Opción para la acción “rmi”:

-f : fuerza el proceso de eliminación de una imagen

Ejemplo:

Proceso de eliminación de la imagen Ubuntu:

```
vector@Inspiron-3542: ~/Documentos
vector@Inspiron-3542:~/Documentos$ docker images -a
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
ubuntu        latest   58db3edaf2be   2 weeks ago   77.8MB
alpine        latest   042a816809aa   4 weeks ago   7.05MB
hello-world    latest   feb5d9fea6a5   16 months ago 13.3kB
vector@Inspiron-3542:~/Documentos$ docker rmi ubuntu
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:9a0bdde4188b896a372804be2384015e90e3f84906b750c1a53539b585fbb7f
Deleted: sha256:58db3edaf2be6e80f628796355b1bdeaf8bea1692b402f48b7e7b8d1ff100b02
Deleted: sha256:c5ff2d88f67954bdcf1cfdd46fe3d683858d69c2cadd6660812edfc83726c654
vector@Inspiron-3542:~/Documentos$ docker images -a
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
alpine        latest   042a816809aa   4 weeks ago   7.05MB
hello-world    latest   feb5d9fea6a5   16 months ago 13.3kB
vector@Inspiron-3542:~/Documentos$
```

Ilustración 9: Eliminar una imagen

Etiquetar

Para añadir una versión a una imagen usa el siguiente comando:

➤ `docker tag [nombre-imagen] [nombre-de-la-imagen:tag]`

Ejemplo:

En este ejemplo se observa el proceso del etiquetado de una imagen, como base se tomó la imagen de Alpine, y como una nueva instancia se creó la imagen con etiqueta “latest1”.



```
jason@Inspiron-3542:~/Documentos$ docker images -a
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
ubuntu        latest    a6c760dce44b  3 seconds ago 80.3MB
alpine        latest    042a816809aa  4 weeks ago   7.05MB
hello-world   latest    feb5d9fea6a5  16 months ago 13.3kB
jason@Inspiron-3542:~/Documentos$ docker tag alpine latest1
jason@Inspiron-3542:~/Documentos$ docker images -a
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
ubuntu        latest    a6c760dce44b  17 minutes ago 80.3MB
latest1       latest    042a816809aa  4 weeks ago   7.05MB
alpine        latest    042a816809aa  4 weeks ago   7.05MB
hello-world   latest    feb5d9fea6a5  16 months ago 13.3kB
jason@Inspiron-3542:~/Documentos$
```

Ilustración 10: Etiquetado de una imagen

Contenedores

Definición

Los contenedores son un encapsulamiento de una aplicación con sus dependencias. A primera vista, parecen ser solo una forma liviana de máquinas virtuales (VM), al igual que un VM, un contenedor contiene una instancia aislada de un sistema operativo (SO), que puede utilizar para la ejecución de una aplicación.

No obstante, los contenedores tienen varias ventajas que permiten casos de uso que para una máquina virtual son difíciles o imposibles.

- Los contenedores comparten recursos con el sistema operativo host, lo cual lo convierte en una orden de magnitud más eficiente. Estos contenedores se pueden iniciar y todo en una fracción de segundo. Las aplicaciones que se ejecutan en contenedores incurrir en poca o ninguna sobrecarga. En comparación con las aplicaciones los contenedores también se ejecutan de una forma nativa en el sistema anfitrión.
- La portabilidad que presentan los contenedores hace que tengan un gran potencial cuando se trata de eliminar toda una clase de errores causados por cambios sutiles en del entorno de ejecución.



Sintetizando el propósito de un contenedor, es que las aplicaciones sean portátiles y autónomas (Mouat, 2016).

Ejecutar contenedor

- Docker run [nombre-de-la-imagen]

Comandos suplementarios

- --name: nombre-del-contenedor
- -it : contenedor interactuar
- -d : ejecución en segundo plano
- -v : volumen
- -p puertos expuestos

Ejemplo:

```
jason@Inspiron-3542: ~  
jason@Inspiron-3542: ~ x jason@Inspiron-3542: ~ x jason@Inspiron-3542: ~ x  
jason@Inspiron-3542: ~$ docker run -i -t alpine /bin/sh  
/# ls  
bin  etc  lib  mnt  proc  run  srv  tmp  var  
dev  home media opt  root  sbin sys  usr  
/#
```

Ilustración 11: Ejecución de un contenedor

En el ejemplo anterior mostrado se muestra en ejecución de un contenedor de ALPINE, primero se revisa que la imagen existe internamente, sino se descarga desde el repositorio de dockerhub, la acción “run” crea y ejecuta el contenedor a diferencia de la acción “create” que solo crea el contenedor, sin embargo, ambas se orientan a distintos enfoques.

Crear contenedor

Para la creación de un contenedor se usa:

- Docker create [opciones] [nombre-de-la-imagen]



Ejemplo:

Proceso de creación de un contenedor del servidor web utilizando el parámetro de la exposición de puertos 8081 para el anfitrión y 80 para el contenedor, esto se verá mucho más a fondo en el transcurso del documento.

```
vector@Inspiron-3542: ~  
vector@Inspiron-3542: ~$ docker create -p 8081:80 alpine  
5e704dfaed5b90dbd6e91e68610c307a0d1e7af628795e98083eef0d2729342c  
vector@Inspiron-3542: ~$
```

Ilustración 12: Creación de un contenedor

Es importante mencionar que “create” a diferencia de “run” permite crear y establecer distintos parámetros para futuras ejecuciones del contenedor con “run” solo se establecen los parámetros de la actual ejecución del contenedor.

Listar contenedores

Para listar únicamente los contenedores en ejecución utilice el comando:

➤ Docker ps

Opciones de parámetros para la acción “ps” :

-a : lista todos los contenedores sin importa estado.

-f : filtra la salida a partir de un filtro en específico.

-n : muestra los últimos contenedores creados.

-l : muestra el ultimo contenedor creado.

-s : muestra el tamaño total.

Ejemplo:

Listado de contenedores en la máquina de ejemplos:



```
jaso@inspiron-3542:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
5e794dfa8d5b   alpine   "/bin/sh"               2 minutes ago Created
66cd4220be8c   alpine   "/bin/sh"               4 minutes ago Exited (0) 4 minutes ago
858ce0e675f4   alpine   "/bin/sh"               4 minutes ago Created
ca17bd31c99b   alpine   "/bin/sh"               5 minutes ago Created
c489df0edde9   alpine   "/bin/sh"               29 minutes ago Created
66bb2ee7a392   alpine   "/bin/sh"               33 minutes ago Exited (137) 27 minutes ago
24606eb8cb17   alpine   "/bin/sh"               2 hours ago   Exited (137) 35 minutes ago
8fa8080b4f2    nginx   "/docker-entrypoint..." 2 hours ago   Exited (0) 33 minutes ago
c96eb925a5b0   alpine   "--name"                 2 hours ago   Created
35e173d2a868   alpine   "-it"                    2 hours ago   Created
200911881ca1   alpine   "-d"                     2 hours ago   Created
69f4bde0a0c4   alpine   "/bin/sh"               2 hours ago   Exited (0) 2 hours ago
3f3d27fbc5b1   hello-world "-d"                     2 hours ago   Created
6627da2cf217   hello-world "/hello"                 2 hours ago   Exited (0) 2 hours ago
ef9c5a842f5b   ubuntu   "-it"                    2 hours ago   Created
31758a835532   ubuntu   "-d"                     2 hours ago   Created
22bbb5b41933   ubuntu   "-it -d"                 2 hours ago   Created
84fe06b08ef    hello-world "/hello"                 2 hours ago   Exited (0) 2 hours ago
c2f648e98e4a   hello-world "/hello"                 5 weeks ago   Exited (0) 5 weeks ago
7557a0ee883d   hello-world "/hello"                 5 weeks ago   Exited (0) 5 weeks ago
jaso@inspiron-3542:~$ docker ps -l
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES
5e794dfa8d5b   alpine   "/bin/sh"               2 minutes ago Created        distracted_goldberg
```

Ilustración 13: Listar contenedores

Significado de las columnas plasmadas en el ejemplo anterior:

CONTAINER ID: identificador del contenedor.

IMAGE: imagen base que se usó para crear el contenedor.

COMMAND: es el comando de inicio del contenedor.

CREATED: este comando indica hace cuanto tiempo se ejecutó el contenedor.

STATUS: indica el estado actual del contenedor.

PORTS: indica los puertos expuestos del “host” y del contenedor.

NAME: nombre asignado al contenedor.

Inspeccionar

Para inspeccionar un contenedor utilice el comando:

- Docker inspect [ID-del-contenedores]

Ejemplo:

Uso de “inspect” para visualizar las configuraciones y valores de los contenedores:



```
Jason@Inspiron-3542: ~  
Jason@Inspiron-3542: ~  
Jason@Inspiron-3542:~$ docker ps -l  
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES  
5e704dfaed5b   alpine   "/bin/sh" 12 minutes ago Created     
distracted_goldberg  
Jason@Inspiron-3542:~$ docker inspect 5e704dfaed5b  
[  
  {  
    "Id": "5e704dfaed5b90dbd0e91e68610c307a0d1e7af628795e98083eef0d2729342c",  
    "Created": "2023-02-11T03:34:16.891609544Z",  
    "Path": "/bin/sh",  
    "Args": [],  
    "State": {  
      "Status": "created",  
      "Running": false,  
      "Paused": false,  
      "Restarting": false,  
      "OOMKilled": false,  
      "Dead": false,  
      "Pid": 0,  
      "ExitCode": 0,  
      "Error": "",  
      "StartedAt": "0001-01-01T00:00:00Z",  
      "FinishedAt": "0001-01-01T00:00:00Z"  
    },  
    "Image": "sha256:042a816809aac8d0f7d7cacac7965782ee2ecac3f21bcf9f24b1de1a7387b769",  
    "ResolvConfPath": "",  
    "HostnamePath": "",  
    "HostsPath": "",  
    "LogPath": "",  
    "Name": "/distracted_goldberg",  
    "RestartCount": 0,  
    "Driver": "overlay2",  
    "Platform": "linux",  
    "MountLabel": ""  
  }  
]
```

Ilustración 14: Inspección de un contenedor

Pausar

Para pausar la ejecución de un contenedor utilice el comando:

- Docker pause [nombre-contenedor/id-contenedor]

Ejemplo:

Proceso para pausar un contenedor, usando su id:

```
Jason@Inspiron-3542:~$ docker ps  
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES  
f87bf8c56ab5   alpine   "/bin/sh" 4 seconds ago Up 3 seconds  
mycontainer  
Jason@Inspiron-3542:~$ docker pause f87  
f87  
Jason@Inspiron-3542:~$ docker ps  
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES  
f87bf8c56ab5   alpine   "/bin/sh" 15 seconds ago Up 14 seconds (Paused)  
mycontainer  
Jason@Inspiron-3542:~$
```

Ilustración 15: Pausar un contenedor

Como podemos ver no es necesario el escribir todo el ID completo, simplemente una parte es suficiente.



Detener

Para detener un contenedor use:

- Docker stop [ID-contenedor]

Ejemplo:

Proceso de detención de un contenedor por ID.

```
Jason@Inspiron-3542:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
f87bf8c56ab5  alpine   "/bin/sh" 15 seconds ago  Up 14 seconds (Paused)   mycontainer

Jason@Inspiron-3542:~$ docker stop f87
f87

Jason@Inspiron-3542:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
f87bf8c56ab5  alpine   "/bin/sh" 2 minutes ago  Exited (137) 9 seconds ago   mycontainer
```

Ilustración 16: Detener un contenedor

Iniciar

Para iniciar un contenedor use:

- Docker start [ID-contenedor]

Ejemplo:

Proceso para iniciar contenedor por su identificador (ID):

```
Jason@Inspiron-3542:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
f87bf8c56ab5  alpine   "/bin/sh" 2 minutes ago  Exited (137) 9 seconds ago   mycontainer

Jason@Inspiron-3542:~$ docker start f87
f87

Jason@Inspiron-3542:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
f87bf8c56ab5  alpine   "/bin/sh" 3 minutes ago  Up 4 seconds   mycontainer
```

Ilustración 17: Iniciar un contenedor

Exponer puertos

Para exponer puertos en el host y en su contenedor al ejecutarlo, utilice la siguiente sintaxis:



- Docker run -p [puerto-del-host: puerto-del-contenedor] [nombre-de-la-imagen]

Ejemplo:

Proceso de la ejecución de un contenedor ALPINE exponiendo el puerto 8181 para el anfitrión y el puerto 80 para el contenedor a partir del ID:

```
Jason@Inspiron-3542:~$ docker run -itd -p 8080:80 --name puerto alpine
3dd476f96a367ff3d2e525225ced5d7db479d4434d7b4e91ddebaf71b57cc67
Jason@Inspiron-3542:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS                               NAMES
3dd476f96a3   alpine   "/bin/sh" 5 seconds ago  Up 3 seconds  0.0.0.0:8080->80/tcp, :::8080->80/tcp  puerto
```

Ilustración 18: Especificar un puerto

Eliminar

Para poder eliminar un contenedor debe usar el siguiente comando:

- Docker rm [ID-del-contenedor]

Ejemplo:

Proceso de eliminación del contenedor anterior a partir de su identificador (ID):

```
Jason@Inspiron-3542:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS                               NAMES
3dd476f96a3   alpine   "/bin/sh" 3 minutes ago  Exited (137) 44 seconds ago  puerto
0bc203fbb752   alpine   "/bin/sh" 3 minutes ago  Exited (0) 3 minutes ago  cont_puert
3901cc5e0205   alpine   "/bin/sh" 4 minutes ago  Exited (0) 4 minutes ago  admiring_villani
f87bf8c56ab5   alpine   "/bin/sh" 10 minutes ago  Exited (137) 5 minutes ago  mycontainer
Jason@Inspiron-3542:~$ docker rm puerto
puerto
Jason@Inspiron-3542:~$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS                               NAMES
0bc203fbb752   alpine   "/bin/sh" 3 minutes ago  Exited (0) 3 minutes ago  cont_puert
3901cc5e0205   alpine   "/bin/sh" 4 minutes ago  Exited (0) 4 minutes ago  admiring_villani
f87bf8c56ab5   alpine   "/bin/sh" 10 minutes ago  Exited (137) 5 minutes ago  mycontainer
```

Ilustración 19: Eliminar contenedor (por nombre)

Exportar

Para poder exportar un contenedor use:

- Docker export [ID-del-contenedor] > archivo.tar

Ejemplo:

Proceso de exportación de un contenedor:



```
Jason@Inspiron-3542: ~/Documentos
Jason@Inspiron-3542:~/Documentos$ ls
cs_ubuntu.tar 'instalacion de docker y docker compose'
Jason@Inspiron-3542:~/Documentos$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
0bc203fbb752   alpine   "/bin/sh" 9 minutes ago   Exited (0) 9 minutes ago
3901cc5e0205   alpine   "/bin/sh" 9 minutes ago   Exited (0) 9 minutes ago
f87bf8c56ab5   alpine   "/bin/sh" 15 minutes ago   Exited (137) 10 minutes ago
Jason@Inspiron-3542:~/Documentos$ docker export 0b > exp_alp.tar
Jason@Inspiron-3542:~/Documentos$ ls
cs_ubuntu.tar  exp_alp.tar 'instalacion de docker y docker compose'
```

Ilustración 20: Exportación de un contenedor

Importar

Para importar un contenedor utilice el comando:

- Docker import [archivo.tar] [contenedor:tag]

Ejemplo:

Proceso de importación de un contenedor:

```
Jason@Inspiron-3542: ~/Documentos
Jason@Inspiron-3542:~/Documentos$ ls
cs_ubuntu.tar  exp_alp.tar 'instalacion de docker y docker compose'
Jason@Inspiron-3542:~/Documentos$ docker import exp_alp.tar Inportado:latest
sha256:f8ced7d95b782bb81fcd96179bdd2df28ca41794c247941c7c13248ad732fa3
Jason@Inspiron-3542:~/Documentos$
```

Ilustración 21: Importación de un contenedor

Ejecutar comando dentro de un contenedor en ejecución

Para poder ejecutar un comando dentro de un contenedor que se encuentra en ejecución usaremos:

- Docker exec [opciones] [nombre-del-contenedor]

Algunas de las opciones para la acción de “exec” son:

-d : ejecución en segundo plano del comando.

-i : modo interactivo del contenedor.

-t : asigna una terminal.

Ejemplo:



Proceso de uso del modo interactivo en un contenedor:

```
vector jason@Inspiron-3542: ~  
jason@Inspiron-3542:~$ docker ps  
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS        NAMES  
296e94d1e587   alpine   "/bin/sh"               51 seconds ago Up 50 seconds  
jason@Inspiron-3542:~$ docker exec -it prueba /bin/sh  
/#
```

Ilustración 22: Ejecución de un comando interno en contenedor

Dockerfile

Definición

Un dockerfile es simplemente un archivo de texto plano el cual contiene un conjunto de comandos definidos previamente por el usuario, que al momento de ser ejecutados por el comando de **Docker image build** estos ensamblan una imagen de contenedor (McKendrick & Gallagher, 2017)

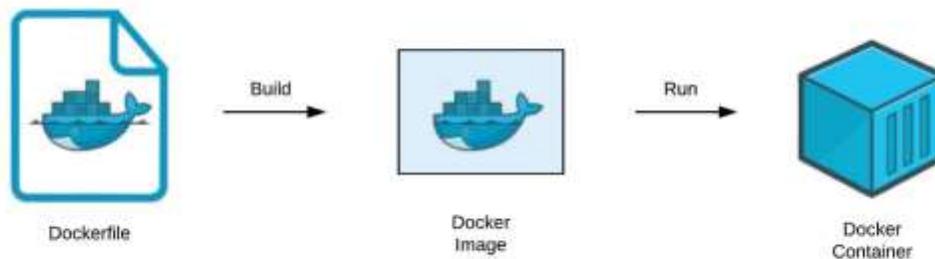


Ilustración 23: Dockerfile

A continuación, veremos las expresiones más importantes que se utilizan generalmente dentro de un archivo dockerfile, también mostraremos ejemplos concretos con cada una de ellas (González, 2017).



FROM

Esta expresión especifica la imagen base del contenedor y esta tiene que ser la primera línea del fichero de texto Dockerfile (aunque en ocasiones puede aparecer varias veces para crear varias imágenes) (González, 2017) Sus posibles sintaxis son las siguientes:

- FROM [imagen]: indica la última versión de la imagen.
- FROM [imagen:tag]: esta indica una versión específica de la imagen.

Ejemplo:

- FROM ubuntu:22.04

MAINTAINER

Este especifica al creador de la imagen, ya sea nombre o por correo, su sintaxis es:

- MAINTAINER <name>

Ejemplo:

- MAINTAINER Jason Flores jasonflores@gmail.com

RUN

Esta expresión ejecuta comandos sobre la imagen especificada para poder generar la definitiva deseada (González, 2017) su sintaxis es:

- RUN <comando>

Ejemplo:

- RUN apt-get install -y apache2

CMD

La expresión CMD, sirve para proporcionar un comando por defecto al momento de crear un contenedor basado en esa imagen, esta expresión solo se puede estar



especificada una vez dentro de todo el fichero Dockerfile (González, 2017) su sintaxis en la siguiente:

- CMD [<comando><parametro1, parametro2>...]

Ejemplo:

- CMD ["cat", "/etc/debian_version"]

EXPOSE

La expresión EXPOSE se utiliza solo para especificar los puertos que estarán a la escucha dentro del contenedor, basado en la imagen. En ningún caso se expondrá el puerto automáticamente (González, 2017), su sintaxis es la siguiente:

- EXPOSE <puerto1, puerto2>...

Ejemplo:

- EXPOSE 80 443

ENTRYPOINT

Por defecto, las imágenes son creadas para que todos los comandos indicados se ejecuten en /bin/bash. A través de la instrucción de ENTRYPOINT se pueden cambiar el comportamiento por defecto (González, 2017) cuya sintaxis es la siguiente:

- ENTRYPOINT <comando>

Ejemplo:

- ENTRYPOINT ["cat"]



VOLUME

VOLUME crea un punto de montaje el cual podrá ser accesible por otros contenedores o poder enlazar un directorio de servidor al contenedor en la imagen (González, 2017) la sintaxis es la siguiente:

- VOLUME <directorio1><directorio2>

Ejemplo:

- VOLUME /var/tmp

ENV

Dicha instrucción configura las variables de entorno, estos valores estarán en los entornos de todos los comandos que sigan en el dockerfile, su sintaxis la cual, se puede hacer de 2 formas es la siguiente:

- ENV <key><valor>, variable única a un valor
- ENV <key><valor> ..., Múltiples variables a un valor

Ejemplo:

- ENV myName=Jason Flores

USER

Al usar esta expresión, se puede especificar un usuario, puede usar el nombre de usuario, UID (identificador de usuario) o GID (identificador de grupo), o una combinación de estos 2, su sintaxis sería la siguiente:

- USER <user>

Ejemplo:

- USER Ubuntu



WORKDIR

La instrucción WORKDIR establece el directorio de trabajo para cualquier comando, ya sea RUN, CMD, ENTRYPOINT y demás instrucciones que le siguen en el archivo de dockerfile. Si el WORKDIR no existe, se creará, aunque no se utilice en ninguna subsecuente instrucción dockerfile. Su sintaxis sería la siguiente:

- WORKDIR <ruta>

Ejemplo:

- WORKDIR /path/to/workdir

Creación de una imagen

En el siguiente apartado se muestra el proceso de creación de una imagen con dockerfile, la imagen a crear será un servidor web apache usando como base un sistema Linux Debian (bastante liviano):

- **Crear un directorio con el nombre de la imagen**

Use el comando mkdir [nombre-de-la-carpeta]

- **Crear un archivo de texto llamado “dockerfile” dentro de la carpeta creada**

Para ello usa el comando “touch” o “nano” seguido del nombre del archivo “dockerfile”

- **Agregar lo siguiente en el archivo**



```
#imagen base que se va a utilizar
FROM debian:stretch-slim
#se indica el creador de la imagen y su respectivo contacto
MAINTAINER Jason Flores jasonflores@gmail.com
#Programas preinstalados que incluye el apache para el
funcionamiento del contenedor
RUN apt-get update && apt-get install -y apache2 && apt-get
clean && rm -rf /var/lib/apt/lists"
#Variables de entorno para el buen funcionamiento del servicio
apache2
ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2
#Puerto expuesto
EXPOSE 80
#Comando a ejecutar para que apache2 funcione de forma
correcta
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```

➤ Ejecutar el archivo dockerfile

Use el comando **docker build -t** [nombre-autor/nombre-imagen:etiqueta].

Proceso de creación de la imagen de apache cuyo nombre establecido es "debapach:1".

```
jason@Inspiron-3542:~/Documentos$ docker build -t debapach:1 .
[+] Building 0.5s (6/6) FINISHED
=> [internal] load build definition from dockerfile                                0.0s
=> => transferring dockerfile: 685B                                             0.0s
=> [internal] load .dockerignore                                                0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/debian:stretch-slim          0.4s
=> [1/2] FROM docker.io/library/debian:stretch-slim@sha256:abaa313c7e1dfe16069a1a42fa25401478 0.0s
=> CACHED [2/2] RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm -rf / 0.0s
=> exporting to image                                                            0.0s
=> => exporting layers                                                            0.0s
=> => writing image sha256:6bf9de781372d8f1cc32fc879d326ac02488de9beb0bd21b6e28490be761b5d8 0.0s
=> => naming to docker.io/library/debapach:1                                    0.0s
```

Ilustración 24: Creación de una imagen con docker build

Al inspeccionar las imágenes en el sistema se observa que ahora aparece una imagen llamada **debapach:1**



```
jason@Inspiron-3542:~/Documentos$ docker images
REPOSITORY      TAG          IMAGE ID      CREATED        SIZE
debapach        1           6bf9de781372 3 minutes ago 155MB
alpine          latest      b28839c304e2 5 weeks ago   7.85MB
ubuntu          latest      a6c760dce44b 5 weeks ago   80.3MB
nginx           latest      3f8a00f137a0 6 weeks ago   142MB
hello-world     latest      feb5d9fea6a5 18 months ago 13.3kB
```

Ilustración 25: Demostración de la imagen creada

Docker Compose



Ilustración 26: Docker compose

Definición

Según Raj Pethuru (2015) Docker compose es una herramienta sencilla pero poderosa, diseñada para simplificar el funcionamiento de un conjunto de contenedores Docker, o dicho también de otra manera, esta es una herramienta de orquestación que permite definir y controlar un servicio de múltiples contenedores.

También, permite la creación de un entorno de desarrollo rápido y aislado, así como también orquestar varios contenedores Docker ya que utiliza el motor de Docker para extraer y construir imágenes, iniciar los contenedores en la secuencia correcta y poder realizar la conectividad/vinculación correcta entre los contenedores/servicios según la configuración establecida en el archivo *docker-compose.yml*



Características que posee Docker-compose

Entre las principales características que presenta este componente son:

- Desplegar aplicaciones de diferentes entornos aislados entre ellos.
- Utilizar volúmenes de forma persistente, aunque se actualiza la plantilla para utilizar nuevas versiones de contenedores.
- Solo se recrean los contenedores modificados: reutiliza los contenedores en caso de cambios en la plantilla.
- Variables que serán utilizadas dentro de los contenedores para su comunicación: como, por ejemplo, el nombre de una base de datos. (González, 2017)

Contenido de Docker-compose

Esta plantilla contendrá:

- Lista de imágenes a utilizar para la ejecución de contenedores.
- La ruta a los Dockerfile que crearan las imágenes previamente especificadas (de ser necesario).
- Los puertos a exponer para poder acceder a dicho contenedor.
- Los respectivos volúmenes a utilizar.
- Las variables necesarias para la ejecución de aplicaciones. (González, 2017)

Estructura de un archivo docker-compose.yml

Según Zepeda (2020) un archivo de docker-compose posee una extensión y un formato yml, basta con crearlo y agregar contenido dentro de dicho archivo para que pueda ser utilizado. Además, afirma que la sintaxis de estos es muy fácil de comprender por lo que a continuación se abordará la estructura y las sentencias más comunes que contienen estos archivos:



```
GNU nano 6.2
version: '3.8'
services:
  nombre_del_servicio:
    variable_de_configuracion:
      valores
    variable_de_configuracion:
      valores
  nombre_de_otro_servicio:
nombre_de_otro_servicio:
  varibale_de_configuracion:
    valores
```

Ilustración 27: Estructura de Docker-compose.yml

Siempre el inicio del contenido del archivo se debe especificar la versión de Docker-compose a utilizar, seguidamente se encuentra la sección denominada “services”, en la cual se definen los servicios que se van a utilizar, cada uno de estos servicios contara con sus propias variables de entorno con su correspondiente valor.

Nombres de servicios

El nombre definido en cada uno de los servicios será utilizado como referencia para su utilización en otros servicios

Como, por ejemplo, si un servicio se hace llamar “db”, este es el nombre que debe utilizar en otras aplicaciones para poder referirnos a un host o ubicación.



```
DATABASES = {  
  'default': {  
    # ...  
    'HOST': 'db',  
    # ...  
  }  
}
```

Ilustración 28: Nombre de servicios en docker-compose.yml

Opciones de configuración de docker compose

Las variables de entorno son las encargadas de indicarle a cada servicio como debe comportarse.

Image:

Establece la imagen que será utilizada para poder generar el servicio.

```
GNU nano 6.2  
version: '3.8'  
services:  
  
  db:  
  
    image: postgres
```

Ilustración 29: Opción imagen en docker-compose.yml

Build

Establece la ubicación donde se encuentra el archivo Dockerfile en caso de que utilice una imagen personalizada.



```
GNU nano 6.2
version: '3.8'
services:

  webapp:

    build: ./ubicacion_dockerfile
```

Ilustración 30: Opción build en docker-compose.yml

Context y dockerfile

“Context” establece la ubicación de un archivo Dockerfile personalizado que será utilizado en lugar del predeterminado y dockerfile establece el nombre de dicho archivo.

```
GNU nano 6.2
version: '3.8'
services:

  webapp:

    build:

      context: ./ubicacion_dockerfile

      dockerfile: dockerfile_personalizado
```

Ilustración 31: Opciones context y dockerfile en docker-compose.yml

Command

Se utiliza para reemplazar el comando predeterminado del contenedor, es muy útil para ejecutar un comando cuando se inicia un servicio.

```
version: '3.8'
services:

  web:

    build: .

    command: python manage.py runserver 0.0.0.0:8000
```

Ilustración 32: Opción command en docker-compose.yml



Ports

Indican los puertos que serán expuestos al exterior y el puerto del computador con cual se vincularán, utiliza el formato **HOST: CONTENEDOR**.

Opcionalmente puede especificar el protocolo **UDP** o **TCP**.

```
version: '3.8'
services:
  web:
    build: .
    command: python manage.py runserver 0.0.0.0:8000
    ports:
      - "80:8000"
```

Ilustración 33: Opción ports en docker-compose.yml

En el ejemplo anterior se indica que el puerto 80 de la maquina física estará vinculado con el puerto 8000 del contenedor.

Expose

Al igual que ports también se puede utilizar para la exposición de puertos, pero la principal diferencia con este es que los puertos expuestos con expose estarán disponibles solamente para los servicios vinculados no para la maquina física donde estamos ejecutando docker.

```
version: '3.8'
services:
  redis:
    image: redis
    expose:
      - '6379'
```

Ilustración 34: Opción expose en docker-compose.yml



Depends_on

Esta opción permite que uno de los servicios se ejecute únicamente después de otro, en otras palabras, esta opción hace posible que el arranque de un servicio dependa de la ejecución de otros.

```
version: '3.8'
services:

  web:

    build: .

    command: python manage.py runserver 0.0.0.0:8000

    depends_on:

      - db

      - redis
```

Ilustración 35: Opción depends_on en docker-compose.yml

En el ejemplo anterior se ejecutará el servicio web solo si ya están disponibles los servicios de db y redis.

Environment

Permite establecer una serie de variables de entorno que están disponibles en el servicio que se desea ejecutar.

```
version: '3.8'
services:

  db:

    image: postgres

    environment:

      - POSTGRES_USER=usuario

      - POSTGRES_PASSWORD=contrasenia
```

Ilustración 36: Opción environment en docker-compose.yml



Env_file

Permite establecer los valores a múltiples variables de entorno en un solo archivo en lugar de especificarlas una por una.

```
version: '3.8'
services:
  web:
    build: .
    command: python manage.py runserver 0.0.0.0:8000
    env_file: common.env
```

Ilustración 37: Opción env_file en docker-compose.yml

Volumes

Permite montar archivos o directorios ubicados en el sistema operativo de la máquina física en un contenedor o servicio de Docker, también nos permite guardar datos en un contenedor de manera persistente en una ubicación del sistema operativo de la máquina física, su sintaxis es **HOST: CONTENEDOR**, en donde el Host puede ser la ubicación en el S.O (sistema operativo) de la máquina física o el nombre de un volumen creado con Docker.

De manera opcional puedes especificar si los volúmenes serán solamente de lectura “ro” o de lectura y escritura “rw”.

```
version: '3.8'
services:
  db:
    image: postgres:latest
    volumes:
      - "/var/run/postgres/postgres.sock:/var/run/postgres/postgres.sock"
      - "dbdata:/var/lib/postgresql/data:ro"
```

Ilustración 38: Opción volumes en docker-compose.yml



Restart

Esta opción permite aplicar políticas de reinicio a los servicios en Docker, se podrían establecer los siguientes valores a esta opción:

no: no reiniciar el contenedor.

always: reiniciar siempre el contenedor.

on-failure: reiniciar el contenedor si este devuelve un estado de error.

unless: reiniciar el contenedor en todos los casos a excepción de cuando se detiene.

```
version: '3.8'
services:

  db:

    image: postgres:latest

    restart: "on-failure"
```

Ilustración 39: Opción restart en docker-compose.yml

Comandos básicos para Docker Compose

Después de crear un archivo docker-compose.yml, es necesarios ejecutar determinados comandos para que funcione.

- **Docker-compose up**

Este comando nos ayuda a la construcción de la imagen, tras lo cual creara y lanzara los contenedores Docker. Los contenedores provienen de los servicios especificados en el archivo de configuración. Si los contenedores ya se están ejecutando y luego se ejecuta docker-compose up, el contenedor se creará de nuevo

- **Docker-compose start**



Este comando de Docker Compose inicia los contenedores de Docker, pero no crea imágenes ni tampoco crea contenedores; solo funciona para iniciar los contenedores creados previamente

- **Docker-compose stop**

A menudo es necesario detener los contenedores tras haberlos creado e inicia. Aquí es donde el comando de apagado de Docker Compose resulta muy útil. Básicamente este comando detiene los servicios que se están ejecutando, pero los contenedores de instalación y las redes permanecen intactos.

- **Docker-compose down**

Este comando también permite la detención de los contenedores de Docker al igual que el caso del comando <<stop>>, pero no solo se encarga de detener los contenedores, sino que también los elimina. Esto también se aplica a las redes, los volúmenes y las imágenes de Docker que se puede eliminar si se utilizan ciertos argumentos

- **Docker-compose down –volumes**

Elimina todos los volúmenes

- **Docker-compose down –rmi all**

Elimina todas las imágenes. (cobertti, 2021)

Almacenamiento de Docker

De forma predeterminada, todos los archivos creados dentro de un contenedor se almacenan en una capa de contenedor de escritura. Esto significa que:

- Los datos no persisten cuando ese contenedor ya no existe y puede ser difícil sacar los datos del contenedor si otro proceso los necesita.



- La capa de escritura de un contenedor está estrechamente acoplada a la maquina host donde se ejecuta el contenedor. No puede mover fácilmente los datos a otro lugar.
- Escribir en la capa de escritura de un contenedor se requiere un controlador de almacenamiento para administrar el sistema de archivos. El controlador de almacenamiento proporciona un sistema de archivos de unión. Utilizando el Kernel de Linux. Esta abstracción adicional reduce el rendimiento en comparación con el uso de volúmenes de datos, que escriben directamente en el sistema de archivos del host.

Docker tiene 2 opciones para que los contenedores almacenen archivos en la maquina host, de modo que los archivos persistan incluso después de que el contenedor se detenga: volúmenes y bind mounts.

Docker también admite a los contenedores que almacenan archivos en memoria en la maquina host. Dichos archivos no se conservan. Si se está ejecutando Docker en Linux, el montaje *tmpfs* (*sistema de almacenamiento típico de sistemas UNIX y similares, como **BSD** o **GNU Linux**. Se trata de un sistema de ficheros montado que utiliza memoria volátil. Por lo que los datos que pueda contener se pierden al reiniciar el equipo*) se usa para almacenar archivos en la memoria del sistema del host.

Elegir el tipo de montaje adecuado

Independientemente del tipo de montaje que elija usar, los datos se ven iguales desde dentro del contenedor. Se expone como un directorio o como un archivo individual en el sistema de archivos del contenedor.

Una manera fácil de visualizar la diferencia ente volúmenes, bind mounts y montajes tmps es pensar en donde residen los datos en el host de Docker.

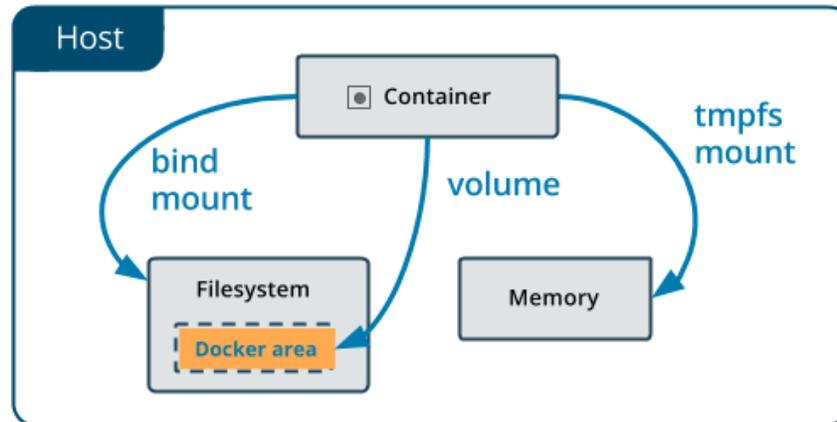


Ilustración 40: Tipos de montaje en Docker

Los volúmenes se almacenan en una parte del sistema de archivos del host que es administrado por Docker (`/var/lib/docker/volumenes/` en Linux). Los procesos que no son de Docker no deben modificar esta parte del sistema de archivos. Los volúmenes son la mejor manera de conservar los datos en Docker.

Los bind mount se pueden almacenar en cualquier parte del sistema host. Incluso pueden ser archivos o directorios importantes del sistema. Los procesos que no son de Docker en el host de Docker o en un contenedor de Docker pueden modificarlos en cualquier momento.

Los montajes tmpfs se almacenan últimamente en la memoria del sistema host y nunca se escriben en el sistema de archivos del sistema host.

Más detalles sobre los tipos de montaje

Volúmenes: creado y administrado por Docker. Puede crear un volumen explícitamente usando el comando `docker volume create`. O Docker puede crear un volumen durante la creación del contenedor o servicio.

Cuando se crea un volumen, se almacena dentro de un directorio en el host de Docker cuando monta el volumen en un contenedor, este directorio es lo que se monta en el contenedor. Esto es similar a la forma en que funcionan los bind mounts



excepto que los volúmenes son administrados por Docker y están aislados de la funcionalidad principal de la maquina host.

Es posible montar un volumen dado en varios contenedores simultáneamente. Cuando ningún contenedor que se encuentra en ejecución usa un volumen, este volumen aún está disponible en Docker y no se elimina automáticamente. Puede eliminar los volúmenes no utilizados mediante *docker volume prune*.

Bind mounts: Disponible desde los primeros de Docker. Los bind mounts (o montajes de enlace) tienen una funcionalidad limitada en comparación con los volúmenes. Cuando se utiliza un montaje de enlace un archivo o directorio en la maquina host se monta en un contenedor. Se hace referencia al archivo o directorio mediante su ruta completa en la máquina host. No es necesario que el archivo o directorio ya exista en el host de Docker. Se crea bajo demanda si aún no existe. Los bind mounts son muy eficaces, pero estos dependen de que el sistema de archivos de la maquina host tengan una estructura de directorio especifica disponible. Si se está desarrollando nuevas aplicaciones de Docker, se debe considerar usar volúmenes con nombre en su lugar. No se pueden usar los comandos de la CLI de Docker para administrar directamente los montajes de enlace o *bind mounts*.

Montaje tmpfs: un montaje tmpfs no se conserva en el disco, ya sea en el host de Docker o dentro de un contenedor. Puede ser utilizado por un contenedor durante la vida útil del contenedor, para almacenar estado no persistente o información confidencial.

Los bind mounts y los volúmenes se pueden montar en contenedores con el indicador **-vo --volume**, pero la sintaxis de cada uno es ligeramente diferente. Para tmpfs las monturas pueden usar la bandera **-tmpfs**. Se puede usar la marca **-mount** tanto para contenedores como para servicios, para bind mounts, volúmenes o tmpfs montajes, ya que esta sintaxis es mucho más clara.



Casos de uso para:

A) Volúmenes:

Estos son la forma preferida de conservar los datos en los contenedores y servicios de Docker. Algunos casos de uso para volúmenes incluyen:

- Compartir datos entre varios contenedores en ejecución. Si no lo crea explícitamente, se crea un volumen la primera vez que se monta en un contenedor. Cuando ese contenedor se detiene o se retira, el volumen aún existe. Varios contenedores pueden montar el mismo volumen de manera simultánea, esto ya sea de lectura y escritura o de solo lectura. Los volúmenes solo se eliminarán cuando los eliminas explícitamente
- Cuando no se garantiza que el host de Docker tenga un directorio o una estructura de archivos determinados. Los volúmenes lo ayudan a desacoplar la configuración del host de Docker del tiempo de ejecución del contenedor.
- Cuando se necesite realizar una copia de seguridad, o restaurar, o migrar datos de un host Docker a otro, los volúmenes son una mejor opción. Puede detener los contenedores que usan el volumen y luego hacer una copia de seguridad del directorio del volumen (como `/var/lib/docker/volumes/<volume-name>`)
- Cuando una aplicación requiere E/S de alto rendimiento en Docker Desktop, los volúmenes se almacenan en la VM de Linux en lugar del host, lo que significa que las lecturas y escrituras tendrán una latencia mucha más baja y un rendimiento mucho más alto



B) Bind mount

De manera generalizada se debe utilizar volúmenes siempre que sea posible, pero los bind mount son útiles para lo siguiente:

- Compartir archivos de configuración desde la maquina host a los contenedores. Así es como Docker proporciona la resolución de DNS a los contenedores de forma predeterminada, montándolo en `/etc/resolv.conf` desde la máquina host en cada contenedor
- Cuando se garantiza que la estructura de archivos o directorios del host de Docker es coherente con los bind mounts que requieren los contenedores.

C) Tmpfs

Los montajes tmpfs se utilizan para los casos en los que no desea que los datos persistan en la maquina host o dentro del contenedor. Esto puede deberse a motivos de seguridad o para proteger el rendimiento del contenedor cuando una aplicación necesita escribir un gran volumen de datos de estado no persistente.

Redes Docker

Controladores de red

El subsistema de red de Docker es conectable y utiliza controladores. Existen varios controladores que hay de forma predeterminada y que nos proporcionan la funcionalidad de red básica.

Bridge: Este controlador de red es el predeterminado, si no se especifica un controlador, este es el tipo de red que se está creando. Las redes bridge generalmente son usadas cuando las aplicaciones se van a ejecutar en contenedores independientes que necesitan comunicarse.



Host: Para contenedores independientes, quita el aislamiento de red entre contenedor y el host de Docker, y use la red del host directamente.

Overlay: Las redes overlay conectan varios demonios de Docker y permiten que los servicios de enjambre se comuniquen entre sí. También puedes usar el controlador overlay para facilitar la comunicación entre el servicio de enjambre y un contenedor independiente o entre los contenedores independientes en diferentes demonios Docker. Esta estrategia elimina la necesidad de realizar enrutamiento a nivel de sistema operativo entre los contenedores.

ipvlan: es las redes brindan a los usuarios un control total sobre las direcciones IPv4 e IPv6.

macvlan: Estas redes permite a los usuarios asignar una dirección MAC a un contenedor haciéndole aparecer como un dispositivo físico en su red. El demonio de Docker enruta el tráfico a los contenedores por sus direcciones MAC.

None: deshabilita todas las redes. Por lo general, se usa junto con un controlador de red personalizado. None no está disponible para los servicios de enjambre.

Complementos de red: Puede instalar y usar complementos de red de terceros con Docker. Estos complementos están disponibles en Docker Hub o en proveedores externos. Siempre se debe consultar la documentación del proveedor para instalar y usar un complemento de red determinado.

Escenarios de uso para controladores de red

- Las redes bridge definidas por el usuario son mejores cuando necesita varios contenedores para comunicarse en el mismo host de Docker.

- Las redes host son mejores cuando la pila de red no debe estar aislada del host de Docker, pero desea aislar otros aspectos del contenedor.



- Las redes overlay son mejores en los casos que se necesite contenedores que se ejecutan en diferentes hosts de Docker para comunicarse, o cuando hay varias aplicaciones trabajando juntas usando servicios de enjambre.
- Las redes Macvlan son mejores cuando se está migrando desde una configuración de VM o necesita que sus contenedores se vean como hosts físicos en su red, cada uno con una dirección MAC única.
- Los complementos de red de terceros permiten integrar Docker con pilas de red especializadas. (Networking overview, s.f.)

Docker Hub

Definición

Según (Villacampa, 2021) Docker Hub es un repositorio en línea basado en la nube el cual fue diseñado para la distribución de imágenes, ofrece una gran cantidad de estas de manera gratuita las cuales pueden ser descargadas y utilizadas ahorrando así el trabajo de hacer todo desde cero ya que se pueden aprovechar estas plantillas

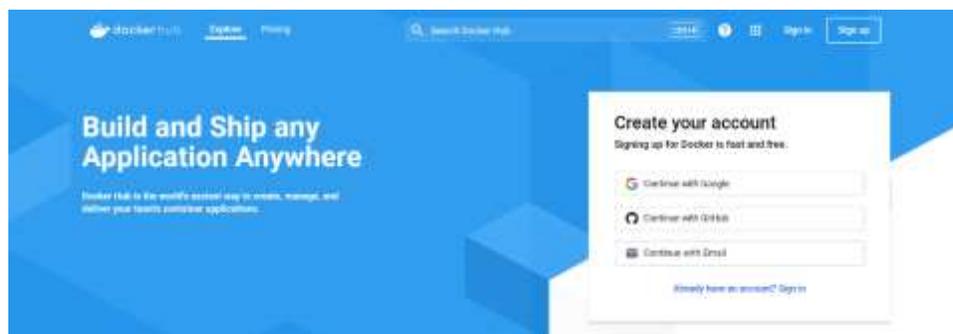


Ilustración 40: Docker Hub



Características de Docker Hub

Las principales características con las que cuenta Docker Hub que destaca (Ali, 2022) son las siguientes:

- **Repositorios de imágenes**
Permite encontrar y descargar imágenes de contenedores, así como subir imágenes propias a repositorios ya sean públicos o privados.
- **Equipos y organizaciones**
Permite crear equipos de trabajo y administrar el acceso a repositorios privados de imágenes de contenedores de manera que estos sean usados solamente por la organización.
- **Integración de GitHub**
Permite la integración con repositorios de código fuente como GitHub y BitBucket.
- **Construcciones automatizadas**
Si se le envió alguna modificación en el código fuente a los repositorios de código fuente, crea automáticamente imágenes de contenedores desde GitHub o BitBucket y las envía a Docker Hub.
- **Webhooks**
Permite realizar acciones luego del envío exitoso de una imagen a un repositorio para integrar Docker Hub con otros servicios.
- **Imágenes oficiales y del editor**
Nos permite descargar y usar imágenes oficiales de contenedores las cuales son de alta calidad y son proporcionadas por proveedores externos, son imágenes de editores o también llamadas imágenes certificadas, las cuales nos brindan soporte y garantía de compatibilidad con Docker Enterprise.



Comandos básicos para trabajar con Docker Hub

Inicio de sesión en Docker Hub

Para iniciar sesión en Docker Hub utilice el siguiente comando:

- **Docker login**

Ejemplo:

```
(user@kali)-[~]
└─$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID,
head over to https://hub.docker.com to create one.
Username: jasonflores23
Password:
WARNING! Your password will be stored unencrypted in /home/user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded

(user@kali)-[~]
└─$
```

Ilustración 41: Login de Docker hub

El cual luego nos solicitara nuestro nombre de usuario más nuestra contraseña.

Cerrar sesión en Docker Hub

Para cerrar nuestra sesión en Docker Hub se utiliza el siguiente comando:

- **Docker logout**

Ejemplo:

```
user@kali ~
File Actions Edit View Help

(user@kali)-[~]
└─$ docker logout
Removing login credentials for https://index.docker.io/v1/

(user@kali)-[~]
└─$
```

Ilustración 42: Logout de Docker hub

Buscar imágenes

Para buscar una imagen en el Docker Hub se utiliza el siguiente comando:

- **Docker search <nombre de la imagen a buscar>**

-



Ejemplo:

En este caso buscaremos las imagenes disponibles para Wordpress.

```
user@kali:~$ docker search wordpress
```

NAME	DESCRIPTION	STARS	OFFICIAL
wordpress	The WordPress rich content management system.	5468	[OK]
bitnami/wordpress	Bitnami container image for WordPress	247	
bitnami/wordpress-nginx	Bitnami Docker Image for WordPress with NGINX	84	
rapidfort/wordpress	Rapidfort optimized, hardened image for word...	19	
wordpressdevelop/php	PHP images for the WordPress local develop...	13	
wordpressdevelop/cli	WP-CLI images for the WordPress local devel...	8	
wodby/wordpress-php	PHP for WordPress	6	
wordpressdevelop/phpunit	PHPUnit images for the WordPress local devel...	6	
wodby/wordpress-nginx	Nginx for WordPress	6	
wodby/wordpress	Vanilla WordPress container image	2	
xulhub/wordpress-4.9-rc1	wordpress 4.9 任意命令执行漏洞 (PHPMailer)	2	
xulhub/wordpress		1	
stleky/wordpresswithlogins	A wordpress extension of original wordpress ...	1	
wordpresscharts/wordpress	The WordPress Kubernetes Image	1	
betterweb/wordpress	Docker wordpress with Redis	0	

Ilustración 43: Buscar imágenes en Docker hub

Descargar imágenes

Para poder descargar una imagen se utiliza el siguiente comando:

- **Docker pull <nombre de la imagen a descargar>**

Ejemplo:

En este caso descargaremos una imagen de wordpress.

```
user@kali:~$ docker pull wordpress
```

```
Using default tag: latest
```

```
latest: Pulling from library/wordpress
```

```
a1b7e978171: Pull complete
```

```
6a88e6e0142: Pull complete
```

```
95f5176ace8b: Pull complete
```

```
8a6e7e1324ca: Pull complete
```

```
073e01769ec9: Pull complete
```

```
7af8c58b3897: Pull complete
```

```
1a19a72e0529: Pull complete
```

```
224daec82f96: Pull complete
```

```
9b0227b1e274: Pull complete
```

```
b09d50f09f8: Pull complete
```

```
991e8552ac4: Pull complete
```

```
118e3b879f1: Pull complete
```

```
94fbc73d449: Pull complete
```

```
0918769e91bd: Pull complete
```

```
2a1351a2f941: Pull complete
```

```
080bc185f188: Pull complete
```

```
8ad29e7aa19: Pull complete
```

```
e95473a7f69c: Pull complete
```

```
b7d81c854f88: Pull complete
```

```
a678c25c08f6: Pull complete
```

```
dc9ce32006dc: Pull complete
```

```
Digest: sha256:20c31ae724831f18089be7943e957ce66d0e0d0d171c2b185cea9740e221
```

```
Status: Downloaded newer image for wordpress:latest
```

```
docker.io/library/wordpress:latest
```

Ilustración 44: Descargar imágenes de Docker hub



Cargar imágenes

Para cargar una imagen utilice el siguiente comando:

- **Docker push <usuario/nombre de la imagen>**

Ejemplo:

Ahora vamos a crear un tag a la imagen para indicar la versión que subiremos al repositorio, para ello seguimos el siguiente ejemplo donde tenemos en cuenta la nomenclatura para crear la imagen. En este caso tomaremos una imagen de Wordpress que se encuentra en nuestro repositorio de imágenes.

```
File Actions Edit View Help
[user@kali ~]
└─ docker tag wordpress jasonflores23/wordpress:v1

[user@kali ~]
└─ docker images

REPOSITORY          TAG          IMAGE ID       CREATED        SIZE
jasonflores23/wordpress  v1          9071407ed1c0  5 weeks ago   740MB
wordpress           latest      9071407ed1c0  5 weeks ago   740MB
bitnami/mariadb     10.6        c05eef404e28  3 months ago  136MB
bitnami/mariadb     latest      93295323bf3a  4 months ago  177MB
nginx               latest      f5a60296b8a2  4 months ago  187MB
bitnami/noodle      latest      022a90f66afd  4 months ago  641MB
mariadb             10         580f8d036e3c  4 months ago  403MB
wordpress           <none>     efa186f6484e  4 months ago  666MB
mariadb             10.4       afhaf7b1e72c  4 months ago  385MB
mysql               8          99afc800f15b  5 months ago  577MB
mysql               latest     99afc800f15b  5 months ago  577MB
prestashop/prestashop latest     56ec03fd8eb1  5 months ago  1.42GB
mysql               5.7       92034fe9a43f  5 months ago  581MB
phpmyadmin/phpmyadmin latest     933569f3a9f6  5 months ago  562MB
bitnami/noodle      4          6e224a48cf1  10 months ago  610MB
bitnami/noodle      3          77528c7e710b  11 months ago  610MB
wonderos/php-apache 7.3       8ff972b2d052  13 months ago  1.12GB
redis               5.0-alpine 7530bc5e0ea2  14 months ago  22.9MB
seru/noodle-trun    php74-fpm 51ba437b0e4  15 months ago  949MB
seru/noodle         php74-fpm 6f4b8372ff3a  15 months ago  949MB
mariadb             10.7       025e9a493bb0  17 months ago  130MB
lithub/noodle       latest     3da31a5d4bc  2 years ago   934MB
bitnami/mariadb     10         212f927a0dca  6 years ago   590MB
joomla              3.6.2-apache d4909493022b 7 years ago   429MB
```

Ilustración 45: Insertar tag a una imagen en Docker hub

Luego solo haría falta realizar el comando push.

NOTA: Para poder compartir o subir la imagen usando el Docker push se debe de haber iniciado sesión previamente utilizando el comando Docker login.



```
user@kali: ~$ docker push jasonflores23/wordpress:v1
The push refers to repository [docker.io/jasonflores23/wordpress]
343629feca26: Mounted from library/wordpress
8c7526a65d8c: Mounted from library/wordpress
777ea2592782: Mounted from library/wordpress
ad0e9cdf7759: Mounted from library/wordpress
daa03278265f: Mounted from library/wordpress
5c2913c39384: Mounted from library/wordpress
da53f630613f: Mounted from library/wordpress
bd7003052d10: Mounted from library/wordpress
55bc50001d55: Mounted from library/wordpress
c09c3a334b26: Mounted from library/wordpress
32e314cd52b8: Mounted from library/wordpress
92b404f1327b: Mounted from library/wordpress
eF0d6787d529: Mounted from library/wordpress
015536319b33: Mounted from library/wordpress
a74fbc9c7b18: Mounted from library/wordpress
5dd2e9e27d29: Mounted from library/wordpress
20ba537eb762: Mounted from library/wordpress
38a6e0f337b9: Mounted from library/wordpress
a84f293bd704: Mounted from library/wordpress
e51e590b299b: Mounted from library/wordpress
7292c778baa8: Mounted from library/wordpress
v1: digest: sha256:9b9774b0994c165e0aeb3bc4889abb49313a7b908cccd323f6ee62bccedaa9ca size: 4711
```

Ilustración 46: Cargar una imagen en Docker hub

Una vez cargada la imagen puede verificar que la misma se encuentra entre sus repositorios de imágenes dentro de su perfil de Docker Hub.

Content Management System (CMS)

WordPress

WordPress es una plataforma de gestión de contenidos, representa un software de código abierto altamente versátil que permite a los usuarios crear y desarrollar sitios web personalizados de manera sencilla, adaptable y profesional, con una muy buena experiencia de usuario (WordPress, 2023).

Posee una amplia gama de características y flexibilidad, WordPress con el tiempo se ha convertido en la elección preferida tanto para principiantes que buscan establecer una presencia en línea, como para profesionales que necesitan una plataforma potente para construir sitios web dinámicos y atractivos.

Aprovechando su naturaleza intuitiva, la comunidad activa de desarrolladores, el sólido soporte de plugins y temas personalizables que posee lo hacen destacar como la opción líder para impulsar la visibilidad en línea de cualquier negocio o proyecto, garantizando una experiencia optimizada tanto para los visitantes del sitio como para los administradores del mismo.



PrestaShop

Prestashop es un CMS diseñado especialmente para la creación y gestión de tiendas online. Se trata de una plataforma gratuita orientada a pequeñas y grandes empresas. Su aparición no hizo más que revolucionar el mundo del ecommerce. Con el paso del tiempo, cada vez fueron más los negocios que se fueron sumando a esta fiebre. Hoy día, ya son más de 250.000 tiendas online creadas a partir de Prestashop. Estamos ante un software gratuito, de código abierto y con más de 310 funcionalidades que siguen ampliándose día a día. Su funcionamiento es bastante sencillo, por lo cual se debe su gran éxito (Prestashop, 2018).

El propósito de esta herramienta es ofrecerles a sus usuarios un entorno similar al que presenta WordPress, pero enfocado en el comercio minorista digital, desde la creación hasta el mantenimiento. Dado que los ecommerce son una opción para que las marcas que no tienen tiendas físicas puedan comercializar sus productos, profesionalizar esta forma de trabajar se vuelve cada vez más necesario.

Paralelamente a esto, las empresas que desean tener sistemas de ecommerce efectivos, sin necesidad de tanta complejidad, también buscan soluciones simples y accesibles. PrestaShop puede serle útil a estos dos públicos, gracias a su amplitud de recursos, su facilidad de operación y, principalmente, porque es gratuito.

Drupal

Drupal es una plataforma para construir sitios web flexibles, dinámicos y potentes. Es un sistema de gestión de contenidos de código abierto (es gratuito y no tiene licencia de uso) y podemos pensarlo como una tecnología que facilita la creación, actualización y gestión de contenidos de un sitio web.

El rango de sitios que se pueden hacer con Drupal es enorme ya que hablamos de una plataforma muy flexible para hacer cualquier tipo de desarrollo en función de los objetivos de una empresa u organización (Drupal, 2023).



Drupal permite publicar archivos, imágenes, artículos, al igual que crear y administrar todo tipo de contenidos como votaciones, encuestas, foros, entre otros. A pesar de no ser tan popular como WordPress, Drupal es uno de los CMS más completos para grandes portales corporativos, además se trata de una plataforma flexible y fácilmente integrable con otras soluciones de negocio.

Por otro lado, también es de las plataformas más flexibles, siendo ideal para usuarios con conocimientos avanzados, que los ayudará a crear proyectos complejos, generar una cantidad de tráfico elevada, procesar datos y desarrollar funcionalidades específicas.

Joomla

Joomla es un CMS gratuito y de código abierto más populares a nivel mundial que puedes emplear para crear y administrar tu sitio web, blog e incluso aplicaciones para móviles. Joomla permite gestionar tu web fácilmente. Esto significa que puedes cambiar las imágenes y el texto de tu sitio web siempre que lo desees y no necesitas ninguna experiencia en codificación ni conocimientos de informática para ello. Una de las características más importantes de Joomla es la posibilidad de personalizarlo a través de módulos y herramientas para adaptarlo a las necesidades de tu negocio (Joomla, 2024).

Joomla permite gestionar con mucha facilidad toda tu web, crear un nuevo apartado, modificar los actuales, añadir nuevas imágenes, crear nuevas opciones de menú y casi cualquier cosa que puedas necesitar la podrás hacer rápidamente y sin tener conocimientos técnicos, conociendo Word o algún editor de textos podrás manejar tu web.

Moodle

Moodle es una de las plataformas eLearning más utilizadas en el sector de la teleformación. Las posibilidades que ofrece al centro son bastante amplias y el



desarrollo de este espacio virtual de aprendizaje son tan grandes como la ambición del profesional. Esto, unido a su gratuidad hace de esta una opción muy atractiva para desarrollar todo tipo de acciones de aprendizaje online. Si queremos definir su aspecto técnico podríamos decir que es un LMS (learning management system) específicamente creado para crear entornos de aprendizaje online.

La plataforma Moodle es un sistema de enseñanza diseñado para crear y gestionar espacios de aprendizaje online adaptados a las necesidades de profesores, estudiantes y administradores. Es un sistema web dinámico creado para gestionar entornos de enseñanza virtual, basado en tecnología PHP y bases de datos MySQL.

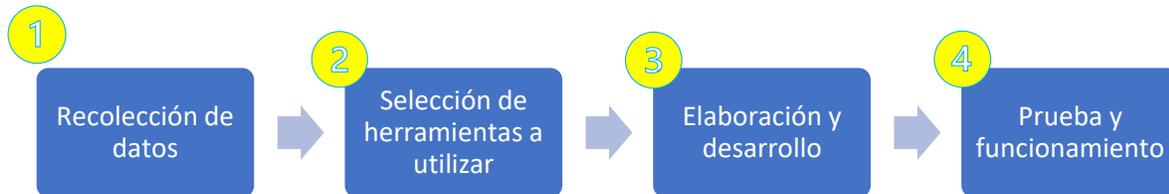
Las posibilidades que ofrece este LMS son tantas que muchas empresas, academias o escuelas se decantan por este software para subir el contenido de sus cursos. Es una plataforma fácil de utilizar para los profesores, no requiere una gran inversión para ponerla en funcionamiento y, además, el acceso y funcionamiento para los alumnos también es muy intuitivo (Herrera, 2021).



Diseño metodológico

En esta sección se abordará sobre el proceso de la realización del presente trabajo, mencionar sobre el tipo de investigación que se está llevando, también se menciona sobre la obtención de datos hasta la selección de las herramientas a utilizar y sobre las etapas del presente proyecto.

En el presente trabajo investigativo se lleva a cabo un tipo de investigación aplicada, ya que resultó un tema poco abordado y de gran interés para los estudiantes de la carrera de Ingeniería en sistemas del Departamento de Computación de la UNAN-León, además de ser un área poco explorada por los autores del presente trabajo.



Etapa de recolección de datos

Durante la primera etapa de la investigación se realizó una revisión de diversos documentos, sitios webs, canales de YouTube que abordan temas acerca de contenedores Docker, CMS (System Management Content), donde desde el punto practico se encontró un curso llamado “Labs – Docker for the Absolute Beginner – Hands On” el cual se encuentra a través de la plataforma KodeKloud que en 8 laboratorios enseñan temas importantes sobre contenedores Docker para principiante (desde comandos básicos hasta llegar a lo que son redes Docker), donde el ambiente de trabajo es muy completo debido a que la plataforma otorga el software necesario para ser resuelta dicha guía de trabajo.

Etapa de selección de herramientas a utilizar

Recursos hardware

Equipos hardware que se utilizaron para la realización del proyecto.



Ordenadores				
Equipos	CPU	RAM	Almacenamiento	S. O
Lenovo Ideapad 3	Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz	8.00 GB	256 SSD	Windows 10 Home
DELL Inspiron 3542	Intel® Core™ i3-4030U CPU @ 1.90GHz (4 CPUs)	8.00 GB	256 SSD	Ubuntu 22.04 LTS
HP EliteBook 840G1	Intel® Core™ i5-4300U CPU @ 1.90 GHz (4CPUs) ~2.5 GHz	8.00 GB	512 HDD	Ubuntu 22.04 LTS y Windows 10 education

Recursos Software

Herramientas software que se utilizaron para la realización del proyecto.

Herramientas Software	
Equipos	CPU
Ubuntu 22.04 LTS	Es un sistema operativo GNU/Linux de código abierto basado en Debian. Actualmente se ejecuta en ordenadores y servidores
Kali 2023.2	Distribución de Linux basada en Debian, diseñada para temas de seguridad, como análisis de redes, ataques inalámbricos, análisis forenses entre otros.
Windows 10	Es un sistema operativo para computadoras desarrollado por Microsoft.



Chrome, Edge y Firefox	Navegadores web, diseñados para navegar por internet de manera rápida y simple.
Terminal de Linux	Es un programa cuyo objetivo principal es leer comandos y ejecutar otros programas.
Docker versión nativa de Linux	Es un proyecto de código abierto que automatiza la implementación de aplicaciones dentro de los contenedores permitiendo empaquetar una aplicación con sus dependencias en un contenedor.

Etapas de elaboración y desarrollo

Las etapas del proceso de elaboración y desarrollo del presente trabajo fueron.

Organización del desarrollo

En esta etapa se ha trabajado en las guías "Labs - Docker for the Absolute Beginner - Hands On", las cuales se explicaron y se documentaron las soluciones de cada uno de los incisos de las diferentes guías.

Así mismo se facilitaron archivos docker-compose.yml para automatizar la construcción de los ejemplos de contenedores que pueden ser utilizados en los componentes seleccionados haciendo uso de las imágenes de Docker Hub seleccionadas en la etapa de recolección de datos.

Formato de guías "Labs- Docker for the Absolute Begginer - Hands On"

El formato para enunciar cada una de las guías es el siguiente:

**Título**

Indica el nombre de la guía

Introducción

Describe los aspectos generales del desarrollo del contenido de la guía

Duración estimada de la guía

Tiempo estimado para dar solución a la guía

Desarrollo de la guía

Comprender cada uno de los incisos que deben ser resueltos durante la guía.

Formato de la documentación de los contenedores

El formato para enunciar cada uno de los contenedores es el siguiente:

Título

Indica el nombre de la guía.

Descripción

Describe la información respecto a la temática del contenedor.

Requisitos software

Indica que recursos software se requieren para levantar el contenedor.

Requisitos hardware

Indica los recursos hardware se necesitan para ejecutar el contenedor.

Software y servicios incluidos en el contenedor

Indica el software y servicios que utiliza el contenedor para su correcto funcionamiento.

Proceso de creación

Indica el proceso total para la creación del contenedor, así mismo como su funcionamiento.



Etapas de prueba y funcionamiento

Las guías del curso “Labs - Docker for the Absolute Beginner – Hands On” se desarrollaron en entorno web de la plataforma kodeKloud, este provee las herramientas necesarias para la resolución de las guías planteadas.

Desarrollo

Desarrollo de las guías "Labs - Docker for the Absolute Beginner – Hands On" de la plataforma KodeKloud

Para la elaboración de estas guías se ha tomado como fuente de información el curso "Labs - Docker for the Absolute Beginner - Hands On" el cual pertenece a KodeKloud que es una plataforma de aprendizaje práctico que brinda capacitación de calidad en DevOps y tecnologías de automatización como Kubernetes, Docker, Ansible, OpenShift, Puppet, Chef y muchas más. (KodeKloud, 2022)

Este curso ha sido seleccionado porque se considera muy completo además de que brinda conocimientos concretos sobre Docker a un nivel básico, para la realización de estas guías el estudiante necesitará únicamente disponer de una computadora con un navegador web instalado y acceso a Internet para poder entrar en el siguiente enlace:

<https://kodekloud.com/courses/labs-docker-for-the-absolute-beginner-hands-on/>

posteriormente deberá crear una cuenta gratuita para poder empezar a utilizar el entorno que brinda la plataforma.

Programación

- Guía de laboratorio 1 - Comandos básicos de Docker.
- Guía de laboratorio 2 - Comandos de ejecución de Docker.
- Guía de laboratorio 3 - Imágenes en Docker.
- Guía de laboratorio 4 - Variables de entorno.



- Guía de laboratorio 5 - CMD y puntos de entrada.
- Guía de laboratorio 6 - Docker compose.
- Guía de laboratorio 7 - Almacenamiento de Docker.
- Guía de laboratorio 8 - Redes Docker.

Guías	Título	Tiempo
Guía de laboratorio 1	Comandos básicos para Docker	45 minutos
Guía de laboratorio 2	Comandos de ejecución de Docker	30 minutos
Guía de laboratorio 3	Imágenes de Docker	45 minutos
Guía de laboratorio 4	Variables de entorno	30 minutos
Guía de laboratorio 5	CMD y puntos de entrada	35 minutos
Guía de laboratorio 6	Docker compose	40 minutos
Guía de laboratorio 7	Almacenamiento de Docker	45 minutos
Guía de laboratorio 8	Redes de Docker	45 minutos

Nota: La plataforma KodeKloud brinda un tiempo máximo de 1 hora para darle solución a cada una de las guías, aunque el tiempo que tome resolverlas podría ser menor, si tiene pocos o nulo conocimiento sobre la tecnología de contenedores Docker le motivamos a revisar la documentación teórica proporcionada en este trabajo investigativo para que sirva como material de apoyo.

Guía de laboratorio 1 - Comandos básicos de Docker

Introducción

En esta guía de laboratorio se abordarán los comandos básicos de Docker para realizar acciones como verificar versión de Docker instalada en el equipo, listar los contenedores en ejecución, conocer la cantidad de imágenes disponibles, ejecutar, detener y eliminar contenedores, descargar y eliminar imágenes, etc., tiene como objetivo ayudar a los estudiantes a comprender y familiarizarse con los comandos que permiten realizar las acciones básicas de Docker.

Duración estimada en la realización de la guía

45 minutos.

Desarrollo de la guía



1) ¿Cuál es la versión del servidor Docker que se ejecuta en el host?

Ejecute el comando “docker versión” para saber la versión del servidor de Docker. Donde podemos apreciar que la versión del servidor Docker es la 19.03.15.

```
$ docker version
Client: Docker Engine - Community
Version: 19.03.15
API version: 1.40
Go version: go1.13.15
Git commit: 99e3ed8919
Built: Sat Jan 30 03:17:11 2021
OS/Arch: linux/amd64
Experimental: false

Server: Docker Engine - Community
Engine:
Version: 19.03.15
API version: 1.40 (minimum version 1.12)
Go version: go1.13.15
Git commit: 99e3ed8919
Built: Sat Jan 30 03:15:40 2021
OS/Arch: linux/amd64
Experimental: false
containerd:
Version: 1.4.3
GitCommit: 269548fa27e8089a8b8278fc4fc781d7f65a939b
runc:
Version: 1.0.0-rc92
GitCommit: ff819c7e9184c13b7c2607fe6c30ae19403a7aff
docker-init:
Version: 0.18.0
GitCommit: fec3683
```

Ilustración 47: Verificación de la versión de Docker

2. ¿Cuántos contenedores se ejecutan en este host?

Ejecutar el comando “**docker ps**” para poder ver la cantidad de contenedores en ejecución, al utilizarlo se puede observar que no hay contenedores en ejecución.

```
$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
$
```

Ilustración 48: Verificación de la cantidad de contenedores activos

3. ¿Cuántas imágenes hay disponibles en este host?

Ejecutar el comando “**docker images**” para poder ver la cantidad de imágenes disponibles, donde se aprecia que hay 9 host disponibles.



```
$ docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
redis                latest      ccee4cdf984f     21 months ago  105MB
ubuntu              latest      7e0aa2d69a15     21 months ago  72.7MB
mysql               latest      0627ec6901db     22 months ago  556MB
nginx               alpine     a64a6e03b055     22 months ago  22.6MB
alpine              latest      6dbb9cc54074     22 months ago  5.61MB
nginx               latest      62d49f9bab67     22 months ago  133MB
postgres            latest      26c8bcd8b719     22 months ago  314MB
kodekloud/simple-webapp-mysql latest      129dd9f67367     4 years ago    96.6MB
kodekloud/simple-webapp latest      c6e3cd9aae36     4 years ago    84.8MB
$
```

Ilustración 49: Verificación de la cantidad de imágenes disponibles

4. Ejecute un contenedor usando la imagen redis.

Ejecutar el comando “**docker run redis**”.

```
$ docker run redis
1:C 11 Feb 2023 04:42:05.046 # 000000000000 Redis is starting 000000000000
1:C 11 Feb 2023 04:42:05.046 # Redis version=6.2.3, bits=64, commit=00000000, modified=0, pid=1, just started
1:C 11 Feb 2023 04:42:05.046 # Warning: no config file specified, using the default config. In order to specify a config file use redis
-server /path/to/redis.conf
1:M 11 Feb 2023 04:42:05.046 * monotonic clock: POSIX clock_gettime
1:M 11 Feb 2023 04:42:05.047 * Running mode=standalone, port=6379.
1:M 11 Feb 2023 04:42:05.047 # Server initialized
1:M 11 Feb 2023 04:42:05.048 # WARNING overcommit_memory is set to 0! Background save may fail under low memory condition. To fix this
issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to
take effect.
1:M 11 Feb 2023 04:42:05.048 * Ready to accept connections

```

Ilustración 50: Ejecución de un contenedor

5. Detenga el contenedor que acaba de crear.

Ejecutar el comando `ctrl+c`, si el contenedor se encuentra en segundo plano ejecutar el comando “**docker stop [id container]**”, para poder extraer el id del contenedor utilizara el comando “**docker ps**”.

```
$ docker stop 27097276b61c
27097276b61c
$
```

Ilustración 51: Deteniendo un contenedor

6. ¿Cuántos contenedores se están ejecutando en este host ahora?

Ejecutar el comando “**docker ps**” para ver la cantidad de contenedores ejecutándose en el momento, y se puede observar que no hay ningún contenedor en ejecución.



```
$ docker ps
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS        PORTS          NAMES
$
```

Ilustración 52: Cantidad de contenedores en ejecución

7. ¿Cuántos contenedores están presentes en el host ahora? (incluidos los que se ejecutan y los que no se ejecutan).

Ejecutar el comando “**docker ps -a**” para poder ver la cantidad de contenedores dentro del host, que son 7 contenedores.

```
$ docker ps -a
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS        PORTS          NAMES
835f63de9849   alpine        "/bin/sh"      49 seconds ago Exited (0) 47 seconds ago
01d75953aaf0   alpine        "sleep 1000"   50 seconds ago Up 48 seconds
day
38ba384aa47e   nginx:alpine  "/docker-ent... 52 seconds ago Up 50 seconds        80/tcp          nginx-2
e9553e08912d   nginx:alpine  "/docker-ent... 53 seconds ago Up 51 seconds        80/tcp          nginx-1
0646ab72fe9f   ubuntu        "sleep 1000"   54 seconds ago Up 52 seconds
ut
27097270b61c   redis        "docker-ent...  4 minutes ago  Exited (0) 2 minutes ago
talcini
5c484e66291b   redis        "docker-ent...  5 minutes ago  Exited (0) 4 minutes ago
on
pedantic_johns
```

Ilustración 53: Verificación de los contenedores existentes

8. ¿Cuál es la imagen utilizada para ejecutar el contenedor nginx-1?

Ejecutar el comando “**docker ps**” y observar la columna “IMAGE” donde la imagen utilizada en el contenedor nginx-1 es nginx:alpine.

```
$ docker ps
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS        PORTS          NAMES
83d75953aaf0   alpine        "sleep 1000"   6 minutes ago Up 6 minutes
inspiring_faraday
38ba384aa47e   nginx:alpine  "/docker-ent... 6 minutes ago Up 6 minutes        80/tcp          nginx-2
e9553e08912d   nginx:alpine  "/docker-ent... 6 minutes ago Up 6 minutes        80/tcp          nginx-1
0646ab72fe9f   ubuntu        "sleep 1000"   6 minutes ago Up 6 minutes
awesome_northcut
```

Ilustración 54: Verificación de la imagen que utiliza un contenedor

9. ¿Cuál es el nombre del contenedor creado usando la imagen de Ubuntu?

Ejecutar el comando “**docker ps**” observar la columna “NAMES” donde podemos observar que para Ubuntu el contenedor creado es “Awesome_northcut”.



```
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
01d75853aaf0      alpine             "sleep 1000"       7 minutes ago      Up 7 minutes       80/tcp            inspiring_faraday
30ba384aa47e      nginx:alpine      "/docker-entrypoint..." 7 minutes ago      Up 7 minutes       80/tcp            nginx-2
e9553e88912d      nginx:alpine      "/docker-entrypoint..." 7 minutes ago      Up 7 minutes       80/tcp            nginx-1
0646ab72fe9f      ubuntu            "sleep 1000"       7 minutes ago      Up 7 minutes
$
```

Ilustración 55: Verificación del nombre de un contenedor

10. ¿Cuál es el ID del contenedor que usa la imagen alpine y no se está ejecutando?
Ejecutar el comando “**docker ps -a**” donde el ID se identifica con el contenedor que utiliza alpine que es 035f63de9849.

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
035f63de9849      alpine             "/bin/sh"          10 minutes ago     Exited (0) 10 minutes ago
01d75853aaf0      alpine            "sleep 1000"       10 minutes ago     Up 10 minutes
day
30ba384aa47e      nginx:alpine      "/docker-entrypoint..." 10 minutes ago     Up 10 minutes       80/tcp            nginx-2
e9553e88912d      nginx:alpine      "/docker-entrypoint..." 10 minutes ago     Up 10 minutes       80/tcp            nginx-1
0646ab72fe9f      ubuntu            "sleep 1000"       10 minutes ago     Up 10 minutes
ut
27897276b61c      redis             "docker-entrypoint.s..." 13 minutes ago     Exited (0) 11 minutes ago
mystifying_mon
talcini
5c404e66291b      redis             "docker-entrypoint.s..." 15 minutes ago     Exited (0) 14 minutes ago
pedantic_johns
on
$
```

Ilustración 56: Verificación del ID de un contenedor

11. ¿Cuál es el estado del contenedor alpine detenido?
Ejecutar el comando “**docker ps -a**” y se observa la columna “STATUS” el estado del contenedor es exited, el cual indica que no está en ejecución.

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
035f63de9849      alpine             "/bin/sh"          13 minutes ago     Exited (0) 13 minutes ago
01d75853aaf0      alpine            "sleep 1000"       13 minutes ago     Up 13 minutes
inspiring_fara
day
30ba384aa47e      nginx:alpine      "/docker-entrypoint..." 13 minutes ago     Up 13 minutes       80/tcp            nginx-2
e9553e88912d      nginx:alpine      "/docker-entrypoint..." 13 minutes ago     Up 13 minutes       80/tcp            nginx-1
0646ab72fe9f      ubuntu            "sleep 1000"       13 minutes ago     Up 13 minutes
awesome_northc
ut
27897276b61c      redis             "docker-entrypoint.s..." 16 minutes ago     Exited (0) 14 minutes ago
mystifying_mon
talcini
5c404e66291b      redis             "docker-entrypoint.s..." 17 minutes ago     Exited (0) 16 minutes ago
pedantic_johns
on
$
```

Ilustración 57: Verificación del estado de un contenedor

12. Elimine todos los contenedores del Docker Host, tanto los que se ejecutan como los que no se ejecutan.



Nota: Recordar que es posible que se deba detener todos los contenedores para realizar dicha acción.

Ejecutar el comando “**docker stop \$(docker ps -aq)**” que se utiliza para detener todos los contenedores a la vez, luego utilizar el comando “**docker rm \$(docker ps -aq)**” que se utiliza para eliminar todos los contenedores.

```
$ docker stop $(docker ps -aq)
035f63de9849
01d75053aaf0
30ba384aa47e
e9553e08912d
0646ab72fe9f
27097276b61c
5c484e66291b
$ docker rm $(docker ps -aq)
035f63de9849
01d75053aaf0
30ba384aa47e
e9553e08912d
0646ab72fe9f
27097276b61c
5c484e66291b
$
```

Ilustración 58: Detener y eliminar todos los contenedores

13. Elimine la imagen de Ubuntu.

Ejecutar el comando “**docker rmi ubuntu**” para eliminar la imagen Ubuntu.

```
$ docker rmi ubuntu
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:cf31af331f38d1d7158470e095b132acd126a7180a54f263d386da88eb681d93
Deleted: sha256:7e0aa2d69a153215c790488ed1fcec162015e973e49962d438e18249d16fa9bd
Deleted: sha256:3dd8c8d4fd5b59d543c8f75a67cdfaab30aef5a6d99aea3fe74d8cc69d4e7bf2
Deleted: sha256:8d8dceacec7085abcab1f93ac1128765bc6cf0caac334c821e01546bd96eb741
Deleted: sha256:ccdbb80308cc5ef43b605ac28fac29c6a597f89f5a169bbdbb8dec29c987439
$
```

Ilustración 59: Eliminar una imagen

14. Extraer una imagen que se utilizara para ejecutar un contenedor más adelante. Descargue la imagen nginx:1.14-alpine, solo descargue dicha imagen, no cree un contenedor.



Ejecutar el comando “**docker pull nginx:1.14-alpine**”

```
$ docker pull nginx:1.14-alpine
1.14-alpine: Pulling from library/nginx
bdf0281b3a85: Pull complete
3d8a573c81ed: Pull complete
8129faeb2eb6: Pull complete
3dc99f571daf: Pull complete
Digest: sha256:485b610fefec7ff6c463ced9623314a04ed67e3945b9c08d7e53a47f6d106dc7
Status: Downloaded newer image for nginx:1.14-alpine
docker.io/library/nginx:1.14-alpine
$
```

Ilustración 60: Descargar una imagen

15. Ejecute un contenedor con la imagen nginx:1.14-alpine y asígnele el nombre webapp.

Ejecutar el comando “**docker run -d --name webapp nginx:1.14-alpine**” y verifique el estado del contenedor creado con el comando “**docker ps**”, se observa que el contenedor fue creado exitosamente y se encuentra en ejecución. El indicador -d permite ejecutar el contenedor en segundo plano.

```
$ docker run -d --name webapp nginx:1.14-alpine
78b628587801
$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS        NAMES
78b628587801   nginx:1.14-alpine  "nginx -g 'daemon of..." 13 seconds ago  Up 12 seconds  80/tcp      webapp
```

Ilustración 61: Ejecución y asignación de nombre a un contenedor

16. Elimine todas las imágenes en el host, retire los contenedores según sea necesario.

Note: Es importante que detenga y elimine todos los contenedores que utilizan imágenes.

Ejecutar el comando “**docker stop \$(docker ps -aq)**” para detener todos los contenedores, el comando “**docker rm \$(docker ps -aq)**” para eliminar todos los contenedores, y el comando “**docker rmi \$(docker images -aq)**” para eliminar todas las imágenes disponibles.



```
$ docker rm $(docker ps -aq)
20b028587801
$ docker rmi $(docker images -aq)
Untagged: nginx:1.14-alpine
Untagged: nginx@sha256:485b610fefec7ff6c463ced9623314a04ed67e3945b9c08d7e53a47f6d108dc7
Deleted: sha256:8a2fb25a19f5dc1528b7a3fabe8b3145ff57fe10e4f1edac6c718a3cf4aa4b73
Deleted: sha256:f68a8bcb9dbd06e0d2750eabf63c45f51734a72831ed650d2349775865d5fc20
Deleted: sha256:cbf2c7789332fe231e8defa490527a7b2c3ae8589997ceee00895f3263f0a8cf
Deleted: sha256:894f3fad7e6ecd7f24e88340a44b7b73663a85c0eb7740e7ade169e9d8491a4c
Deleted: sha256:a464c54f93a9e88fc1d33df1e0e39cca427d60145a360962e8f19a1dbf900da9
$ █
```

Ilustración 62: Detención de contenedores y eliminación de imágenes

```
$ docker rmi $(docker images -aq)
Untagged: redis:latest
Untagged: redis@sha256:eff56acc5fc7b909183da93236ba09d3b8cb7d6db31d5b25e9a46dac9b5e699b
Deleted: sha256:ccce4cdf984f11952441cbab30146dae792c8d9fb668e762c78a49e7db858082
Deleted: sha256:a4f3d68d06ee9aec1862695818528930f0e889f5c892da8be4c06759dc2e8f86
Deleted: sha256:02c22c39334cefb22054330d3be804a3046286d7f0a47d83f5f25be13f4e635e
Deleted: sha256:2d01997f1c500789fc369f5440ea810257641af6fbb1d4eae9558d94ea8879d
Deleted: sha256:f448a5cc1367794bb2f15d887438c8b43f2324ee3523f5d95c15997685cc816d
Deleted: sha256:e481db33c0c73ee7de5ecda57c21fcfb7eeeff7d22fcc4ad0166b0bf64a63a32
Untagged: mysql:latest
Untagged: mysql@sha256:04ee7141256e83797ea4a84a4d31b1f1bc10111c8d1bc1879d52729ccd19e20a
Deleted: sha256:0627ec6901db4b2aed6ca7ab35e43e19838ba079fffe8fe1be66b6feaad694de
Deleted: sha256:94d5db550d62032ddc8ad8d4cfe8bac06fe7d35757deb2f8638d0fcd1e89217
Deleted: sha256:5fbbb0e9bfb8f2a4d665cc60aaf4876191c057898db276d2d030a0d6123afc8e
Deleted: sha256:124a9d262ebecea284b6438b0a04766b076bd3f89cb0f29a8a475f26bf93911f
Deleted: sha256:5aa37b6c8e31197ab2f357c09755bcb83228ff0fb69c2009a12b6eddb087e884
Deleted: sha256:d0b1d5665c308823bbc5b8986d405e5202ade6994dfa00d8ff576eff372dd045
Deleted: sha256:cd35e2328f0670969657f1abae8beffbc1eb1fddba667e1e6e6286598500a35
Deleted: sha256:068b92efc0504adcd3c23f16fde80775a2f4dfe485e242206f638eae72c4fa1b
Deleted: sha256:7c8818a166d9666c68fcdbe421c30568d60d51a505e540f42901664113047a75
Deleted: sha256:5aa8f65565168fd7db2aa6b9f8fb1db746aa598fa3854dcbdbb49d5a29f6d8a5
Deleted: sha256:cca9d1bafa1ee67bb4d7178732c0955a40a5dea6e5b989f61248984f26f7306b
Deleted: sha256:34ca91e79c4027120ca740231d415c739cccad57d1ee68d6a6e67ca60bbaf3a4
Untagged: nginx:alpine
Untagged: nginx@sha256:07ab71a2c8e4ecb19a5a5abcfb3a4f175946c001c8af288b1aa766d67b0d05d2
Deleted: sha256:a64a6e03b0551e1cefa94db6cc6677fb1efed3c557d173f79584ff4ec474b5ae
Deleted: sha256:d950b497e5a0787af1b4a04e0298b693501d756b610b09e5501bc0d1feb02465
Deleted: sha256:01270ad0039edf3793b69b5374505aad02fc2e4464f460215a803fa728eae8c
Deleted: sha256:ac0a87a7573b9beacc5b597fdf1165de47504fd447b5cb7ee4e97a433f8e7895
Deleted: sha256:163a1da5e695d11c7f885e360da692ad126f51e9456d1b0b91266773e6c732b4
Deleted: sha256:d3658096dce40fe9c464f3d02006e3945545827c8581212c1e274da7dc033cbc
Untagged: alpine:latest
Untagged: alpine@sha256:69e70a79f2d41ab5d637de98c1e0b055206ba40a8145e7bddd55ccc04e13cf8f
Deleted: sha256:6dbb9cc54074106d46d4ccb330f2a40a682d49dda5f4844962b7dce9fe44aaec
Deleted: sha256:b2d5eeeba3a22b9b8aa97261957974a6bd65274ebd43e1d81d0a7b8b752b116
```

Ilustración 63: Eliminación de todas las imágenes



Guía de laboratorio 2 – Comandos de ejecución de Docker

Introducción

En esta guía se abordarán comandos básicos sobre ejecución de contenedores Docker, así como la identificación de datos importantes que hay que tener en cuenta para un buen funcionamiento de estos, esta guía tiene como objetivo que el estudiante pueda comprender sobre la ejecución de contenedores y sobre cómo puede establecer parámetros al momento de ejecutarlo.

Duración estimada en la realización de la guía es de:

30 minutos

Desarrollo de la guía

1. ¿Cuántos contenedores se ejecutan en este host actualmente?

Para ver el resultado de cuantos contenedores se están ejecutando, utilizar el comando **“docker ps”** que permite ver la cantidad de contenedores ejecutándose, donde se están ejecutando 1 contenedor.

```
$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
c48a896811c5  nginx:alpine  "/docker-entrypoint..."  14 seconds ago  Up 13 seconds  0.0.0.0:3456->3456/tcp, 0.0.0.0:38880->80/tcp  busy_ellis
```

Ilustración 64: Contenedores en ejecución

2. ¿Cuál es la imagen utilizada por el contenedor?

Para saber el nombre de la imagen utilizada por un contenedor ejecute el comando **“docker ps -a”** donde observamos que el valor en la columna **“IMAGE”** teniendo como resultado que la imagen utilizada en el contenedor es **“nginx:alpine”**

```
$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
c48a896811c5  nginx:alpine  "/docker-entrypoint..."  6 minutes ago  Up 6 minutes  0.0.0.0:3456->3456/tcp, 0.0.0.0:38880->80/tcp  busy_ellis
```

Ilustración 65: Imagen que usa un contenedor



3. ¿Cuántos puertos se publican en este contenedor?

Ejecutar el comando “**docker ps**” para observar los puertos que publica este contenedor viendo la columna “PORTS” donde el resultado que se muestra en este contenedor son 2 puertos.

```
$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
de54975e6fba  nginx:alpine  "/docker-entrypoint..."  20 minutos ago  Up 20 minutos  0.0.0.0:3456->3456/tcp, 0.0.0.0:38080->80/tcp  vigilant_mandelstov
```

Ilustración 66: Puertos publicados por el contenedor

4. ¿Cuáles son los números de puertos expuestos en el contenedor?

Ejecutar de nuevo comando “**docker ps**” donde los puertos expuestos son los 3456 y el 80.

```
$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
de54975e6fba  nginx:alpine  "/docker-entrypoint..."  20 minutos ago  Up 20 minutos  0.0.0.0:3456->3456/tcp, 0.0.0.0:38080->80/tcp  vigilant_mandelstov
```

Ilustración 67: Número de puertos expuestos en el contenedor

5. ¿Cuáles son los números de puertos que están publicados en el host?

Los números de puertos publicados en el host son el 3456 y el 38080.

```
$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
de54975e6fba  nginx:alpine  "/docker-entrypoint..."  20 minutos ago  Up 20 minutos  0.0.0.0:3456->3456/tcp, 0.0.0.0:38080->80/tcp  vigilant_mandelstov
```

Ilustración 68: Numero de puertos expuestos en el host

6. Ejecutar una instancia de kodekloud/simple-webapp con una etiqueta blue y map port 8080 en contenedor con puerto 38282 en el host.

Es importante ejecutar imágenes con un puerto no solo para el contenedor sino también para el host. Ya que así se podrá acceder desde el puerto indicado en el host al puerto especificado en el contenedor, manteniendo comunicación en ambos sentidos, es este inciso primero se buscará la imagen que se indica.



```
$ docker images
REPOSITORY          TAG          IMAGE ID          CREATED          SIZE
redis               latest      ccee4cdf984f     21 months ago  105MB
ubuntu             latest      7e0aa2d69a15     22 months ago  72.7MB
mysql              latest      0627ec6901db     22 months ago  556MB
nginx              alpine      a64a6e03b055     22 months ago  22.6MB
alpine             latest      6dbb9cc54074     22 months ago  5.61MB
nginx              latest      62d49f9bab67     22 months ago  133MB
postgres           latest      26c8bcd8b719     22 months ago  314MB
kodekloud/simple-webapp-mysql latest      129dd9f67367     4 years ago    96.6MB
kodekloud/simple-webapp latest      c6e3cd9aae36     4 years ago    84.8MB
$
```

Ilustración 69: Búsqueda de imagen KodeKloud/simple-webapp

Cuando se encuentre la imagen deseada se abordará un concepto nuevo como son los TAG o etiquetas, las cuales brinda una mejor administración del control de versiones, ya que es posible tener varias instancias de una misma imagen y de esta manera saber cuál de ellas se está trabajando.

Utilizaremos el comando “**docker run -p 38282:8080 kodekloud/simple-webapp:blue**” donde “docker run” ejecuta la imagen, “-p 38282:8080 “ publica un puerto 38282 para redirigirlo al puerto 8080 en el contenedor y “**kodekloud/simple-webapp:blue**” se hace la asignación de etiqueta blue a la instancia.

```
$ docker run -p 38282:8080 kodekloud/simple-webapp:blue
Unable to find image 'kodekloud/simple-webapp:blue' locally
blue: Pulling from kodekloud/simple-webapp
4fe2ade4980c: Already exists
7cf6a1d62200: Already exists
f0d690b9e495: Already exists
fac5d45ad062: Already exists
a6fc8a0deb7d: Pull complete
f43c8e496f88: Pull complete
58ca939f7651: Pull complete
095a1a007cdb: Pull complete
Digest: sha256:9caf15476dc60b77c7460791bea8ea5f6ca02b90199aabe088beea83bc943fe5
Status: Downloaded newer image for kodekloud/simple-webapp:blue
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
█
```

Ilustración 70: Instancia de KodeKloud/simple-webapp con un tag



Guía de laboratorio 3 – Imágenes en Docker

Introducción

En la presente guía de laboratorio se abordarán las imágenes en docker las cuales son plantillas que cuentan con una aplicación, los binarios, librerías y todo lo necesario para construir contenedores Docker, el objetivo de la presente guía es ayudar a los estudiantes a comprender y familiarizarse los comandos básicos para trabajar con imágenes docker.

Duración estimada para la realización de la guía:

45 minutos

Desarrollo de la guía

1. ¿Cuántas imágenes hay disponibles en el host?

Utilizando el comando “**docker images**” para ver todas las imágenes disponibles, en total hay 9 imágenes.

```
$ docker images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
redis               latest      ccee4cdf984f     21 months ago    105MB
ubuntu             latest      7e0aa2d69a15     22 months ago    72.7MB
mysql              latest      0627ec6901db     22 months ago    556MB
nginx              alpine     a64a6e03b055     22 months ago    22.6MB
alpine             latest      6dbb9cc54074     22 months ago    5.61MB
nginx              latest      62d49f9bab67     22 months ago    133MB
postgres           latest      26c8bcd8b719     22 months ago    314MB
kodekloud/simple-webapp-mysql latest      129dd9f67367     4 years ago      96.6MB
kodekloud/simple-webapp latest      c6e3cd9aae36     4 years ago      84.8MB
$
```

Ilustración 71: Verificación de la cantidad de imágenes disponibles

2. ¿Cuál es el tamaño de la imagen Ubuntu?

Ejecutar el comando “**docker images**” y busque en la columna “SIZE” donde la imagen Ubuntu tiene un tamaño de 72.7 MB.



```
$ docker images
REPOSITORY          TAG          IMAGE ID        CREATED         SIZE
redis               latest      ccee4cdf984f   21 months ago  105MB
ubuntu             latest      7e0aa2d69a15   22 months ago  72.7MB
mysql              latest      0627ec6901db   22 months ago  556MB
nginx              alpine     a64a6e03b055   22 months ago  22.6MB
alpine             latest      6dbb9cc54074   22 months ago  5.61MB
nginx              latest      62d49f9bab67   22 months ago  133MB
postgres           latest      26c8bcd8b719   22 months ago  314MB
kodekloud/simple-webapp-mysql latest      129dd9f67367   4 years ago    96.6MB
kodekloud/simple-webapp latest      c6e3cd9aae36   4 years ago    84.8MB
$
```

Ilustración 72: Verificación del tamaño de una imagen

3. Se agrego una nueva imagen. ¿Cuál es la etiqueta en la imagen NGINX recién extraída?

Ejecutar el comando “**docker images**” visualizar la columna TAG la etiqueta en la imagen NGINX recién extraída es 1.14-alpine.

```
$ docker images
REPOSITORY          TAG          IMAGE ID        CREATED         SIZE
redis               latest      ccee4cdf984f   21 months ago  105MB
ubuntu             latest      7e0aa2d69a15   22 months ago  72.7MB
mysql              latest      0627ec6901db   22 months ago  556MB
nginx              alpine     a64a6e03b055   22 months ago  22.6MB
alpine             latest      6dbb9cc54074   22 months ago  5.61MB
nginx              latest      62d49f9bab67   22 months ago  133MB
postgres           latest      26c8bcd8b719   22 months ago  314MB
nginx              1.14-alpine 8a2fb25a19f5   3 years ago    16MB
kodekloud/simple-webapp-mysql latest      129dd9f67367   4 years ago    96.6MB
kodekloud/simple-webapp latest      c6e3cd9aae36   4 years ago    84.8MB
$
```

Ilustración 73: Verificación de la etiqueta de una imagen

4. Se descargó el código de una aplicación ¿Cuál es la imagen base utilizada en el Dockerfile?

Abra el archivo Dockerfile ubicado dentro del directorio webapp-color usando el comando “**nano /root/webapp-color/Dockerfile**” y buscar la instrucción “FROM” o búsqueda directamente utilizando el comando “**grep -i FROM /root/webapp-color/Dockerfile**” donde el resultado en ambas soluciones es que la imagen base utilizada en el Dockerfile es python3.6.



Solución 1

```
GNU nano 2.7.4 File: /root/webapp-color/Dockerfile
FROM python:3.6
RUN pip install flask
COPY . /opt/
EXPOSE 8080
WORKDIR /opt
ENTRYPOINT ["python", "app.py"]
```

Ilustración 74: Verificación de imagen usada a través del Dockerfile

Solución 2

```
$ grep -i FROM /root/webapp-color/Dockerfile
FROM python:3.6
$
```

Ilustración 75: Verificación de imagen usada a través de la terminal

5. ¿En qué ubicación dentro del contenedor se copia el código de la aplicación durante una compilación de Docker?

Abra el archivo Dockerfile ubicado en el directorio “webapp-color” y busque el comando “COPY”, donde el código de la aplicación se copia en el directorio /opt dentro del contenedor durante el proceso de compilación.

```
GNU nano 2.7.4 File: /root/webapp-color/Dockerfile
FROM python:3.6
RUN pip install flask
COPY . /opt/
EXPOSE 8080
WORKDIR /opt
ENTRYPOINT ["python", "app.py"]
```

Ilustración 76: Uso del comando COPY en un Dockerfile

6. Cuando se crea un contenedor usando la imagen creada con este Dockerfile, ¿cuál es el comando que se usa para EJECUTAR la aplicación dentro de él?



Abrir el fichero Dockerfile ubicado en el directorio webapp-color y busque el comando “ENTRYPOINT” donde el comando utilizado para ejecutar la aplicación dentro del contenedor es Python app.py

```
GNU nano 2.7.4 File: /root/webapp-color/Dockerfile
FROM python:3.6
RUN pip install flask
COPY . /opt/
EXPOSE 8080
WORKDIR /opt
ENTRYPOINT ["python", "app.py"]
```

Ilustración 77: Uso del comando ENTRYPOINT en un Dockerfile

7. ¿En qué puerto se ejecuta la aplicación web dentro del contenedor?

Abrir el fichero Dockerfile y buscar el puerto en el comando “EXPOSE” la aplicación web se ejecuta en el puerto 8080 dentro del contenedor.

```
GNU nano 2.7.4 File: /root/webapp-color/Dockerfile
FROM python:3.6
RUN pip install flask
COPY . /opt/
EXPOSE 8080
WORKDIR /opt
ENTRYPOINT ["python", "app.py"]
```

Ilustración 78: Uso del comando EXPOSE en un Dockerfile

8. Cree una imagen con Dockerfile y asígnele el nombre webapp-color.

Dirigirse al directorio en donde se encuentra ubicado el Dockerfile usando el comando “**cd /root/webapp-color**” y ejecute el comando de compilación “**docker build -t webapp-color**” dentro de dicho directorio.



```
$ pwd
/root/webapp-color
$ docker build -t webapp-color .
Sending build context to Docker daemon 121.3kB
Step 1/6 : FROM python:3.6
3.6: Pulling from library/python
0e29546d541c: Download complete
9b829c73b52b: Download complete
cb5b7ae36172: Download complete
6494e4811622: Downloading [=====>] 36.04MB/54.57MB
6f9f74896dfa: Downloading [=====] 33.29MB/196.5MB
5e3b1213efc5: Downloading [>] 63.13kB/6.291MB
9fddfdc56334: Pulling fs layer
404f82044bac: Waiting
c4f42be2be53: Waiting
█
```

Ilustración 79: Creación de la imagen webapp-color

9. Ejecute una instancia de la imagen webapp-color y publique el puerto 8080 en el contenedor al 8282 en el host.

9.1. Aspectos para tener en cuenta.

- Contenedor con imagen 'webapp-color'
- Puerto en el contenedor: 8080
- Puerto en el host: 8282

Para ello ejecutar el comando “**docker run -p 8282:8080 webapp-color**”.

```
$ docker run -p 8282:8080 webapp-color
This is a sample web application that displays a colored background.
A color can be specified in two ways.

1. As a command line argument with --color as the argument. Accepts one of red,green,blue,blue2,pink,darkblue
2. As an Environment variable APP_COLOR. Accepts one of red,green,blue,blue2,pink,darkblue
3. If none of the above then a random color is picked from the above list.
Note: Command line argument precedes over environment variable.

No command line argument or environment variable. Picking a Random Color =green
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://172.12.0.2:8080/ (Press CTRL+C to quit)
█
```

Ilustración 80: Ejecución de una instancia de la imagen webapp-color



10. Acceda a la aplicación haciendo click en la pestaña HOST:8282 arriba de su terminal.



Ilustración 81: Funcionamiento de la aplicación webapp-color

Como se puede observar se accedió correctamente al contenedor, y para poderlo detener solo hacemos “control + c” para poder detener el contenedor.

11. ¿Cuál es el sistema operativo base utilizado por la imagen python:3.6? Ejecutar una instancia de la imagen y luego ejecute el comando “**docker run python:3.6 cat /etc/*release**”. El sistema operativo base utilizado por la imagen python3.6 es Debian 11 Bullseye.

```
$ docker run python:3.6 cat /etc/*release*
PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"
NAME="Debian GNU/Linux"
VERSION_ID="11"
VERSION="11 (bullseye)"
VERSION_CODENAME=bullseye
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
$
```

Ilustración 82: Verificación del S.O utilizado por una imagen

12. ¿Cuál es el tamaño aproximado de la imagen webapp-color? El tamaño de la imagen webapp-color es de 913 MB.



```
$ docker images
REPOSITORY          TAG          IMAGE ID      CREATED       SIZE
webapp-color        latest      e2544653cb08 47 minutes ago 913MB
python              3.6        54260638d07c 14 months ago 902MB
redis               latest     ccee4cdf984f 22 months ago 105MB
ubuntu              latest     7e0aa2d69a15 22 months ago 72.7MB
mysql               latest     0627ec6901db 22 months ago 556MB
nginx               alpine     a64a6e03b055 22 months ago 22.6MB
alpine              latest     6dbb9cc54074 22 months ago 5.61MB
nginx               latest     62d49f9bab67 22 months ago 133MB
postgres            latest     26c8bcd8b719 22 months ago 314MB
nginx               1.14-alpine 8a2fb25a19f5 3 years ago 16MB
kodekloud/simple-webapp-mysql latest     129dd9f67367 4 years ago 96.6MB
kodekloud/simple-webapp latest     c6e3cd9aae36 4 years ago 84.8MB
$
```

Ilustración 83: Tamaño de la imagen webapp-color

13. Cree una nueva imagen más pequeña modificando el mismo Dockerfile y asígnele el nombre webapp-color y etiquételo como lite.

Encuentre una imagen base más pequeña para python:3.6 y asegúrese de que la imagen final tenga menos de 150 MB. Modifique el Dockerfile para usar la imagen python:3.6-alpine y luego construya una usando **“docker build -t webapp-color:lite .”**

```
GNU nano 2.7.4 File: /root/webapp-color/Dockerfile Modified
FROM python:3.6-alpine
RUN pip install flask
COPY . /opt/
EXPOSE 8080
WORKDIR /opt
ENTRYPOINT ["python", "app.py"]
```

Ilustración 84: Modificación del Dockerfile de webapp-color



```
$ nano /root/webapp-color/Dockerfile
$ docker build -t webapp-color:lite .
Sending build context to Docker daemon 121.3kB
Step 1/6 : FROM python:3.6-alpine
3.6-alpine: Pulling from library/python
59bf1c3509f3: Pull complete
8786870f2876: Pull complete
acb0e804800e: Pull complete
52bedcb3e853: Pull complete
b064415ed3d7: Pull complete
Digest: sha256:579978dec4602646fe1262f02b96371779bfb0294e92c91392707fa999c0c989
Status: Downloaded newer image for python:3.6-alpine
--> 3a9e80fa4606
Step 2/6 : RUN pip install flask
--> Running in b6fa80778695
█
```

Ilustración 85: Construyendo la imagen webapp-color:lite

Verificar el tamaño de la imagen final usando el comando “**docker images**” donde podemos ver que la imagen webapp-color con el TAG “lite” ahora tiene un tamaño de 51.8 MB.

```
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
webapp-color        lite                f05c510c9cd4       25 seconds ago    51.8MB
webapp-color        latest             9bad670a74f9       11 minutes ago    913MB
python              3.6                54260638d07c       14 months ago     902MB
python              3.6-alpine        3a9e80fa4606       15 months ago     40.7MB
redis               latest             ccee4cdf984f       22 months ago     105MB
ubuntu              latest             7e0aa2069a15       22 months ago     72.7MB
mysql               latest             0627ec6901db       22 months ago     556MB
nginx               alpine            a64a6e03b055       22 months ago     22.6MB
alpine              latest             6d0b9cc54074       22 months ago     5.61MB
nginx               latest             62d49f9bab67       22 months ago     133MB
postgres            latest             26c8bcd8b719       22 months ago     314MB
nginx               1.14-alpine       8a2fb25a19f5       3 years ago        16MB
kodekloud/simple-webapp-mysql latest             129dd9f67367       4 years ago        96.6MB
kodekloud/simple-webapp latest             c6e3cd9aae36       4 years ago        84.8MB
$ █
```

Ilustración 86: Verificación del tamaño de la imagen webapp-color:lite

14. Ejecute una instancia de la nueva imagen webapp-color:lite y publique el puerto 8080 en el contenedor al 8383 en el host.

14.1. Aspectos a tener en cuenta:

- Contenedor con imagen 'webapp-color:lite'.
- Contenedor para publicar puerto 8080 a 8383.



Ejecutar el comando `docker run -p 8383:8080 -d webapp-color:lite` donde el indicador `-d` permite ejecutar el contenedor en segundo plano.

```
$ docker run -p 8383:8080 -d webapp-color:lite
ec63f648417e20d58a5982d066f43950f0bf4f94422332cfbdfbc5f3a1615699
$
```

Ilustración 87: Ejecución de contenedor webapp-color:lite con redirección de puertos

Guía de laboratorio 4 – variables de entorno

Introducción

En esta guía se abordarán las variables de entorno, que no son más que valores dinámicos que pueden ser establecidos para controlar el comportamiento de los contenedores, esta guía tiene como objetivo que el estudiante pueda entender que son las variables de entorno y lo importante que son a la hora de ejecutar contenedores.

Duración estimada en la realización de la guía

30 minutos

Desarrollo de la guía

1. Inspeccione las variables de entorno establecidas en el contenedor en ejecución e identifique el valor establecido en la variable `APP_COLOR`.

Ejecutar el comando “**docker ps**” para listar los contenedores en ejecución e identifique el nombre o ID del contenedor en ejecución.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
e1294364f9bc	kodakloud/simple-webapp	"python app.py"	14 seconds ago	Up 13 seconds	8080/tcp	missing_kaldysh

Ilustración 88: Listar contenedores en ejecución

Después de conocer el nombre o el ID del contenedor, ejecutar el siguiente comando para obtener los campos “`env`” del comando de inspección:



“docker inspect <nombre del contenedor / ID del contenedor> | grep -A 10 Env”

el comando identifica el contenedor utilizando su ID y con `grep -A 10` se toman las 10 primeras líneas del campo `Env` del resultado obtenido con el comando `inspect`. Donde el resultado indica que el valor establecido en la variable `APP_COLOR` es `Pink`.

```
$ docker inspect da69d503d639 | grep -A 10 Env
  "Env": [
    "APP_COLOR=pink",
    "PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
    "LANG=C.UTF-8",
    "GPG_KEY=0D96DF4D4110E5C43FBFB17F2D347EA6AA65421D",
    "PYTHON_VERSION=3.6.6",
    "PYTHON_PIP_VERSION=18.1"
  ],
  "Cmd": null,
  "Image": "kodekloud/simple-webapp",
  "Volumes": null,
$
```

Ilustración 89: Valor de la variable App_color en un contenedor

2. Ejecute un contenedor llamado `blue-app` usando la imagen `kodekloud/simplewebapp` y establezca la variable de entorno `APP_COLOR` en `blue`, haga que la aplicación esté disponible en el puerto `38282` del host. La aplicación escucha en el puerto `8080` en el contenedor.

Ejecutar el comando **“docker run -p 38282:8080 --name blue-app -e APP_COLOR=blue -d kodekloud/simple-webapp”**

```
$ docker run -p 32282:8080 --name blue-app -e APP_COLOR=blue -d kodekloud/simple-webapp
8c45825b6b1b0c37558d431c3409dbfc8a06ce3893dc67655c42b32e5c6dca90
$
```

Ilustración 90: Creación del contenedor blue_app con valor en la variable App_color

Para conocer el campo `Env` desde el contenedor de la aplicación web, ejecute el comando **“docker exec -it <Nombre o ID del contenedor> env”**, de modo que el comando quedaría de la siguiente manera **“docker exec -it blue-app env”**.



```
$ docker exec -it blue-app env
PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=8c45825b6b1b
TERM=xterm
APP_COLOR=blue
LANG=C.UTF-8
GPG_KEY=0D96DF4D4A110E5C43FBFB17F2D347EA6AA65421D
PYTHON_VERSION=3.6.6
PYTHON_PIP_VERSION=18.1
HOME=/root
$
```

Ilustración 91: Verificación del nuevo valor de App_color

El resultado obtenido indica que el valor “blue” fue establecido en la variable APP_COLOR de manera exitosa.

3. Vea la aplicación haciendo clic en el enlace HOST:38282 arriba de su terminal y asegúrese de que tenga el color correcto.

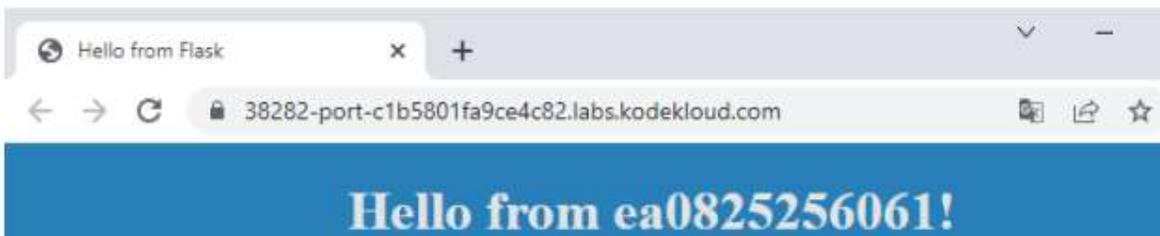


Ilustración 92: Verificación del color en la aplicación

4. Implemente una base de datos mysql utilizando la imagen mysql y asígnele el nombre mysql-db.

Establezca la contraseña de la base de datos para usar db_pass123. Busque la imagen mysql en Docker Hub e identifique la variable de entorno correcta que se usará para establecer la contraseña del usuario ROOT.



Variables de entorno

Cuando inicia la `mysql` imagen, puede ajustar la configuración de la instancia de MySQL pasando una o más variables de entorno en la `docker run` línea de comando. Tenga en cuenta que ninguna de las variables a continuación tendrá ningún efecto si inicia el contenedor con un directorio de datos que ya contiene una base de datos: cualquier base de datos preexistente siempre permanecerá intacta al iniciar el contenedor.

Consulte también <https://dev.mysql.com/doc/refman/5.7/en/environment-variables.html> para ver la documentación de las variables de entorno que MySQL respeta (especialmente variables como `MYSQL_HOST`, que se sabe que causa problemas cuando se usa con esta imagen).

`MYSQL_ROOT_PASSWORD`

Esta variable es obligatoria y especifica la contraseña que se establecerá para la `root` cuenta de superusuario de MySQL. En el ejemplo anterior, se configuró en `my-secret-pw`.

`MYSQL_DATABASE`

Esta variable es opcional y le permite especificar el nombre de una base de datos que se creará al iniciar la imagen. Si se proporcionó un usuario/contraseña (ver a continuación), a ese usuario se le otorgará acceso de superusuario (`correspondiente a GRANT ALL`) a esta base de datos.

`MYSQL_USER` , `MYSQL_PASSWORD`

Estas variables son opcionales, se usan en conjunto para crear un nuevo usuario y establecer la contraseña de ese usuario. A este usuario se le otorgarán permisos de superusuario (ver arriba) para la base de datos especificada por la `MYSQL_DATABASE` variable. Ambas variables son necesarias para crear un usuario.

Tenga en cuenta que no es necesario utilizar este mecanismo para crear el superusuario raíz, ese usuario se crea de forma predeterminada con la contraseña especificada por la `MYSQL_ROOT_PASSWORD` variable.

Ilustración 93: Documentación de las variables de entorno de imagen mysql en dockerhub

Según la documentación oficial de MySQL en DockerHub la variable de entorno que se utiliza para establecer la contraseña del usuario `root` es: **“`MYSQL_ROOT_PASSWORD`”**. Ejecutar el comando **“`docker run -d -e MYSQL_ROOT_PASSWORD=db_pass123 --name mysql-db mysql`”**.

```
$ docker run -d -e MYSQL_ROOT_PASSWORD=db_pass123 --name mysql-db mysql
dd655c1882b1cbded45b9cb8aaa77c5d88189d8c6364bf1e42251f5ff86bcd6
$
```

Ilustración 94: Creación de contenedor asignando valor a `MYSQL_ROOT_PASSWORD`

Para conocer el campo `env` desde el contenedor `mysql-db` ejecutar el comando **“`docker exec -it mysql-db env`”**.



```
$ docker exec -it mysql-db env
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOSTNAME=20fe6a34d8fb
TERM=xterm
MYSQL_ROOT_PASSWORD=db_pass123
GOSU_VERSION=1.12
MYSQL_MAJOR=8.0
MYSQL_VERSION=8.0.24-1debian10
HOME=/root
```

Ilustración 95: Conociendo las variables de entorno de un contenedor

El resultado obtenido indica algunos datos importantes sobre el contenedor como por ejemplo que la versión de MYSQL es 8.0.24 y que la contraseña del usuario root es “db_pass123”.

Guía de laboratorio 5 – CMD y puntos de entrada

Introducción

En esta guía se abordará información sobre CMD y puntos de entrada, donde punto de entrada se refiere a “entrypoint” que se encuentra al final de un Dockerfile, tiene la función de utilizar el ejecutable que se indique, si no se ubica nada se ejecuta por defecto “/bin/sh -c”, CMD por su parte establece el parámetro que se va a ejecutar, el objetivo de esta guía es que el estudiante logre entender para que se utilizan estos campos y como establecen valores en los mismos.

Duración estimada en la realización de la guía:

35 minutos

Desarrollo de la guía

1. ¿Cuál es el “ENTRYPOINT” configurado en la imagen de “mysql”?

Localice el Dockerfile de la imagen mysql, luego acceda al archivo con el comando nano, y localice el campo ENTRYPOINT. Puede usar ls.



```
$ ls
Dockerfile-mysql Dockerfile-python Dockerfile-ubuntu Dockerfile-wordpress app
$
```

Ilustración 96: Localización del Dockerfile de la imagen mysql

El ENTRYPOINT es “docker-entrypoint.sh”.

```
# the "/var/lib/mysql" stuff here is because the mysql-server postinst doesn't have an explicit way to disable the mysql_install_db ca
cpath besides having a database already "configured" (ie, stuff in /var/lib/mysql/mysql)
# also, we set debconf keys to make APT a little quieter
RUN { \
    echo mysql-community-server mysql-community-server/data-dir select '' ; \
    echo mysql-community-server mysql-community-server/root-pass password '' ; \
    echo mysql-community-server mysql-community-server/re-root-pass password '' ; \
    echo mysql-community-server mysql-community-server/remove-test-db select false; \
    } | debconf-set-selections \
    && apt-get update && apt-get install -y mysql-community-client-${MYSQL_VERSION} mysql-community-server-core-${MYSQL_VERSION}
    && rm -rf /var/lib/apt/lists/* \
    && rm -rf /var/lib/mysql && mkdir -p /var/lib/mysql /var/run/mysqld \
    && chown -R mysql:mysql /var/lib/mysql /var/run/mysqld \
# ensure that /var/run/mysqld (used for socket and lock files) is writable regardless of the UID our mysqld instance ends up having at
runtime
    && chmod 777 /var/run/mysqld

VOLUME /var/lib/mysql
# Config files
COPY config/ /etc/mysql/
COPY docker-entrypoint.sh /usr/local/bin/
RUN ln -s /usr/local/bin/docker-entrypoint.sh /entrypoint.sh # backwards compat
ENTRYPOINT ["docker-entrypoint.sh"]
EXPOSE 3306 33068
CMD ["mysqld"]
```

Ilustración 97: Verificación del valor entrypoint en la imagen mysql

2. ¿Cuál es el “CMD” configurado en la imagen de “wordpress”?

Se hará algo similar al punto anterior, pero con otro documento, en este caso será el documento “wordpress”.

```
$ ls
Dockerfile-mysql Dockerfile-python Dockerfile-ubuntu Dockerfile-wordpress app
$
```

Ilustración 98: Encontrar Dockerfile de wordpress

Al inspeccionar el código, en el campo CMD es “apache2-foreground”. Utilizar vim para poder ver el archivo.



```
RUN a2enmod rewrite expires
VOLUME /var/www/html
ENV WORDPRESS_VERSION 5.2.2
ENV WORDPRESS_SHA1 3605bcb9ea48d714efa59b0eb2d251657e7d5b0
RUN set -ex; \
  curl -o wordpress.tar.gz -fSL "https://wordpress.org/wordpress-${WORDPRESS_VERSION}.tar.gz"; \
  echo "wordpress.tar.gz" | sha1sum -c -; \
# upstream tarballs include ./wordpress/ so this gives us /usr/src/wordpress
tar -xzf wordpress.tar.gz -C /usr/src; \
rm wordpress.tar.gz; \
chown -R www-data:www-data /usr/src/wordpress
COPY docker-entrypoint.sh /usr/local/bin/
ENTRYPOINT ["docker-entrypoint.sh"]
CMD ["apache2-foreground"]
```

Ilustración 99: CMD del Dockerfile de la imagen wordpress

3. Cuando se lanza la imagen de wordpress, ¿Cuál es el comando final que se ejecuta al iniciar?, Considere ambos, tanto el ENTRYPOINT y CMD.

Como se vio en los conceptos los “ENTRYPOINT” se ejecutan después de los comandos de docker run, por lo tanto, el comando se definiría por el script a ejecutar y el parámetro pasado por el campo “CMD”, lo que daría:

(Secuencia de docker run) docker-entrypoint.sh apache2-foreground

4. ¿Cuál es el comando que se ejecuta al inicio cuando se lanza la imagen “Ubuntu”?

Para saber esto debe revisar en el archivo de configuración si hay un “CMD” o un “ENTRYPOINT” y así poder determinar que se ejecuta primero, por ejemplo, si hay un “ENTRYPOINT” lo que este contenga es lo primero que se ejecuta, de otra forma será un “CMD”.

```
# Set environment variables.
ENV HOME /root

# Define working directory.
WORKDIR /root

# Define default command.
CMD ["bash"]
```

Ilustración 100: Comando ejecutado al lanzar la imagen de "Ubuntu"



Una vez localizado el archivo y estando dentro del mismo, verifique los campos “CMD” y “ENTRYPOINT”, en este caso se ve que solo está el “CMD” y que ejecuta el comando por defecto “bash” (intérprete de comandos), así que “bash” es el primer comando que se ejecuta al poner a correr la imagen “ubuntu”.

5. Ejecute una instancia de la imagen ubuntu para ejecutar el comando “sleep 1000” al inicio.

Ejecutar en modo separado.

Este ejercicio se puede hacer de 2 formas, la primera es ir al dockerfile de la imagen “Ubuntu” y establecer en el punto de entrada (ENTRYPOINT) el valor “sleep” y en “CMD” se pasa el parámetro con valor 1000, este es un comando que suspende la ejecución actual del contenedor por un intervalo de tiempo igual a 1000.

La segunda es ingresar el comando directamente desde el “**docker run -d Ubuntu sleep 1000**”, como se ve a continuación:

```
$ docker run -d ubuntu sleep 1000
80dec549f58508ce134ae07fd36abecc57ca60fe15158f02914e0c42c14cba7b
$ █
```

Ilustración 101: Ejecución de una instancia de la imagen ubuntu + “sleep 1000”

Aquí se observa que se ejecuta el comando directamente desde la consola sin editar ningún archivo.

Guía de laboratorio 6 – Docker compose

Introducción

En esta guía se abordará el tema de Docker-compose que no es más que un archivo en texto plano estructurado que permite armar un contenedor desde los pies a la cabeza, los Docker-compose son una buena forma de ejecutar múltiples servicios en contenedores distintos entrelazándolos entre sí y hacerlo en un solo archivo sin



necesidad de crearlos uno a uno, la idea de esta guía es que el lector se familiarice con los Docker-compose, desde su estructura hasta su ejecución.

Duración estimada en la realización de la guía:

40 minutos.

Desarrollo de la guía

1) Cree un contenedor de base de datos “redis” llamado “redis”, cuya imagen es redis:alpine.

Utilice el comando “**docker run --name redis -d redis:alpine**” para ejecutar en segundo plano la imagen ligera “redis:alpine”, sino se encontró se descarga automáticamente, todo bajo el nombre del contenedor redis.

```
$ docker run --name redis -d redis:alpine
Unable to find image 'redis:alpine' locally
alpine: Pulling from library/redis
63b65145d645: Already exists
6a83e1b979d3: Pull complete
33568fda55fd: Pull complete
d7b8c87f98b0: Pull complete
a233cbd936e4: Pull complete
8182adae0173: Pull complete
Digest: sha256:eda49a8747271c820ee64ae71b360154af91ab54e91dc382b6f4fa3ff9621b05
Status: Downloaded newer image for redis:alpine
0d903013420c639cbb6ff8844a4a0644f3ad4284d5d671e326da806274ba18f8
$
```

Ilustración 102: Creación de contenedor redis

2) A continuación, cree un contenedor simple llamado “clickcounter” con la imagen “kodekloud/click-counter”, vinculado al contenedor “redis” que creado en el inciso anterior y luego exponerlo en el puerto del host 8085.

El contenedor clickcounter se ejecuta en el puerto 5000.



Esto se puede hacer de 2 maneras, la primera es hacerlo con un comando:

```
$ docker run -d --name=clickcounter --link redis:redis -p 8885:5000 kodekloud/click-counter
Unable to find image 'kodekloud/click-counter:latest' locally
latest: Pulling from kodekloud/click-counter
548db60ca938: Pull complete
a7ad1a75a999: Pull complete
37ce8546d5dd: Pull complete
ec9e91bed5a2: Pull complete
767433e10bb0: Pull complete
156f0b0493cb: Pull complete
3fe82d8a2401: Pull complete
4a41f7c94204: Pull complete
473063430a4f: Pull complete
452c68a16ccd: Pull complete
Digest: sha256:530e4532a718e8f5cbda05844a6c0638ebe8898fa4c4307ee6afbdd5d1f213db
Status: Downloaded newer image for kodekloud/click-counter:latest
73c3103107751a22074a44532de4cea8a5238a75975a451e5a51d5cf25937c13
$
```

Ilustración 103: Creación de contenedor llamado clickcounter vinculando el contenedor redis

Si revisa la documentación verá para qué sirven los enlaces y cómo es su sintaxis desde un “docker compose”, al realizar esto con un comando se usa --link, al crear un enlace, un contenedor puede compartir con otro ciertas funciones e información, en este caso redis y clickcounter, los demás campos del comando ya se conocen.

La segunda forma es lanzar el contenedor desde un “docker compose”, donde se especificuen todos los campos incluido los enlaces:

links:

-redis:redis

Esta sintaxis de enlaces debe estar dentro del archivo yml junto a los demás, recuerda que en el compose se definen los servicios a correr, así que si se desea ejecutar este archivo con 2 contenedores vinculados debe tener los enlaces apuntando al nombre del contenedor que quiere enlazar.

3) Ahora puede acceder a esta aplicación usando la pestaña Click-Counter encima de la terminal.

Actualice la página y vea cómo aumenta el conteo.



Ilustración 104: Aplicación clickcounter

Si le damos otra vez a “Click-Counter” la cantidad de clicks aumentara.



Ilustración 105: Aumentando el número de clicks

4) Limpiemos las acciones realizadas en los pasos anteriores. Eliminar los contenedores redis y el clickcounter.

Lo primero es localizar los “ID” de los contenedores para luego detenerlos con stop.



Ilustración 106: Identificar y detener contenedores



Ilustración 107: Deteniendo contenedor redis



Lo segundo será eliminarlos por sus mismos “ID”, y luego verificar que se han borrado.

```
$ docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS          PORTS
73c318318775   kodekloud/click-counter            "Flask run"             18 minutes ago  Exited (137) 2 minutes ago
clickcounter
0d983813420c   redis:alpine                       "docker-entrypoint.s..." 20 minutes ago  Exited (0) About a minute ago
redis
$ docker rm 73c3
73c3
$ docker rm 0
0
$
```

Ilustración 108: Identificando y eliminando contenedores redis y clickcounter

5) Crear un archivo “docker-compose.yml” bajo el directorio /root/clickcounter.

Una vez hecho esto, ejecuta docker-compose up.

El archivo debe tener la especificación exacta de la siguiente manera: “redis” especificación del servicio - el nombre de la imagen debe ser redis:alpine.

“Clickcounter” especificación del servicio - el nombre de la imagen debe ser “kodekloud/click-counter”, la aplicación se ejecuta en el puerto 5000 y exponerlo en el puerto host 8085 en el archivo de redacción.

Primero nos colocamos en el directorio “/root/clickcounter”:

```
$ pwd
/root/clickcounter
$
```

Ilustración 109: Verificación del directorio actual

Segundo procedemos a crear el archivo “docker-compose.yml” dentro de la carpeta “clickcounter” utilizando el siguiente comando “nano docker-compose.yml”, dentro del archivo recién creado colocaremos las siguientes líneas que aparecen en la captura, donde se utiliza la versión 3.0, por temas de compatibilidad entre los servicios tenemos a “redis” y “clickcounter” con sus imágenes y el puerto para “clickcounter”.



```
GNU nano 2.7.4 File: docker-compose.yml Modified
version: '3.0'
services:
  redis:
    image: redis:alpine
  clickcounter:
    image: kodekloud/click-counter
    ports:
      - 8085:5000
```

Ilustración 110: Creación de docker-compose de redis y clickcounter

Una vez creado el archivo se ejecuta en segundo plano con el siguiente comando **“docker-compose up -d”**.

```
$ docker-compose up -d
Creating network "clickcounter_default" with the default driver
Creating clickcounter_redis_1 ... done
Creating clickcounter_clickcounter_1 ... done
$
```

Ilustración 111: Ejecución del docker-compose

Por último, verifique si se crearon los contenedores. Como podrá ver los archivos “docker-compose” son muy útiles cuando se necesita trabajar con dos o más contenedores entrelazados o directamente ejecutar varios a la vez, ya que se hace de manera fácil y cómoda, sabiendo que docker crea una red interna y cada contenedor que se ejecuta, se conecta directamente en esta red, podemos enlazar a estos mediante enlaces, pero esto se verá más a fondo en la guía 8.

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
e64d260203e7      kodekloud/click-counter   "Flask run"        About a minute ago   Up About a minute   0.0.0.0:8085->5000/tcp   clickcounter_clickcounter_1
b67616bfbbcf      redis:alpine          "docker-entrypoint.s"   About a minute ago   Up About a minute   6379/tcp             clickcounter_redis_1
```

Ilustración 112: Verificación de la creación de los contenedores



Guía de laboratorio 7 – Almacenamiento de Docker

Introducción

En esta guía de laboratorio se abordarán aspectos relacionados con el almacenamiento de datos de Docker ya que la persistencia de datos de los contenedores es fundamental en ciertas aplicaciones, el objetivo principal de la presente guía es ayudar al estudiante a comprender los volúmenes de Docker, como funcionan y como puede implementar este mecanismo en los contenedores.

Duración estimada en la realización de la guía:

45 minutos

Desarrollo de la guía

1. ¿En qué ubicación se almacenan los archivos relacionados con los contenedores y las imágenes?

La ubicación en la que se almacenan dichos archivos es “**/var/lib/docker**”.

```
$ cd /var/lib/docker/  
$ ls  
builder buildkit containerd containers image network overlay2 plugins runtimes swarm tmp trust volumes  
$ █
```

Ilustración 113: Ubicación de contenedores e imágenes

2. ¿En qué directorio de /var/lib/docker se almacenan los archivos relacionados con la imagen del contenedor alpine-3?

El nombre del directorio es el mismo que el ID del contenedor, dicho directorio se encuentra disponible en la ubicación /var/lib/docker/containers/.

Ejecute el comando `docker ps -a` y haga coincidir la identificación del contenedor de alpine-3 con el nombre del directorio.



```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
1784ec52dc4c       alpine             "/bin/sh"          2 minutes ago      Exited (0) 2 minutes ago              alpine-3
f99b03c3e509       alpine             "/bin/sh"          2 minutes ago      Exited (0) 2 minutes ago              alpine-2
9b4bcb5f99cd       alpine             "/bin/sh"          2 minutes ago      Exited (0) 2 minutes ago              alpine-1
$ cd /var/lib/docker/container
-su: cd: /var/lib/docker/container: No such file or directory
$ cd /var/lib/docker/containers/
$ ls
1784ec52dc4cad5126ebb55d8fa41f1d0bd2eed4058b2ff40a3bcb0a7a9499c  f99b03c3e509bf9f93f46d865a331d2ea460664f4e343a6c45c2c6815b400021
9b4bcb5f99cd0c7be962cb9e7150b14bdc99627be948e6e812cad6951c2b0dac
$
```

Ilustración 114: Identificación de contenedor

Los archivos relacionados con la imagen del contenedor alpine-3 se almacenan en el directorio `/var/lib/docker/containers/1784ec52dc4cad5126ebb55d8fa41f1d0bd2eed4058b2ff40a3bcb0a7a9499c`.

3. Ejecute un contenedor mysql llamado mysql-db usando la imagen mysql. Establezca la contraseña de la base de datos en db_pass123

Nota: Recuerde ejecutarlo en el modo segundo plano (utilice la opción -d).

Ejecute el comando: `docker run -d --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 mysql`.

```
$ docker run -d --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 mysql
d083a00443a4d51d68b242d38cee38a3eb0476e0fda0e6e9ef830dab9edb4b2a
$
```

Ilustración 115: Establecer contraseña en la base de datos

4. Se ingresaron algunos datos en la base de datos. Para ver la información que se ingresó recientemente, ejecute el script get-data.sh disponible en el directorio /root. ¿Cuántos datos de clientes se han escrito en la base de datos?

Lo primero que debemos de hacer es ubicarnos en el directorio de "/root" seguido de eso ocuparemos el comando "sh get-data.sh" para ver la información que se ingresó.



```
$ cd /root/
$ sh get-data.sh
mysql: [Warning] Using a password on the command line interface can be insecure.
id      Name    Phone  Email
1       Kareem  130-5655    Duis.volutpat.nunc@quamCurabitur.org
2       Ruby    1-584-149-0770  Nulla.tempor@vitaeorciPhasellus.org
3       Rowan   199-8663    consectetuer.adipiscing.elit@Sedmalesuada.co.uk
4       Alisa   220-6017    elementum.sem.vitac@enimMauris.edu
5       Ella    731-0337    fermentum@nec.net
6       Tiger   658-4480    quis.diam@odiovelest.net
7       Felix   1-274-848-3378  Mauris.vel@arcu.com
8       Karina  1-390-796-3451  sagittis.semper@odioapuris.co.uk
9       Davis   605-8539    venenatis.vel@risusDonecnihb.com
10      Mohammad 1-590-174-1489  ornare.sagittis.felis@natoque.ca
11      Zane    362-1770    Aenean.euismod@condimentum.co.uk
12      Piper   1-231-386-6903  nunc.sed.pede@nascetur.ca
13      Marshall 1-383-729-4990  Cras.interdum.Nunc@neceuismod.ca
14      Zena    241-6641    Fusce.mollis.Duis@lobortis.org
15      Abdul   1-748-387-9935  eget.lacus.Mauris@Crasvehicula.com
16      Chase   1-401-241-9169  ante.dictum.mi@nascetur.org
17      Zahir   921-0663    nor@nonummyutmolestie.edu
18      Brenda  1-691-909-5827  Quisque.ac@magnaCras.co.uk
19      Laura   1-562-983-9565  Quisque.ornare.tortor@sollicitudinadipiscing.ca
20      Madison 1-348-737-0587  Quisque.varius@Intinciduntcongue.org
21      Tanek   991-6278    dignissim.magna@Pellentesqueutipsun.net
22      Dakota  893-0792    Nullam.enim.Sed@nulla.net
23      Boris   1-297-302-5792  non.sollicitudin@eleifendegestasSed.co.uk
24      Celeste 723-6729    mauris.rhonus@eunulla.edu
25      Connor 1-203-901-7531  et@loremipsumsodales.edu
26      Perry   1-756-607-9187  eros.turpis@tristiquepharetra.co.uk
27      Hayfa   1-609-407-3019  non.lobortis.quis@malesuadafringilla.net
28      Todd    343-0454    id.erat@arcu.org
29      Fuller  881-7273    non.feugiat.nec@adipiscingelit.net
30      Rama    1-927-605-0610  nonummy.ultrices.ornare@malesuada.co.uk
$
```

Ilustración 116: Ejecución de `sh get-data.sh` con datos

Se hizo el ingreso de 30 clientes.

5. La base de datos colapsó. ¿Puedes ver los datos ahora?

Usar el mismo comando para probar y obtener los datos. Tratar de encontrar el contenedor.

```
$ cd /root/
$ sh get-data.sh
Error: No such container: mysql-db
$
```

Ilustración 117: Ejecución de `sh get-data.sh` sin datos

Los datos de los clientes se perdieron.

6. Vuelva a ejecutar un contenedor mysql, pero esta vez asigne un volumen al contenedor para que los datos almacenados por el contenedor se almacenen en el directorio `/opt/data` en el host. Use el mismo nombre: `mysql-db` y la misma contraseña: `db_pass123` que antes. Mysql almacena datos en



/var/lib/mysql dentro del contenedor.

Ejecutar el comando: `docker run -v /opt/data:/var/lib/mysql -d --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 mysql`

```
$ docker run -v /opt/data:/var/lib/mysql -d --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 mysql
31d7a09b3cea7387a80ab6b82ac67073de1c3219073866adf4d83afe53b1d464
$
```

Ilustración 118: Asignación de volumen

7. Ahora se han escrito los datos nuevamente. Ejecute el script `get-data.sh` para asegurarse de que haya datos.

```
$ cd /root/
$ sh get-data.sh
mysql: [Warning] Using a password on the command line interface can be insecure.
id  Name  Phone  Email
1   Kareem 130-5655  Duis.volutpat.nunc@quamCurabitur.org
2   Ruby   1-584-149-0770  Nulla.tempor@vitaeorciPhasellus.org
3   Rowan  199-8663  consectetuer.adipiscing.elit@Sedmalesuada.co.uk
4   Alisa  220-6017  elementum.sem.vitae@enimMauris.edu
5   Ella   731-0337  fermentum@nec.net
6   Tiger  658-4480  quis.diam@odiovelest.net
7   Felix  1-274-848-3378  Mauris.vel@arcu.com
8   Karina 1-390-796-3451  sagittis.semper@odioapurus.co.uk
9   Davis  605-8539  venenatis.vel@risusDonecnibh.com
10  Mohammad  1-590-174-1489  ornare.sagittis.felis@natoque.ca
11  Zane   362-1770  Aenean.euismod@condimentum.co.uk
12  Piper  1-231-386-6903  nunc.sed.pede@nascetur.ca
13  Marshall  1-383-729-4990  Cras.interdum.Nunc@neceuismod.ca
14  Zena   241-6641  Fusce.mollis.Duis@lobortis.org
15  Abdul  1-748-387-9935  eget.lacus.Mauris@Crasvehicula.com
16  Chase  1-401-241-9169  ante.dictum.mi@nascetur.org
17  Zahir  921-0663  non@nonummyutmolestie.edu
18  Brenda 1-691-909-5827  Quisque.ac@magnaCras.co.uk
19  Laura  1-562-983-9565  Quisque.ornare.tortor@sollicitudinadipiscing.ca
20  Madison 1-348-737-0587  Quisque.varius@Intinciduntcongue.org
21  Tanek  991-6278  dignissim.magna@Pellentesquetipsum.net
22  Dakota 893-0792  Nullam.enim.Sed@nulla.net
23  Boris  1-297-302-5792  non.sollicitudin@eleifendegestasSed.co.uk
24  Celeste 723-6729  mauris.rhonus@eunulla.edu
25  Connor 1-203-901-7531  et@loremipsumsodales.edu
26  Perry  1-756-607-9187  eros.turpis@tristiquepharetra.co.uk
27  Hayfa  1-609-407-3019  non.lobortis.quis@malesuadafringilla.net
28  Todd   343-0454  id.erat@arcu.org
29  Fuller 881-7273  non.feugiat.nec@adipiscingelit.net
30  Rama   1-927-605-0610  nonummy.ultrices.ornare@malesuada.co.uk
$
```

Ilustración 119: Ejecución de `sh get-data.sh` con datos



- Ocurrió un error... ¡otra vez! y la base de datos volvió a fallar. Pero esta vez se tienen los datos almacenados en el directorio /opt/data. Vuelva a implementar una nueva instancia de mysql usando las mismas opciones que antes.

Aspectos a tener en cuenta:

- Configuración correcta de contraseña
- Anfitrión
- Contenedor: /var/lib/mysql

Simplemente ejecute el comando:

```
docker run -v /opt/data:/var/lib/mysql -d --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 mysql
```

```
$ docker run -v /opt/data:/var/lib/mysql -d --name mysql-db -e MYSQL_ROOT_PASSWORD=db_pass123 mysql
cF38330a311e959b99ea9a9bbf5a393e8d4bb8ddfe81dd19fc68a8f1a32eb18f
$ █
```

Ilustración 120: Ejecución de una nueva instancia de mysql

- Obtenga los datos y asegúrese de que estén presentes.

```
$ cd /root/
$ sh get-data.sh
mysql: [Warning] Using a password on the command line interface can be insecure.
id      Name      Phone    Email
1       Kareem   138-5655   Duis.volutpat.nunc@quamCurabitur.org
2       Ruby     1-584-149-0778   Nulla.tempor@vitaeorciPhasellus.org
3       Rowan    199-8663   consectetur.adipiscing.elit@Sodalesuada.co.uk
4       Alisa    220-6817   elementum.sem.vitae@erisMauris.edu
5       Ella     731-8337   fermentum@nec.net
6       Tiger    658-4488   quis.diam@odionolest.net
7       Felix    1-274-848-3378   Mauris.vel@arcu.com
8       Karina   1-390-796-3451   sagittis.semper@odioapurus.co.uk
9       Davis    685-8539   venenatis.vel@visusDonecridibb.com
10      Mohammad 1-590-174-1489   ornare.sagittis.felis@hatoque.ca
11      Zane     362-1778   Aenean.euismod@condimentum.co.uk
12      Piper    1-231-386-6983   nunc.sed.pede@nascetur.ca
13      Marshall 1-383-729-4990   Cras.interdum.nunc@neceismod.ca
14      Zena     241-6641   Fusce.mollis.Duis@lobortis.org
15      Abdul    1-748-387-9935   eget.lacus.Mauris@Crasvehicula.com
16      Chase    1-481-241-9169   ante.dictum.mj@nascetur.org
17      Zahir    921-8663   non@nonummytolestie.edu
18      Brenda   1-691-909-5827   Quisque.ac@sagnaCras.co.uk
19      Laura    1-562-983-9565   Quisque.ornare.tortor@sollicitudinadipiscing.ca
20      Madison  1-348-737-0587   Quisque.varius@Intinciduntcongue.org
21      Tanek    991-6278   dignissim.magnis@Pellentesqueutipsum.net
22      Dakota   893-8792   Nullam.enim.Sed@nulla.net
23      Boris    1-297-302-5792   non.sollicitudin@eleifendegestasSed.co.uk
24      Celeste  723-6729   mauris.rhonus@eumulla.edu
25      Connor   1-283-981-7531   at@lorempisusodales.edu
26      Perry    1-756-687-9187   eros.turpis@tristiquepharetra.co.uk
27      Hayfa    1-689-487-3819   non.lobortis.quis@alesuadafringilla.net
28      Todd     343-8454   id.erat@arcu.org
29      Fuller   881-7273   non.feugiat.nec@odipiscingelit.net
30      Rama     1-927-685-8610   nonummy.ultrices.ornare@salesuada.co.uk
$ █
```

Ilustración 121: Verificación de persistencia de datos



Guía de laboratorio 8 – Redes Docker

Introducción

En esta guía se abordarán las redes Docker que son mecanismos de comunicación entre contenedores que pertenecen a las mismas redes utilizando una Ip distinta a la del servidor, los contenedores pueden pertenecer a cualquiera de las 3 redes predefinidas que son “bridge”, “host” y “none”, para más información puede revisar la información de este mismo documento, el objetivo principal de esta guía es en esencia que el estudiante pueda familiarizarse con las redes Docker, su funcionamiento y el establecimiento en los contenedores.

Duración estimada en la realización de la guía:

45 minutos

Desarrollo de la guía

1. Explore la configuración actual e identifique la cantidad de redes que existen en este sistema.

Ejecutar el comando “docker network ls” para verificar las redes existentes en este sistema, donde nos dice que existe 3 redes en el sistema.

```
$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
66b15e128ec9       bridge             bridge              local
ad8000dae012       host               host                local
a7022101d85d       none               null                local
$
```

Ilustración 122: Comprobación de redes existentes

2. ¿Cuál es el ID asociado con la red del “bridge”?

Utilizamos el comando “docker network ls” y vemos que el Id de la red bridge es: “66b115e128ec9”.



```
$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
66b15e128ec9       bridge             bridge             local
ad8000dae012       host              host              local
a7022101d85d       none              null              local
$
```

Ilustración 123: Verificación del ID asociado a la red bridge

3. Se ejecutó un contenedor llamado alpine-1. Identifique la red a la que está conectado.

Verifique que el contenedor este en ejecución utilizando el comando “docker ps -a”.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
832e4c3cf5aa	alpine	"sleep 1000"	2 minutes ago	Up 2 minutes		alpine-1

```
$
```

Ilustración 124: Verificación de creación y ejecución del contenedor alpine-1

Una vez verificado use el comando “**docker inspect alpine-1 | grep “NetworkMode”**” para revisar la red en la que está conectado.

```
$ docker inspect alpine-1 | grep "NetworkMode"
"NetworkMode": "host",
$
```

Ilustración 125: Identificación de la red del contenedor alpine-1

El contenedor llamado alpine-1 esta conectado a la red “host”.

4. ¿Cuál es la subred configurada en la red “bridge”

Ejecutar el comando “**docker network inspect bridge | grep “Subnet”**”, el cual devolverá el resultado de la información de la red “bridge”, obtenida esta información con grep se toma solo el campo “Subnet”, esto devolverá la subred.



```
Terminal 1 +
$ docker network inspect bridge | grep "Subnet"
"Subnet": "172.12.0.0/24",
$
```

Ilustración 126: Inspeccionando la subred configurada en la red bridge

5. Ejecutar un contenedor llamado “alpine-2” utilizando la imagen “alpine” y adjuntarla a la red “none”.

Ejecutar el comando “**docker run --name alpine-2 --network=none alpine**” para configurar en el contenedor la red por defecto “none”.

Ejecute el comando y luego verifique si el contenedor está en ejecución.

```
$ docker run --name alpine-2 --network=none alpine
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
0cb7b2619395      alpine             "/bin/sh"          4 seconds ago      Exited (0) 3 seconds ago
a32e9c3c13aa      alpine             "sleep 1000"       22 minutes ago     Exited (0) 5 minutes ago
$
```

Ilustración 127; Creación de contenedor alpine-2 | con imagen alpine | red none

Luego ejecutar el comando “**docker inspect alpine-2 | grep "NetworkMode"**” para verificar que la red del contenedor se ha establecido en “none”.

```
$ docker inspect alpine-2 | grep "NetworkMode"
"NetworkMode": "none",
$
```

Ilustración 128: Verificación de red none

6. Crear una red llamada “wp-mysql-network” utilizando el controlador “bridge”, asigna la subred 182.18.0.0/24 y configurar Gateway 182.18.0.1.

Ejecutar el comando “**docker network create --driver bridge --subnet 182.18.0.0/24 --gateway 182.18.0.1 wp-mysql-network**”.



```
Terminal 1 +
$ docker network create --driver bridge --subnet 182.18.0.1/24 --gateway 182.18.0.1 wp-mysql-network
d43f30c91b5aa6293c75ddd2f52492852ba2faa727b6e84edbad7705dd31fe46
```

Ilustración 129: Creación de red wp-mysql-network

Puede verificar la creación de la red ejecutando el comando “docker network ls”

```
$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
66b15e128ec9       bridge             bridge             local
ad8000dae012       host              host              local
a7022101d85d       none              null              local
d43f30c91b5a       wp-mysql-network  bridge            local
```

Ilustración 130: Verificación de creación de red wp-mysql-network

7. Desplegar una base de datos “mysql” usando la imagen mysql:5.6 y de nombre “mysql-db”. Adjuntarlo a la red creada en el apartado anterior “wp-mysql-network”.

Establecer la contraseña de la base de datos para el usuario root en “db_pass123” utilizando la variable de entorno MYSQL_ROOT_PASSWORD.

Ejecute el comando “**docker run -d -e MYSQL_ROOT_PASSWORD=db_pass123 --name mysql-db --network wp-mysql-network mysql:5.6**”, el cual permite ejecutar en segundo plano el contenedor de nombre “mysql-db” cuya imagen es “mysql:5.6”, establecer la variable de entorno que se utiliza para definir la contraseña del usuario root a “db_pass123” y la red por defecto que se le configura es la wp-mysql-network creada anteriormente.



```
$ docker run -d -e MYSQL_ROOT_PASSWORD=db_pass123 --name mysql-db --network wp-mysql-network mysql:5.6
Unable to find image 'mysql:5.6' locally
5.6: Pulling from library/mysql
35b2232c987e: Pull complete
fc55c00e48f2: Pull complete
0030405130e3: Pull complete
e1fef7f6a8d1: Pull complete
1c76272398bb: Pull complete
f57e698171b6: Pull complete
f5b825b269c0: Pull complete
dcb0af686073: Pull complete
27bbfeb886d1: Pull complete
6f70cc868145: Pull complete
1f6637f4600d: Pull complete
Digest: sha256:20575e3e3e6216036d25dab5903808211f1e9ba63dc7825ac20cb975e34cfcae
Status: Downloaded newer image for mysql:5.6
eb23b991ca0f78eced2c66c0589b45e51eee95bae2e208e89f90543ae39586e0
$
```

Ilustración 131: Creación de un nuevo contenedor mysql-db

Ejecute el comando **“docker inspect mysql-db | grep NetworkMode”** para verificar que el contenedor usa la red recién creada.

```
$ docker inspect mysql-db | grep "NetworkMode"
"NetworkMode": "wp-mysql-network",
$
```

Ilustración 132: Inspección de red del contenedor

8. Desplegar una aplicación web denominada “webapp” utilizando el “kodekloud/simple-webapp-mysql” como imagen. Exponga el puerto a 38080 en el host y 8080 en el contenedor.

La aplicación hace uso de dos variables de entorno:

- 1: DB_Host con el valor mysql-db.
- 2: DB_Password con el valor db_pass123.

Asegurarse de adjuntarlo a la red recién creada llamada wp-mysql.network y vincular el MySQL y el contenedor webapp.

Ejecute el comando **“docker run --network=wp-mysql-network -e DB_Host=mysql-db -e DB_Password=db_pass123 -p 38080:8080 --name webapp --link mysql-db:mysql-db -d kodekloud/simple-webapp-mysql”**, esto



Creación de contenedores para CMS

Para la creación de cada uno de los contenedores, usaremos el componente docker-compose, ya que, debido a que en cada contenedor se usará más de un servicio y será necesario de esta característica de Docker para levantar cada CMS.

Entre las partes importantes a destacar tenemos las siguientes secciones:

*Versión (**obligatorio**)*: esta referencia a la versión de sintaxis del compose, y está asociada a su vez a la versión de Docker. Si no se indica una versión se trataría de la versión 1 (la cual se encuentra en desuso).

*Services(**obligatorio**)*: esta sección nos sirve para configurar todos nuestros contenedores partiendo de una imagen base además de especificar variables de entorno de dicho servicio con el fin de mejorar la configuración de este (a como se puede ver reflejado en la sección **ENVIRONMENT**). El fichero debe incluir al menos un servicio.

*Volumes(**Opcional**)*: En esta sección definimos la persistencia de datos de nuestros contenedores (de la cual existen diferentes maneras de hacerlo).

En este caso particular utilizamos un método de volumen conocido como volumen Anónimos (por lo general, es más fácil gestionar y almacenar volúmenes debido a que carecen de un identificador legible por humanos).



el caso de phpmyadmin, se declara cual será en ambiente de base de datos del cual va a depender, así declaramos que es de nuestra base de datos creada.

Entre otras secciones a tener en cuenta tenemos *depends on* que sirve para indicar la dependencia entre los contenedores (no se va a iniciar un contenedor hasta que otro se encuentre en funcionamiento, en este caso el contenedor de “web” depende de que el contenedor “db” se encuentre previamente en funcionamiento) y la sección de *ports* cuya función es definir que puertos se va a utilizar para conectarse a ese contenedor (su sintaxis es “puerto_maquina_anfitrión” : “puerto_del_contenedor”).

Joomla

```
~/DOCKER/Joomla/docker-compose.yml - Mousepad
File Edit Search View Document Help
1 version: '3.3'
2
3 services:
4   db:
5     image: mysql:5.7
6     volumes:
7       - ./db_data:/var/lib/mysql
8     environment:
9       MYSQL_ROOT_PASSWORD: joomla
10      MYSQL_DATABASE: joomla
11      MYSQL_USER: joomla
12      MYSQL_PASSWORD: joomla
13
14   pma:
15     image: phpmyadmin/phpmyadmin #v4.6.4-1
16     links:
17       - db
18     ports:
19       - "82:80"
20     environment:
21       - PMA_HOST=db
22
23   web:
24     depends_on:
25       - db
26
27     image: joomla:3.6.2-apache
28     ports:
29       - 8001:80
30     environment:
31       JOOMLA_DB_HOST: db
32       JOOMLA_DB_USER: joomla
33       JOOMLA_DB_PASSWORD: joomla
34       JOOMLA_DB_NAME: joomla
35     volumes:
36       - ./web_data:/var/www/html
37 volumes:
38   db_data:
39   web_data:
40
```

Ilustración 137: docker-compose.yml de Joomla



Se muestra el fichero Docker-compose.yml encargado de la creación del contenedor de Joomla

Imágenes:

- web: joomla:3.6.2_apache
- db: mysql:5.7
- PhPMyAdmin: phpmyadmin/phpmyadmin

Servicios de **web** y **db** se declaran atributos en el apartado de los environments, con el objetivo de definir las variables necesarias para la creación tanto de la base de datos, así como la definición de las variables de WordPress, para la creación del mismo. En el caso de phpmyadmin, se declara cual será en ambiente de base de datos del cual va a depender, así declaramos que es de nuestra base de datos creada.

Secciones a tener en cuenta, tenemos *depends on* que sirve para indicar la dependencia entre los contenedores (el contenedor de “web” depende de que el contenedor “db” se encuentre previamente en funcionamiento) y la sección de *ports* cuya función es definir que puertos se va a utilizar para conectarse a ese contenedor (su sintaxis es “puerto_maquina_anfitrión” : “puerto_del_contenedor”).



Drupal

```
~|DOCKER/Drupal/docker-compose.yml - Mousepad
File Edit Search View Document Help
1 version: '3.3'
2
3 services:
4   web:
5     depends_on:
6       - db
7     image: drupal:9.5.10
8     ports:
9       - 8082:80
10    volumes:
11      - ./drupal-9.5.10:/var/www/html
12      #- ./web_data:/var/www/html
13    restart: always
14
15   db:
16     image: mysql:5.7.8
17     volumes:
18       - ./db_data:/var/lib/mysql
19     environment:
20       MYSQL_ROOT_PASSWORD: drupal|
21       MYSQL_DATABASE: drupal
22       MYSQL_USER: drupal
23       MYSQL_PASSWORD: drupal
24
25   pma:
26     image: phpmyadmin/phpmyadmin #:4.6.4-1
27     links:
28       - db
29     ports:
30       - "83:80"
31     environment:
32       - PMA_HOST=db
33
34 volumes:
35   drupal_modules:
36   drupal_profiles:
37   drupal_themes:
38   drupal_sites:
39   db_data:
40
```

Ilustración 138: docker-compose.yml de Drupal

Se muestra el fichero Docker-compose.yml encargado de la creación del contenedor de Drupal

Imágenes:

- web: drupal:10
- db: mysql:5.7.8
- PhPMyAdmin: phpmyadmin/phpmyadmin

Servicios de **db** se declaran atributos en el apartado de los environments, con el objetivo de definir las variables necesarias para la creación de la base de datos. En el caso de phpmyadmin, se declara cual será en ambiente de base de datos del cual va a depender, así declaramos que es de nuestra base de datos creada.



Secciones a tener en cuenta, tenemos *depends on* que sirve para indicar la dependencia entre los contenedores (el contenedor de “web” depende de que el contenedor “db” se encuentre previamente en funcionamiento) y la sección de *ports* cuya función es definir que puertos se va a utilizar para conectarse a ese contenedor (su sintaxis es “puerto_maquina_anfitrión” : “puerto_del_contenedor”).

PrestaShop

```
-/DOCKER/Prestashop/docker-compose.yml - Mousepad
File Edit Search View Document Help
1 version: "3"
2 services:
3   db:
4     image: mysql:5.7
5     restart: unless-stopped
6     environment:
7       MYSQL_ROOT_PASSWORD: Password
8       MYSQL_DATABASE: prestashop
9       MYSQL_USER: ps_user
10      MYSQL_PASSWORD: Password
11     volumes:
12       - ./db_data:/var/lib/mysql
13
14   pma:
15     image: phpyadmin/phpmyadmin #/4.6.4-1
16     links:
17       - db
18     ports:
19       - "84:80"
20     environment:
21       - PMA_HOST=db
22
23   web:
24     image: prestashop/prestashop:latest
25     restart: unless-stopped
26     depends_on:
27       - db
28     ports:
29       - 8083:80
30     environment:
31       PRESTASHOP_DB_SERVER: mysql
32       PRESTASHOP_DB_NAME: prestashop
33       PRESTASHOP_DB_USER: ps_user
34       PRESTASHOP_DB_PASSWORD: Password
35     volumes:
36       - ./web_data:/var/www/html
37
38 volumes:
39   web_data: {}
40   db_data: {}
41
```

Ilustración 139: docker-compose.yml de PrestaShop

Se muestra el fichero Docker-compose.yml encargado de la creación del contenedor de PrestaShop

Imágenes:



- web: prestashop/prestashop:latest
- db: mysql:5.7
- PhPMyAdmin: phpmyadmin/phpmyadmin

Servicios de **web** y **db** se declaran atributos en el apartado de los environments, con el objetivo de definir las variables necesarias para la creación tanto de la base de datos, así como la definición de las variables de WordPress para la creación del mismo. En el caso de phpmyadmin, se declara cual será en ambiente de base de datos del cual va a depender, así declaramos que es de nuestra base de datos creada.

Secciones a tener en cuenta, tenemos *depends on* que sirve para indicar la dependencia entre los contenedores (el contenedor de “web” depende de que el contenedor “db” se encuentre previamente en funcionamiento) y la sección de *ports* cuya función es definir que puertos se va a utilizar para conectarse a ese contenedor (su sintaxis es “puerto_maquina_anfitrión” : “puerto_del_contenedor”)



Moodle

```
~/DOCKER/Moodle/docker-compose.yml - Mousepad
File Edit Search View Document Help
1 version: '2'
2 services:
3   mariadb:
4     image: docker.io/bitnami/mariadb:10.6
5     environment:
6       # ALLOW_EMPTY_PASSWORD is recommended only for development.
7       - ALLOW_EMPTY_PASSWORD=yes
8       - MARIADB_USER=bn_moodle
9       - MARIADB_DATABASE=bitnami_moodle
10      - MARIADB_CHARACTER_SET=utf8mb4
11      - MARIADB_COLLATE=utf8mb4_unicode_ci
12     volumes:
13       - ./mariadb_data:/var/lib/mariadb
14
15   moodle:
16     image: docker.io/bitnami/moodle:4
17     ports:
18       - '80:8080'
19       - '443:8443'
20     environment:
21       - MOODLE_DATABASE_HOST=mariadb
22       - MOODLE_DATABASE_PORT_NUMBER=3306
23       - MOODLE_DATABASE_USER=bn_moodle
24       - MOODLE_DATABASE_NAME=bitnami_moodle
25       # ALLOW_EMPTY_PASSWORD is recommended only for development.
26       - ALLOW_EMPTY_PASSWORD=yes
27     volumes:
28       - ./moodle_data:/var/www/html/moodle
29       - ./moodledata_data:/var/www/html/moodledata
30     depends_on:
31       - mariadb
32
33 volumes:
34   mariadb_data:
35     driver: local
36   moodle_data:
37     driver: local
38   moodledata_data:
39     driver: local
40
```

Ilustración 140: docker-compose.yml de Moodle

Se mostrará y explicará el archivo docker-compose.yml, encargado de crear y ejecutar el CMS de Moodle.

Como primera instancia, tenemos la versión que se ejecuta de nuestro archivo .yml.

En el apartado de los servicios, en este caso tenemos 2 servicios:



- Base de datos (mariadb): encargado de la creación y ejecución del contenedor, donde se alojará la base de datos de Moodle, en este caso el motor de base de datos es MariaDB.

En lo que al servicio se refiere, indicamos primeramente el nombre que tendrá este servicio:

- Image: tenemos lo que es la imagen que se utilizará **docker.io/bitnami/mariadb:10.6**.
 - Environment: en el apartado de los environment, lo principal a destacar la línea **ALLOW_EMPTY_PASSWORD=yes**, en la cual estamos definiendo que este contenedor no será obligatorio una contraseña y un usuario, asimismo, tenemos las demás variables de entornos, donde definimos el usuario, la base de datos y donde se especifica el set de caracteres que vamos a utilizar en la base de datos junto con el cotejamiento.
 - Volumes: se utiliza el punto de montaje, vemos que estas carpetas se crean en el directorio actual (donde se encuentra el archivo .yaml)
- CMS (moodle): se crea y ejecuta el contenedor del CMS en este caso, Moodle, de igual forma se definen ciertos parámetros para el correcto funcionamiento de este:
 - Image: se utilizará la imagen de Moodle en la siguiente versión **docker.io/bitnami/Moodle:4**.
 - Ports: se definen los puertos a usar para acceder desde la web a Moodle, en este caso se definen los puertos 80:8080 y 43:8443.
 - Environment: en las definiciones de los environments, tenemos, el host, donde especificamos el nombre del servicio de la base de datos, el puerto de la base de datos, definimos un usuario, también, se define el nombre de la base de datos y por último, se define lo que es no usar un usuario para este contenedor de igual forma.
 - Volumes: se definen los puntos de montajes de los volúmenes para el alojamiento de la información.



- Depends_on: Se especifica que la información del contenedor depende de la base de datos de MariaDB, sin este parámetro no habría conexión con el servicio de base de datos.
- Volumes: en esta sección, se especifican los nombres de los volúmenes creados en los servicios anteriores, definiendo en cada uno el driver localmente, este para que sea alojado en la máquina física.

Resaltar que a diferencia de los demás, que la imagen de la base de datos y del CMS, son descargados de la página oficial de Bitnami, donde en la imagen se han declarado de forma predeterminada las credenciales del usuario administrador, en el login de Moodle.

También nos proporciona la facilidad de no tener que realizar el proceso de instalación de Moodle, ya que, de igual forma Bitnami nos ha proporcionado la imagen lista. Solo para empezar con la creación del sitio. (Docker I. , 2023)

Ejemplos de guías prácticas

WordPress

Instalación de WordPress

Instalación básica:

Como primer paso debemos escribir en la URL localhost:8000, ya que estaremos trabajando WordPress en ese puerto. Luego debemos elegir el lenguaje que deseamos, elegimos español:

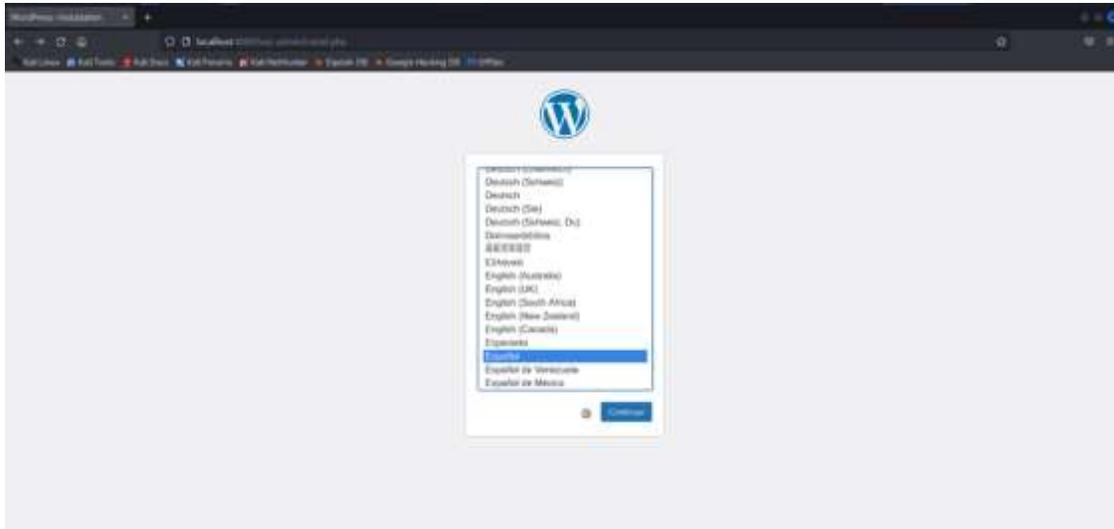


Ilustración 141: Selección de lenguaje

Luego en la siguiente página que se nos muestra, debemos rellenar la información que se nos pide de nuestro sitio: como nombre del sitio, nombre de usuario, contraseña y confirmar contraseña, un correo electrónico (puede ser ficticio):

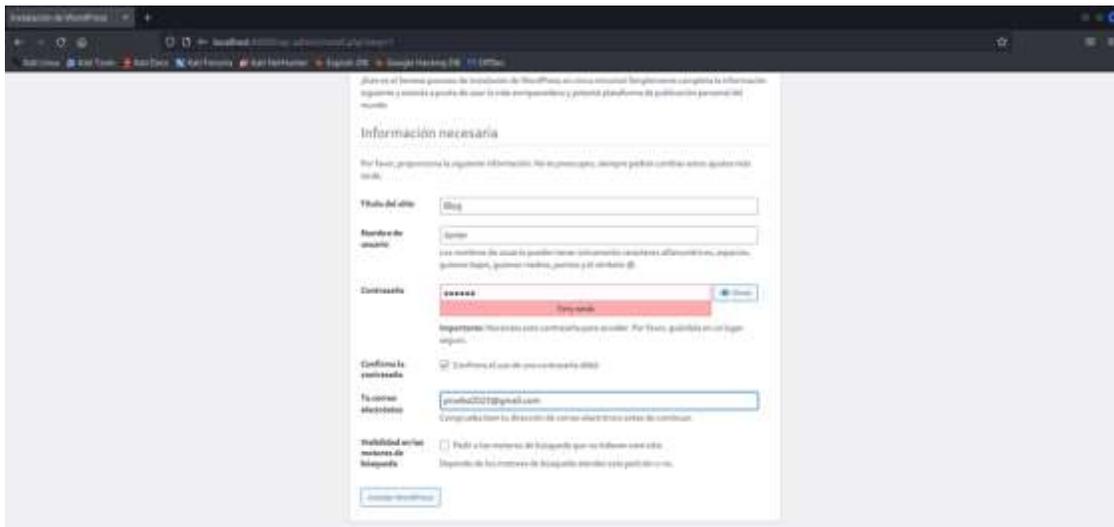


Ilustración 142: Definición de datos

Y con esto ya habremos configurado nuestro sitio:



Ilustración 143: Sitio creado

Acceder como administrador

Damos clic en acceder y se nos redirigirá al login de WordPress (o escribimos en la URL localhost:8000/wp-admin), aquí usaremos el usuario y contraseña que creamos anteriormente

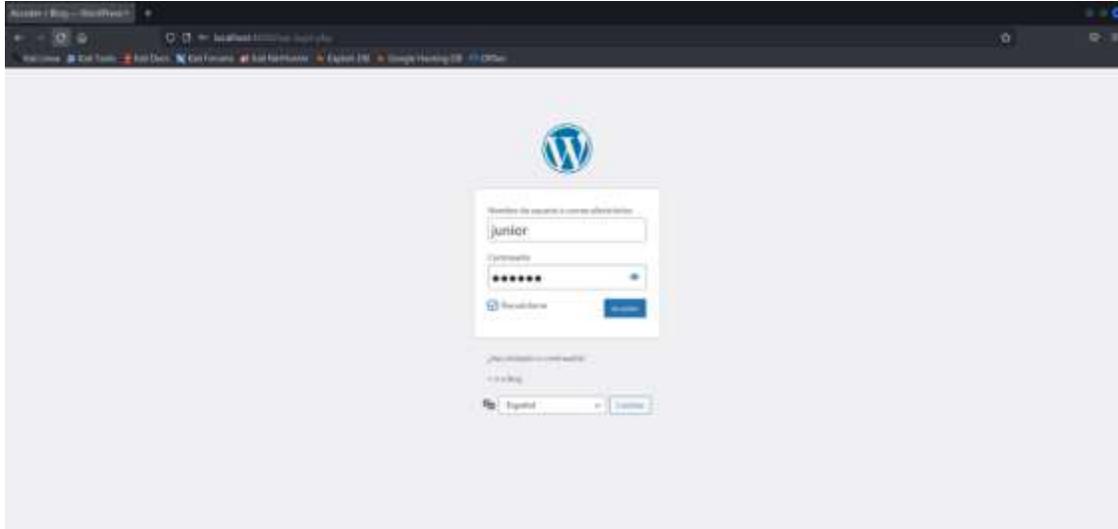


Ilustración 144: Login

Y tendríamos acceso al administrador de nuestra página de WordPress:

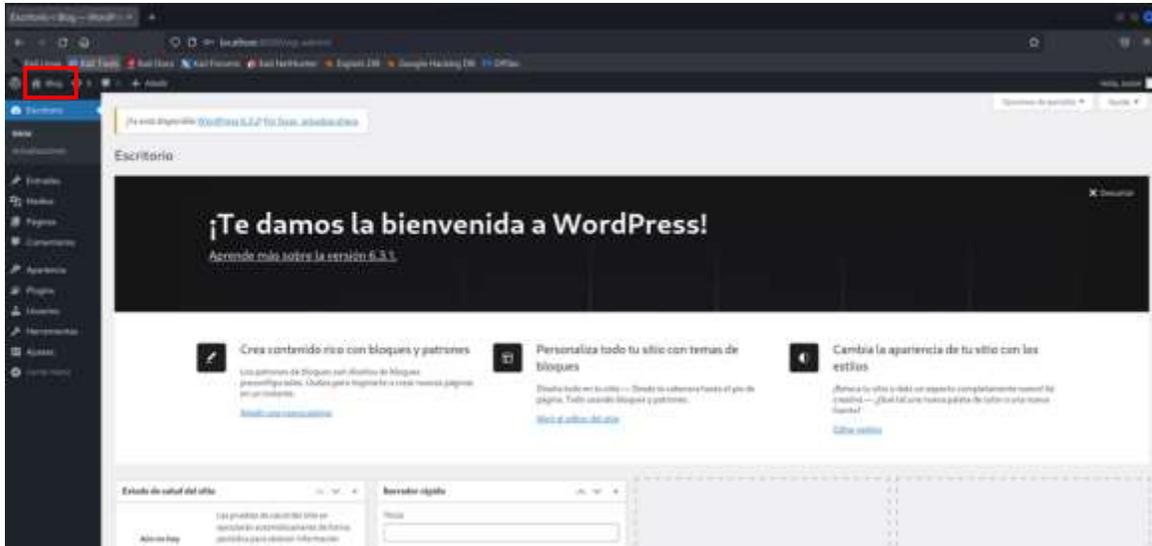


Ilustración 145: Administración

Luego en la esquina superior derecha damos clic en el nombre de nuestro sitio para visualizar la vista que tendría un cliente y veremos la plantilla por defecto que trae WordPress:

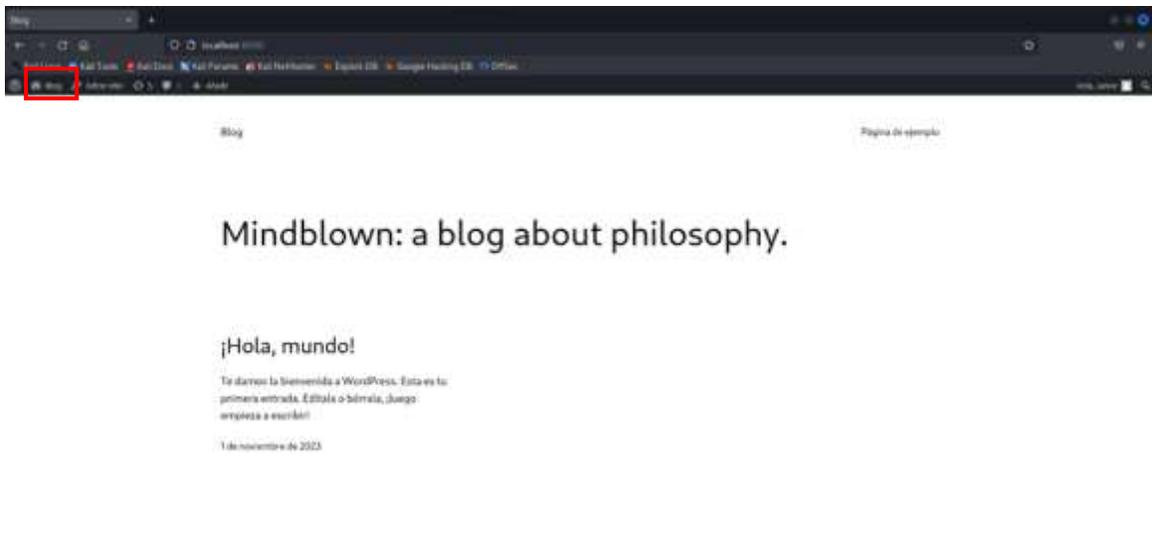


Ilustración 146: Index

Para volver solo damos clic al icono de WordPress.



Actualizar

En caso de que haya elementos actualizables (la aplicación, los plugins, los temas o las traducciones), en el Escritorio de WordPress se muestra el aviso de actualización disponible. El número que se muestra en el menú indica el número de actualizaciones disponibles y si está disponible una nueva versión de WordPress y todavía no se ha actualizado automáticamente, un aviso lo indica:

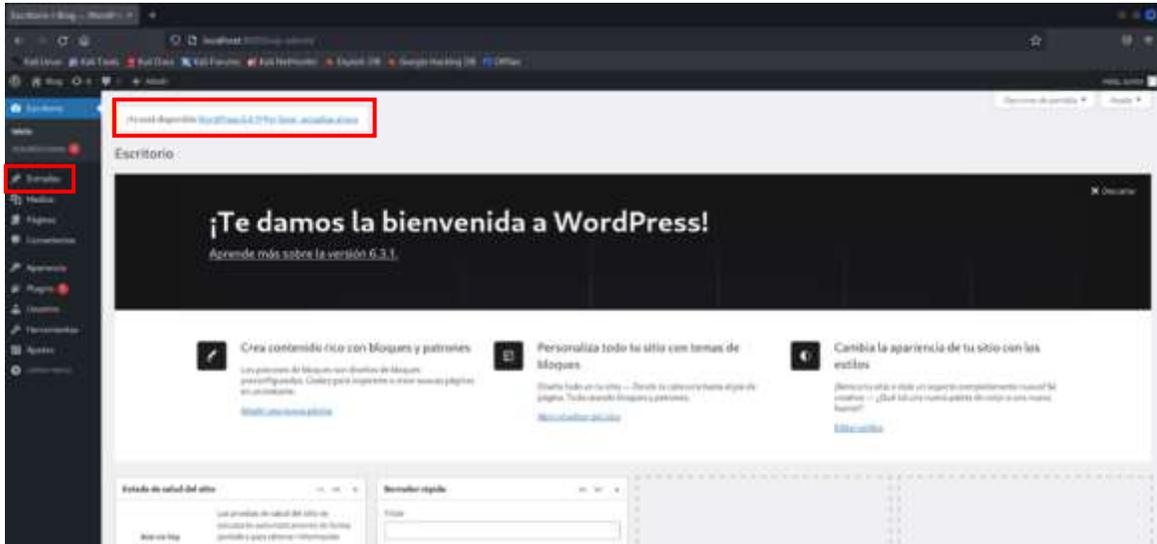


Ilustración 147: Actualización

Hacemos clic en la pestaña de "Actualizaciones" para ver las actualizaciones disponibles, si hay una actualización de la aplicación, esta se muestra en primer lugar. Cuando se trata de una actualización menor (el tercer número en el número de versión), la actualización se realiza automáticamente, por lo que a veces no llegamos a verla. Podemos pedir a WordPress que se actualice en ese momento haciendo clic en el botón "Actualizar", así como también las actualizaciones para los plugins y temas.

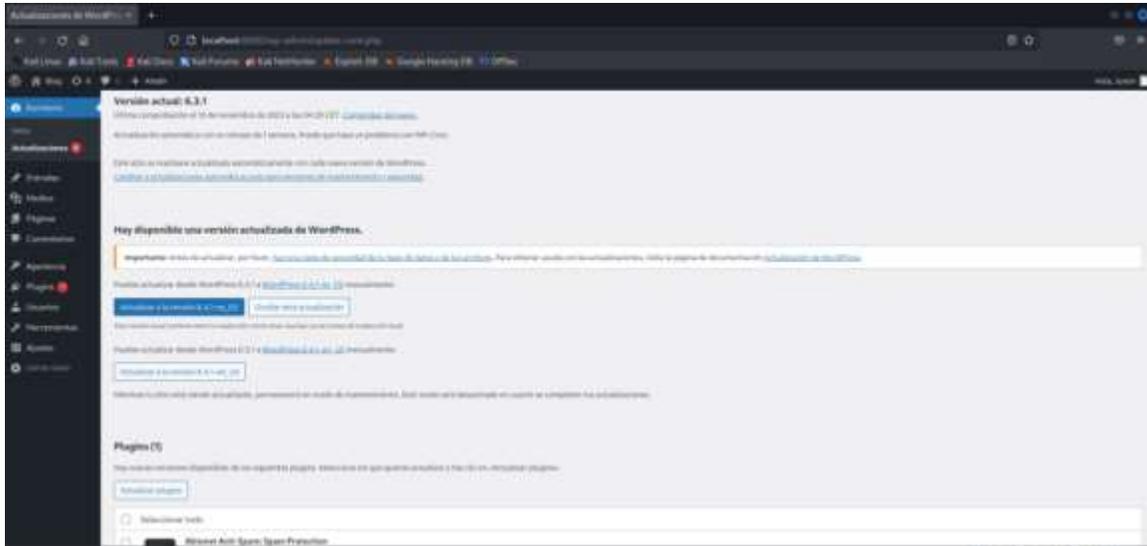


Ilustración 148: Instalar actualizaciones

Realizar copia de seguridad manual completa:

Al realizar la copia de seguridad de la base de datos utilizamos la herramienta de phpMyAdmin ya que nos provee una interfaz gráfica.

Para acceder a phpMyAdmin abrimos una nueva pestaña y colocamos en la URL localhost:81 que 81 es el número de puerto en el que está corriendo:



Ilustración 149: Login phpMyAdmin



Luego colocamos las credenciales. Para poder tener todo control y permisos de la base de datos usaremos el usuario “root” y la contraseña que especificamos en el archivo yml que es “wordpress” e iniciamos sesión:

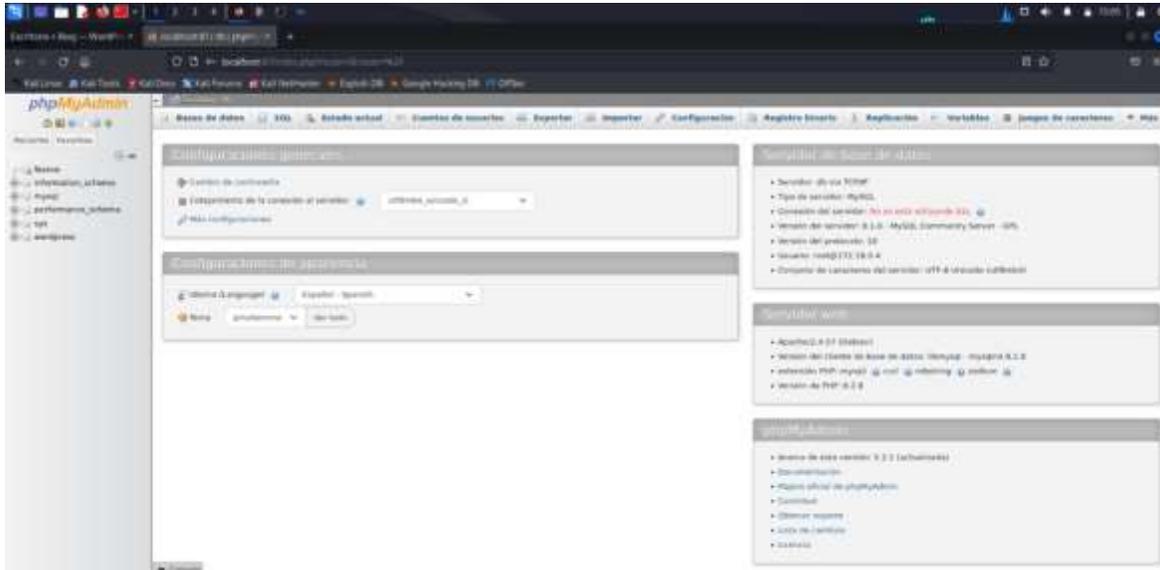


Ilustración 150: Página de inicio phpMyAdmin

Seleccionamos la base de datos de wordpress:

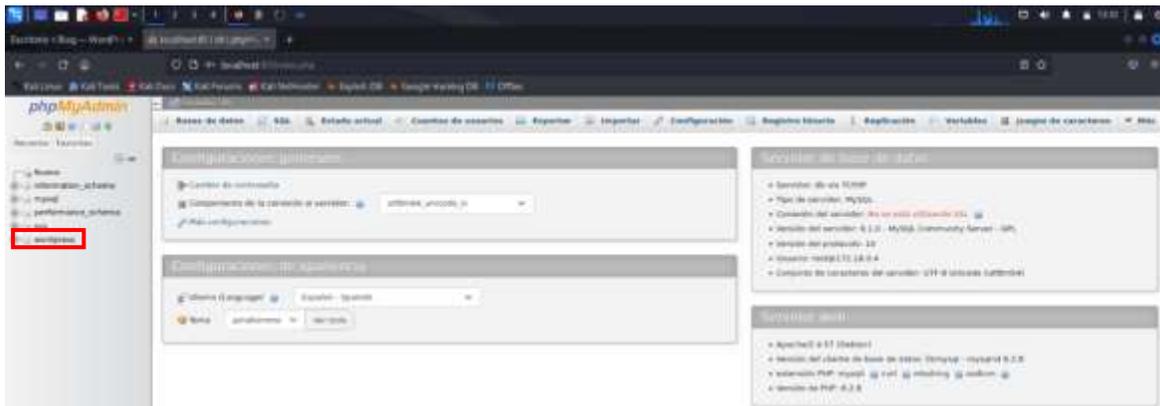


Ilustración 151: Selección base de datos WordPress



Damos clic a exportar:

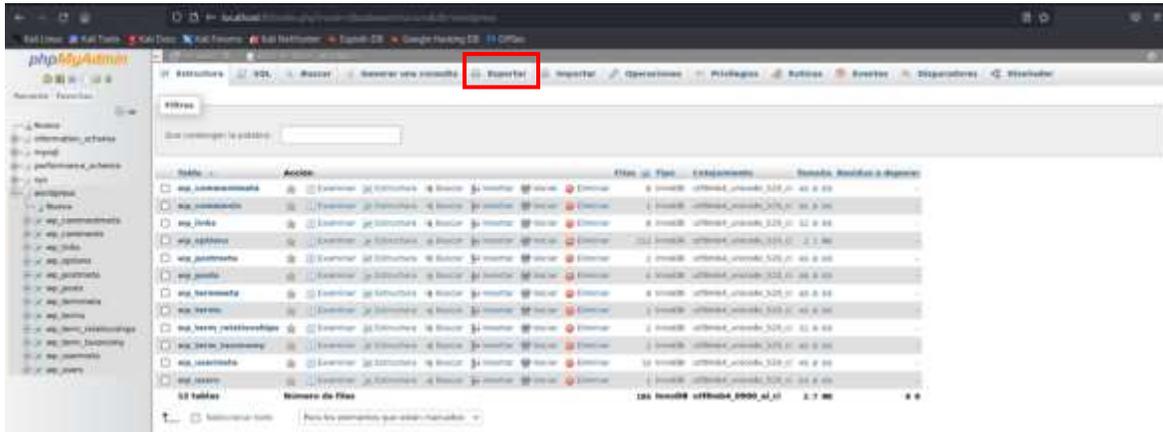


Ilustración 152: Tablas de base de datos WordPress

Dejamos las opciones por defecto y damos clic en exportar:

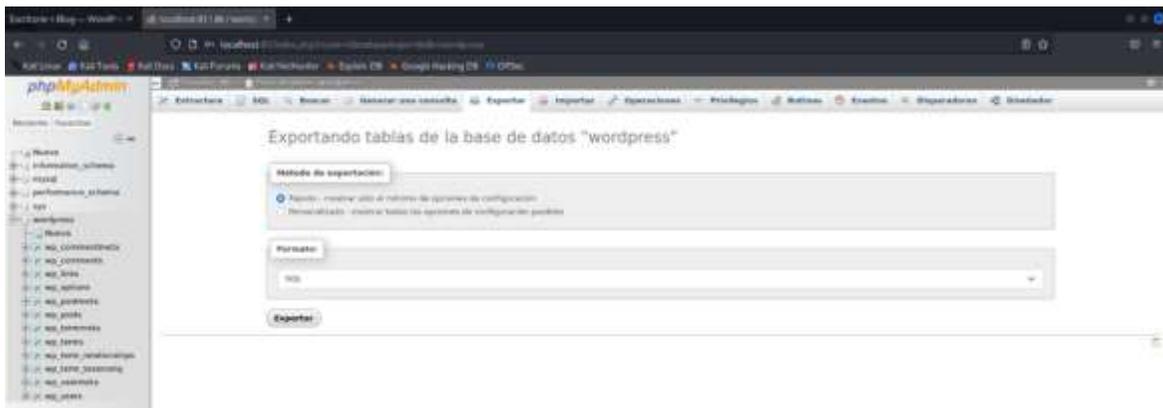


Ilustración 153: Exportar base de datos

Luego se procederá a descargar el backup del sitio:

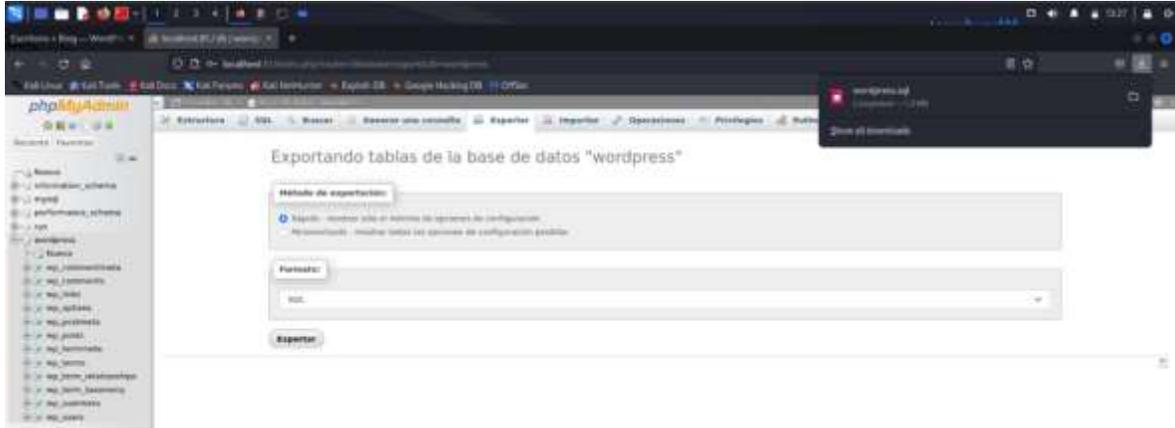


Ilustración 154: Backup exportado

Este archivo se encontrará en la carpeta de descargas:



Ilustración 155: Ubicación de backup

Para verificar que la copia de seguridad función correctamente procederemos a borrar la base de datos de worpress, nos dirigimos el menú principal de phpMyAdmin y seleccionamos bases de datos:

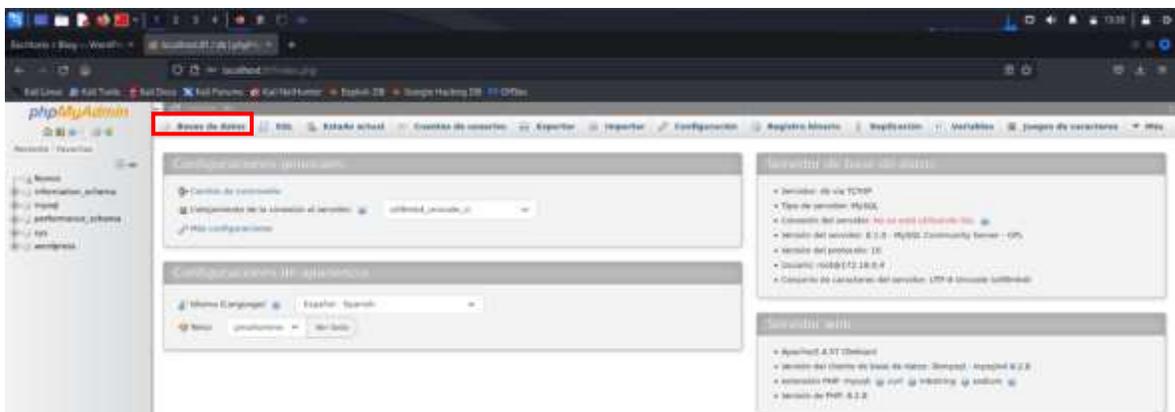


Ilustración 156: Selección de base de datos WordPress a eliminar



Luego seleccionamos la base de datos de wordpress y le damos en eliminar:

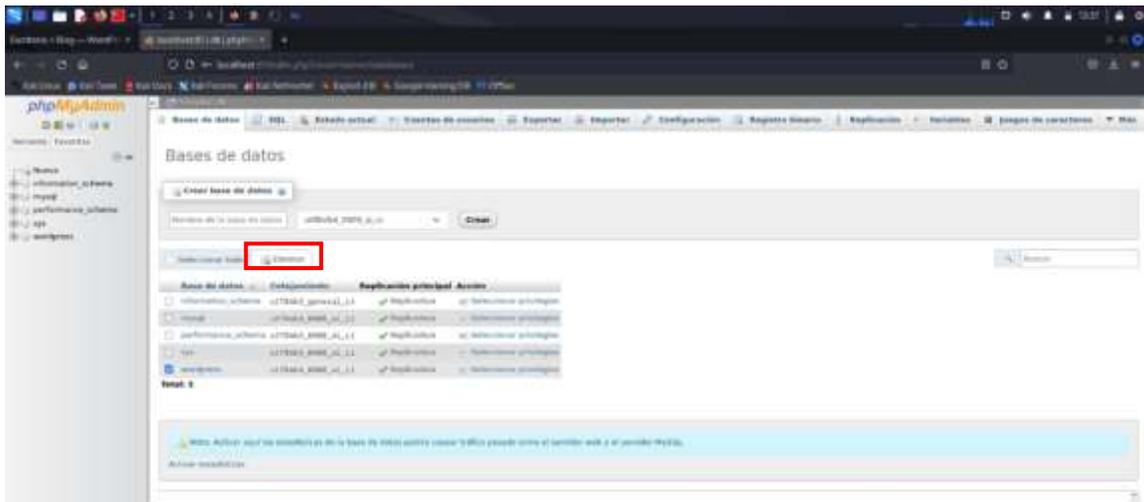


Ilustración 157: Proceso de eliminación

Nos muestra un cuadro de confirmación y aceptamos:



Ilustración 158: Eliminación de base de datos

Verificamos en nuestro sitio “Blog” y recargamos la página, veremos que nos muestra un error de que no tiene una base de datos seleccionada:

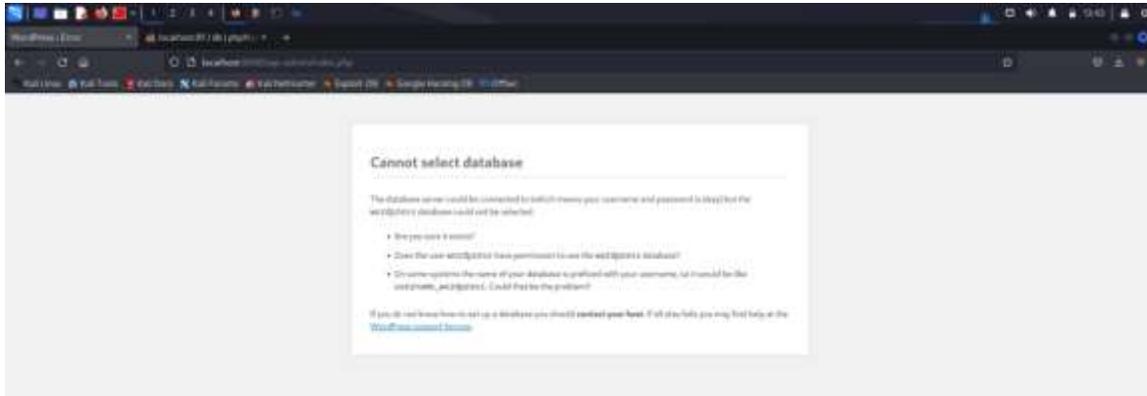


Ilustración 159: Error de página

Así que regresamos a phpMyadmin y debemos crear una base de datos nueva, ya que si no existe la base de datos nos devolverá un error, así que creamos en nueva:

Le asignamos un nombre a la nueva base de datos que será el mismo que el anterior "wordpress" y damos clic en crear:

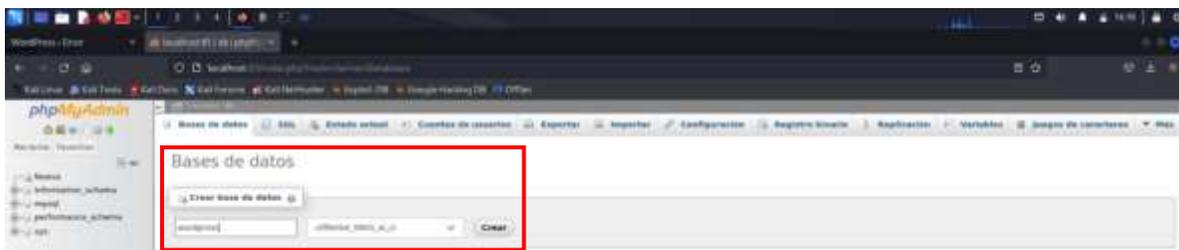


Ilustración 160: Asignación de nombre BD

Una vez creada vemos que está vacía:

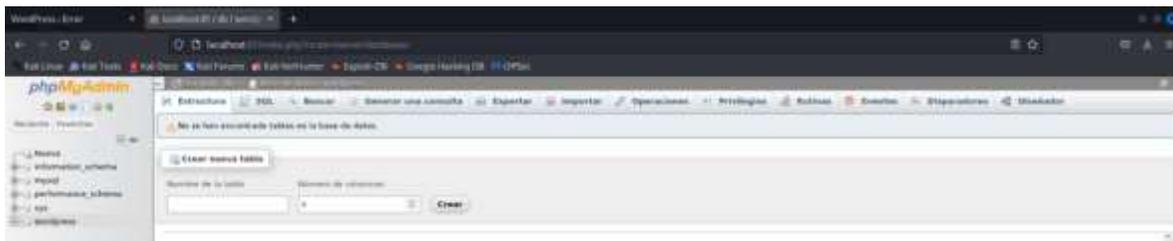


Ilustración 161: Creación BD



Así que seleccionamos la opción de importar:

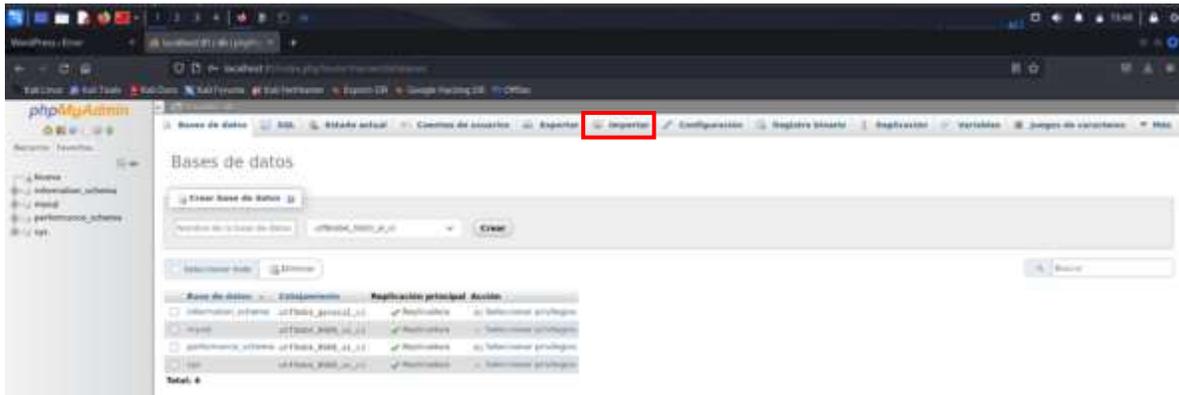


Ilustración 162: Importar BD

Nos muestra una vista en donde debemos seleccionar la base de datos que queremos exportar:

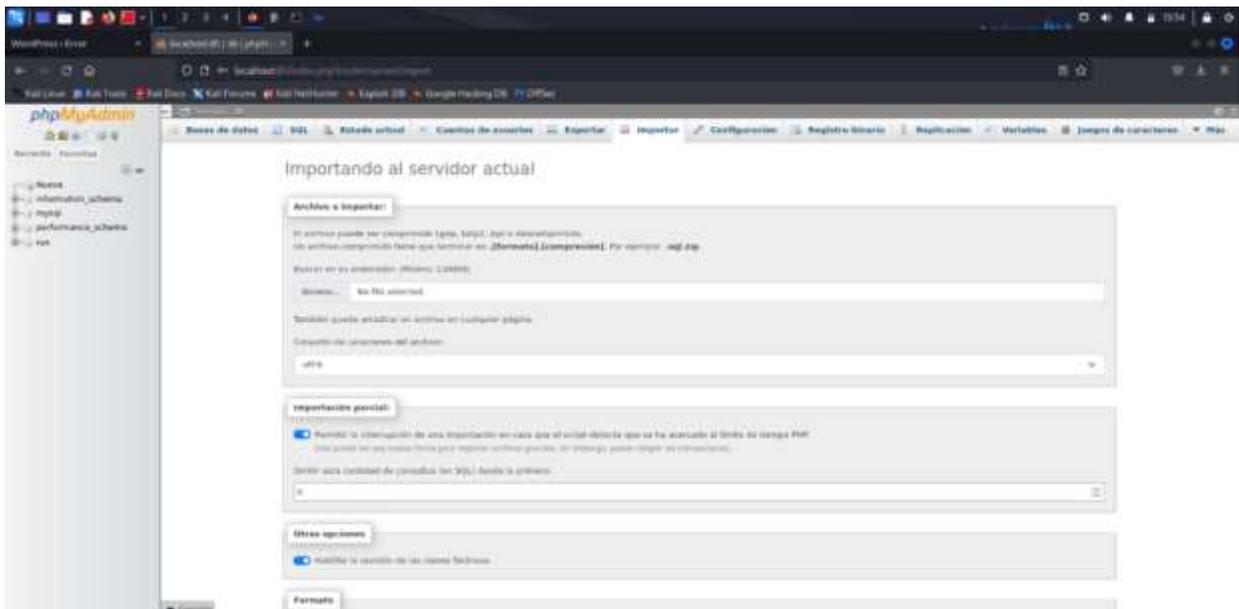


Ilustración 163: Selección BD



Damos clic en browse y seleccionaremos el respaldo que creamos:

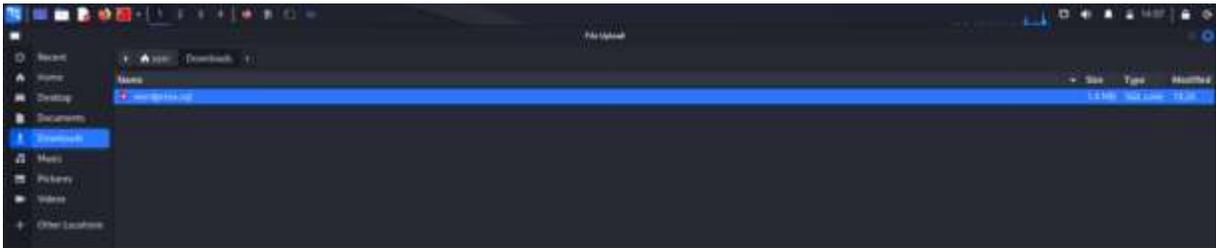


Ilustración 164: Abrir BD

Luego de seleccionar la base de datos nos dirigimos al final de la página y damos clic en importar

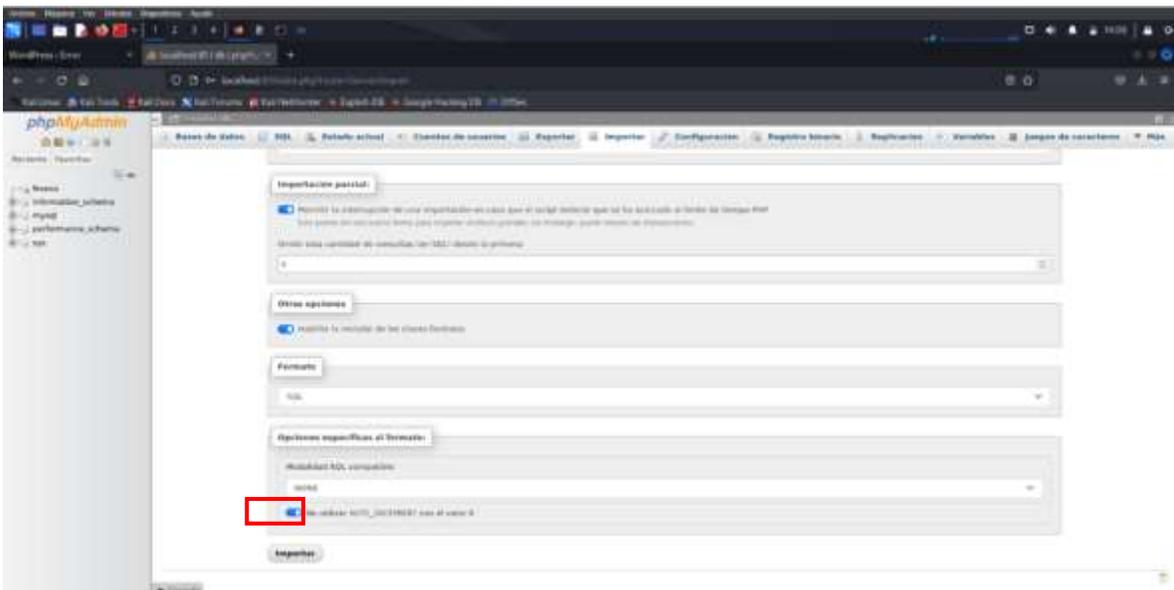


Ilustración 165: Importar BD

Vemos que la importación se hizo correctamente y que la base de datos que creamos ya no está vacía:



Enunciados:

1. Modificación del aspecto de WordPress mediante "Temas".
 - 1.1. Imagen destacada: Asocie una imagen a una de las entradas, que se mostrará en la parte superior
 - 1.2. Cambiar de tema o modificar el actual: En este ejercicio se hará un cambio de tema y personalizará el tema deseado.
2. Modificación del sitio web creado en WordPress:
 - 2.1. Primeros pasos: modificar ajustes básicos como: título del sitio, descripción, zona horaria, formato de hora y formato de fecha, modificar una entrada existente (título, contenido) y realizar un comentario a dicha entrada
 - 2.2. Entradas: crear una o varias entradas y que se muestren en la página principal
 - 2.3. Páginas: Añadir una página nueva y comprobar que no se pueden crear comentarios de forma predeterminada.

Soluciones:

1. Modificación del aspecto de WordPress mediante "Temas".

1.1 Imagen destacada:

Para asignar una imagen a una entrada, debemos editar la entrada y damos clic en "añadir nueva":

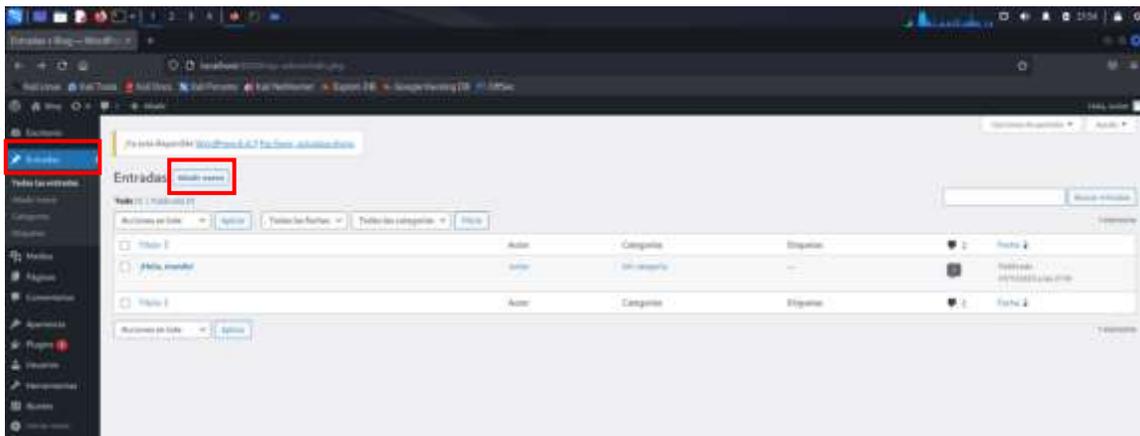


Ilustración 168: Añadir entrada



Utilizaremos la herramienta “imagen destacada”:



Ilustración 169: Imagen destacada

Se desplegará una ventana en la que seleccionaremos una imagen, ya sea que se encuentre en la biblioteca del sitio o subiremos una en caso de no tener ninguna:



Ilustración 170: Seleccionar imagen



La imagen elegida se mostrará en la herramienta “imagen destacada”:



Ilustración 171: Imagen destacada

Ahora solo debemos colocar un título al bloque y podremos guardar los cambios en “publicar”:

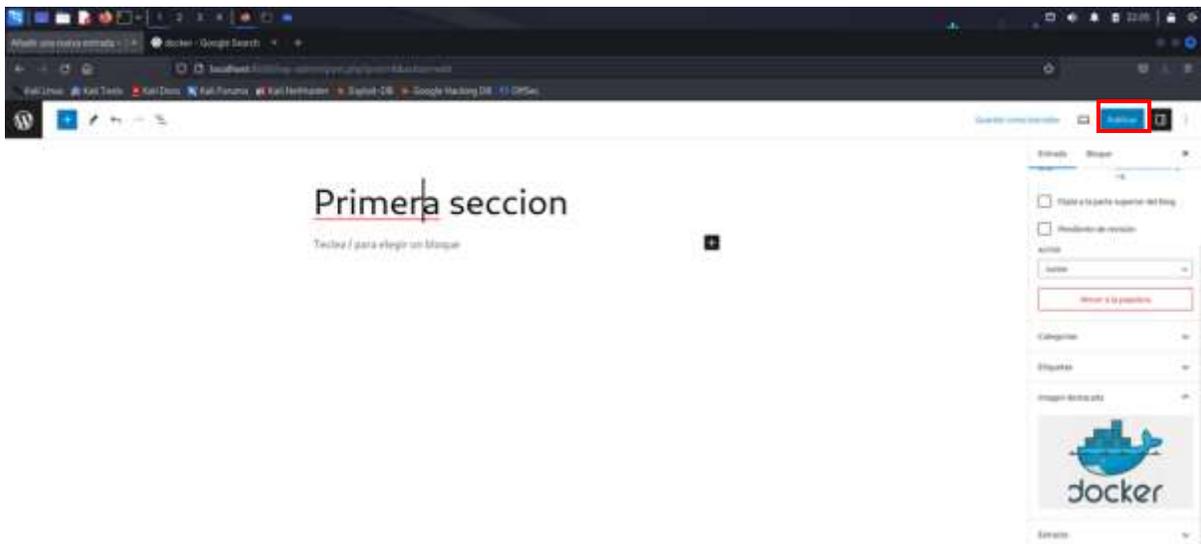


Ilustración 172: Ingresar título



Luego volveremos al menú principal dando clic en el icono de wordpress:



Ilustración 173: Menú principal

Para ver los cambios que hicimos visitaremos el sitio desde la vista de un cliente haciendo clic en el nombre del sitio:

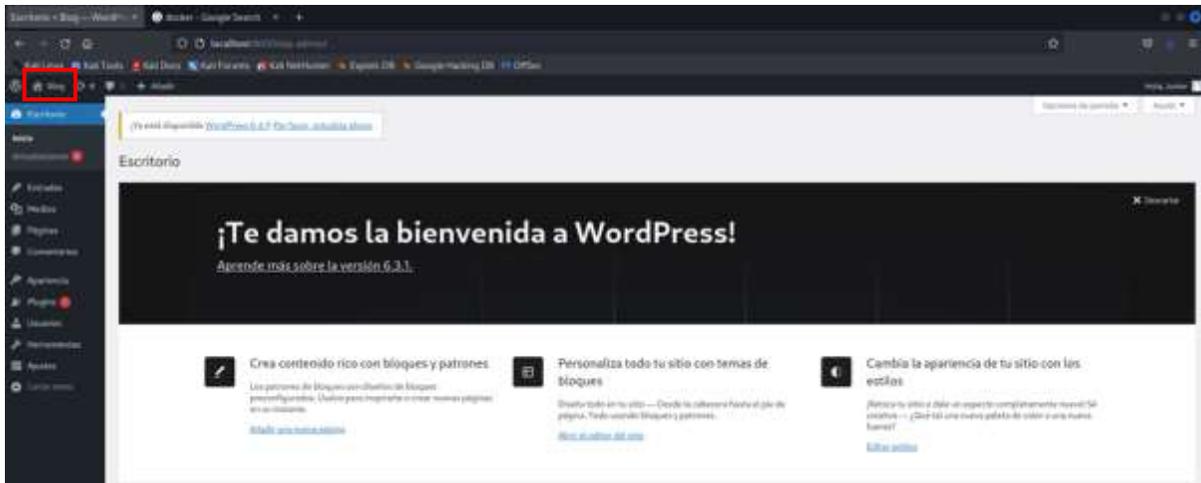


Ilustración 174: Sitio web cliente



Veremos que la imagen destaca que elegimos y el título del bloque se muestran correctamente:



Ilustración 175: Título e imagen

1.2 Cambiar tema o modificar el tema actual:

Para cambiar el tema del blog solo debemos ir a la opción de “apariencia”:



Ilustración 176: Apariencia



Una vez en este menú, tenemos la opción de cambiar de tema o en caso de no tener solo deberíamos descargar alguno, también está la opción de personalizar el tema:

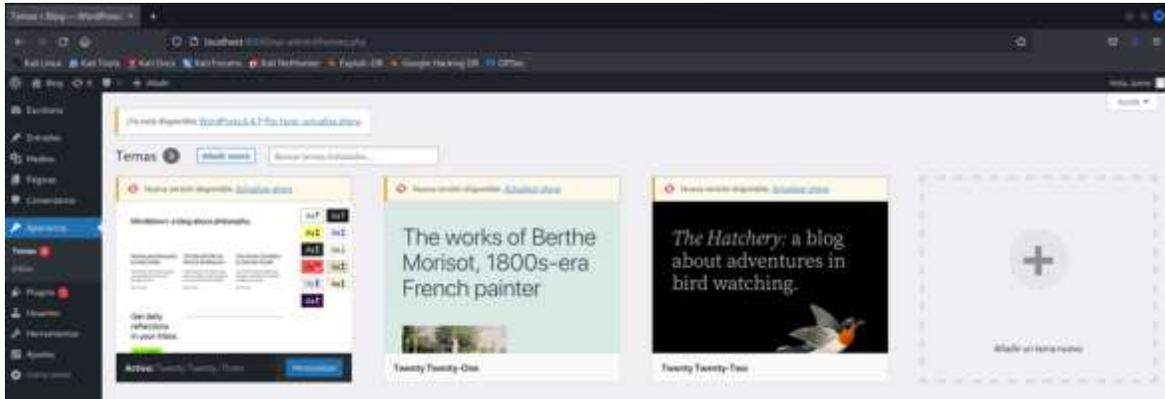


Ilustración 177: Opciones de apariencia

Si se desea cambiar de tema solo seleccionamos el que se quiera colocar y lo activamos:

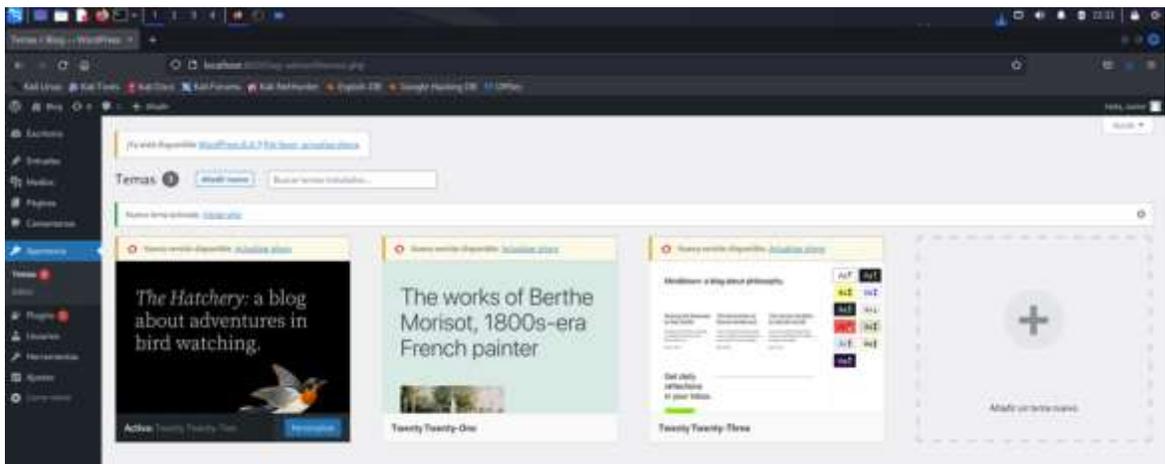
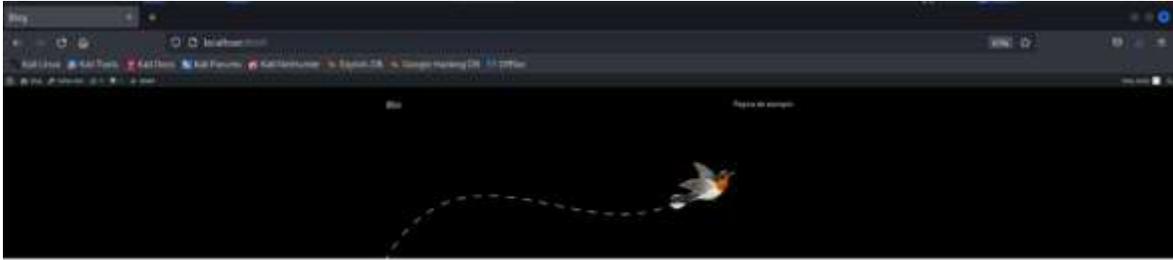


Ilustración 178: Selección de tema

Para verificar el cambio solo hacemos el cambio de vista que hicimos en el paso anterior y veremos que el tema del sitio cambió:



Primera sección



Ilustración 179: Verificar tema

Para tener un tema personalizado solo debemos dar clic en personalizar en el tema que se desee:

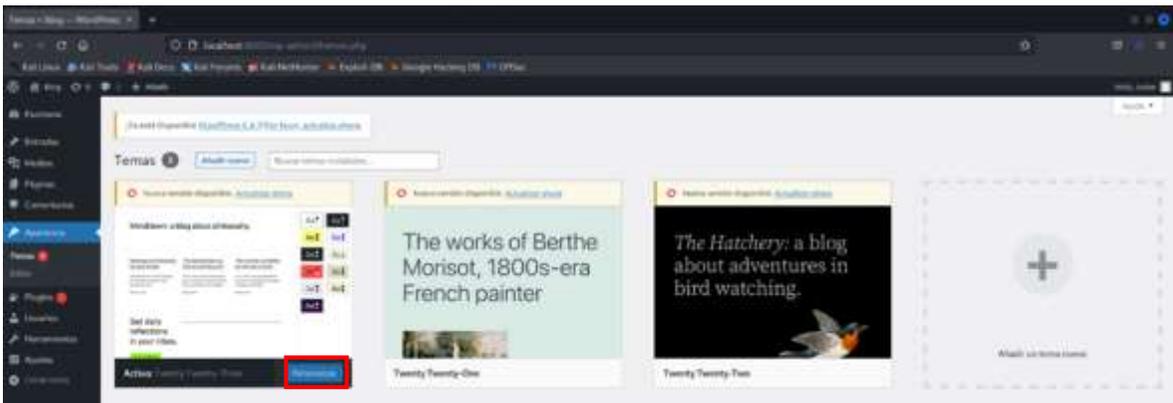


Ilustración 180: Personalizar tema

Una vez ahí tenemos muchas opciones para hacer cambios:
Elegimos la opción de “estilos” y veremos diferentes opciones:

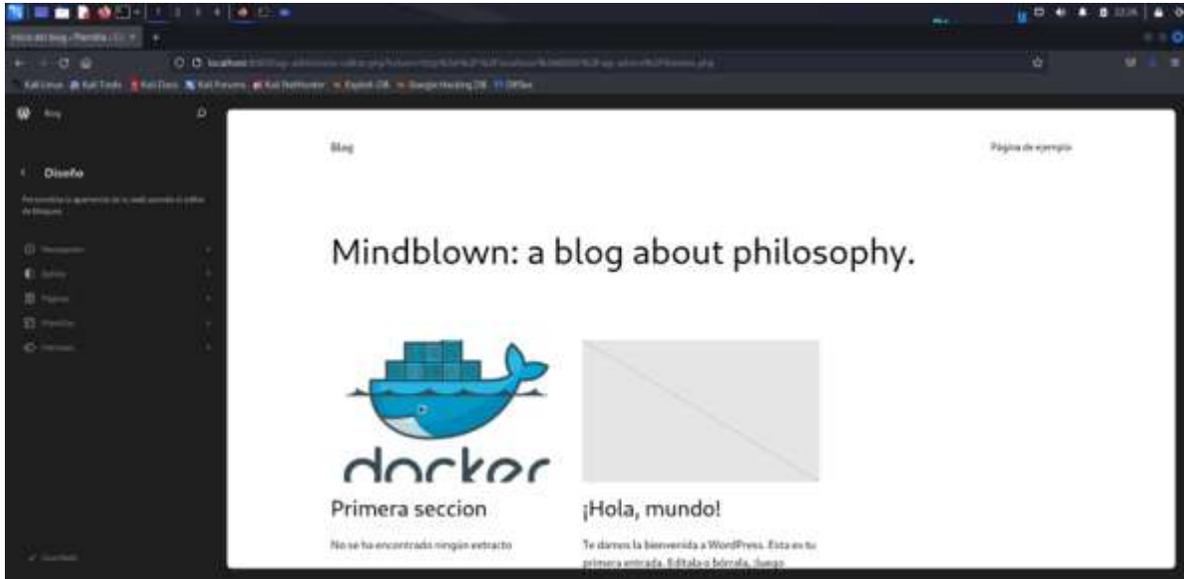


Ilustración 181: Estilos

Seleccionamos la que más nos llame la atención y guardamos los cambios:

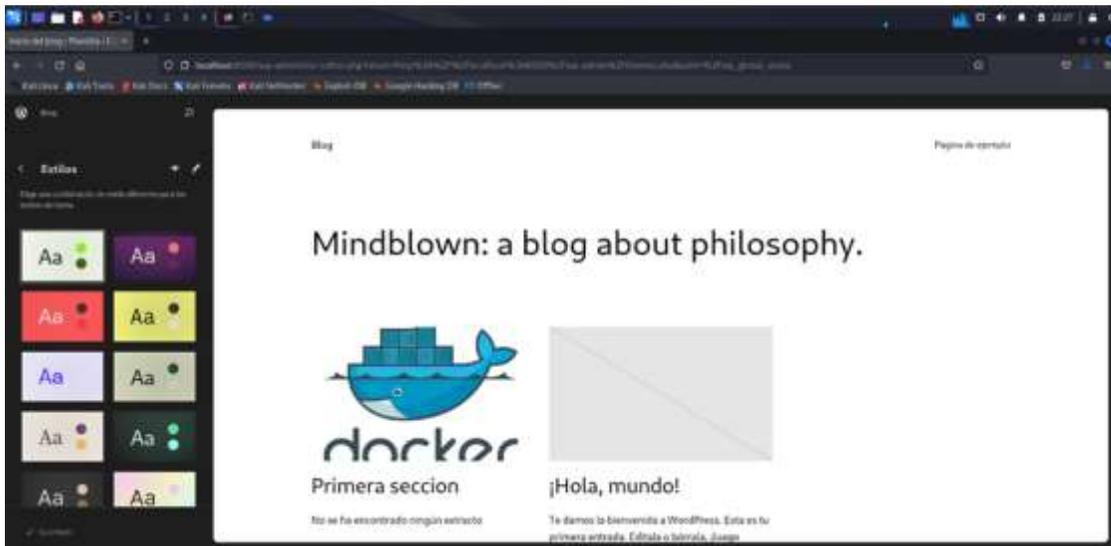


Ilustración 182: Selección de estilo



Ilustración 183: Guardar estilo

Una vez guardados los cambios, regresamos al menú principal y hacemos el cambio de vista para verificar los cambios:

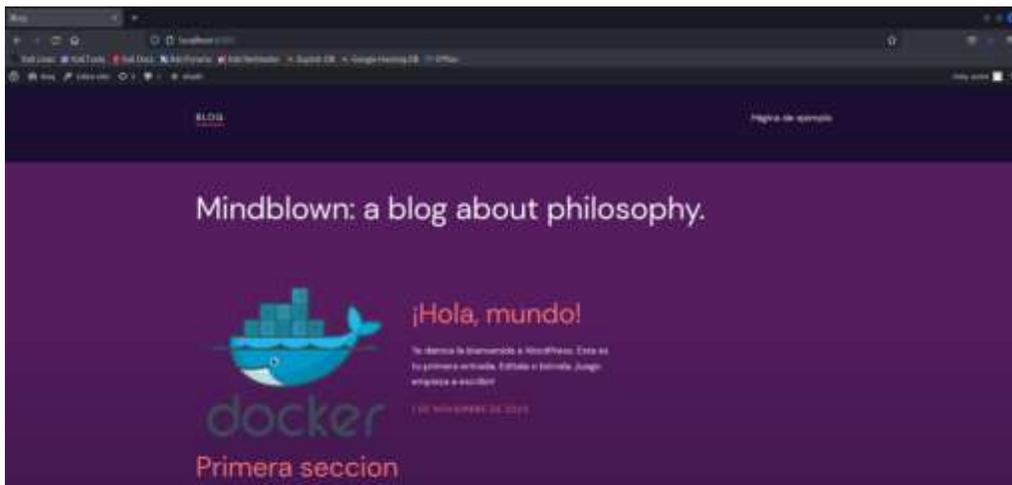


Ilustración 184: Verificar estilo

2. Modificación del sitio web creado en WordPress:

2.1 Primeros pasos

Al instalar WordPress, el blog contiene ya algunos elementos predeterminados. Así trataremos de modificar algunos de estos elementos, sin crear ninguno más. En algunos casos, estos elementos se pueden modificar desde varios sitios.



Ajustes

básicos:

Si queremos cambiar el título del sitio, descripción entre otras características, nos dirigimos a ajustes y daremos clic en generales:

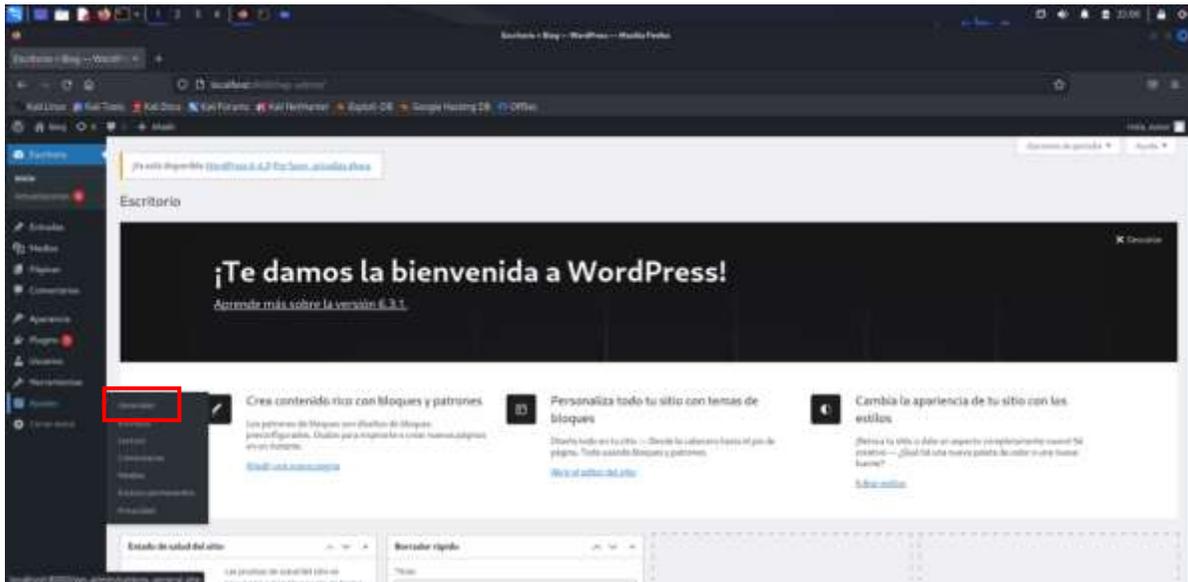


Ilustración 185: Ajustes en pantalla principal de WordPress

Colocaremos un nuevo título:



Ilustración 186: Ajustes generales (1)



Una descripción corta:



Ilustración 187: Ajustes generales (2)

Nuestra zona horaria en este caso es Managua:



Ilustración 188: Ajustes generales (3)

Cambiamos el formato de fecha:



Ilustración 189: Ajustes generales (4)



El formato de hora lo dejamos como está y como último paso, guardamos los cambios.



Ilustración 190: Guardar cambios hechos en ajustes generales

Entradas

El blog por defecto contiene una entrada que se titula "¡Hola mundo!".

Para modificar las entradas nos dirigimos a “Entradas”, luego en “Todas las entradas”

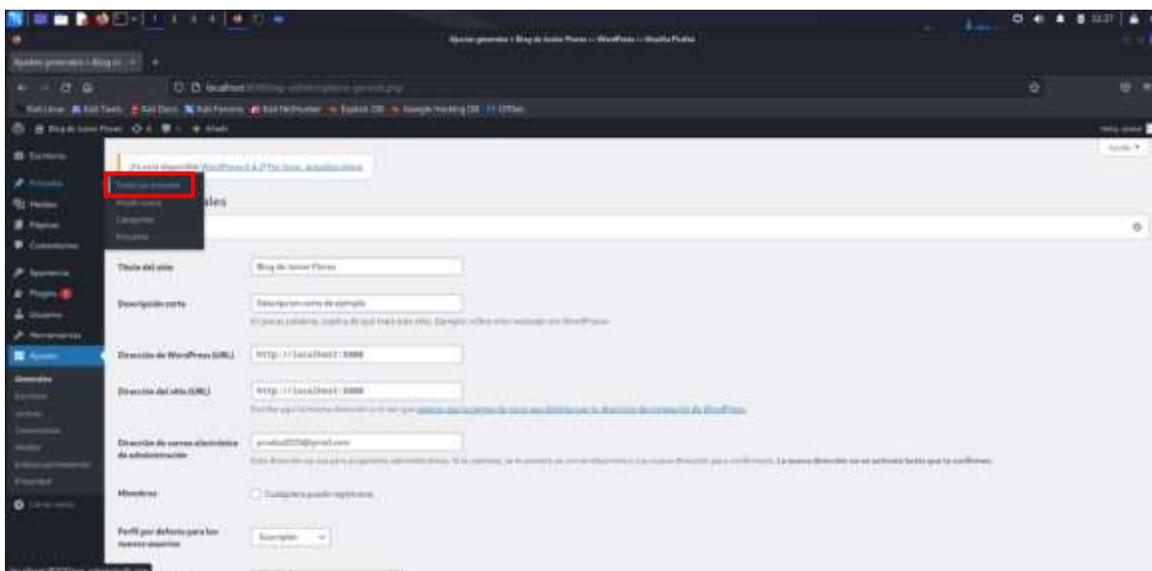


Ilustración 191: Seleccionando menú de entradas



Luego haremos clic en la entrada que se llama “Hola mundo”:

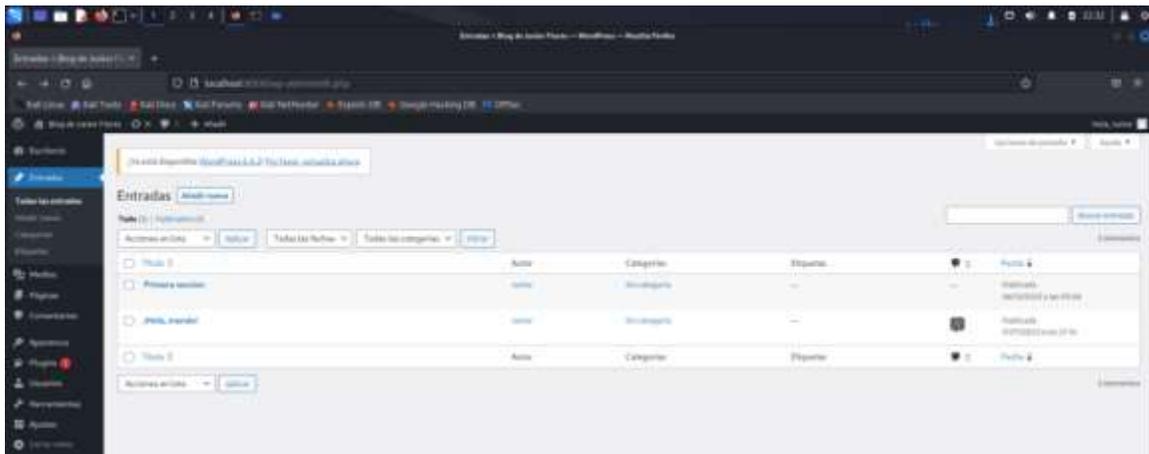


Ilustración 192: Menú de entradas

Modificaremos el título y contenido de la primera entrada. Una vez modificados debemos dar clic en actualizar:

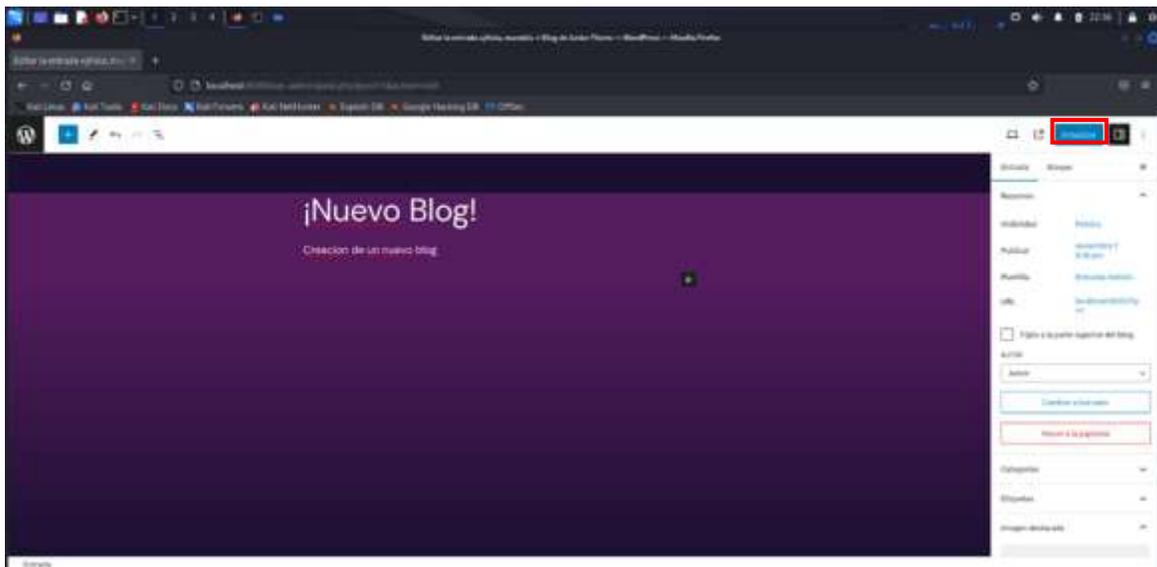


Ilustración 193: Modificando una entrada

Regresamos al menú principal y nos colocamos en la vista de cliente para ver los cambios:

Podemos ver que la entrada de “Hola mundo” ya cambio por la que nosotros colocamos.



Ilustración 194: Verificando el cambio realizado a la entrada

También podemos modificar comentarios de las entradas:

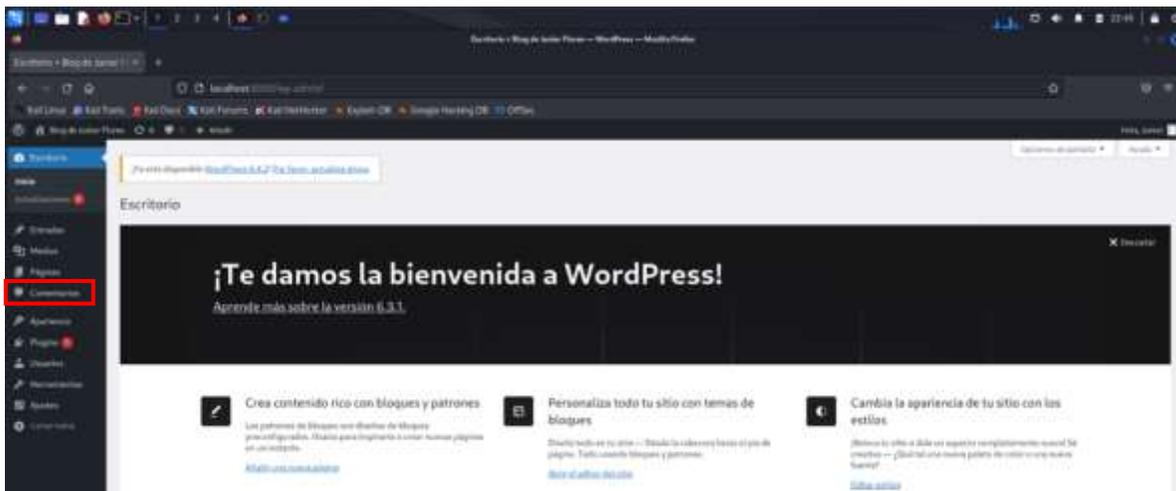


Ilustración 195: Ubicando comentarios en la pantalla principal de WordPress

Situando el cursor sobre un comentario, se muestran opciones para editar el comentario:

La opción Edición rápida permite cambiar varios elementos de un comentario: el contenido, los datos del autor del comentario, etc.

La opción Editar permite cambiar cualquier elemento del comentario.

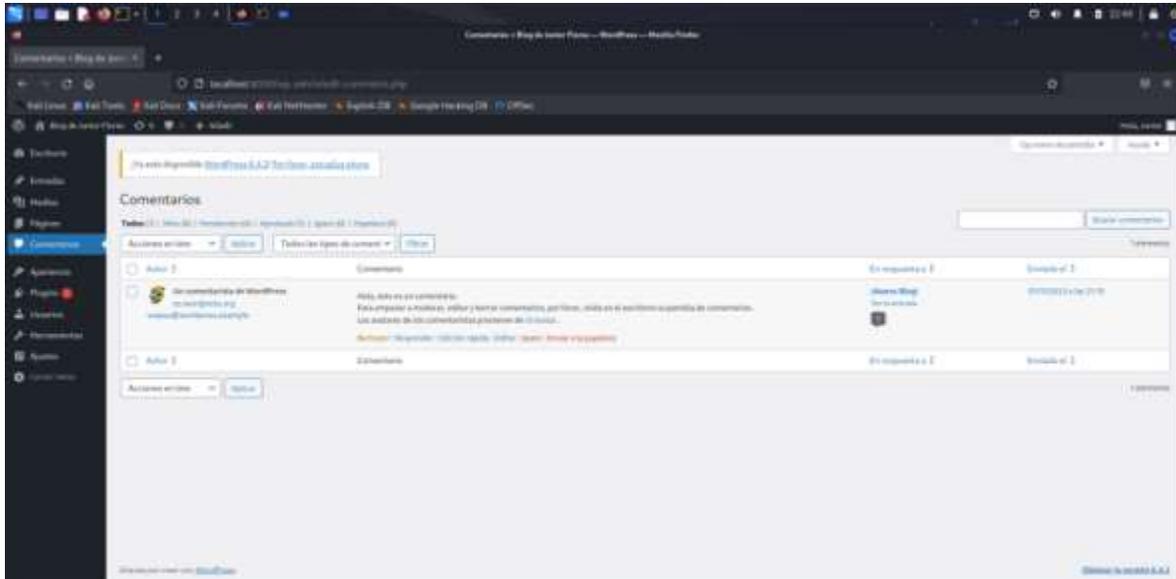


Ilustración 196: Menú principal de comentarios

Usaremos la opción de edición rápida y cambiaremos el comentario:

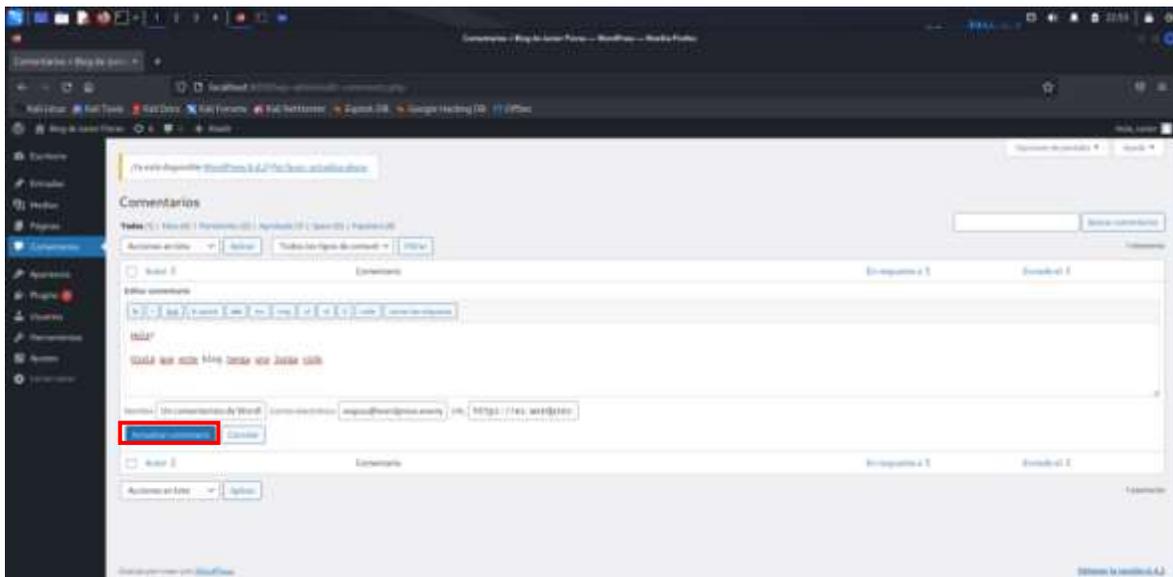


Ilustración 197: Editando un comentario

Luego damos clic en actualizar comentario

Nos dirigimos a la vista de cliente y damos clic en “¡Nuevo Blog!”



Ilustración 198: Más detalles de una entrada

Aquí podremos ver más detalles acerca de esta entrada:

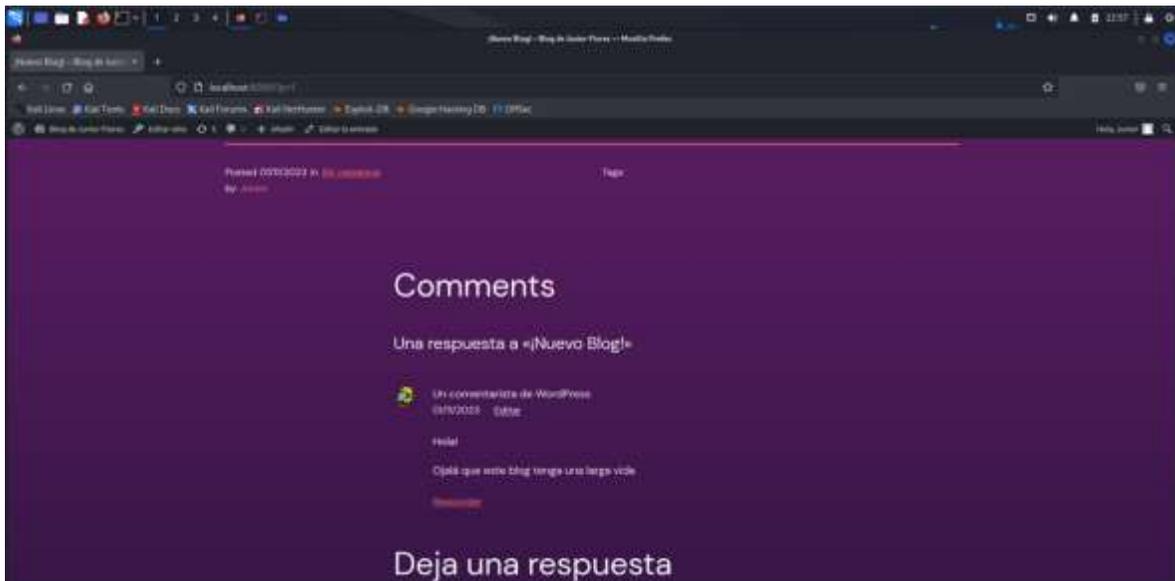


Ilustración 199: Comentarios de una entrada

2.2 Entradas

Las entradas se muestran automáticamente en la web. Además, de forma predeterminada, se pueden crear comentarios en las páginas (aunque se puede configurar una página para que no admita comentarios).



Crearemos unas cuantas entradas en el blog:

Nos dirigimos a “Entradas” en el panel principal y le damos en crear una nueva entrada:

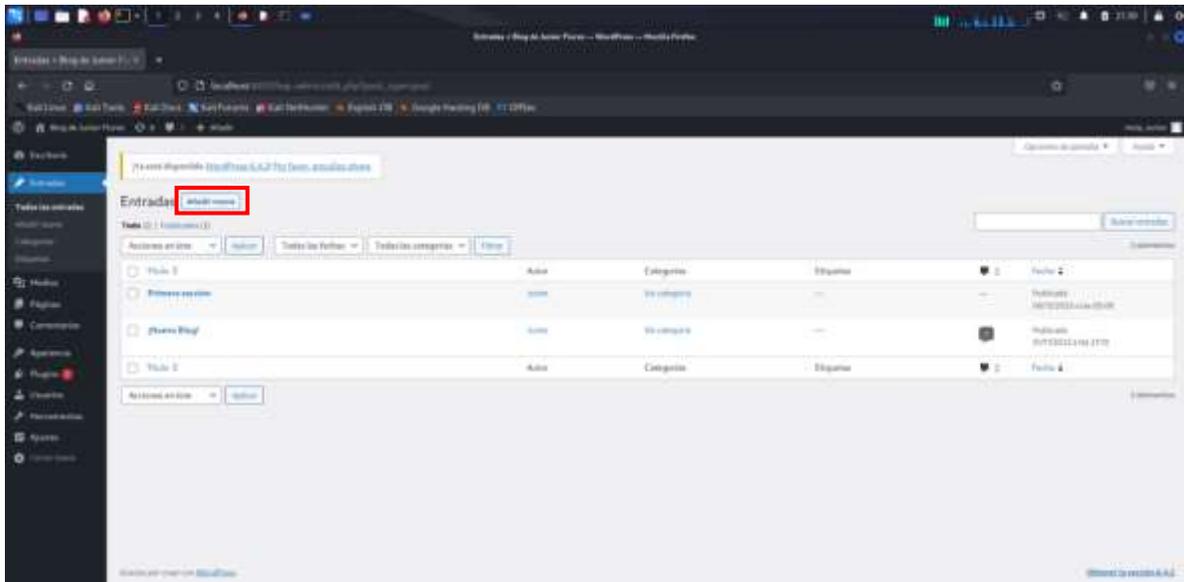


Ilustración 200: Agregando una nueva entrada

En el menú de crear una nueva entrada como mínimo debemos colocar un título a la entrada y luego damos clic en publicar:

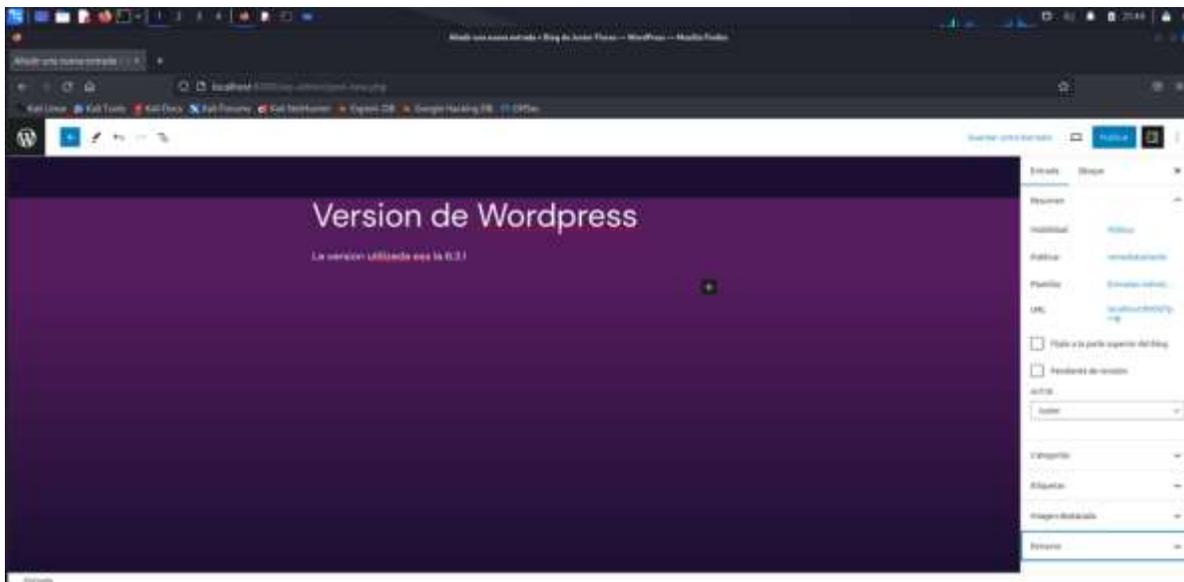


Ilustración 201: Nueva entrada



Luego de publicar la entrada nos dirigimos a la vista de cliente para verificar que la entrada se ha publicado correctamente:



Ilustración 202: Verificando la nueva entrada

De igual manera podemos crear y modificar las entradas que se requiera:

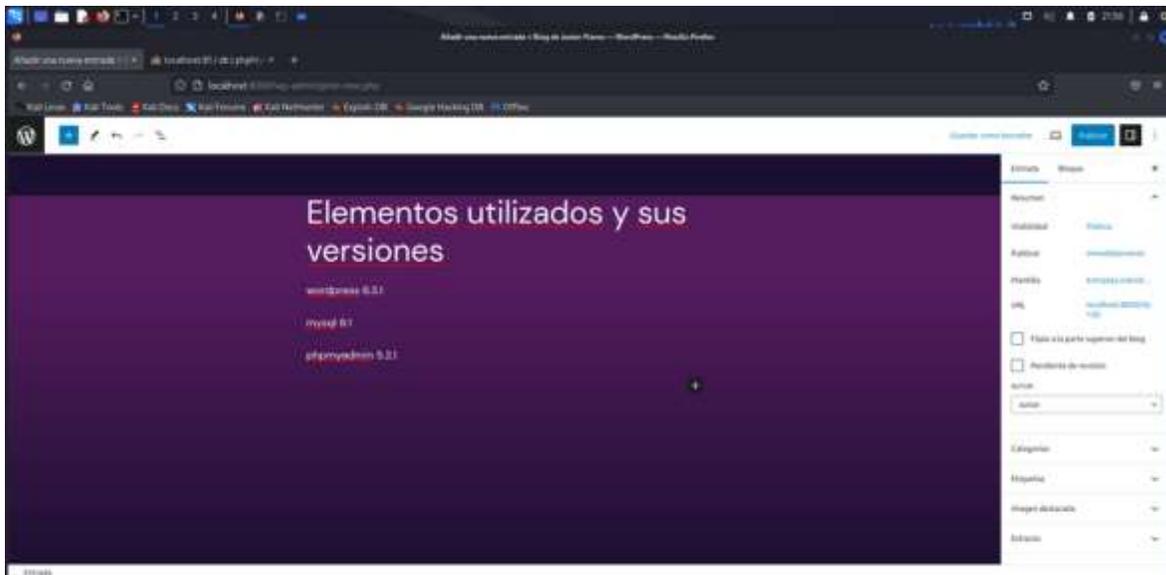


Ilustración 203: Agregando otra entrada



Y revisarlas en la vista cliente:



Ilustración 204: Verificando la nueva entrada creada

2.3 Páginas

Debemos mencionar que las páginas que crearemos no se muestran automáticamente en la web. Además, de forma predeterminada, no se pueden crear comentarios en las páginas (aunque se puede configurar una página para que admita comentarios).

Para crear una página debemos ir al menú de páginas que se encuentra en el panel del menú principal:



Ilustración 205: Ubicando el menú de páginas en la Pantalla principal de WordPress



Damos clic en añadir nueva:

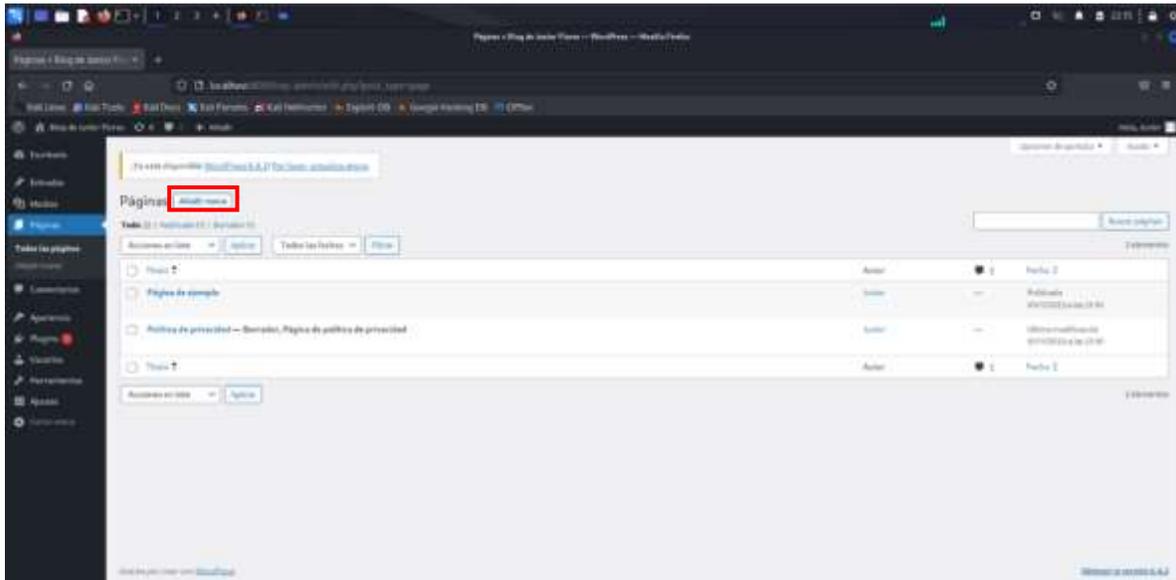


Ilustración 206: Menú de páginas

Se nos abrirá el menú para crear nuestra página según se requiera o el administrador lo desee:

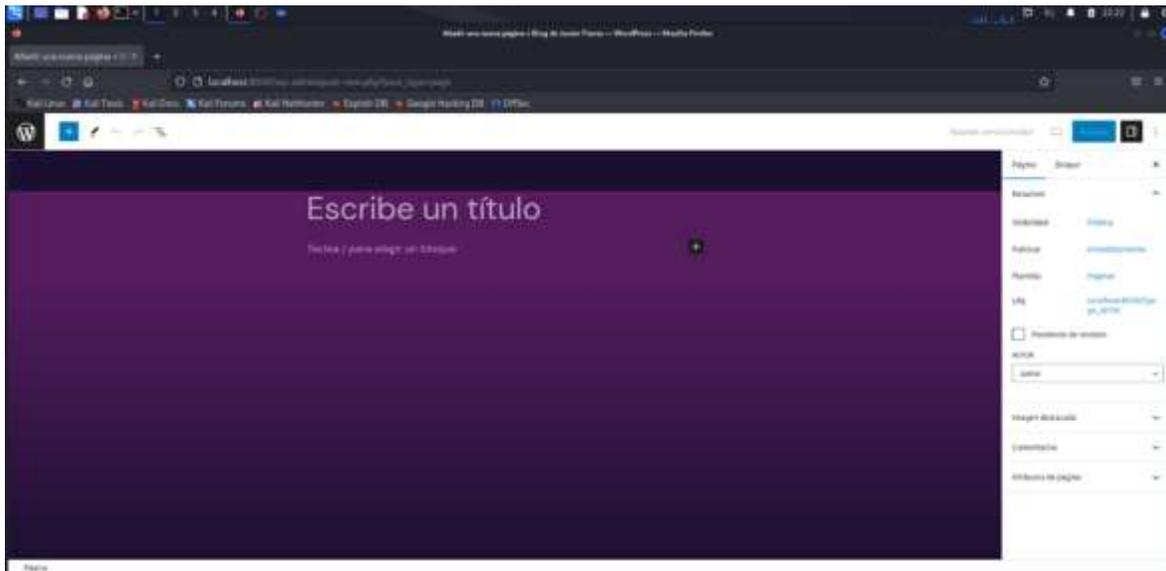


Ilustración 207: Creando una nueva página



Agregamos unas imágenes y un título. También revisamos en el panel derecho la opción de comentarios y vemos que por defecto esta deshabilitado “Permitir comentario”:

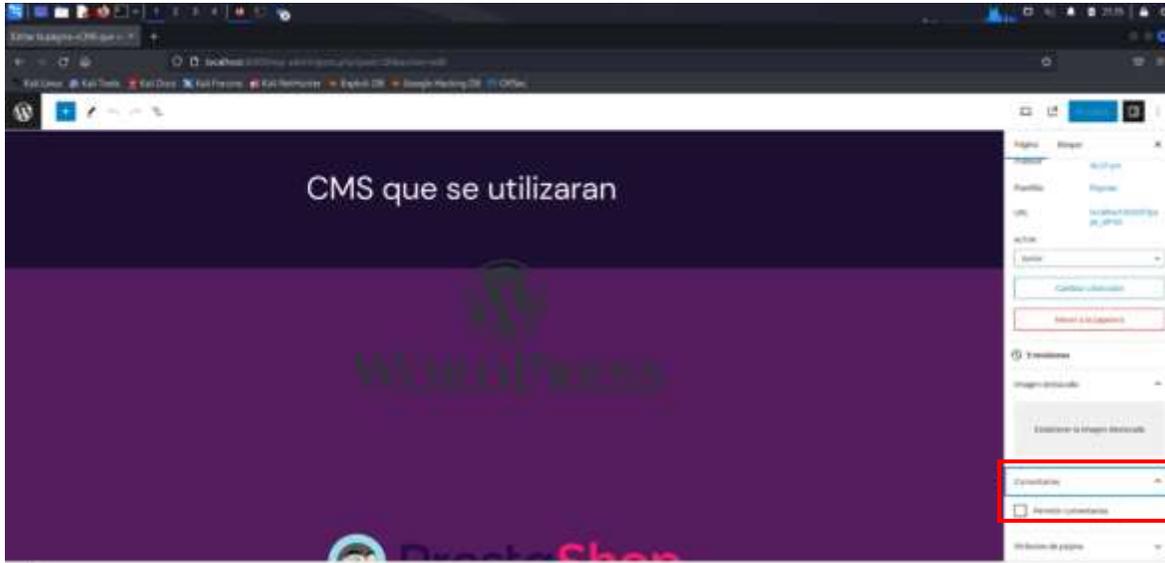


Ilustración 208: Opción de permitir comentarios

Luego de crear nuestra página, solo debemos publicarla:

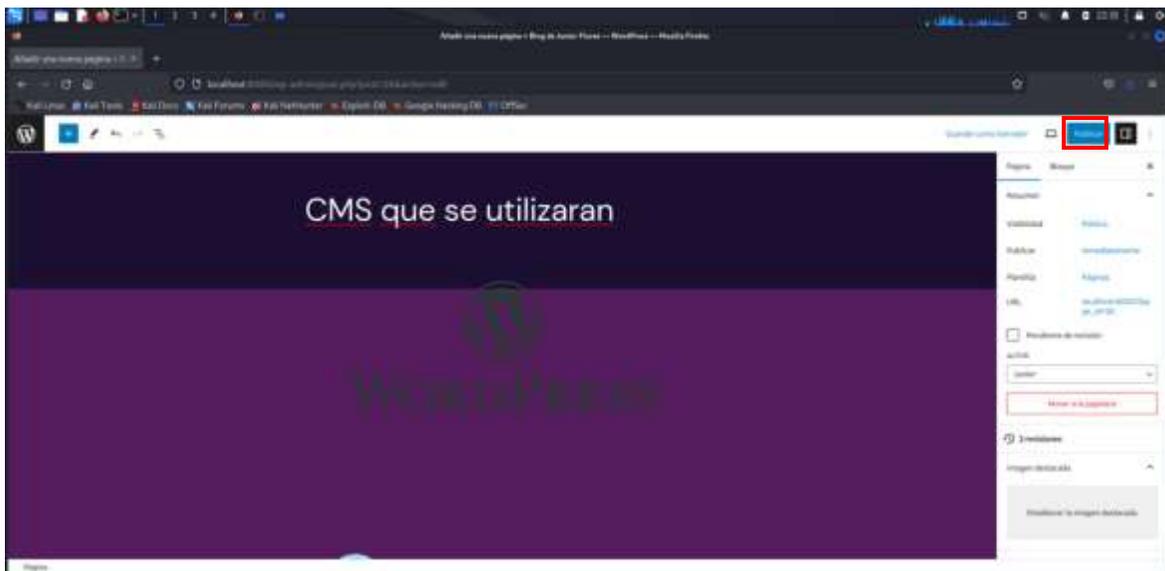


Ilustración 209: Guardando los cambios



Luego de crearla, lo verificamos en la vista de cliente:

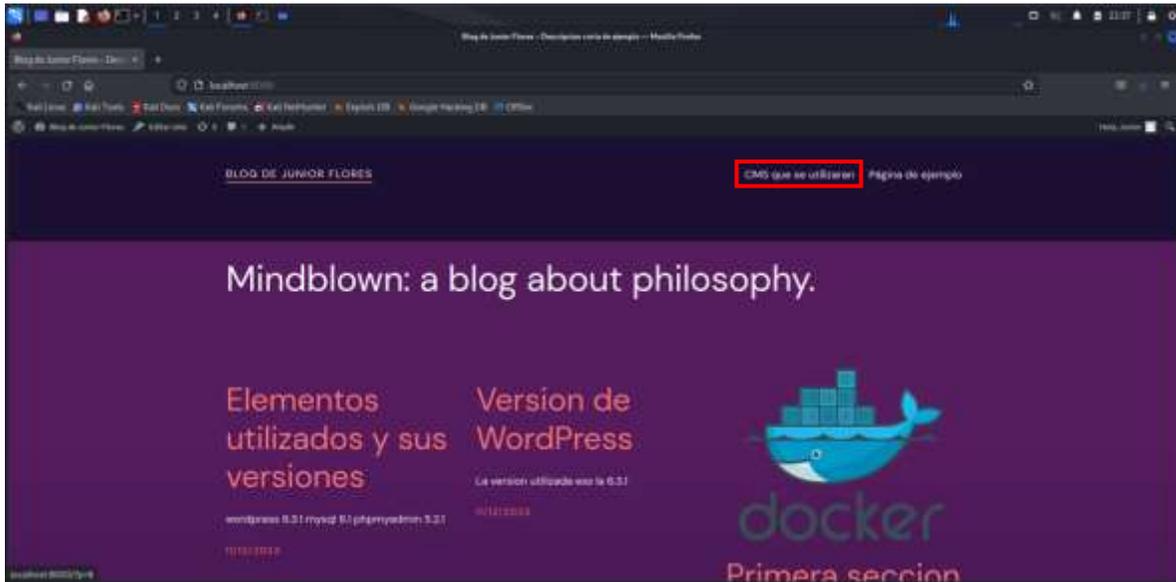


Ilustración 210: Verificación de la página nueva agregada

Y vemos que el acceso a nuestra página creada se encuentra anclada en la página principal, así que solo debemos dar clic para acceder a ella:

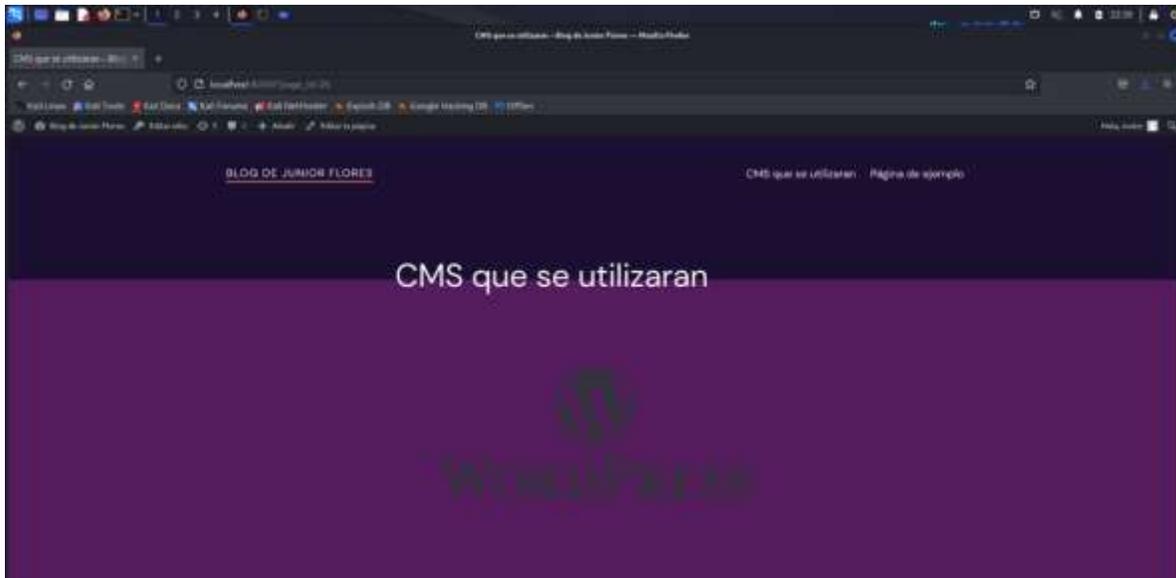


Ilustración 211: Página nueva (1)



Ilustración 212: Página nueva (2)



Ilustración 213: Página nueva (3)

Y vemos que la pagina con imágenes de distintos CMS que creamos se muestra correctamente.

Y así podemos crear las páginas que sean necesarias o se requieran.



Y se observa que al final de la página no se encuentra el panel para dejar comentarios, mientras que si la opción se hubiera habilitado nos aparecería de la siguiente manera:

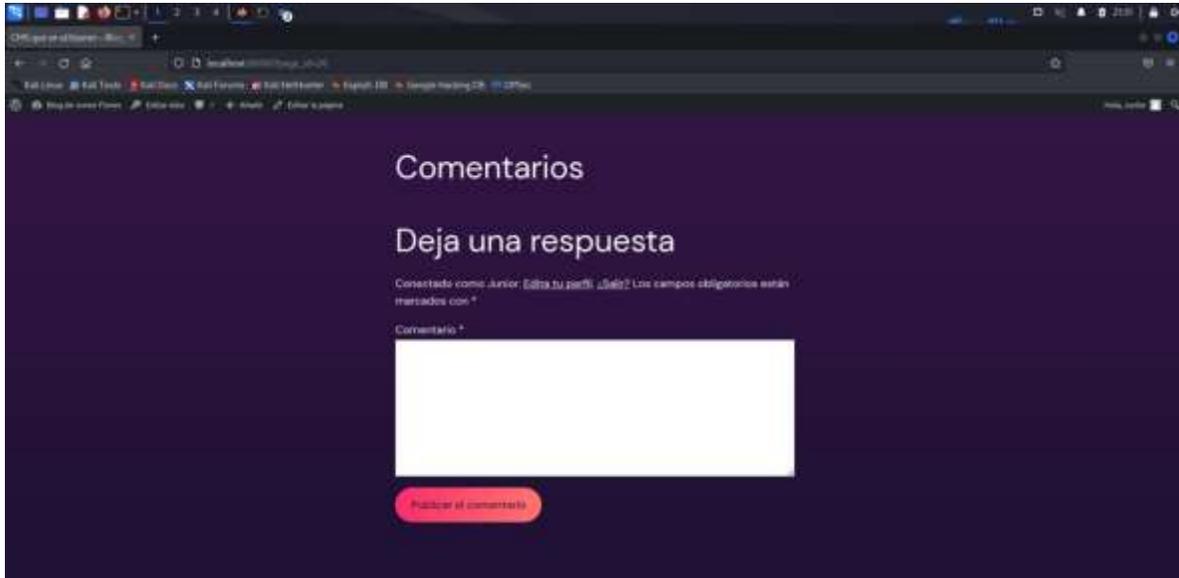


Ilustración 214: Muestra de si la opción de Permitir comentarios estuviera habilitada

Joomla

Joomla (1) Instalación del CMS

1. habilitar el contenedor de dicho cms para empezar el funcionamiento de sus respectivos servicios
2. Una vez habilitado solo es cuestión de abrir el navegador y en el buscador ingresar como localhost:"Numero_de_puerto" (el número de puerto se proporciona al momento de habilitar el contenedor en cuestión)
3. Aquí comienza la instalación del CMS:

* Configuración o relleno de la información necesaria para la creación del blog.

Por ejemplo

- Nombre del sitio



Instalador de Joomla Joomla! 4.3.4

A-Z Seleccione el idioma de instalación

Seleccionar el idioma

Español (España)



⚙️ Configurar el nombre del sitio

Introduzca el nombre de su sitio Joomla! *

|

Configurar los datos de inicio de sesión >

Ilustración 215: Configuración del nombre del sitio

- Descripción (descripción general de todo el sitio)
- Correo electrónico
- Nombre de usuario del administrador, contraseña... etc.



Instalador de Joomla Joomla! 4.3.4

Datos de acceso

Asigne el nombre, 'nick' o alias que desee mostrar en su cuenta de superusuario. *

Asigne el usuario con el que poder acceder a su cuenta con privilegios de superadministrador. *

Asigne la contraseña para su cuenta de superusuario. *



Como mínimo debe introducir 12 caracteres.

Introduzca la dirección de correo electrónico del superusuario del sitio. *

Ilustración 216: Información vital para el superusuario

* Base de datos o configuración de la base de datos

- Tipo de la base de datos

- Hospedaje



Instalador de Joomla Joomla! 4.3.4

Configuración de la base de datos

Seleccione el tipo de base de datos. *

MySQLI

Introduzca el nombre del hospedaje. Normalmente es "localhost" o el nombre proporcionado por su hospedaje. *

localhost

El nombre de usuario que haya elegido o el facilitado por su proveedor de hospedaje. *

Introduzca la contraseña para la base de datos que creó o le facilitó su proveedor de hospedaje.

Ilustración 217: Información requerida de la base de datos del sitio

NOTA: en esta sección siempre se debe colocar el nombre del servicio de la base de datos que se utiliza en el contenedor, en este caso llamado "db"



Seleccione el tipo de base de datos. *

MySQLi

Introduzca el nombre del hospedaje. Normalmente es "localhost" o el nombre proporcionado por su hospedaje. *

db

El nombre de usuario que haya elegido o el facilitado por su proveedor de hospedaje. *

joomla

Introduzca la contraseña para la base de datos que creó o le facilitó su proveedor de hospedaje.

.....

Introduzca el nombre de la base de datos. *

joomla

Introduzca un prefijo para la base de datos o use uno generado aleatoriamente. *

Ilustración 218: Configuración de la base de datos del sitio

- Usuario, contraseña y Base de datos, en este caso práctico para mayor facilidad, rapidez y entendimiento, estas 3 secciones se rellenan con la misma credencial que es simplemente escribir "Joomla"

Y otras secciones prefijas de las tablas y procesos para una base de datos antigua, los cuales no tomaremos en cuenta en este momento

* Visión general o finalización

- finalización: en esta sección podemos optar por instalar datos de ejemplos, estos datos de ejemplos hacen que Joomla se instale con algunos artículos o también módulos relacionados a diversos temas, en este caso lo dejaremos en ninguno

- Visión General: aquí podríamos enviar todos los datos de configuración por correo electrónico a un correo ya establecido anteriormente una vez terminado la instalación



- configuración Principal y de base de datos: nos muestra todos los datos anteriormente agregados

- Comprobaciones previas y configuraciones recomendadas

Una vez todo listo daremos en instalar



Ilustración 219: Configuración realizada para la instalación

4. por último se nos pedirá eliminar la carpeta de instalación lo cual podemos hacer desde ahí mismo

5. Una vez instalado el CMS podemos visitar el blog siempre usando la misma dirección de puerto como en el paso anterior o entrar a la parte de administración agregando /administrador como se puede ver en las capturas de pantalla posteriores



Ilustración 220: Panel principal de Joomla

Enunciados

Ahora procederemos a crear un portal web que cumpla con los requisitos básicos, desde la creación de artículos y menús, hasta la instalación de extensiones útiles y creación de distintos usuarios con diferentes permisos para el portal web

1. Creación de artículos (al menos 3)
2. En el Menú principal debe existir una opción para navegar a una galería de imágenes
3. En el menú debe existir una opción para navegar a una página de noticias(feeds)
4. Alguno de los artículos debe tener un video de YouTube
5. Debe existir al menos 3 tipos de usuarios del sitio
6. Alguno de los artículos debe tener un mapa de Google Maps
7. Se debe crear una página de contactos, que sera accedida desde el menú Principal

Soluciones:

1. Creación de artículos (al menos 3)

- Una vez que estemos en la parte administrativa del panel de inicio nos fijamos en la parte de sitio y damos click en la sección de artículos



Ilustración 221: Selección del menú "Artículos"

- De momento no tenemos ningún tipo de artículo en el portal, por lo que daremos click directamente en "añadir primer artículo"



Ilustración 222: Menú de "Artículos"



Ilustración 225: Visualización del artículo

2. En el Menú principal debe existir una opción para navegar a una galería de imágenes

- primero nos situamos en el menú de la izquierda y damos click en la parte de sistema y luego en la sección "instalar" daremos click en "Extensiones"



Ilustración 226: Panel de control del menú sistema



- Luego se nos muestra la siguiente pantalla

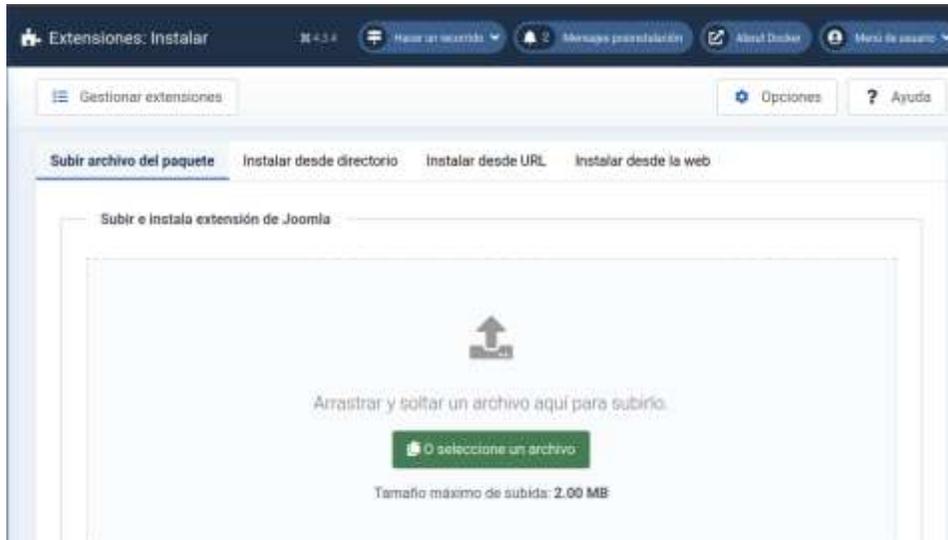


Ilustración 227: Menú de extensiones

- Se nos sitúa de forma predeterminada en la sección de "Subir archivo del paquete", Aquí podemos simplemente subir un archivo que hayamos previamente descargado desde la página de Joomla (la forma la cual implementaremos)

- Otra forma de hacer sería en la sección de instalar desde la web donde se nos mostrara diversas extensiones que pueden ser instaladas y una vez ahí buscar extensiones para implementar galerías en el portal web

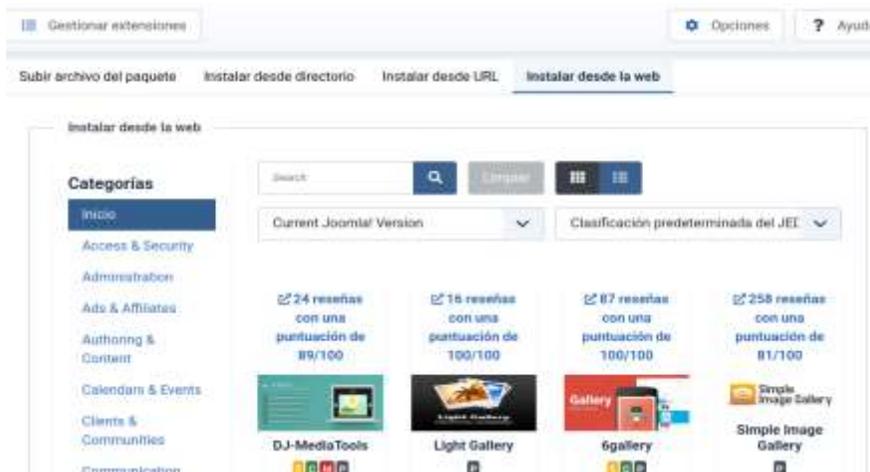


Ilustración 228: Implementar galería desde la web



NOTA: De cualquiera de las formas que se instale una extensión siempre se debe tener en cuenta algunas cosas:

- Si esta es compatible con la versión de Joomla que estamos utilizando (en este caso Joomla 4)
 - Si esta es una versión estable (las versiones beta también pueden funcionar correctamente, pero existe la posibilidad de que esta falle de alguna forma una vez funcionando)
 - Que la extensión sea gratis ("Free to download")
- En este caso utilizaremos la extensión de Simple_Image_Gallery el cual nos ofrece un formato bastante simple al momento de agregar una galería a nuestro portal web

JoomlaWorks 'Simple Image Gallery' Plugin v4.2 for Joomla

Simple Image Gallery Adding image galleries inside your Joomla articles is now super-easy and simple, using the magical Simple Image Gallery plugin for Joomla.

The plugin can turn any folder of images located inside your Joomla website into a grid-style image gallery with cool lightbox previews. And all that using a simple plugin tag like `{gallery}myphotos{/gallery}`.

So for example, if we have a folder called `my_trip_to_Paris` located in `images/stories/my_trip_to_Paris`, then we can create our gallery by simply entering the tag `{gallery}my_trip_to_Paris{/gallery}` into some Joomla article.

The galleries created are presented in a grid using a polaroid-like background for the thumbnails. When your visitors click on a thumbnail, they see the original image in a lightbox popup. The thumbnails are generated and cached using PHP for better results.

So let's briefly see what are the main advantages of using Simple Image Gallery: a) You don't need to have an additional gallery component to display a few images, b) you don't need to tell your visitors to see our photos from Paris click here and c) you focus more on content writing and less on administering the images!

The plugin is ideal for news portals wanting to display some product images, for example, inside their articles. Or for people who



Ilustración 229: Extensión "Simple_Image_Gallery "



- Para su funcionamiento primero debemos dirigirnos a la sección de multimedia que a su vez se encuentra en la sección de Contenido

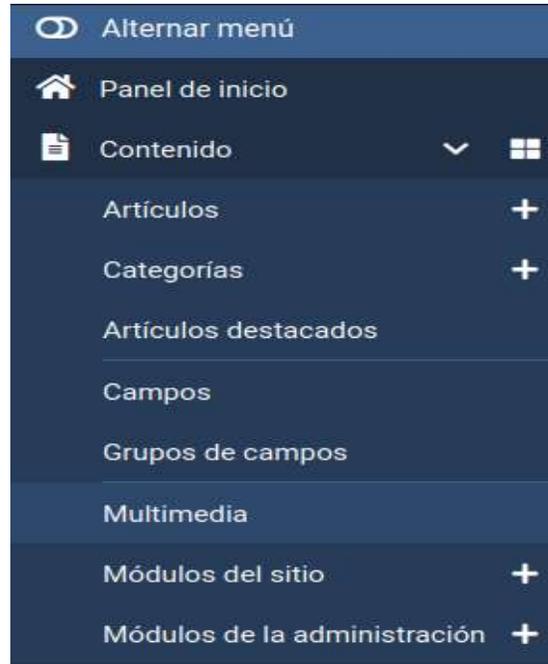


Ilustración 230: Opciones en la sección "contenido"

- Una vez ahí crearemos una nueva carpeta (en este caso llamada galería) que contendrá todas las imágenes que querremos se muestren en nuestra galería, quedará tal que así

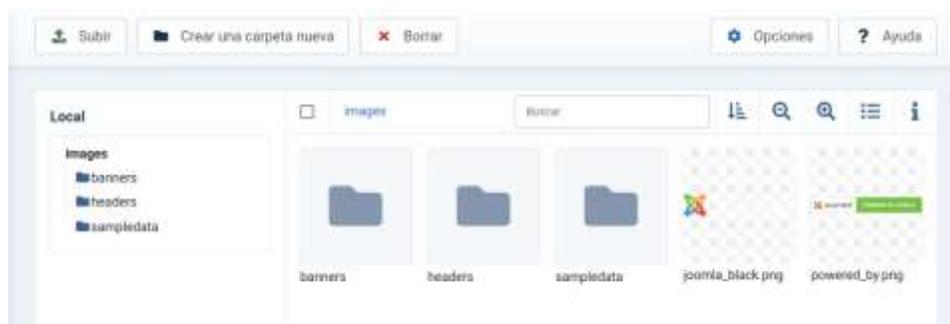


Ilustración 231: Creación de la carpeta galería

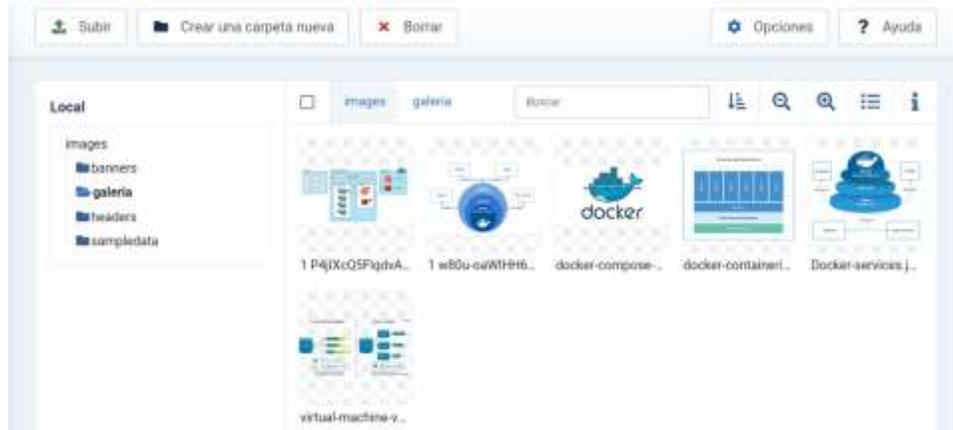


Ilustración 232: Contenido de la carpeta galería

- luego crearemos un nuevo artículo (que tendrá como título "Galería") de la misma forma que se hizo anteriormente, solo que esta vez agregaremos nuestra galería a este utilizando la extensión de Simple_Image_Gallery que instalamos lo cual hacemos con la siguiente sintaxis `{gallery}nombre_de_la_carpeta_con_imagenes{/gallery}`



Ilustración 233: Creación de un artículo para la galería

NOTA: hay que revisar si la extensión se encuentra habilitada, lo cual podemos revisar en la sección de sistema y luego en "gestionar" damos click en "Extensiones" y buscarla por su nombre para ver el estado en que se encuentra

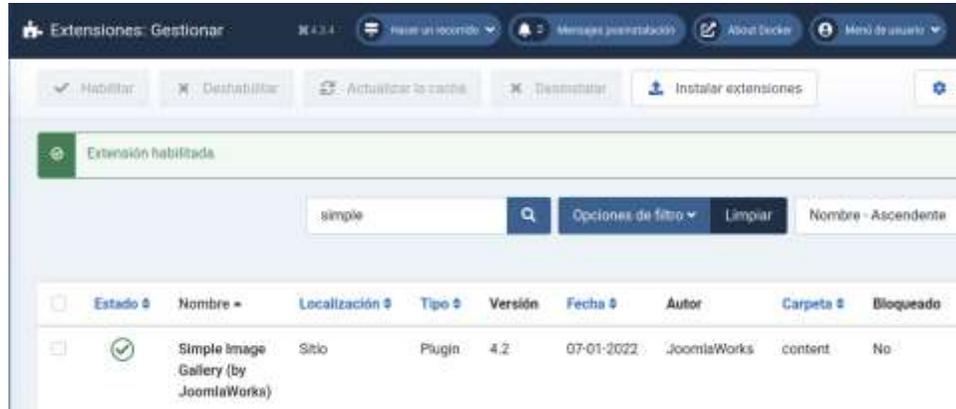


Ilustración 234: verificación de la extensión de galería

- Luego nos iremos al menú principal

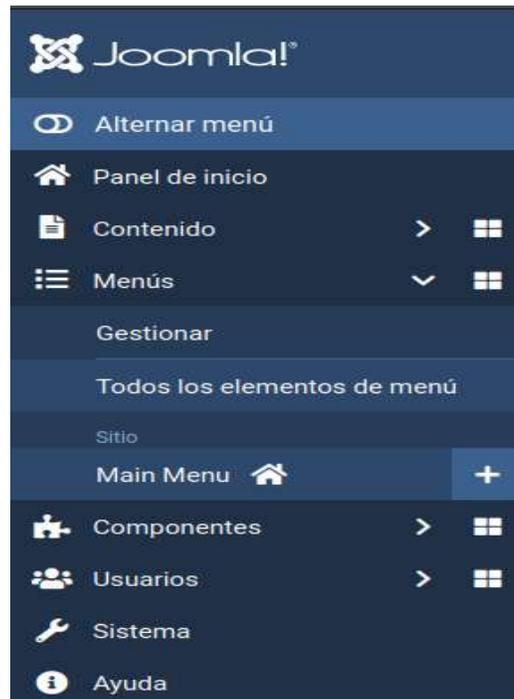


Ilustración 235: panel izquierdo del menú principal de Joomla



una vez ahí agregaremos un nuevo elemento el cual a su vez se agregará al menú principal de nuestro portal web

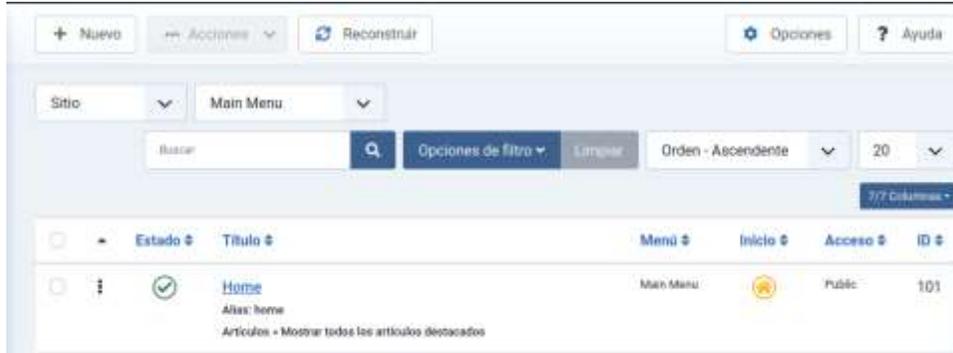


Ilustración 236: Panel de menús

y la información deberá quedar tal que así y así nuestra galería está lista para ser visualizada en el portal web



Ilustración 237: Agregando un nuevo menú

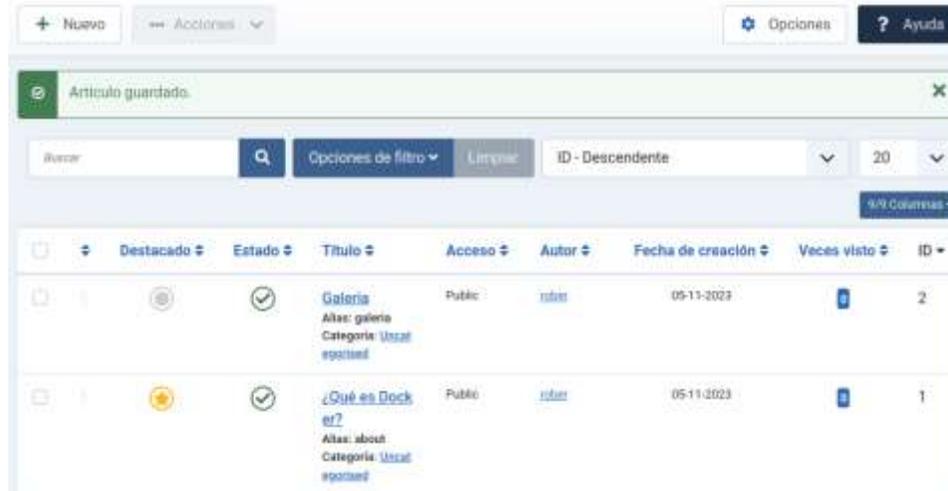


Ilustración 238: Verificación del nuevo menú



Ilustración 239: Visualización de la galería

3. En el menú debe existir una opción para navegar a una página de noticias(feeds)

- seguiremos los mismos pasos del inciso anterior para la parte de agregar un nuevo elemento al menú principal de la pagina

- pero esta vez el tipo de elemento de menú será "URL embebida"



The screenshot shows a configuration panel with the following fields and options:

- Tipo de elemento del menú:** "Mostrar una página embebida" (Selected)
- URL:** "https://www.itmastermag.com/noticias/analisis/que-son-los-contenedores-de-software-y-c" (Truncated)
- Enlace:** "index.php?option=com_wrapper&view=wrapper"
- Abrir en:** "Misma ventana"
- Estilo de la pantalla:** "- Usar la predeterminada -"
- Menú:** "Main Menu"
- Elemento principal:** "- Sin principal -"
- Orden:** "El orden estará disponible después de guardar."

Ilustración 240: Agregando "URL embebida" al menú principal

- luego colocaremos la URL de una página de noticias (en relación a nuestro tema en particular), guardamos lo cambios y revisamos nuestra pagina



Ilustración 241: Visualización del menú de noticias

4. Alguno de los artículos debe tener un video de YouTube

- Primero que nada, debemos conseguir el enlace del video, pero como código fuente, esto nos permite pegarlo en el artículo de nuestro portal web

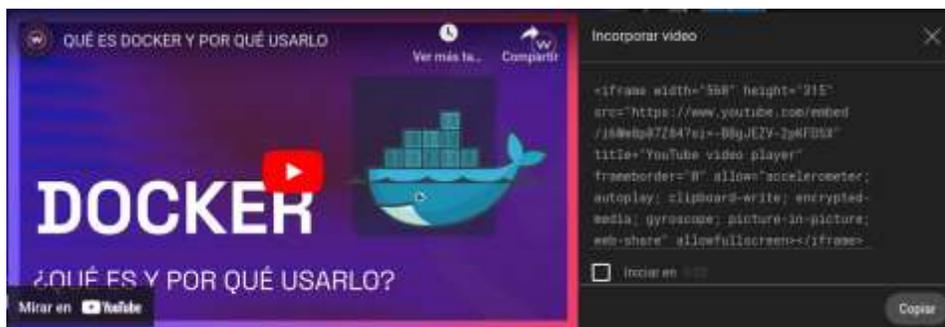


Ilustración 242: Código fuente de un video de YouTube



- En el panel de administración de Joomla, haz clic en Plugin. Encuentra el plugin llamado Editor-TinyMCE y haz clic en él. Cuando la página de configuración se abra, busca el cuadro de texto llamado Prohibited Elements. Verás – script,applet,iframe – escrito dentro del área. Elimina iframe desde este cuadro de modo que el texto final sea: script,applet.

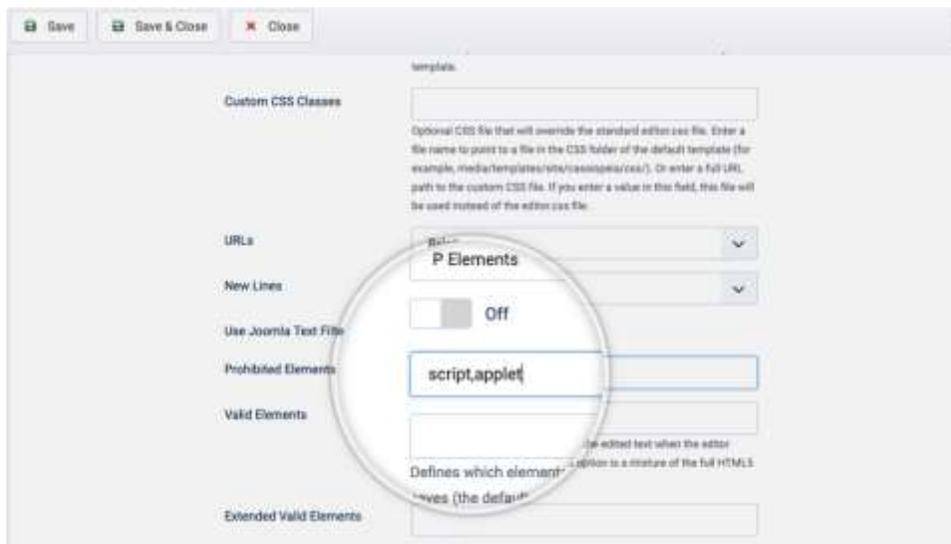


Ilustración 243: Configuración del plugin "Editor-TinyMCE "

- luego una vez en el artículo daremos click en los 3 puntos del editor para desplegar más opciones y daremos click en el botón con icono de video para pegar nuestro enlace

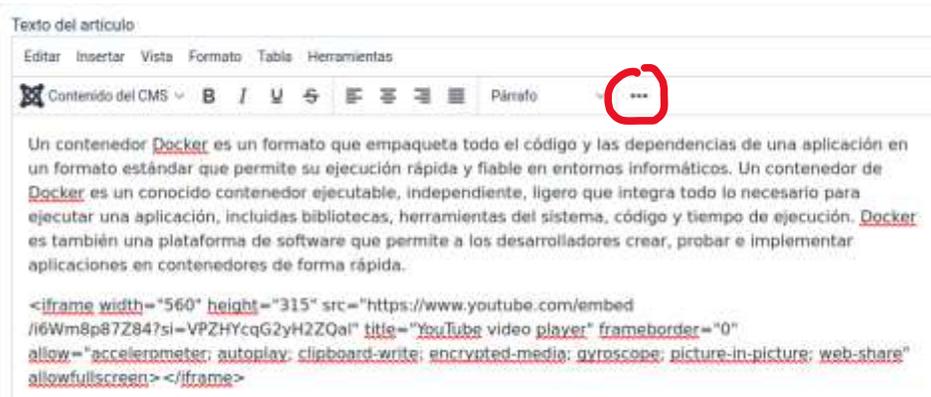


Ilustración 244: Nuevo artículo con video de YouTube

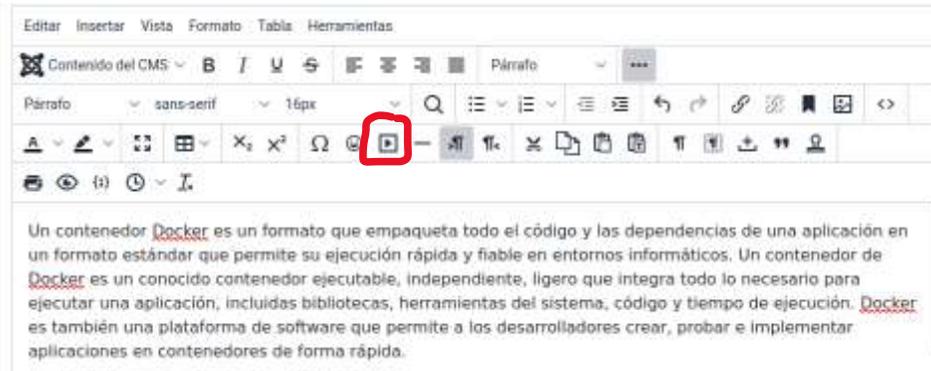


Ilustración 245: Asignación del enlace del video

5. Debe existir al menos 3 tipos de usuarios del sitio

- nos dirigimos a la siguiente dirección

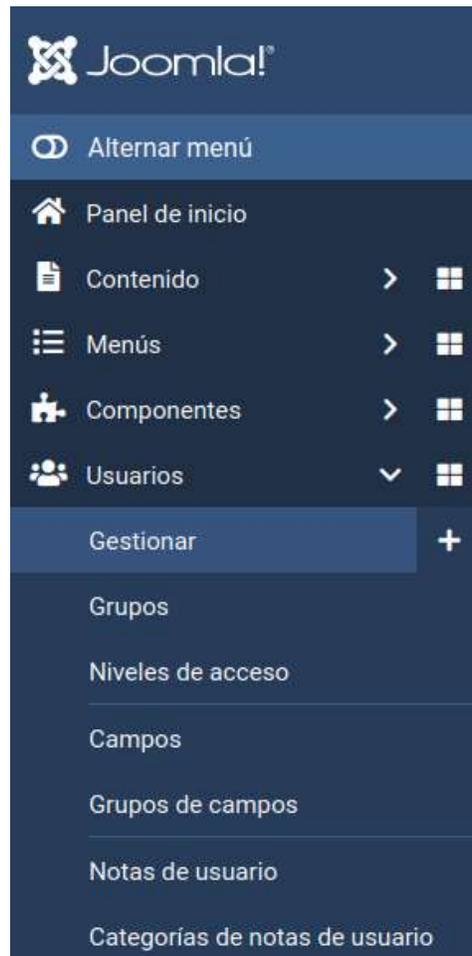


Ilustración 246: Menú de usuarios



- por el momento solo tenemos al super usuario, así que crearemos otros 2 usuarios mas

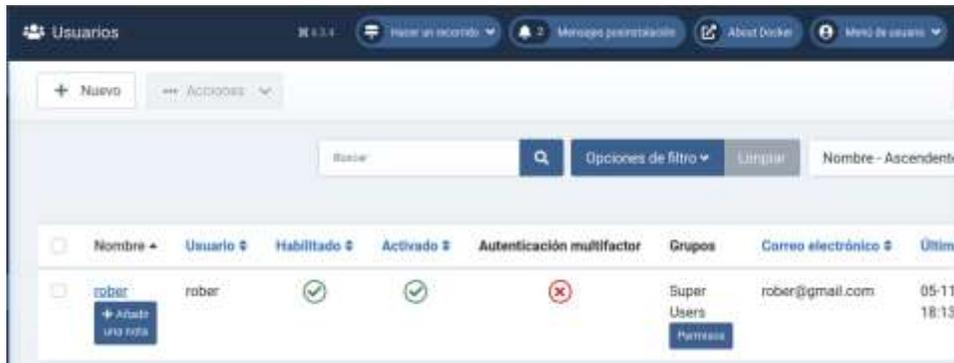


Ilustración 247: Usuarios existentes

- solo es cuestión de dar a nuevo y rellenar todos campos requeridos para cada usuario

- En este caso particular tendremos a un super usuario y 2 usuarios que se encuentran nada más registrados a espera de un tipo de acceso en específico



Ilustración 248: Nuevos usuarios agregados

6. Alguno de los artículos debe tener un mapa de Google Maps

- Para ello accedemos a Google Maps y buscamos una dirección cualquiera en la barra de búsqueda.

- Una vez localizada nuestra geolocalización, en la parte baja de la pantalla veremos el icono de una rueda dentada, en la que haremos click para después seleccionar la opción "Compartir e insertar mapa" y daremos en insertar mapa



- En esta sección nos encontramos el código para embeber un mapa en nuestra publicación. En la pestañita en la que por defecto aparece “Mediano” podemos escoger el tamaño del mapa que queremos publicar entre tres tamaños predeterminados (Pequeño, Mediano y Grande), e incluso podemos darle el tamaño del mapa en píxeles escogiendo la opción “Tamaño personalizado”.
- Una vez tengamos el código copiado, lo tendríamos que añadir en nuestra publicación vía texto. Para ello, hacemos click en el botón “Alternar editor” de nuestra entrada para que este nos permita introducir texto con etiquetas e introducimos el código copiado en Google Maps en el lugar escogido.

```
Texto del artículo
<p>Un contenedor Docker es un formato que empaqueta todo el código y las dependencias de una aplicación en un formato estándar que permite su ejecución rápida y fiable en entornos informáticos. Un contenedor de Docker es un conocido contenedor ejecutable, independiente, ligero que integra todo lo necesario para ejecutar una aplicación, incluidas bibliotecas, herramientas del sistema, código y tiempo de ejecución. Docker es también una plataforma de software que permite a los desarrolladores crear, probar e implementar aplicaciones en contenedores de forma rápida.</p>
<p><iframe src="https://www.youtube.com/embed/i6Wm8p87Z84?si=NGahNnTN9k8t3hpo" width="560" height="314" allowfullscreen="allowfullscreen"></iframe></p>
<p></p>
<p></p>
<iframe src="https://www.google.com/maps/embed?pb=!1m14!1m12!1m3!1d124829.44923327403!2d-86.29452799999999!3d12.1176064!2m3!1f0!2f0!3f0!3m2!1!1024!2!768!4f13.1!5e0!3m2!1ses!2sni!4v1699216088499!5m2!1ses!2sni" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy" referrerpolicy="no-referrer-when-downgrade"></iframe>
```

Ilustración 249: Agregando código de un mapa a un artículo

- y nuestro mapa de Google maps estaría listo en el portal web



7. Se debe crear una página de contactos, que será accedida desde el menú Principal

- Nos dirigimos a la siguiente sección



Ilustración 250: Menú de contactos

- de momento no hay ningún contacto disponible y tendremos que agregar nuevos contactos que luego se mostraran en una sección del menú principal

- solo es cuestión de agregar unos cuantos y rellenar los datos que se nos solicitan para agregar nuevos contactos

Ilustración 251: Información para un nuevo contacto



- A estos contactos se le pueden enlazar usuarios existentes
- una vez hecho los contactos nos dirigimos al menú principal para agregar un nuevo elemento (a como se ha hecho en incisos anteriores), solo que esta vez se rellenara la información de la siguiente forma

La imagen muestra la interfaz de configuración de un contacto en Drupal. Se encuentran en la pestaña 'Integración' y 'Tipo de enlace'. El campo 'Tipo de elemento del menú' está configurado en 'Mostrar todos los contactos de una categoría'. 'Seleccione una categoría' está configurado en 'Uncategorised'. El 'Enlace' es 'index.php?option=com_contact&view=category'. 'Abra en' está configurado en 'Misma ventana'. 'Estado de la plantilla' está configurado en 'Usar la predeterminada'. En el panel de la derecha, 'Menú' está configurado en 'Main Menu' y 'Elemento principal' está configurado en '- Sin principal -'. El 'Orden' está configurado en 'El orden estará disponible después de guardar'.

Ilustración 252: Agregando elemento de contactos al menú principal

- Luego guardamos los cambios y ya podemos ver nuestra lista de contactos agregada a los elementos del menú

La imagen muestra la interfaz de la lista de contactos en Drupal. El título de la página es 'CASSIOPEIA'. El breadcrumb indica 'Está aquí: Inicio / Lista de Contactos'. El título de la sección es 'Uncategorised'. La lista de contactos tiene las siguientes columnas: 'Nombre' y 'Detalles'.

Nombre	Detalles
Jesús	Autur Chavezdega, ricaragua
Judith	Tercero Chavezdega, ricaragua
Ruben	Primeri Leon, ricaragua

En el panel de la derecha, se encuentran los menús 'Main Menu' y 'Login Form'. El 'Main Menu' incluye 'Home', 'Galeria', 'Sobre Docker', 'Lista de Contactos' y 'Login Form'. El 'Login Form' incluye campos para 'Usuario' y 'Contraseña', un checkbox para 'Recordarme' y un botón 'Identificarse'.

Ilustración 253: Verificación del menú contactos

Drupal

Instalación

Se mostrará el proceso para la instalación del CMS de Drupal



Para esta instalación, primeramente, descargaremos los archivos de Drupal, para agilizar la instalación. Posterior a la descarga, encenderemos el contenedor

Descargaremos la versión 9.5 de la página oficial, descargamos el archivo.tar. Y este lo descomprimiremos dentro del directorio donde se encuentre el archivo.yml

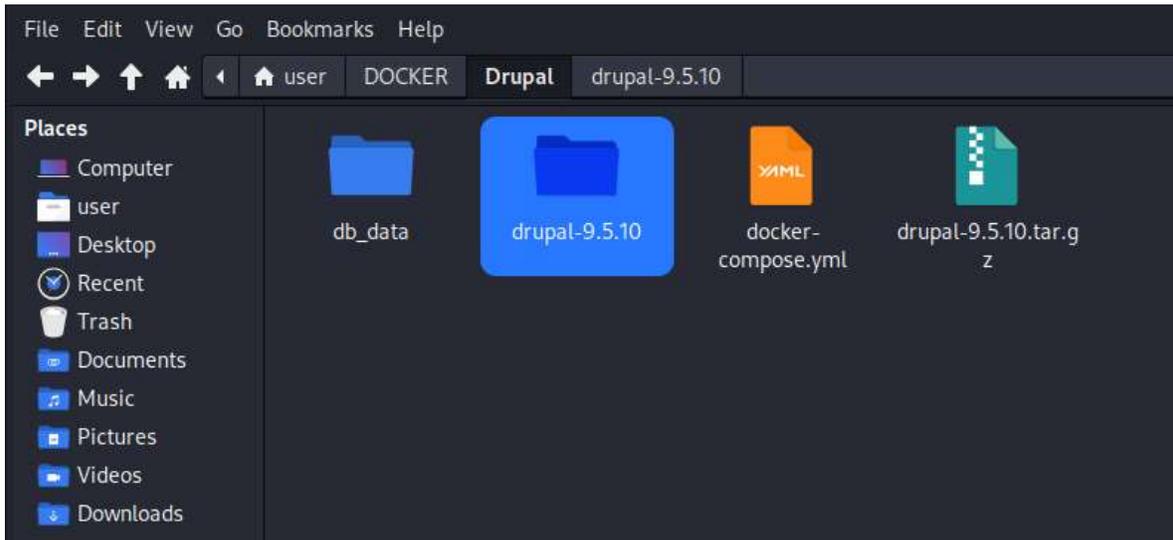


Ilustración 254: Localización de la carpeta de Drupal

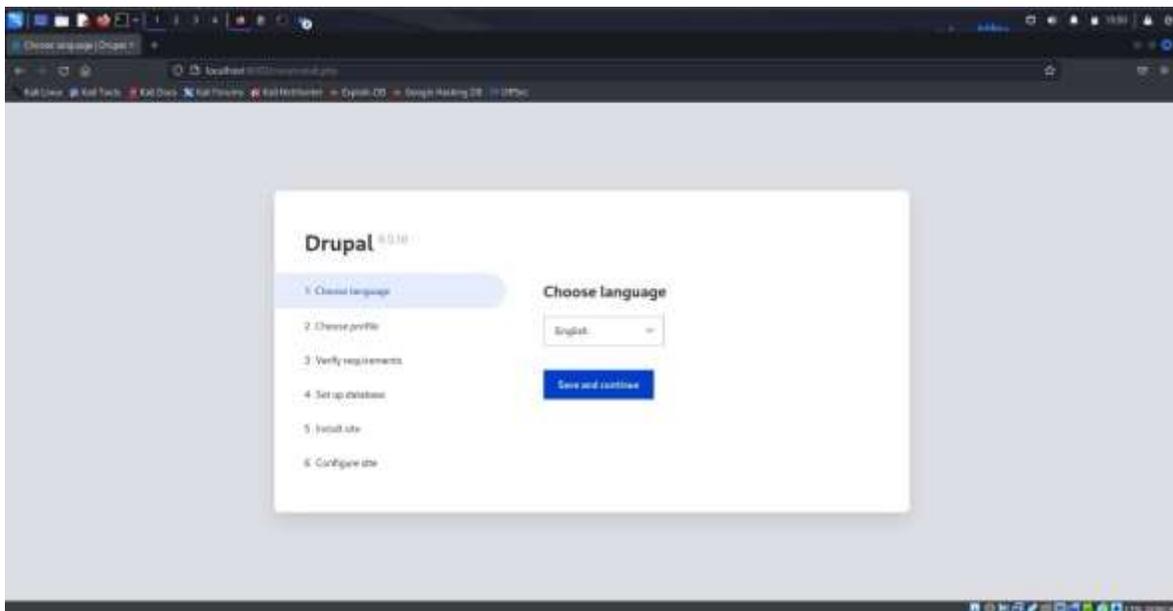


Ilustración 255: Selección de lenguaje

Por defecto escogeremos el lenguaje en inglés, al terminar la instalación se puede instalar el paquete para cambiar el lenguaje a español

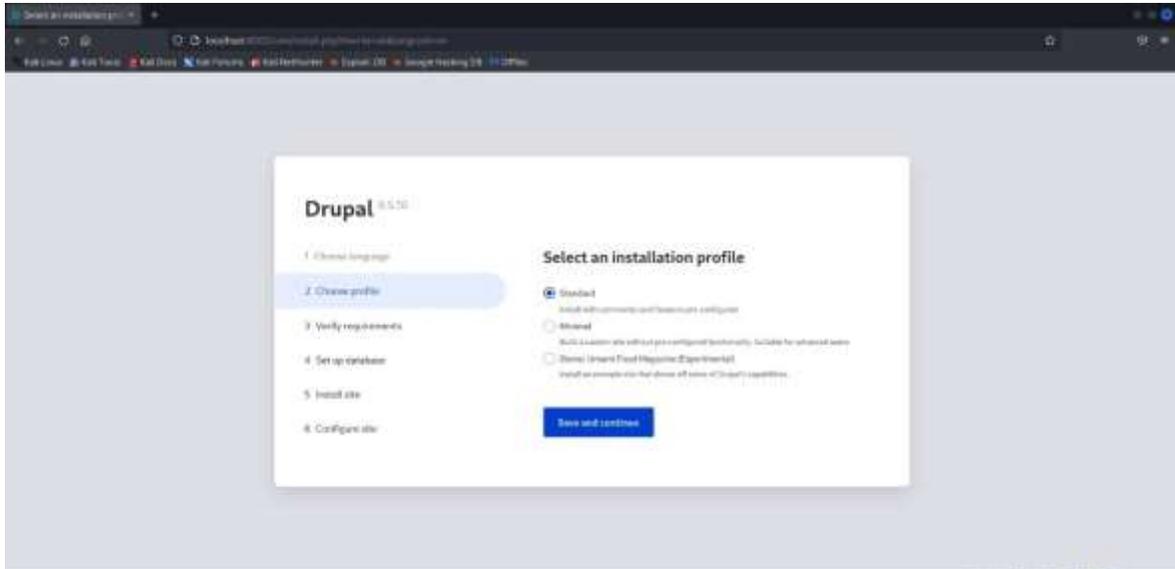


Ilustración 256: Selección del tipo de perfil de instalación

El perfil de instalación se dejará en estándar.

En el siguiente paso nos saldrá los siguientes errores

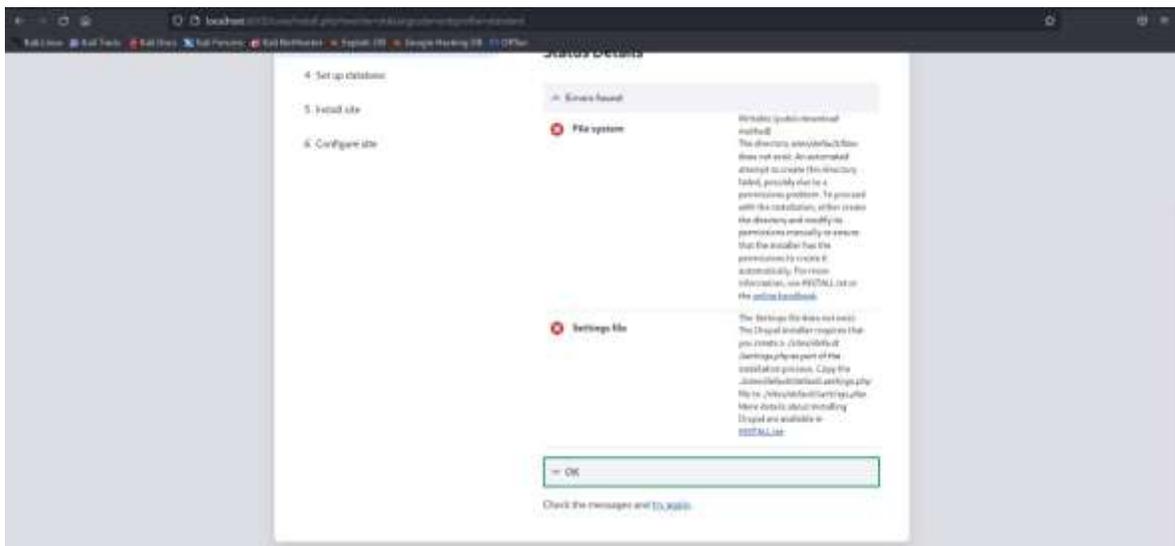


Ilustración 257: Errores al momento de la instalación

Nota: en la descripción de los errores, nos indica que se debe de realizar para continuar con la instalación.

Lo que deberemos realizar a continuación es:



- Ir a la carpeta de drupal (9.5.10): /sites/default/
- Copiar el archivo **default.settings.php** y pegarlo en el mismo lugar con el nombre de **settings.php**
- Crear un directorio llamado **files**
- Dar permisos de escritura y ejecución a la carpeta **sites**

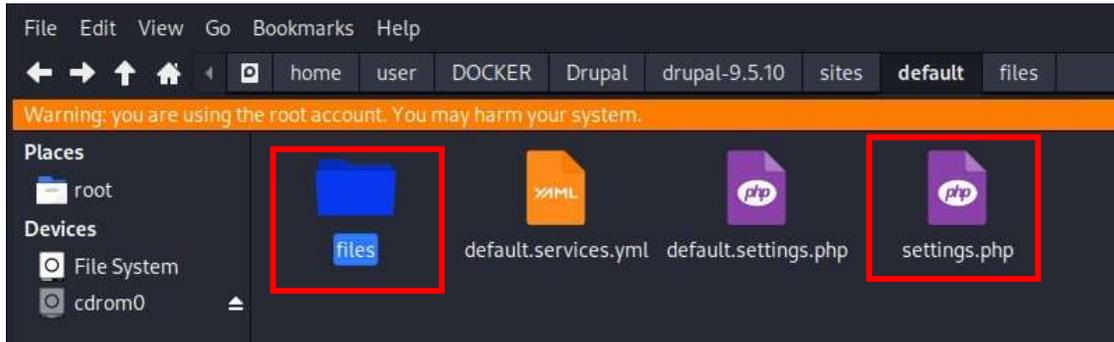


Ilustración 258: Creación de la carpeta files y el archivo settings.php

```
(root@kali) ~ /home/user/DOCKER/Drupal/drupal-9.5.10
# chmod 777 -R sites/*

(root@kali) ~ /home/user/DOCKER/Drupal/drupal-9.5.10
# ls -l
total 256
-rw-r--r-- 1 user user 312 Jul 5 03:57 autoload.php
-rw-r--r-- 1 user user 3460 Jul 5 03:57 composer.json
-rw-r--r-- 1 user user 178823 Jul 5 03:57 composer.lock
drwxr-xr-x 12 user user 4096 Jul 5 03:57 core
-rw-r--r-- 1 user user 1507 Jul 5 03:57 example.gitignore
-rw-r--r-- 1 user user 549 Jul 5 03:57 index.php
-rw-r--r-- 1 user user 94 Jul 5 03:57 INSTALL.txt
-rw-r--r-- 1 user user 18092 Nov 16 2016 LICENSE.txt
drwxr-xr-x 2 user user 4096 Jul 5 03:57 modules
drwxr-xr-x 2 user user 4096 Jul 5 03:57 profiles
-rw-r--r-- 1 user user 3205 Jul 5 03:57 README.md
-rw-r--r-- 1 user user 1786 Jul 5 03:57 robots.txt
drwxr-xr-x 3 user user 4096 Jul 5 03:57 sites
drwxr-xr-x 2 user user 4096 Jul 5 03:57 themes
-rw-r--r-- 1 user user 804 Jul 5 03:57 update.php
drwxr-xr-x 20 user user 4096 Jul 5 03:57 vendor
-rw-r--r-- 1 user user 4039 Jul 5 03:57 web.config

(root@kali) ~ /home/user/DOCKER/Drupal/drupal-9.5.10
# cd sites

(root@kali) ~ /home/.../DOCKER/Drupal/drupal-9.5.10/sites
# ls -l
total 24
drwxrwxrwx 3 user user 4096 Nov 1 13:54 default
-rwxrwxrwx 1 user user 310 Jul 5 03:57 development.services.yml
-rwxrwxrwx 1 user user 5731 Jul 5 03:57 example.settings.local.php
-rwxrwxrwx 1 user user 2353 Jul 5 03:57 example.sites.php
-rwxrwxrwx 1 user user 515 Jul 5 03:57 README.txt

(root@kali) ~ /home/.../DOCKER/Drupal/drupal-9.5.10/sites
#
```

Ilustración 259: Permisos a la carpeta "sites"



Observamos que los directorios en los que se encuentran los ficheros y directorios dentro de la carpeta de **sites** ya tiene los permisos de lectura, escritorio y ejecución.

Siguiendo con el procedimiento de instalación

Drupal 9.5.30

- 1 Choose language
- 2 Choose profile
- 3 Verify requirements
- 4 Set up database
- 5 Install site
- 6 Configure site

Database configuration

Database type*

MySQL, MariaDB, Percona Server, or equivalent

PostgreSQL

SQLite

Database name*

drupal

Database username*

drupal

Database password

Advanced options

Save and continue

Ilustración 260: Configuración de la base de datos

Agregaremos el nombre de la base de datos que hemos especificado en el archivo. yml así mismo como el user, y la contraseña.

5. Install site

Configure site

SITE INFORMATION

Site name*

MARVEL

Site email address*

jason@gmail.com

Automatic emails, such as registration information, will be sent from this address. Use an address ending in your site's domain to help prevent these emails from being flagged as spam.

SITE MAINTENANCE ACCOUNT

Username*

jason

Special special characters are allowed, including space, period (.), hyphen (-), apostrophe ('), underscore (_), and the @ sign.

Password*

Password strength: **Weak**

Confirm password*

Passwords don't: yes

Recommendations to make your password stronger:

→ Make it at least 12 characters

Ilustración 261: Configuración del sitio (1)



Recommendations to make your password stronger:

- Make it at least 12 characters
- Add uppercase letters
- Add punctuation

Email address*

jason@gmail.com

REGIONAL SETTINGS

Default country

Nicaragua

Default time zone

Managua

UPDATE NOTIFICATIONS

Check for updates automatically

Receive email notifications

When checking for updates, anonymous information about your site is sent to [Drupal.org](https://drupal.org).

Save and continue

Ilustración 262: Configuración del sitio (2)

Finalmente, indicaremos el nombre de nuestro sitio, así como los datos del usuario que usaremos para administrar nuestra página, el país y el email.

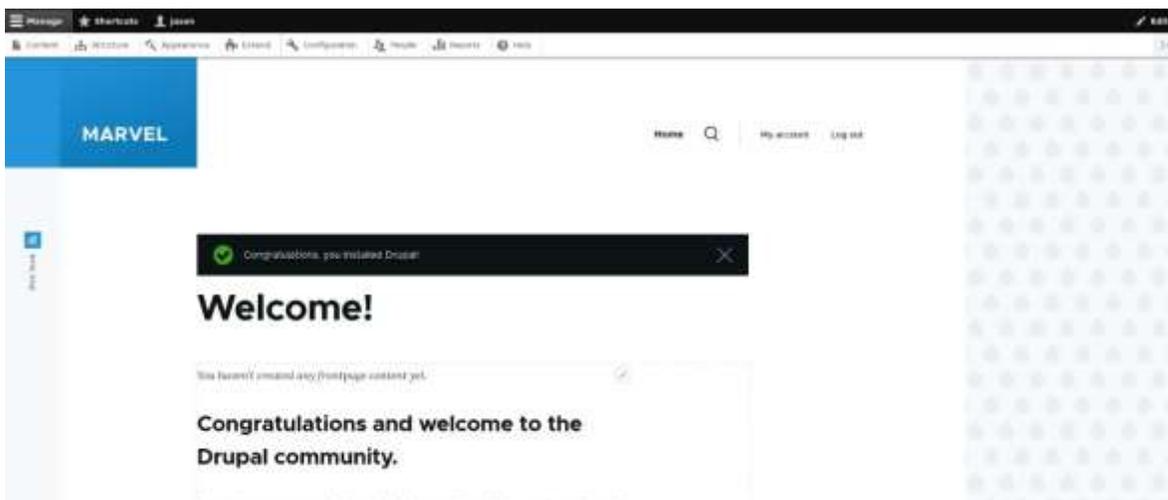


Ilustración 263: Menú principal de Drupal



Backups y restauración

Se mostrará el proceso para realizar un backup de la base de datos de Drupal

Lo primero será identificar el id contenedor de mysql que drupal está utilizando, esto con el comando **docker ps -a**

```
File Actions Edit View Help
└─# docker ps -a
CONTAINER ID   IMAGE                                COMMAND                                CREATED        STATUS
0d949ba2ae4c   phpmyadmin/phpmyadmin              "/docker-entrypoint..."            7 days ago    Up 31 minutes
acfc71868350   drupal:9.5.10                      "docker-php-entrypoi..."          7 days ago    Up 2 hours
a3a99746625a   mysql:5.7.8                         "/entrypoint.sh mysql..."         7 days ago    Up 31 minutes
```

Ilustración 264: Identificar ID del contenedor de mysql

Una vez identificado el Id, procederemos a ingresar al contenedor con el siguiente comando:

El comando utilizado es: **docker exec -ti *id_contenedor* /bin/bash**

El parámetro -t se utiliza para un pseudo-tty, el parámetro -i se usa de manera interactiva. También se puede especificar un usuario, este debe existir en el contenedor, y se utiliza el parámetro -u, su estructura sería: **docker exec -ti -u *usuario* *id_contenedor* /bin/bash**

```
(root@kali) - [~/home/user/DOCKER/Drupal]
└─# docker exec -ti a3a997 /bin/bash
root@a3a99746625a:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.8-rc MySQL Community Server (GPL)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| drupal |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.00 sec)
```

Ilustración 265: Accediendo al contenedor de mysql



Una vez dentro del contenedor, podemos observar que hemos ingresado como root, de forma inmediata ingresamos a mysql, y verificamos rápidamente que exista la base de datos de Drupal, que se observa que tiene el nombre que le hemos especificado en el archivo. yml.

Una vez verificado la base de datos, saldremos de la terminal de mysql y procederemos a crear el backup.

El comando a utilizar: `mysqldump -u root -p nombre_bd > nombre_backup.sql`

```
root@31a99746625a:/# ls
bin boot dev docker-entrypoint-initdb.d entrypoint.sh etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
root@31a99746625a:/# mysqldump -u root -p drupal > backup.sql
Enter password:
root@31a99746625a:/# ls
backup.sql bin boot dev docker-entrypoint-initdb.d entrypoint.sh etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
root@31a99746625a:/#
```

Ilustración 266: Creando backup de drupal

De primera instancia observamos como ejecutamos un ls, esto para verificar que no hay un archivo.sql creado. Al momento de ejecutar el comando, nos pide contraseña de root, contraseña que hemos especificado en el archivo.yml.

Una vez ha terminado de crear el backup, hemos ejecutado el comando ls para verificar que se ha creado nuestros backup.

Para probar el punto de restauración, añadiremos un artículo en Drupal, para luego restaurar la base de datos y ver como se restaura al punto de donde creamos el backup

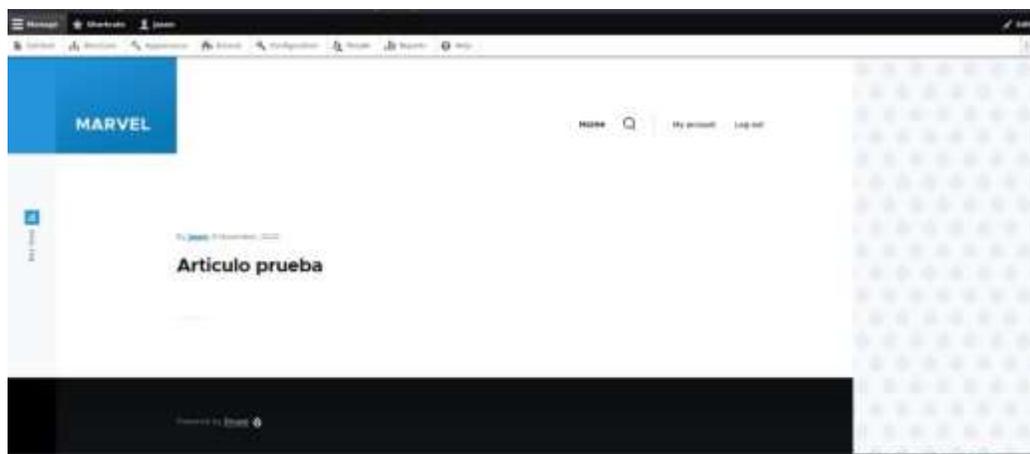


Ilustración 267: Artículo de prueba



Hemos añadido este artículo de prueba.

Nos devolveremos a la terminal del contenedor y ejecutaremos el siguiente comando:

mysql -u root -p -D drupal < backup.sql

```
root@a3a99746625a:/# ls
backup.sql bin boot dev docker-entrypoint-initdb.d entrypoint.sh
root@a3a99746625a:/# mysql -u root -p -D drupal < backup.sql
Enter password:
root@a3a99746625a:/#
```

Ilustración 268: Cargando backup de drupal

Una vez, terminado el proceso de restauración, recargaremos la página de Drupal, y veremos que se restaurara hasta el punto donde se creó el backup.

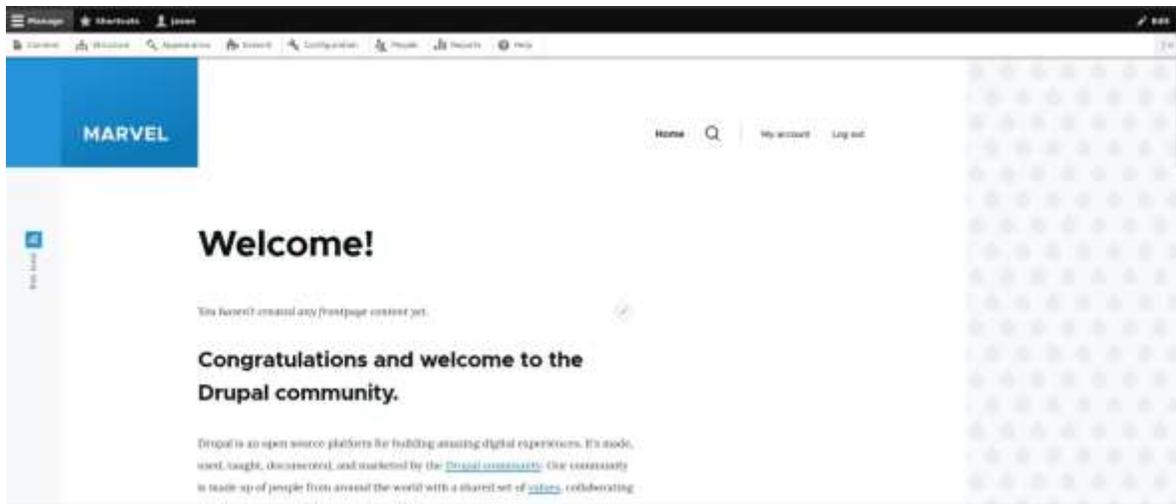


Ilustración 269: Menú principal sin el artículo de prueba

Enunciados

1. Artículos y páginas

El objetivo de este ejercicio es conocer los dos tipos de contenidos básicos de Drupal: artículos y páginas básicas. Para ello, se propone al alumno:

- Leer la explicación sobre artículos y páginas básicas de la lección Drupal.



- Crear algunos artículos y páginas básicas, probando las opciones que permite Drupal al crearlas. El contenido de los artículos y páginas básicas puede ser cualquiera.
- Eliminar las páginas básicas y artículos de prueba creados en el punto anterior. Básicas

2. Roles y usuarios

Drupal permite crear usuarios y concederles diferentes posibilidades (permisos) de edición en el portal. Para facilitar la organización de los permisos otorgados a los diferentes usuarios, Drupal permite crear categorías de usuarios (denominadas Roles).

Cree los roles:

- Minieditor: que pueda editar comentarios, publicar comentarios, omitir aprobación de comentarios

Cree los usuarios:

- minieditor_1: agregar el rol minieditor y contraseña minieditor_1

3. Subir imágenes y archivos: módulo IMCE

En este ejercicio se instalará un módulo para subir archivos, concretamente, el módulo IMCE.

- Descargue la versión 1.9 del módulo IMCE para Drupal 7
- Entre en Drupal como usuario admin y active el módulo IMCE.
- Compruebe el módulo IMCE

Soluciones

1. Artículos y paginas

Para la creación de estos puntos, nos dirigiremos al apartado de content

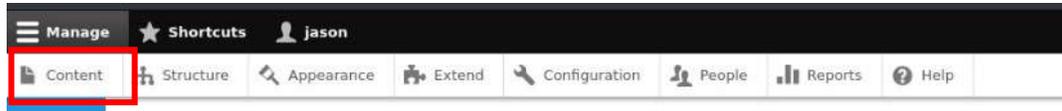


Ilustración 270: Panel de "content"

Add content, y nos mostrará que deseamos publicar.

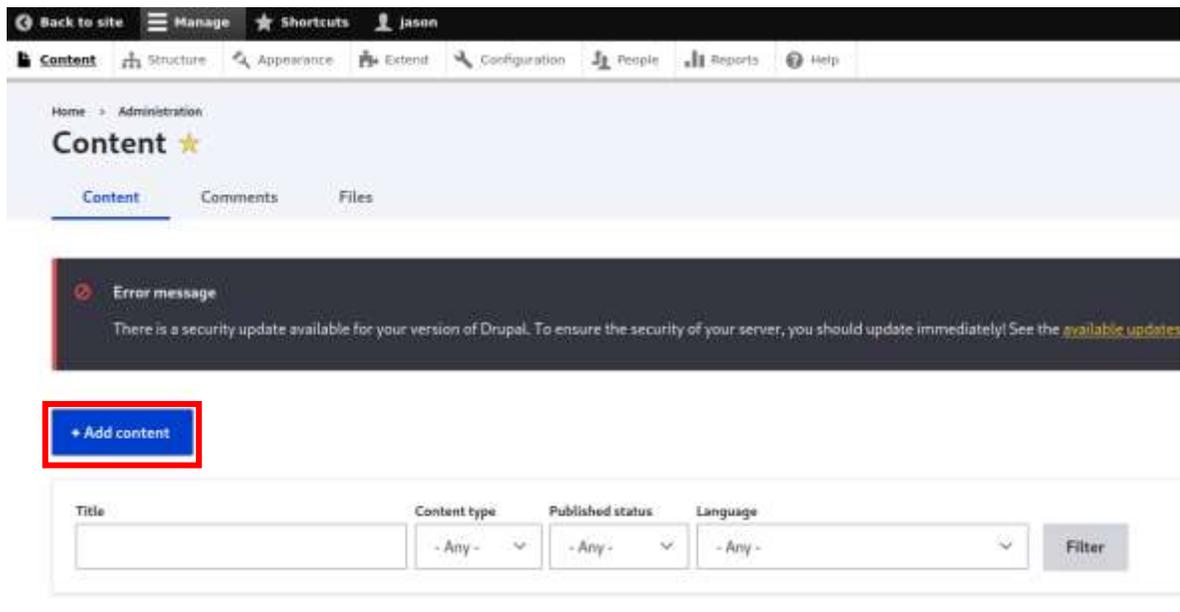


Ilustración 271: Agregando contenido

> Article

Use *articles* for time-sensitive content like news, press releases or blog posts.

> Basic page

Use *basic pages* for your static content, such as an 'About us' page.

Ilustración 272: Nuevo artículo y nueva página

Procederemos a la creación de los enunciados

Se ha creado un artículo de prueba



Ilustración 273: Verificación de artículo nuevo creado

Se ha creado una página de prueba



Ilustración 274: Verificación de la nueva página creada

En este apartado se muestra la página principal de Drupal con la publicación de ambos apartados creados anteriormente

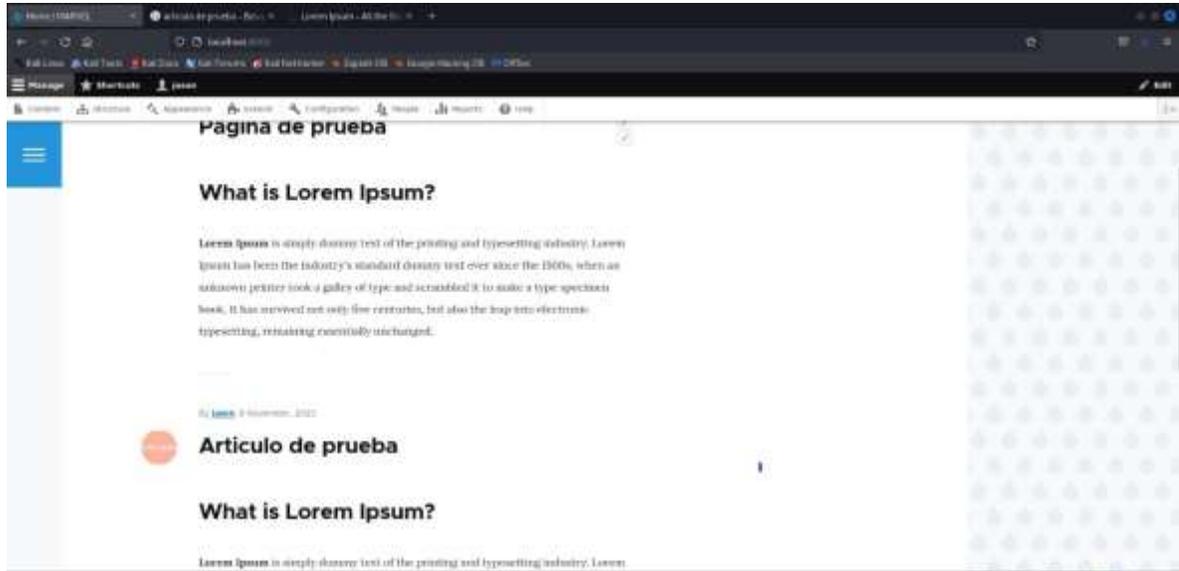


Ilustración 275: Menú principal de Drupal con los nuevos elementos creados

2. Roles y usuarios

Creación del rol minieditor

Nos dirigimos al apartado people y nos ubicamos en la sección de role, y en la opción de añadir rol

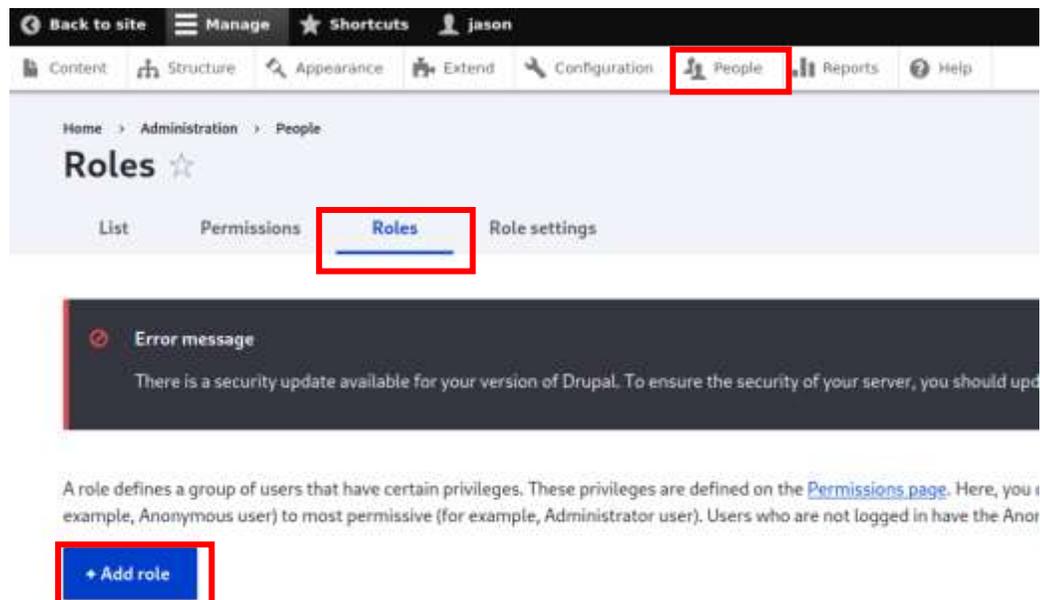


Ilustración 276: Agregando roles



Una vez añadido el nombre del rol, editaremos los accesos que tendrá

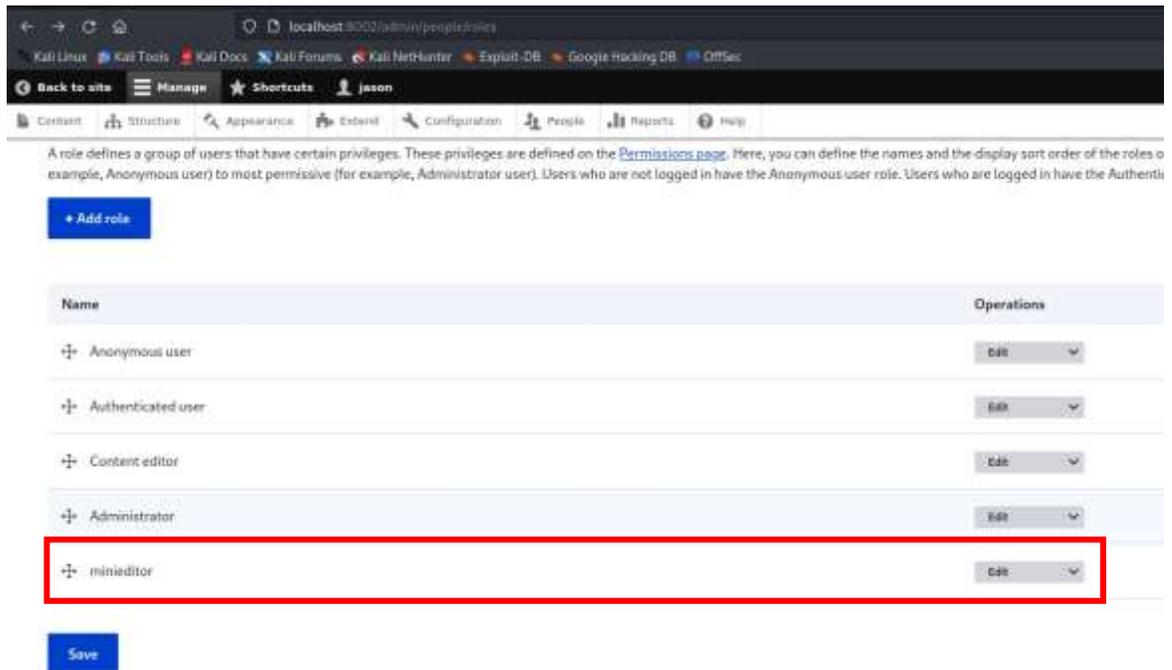


Ilustración 277: Nuevo rol creado

Accedemos a la opción de edit, y agregamos los accesos del enunciado

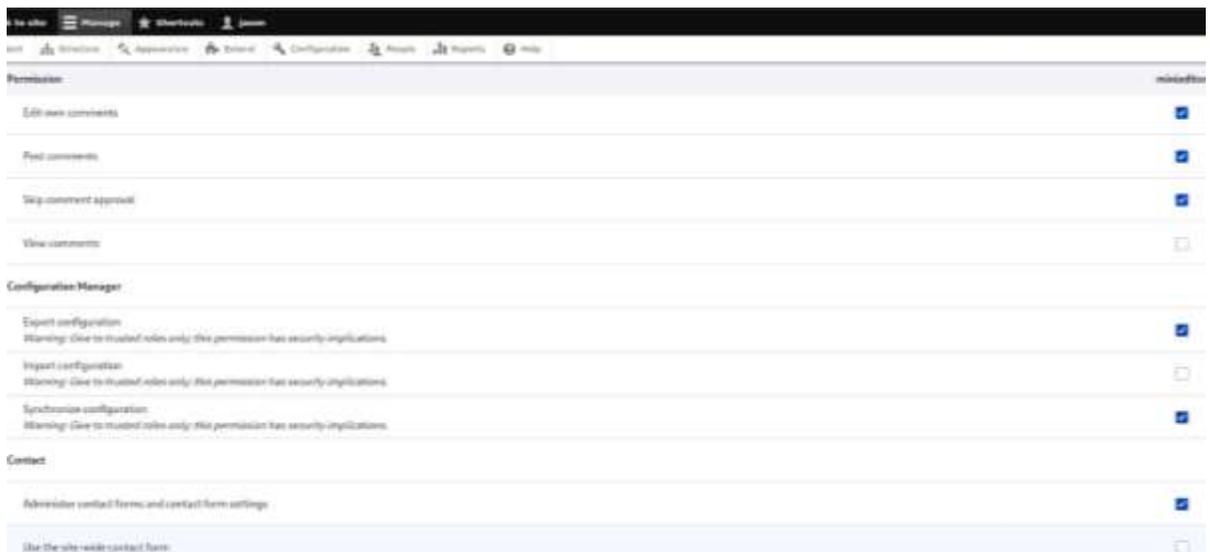


Ilustración 278: Accesos del rol "minieditor"

Hay diversos accesos, por si se requiere crear más roles y que cada uno tenga diferentes accesos.



Creación del usuario minieditor_1

Nos dirigiremos a la pestaña de List

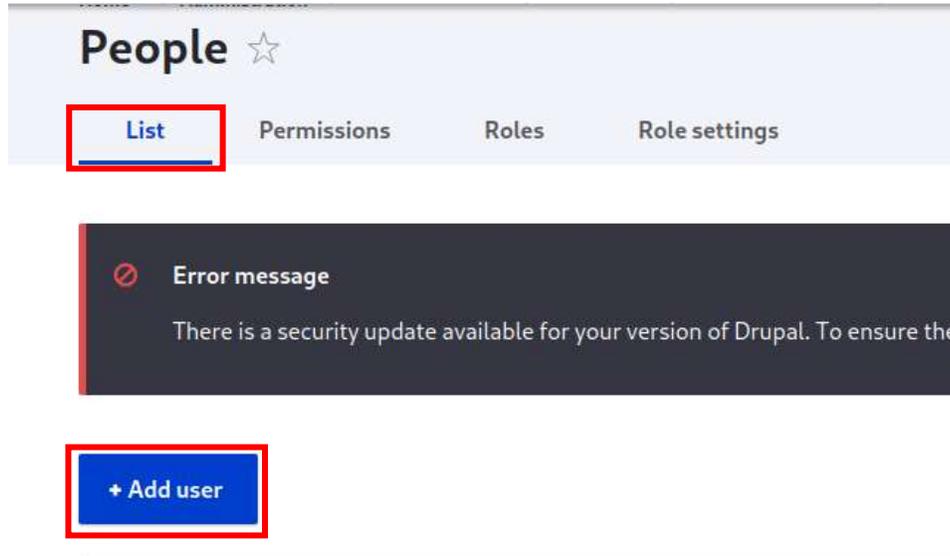


Ilustración 279: Agregando usuarios

Una vez abierto el formulario, rellenaremos los campos según el enunciado o de la necesidad de cada uno

Back to site | Manage | Shortcuts | Jason

Content | Structure | Appearance | Extend | Configuration | People | Reports | Help

Email address
minieditor_1@drupal.com
The email address is not made public; it will only be used if you need to be contacted about your account or for optional notifications.

Username*
minieditor_1
Several special characters are allowed, including space, period (.), hyphen (-), apostrophe ('), underscore (_), and the @ sign.

Password*
●●●●●●●●
Password strength: Strong

Confirm password*
●●●●●●●●

Passwords match: yes
For recommendations to make your password stronger:
+ Add punctuation
Provide a password for the new account in both fields.

Status
 Blocked
 Active

Roles

Ilustración 280: Datos del nuevo usuario



En la creación del usuario, se le especificará que rol tendrá, en este caso el rol que creamos anteriormente **minieditor**

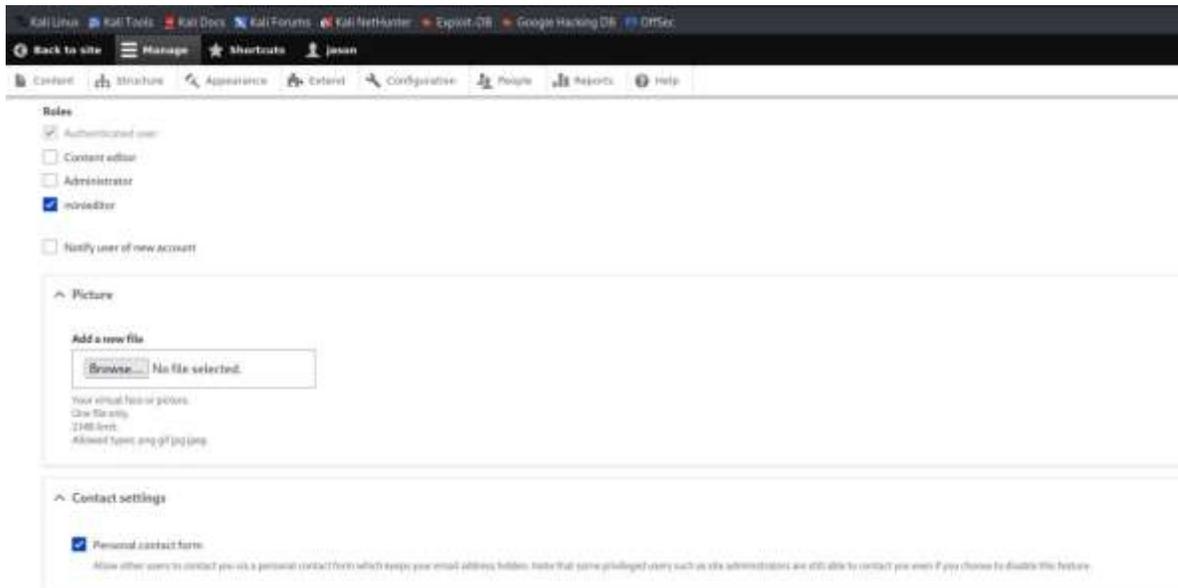


Ilustración 281: Rol del nuevo usuario

Una vez finalizado, en la pantalla de List, nos muestra la lista de todos los usuarios

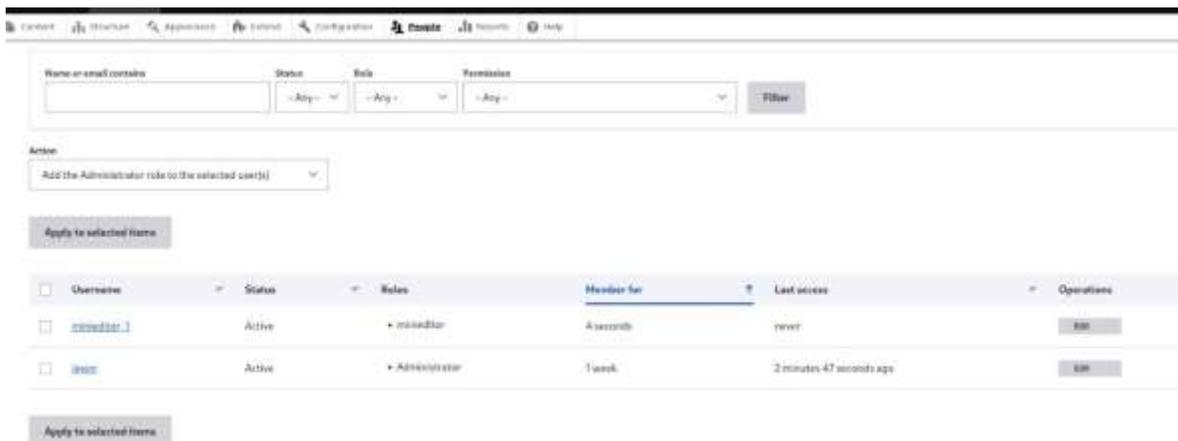


Ilustración 282: Usuarios existentes



Ya una vez creado el usuario, nos podemos logear en esta página de Drupal

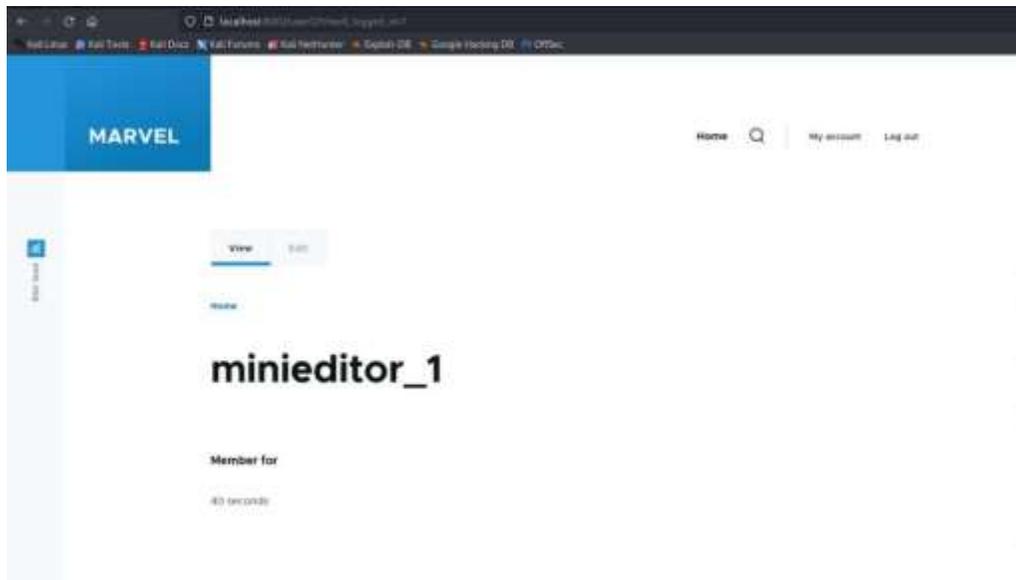


Ilustración 283: Sesión iniciada con el nuevo usuario

3. Subir imágenes y archivos: modulo IMCE

Para la instalación del módulo, lo descargaremos de la página de drupal

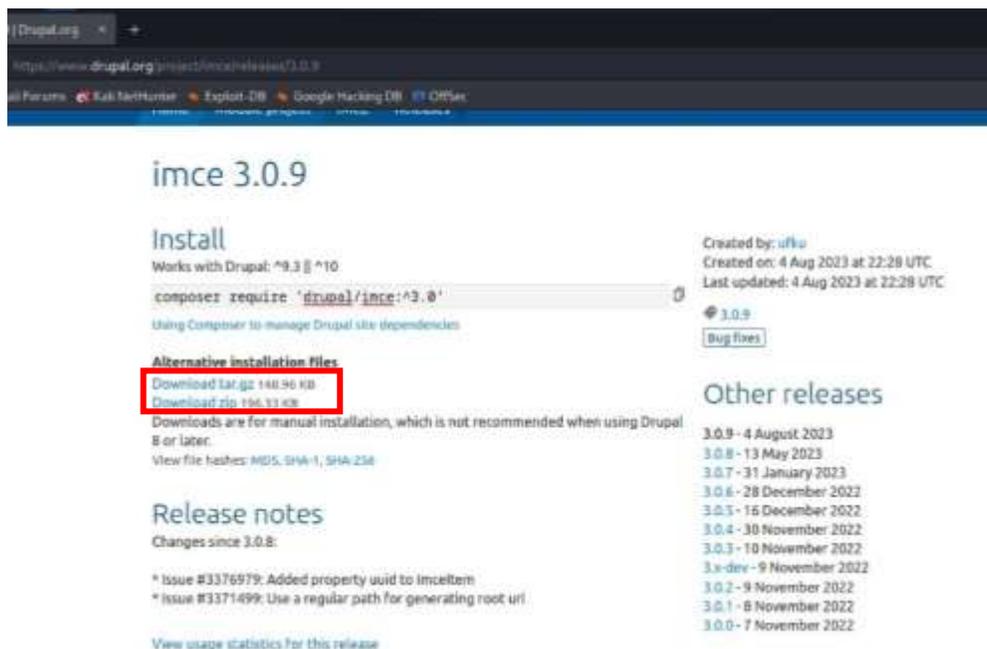


Ilustración 284: Descarga del módulo IMCE



Descargaremos la opción.tar.gz

Este archivo lo moveremos en la carpeta de los archivos de Drupal, en la carpeta de módulos y descomprimos el archivo.

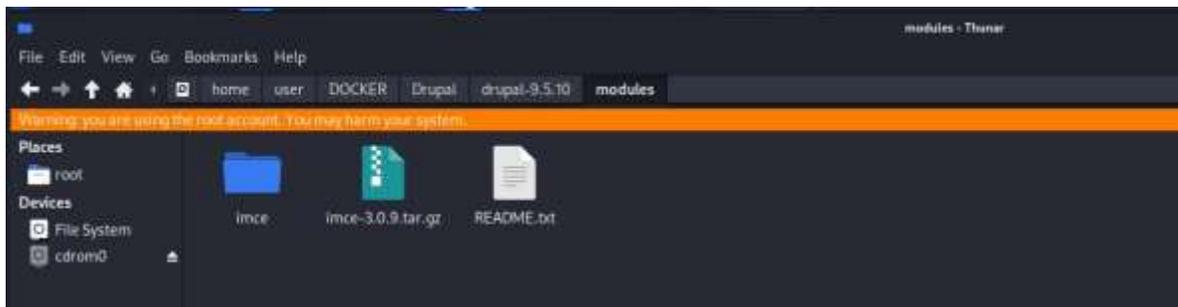


Ilustración 285: Ubicación del módulo IMCE en el administrador de archivos

Una vez hecho esto, nos iremos a la sección de Extend, y buscaremos el módulo de IMCE

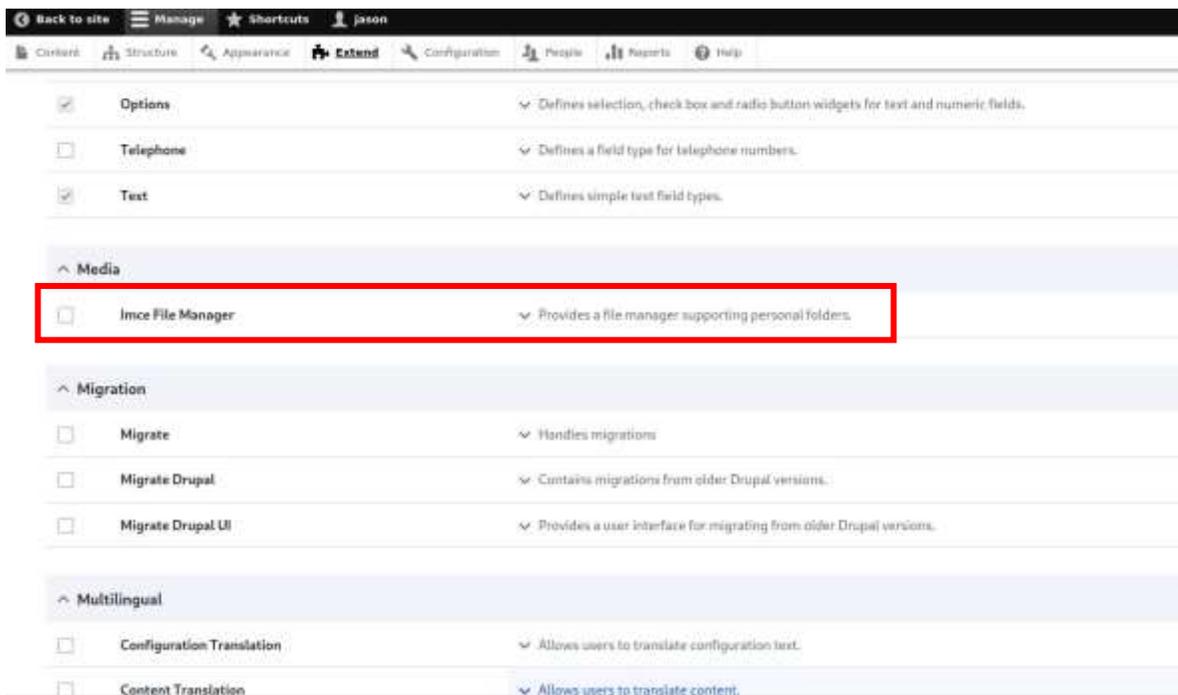


Ilustración 286: Ubicación del módulo IMCE en Drupal

Los seleccionamos y lo instalamos. Si la instalación termino correctamente nos saldrá una notificación exitosa.



Ilustración 287: Instalación del módulo IMCE

Para agregar este módulo, al momento de una edición de artículo o página.

Nos dirigiremos a configuración en el panel superior y luego en buscamos la opción Content Authoring

Y editaremos el formato de HTML básico

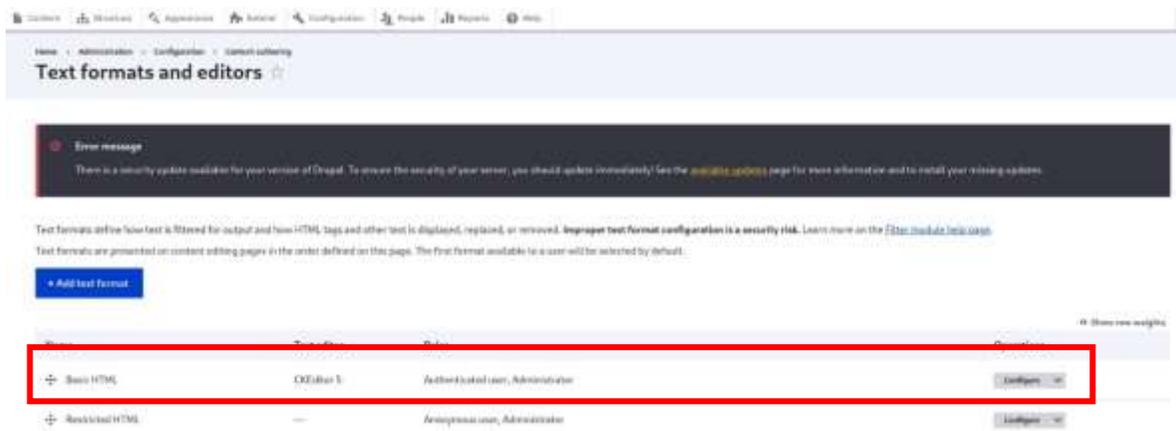


Ilustración 288: Ubicación del formato "HTML básico"

En el apartado de CKEDITOR 5

Nos muestra los botones disponibles y en otra sección los botones activos (estos son los que se muestran al momento de creación de artículos/páginas)



Ilustración 289: Botones disponibles al momento crear un artículo



Aquí ya tenemos disponible el módulo IMCE, en la sección de botones disponibles, lo que se hará es mover el módulo a los botones activos

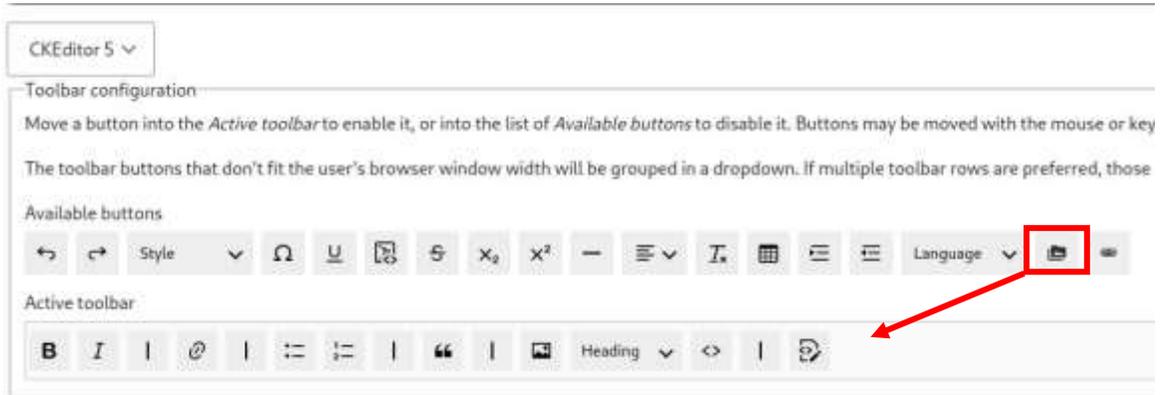


Ilustración 290: IMCE disponible para crear artículos/páginas

Una vez guardado los cambios, verificamos como nos aparece IMCE. Al seleccionarlo nos saldrá una ventana con el menú desplegable de con directorios predeterminados.

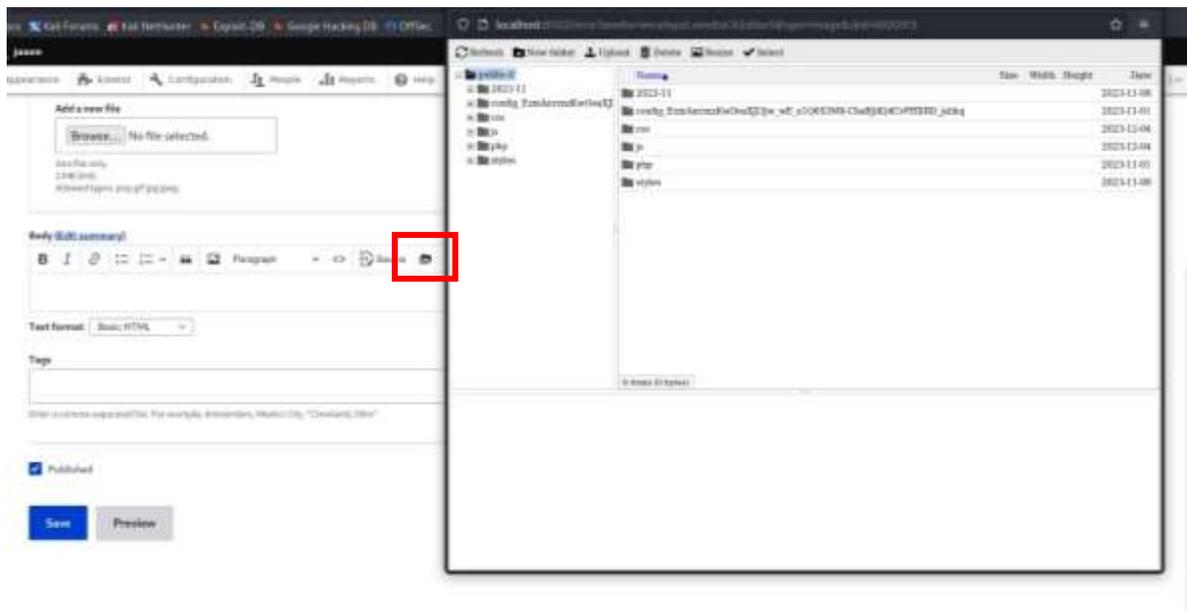


Ilustración 291: IMCE listo para agregar imágenes al momento de crear artículos/páginas



PrestaShop

Instalación

El primer paso básico para la instalación del cms es la selección del idioma, en la mayoría de las versiones aparece con una instalación en inglés, de todos modos, nosotros podemos cambiarlo a español con desplegar el selector

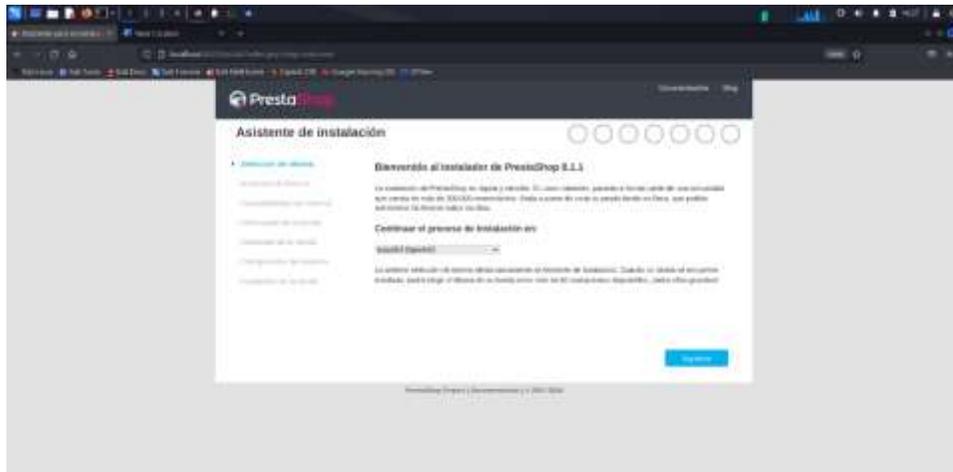


Ilustración 292: Selección de idioma

El segundo paso básico es aceptar los términos y condiciones sobre el uso de este CMS

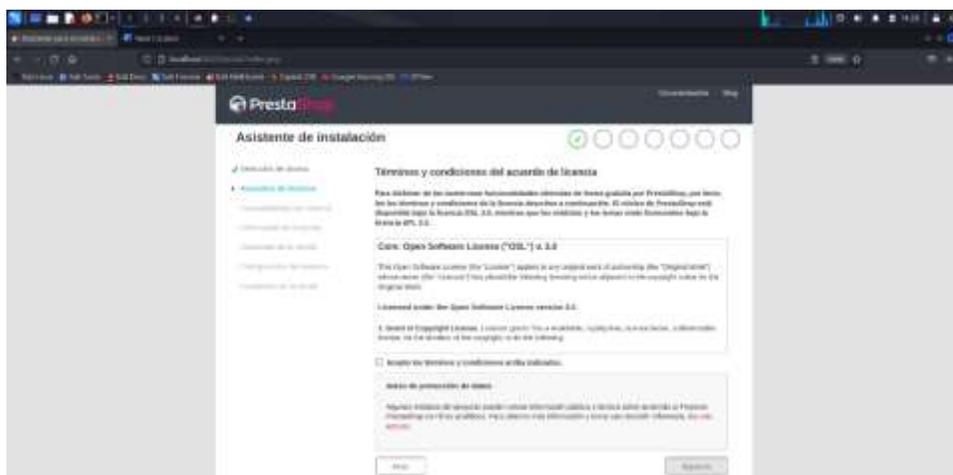


Ilustración 293: Aceptar terminos y condiciones



Tercer paso es agregar información general con respecto a nuestra tienda virtual, información que va desde:

- * Nombre de la tienda
- * Actividad principal por la que se destacara nuestra tienda
- * País/Zona horaria

Mas importante aún es la información de nuestra cuenta con la cual vamos a iniciar sesión en la parte administrativa y comenzar a trabajar en los módulos y productos de la tienda, esta información consiste en:

- * Nombre
- * Apellido
- * correo electrónico
- * Y una contraseña fuerte (debe ser lo suficientemente larga o de lo contrario no nos dejare continuar con la instalacion)

Ilustración 294: Datos de cuenta

Como cuarto paso es meramente decidir si deseamos agregar contenido de prueba como son los módulos y los productos que se muestran en nuestra tienda (esta opción es muy recomendada para los nuevos usuarios)

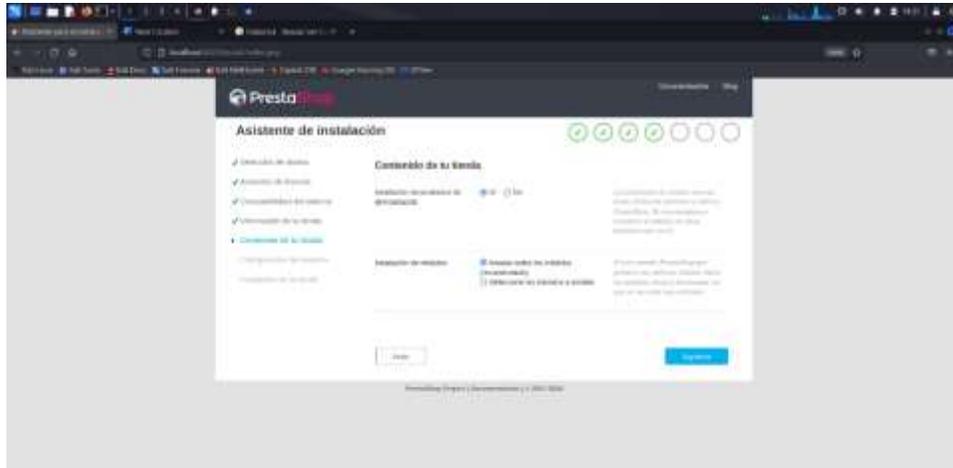


Ilustración 295: Contenido de prueba

El quinto paso es el más importante, ya que aquí debemos configurar la conexión de nuestra base de datos rellenando los campos solicitados

Para que todo funcione correctamente los datos ingresados deben quedar de la siguiente manera

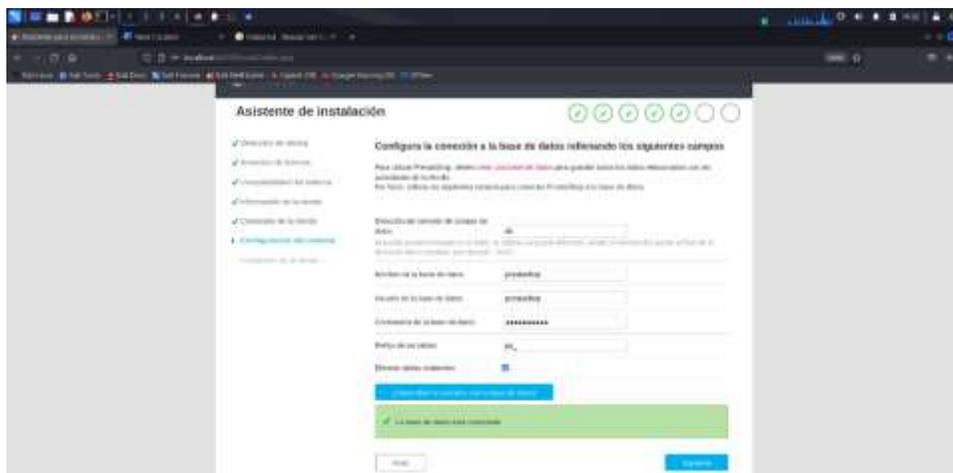


Ilustración 296: Conexión a base de datos

* “db” equivale al servicio de base de datos que estamos empleando

* la contraseña en cuestión es “prestashop” (al igual que los demás datos)

Luego solo comprobaremos la conexión, si todo está bien aparecerá el mensaje de color verde, y con eso culminamos la información de nuestra tienda, solo queda



esperar a que se instale todo (incluyendo los módulos y productos de prueba seleccionados anteriormente)

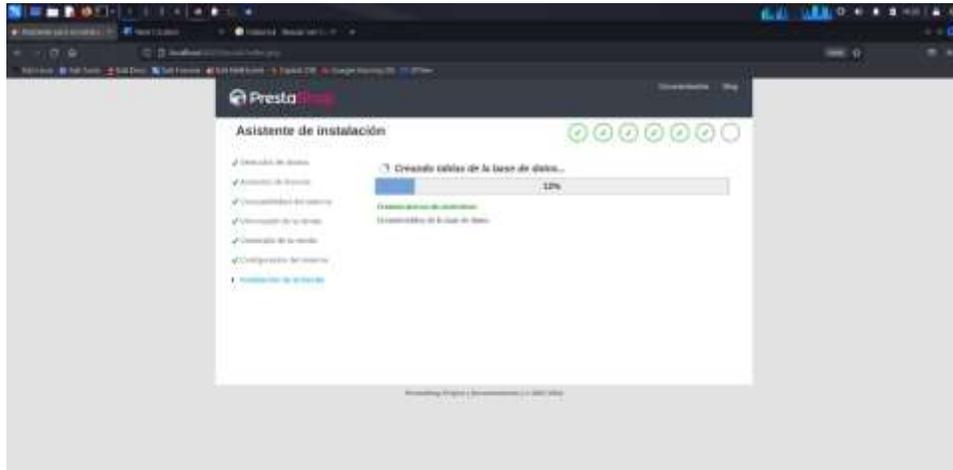


Ilustración 297: Creando sitio

Ya instalado la tienda y la parte administrativa, podemos escoger a cuál queremos ingresar, sin embargo, hay 2 pasos más antes de terminar la instalación (por cuestiones de seguridad)

* La primera es cambiar el nombre de la carpeta “admin”, puede ser a algo sencillo como agregarle numeros o cambiar el nombre totalmente

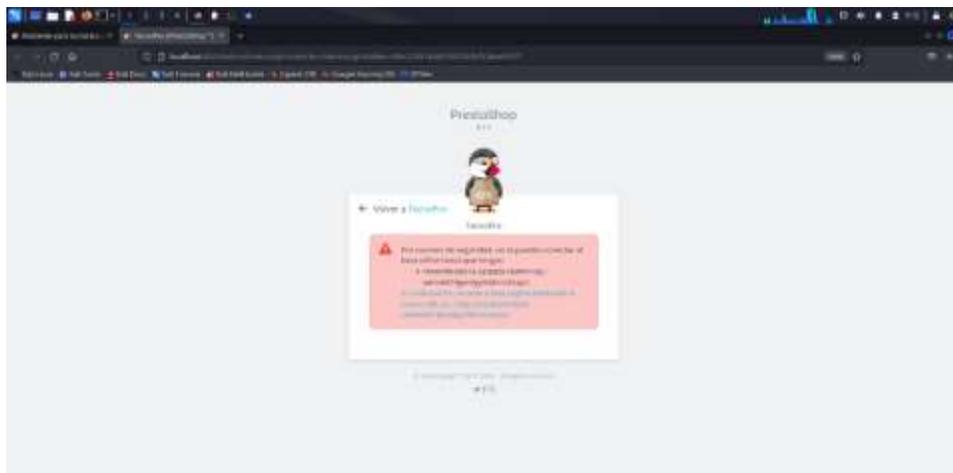


Ilustración 298: Cambiar nombre de carpeta admin



* Eliminar la carpeta “install”

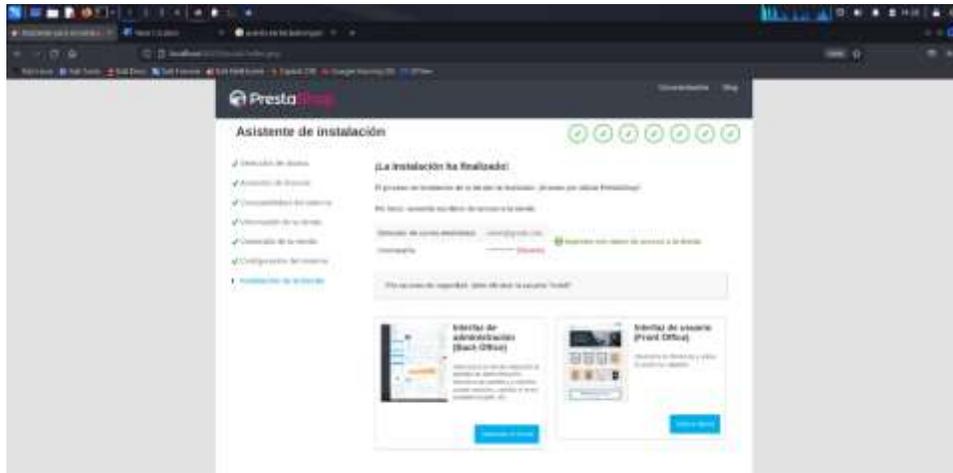


Ilustración 299: Eliminar carpeta Install

Con todo eso hecho ya podemos ver nuestra tienda y lo más importante entrar a la parte administrativa (que estaba bloqueada anteriormente)

Las credenciales son las que fueron agregadas en la instalación

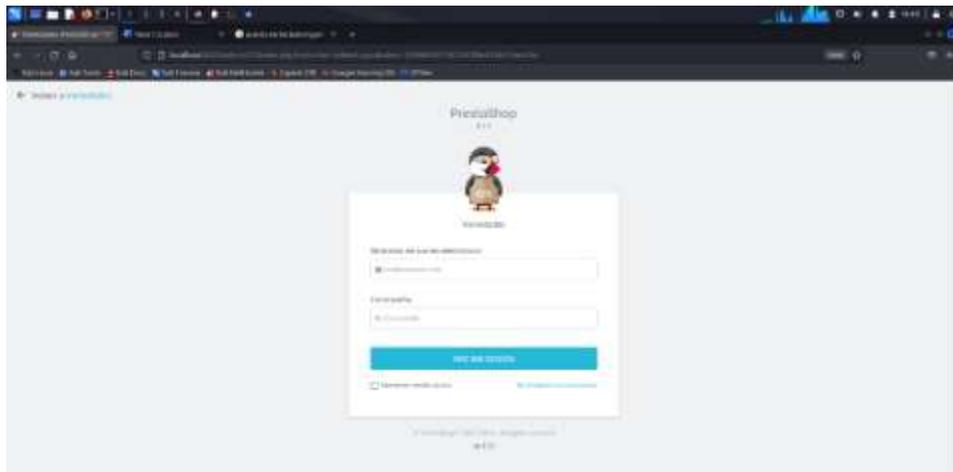


Ilustración 300: Login



Tienda online

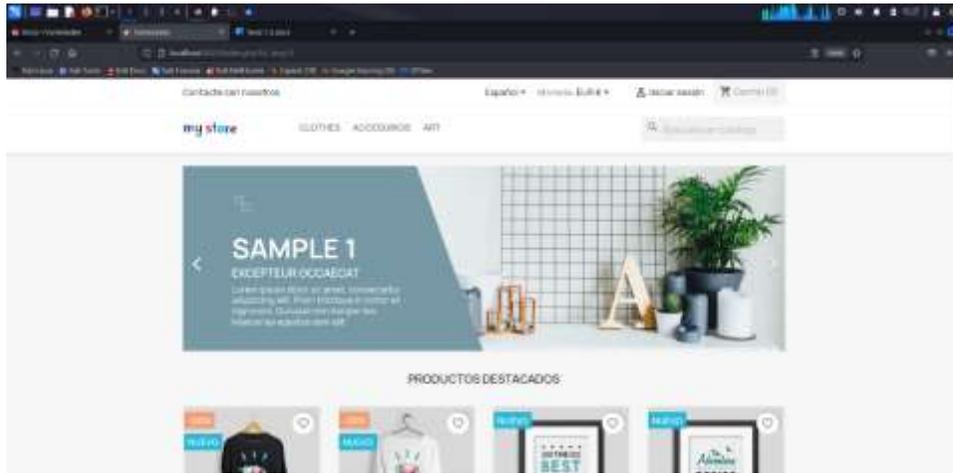


Ilustración 301: Tienda Online

Parte administrativa

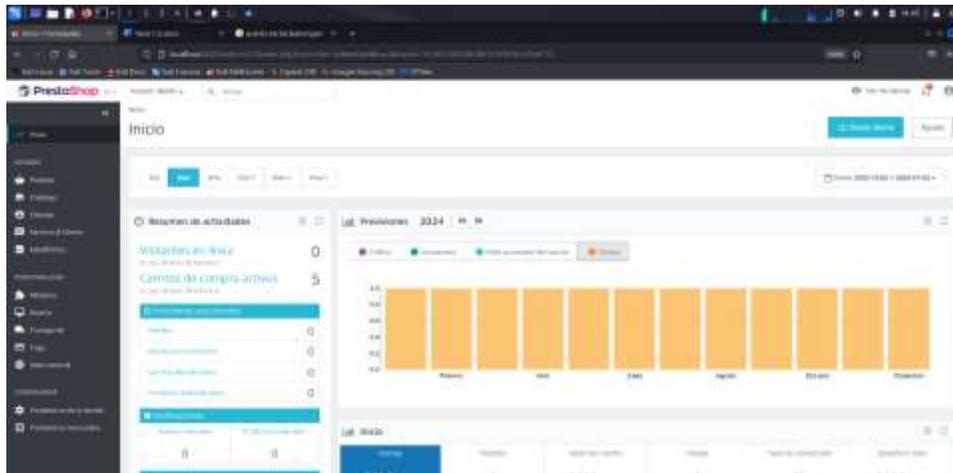


Ilustración 302: Administración

En el menú desplegable de la parte izquierda se encuentra todo lo que necesitamos para trabajar, desde agregar módulos, productos, clientes, marcas etc... Hasta agregar transporte, descuentos y pedidos



Comenzaremos con algo básico

Enunciados:

1. Agregar un producto
2. Clientes
3. Realizar pedidos

Soluciones:

1. Agregar un producto

Nos situamos en la sección de catalogo del menú desplegable a nuestra izquierda y seleccionamos la primera entrada, “productos”

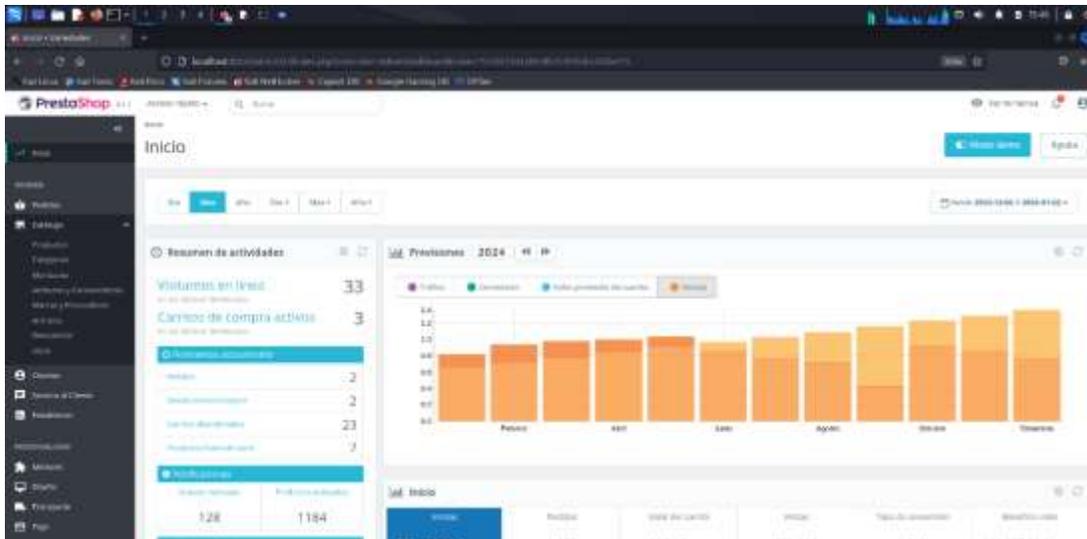


Ilustración 303: Agregar producto

Dentro nos aparecerá una lista de productos ya agregados (que forman parte de los productos de ejemplo que se instaló previamente)

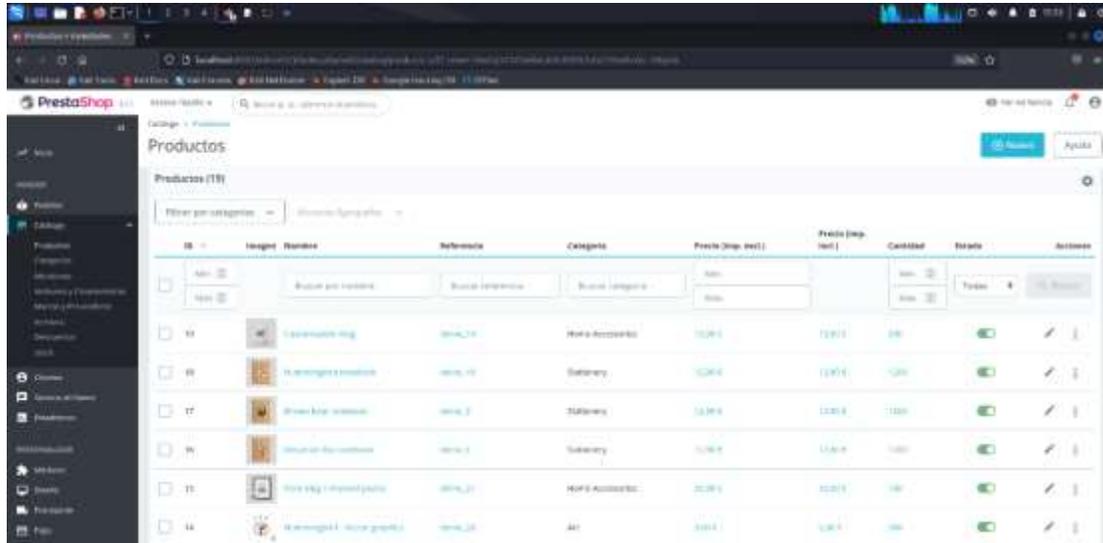


Ilustración 304: Productos predeterminados

A como se puede observar los productos se pueden buscar por diferentes tipos de filtrado y para agregar uno nuevo solo debemos pulsar el botón celeste de la parte superior derecha

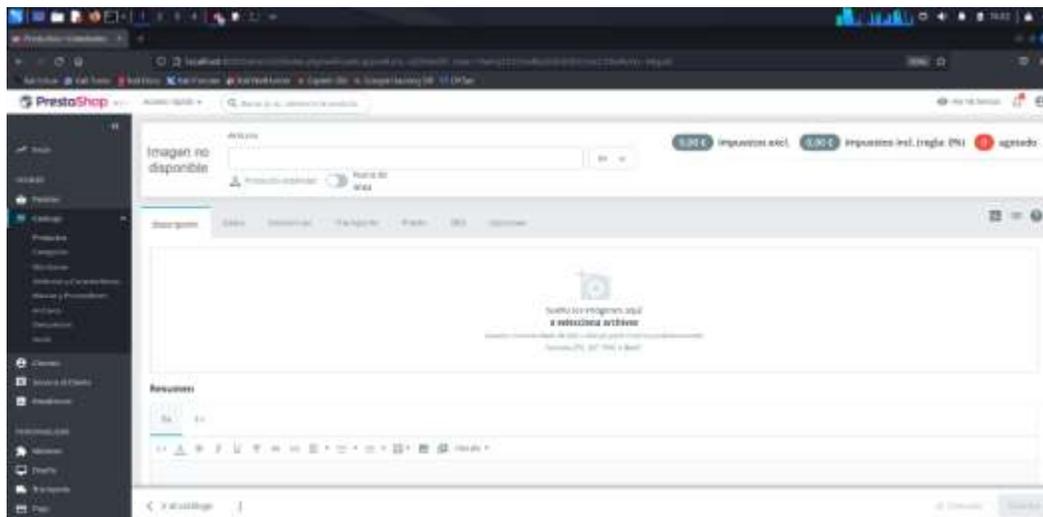


Ilustración 305: Agregar producto

Se debe rellenar algunos parámetros, aunque algunos pueden dejarse en blanco, algunos importantes a destacar tenemos:

- * Nombre del artículo
- * Descripción sobre el



- * Las existencias de dicho producto
- * Foto descriptiva del producto

Entre otros parámetros también están:

- * Marca
- * Categoría
- * Producto relacionado

una vez agregado todo podemos guardar los cambios e ir a ver el nuevo producto agregado (ya sea desde el catálogo de la tienda o de la parte administrativa)

2. Clientes

Ahora continuaremos con la parte de los clientes, que se encuentra de igual forma en el menú desplegable de la izquierda

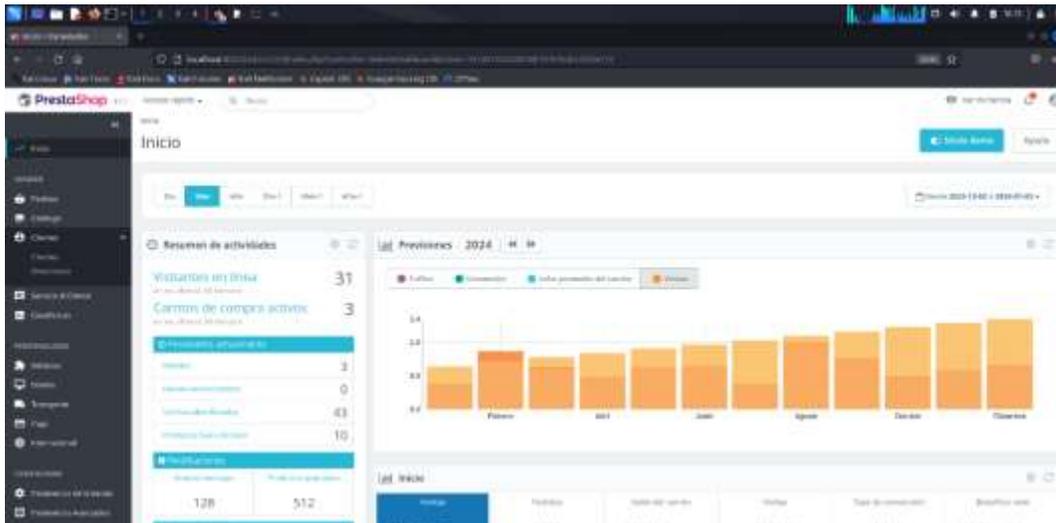


Ilustración 306: Clientes

Como primera instancia podemos ver nuestros 2 clientes de prueba junto con toda la información relevante de estos

Pero también podemos agregar nuestros propios clientes

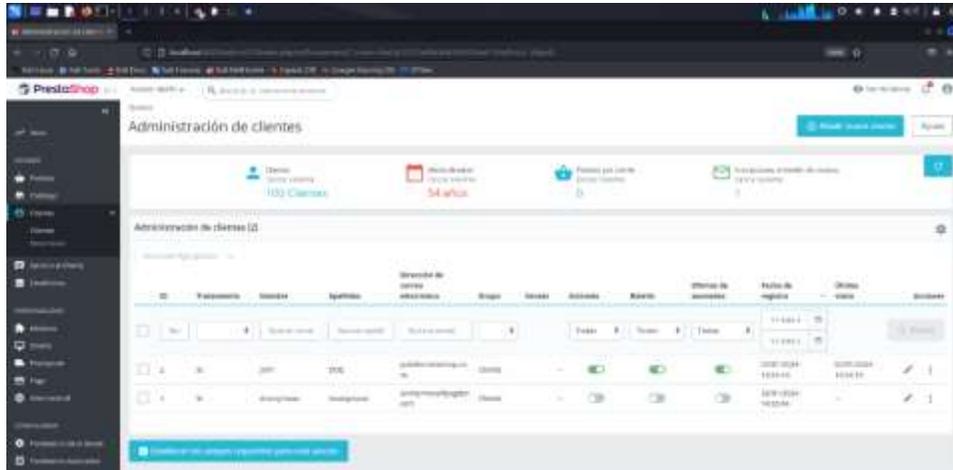


Ilustración 307: Listado clientes

Para nuestro nuevo cliente solo debemos agregar la información correspondiente que se nos muestra en pantalla, guardamos los datos y actualizamos la lista de los clientes

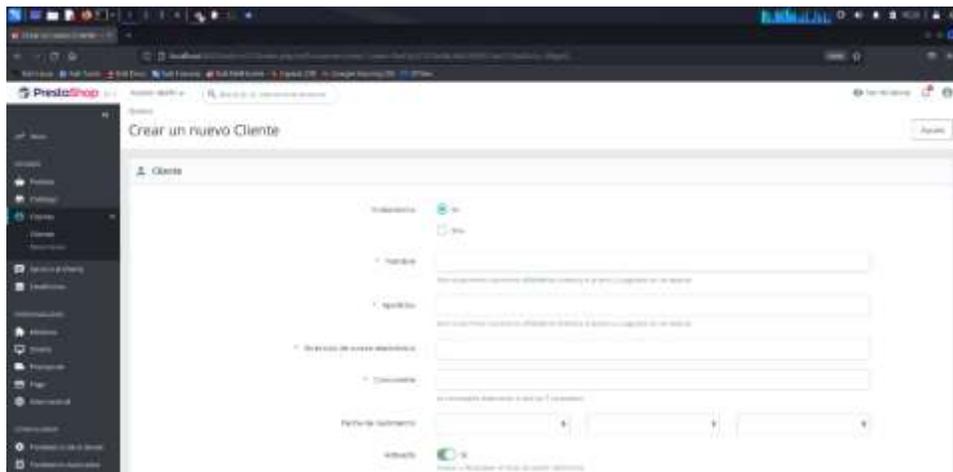


Ilustración 308: Agregar cliente

3. Realizar pedidos

Como siempre situarnos en el menú desplegable de la izquierda y seleccionar pedidos y primera opción, pedidos

En esta sección estarán todos los pedidos que hagan nuestros clientes antes mostrados en la sección anterior



En esta información se nos muestra las referencias, si es de un cliente nuevo, nombre del cliente, lugar de la entrega, un identificador (ID), total a pagar y fecha de realizado

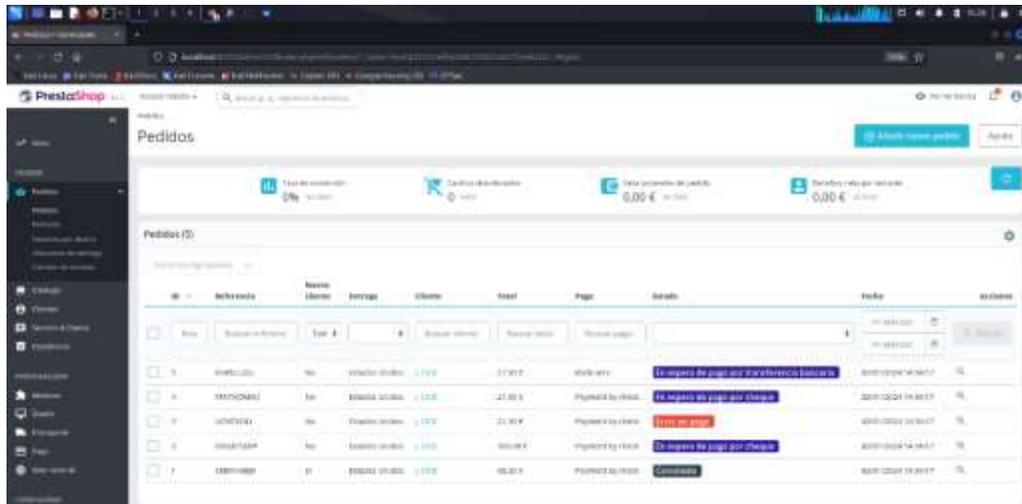


Ilustración 309: Información de pedidos

Una sección sumamente importante a tener en cuenta es la sección de estado, que nos muestra en qué estado se encuentra dicho pedido

Para realizar un pedido nuevo, primero se nos pide a nombre de que cliente deseamos realizar el pedido de ser existente, de lo contrario se puede agregar uno ahí mismo para realizar el pedido

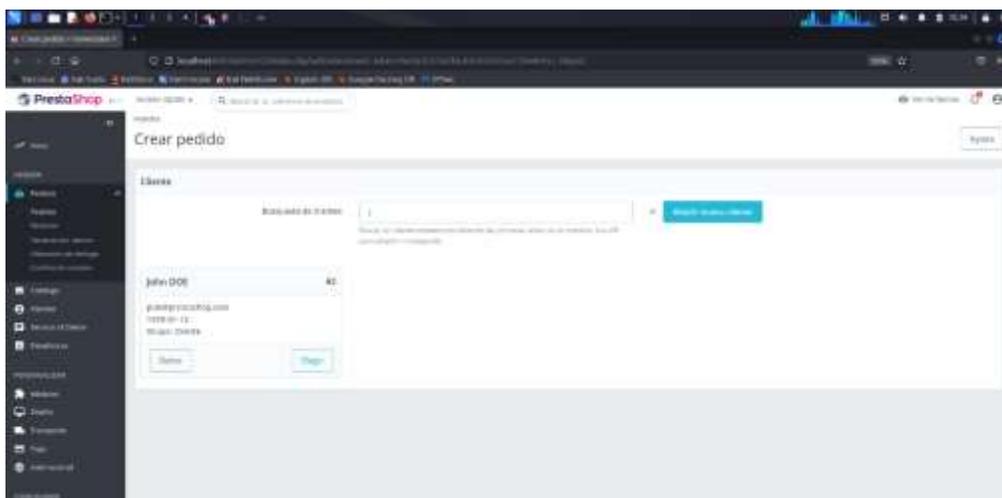


Ilustración 310: Realizar pedido



De haber seleccionado uno existente se nos mostrara el carrito del cliente para poder agregarle algún otro producto a este cliente, solo es cuestión de rellenar la información requerida

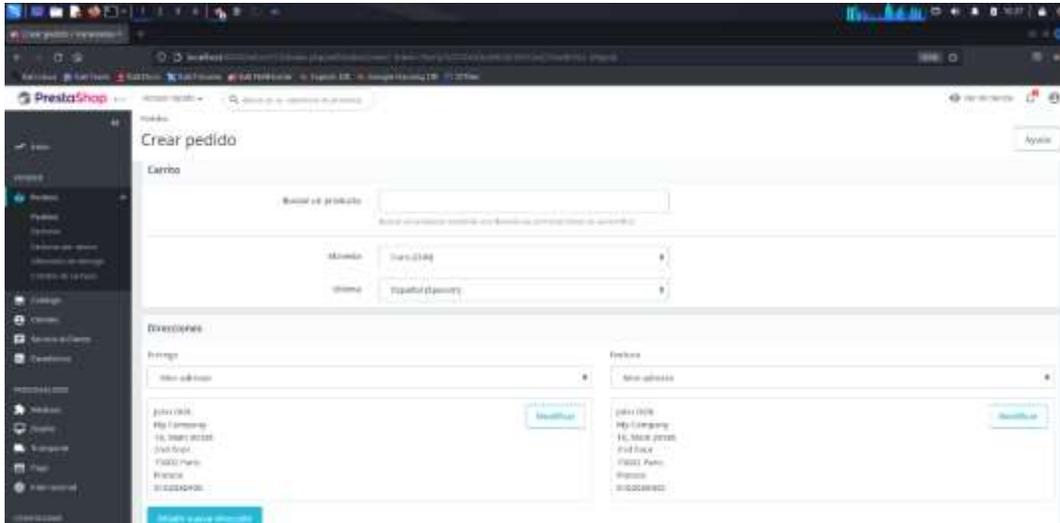


Ilustración 311: Carrito de cliente

O bien también podemos revisar los pedidos que haya realizado anteriormente

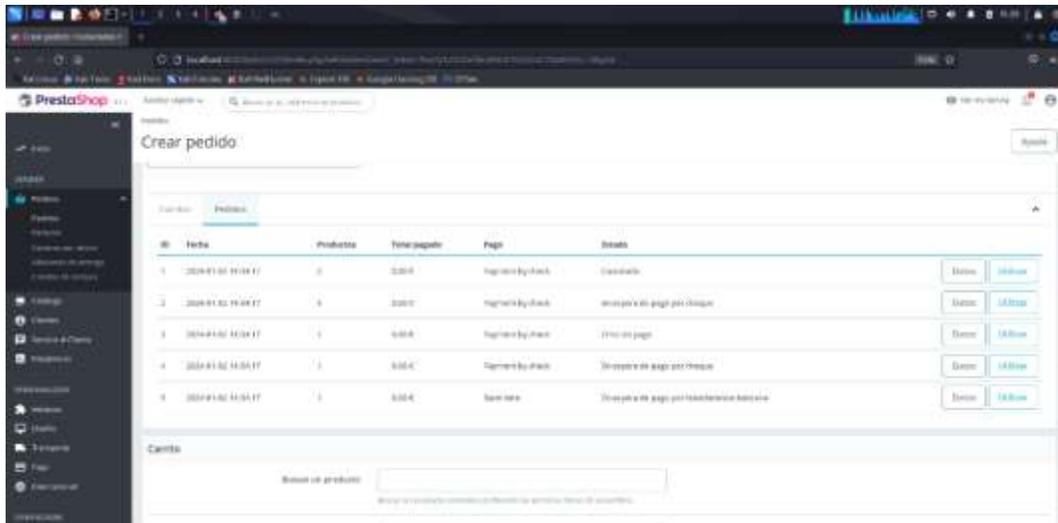


Ilustración 312: Pedidos realizados

Por último, realizaremos un pedido desde la cuenta de una cliente creada previamente

Para ello debemos entrar a la tienda virtual e iniciar sesión con este cliente

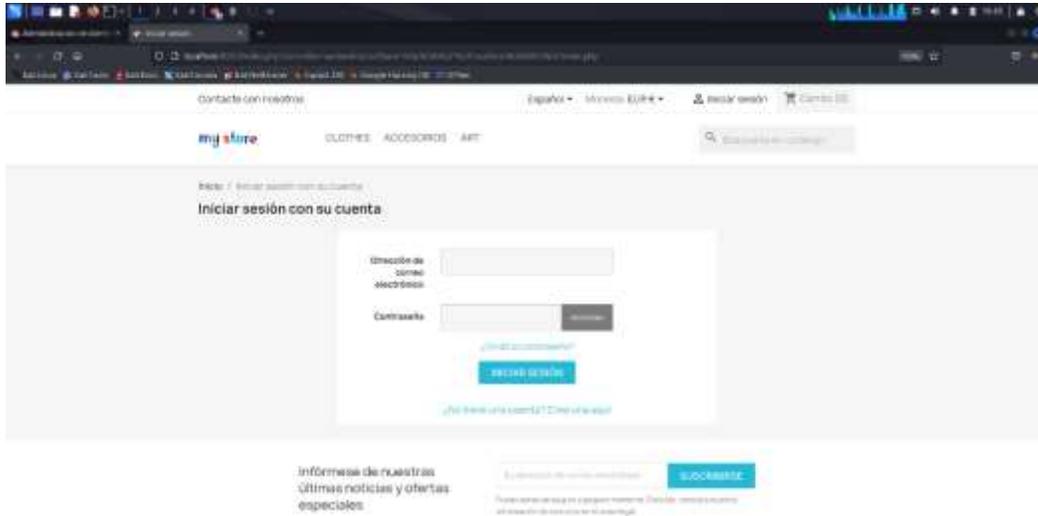


Ilustración 313: Inicio de sesión como cliente

Una vez dentro solo debemos agregar productos a nuestro carrito de compras

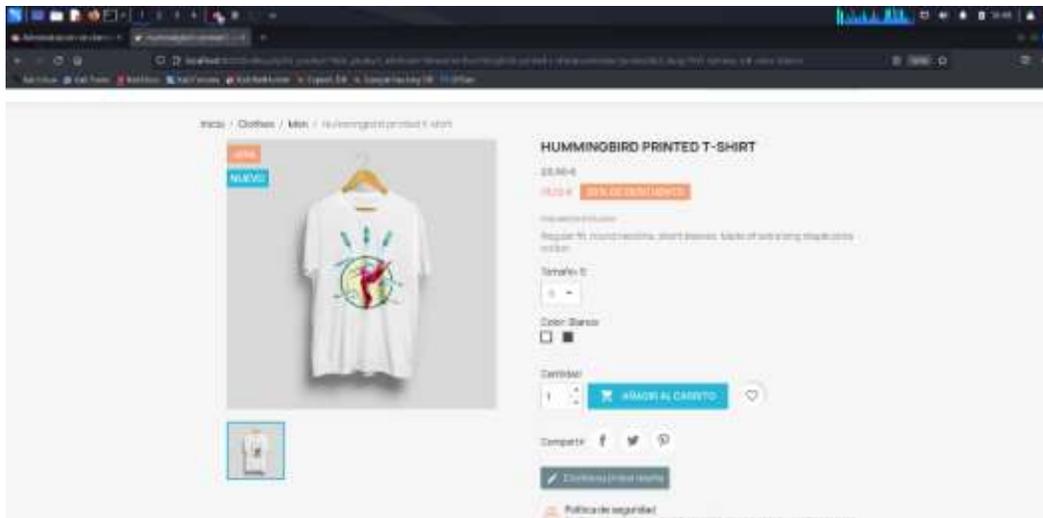


Ilustración 314: Agregar productos



Luego vamos a darle finalizar a nuestra compra

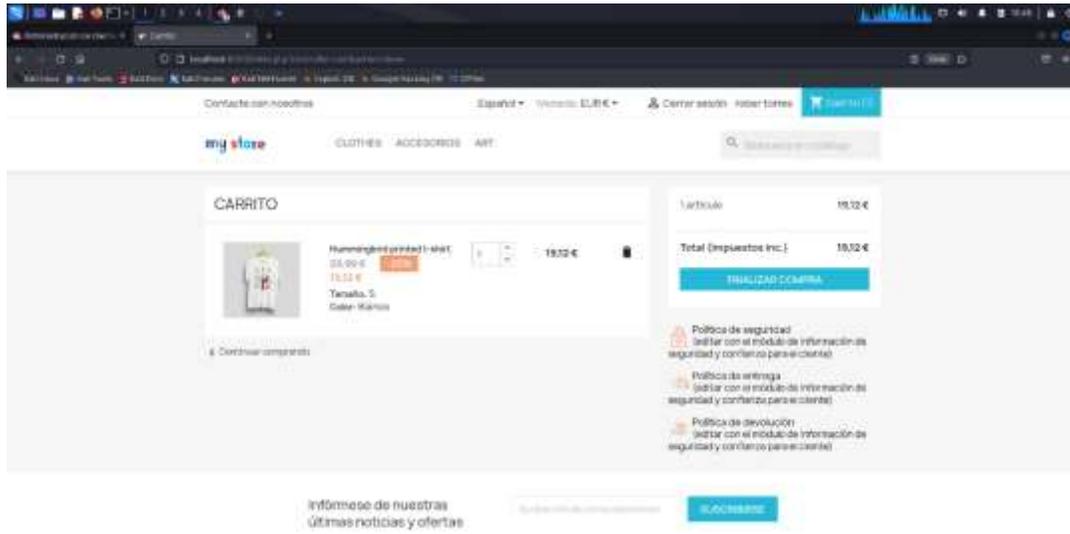


Ilustración 315: Finalizar compra

Agregaremos datos personales correspondientes

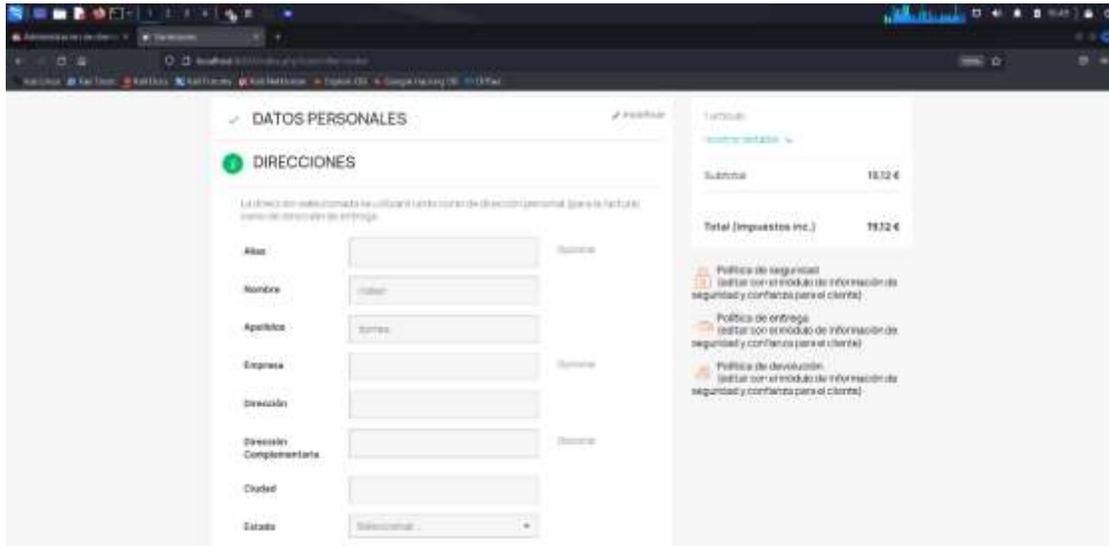


Ilustración 316: Datos personales



Seleccionamos transporte

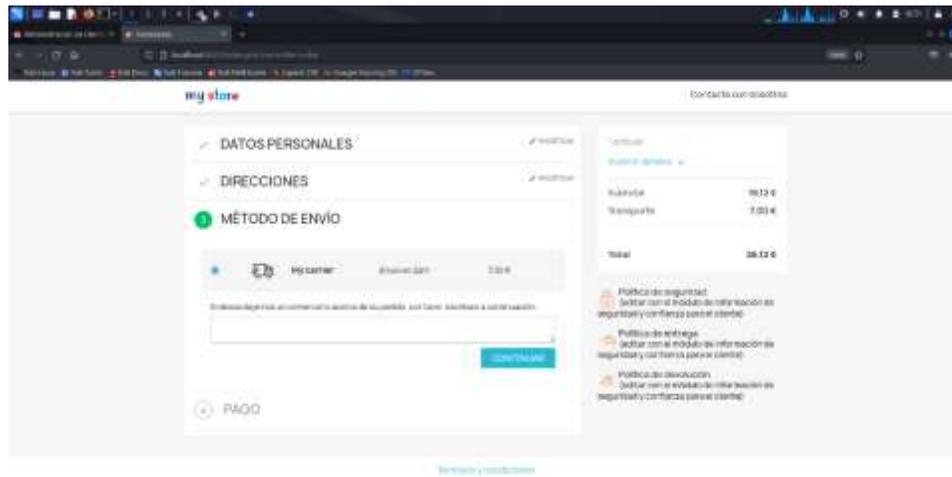


Ilustración 317: Selección de logística

Por último, realizamos el pago

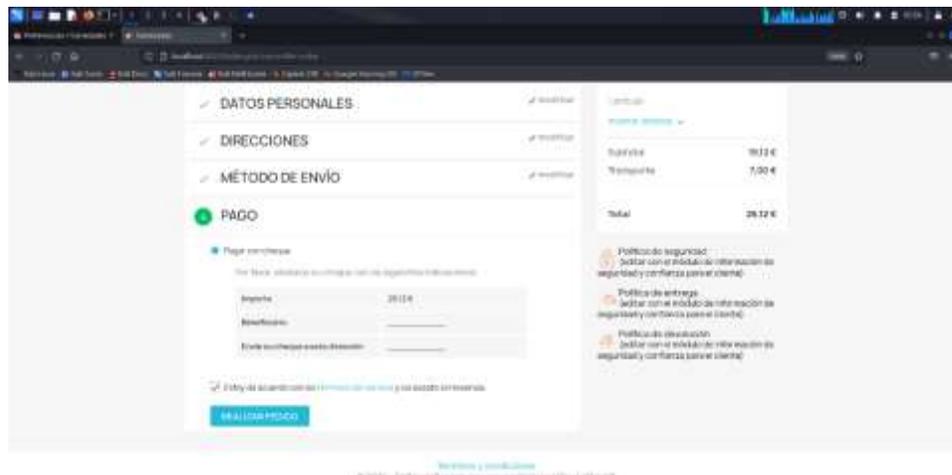


Ilustración 318: Realizar pago

NOTA: Existe la probabilidad de que las opciones de pago no se encuentren disponibles, solo es cuestión de dirigirnos a la sección de "pago" y luego en preferencias, una vez aquí dentro solo debemos buscar la sección de "Restricción por país" esto porque por defecto no se encuentra seleccionado ningún método de pago para ningún país, esto se solucionado seleccionando el método de pago que deseamos y luego marcar las casillas de los países que queremos habilitar

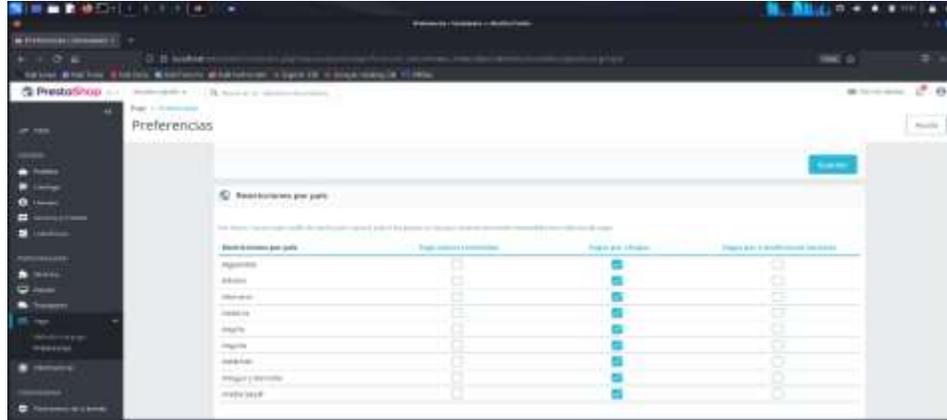


Ilustración 319: Habilitar pago

Y el pedido del cliente estaría confirmado

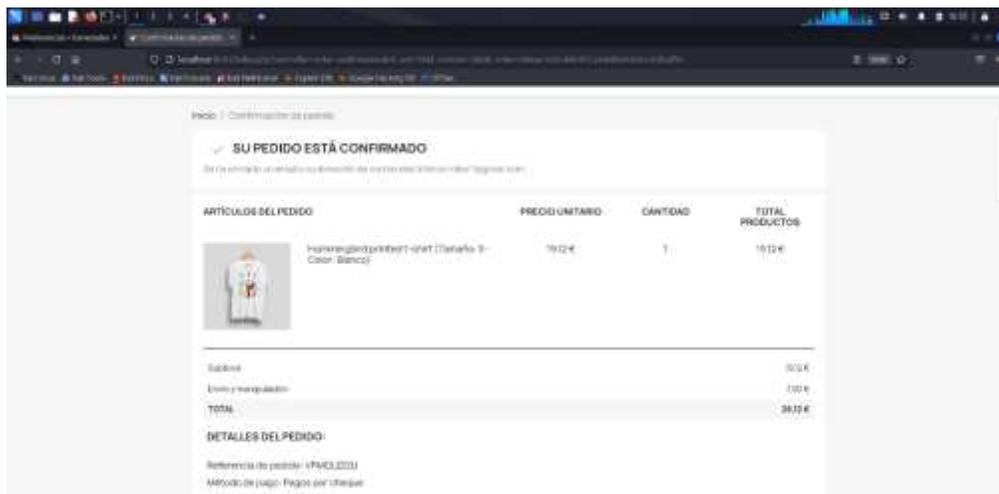


Ilustración 320: Pedido confirmado

Moodle

Instalación

La facilidad que nos proporciona esta imagen de Bitnami de Moodle, es que ya viene instalado, solo se levanta el contenedor y ya podemos ingresar en el puerto 80, y ya nos abre la página con la instalación predeterminada.

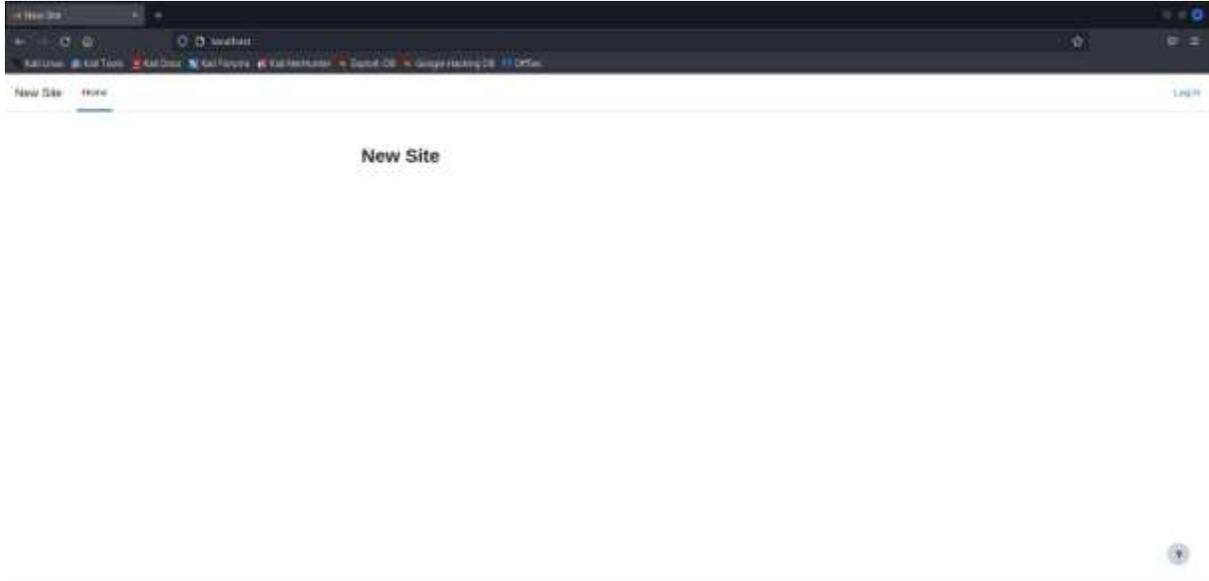


Ilustración 321: Index

Nos vamos a iniciar sesión, y como mencionamos anteriormente las credenciales de administrador predeterminada son:

Usuario: **user** Contraseña: **bitnami**

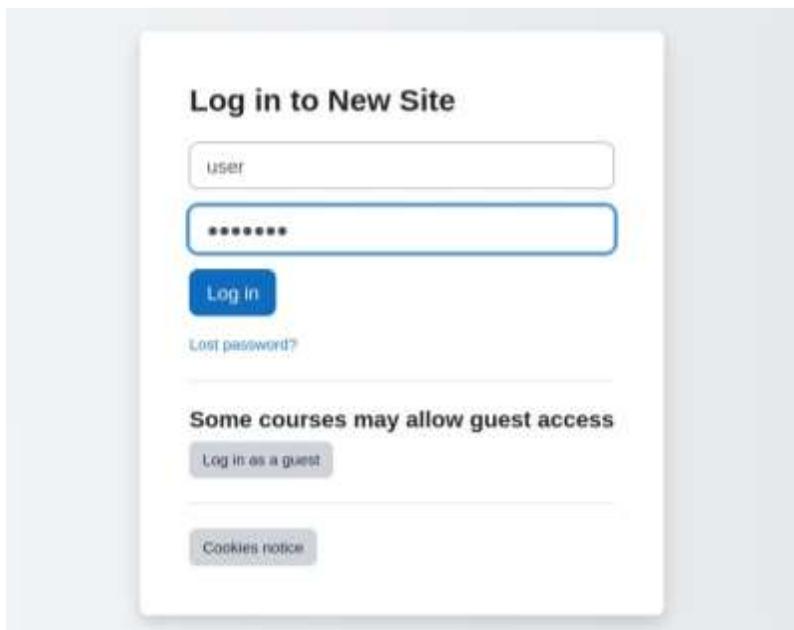


Ilustración 322: Login



Una vez hemos ingresado la sesión, ya estaremos logeados como admin



Ilustración 323: Página de inicio como admin

Para cambiar el lenguaje, nos dirigimos a la sección de **administración del sitio** y en la parte de Idioma seleccionamos ajustes de idioma, ahí buscamos el idioma que queremos y los instalamos.



Ilustración 324: Cambiar idioma

Para configurar las credenciales de administrador, nos dirigimos al perfil y en la sección general, podemos cambiar las credenciales

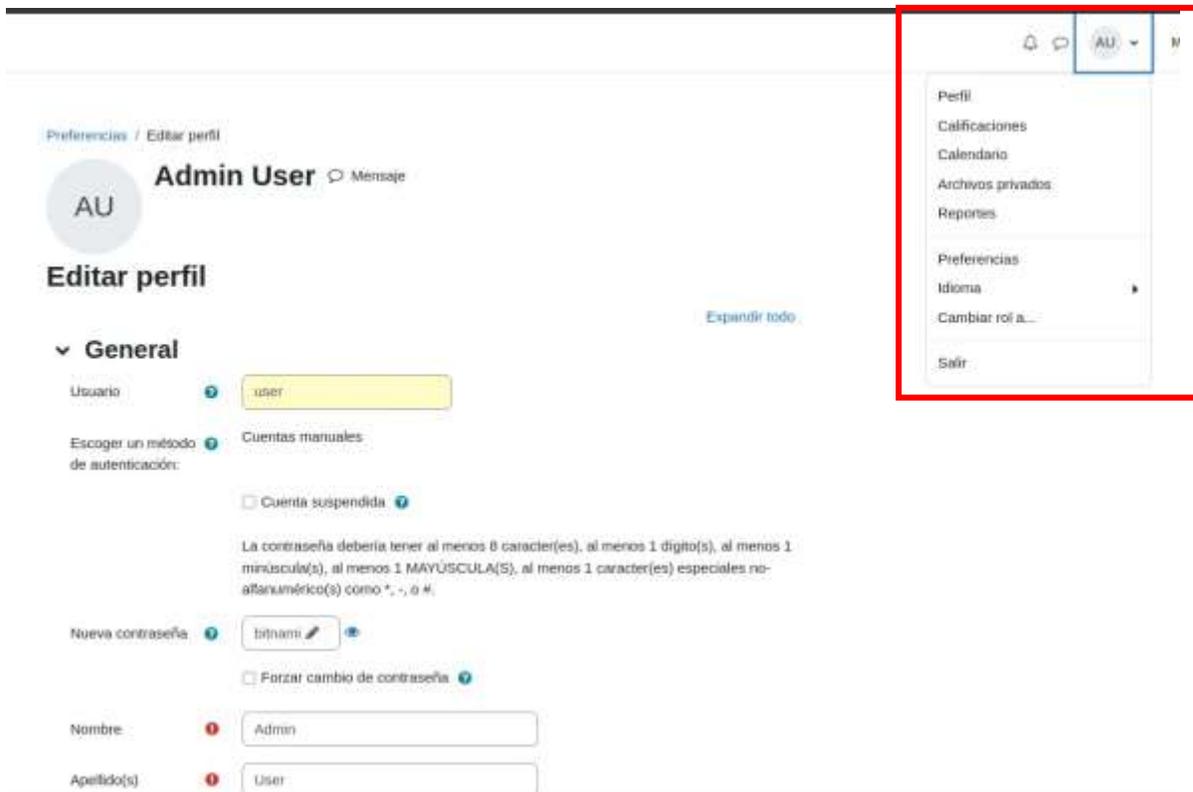


Ilustración 325: Cambiar credenciales

Así mismo, en la sección marcada en rojo, se puede cambiar el idioma.

Backups y restauración

Se mostrará el proceso para realizar un backup de la base de datos de Moodle

Lo primero será identificar el id contenedor de mysql que Moodle está utilizando, esto con el comando **docker ps -a**

```
(root@kali)~# docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS
5bc924973cc7  bitnami/moodle:4                   "/opt/bitnami/script..." 9 days ago    Up 2 seconds
30e81901ce00  bitnami/mariadb:10.6               "/opt/bitnami/script..." 9 days ago    Up 1 second
```

Ilustración 326: Id contenedor



Una vez identificado el Id, procederemos a ingresar al contenedor con el siguiente comando:

El comando utilizado es: **docker exec -u root -ti *id_contenedor* /bin/bash**

El parámetro -t se utiliza para un pseudo-tty, el parámetro -i se usa de manera interactiva. También se puede especificar un usuario, este debe existir en el contenedor, y se utiliza el parámetro -u, su estructura sería: `docker exec -ti -u usuario id_contenedor /bin/bash`

En este caso, como es una imagen ya predeterminada, utilizamos el usuario root, para obtener los privilegios de administrador.

De igual manera, como en el archivo yml, en la sección de base de datos, hemos definido el parámetro de contraseña vacía. Al ingresar a MariaDB, simplemente dejamos en blanco la contraseña.

```
(root@kali)-[~]
└─# docker exec -u root -ti 3de81 /bin/bash
root@3de81901ce00:/# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 13
Server version: 10.6.15-MariaDB Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| bitnami_moodle |
| information_schema |
| mysql |
| performance_schema |
| sys |
| test |
+-----+
6 rows in set (0.006 sec)

MariaDB [(none)]> █
```

Ilustración 327: Ingresar a contenedor mariadb



Una vez dentro del contenedor, podemos observar que hemos ingresado como root, de forma inmediata ingresamos a mysql, y verificamos rápidamente que exista la base de datos de Moodle.

Una vez verificado la base de datos, saldremos de la terminal de mysql y procederemos a crear el backup.

El comando a utilizar: `mysqldump -u root -p nombre_bd > nombre_backup.sql`

```
root@3de81901ce00:/# ls
bin bitnami boot dev docker-entrypoint-initdb.d etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
root@3de81901ce00:/# mysqldump -u root -p bitnami moodle > Backup_moodle.sql
Enter password:
root@3de81901ce00:/# ls
Backup_moodle.sql bin bitnami boot dev docker-entrypoint-initdb.d etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
root@3de81901ce00:/#
```

Ilustración 328: Ejecutar backup

De primera instancia observamos como ejecutamos un ls, esto para verificar que no hay un archivo.sql creado. Al momento de ejecutar el comando, nos pide contraseña de root, y acá la dejamos vacía.

Una vez ha terminado de crear el backup, hemos ejecutado el comando ls para verificar que se ha creado nuestros backup.

Para probar el punto de restauración, añadiremos un foro en Moodle, para luego restaurar la base de datos y ver como se restaura al punto de donde creamos el backup



Ilustración 329: Añadir foro de prueba



Hemos añadido este foro de prueba.

Nos devolveremos a la terminal del contenedor y ejecutaremos el siguiente comando:

```
mysql -u root -p -D bitnami_moodle < Backup_moodle.sql
```

```
root@3de81901ce00:/# mysql -u root -p -D bitnami_moodle < Backup_moodle.sql
Enter password:
```

Ilustración 330: Restaurar base de datos

Una vez, terminado el proceso de restauración, recargaremos la página de Moodle, y veremos que se restaurara hasta el punto donde se creó el backup.

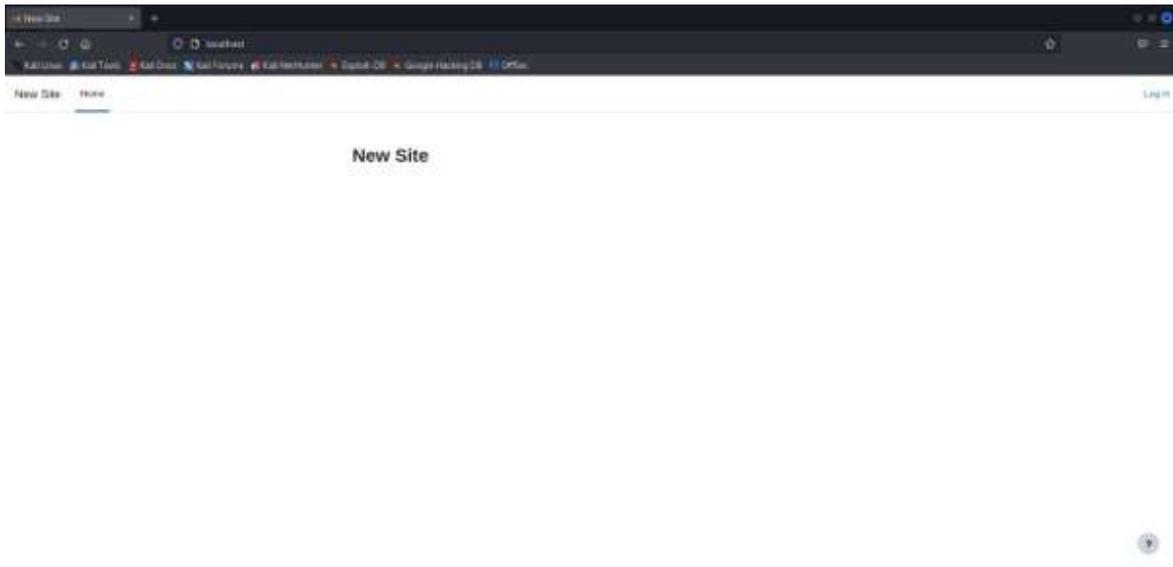


Ilustración 331: Index

Enunciados

1. Cambio de tema
2. Instalación plugin
3. Imagen alusiva al curso
4. Creación de la sección de recurso
5. Creación de la sección de actividades
6. Creación del cuestionario



Debería quedar así. Una vez damos instalar plugin, Moodle validara si este plugin se puede instalar, si todo sale bien, deberá salir este mensaje

Instalar plugin desde archivo ZIP

Validando theme_boost_o365teams ... OK

Validación exitosa, la instalación puede continuar

[Continuar](#) [Cancelar](#)

Ilustración 334: Validación

Para continuar nos mostrará un listado de todos los paquetes que se instalarán y su estado

Moodle 4.1 (Build: 20221128)

Si desea información sobre esta versión de Moodle, por favor vea [Release Notes](#) (o su traducción al Español)

Comprobaciones del servidor

Nombre	Información	Reporte	Plugin	Estatus
moodle		versión 3.9 es obligatoria y está ejecutando 4.1 (Build: 20221128) ↗		
unicode		debe estar instalado y activado ↗		
database	mysqli (10.6.15-MariaDB)	versión 10.4 es obligatoria y está ejecutando 10.6.15 ↗		
php		versión 7.4.0 es obligatoria y está ejecutando 8.0.28 ↗		
pcreunicode		debería estar instalado y activado para conseguir los mejores resultados ↗		
php_extension	iconv	debe estar instalado y activado ↗		
php_extension	mbstring	debe estar instalado y activado ↗		
php_extension	curl	debe estar instalado y activado ↗		
php_extension	openssl	debe estar instalado y activado ↗		
php_extension	tokenizer	debería estar instalado y activado para conseguir los mejores resultados ↗		
php_extension	soap	debería estar instalado y activado para conseguir los mejores resultados ↗		
php_extension	ctype	debe estar instalado y activado ↗		
php_extension	zip	debe estar instalado y activado ↗		
php_extension	zlib	debe estar instalado y activado ↗		
php_extension	gd	debe estar instalado y activado ↗		
php_extension	simplexml	debe estar instalado y activado ↗		
php_extension	spl	debe estar instalado y activado ↗		
php_extension	pcre	debe estar instalado y activado ↗		

Ilustración 335: Listado de paquetes



Para finalizar la instalación y si todo está correcto actualizaremos la base de datos

Comprobación de plugins

Esta página muestra los plugins que pueden requerir su atención durante la actualización, como por ejemplo los plugins nuevos a instalarse, plugins para actualizar, plugins fallidos, etc. Los plugins adicionales son mostrados si hubiera una actualización disponible para ellos. Se recomienda que Usted revise si hay versiones más reciente disponibles de plugins, y que actualice sus códigos fuentes antes de continuar con esta actualización de Moodle.

Revisar actualizaciones disponibles

La última revisión de Moodle es 4.15 de diciembre de 2021, 22:50

Plugins que requieren su atención

Cancelar las nuevas instalaciones (1)

Plugins que requieren su atención (1)

Todos los plugins (48)

Nombre del plugin / Directorio	Versión actual	Nueva versión	Requiere	Origen / Estatus
Temas				
Klass themebuyer		2022051701	<ul style="list-style-type: none">Moodle 2022041900theme_boost (2022041900)	Actualizar Para instalar Cancelar esta instalación

Recargar

Actualizar base de datos Moodle ahora

Ilustración 336: Comprobación de plugin

Para cambiar el tema, basta con ir a la sección de Apariencia -> Temas y Selector de temas

Administración del sitio

General Usuarios Cursos Calificaciones Plugins Apariencia Servidor Reportes Desarrollo

Apariencia

- Logos
- Colores de tarjetas del curso
- Calendario
- Blog
- Navegación
- Ajustes HTML
- Moodle Docs
- Página predeterminada del Tablero
- Página predeterminada de perfil
- Cursos
- AJAX y Javascript
- Administrar marcas
- HTML adicional
- Pantallas
- Tours para usuarios

Temas

- Ajustes de temas
- Selector de temas
- Academi
- Boost (Impulso)
- Classic (Clásico)
- Klass

Ilustración 337: Cambiar tema



Seleccionaremos el tema que hemos instalado, lo aplicamos y nos deberá cambiar el tema

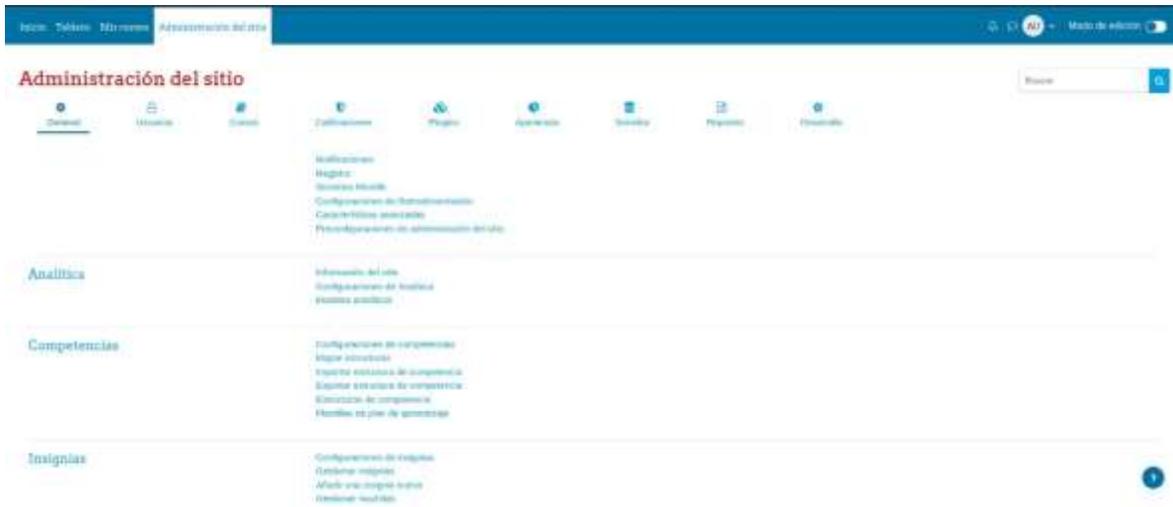


Ilustración 338: Aplicar cambios

2. Instalación plugin

Así mismo como en el paso anterior, se realiza el mismo procedimiento de instalación de algún plugin de nuestro interés o necesidad.

En este caso, se ha instalado este plugin



Ilustración 339: Plugin File

Que este plugin nos agrega una sección más al momento de subir algún documento como entrega de tarea



3. Imagen alusiva al curso

Debido a que escogimos un tema donde nos permite tener un carrusel de imágenes, hemos agregado la galería de carrusel y un logo

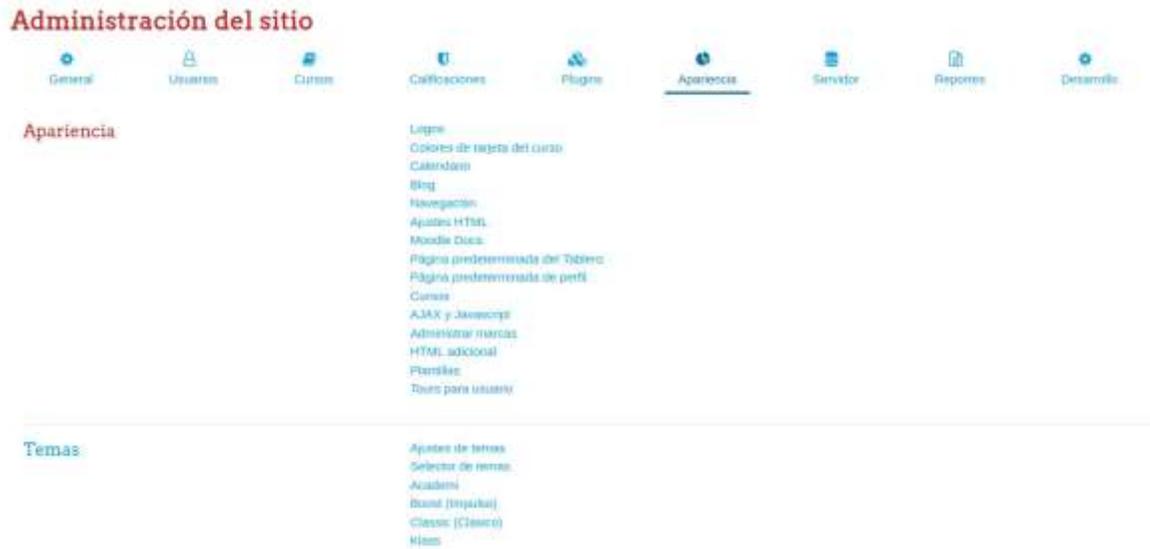


Ilustración 341: Carrusel y logo

En el apartado de **Klass** (nombre de nuestro tema), podremos configurar el logo y el carrusel.



Ilustración 342: Imagen logo

En las opciones del carrusel, este nos permite escoger cuantas imágenes queremos tener. Y se deben agregar una por una en cada sección enumerada.



General Portada Carrusel de imágenes Pie de página

Carrusel para la Portada

Esto crea un carrusel de hasta 12 imágenes, para que Usted promueva elementos importantes en su sitio. La presentación es responsiva, en donde la altura de la imagen es `conf.imagen.default_slide` de la carpeta `pic`.

Mostrar carrusel de imágenes: Si Valor predeterminado: Si
theme_class | toggleImage
Elija si desea ocultar o mostrar el carrusel de imágenes

Número de imágenes: Valor predeterminado: 1
theme_class | numberofslides
Número de imágenes en el carrusel

Imagen 1

Escriba las configuraciones para la imagen 1.

Imagen
theme_class | slideImage

Archivos

Ilustración 343: Imágenes carrusel

Este es el resultado, con el logo y las fotos del carrusel



Ilustración 344: Verificación



4. Creación de la sección de recurso

Se ha creado el siguiente curso



Ilustración 345: Creación de curso

Una vez dentro del curso, activamos el modo edición (botón ubicado en la parte superior derecha)



Ilustración 346: Añadir recurso



Añadimos la actividad, nos saldrá un recuadro con múltiples opciones

Añadir una actividad o recurso ✕

Buscar

Todos **Actividades** Recursos

 Archivo ☆ ⓘ	 Área de texto y medios ☆ ⓘ	 Base de datos ☆ ⓘ	 Carpeta (folder) ☆ ⓘ	 Chat ☆ ⓘ	 Elección ☆ ⓘ
 Encuesta predefinida ☆ ⓘ	 Examen ☆ ⓘ	 Foro ☆ ⓘ	 Glosario ☆ ⓘ	 H5P ☆ ⓘ	 Herramienta externa ☆ ⓘ
 Lección ☆ ⓘ	 Libro ☆ ⓘ	 Página ☆ ⓘ	 Paquete contenido IMS ☆ ⓘ	 Paquete SCORM ☆ ⓘ	 Retroalimentación ☆ ⓘ
 Taller ☆ ⓘ	 Tarea ☆ ⓘ	 URL ☆ ⓘ	 Wiki ☆ ⓘ		

Ilustración 347: Actividades y recursos



En mi caso, en la sección de Recursos, agregue un foro y un archivo ZIP



Ilustración 348: Foro y recurso ZIP

5. Creación de la sección de actividades

De igual forma, en el modo edición, agregamos la actividad **Tarea**

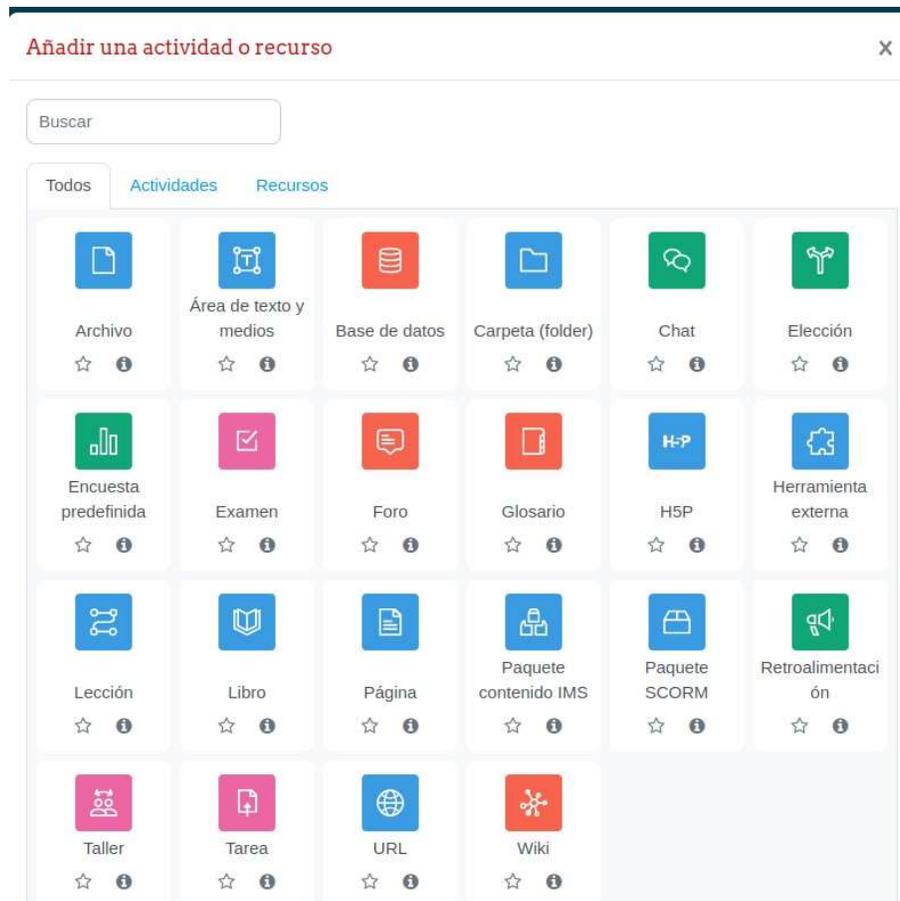


Ilustración 349: Actividad tarea



Rellenamos los campos necesarios y establecemos una fecha inicio y límite para la entrega del mismo

Administración de redes

Actualizando Tarea en ACTIVIDADES

General

Nombre de la tarea: Administración de redes

Descripción:

Ilustración 350: Definir parámetros

Y así se vería la entrega

MARCAR COMO HECHO

Abiertos: domingo, 17 de diciembre de 2023, 00:00
Pendientes: domingo, 24 de diciembre de 2023, 00:00

VER TODOS LOS ENVÍOS

Calificación

Sumario de calificaciones

Oculto para los estudiantes	No
Participantes	0
Enviados	0
Necesita calificarse	0
Tiempo restante	6 días 21 horas

AÑADIR ENVÍO

Ilustración 351: Visualización



6. Creación del cuestionario

Agregamos la actividad de Examen

Añadir una actividad o recurso X

Buscar

Todos **Actividades** Recursos

 Archivo ☆ ⓘ	 Área de texto y medios ☆ ⓘ	 Base de datos ☆ ⓘ	 Carpeta (folder) ☆ ⓘ	 Chat ☆ ⓘ	 Elección ☆ ⓘ
 Encuesta predefinida ☆ ⓘ	 Examen ☆ ⓘ	 Foro ☆ ⓘ	 Glosario ☆ ⓘ	 H5P ☆ ⓘ	 Herramienta externa ☆ ⓘ
 Lección ☆ ⓘ	 Libro ☆ ⓘ	 Página ☆ ⓘ	 Paquete contenido IMS ☆ ⓘ	 Paquete SCORM ☆ ⓘ	 Retroalimentación ☆ ⓘ
 Taller ☆ ⓘ	 Tarea ☆ ⓘ	 URL ☆ ⓘ	 Wiki ☆ ⓘ		

Ilustración 352: Actividad cuestionario



Rellenamos los campos necesarios, podemos especificar cuantos intentos, la fecha de inicio y fecha de finalización y el tiempo que durará, una vez finalizado, nos iremos a la sección de banco de preguntas

Ing Sistema | Examen de redes

EXAMEN **Examen de redes**

Examen Configuración Preguntas Resultados

Banco de preguntas Más

MARCAR COMO HECHO

Abrió: domingo, 17 de diciembre de 2023, 02:55
Cierra: lunes, 18 de diciembre de 2023, 02:55

VISTA PREVIA DEL EXAMEN

Intentos permitidos: 2
Límite de tiempo: 1 hora
Método de calificación: Calificación más alta

Ilustración 353: Banco de preguntas

Y agregaremos tantas preguntas y los tipos, según sea necesario

Preguntas: 3

Banco de preguntas

Seleccionar una categoría: Por defecto en Ing Sistem

Categoría por defecto para preguntas compartidas en el contexto Ing Sistema

Se filtra por mayor a menor

Filtrar por estado...

¿Mostrar el texto de la pregunta en la lista de preguntas?

Opciones de búsqueda

Mostrar también preguntas de las subcategorías

Mostrar también preguntas antiguas

CREAR UNA NUEVA PREGUNTA

Pregunta	Acciones	Estado	Versión	Creado por	Comentarios	¿Necesita revisión?	Índice de dificultad	Eficacia discriminativa	Tipo	Último usado	Modificado por
<input type="checkbox"/> ¿Cómo se llama?	Editar	Lista	v1	Admin User 17 de diciembre de 2023, 03:04	0	-	NO	NO	0	Nunca	Admin User 17 de diciembre de 2023, 03:04
<input type="checkbox"/> Símb	Editar	Lista	v1	Admin User 17 de diciembre de 2023, 03:04	0	-	NO	NO	1	Nunca	Admin User 17 de diciembre de 2023, 03:04
<input type="checkbox"/> Cuáles son los nombres de las unidades binarias?	Editar	Lista	v1	Admin User 17 de diciembre de 2023, 03:04	0	-	NO	NO	0	Nunca	Admin User 17 de diciembre de 2023, 03:04

Ilustración 354: Crear preguntas



7. Matriculación con clave

Para matriculación con clave, nos iremos a las opciones del curso que necesitemos agregarle la clave, y en la sección de participantes, desplegamos la lista de métodos de inscripción



Ilustración 355: Agregar clave de curso

Rellenamos los campos de cómo se verá la etiqueta y agregamos la clave.

Metodos de inscripción

Nombre	Usuarios	Arriba/Abajo	Editar
Inscripciones manuales	1	↓	👁️⬆️⬇️
Acceso de invitados	0	↑↓	👁️⬆️⬇️
Auto-inscripción (Estudiante)	0	↑↓	👁️⬆️⬇️
Invitado	0	↑	👁️⬆️⬇️

Ilustración 356: Listado de métodos de inscripción

En este caso, se le asignó **Auto-inscripción (estudiante)**, además, en la columna **Editar** verificamos que el icono del ojo esté activo, para que se habilite este método de matriculación.



Para verificar que funciona, se creara un usuario normal

Mostrar más...

AÑADIR FILTRO

Nombre / Apellido(s)	Dirección Email
Admin User	user@example.com
Jason Flores	jason@gmail.com

AÑADIR UN NUEVO USUARIO

Ilustración 357: Usuario estudiante

E ingresaremos con este usuario a Moodle



Jason Flores [Mensaje](#)

Detalles de usuario [Editar perfil](#)

Dirección Email
jason@gmail.com (Visible para otros participantes del curso)

País
Nicaragua

Zona horaria
Europe/London

Ilustración 358: Ingresar como estudiante



Si nos vamos a la página inicial de Moodle, veremos cómo nos aparece el curso con contraseña

Ingeniería en Sistemas e Informatica

Opciones de inscripción

Ingeniería en Sistemas e Informatica 🔍



A lo largo de este curso se le mostrara todos los componentes que conlleva esta carrera

Profesor: Admin User

▼ **Estudiante**

Clave de inscripción (¿# de grupo?)

INSCRIBIRME

Ilustración 359: Curso restringido

Al ingresar al curso, este se vería de esta forma



Ilustración 360: Material de curso



8. Realizar tarea y cuestionario

Entrega de tarea

ing Sistema / Administración de redes

TAREA Administración de redes

✓ HECHO

Abiertos: domingo, 17 de diciembre de 2023, 00:00
Pendientes: domingo, 24 de diciembre de 2023, 00:00

[EDITAR ENVÍO](#) [QUITAR ENVÍO](#)

Estatus de la entrega

Estatus de la entrega	Enviado para calificar
Estatus de calificación	No calificado
Tiempo restante	La tarea fue enviada 6 días 20 horas antes
Última modificación	domingo, 17 de diciembre de 2023, 03:39
Envíos de archivo	 docker-compose7.yml 17 de diciembre de 2023, 03:39
Comentarios al envío	Comentarios (0)

Ilustración 361: Entrega de tarea

Cuestionario finalizado

ing Sistema / Examen de redes

EXAMEN Examen de redes

[MARCAR COMO HECHO](#)

Abrió: domingo, 17 de diciembre de 2023, 02:59
Cierre: lunes, 18 de diciembre de 2023, 02:59

[REINTENTAR EL EXAMEN](#)

Intentos permitidos: 2
Límite de tiempo: 1 hora
Método de calificación: Calificación más alta

Resumen de sus intentos previos

Intento	Estado	Calificación / 1.00	Calificación / 10.00	Revisión
1	Terminado Finalizó domingo, 17 de diciembre de 2023, 02:48	1.00	10.00	Revisión

Ilustración 362: Cuestionario finalizado



Programa "Menú de CMS"

Funciones principales del programa:

- 1) Levantar un CMS.
- 2) Opciones de detención de contenedores.
- 3) Crear respaldo de un CMS.
- 4) Eliminar datos de un contenedor (volumen).
- 5) Salir.

```
(root@kali) - [~/home/user/Downloads/docker] # ./script.txt
Seleccione una opcion:
1) Levantar un CMS
2) Opciones de detencion para los contenedores
3) Crear respaldo de un CMS
4) Eliminar datos de un contenedor (Volumen)
5) Salir
Opcion: █
```

Ilustración 363: Funciones principales del programa

1. Levantar un CMS: nos despliega un menú de todos los cms disponibles que podemos levantar

- Wordpress
- Joomla
- Drupal



- Prestashop

- Moodle

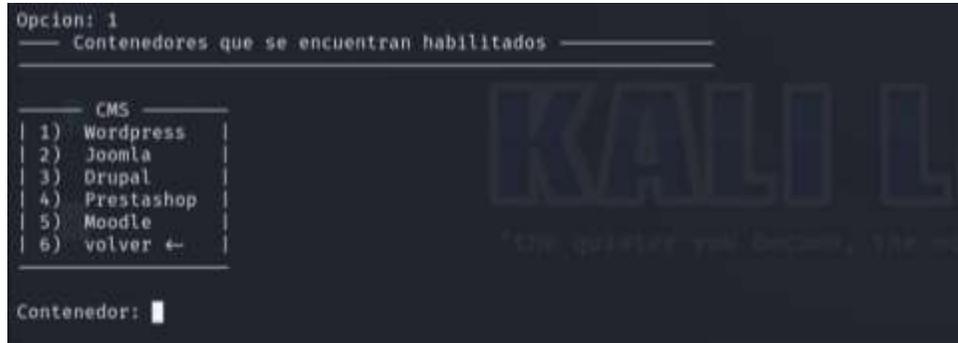


Ilustración 364: Opción de "Levantar un CMS"

Solo es cuestión de introducir el número de opción del CMS que queremos levantar, una vez escogido el programa levantara los 2 contenedores que conforman al CMS en cuestión que son:

- Contenedor web (representado con los nombres de contenedores que tienen la terminación de `_web`).
- Contenedor de base de datos (representado con los nombres de contenedores que tienen la terminación `_db`).

Existe un tercer contenedor que poseen algunos CMS, este es el de phpmyadmin y se refleja en los nombres con contenedores que tenga su nombre con terminación `_pma`, su función es manejar la administración de MySQL (en otras palabras, el contenedor de base de datos) a través de páginas web utilizando un navegador web.

Una vez levantado el cms el programa nos mostrara su información de conexión o los puertos en los que están alojados sus contenedores



```
Opcion: 1
---- Contenedores que se encuentran habilitados ----
-----
      CMS
-----
| 1) Wordpress |
| 2) Joomla    |
| 3) Drupal    |
| 4) Prestashop|
| 5) Moodle    |
| 6) volver ←  |
-----

Contenedor: 1
Creating network "wordpress_default" with the default driver
Creating wordpress_db_1 ... done
Creating wordpress_web_1 ... done
Creating wordpress_pma_1 ... done

-- Informacion de conexion del CMS --

Wordpress = http://localhost:8000      phpmyadmin = http://localhost:81

Desea salir?(s/n): █
```

Ilustración 365: Levantando un CMS

1.1 Sección de información general de los contenedores:

" ---- Contenedores que se encuentran habilitados ---- "

Información que podemos ver en esta sección:

- Nombre del CMS
- Puerto de conexión
- Estado
- Imagen base



```
root@kali: /home/user/Downloads/docker
File Actions Edit View Help
root@kali: /home/user/Downloads/docker
./SCRIPT.LST
Seleccione una opcion:
1) Levantar un CMS
2) Opciones de detencion para los contenedores
3) Crear respaldo de un CMS
4) Eliminar datos de un contenedor (Volumen)
5) Salir
Opcion: 1
----- Contenedores que se encuentran habilitados -----
----- CMS Activo -----
NOMBRE: wordpress
| Cont. Web ESTADO:Up 10 minutos PUERTO:8000 TCP IMAGEN:wordpress:latest
| Cont. BD ESTADO: PUERTO: TCP IMAGEN:mysql:5.7
----- CMS Detenido -----
NOMBRE: joomla
| Cont. Web ESTADO:Exited (0) 3 weeks ago IMAGEN:joomla:4
| Cont. BD ESTADO:Exited (128) 3 weeks ago IMAGEN:phpmyadmin/phpmyadmin
----- CMS Activo -----
NOMBRE: prestashop
| Cont. Web ESTADO:Up 10 minutos PUERTO:8003 TCP IMAGEN:prestashop/prestashop:latest
| Cont. BD ESTADO: PUERTO: TCP IMAGEN:mysql:5.7
----- CMS Detenido -----
NOMBRE: moodle
| Cont. Web ESTADO:Exited (0) 3 weeks ago IMAGEN:bitnami/moodle:4
| Cont. BD ESTADO: IMAGEN:
-----
CMS
| 1) Wordpress |
| 2) Joomla |
| 3) Drupal |
| 4) Prestashop |
| 5) Moodle |
| 6) volver ← |
Contenedor: |
```

Ilustración 366: Información de los contenedores habilitados.

Esta sección está disponible en cada función del programa, para que los usuarios puedan ver todo el tiempo la información relevante de los contenedores de los CMS con los que estamos trabajando. En esta sección se verán reflejados contenedores que se encuentren en 2 estados posibles:

- Up: Es el estado de funcionamiento habitual, un CMS con correcto funcionamiento tendrá sus contenedores en este estado (En la sección se pueden ver en un "CMS Activo").

- Exited: Este es el estado que adquiere cuando el contenedor se encuentra detenido o en algunos casos cuando el contenedor se detuvo por un agente externos, esto lo puede provocar si se apaga el computador aun estando en



funcionamiento o si se cierra abruptamente una máquina virtual (en caso de trabajarlo en una), En la sección se pueden ver en un "CMS inactivo".



2. Opciones de detención para los contenedores

Esta función a su vez se divide en 2:

```
Seleccione una opcion:
1) Levantar un CMS
2) Opciones de detencion para los contenedores
3) Crear respaldo de un CMS
4) Eliminar datos de un contenedor (Volumen)
5) Salir

Opcion: 2
___ Contenedores que se encuentran habilitados _____
___ CMS Activo ___

NOMBRE: wordpress
  ___ Cont. Web ESTADO:Up 11 minutes PUERTO:8000 TCP IMAGEN:wordpress:latest
  ___ Cont. BD ESTADO: PUERTO: TCP IMAGEN:mysql:5.7

___ CMS Detenido ___

NOMBRE: joomla
  ___ Cont. Web ESTADO:Exited (0) 3 weeks ago IMAGEN:joomla:4
  ___ Cont. BD ESTADO:Exited (128) 3 weeks ago IMAGEN:phpmyadmin/phpmyadmin

___ CMS Activo ___

NOMBRE: prestashop
  ___ Cont. Web ESTADO:Up 11 minutes PUERTO:8003 TCP IMAGEN:prestashop/prestashop:latest
  ___ Cont. BD ESTADO: PUERTO: TCP IMAGEN:mysql:5.7

___ CMS Detenido ___

NOMBRE: moodle
  ___ Cont. Web ESTADO:Exited (0) 3 weeks ago IMAGEN:bitnami/moodle:4
  ___ Cont. BD ESTADO: IMAGEN:

Seleccione una opcion:
1) Bajar los contenedores de un CMS
2) Detener los contenedores de un CMS
3) Volver

Opcion: █
```

Ilustración 367: Opciones sobre la detención de un contenedor

2.1 Bajar los contenedores de un CMS

Dar de baja a los contenedores equivale a detener todos sus funcionamientos y eliminarlo de la lista de los contenedores que se encuentran habilitados por lo que en la sección de información ya no se verá reflejado sin embargo no elimina sus datos almacenados (volumen) por lo que si el usuario decide levantarlos nuevamente podrán seguir trabajando de donde lo dejo.



- El parámetro que utiliza para funcionar es pasar el nombre del CMS que queremos dar de baja.

2.2 Detener los contenedores de un CMS

* Detener los contenedores que conforman un CMS, como su nombre lo indica solo detiene temporalmente su funcionamiento hasta que el usuario desee volver a levantar el CMS (ósea los contenedores que lo conforman), los contenedores detenidos si se verán reflejados en la sección de información por lo que aun podemos ver su información relevante (estado, puerto, nombre e imagen base).

```
Opcion: 1
--- Contenedores que se encuentran habilitados -----
--- CMS Detenido ---
NOMBRE: wordpress
| Cont. Web ESTADO:Exited (0) 59 minutes ago IMAGEN:wordpress:latest
| Cont. BD ESTADO:Exited (0) 59 minutes ago IMAGEN:phpmyadmin/phpmyadmin
--- CMS Activo ---
NOMBRE: Joomla
| Cont. Web ESTADO:Up 58 minutes PUERTO:8001 TCP IMAGEN:joomla:3.6.2-apache
| Cont. BD ESTADO:Up 58 minutes PUERTO:83 TCP IMAGEN:mysql:5.7
```

Ilustración 368: Información sobre los contenedores activos y detenidos

- El parámetro que utiliza para funcionar es pasar el nombre del CMS que queremos detener.



```
Opcion: 2
----- Contenedores que se encuentran habilitados -----
----- CMS Activo -----

NOMBRE: wordpress
| Cont. Web ESTADO:Up About a minute PUERTO:8080 TCP IMAGEN:wordpress:latest
| Cont. BB ESTADO:Up About a minute PUERTO:81 TCP IMAGEN:mysql:8

-----

Seleccione una opcion:
1) Bajar los contenedores de un CMS
2) Detener los contenedores de un CMS
3) Volver

Opcion: 2
Nombre del Contenedor: wordpress

----- Bajando Contenedores -----
NOMBRE ID ESTADO
wordpress 9ee ... detenido ...
wordpress_DB adc ... detenido ...

-----

----- El contenedor ha sido detenido (Exited) -----

Desea salir?(s/n):
```

Ilustración 369: Deteniendo un contenedor

3. Crear respaldo de un CMS

- El parámetro que utiliza esta función es el nombre del CMS al que queremos hacer un respaldo, esto con el fin de guardar una copia con lo que quisiéramos trabajar a futuro o si quisiéramos trabajar en otro ordenador.



```
Seleccione una opción:
1) Levantar un CMS
2) Opciones de detención para los contenedores
3) Crear respaldo de un CMS
4) Eliminar datos de un contenedor (Volumen)
5) Salir

Opción: 3
___ Contenedores que se encuentran habilitados _____
___ CMS Activo ___

NOMBRE: wordpress
|___ Cont. Web ESTADO:Up 18 minutes PUERTO:8000 TCP IMAGEN:wordpress:latest
|___ Cont. BD ESTADO: PUERTO: TCP IMAGEN:mysql:5.7

___ CMS Detenido ___

NOMBRE: joomla
|___ Cont. Web ESTADO:Exited (0) 3 weeks ago IMAGEN:joomla:4
|___ Cont. BD ESTADO:Exited (128) 3 weeks ago IMAGEN:phpmyadmin/phpmyadmin

___ CMS Activo ___

NOMBRE: prestashop
|___ Cont. Web ESTADO:Up 18 minutes PUERTO:8003 TCP IMAGEN:prestashop/prestashop:latest
|___ Cont. BD ESTADO: PUERTO: TCP IMAGEN:mysql:5.7

___ CMS Detenido ___

NOMBRE: moodle
|___ Cont. Web ESTADO:Exited (0) 3 weeks ago IMAGEN:bitnami/moodle:4
|___ Cont. BD ESTADO: IMAGEN:

Que CMS desea respaldar?: █
```

Ilustración 370: Opción de crear un respaldo

3.1 El programa lleva el siguiente orden:

Paso 1: Crear un respaldo de la base de datos situada en nuestro contenedor encargado del servicio de MySQL.

Paso 2: Mover dicho respaldo a una carpeta que es específicamente encargada de albergar todos los respaldos que se vayan creando de dicho CMS, cabe aclarar que cada CMS tiene su propia carpeta de respaldos con el fin de llevar un mejor orden en los archivos.

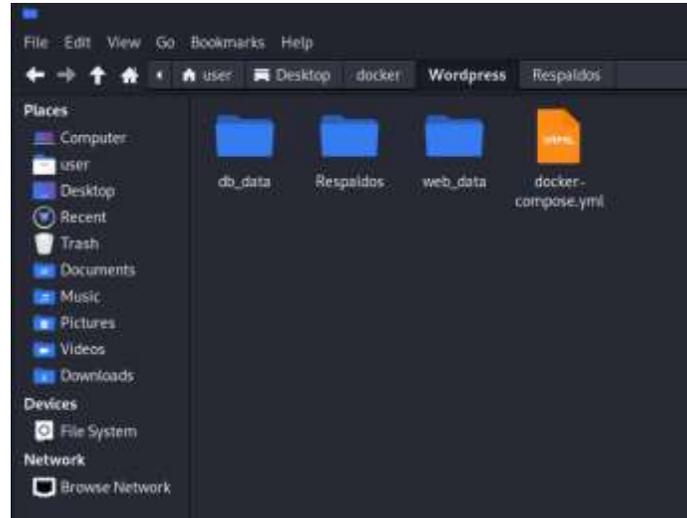


Ilustración 371: Carpeta de respaldos en el CMS

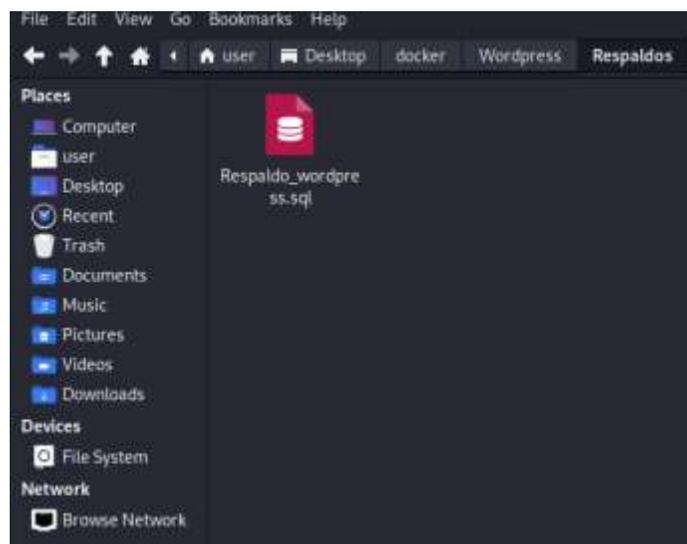


Ilustración 372: Respaldo creado

4. Eliminar datos de un contenedor (Volumen)

Esta función solo utiliza como parámetro el nombre del CMS, con ello sitúa la carpeta del CMS y posteriormente elimina las carpetas que contienen los datos generados por los contenedores (o sus volúmenes), luego solo nos mostrará un pequeño recuadro informativo que nos dirá que las carpetas que conformaban los



datos del CMS han sido eliminadas para posteriormente dar de baja los contenedores que conforman al CMS en cuestión.

```
NOTA: Recuerde realizar un respaldo en caso de querer conservar el sitio actual del CMS
Puede hacerlo desde la opción 'Crear respaldo de un CMS' del menú principal

Nombre del Contenedor: wordpress
[sudo] password for user:

----- Eliminando el volumen de los Contenedores -----
Nombre                Estado del volumen
wordpress_WEB         Eliminado ...
wordpress_DB          Eliminado ...
-----

Stopping wordpress_php_1 ... done
Stopping wordpress_web_1 ... done
Stopping wordpress_db_1 ... done
Removing wordpress_php_1 ... done
Removing wordpress_web_1 ... done
Removing wordpress_db_1 ... done
Removing network wordpress_default

----- El volumen de datos a sido eliminado -----

Desea salir?(s/n): █
```

Ilustración 373: Eliminar datos de un contenedor

5. Salir

Una pequeña función que le permitirá al usuario escoger si desea seguir trabajando dentro del programa o salir de este, esta función lee como parámetro si/no o también s/n, de cualquier forma, al equivocarnos el programa nos lo hará saber.



Conclusiones

La implementación de contenedores Docker como herramienta para el uso en la clase de Introducción a los CMS (Sistema de Gestión de Contenidos) conlleva una serie de ventajas como una fácil implementación ya que la implementación de un CMS en contenedores Docker es más rápida y sencilla en comparación con la configuración manual, lo que ahorra tiempo y también evita los problemas de compatibilidad con los componentes de los CMS.

De igual manera nos proporciona una optimización de recursos ya que los contenedores Docker son versiones simplificadas de sistemas operativos con solo las librerías necesarias para así poder ser implementados en máquinas de pocos recursos y así evitar que los estudiantes tengan que recurrir a una inversión económica innecesaria. También nos proporciona un alto grado de aislamiento, de manera que los problemas en un contenedor no afectarán a otros.



Recomendaciones

Dadas las grandes ventajas de crear un entorno controlado con todas las dependencias necesarias para el perfecto funcionamiento y control de los CMS, se recomienda implementar el uso del proyecto cuando se necesite trabajar diferentes proyectos referentes a los CMS, con el fin de ahorrar el mayor tiempo posible y evitar cualquier tipo de problemas de dependencias o conflictos entre estas al momento de trabajar con más de un CMS a la vez

Seguir las indicaciones que se especifican en la creación de cada contenedor propuesto y cuidar la sintaxis de los comandos a utilizar para lograr así la construcción y ejecución de estos de manera exitosa

Profundizar el estudio acerca de la versatilidad del uso de los contenedores de Docker al momento de trabajar diferentes aplicaciones que necesiten de dependencias y configuraciones específicas para su correcto funcionamiento o en su defecto algo que nuestra computadora no pueda ofrecer correctamente

Desarrollar un proyecto con los mismos objetivos, pero asegurándose de ampliar el paradigma y poniendo a prueba toda clase de implementaciones para otro tipo de aplicaciones fuera de los utilizados en el presente proyecto (para esto se debe tener un conocimiento previo sobre el uso de los contenedores de Docker).



Referencias bibliográficas

- Ali, A. (14 de Febrero de 2022). *Everything that You Should Know About Docker Hub*. Obtenido de <https://geekflare.com/docker-hub-introduction/>
- Charge, M. (2020). *Docker Easy: The Complete Guide on Docker World For Begginer*.
- Corbetti, S. (17 de Mayo de 2021). *Cómo utilizar Docker Compose*. Obtenido de <https://thesolving.com/es/contenerizacion/como-utilizar-docker-compose/>
- Delgado, G., & Mendoza, H. (2017). Recuperado el 10 de 05 de 2022, de Repositorio Institucional UNAN - LEON (RIUL): <http://riul.unanleon.edu.ni:8080/jspui/handle/123456789/6548>
- Docker, I. (2023). *Bitnami - Moodle*. Obtenido de <https://hub.docker.com/r/bitnami/moodle>
- Docker, M. d. (s.f.). *Docker Docs*. Obtenido de <https://docs.docker.com/storage/>
- Drupal. (2023). Obtenido de DrupalSoul: <https://www.drupalsoul.com/que-es-drupal-y-para-que-sirve>
- González, A. (2017). *Docker. Guía Practica. RC libros: Madrid*.
- Herrera, A. (2021). *Moodle*. Obtenido de Innovación y Cualificación: <https://www.innovacionycualificacion.com/plataforma-elearning/que-es-moodle-y-caracteristicas/>
- Joomla. (2024). Obtenido de El Blog de cdmon: <https://www.cdmon.com/es/blog/que-es-joomla-para-que-sirve-y-como-funciona>
- KodeKloud. (9 de Mayo de 2022). *KodeKloud*. Obtenido de <https://kodekloud.com/v3/>
- Mazariego, E., & Mora, L. (Junio de 2022). *Implementación de contenedores Docker como herramienta de virtualización liviana para el apoyo del proceso enseñanza-aprendizaje en la carrera de Ingeniería en Telemática, Dpto. Computación, UNAN-León*. Obtenido de Repositorio Institucional UNAN - LEON (RIUL): <http://riul.unanleon.edu.ni:8080/jspui/bitstream/123456789/9710/1/253094.pdf>
- McKendrick, R., & Gallagher, S. (2017). *Mastering Docker*. Birmingham, UK: Packt Publishing Ltd.
- Mouat, A. (2016). *Using Docker*. Estados Unidos: O'Reilly Media, Inc.



Muñoz, T., & Luis, J. (2017). Recuperado el 10 de 05 de 2022, de UPCommons: <https://upcommons.upc.edu/handle/2117/113040>

Networking overview. (s.f.). Obtenido de Docker Docs: <https://docs.docker.com/network/>

Prestashop. (18 de Mayo de 2018). Obtenido de Nivel de Calidad: <https://niveldecalidad.com/que-es-prestashop-y-para-que-sirve/>

Raj Pethuru, J. S. (2015). *Learning Docker.* Birmingham: Packt Publishing Ltd.

Ramírez, F. (2020). *Docker Certified Associate(DCA): Exam Guide.* Birmingham: Packt Publishing Ltd.

Rivera, A., Osejo, M., & González, V. (Noviembre de 2022). *CREACIÓN DE AMBIENTES AISLADOS DE PRUEBAS PARA LA EJECUCIÓN DE APLICACIONES EN PHPV8, LARAVELV8, .NETV5 Y RUBY ON RAILS V6.* Obtenido de Repositorio Institucional UNAN - LEON (RIUL): <http://riul.unanleon.edu.ni:8080/jspui/bitstream/123456789/9711/1/253095.pdf>

Turnbull, j. (2017). *The Docker Book.*

Villacampa, Ó. (29 de Enero de 2021). *Qué es Docker y para qué sirve.* Obtenido de Ondho: <https://www.ondho.com/que-es-docker-para-que-sirve/>

WordPress. (2023). Obtenido de HubSpot: <https://blog.hubspot.es/website/guia-completa-wordpress#:~:text=WordPress%20es%20una%20plataforma%20de,muy%20buena%20experiencia%20de%20usuario>

Zepeda, E. (29 de Enero de 2020). *Conoce los comandos básicos de Docker Compose.* Obtenido de Coffee bytes: <https://coffeebytes.dev/docker-compose-tutorial-con-comandos-en-gnu-linux/>

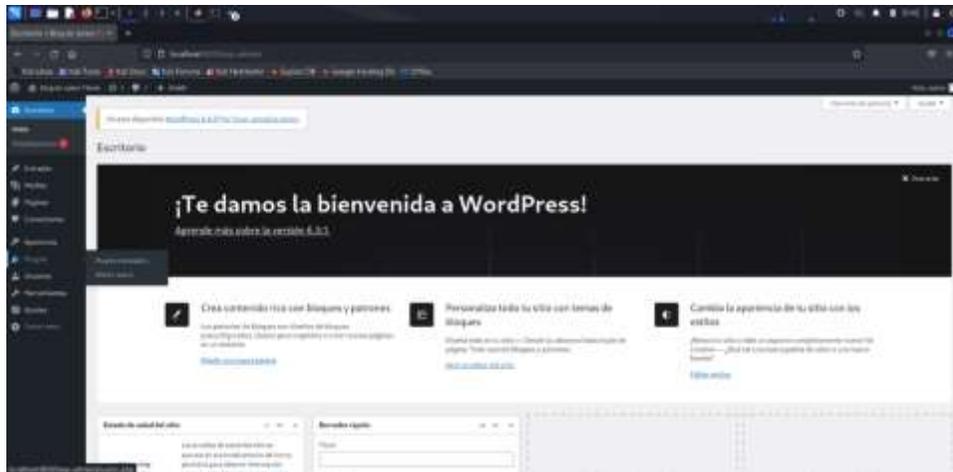


Anexos

La relevancia de esta monografía se extiende a la Carrera de Ingeniería en Telemática, particularmente en los componentes de "Electiva VI: Comercio Electrónico" y "Electiva X: Administración de Servidores". En el contexto de Comercio Electrónico, se emplea el CMS de WordPress, y el script creado facilita la gestión al eliminar posibles problemas de incompatibilidad de versiones. Además, proporciona a los estudiantes una introducción a comandos básicos de Docker, habilidad que resultará fundamental en fases posteriores con el componente de Administración de Servidores.

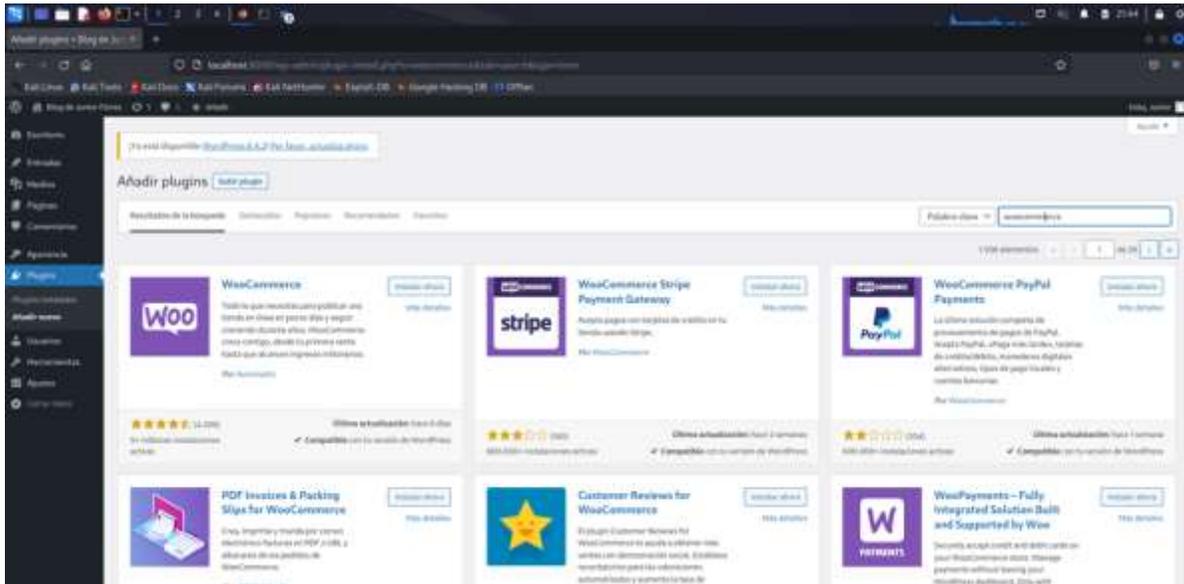
Electiva VI: Comercio Electrónico

Para el componente de "Electiva VI: Comercio Electrónico", se hace el uso del CMS WordPress, en específico el plugin de WooCommerce. El cual para su instalación solo debemos irnos al apartado de plugin del panel principal:

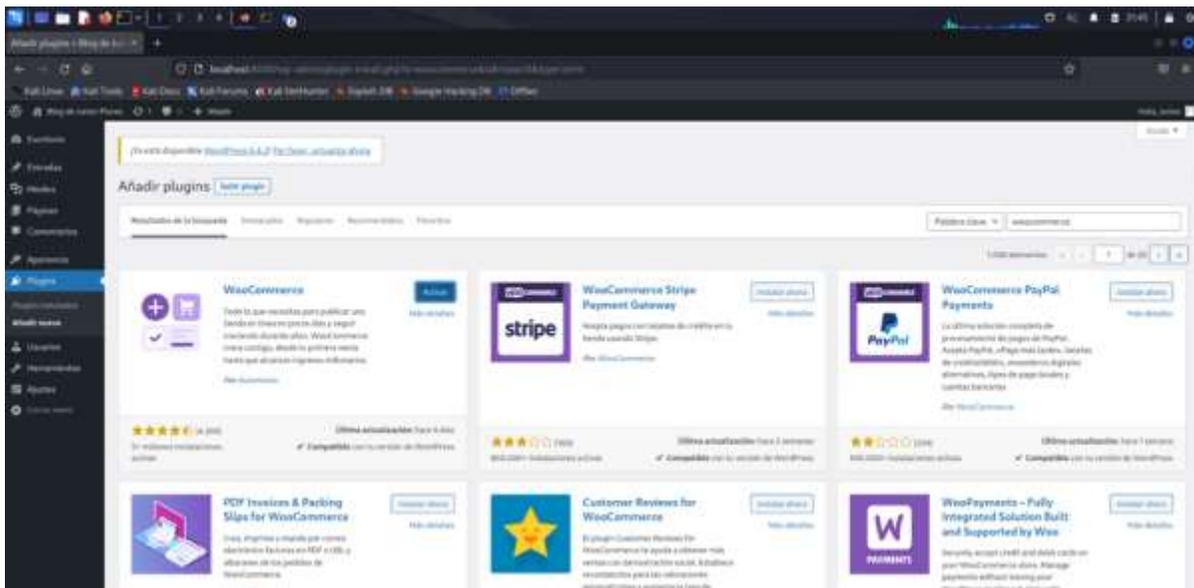




Seleccionamos añadir uno nuevo, buscamos woocommerce y lo instalamos:



Luego de instalarlo debemos activarlo:

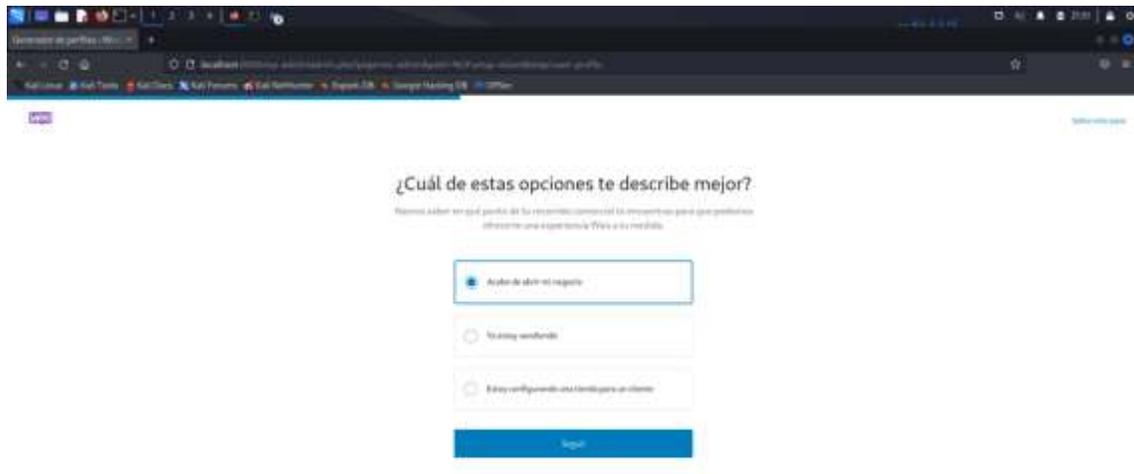




Una vez activado solo debemos realizar unos pasos para la configuración de woocommerce:



Seleccionamos cualquier opción la más acorde a las necesidades del administrador y damos en seguir:





Rellenamos los campos necesarios y damos clic en seguir:

Cuéntanos algo más sobre tu tienda

Únanse las características de tu tienda para ayudarte a configurar pagos, envíos e impuestos, así como para recomendarle el mejor tema para tu tienda.

País o región de tu tienda

Tienda de venta física

El tipo de tienda que quieres configurar

País o región de tu tienda (para el envío de paquetes)

Estados Unidos y territorios

País o región de tu tienda (para impuestos)

Estados Unidos y territorios

Nombre de tu tienda (opcional)

pruebas123@amazon.com

Necesito ayuda para configurar mi tienda. Necesito ayuda para configurar mi tienda.

Seguir

Con las siguientes opciones podemos decidir si activarlas o no, dependiendo de las necesidades de la tienda:

Consigue un empujón con nuestras características gratuitas

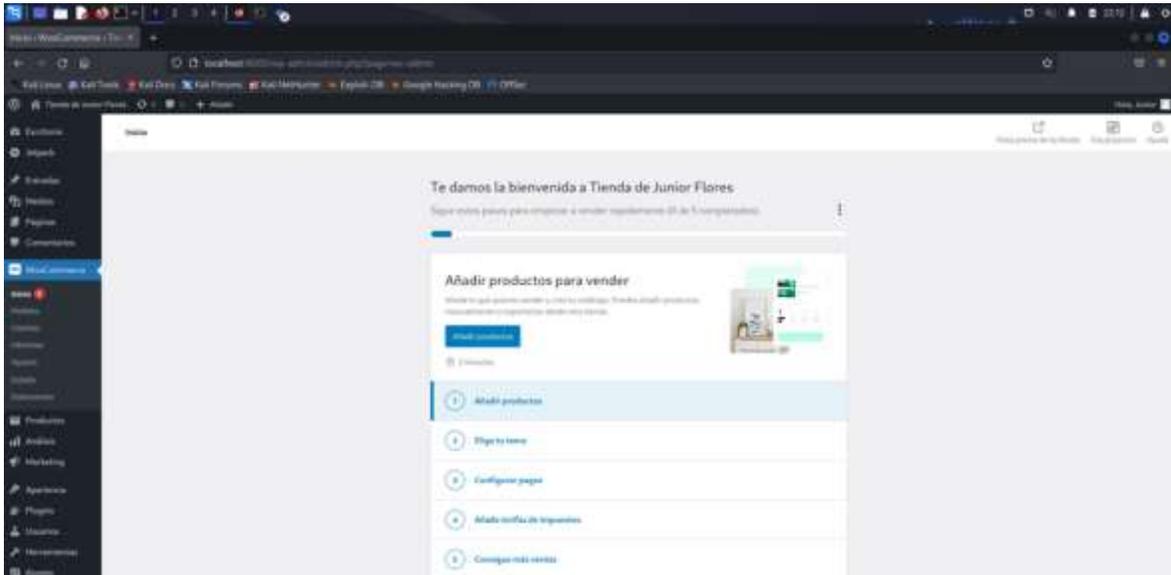
Registra tu tienda y obtén acceso a las características gratuitas para negocios. No se requieren tarjetas de crédito o cuentas de pago para activarlas.

Activación de envío de paquetes con el sistema de IA de Amazon Ofrece a tus clientes el envío de paquetes gratis a través de Amazon de 1 día y protege de fraude con el pago por adelantado.	Hazte un productor con Prime Recibe el primer envío gratuito en Amazon Prime. Disponible.
Llega a los clientes con MailPost Envía tus productos directamente a los departamentos de ventas, marketing y operaciones de Amazon.	Sumerte al marketing con Google Listings & Ads Crea y optimiza tu tienda en Google Shopping y Google Ads para atraer a los clientes que buscan tus productos.

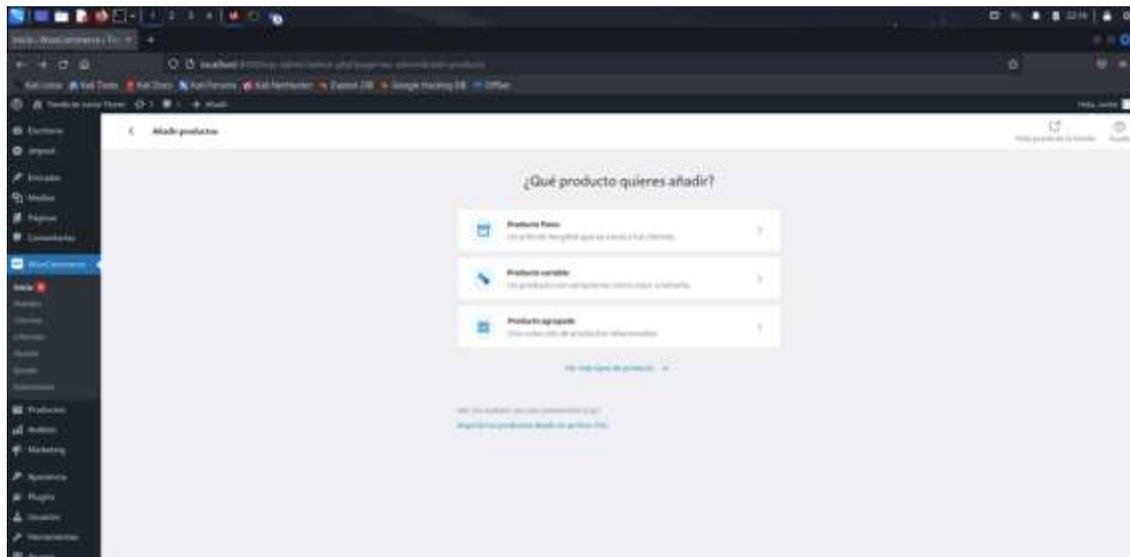
Seguir



Una vez instalado solo debemos completar 5 pasos más para terminar de configurar la tienda:

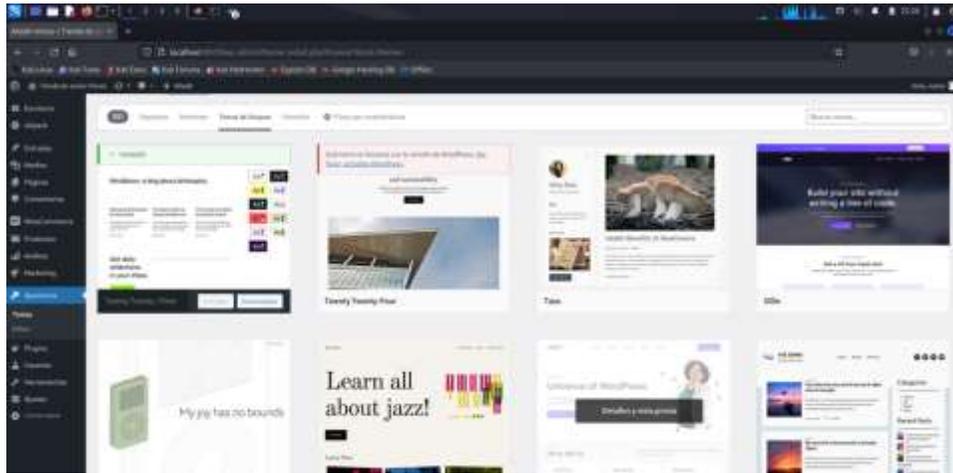


Para agregar un producto se nos presenta 3 opciones:



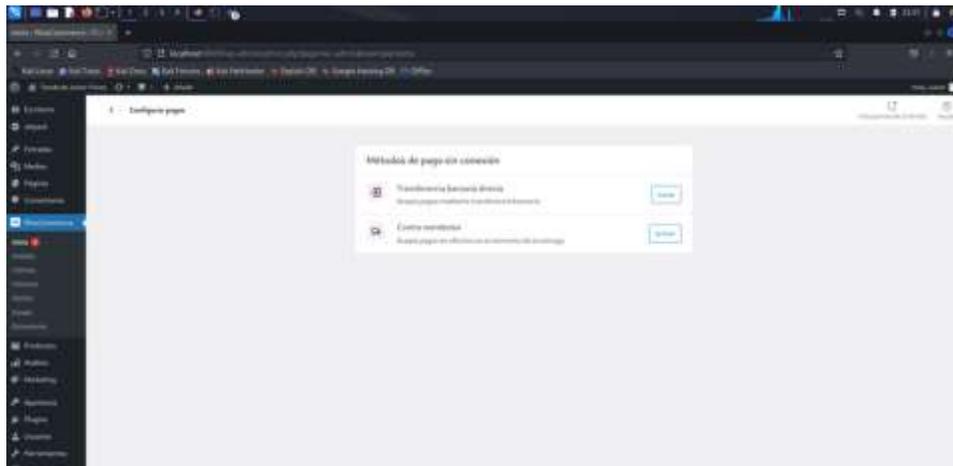


El siguiente paso es elegir un tema:



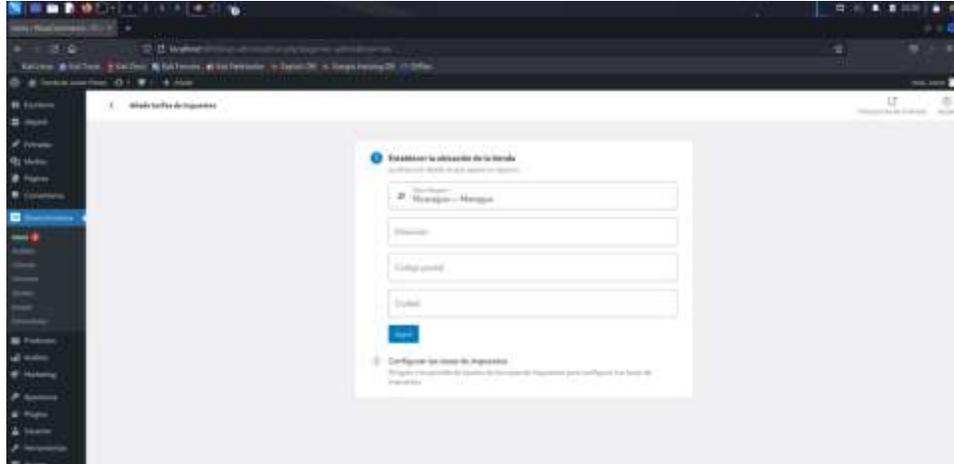
En el cual solo debemos instalar y activar el tema de preferencia.

El siguiente paso es configurar los pagos para la tienda, esto va de acorde a la preferencia del administrador de la tienda:

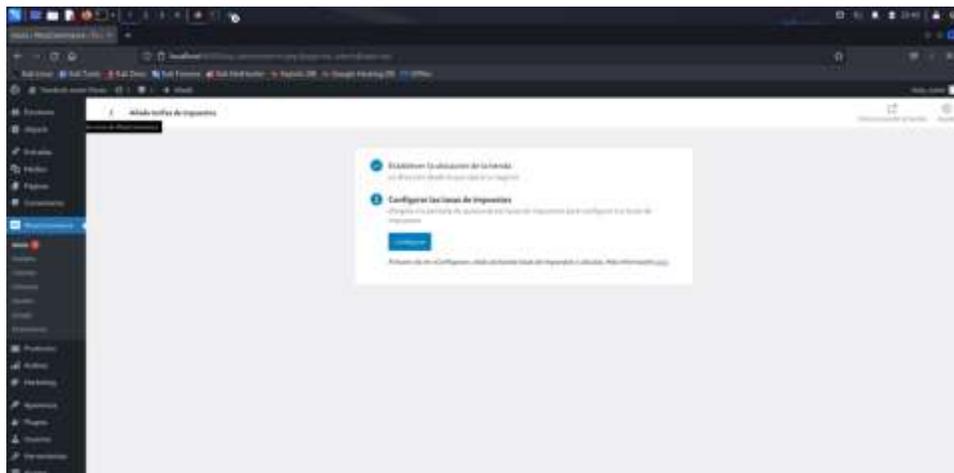


Si se elige las transferencias bancarias de deben agregar los datos de la cuenta a la que se harán las transferencias y si activamos la opción de contra reembolso el cliente podrá ver esa opción disponible al momento de ver un producto.

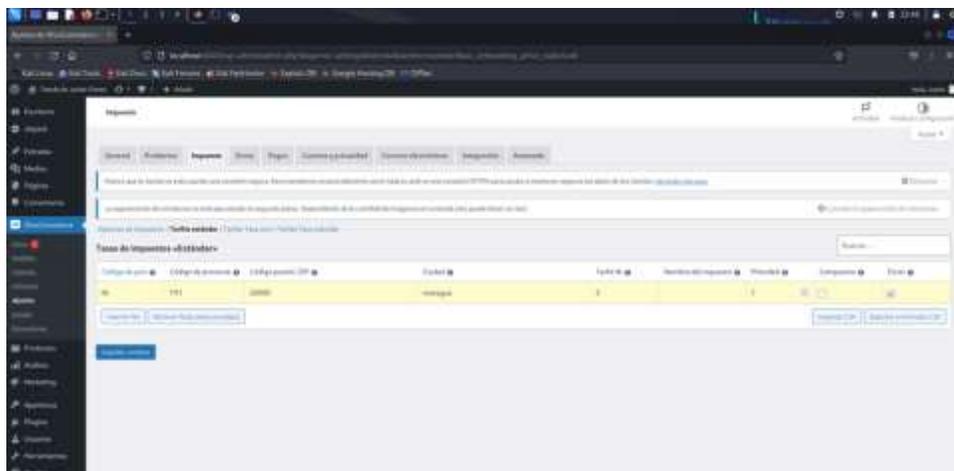
El siguiente paso es de configurar las tarifas de impuestos:



Llenamos los datos que se requieran y damos clic en seguir y luego en configurar:

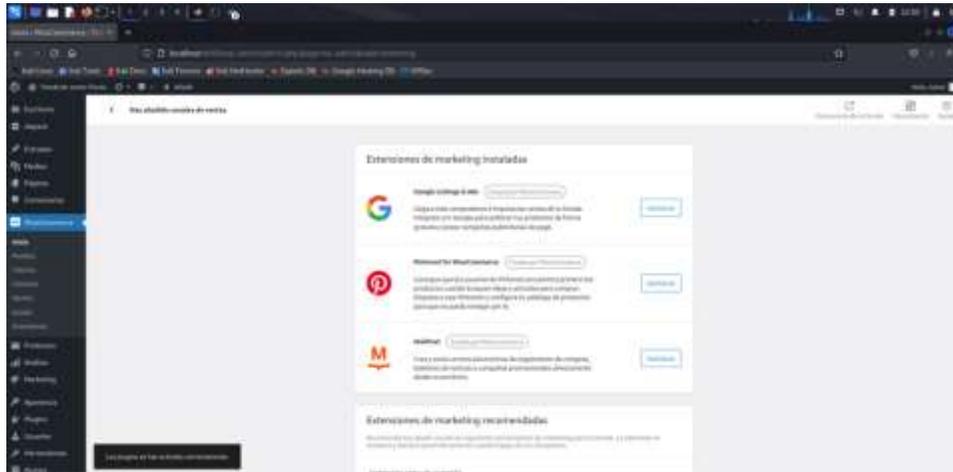


Luego se debe configurar los impuestos a como se requiera y damos clic en finalizar configuración:

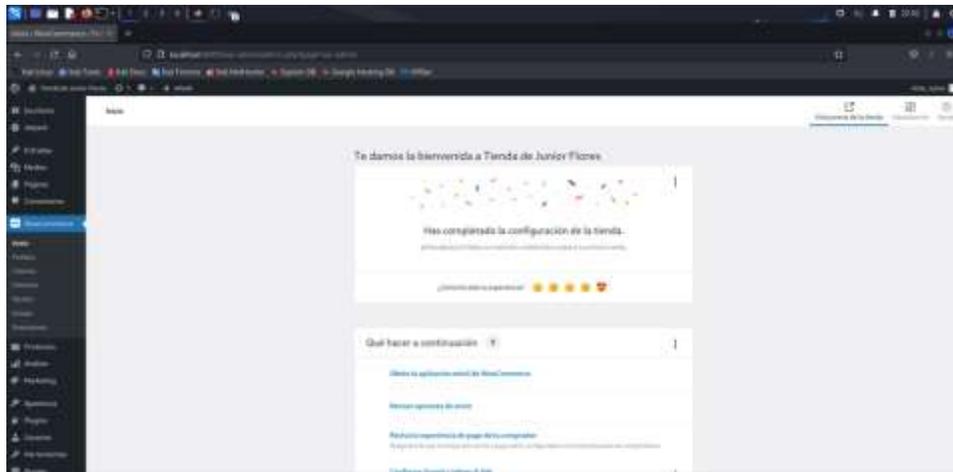




El último paso tiene que ver con la opción de darle un impulso a la tienda, aquí se elige la preferencia del administrador:



Una vez finalizado esto ya tendríamos en marcha nuestra tienda en woocommerce:



Y solo quedaría agregar más productos o hacer ajuste según se desee.



Glosario

- Compose: es una herramienta para definir y ejecutar aplicaciones complejas con Docker. Con Compose, usted define una aplicación de múltiples contenedores en un solo archivo, luego activa su aplicación en un solo comando que hace todo lo necesario para que se ejecute. También conocido como Docker Compose.
- Docker Desktop para Linux: Docker Desktop para Linux es un entorno de desarrollo Docker liviano y fácil de instalar diseñado específicamente para máquinas Linux. Es la mejor solución si desea crear, depurar, probar, empaquetar y enviar aplicaciones Dockerizadas en una máquina Linux.
- Docker Hub: Es un recurso centralizado para trabajar con Docker y sus componentes
- Docker ID: Su Docker ID gratuito le otorga acceso a los repositorios de Docker Hub y a algunos programas beta. Todo lo que necesita es una dirección de correo electrónico.
- Dockerfile: es un documento de texto que contiene todos los comandos que normalmente ejecutaría manualmente para crear una imagen de Docker. Docker puede crear imágenes automáticamente leyendo las instrucciones de un Dockerfile.
- build: es el proceso de crear imágenes de Docker utilizando un Dockerfile. La compilación utiliza un Dockerfile y un "contexto". El contexto es el conjunto de archivos en el directorio en el que se construye la imagen.
- Docker: Plataforma de contenedor de software diseñada para desarrollar, enviar y ejecutar aplicaciones aprovechando la tecnología de los contenedores.



- Contenedor Docker: Formato que empaqueta todo el código y las dependencias de una aplicación en un formato estándar que permite su ejecución rápida y fiable en entornos informáticos
- Image: Las imágenes de Docker son la base de los contenedores. Una imagen es una colección ordenada de cambios en el sistema de archivos raíz y los parámetros de ejecución correspondientes para su uso dentro de un tiempo de ejecución de contenedor. Una imagen normalmente contiene una unión de sistemas de archivos en capas apilados uno encima del otro. Una imagen no tiene estado y nunca cambia.
- Parent Image: La imagen principal de una imagen es la imagen designada en la directiva FROM en el Dockerfile de la imagen. Todos los comandos posteriores se basan en esta imagen principal. Un Dockerfile con la directiva FROM scratch no utiliza ninguna imagen principal y crea una imagen base.
- cluster: Un clúster es un grupo de máquinas que trabajan juntas para ejecutar cargas de trabajo y proporcionar alta disponibilidad.
- Máquina virtual: Una máquina virtual es un programa que emula una computadora completa e imita un hardware dedicado. Comparte recursos físicos de hardware con otros usuarios, pero aísla el sistema operativo. El usuario final tiene la misma experiencia en una máquina virtual que en un hardware dedicado. En comparación con los contenedores, una máquina virtual es más pesada de ejecutar, proporciona más aislamiento, obtiene su propio conjunto de recursos y comparte un mínimo. También conocido como VM
- Nodo: Un nodo es una máquina física o virtual que ejecuta una instancia de Docker Engine en modo enjambre.
- Volume: Un volumen es un directorio especialmente designado dentro de uno o más contenedores que omite el Union File System. Los volúmenes están diseñados para conservar datos, independientemente del ciclo de vida del contenedor. Por lo tanto,



Docker nunca elimina automáticamente volúmenes cuando elimina un contenedor, ni "recoge basura" volúmenes a los que ya no hace referencia un contenedor. También conocido como: volumen de datos.

- Host Volume: Un volumen de host reside en el sistema de archivos del host de Docker y se puede acceder a él desde dentro del contenedor
- Volumen con nombre: Un volumen con nombre es un volumen que Docker administra en el disco donde se crea el volumen, pero se le asigna un nombre.
- Volumen anónimo: Un volumen anónimo es similar a un volumen con nombre; sin embargo, puede resultar difícil hacer referencia al mismo volumen a lo largo del tiempo cuando se trata de un volumen anónimo. Docker maneja dónde se almacenan los archivos.
- Repositorio: Un repositorio es un conjunto de imágenes de Docker. Se puede compartir un repositorio enviándolo a un servidor de registro. Las diferentes imágenes del repositorio se pueden etiquetar mediante etiquetas.
- btrfs: btrfs (sistema de archivos B-tree) es un sistema de archivos Linux que Docker admite como backend de almacenamiento. Es un sistema de archivos de copia sobre escritura.
- copy-on-write: Docker utiliza una técnica de copia en escritura y un sistema de archivos de unión tanto para imágenes como para contenedores para optimizar los recursos y acelerar el rendimiento. Varias copias de una entidad comparten la misma instancia y cada una realiza solo cambios específicos en su capa única.
- Almacenamiento persistente: El almacenamiento persistente o el almacenamiento de volumen proporciona una manera para que un usuario agregue una capa persistente al sistema de archivos del contenedor en ejecución. Esta capa



persistente podría residir en el host del contenedor o en un dispositivo externo. El ciclo de vida de esta capa persistente no está conectado al ciclo de vida del contenedor, lo que permite al usuario conservar el estado

- controlador de almacenamiento superpuesto: OverlayFS es un servicio de sistema de archivos para Linux que implementa un montaje de unión para otros sistemas de archivos. Es compatible con el daemon Docker como controlador de almacenamiento.



Cronograma de actividades

Actividades	2023																2024			
	Septiembre				Octubre				Noviembre				Diciembre				Enero			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Antecedentes		■																		
Planteamiento del problema		■	■																	
Justificación			■	■																
Objetivos				■																
Marco teórico			■	■	■	■	■	■	■											
Diseño metodológico				■	■	■	■	■	■	■										
Creación de contenedores							■	■	■	■	■									
Desarrollo de prácticas							■	■	■	■	■	■	■	■						
Creación de programa										■	■	■	■	■	■	■				
Conclusión																	■	■		
Recomendaciones																	■	■	■	
Referencias bibliográficas																		■	■	■
Anexos																			■	■