UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA.

UNAN-LEÓN.

Facultad de Ciencias.

Monografía para optar al título de Ingeniero en Sistemas de Información.

"DESARROLLO DE UN PORTAL WEB PARA EL INTERCAMBIO DE INFORMACIÓN ENTRE AMIGOS UTILIZANDO EL MODELO DE PROGRAMACIÓN EN TRES CAPAS CON PHP, CSS, AJAX Y CONTROLES SPRY"

Autores:

Br. Ana Indira Gómez García.

Br. Javier Alberto Guido Reyes.

Br. Meylin Danelia Rodríguez Flores.

Tutor:

Ing. Denis Leopoldo Espinoza Hernández.

León, 22 de Mayo del 2009.

AGRADECIMIENTO

Le agradecemos primeramente a ***Dios***; por darnos la vida y por permitirnos llegar a culminar nuestro trabajo**.**

A nuestros ***padres*** por su ayuda incondicional, dedicación y confianza en nosotros, sin ellos no podríamos haber llegado hasta aquí.

A nuestro ***tutor*** Ing. Denis Espinoza, por brindarnos su tiempo, paciencia y colaboración para terminar este trabajo.

A nuestras ***familiares*** y ***amigos*** por su apoyo y motivación para que saliéramos adelante.

Y a todos aquellos que de una manera nos desearon éxitos en nuestra monografía.

DEDICATORIA

La presente tesis se la dedico con todo amor y cariño a:

Dios por haberme dado la oportunidad de vivir y regalarme una familia maravillosa.

Con mucho cariño a ***mis padres*** que han estado conmigo en todo momento y brindarme todo su amor, por darme una carrera y por creer en mí.

A ***mi hijo y esposo*** por ser mi fortaleza y mi espíritu de superación.

A ***mis hermanos*** por estar conmigo y apoyarme siempre, los quiero mucho.

A ***mi tutor*** por confiar en mí, tener toda la paciencia y habernos dedicado tanto tiempo.

A ***mis compañeros de tesis***, no puedo irme sin decirles, que sin ustedes a mi lado no lo hubiera logrado.

Se las dedico a todos ustedes con toda mi alma, por haber llegado a mi vida y compartir momentos agradables y momentos tristes, pero esos momentos son los que nos hacen crecer y valorar a las personas que nos rodean los quiero mucho y nunca los olvidaré.

Y a todas las personas que de una u otra forma hicieron posible la realización de este trabajo.

Ana Indira Gómez García.

DEDICATORIA

A Dios

Por darme el regalo de la vida, y por darme el mayor regalo de todos como fue la vida de su Hijo Jesús para darnos salvación.

A mis Padres: *Gilberto José Guido y Vidalia del Pilar Reyes*

Por enseñarme el verdadero camino desde temprana edad, por su apoyo incondicional durante toda mi formación, quienes depositaron en mí toda su confianza.

A mis Hermanos: *Homer Daniel Guido Reyes y Rebeca de los Ángeles Guido Reyes*

Por su valioso apoyo en cada momento, y por confiar en mis capacidades para alcanzar el éxito.

A mi Sobrina: *María Daniela Guido Picado*

Por ser la nueva luz que llegó a nuestra familia y llegar a convertirse en una gran fuente de inspiración para mí.

A los FIRST

Por su amistad incondicional, preocupación, y por darme esas palabras de ánimo cuando las necesitaba.

Y a todas aquellas personas que dieron una pincelada para terminar de dibujar este éxito de mi vida.

DEDICATORIA

Primeramente a ***Dios*** padre, Hijo e Espíritu Santo que son uno solo; por la vida que me ha dado para seguir luchando por cada triunfo que me he propuesto a lograr, por la consolación espiritual que he otorgado durante dificultades en la vida y por la gran misericordia y el amor que tiene hacia mí.

A mis padres: Enriqueta del Carmen Flores Ramírez y Mario Ramón Rodríguez Escobar por el apoyo económico, moral que me han dado, sin sus ayudas no hubiese seguido mis estudios, les dedico este triunfo a ellos porque es de ellos este mérito y el gran paso que me han dado en mi vida profesional.

A mi tutor Ing., Denis Leopoldo Espinoza por brindarnos de su tiempo y la gran ayuda que nos proporcionó para el logro de la culminación de nuestra monografía.

A mis *compañeros de tesis* les hago el homenaje por compartir nuestros conocimientos, nuestros tiempos, por la unidad y la compresión entre los tres, por el esfuerzo y sobre todo por la amistad que se creó durante este periodo de trabajo.

A todos mis hermanos y hermanas, por el apoyo incondicional que dedicaron en mi en momentos de dificultades personales, sociales, económicas, moral y sobre todo de salud.

A mis amigos y amigas que de una forma incondicional me motivaron para seguir en mis estudios, y a todos aquellos que me apoyaron, que brindaron su confianza en mis capacidades.

INDICE

INDICE DE FIGURAS

INTRODUCCIÓN

Los portales web por sus características como "distribuidores" de información y "almacenes" de recursos, constituyen un soporte ideal y eficiente de información. Por otro lado suponen la plataforma donde un grupo particular pueda establecer un punto de encuentro e intercambiar información.

A la vez facilitan la construcción de comunidades virtuales (chat), facilidades de comercialización (anuncio de clasificación), aplicación de productividad personal (email), avisos, etc.

En el presente documento se pretende detallar como se llevó a cabo la realización de un portal web para amigos en línea utilizando el modelo de programación en tres capas para lo cual se hizo uso de la programación orientada a objetos en PHP. Además se dotó al portal web de gran interactividad gracias al uso de AJAX (***A****synchronous* ***J****avaScript* ***A****nd* ***X****ML*) lo cual permitió dar una mejor experiencia a los usuarios.

Por último se integró dentro del portal web, algunas de las novedades que proporciona para los desarrolladores web el Dreamweaver CS3 dentro de lo cual destacan los controles Spry.

Invitamos al lector interesado a profundizar en las nuevas alternativas para el desarrollo web, leer este documento que le servirá de apoyo y así enriquecer sus conocimientos.

ANTECEDENTES

El nacimiento de los portales web es a finales de 1996, cuando los buscadores, liderados por Yahoo y en menos medida por AltaVista, comenzaron a ampliar sus páginas principales (Home Pagés) y a ofrecer algunos contenidos (noticias, resultados deportivos…) de gran interés para sus millones de visitantes. El objetivo era, y sigue siendo, conseguir la fidelidad a los usuarios y que establezcan la página del portal como Home Page propia.

Paralelamente a esta expansión de los buscadores, se observa durante 1997 una ampliación de contenidos y servicios en las webs que pueden convertirse en portales web.

El uso de los portales web proporciona facilidades como rapidez en la búsqueda de información, transacciones, fiabilidad por lo que éstos han tenido gran progreso a través del tiempo, siendo introducido en nuestra universidad aproximadamente en el año 2000.

En los años anteriores estudiantes de Ingeniería en Sistema de la Universidad Autónoma de Nicaragua UNAN-LEON han realizado otros proyectos parecidos, en concreto tenemos el proyecto o trabajo de investigación:

- "Portal web con información de los egresados de la UNAN-LEON, implementando software libre" (2007), Br. Silvio Heriberto Pérez Delgado, Br. Edgar René Picado Venegas, y Br. Gisela Marbely Salgado Dubón.

JUSTIFICACIÓN

La razón de desarrollar este portal web, es proporcionar a los académicos y estudiantes un documento que pueda servirles de base para ampliar y actualizar los conocimientos que ellos posean acerca del desarrollo de páginas web. Por tal motivo, en el desarrollo del portal web se tuvieron en cuenta la profundización en los siguientes aspectos:

- Implementación de Modelo en tres capas con el objetivo de separar la lógica de negocio de la lógica de diseño. Este modelo nace de la necesidad de solventar problemas en aplicaciones compactas por lo cual se ha generalizado la división de las aplicaciones en capas que normalmente serán tres: una capa que servirá para guardar los datos (base de datos), una capa para centralizar la lógica de negocio en donde van a residir todos los procesos que darán respuesta al usuario y por último una interfaz gráfica que facilite al usuario el uso del sistema.

- Utilización de PHP5 que cuenta con innumerables mejoras que consolidan su éxito. Se ofrece la posibilidad de hacer páginas orientados a objetos, utilización de la base de datos ligera SQLite o la implementación de servicios Web.

- Incorporación de Ajax para garantizar una mejor experiencia al usuario, logrando una mayor interactividad, velocidad y usabilidad en las aplicaciones, al no tener que recargar las páginas constantemente.

- Integración de los controles Spry ya que es un método sencillo y elegante para mejorar el aspecto y el funcionamiento de un sitio Web y crear elementos de página interactivos que muestren datos dinámicos sin necesidad de actualizar la página entera.

- Utilización de hojas de estilo para el formato común de los documentos. La separación del contenido y la presentación hace que resulte mucho más fácil mantener el aspecto del sitio desde una ubicación central, ya que no es necesario actualizar las propiedades de las distintas páginas cuando se desea realizar algún cambio.

OBJETIVOS

Objetivo General:

- Desarrollar un portal web para el intercambio de información entre amigos utilizando el modelo de programación en tres capas con PHP, CSS, AJAX Y Controles Spry.

Objetivos Específicos:

- Implementar el Modelo en tres capas para una mejor organización del código y del diseño dentro del portal web.
- Utilizar la programación orientada a objetos como técnica de programación utilizando el soporte para clases que ofrece PHP para poder implementar el modelo en tres capas.
- Aplicar JavaScript para implementar AJAX y dotar a nuestro portal web de una gran interactividad y mejorar así la experiencia de usuario.
- Integrar los Controles Spry Avanzados proporcionados por Dreamweaver CS3 para mejorar la organización de los elementos dentro del portal web.
- Utilizar las hojas de estilo en cascada (CSS) para dotar de un formato común a todas las páginas que conforman nuestro portal web.
- Generar un Documento que sirva de apoyo para quienes estén interesados en este tipo de trabajo.

DISEÑO METODOLÓGICO

Para llevar a cabo el desarrollo de este trabajo se implementó la metodología del ciclo de vida clásico o en cascada.

Entiéndase por ciclo de vida todas aquellas etapas que asocian una serie de tareas que se deberán realizar incluyendo los documentos que cada una de estas generarán, y que serán las que servirán de entrada a las siguientes fases; por las cuales pasa el software, desde que el proyecto es concebido hasta que éste es dejado de usarse.

Modelo en Cascada

Descompone el proceso de desarrollo en diferentes fases, constituyendo la salida de cada una de ellas la entrada requerida por la siguiente. En éste modelo se supone que todos los requisitos son conocidos y comprendidos perfectamente al iniciar el desarrollo del software.

Actividades del Ciclo de Vida en Cascada

- **Análisis:** Se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge una memoria llamada SRD (documento de especificación de requisitos), que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos.
  Es importante señalar que en esta etapa se deben reunir todo lo que se requiere del sistema y será aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software.

- **Diseño:** Se descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el Documento de Diseño Datos, que contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras. Otros documentos que salen de esta etapa es el diagrama de clase, diagrama relacional, diseño arquitectónico, y el mapa de sitio.

- **Codificación:** Es la fase de programación o implementación propiamente dicha. Aquí se implementa el código fuente, haciendo uso de prototipos así como pruebas y ensayos para corregir errores.
  Dependiendo del lenguaje de programación y su versión se crean las bibliotecas y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.

- **Prueba:** El software debe ser probado para descubrir los defectos que puedan existir en la función, en la lógica y en la implementación.

- **Mantenimiento:** La fase de mantenimiento se centra en el cambio. Esta fase aplica los pasos de las fases de definición y de desarrollo, pero en el contexto del software ya existe. Durante la fase de mantenimiento se centran tres tipos de cambios: *corrección, adaptación y mejora.*

Gráficamente el modelo de **ciclo de vida clásico** se representa de la siguiente manera:

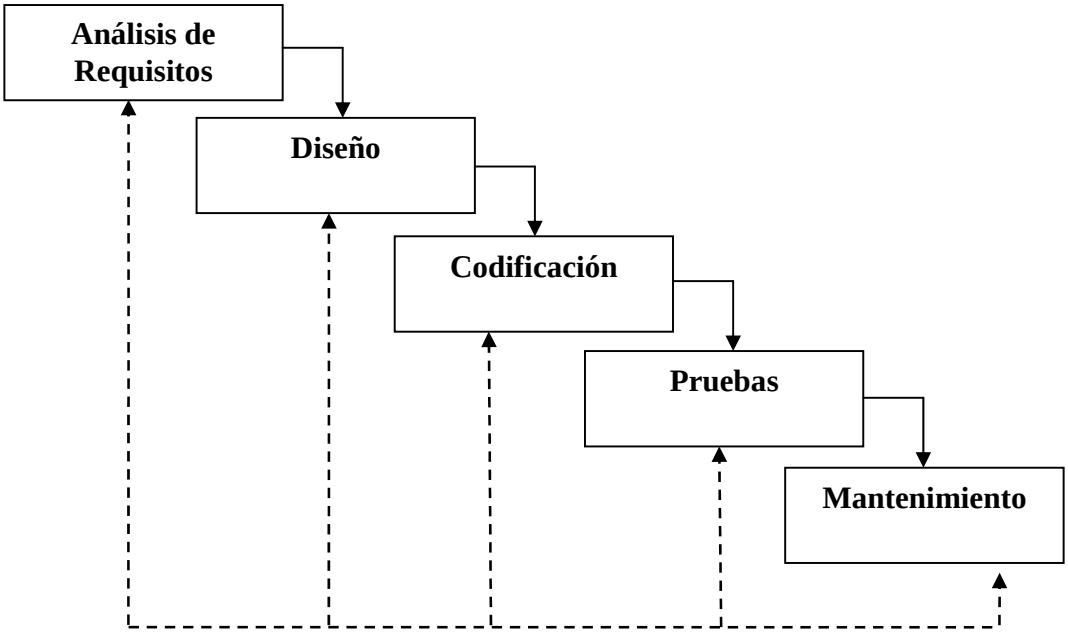


Figura 1 Ciclo de vida clásico

Materiales

a. ***Hardware****:* En nuestro proyecto a nivel de hardware emplearemos las siguientes herramientas:

- 3 PC con las siguientes características:
  - ✓ Memoria RAM de 1 GB
  - ✓ 160 GB en Disco Duro
  - ✓ Procesador Pentium 4 de 2.66 GHz

b. ***Software****:* Las herramientas software que emplearemos en nuestra aplicación son:

- ✓ Microsoft Office Word 2007.
- ✓ Microsoft Office Visio 2007.
- ✓ Windows XP (Sistema Operativo)
- ✓ MySQL Server 4.1
- ✓ PHPEdit 2.12.8.
- ✓ Adobe Dreamweaver CS3.
- ✓ XAMPP Para Windows Versión 1.6.6a
  - Apache http Server Version 2.2.
  - PhpMyAdmin 2.11.4.
  - PHP Versión 5.2.5.

MARCO TEÓRICO

Hoy en día, la Internet es un medio de comunicación pública, cooperativa y autosuficiente en términos económicos, accesible a cientos de millones de personas en el mundo entero. Físicamente, el Internet usa parte del total de recursos actualmente existentes en las redes de telecomunicaciones.

Uno de los servicios que más éxito ha tenido en Internet ha sido la World Wide Web (WWW, o "la Web"), hasta tal punto que es habitual la confusión entre ambos términos.

WEB

World Wide Web, (Red Global Mundial) o simplemente **Web**, es el universo de información accesible a través de Internet, una fuente inagotable del conocimiento humano.

El componente más usado en el Internet es definitivamente el Web. Su característica sobresaliente es el texto remarcado, un método para referencias cruzadas instantáneas. En la mayoría de los Sitios Web, ciertas palabras aparecen en texto de otro color diferente al resto del documento. Por lo general, este texto es subrayado. Al seleccionar una palabra o frase, uno es transferido al sitio o página relacionada a esa frase. En algunas ocasiones hay botones, imágenes, o porciones de imágenes que pueden activarse mediante un clic. Si usted mueve el apuntador sobre el contenido del documento y el apuntador cambia a un símbolo con una mano, eso indica que Usted puede realizar un clic para ser transferido a otro sitio.

PÁGINA WEB

Una página de Internet o **página Web** es un documento electrónico que contiene información específica de un tema en particular y que es almacenado en algún sistema de cómputo que se encuentre conectado a la red mundial de información denominada Internet, de tal forma que este documento pueda ser consultado por cualquier persona que se conecte a esta red mundial de comunicaciones y que cuente con los permisos apropiados para hacerlo.

Una página Web es la unidad básica del World Wide Web.

Una página Web tiene la característica peculiar de que el texto se combina con imágenes para hacer que el documento sea dinámico y permita que se puedan ejecutar diferentes acciones, una tras otra, a través de la selección de texto remarcado o de las imágenes, acción que nos puede conducir a otra sección dentro del documento, abrir otra página Web, iniciar un mensaje de correo electrónico o transportarnos a otro Sitio Web totalmente distinto a través de sus hipervínculos.

SITIO WEB

En inglés website o web site, un **sitio web** es un sitio (localización) en la World Wide Web que contiene documentos (páginas web) organizados jerárquicamente. Cada documento (página web) contiene texto y o gráficos que aparecen como información digital en la pantalla de un ordenador. Un sitio puede contener una combinación de gráficos, texto, audio, vídeo, y otros materiales dinámicos o estáticos.

Cada sitio web tiene una página de inicio **(**en inglés Home Page**)**, que es el primer documento que ve el usuario cuando entra en el sitio web poniendo el nombre del dominio de ese sitio web en un navegador. El sitio normalmente tiene otros documentos (páginas web) adicionales. Cada sitio pertenece y es gestionado por un individuo, una compañía o una organización.

PORTAL WEB

Un **portal web** de Internet es un sitio web cuyo objetivo es ofrecer al usuario, de forma fácil e integrada, el acceso a una serie de recursos y de servicios, entre los que suelen encontrarse buscadores, foros, documentos, aplicaciones, compra electrónica, etc. Principalmente están dirigidos a resolver necesidades específicas de un grupo de personas o de acceso a la información y servicios de una institución pública o privada.

Modalidades

Existen tres modalidades de portales web:

1. Portales horizontales, también llamados portales masivos o de propósito general, se dirigen a una audiencia amplia, tratando de llegar a toda la gente con muchas cosas. Como ejemplo de portales de esta categoría están Terra, AOL, AltaVista, UOL, Lycos, Yahoo, MSN.
2. Portales verticales, se dirigen a usuarios para ofrecer contenido dentro de un tema específico como puede ser un portal de música, empleo, inmobiliario, un portal de finanzas personales, arte o de deportes.
3. Portales diagonales: se trata de una mezcla entre el portal horizontal y el vertical. Se trataría de portales que utilizan redes sociales o aplicaciones generalistas como Facebook, Linkd, Flickr o YouTube… complementados con contenidos y/o utilidades dirigidas a un público muy concreto.

Los portales normalmente tienen programación que requiere muchos recursos computacionales y por su alto tráfico generalmente se hospedan en servidores de Internet dedicados.

Diferencia entre portal web y un sitio web

Desde el punto de vista funcional, un **Portal web** se diferencia de un **Sitio Web** convencional debido a que incluye muchas más Secciones, como pueden ser Noticias, Eventos, Directorios, que requieren estar siempre actualizados. Por otro lado, normalmente se caracterizan por administrar claves de acceso, destinadas a restringir la visita a determinadas páginas con información de alto valor.

Desde el punto de vista técnico, a diferencia del Sitio Web, un Portal es de mayor tamaño, ya que incluye mayor número de Secciones o Páginas, además incorpora información de tipo "Dinámica" que requiere una actualización periódica, para lo cual el desarrollo es más complejo y se utiliza tecnología sobre base de datos, lo que permite una más ágil administración de los contenidos.

LENGUAJE HTML

HTML, siglas de HyperText Markup Language (*Lenguaje de Marcas de Hipertexto*), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de "etiquetas", rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un *script* (por ejemplo Javascript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

```
<!--Documento de Html-->
<HTML>    <!--inicio de html-->
   <HEAD>    <!--Apertura de la cabecera del documento de html-->
     <TITLE>Aquí va el titulo de la página</TITLE>    <!--declaramos el Titulo del documento-->
  </HEAD>    <!--Cierre de la cabecera del documento de html-->
  <BODY>      <!--Apertura del cuerpo del documento de html-->
    Aquí va el cuerpo de la página
 </BODY>    <!--Cierre del cuerpo de html-->
 </HTML>     <!--Cierre del document de html-->
```

LOS CONTROLES SPRY

Los widgets de Spry son componentes comunes de interfaz de usuario listos para ser utilizados que puede personalizar mediante CSS y, posteriormente, añadir a sus páginas Web. Con Dreamweaver, puede añadir a las páginas diversos widgets de Spry, incluidas listas y tablas gestionadas mediante XML, acordeones, interfaces en fichas y elementos de formulario con validación.

Efectos de Spry

Los efectos de Spry ofrecen un método sencillo y elegante para mejorar el aspecto y el funcionamiento de un sitio Web. Puede aplicarlos prácticamente a cualquier elemento de una

página HTML. Puede añadir efectos de Spry para aumentar o reducir el tamaño de elementos; hacer que se desvanezcan o resaltarlos; modificar visualmente un elemento de una página durante un período de tiempo determinado, etc.

Dreamweaver CS3 pone a nuestra disposición su librería de Spry:
Estos controles Spry o widgets harán que tus páginas se muestren de una forma más profesional, vistosa y ordenada.

Podrás encontrarlos bajo el menú Insertar → Spry o en la pestaña Spry de la barra Insertar:

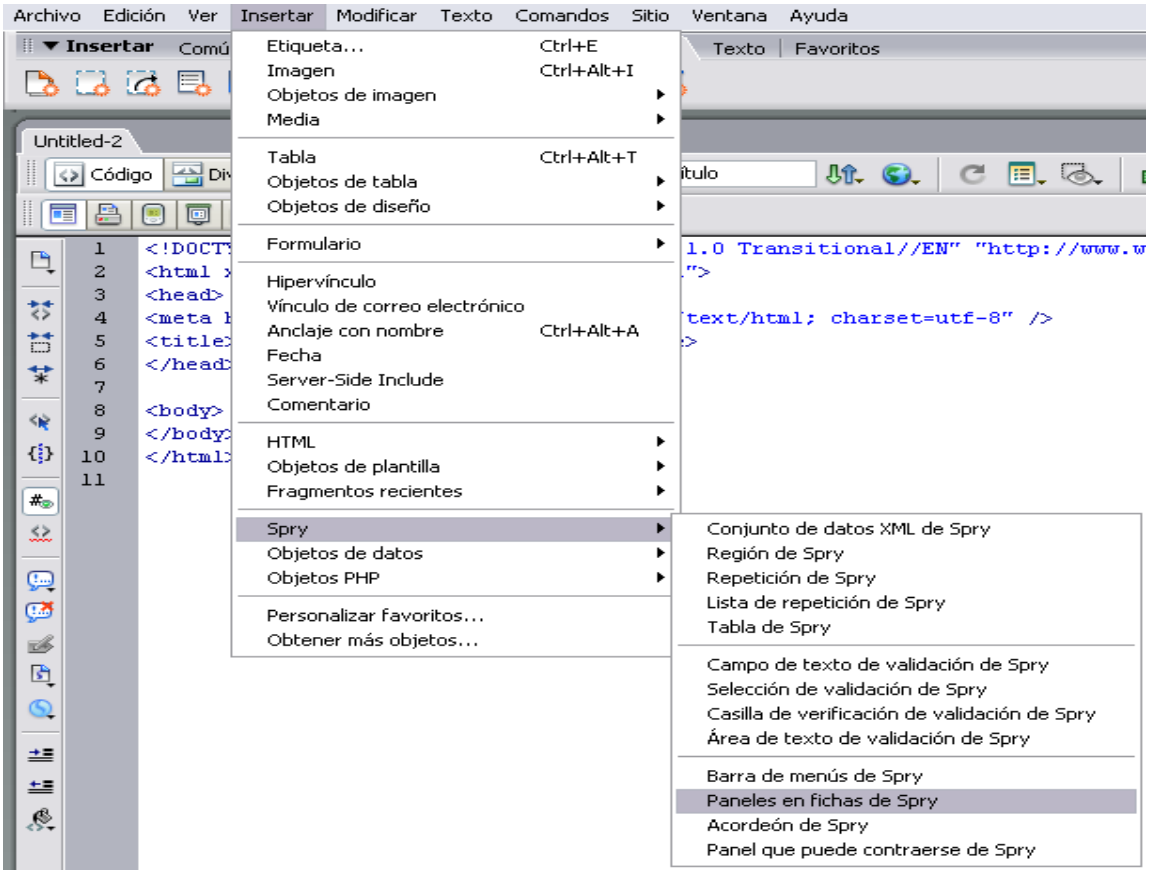


Figura 2 Insertar Spry de Paneles en fichas en Dreamweaver cs3 desde la barra de menú.

O en la barra principal en la cual tenemos los controles de Spry.

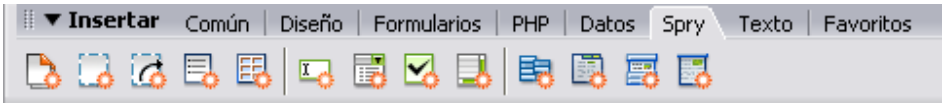


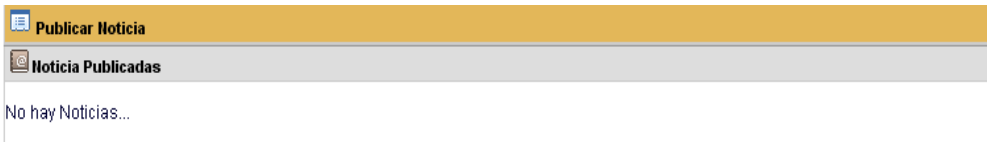
Figura 3 Insertar un Spry en de Paneles en fichas Dreamweaver cs3 desde la barra de herramientas

Estos controles son completamente configurables y además también aceptan modificaciones en su estilo utilizando CSS.

Acerca del Spry de panel que puede contraerse

Un Spry de panel que puede contraerse es un panel que puede almacenar contenido en un espacio reducido. Los usuarios pueden hacer clic en la ficha del Spry para mostrar u ocultar el contenido almacenado en el panel que puede contraerse. En el siguiente ejemplo se muestra un Spry de panel que puede contraerse, ampliado y contraído:

a) Contraído.



b) Ampliado.

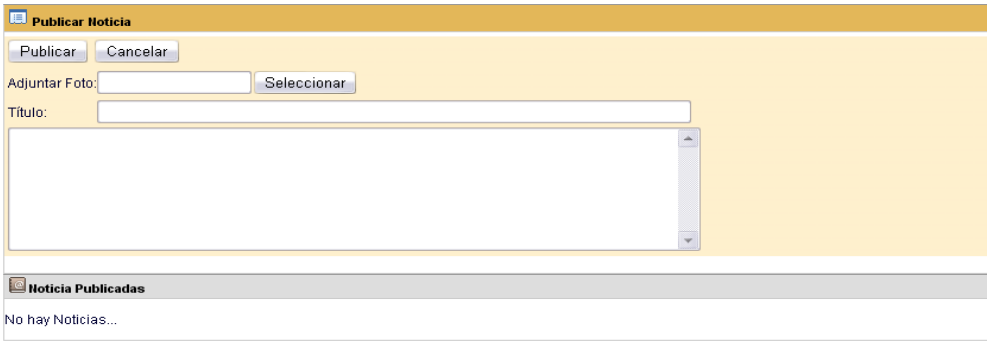


Figura 4 Ejemplo de Spry de panel que puede contraerse

El código HTML del Spry de panel que puede contraerse consta de una etiqueta div externa que contiene la etiqueta div de contenido y la etiqueta div de contenedor de ficha. El código HTML del Spry de panel que puede contraerse también incluye etiquetas script en el encabezado del documento y detrás del formato HTML del panel que puede contraerse.

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title> CONTRAERSE </title>

<script src="SpryAssets/SpryCollapsiblePanel.js" type="text/javascript"></script>
<link href="SpryAssets/SpryCollapsiblePanel.css" rel="stylesheet" type="text/css" />
</head>

<body>
<div id="CollapsiblePanel1" class="CollapsiblePanel">
  <div class="CollapsiblePanelTab" tabindex="0">Publicar Noticia</div>
  <div class="CollapsiblePanelContent">Contenido</div>
</div>
```

```
<script type="text/javascript">
<!--
var CollapsiblePanel1 = new Spry.Widget.CollapsiblePanel("CollapsiblePanel1");
//-->
</script>

</body>
</html>
```

Acerca del Spry de paneles en fichas

Un Spry de paneles en fichas es un conjunto de paneles que pueden almacenar contenido en un espacio reducido. Los visitantes pueden hacer clic en la ficha del panel para mostrar u ocultar el contenido almacenado en los paneles en fichas a los que desean acceder. Los paneles del Spry se amplían o contraen en función de las fichas que elijan los visitantes. Solamente puede haber un panel de contenido abierto en un Spry de paneles de fichas en cada momento. En el siguiente ejemplo se muestra un Spry de paneles en fichas, con el tercer panel abierto:

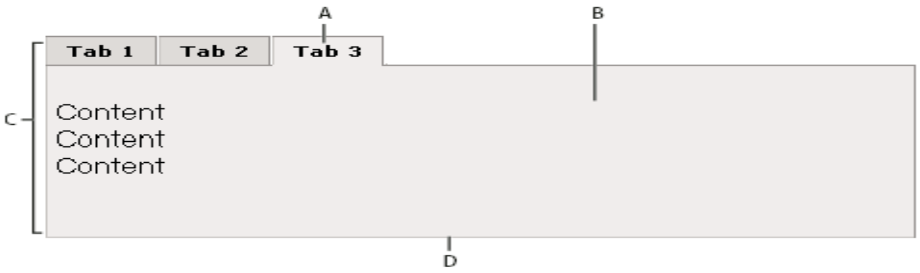


Figura 5 Estructura de un Spry de paneles en ficha

Podemos ver en la vida real el siguiente ejemplo.

Figura 6 Spry de paneles en ficha en vista de presentación

El código HTML de un Spry de paneles en fichas consta de una etiqueta div externa que contiene todos los paneles, una lista de fichas, y una etiqueta div para cada panel de contenido. El código HTML del Spry de paneles en fichas también incluye etiquetas script en el encabezado del documento y detrás del formato HTML del Spry de paneles en fichas.

Para una explicación detallada sobre el funcionamiento del Spry de paneles en fichas, este incluida una anatomía completa del código del Spry:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Documento sin t&iacute;tulo</title>
<script src="SpryAssets/SpryTabbedPanels.js" type="text/javascript"></script>
<link href="SpryAssets/SpryTabbedPanels.css" rel="stylesheet" type="text/css" />
</head>

<body>
<div id="TabbedPanels1" class="TabbedPanels">
  <ul class="TabbedPanelsTabGroup">
    <li class="TabbedPanelsTab" tabindex="0">Bandeja Entrada</li>
    <li class="TabbedPanelsTab" tabindex="0">Mensajes enviados</li>
    <li class="TabbedPanelsTab" tabindex="0">Redactar</li>
  </ul>
  <div class="TabbedPanelsContentGroup">
    <div class="TabbedPanelsContent">Contenido BE</div>
    <div class="TabbedPanelsContent">Contenido ME</div>
    <div class="TabbedPanelsContent">Redactar</div>
  </div>
</div>
<script type="text/javascript">
<!--
var TabbedPanels1 = new Spry.Widget.TabbedPanels("TabbedPanels1");
//-->
</script>
</body>
</html>
```

HOJAS DE ESTILO

Las **hojas de estilo en cascada** (*Cascading Style Sheets*, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). La idea que se encuentra detrás del desarrollo de CSS es separar la *estructura* de un documento de su *presentación*.

Se denominan "hojas de estilo en cascada" porque se pueden definir múltiples hojas y los estilos pueden aplicarse a todas las páginas (con un sistema predefinido para resolver conflictos).

El principio de las hojas de estilo consiste en la utilización de un solo documento para almacenar las características de presentación de las páginas asociadas a grupos de elementos. Esto implica nombrar un conjunto de definiciones y características de presentación de las páginas, y activar esos nombres para aplicarlos a una parte del texto.

En el siguiente ejemplo, se muestra un documento en donde **no** se aplica CSS luego mostraremos más adelante otro documento en el que se aplicará CSS:

index.html

```
<html>
<head>
<title>Bienvenidos a las hojas de estilo</title>
</head>

<body>
<br><br>
<center><p > Sobre las hojas de estilo
<br>
<div >
<br>
<!--Aquí se crea el formulario-->
<form>
Nombre: <input name="nombre" type="text"><br><br>
Clave: <input name="nombre" type="password"><br><br>
<center><button name="Enviar">Entrar</button></center>
</form><!--Cierre del formulario-->
<p> Si introduces tu nombre y tu clave accederás al sistema
<br>
</div>
<div >
<p>Esta es una prueba de Hojas de Estilo Puedes adornarla como quieras.
</div>
</center>
</body><!--Cierre del cuerpo de html-->
</html><!--Cierre del document de html-->
```

Salida:

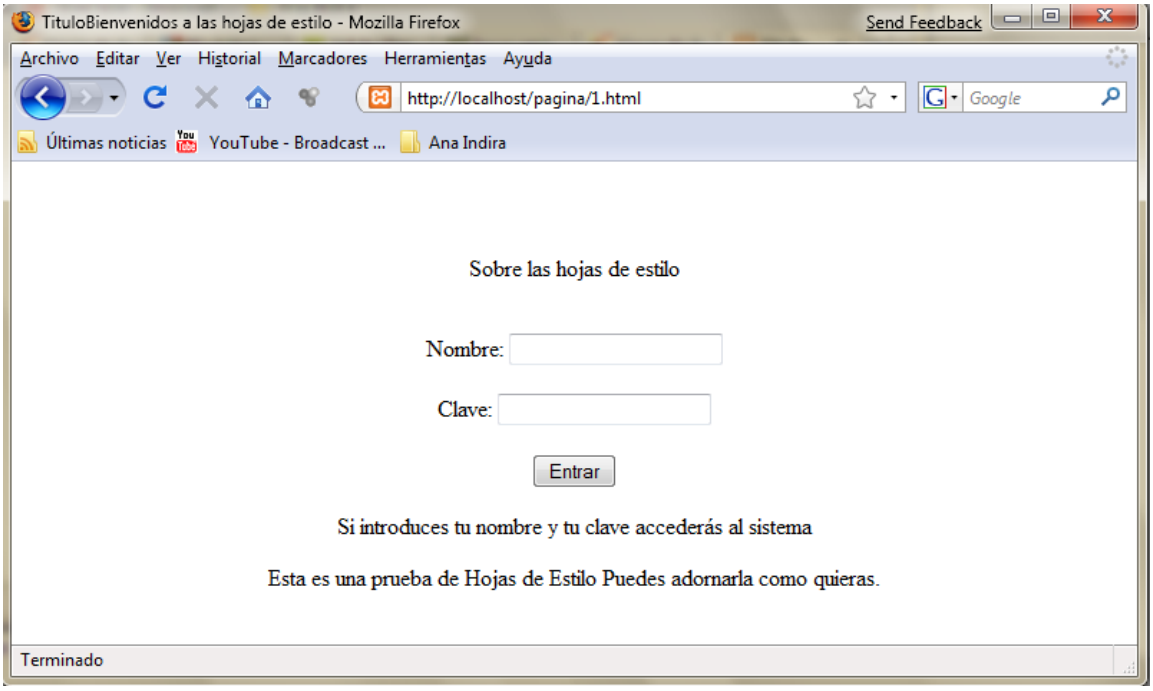


Figura 7 Página web si aplicar hojas de estilo

Ahora para mejorar la presentación del documento, mostraremos un ejemplo de la misma página pero utilizando hojas de estilo. En este caso, lo primero es crear la hoja de estilo que es la que se muestra a continuación:

```
.titulo
{
margin-top: -20px;/*este es un margen hacia arriba con disminución de 20 pixeles distancia
mínima entre un bloque  y los demás elementos*/
color:#CC0000;/*este comando se utiliza para declarar el   color en este caso rojo*/
font-size: 40px;/*este comando es para declarar el tamaño de la letra en este caso es 40 pixeles*/
font-weight:bold; /*este comando es para declarar el  ancho de la letra que sea gruesa */
font-family:"Courier New", Courier, monospace;/*este comando es para declarar el tipo de letra
*/
}

.parrafo2 {
margin-left: 20px;/*margen hacia la izquierda*/
margin-top: -20px;
color: green;/*color verde*/
font-size: 15px;/*con la letra con un tamaño de 15 pixeles*/
font-family:Arial;/*letra arial*/
```

```
}
.parte1{
width: 30%;/*Declaramos el ancho de un bloque que contendrá  un 30%*/
background:#66CCFF;/* Modifica tanto el color de fondo en este caso celeste.*/
margin-left: 10%;/*Distancia mínima de 10% entre un bloque de los elementos hacia la
izquierda*/
margin-top: 1%;/* Distancia mínima de 1% entre un bloque de los elementos hacia arriba */
font-family: Arial;/*Declaramos el tipo de letra */
font-size: small;/*  tamaño de la letra  */
}
.parte2{
width: 60%;/*Declaramos el ancho de un bloque que contendrá  un 60%*/
background:blue; /* Modificando  el color de fondo en este caso azul  contiene el color de fondo
del bloque.*/
margin-left: 10%;/*Distancia mínima de 10% entre un bloque de los elementos hacia la
izquierda*/
margin-top: 1%;/* Distancia mínima de 1% entre un bloque de los elementos hacia arriba */
padding-top: 1%;/* Distancia  hacia arriba entre el borde y el contenido del bloque con 1% */
padding-bottom: 1%;/* Distancia del botón entrar  y el contenido del bloque con 1% */
font-family: Times;
font-style: italic;/*Estilo de tipo de letra */
font-size: medium; /*Declaramos el tamaño de la letra  indicando que sea mediana*/

color: yellow;/*Declaramos el color  Amarillo */
}
```

Finalmente volvemos a crear el documento html solo que ahora, tenemos en cuenta la inclusión de los estilos:

```
/*EL siguiente documento HTML  hace  uso de Hojas de Estilo
Una hoja de estilo externa puede ser enlazada a un documento HTML mediante el elemento LINK
 */

index.html

<html>
<head>

<title>Bienvenidos a las hojas de estilo</title>
<link rel="stylesheet" href="2.css" type="text/css" />
</head>

<body>
<br><br>
<center><p class="titulo" >Sobre las hojas de estilo<!--titulo del contenido -->
<br>
<div class="parte1"><!--permite justificar el contenido que tiene parte 1-->
```

```
<br>
<!--Aquí se crea el formulario-->
<form>
Nombre: <input name="nombre" type="text"><br><br>
Clave: <input name="nombre" type="password"><br><br>
<center><button name="Enviar">Entrar</button></center>
</form><!--Cierre del formulario-->
<p> Si introduces tu nombre y tu clave accederás al sistema
<br>
</div><!--Cierre del div-->
<div class="parte2" ><!--permite justificar el contenido de este div-->
<p>Esta es una prueba de Hojas de Estilo Puedes adornarla como quieras.
</div>
</center>
</body><!--Cierre del cuerpo de html-->
</html><!--Cierre del document de html-->
```

Salida empleando hojas de estilo

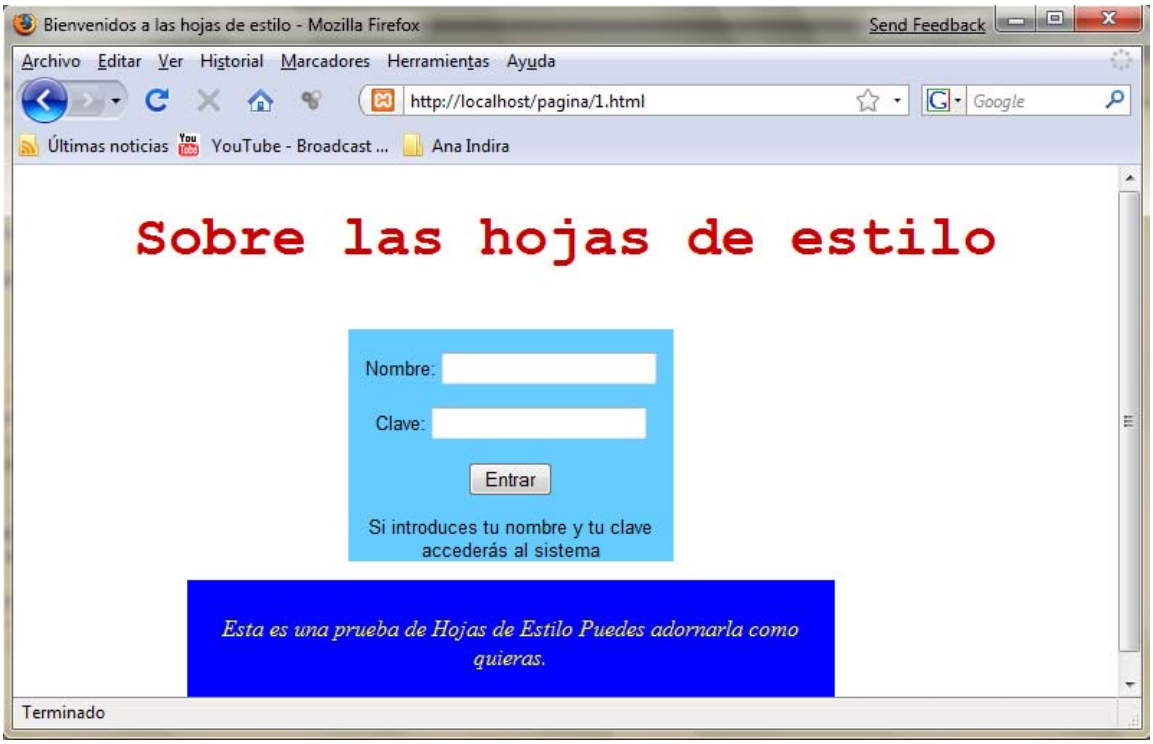


Figura 8 Página Web aplicando hojas de estilo

JAVASCRIPT

JavaScript es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C.

Al igual que Java, JavaScript es un lenguaje orientado a objetos propiamente dicho, ya que dispone de Herencia, si bien esta se realiza siguiendo el paradigma de programación basada en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad.

Javascript (en contra de lo que se podría suponer) es totalmente distinto de java. Java crea programas totalmente independientes y operativos, JavaScript es más sencillo porque lo único que permite es insertar código especial dentro de HTML de una página, su función es ampliar las posibilidades de HTML. Java script no crea programas independientes, dependen por completo del código HTML de la página.

El siguiente ejemplo es un documento de html usando javascript.

Para introducir código JavaScript en una página HTML, debe incluirse el siguiente código (normalmente, entre las etiquetas <head> y </head> de la página, aunque también puede ir en el cuerpo de la misma):

```
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
<!--
Programa JavaScript
//-->
</SCRIPT>
```

En este ejemplo se ha utilizado la variable **nombre** para guardar el nombre del usuario.

```
<HTML>
<HEAD><TITLE>Ejemplo: uso de una variable</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
var name = prompt("Introduce tu nombre:","Nombre")
//--> // comentario de una línea
</SCRIPT></HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
<!—
//Para escribir la cadena podemos usar
document.write('<IMG SRC="welcome.gif">') Como el nombre de la imagen lleva comillas dobles, el texto pasado al método write lleva comillas simples.
document.write("<H1>Hola " + name + ". ¡Bienvenido a mi página!</H1>")
//-->
```

```
</SCRIPT></BODY></HTML>
```

Salida del código JavaScript anterior:

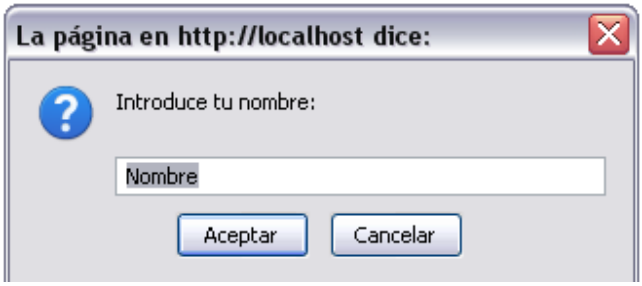


Figura 9 Ventana generada por código JavaScript

Hola Ana. ¡Bienvenida a mi página!

AJAX

AJAX, acrónimo de ***A****synchronous* ***J****avaScript* ***A****nd* ***X****ML* (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o **RIA** (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

Ejemplo básico con Ajax

Cargar contenidos en una página web a través de Ajax es sencillo. Sin embargo Ajax se limita a cargar el contenido en la posición exacta de la página que le indicamos, sin refrescar la página, es importante prestar atención al lugar que le pedimos que inserte el contenido.

El siguiente ejemplo, muestra una página Web genérica en la que se carga contenido a través de Ajax para mostrar diferente información en función del hiperenlace sobre el que se haga clic.

Inicialmente se nos muestra el formulario ingresar donde pueden dar clic para ***Registrarse***

Figura 10 Formulario donde se aplica AJAX

```
<script type="text/javascript">
        function PeticionAJAX()
        {
                //Crear una variable de Bool para comprobar si existe Internet Explorer.
                    var xmlhttp = false;
                //Comprobar si se está usando IE.
                try {
                        //Si la versión de javascript es superior a la 5.
                        xmlhttp = new ActiveXObject("Msxml2.XMLHTTP");
                        alert("You are using Microsoft Internet Explorer.");
                } catch (e) {
                        //Si no, utilizar el tradicional objeto ActiveX.
                        try {
                                //Si se está usando IE .
                                xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
                        } catch (E) {
                                //En caso contrario no debe estar usándose IE.
                                xmlhttp = false;
                        }
                }//cierre de catch(e)

                //Si no estamos usando IE, crear una versión Javascript del objeto.
                if (!xmlhttp && typeof XMLHttpRequest != 'undefined') {
                        xmlhttp = new XMLHttpRequest();
                }
                return xmlhttp;
        }//cierre de la función

//La función create form que es la que llamará al archivo registrar.php  dándole las posiciones específicas.
        function createform(e)
        {
                var     xmlhttp = new PeticionAJAX();
                theObject = document.getElementById("contenido1");
                theObject.style.visibility = "visible";
                var posx = 0;
                var posy = 0;
                posx = e.clientX + document.body.scrollLeft;
                posy = e.clientY + document.body.scrollTop;
                theObject.style.left=10+ "px";
                theObject.style.top=10+ "px";

                //La ubicación donde se va a cargar la pagina.
```

```
                var objID = "contenido1";
                var serverPage = "registrar.php";
                var obj = document.getElementById(objID);
                xmlhttp.open("GET", serverPage);
                xmlhttp.onreadystatechange = function() {
                        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                                obj.innerHTML = xmlhttp.responseText;
                        }
                }
                xmlhttp.send(null);
        }
</script>
<form name="entrar" class="contenido">
<table>
<tr><td>Login: <input type="text" size="10" name="login" onfocus="cancelar('res')"/></td></tr>
<tr><td>Clave: <input type="password" size="10" name="pass" onfocus="cancelar('res')"
onkeypress="pulsar(event,entrar)"/></td></tr>

<tr><td><input type="button" onclick="ValidarSesion(entrar)" value="Entrar" /> </td></tr>
//debemos de agregar la llamada al formulario cada vez que se haga clic sobre el hipervínculo
//Registrar
<tr><td onclick="createform(event)" ><a title="Para registrarte">Registrarse</a></td></tr>
</table>
</form>

El event(createform(event)), permita obtener la posición en la cual se hizo clic y de esa manera
hacer aparecer el formulario en la posición actual.

<!--En este DIV se cargará mediante AJAX el elemento-->
        <div id="contenido1">
      </div>  <!--Fin del DIV donde irá el contenido AJAX -->
```

Con esto tendremos como resultado lo siguiente:

Figura 11 Resultado después de aplicar AJAX

PHP

PHP es un lenguaje de programación que se ejecuta en servidores *web* y permite crear páginas HTML de forma dinámica. Las siglas PHP provienen de *Personal Home Page*, que podemos traducir como Procesador personal de páginas *web* o de Hipertexto.

PHP 5

Con las primeras 2 versiones de PHP, PHP 3 y PHP 4, se había conseguido una plataforma potente y estable para la programación de páginas del lado del servidor. Estas versiones han servido de mucha ayuda para la comunidad de desarrolladores, haciendo posible que PHP sea el lenguaje más utilizado en la web para la realización de páginas avanzadas.

Sin embargo, todavía existían puntos negros en el desarrollo PHP que se han tratado de solucionar con la versión 5, aspectos que se echaron en falta en la versión 4, casi desde el día de su lanzamiento. Nos referimos principalmente a la programación orientada a objetos (POO) que, a pesar de que estaba soportada a partir de PHP3, sólo implementaba una parte muy pequeña de las características de este tipo de programación.

El principal objetivo de PHP5 ha sido mejorar los mecanismos de POO para solucionar las carencias de las anteriores versiones. Un paso necesario para conseguir que PHP sea un lenguaje apto para todo tipo de aplicaciones y entornos, incluso los más exigentes.

Presentamos la implementación de algunas de las características posibles en PHP5 respecto a la POO.

Visibilidad

La visibilidad de una propiedad o método puede ser definida al anteponerle a la declaración con las palabras reservadas: public, protected o private. Los elementos declarados como Public

pueden ser accesados en todos lados. Los atributos declarados como protected limitan el ámbito a la clase que define el elemento (es decir, sólo podemos acceder a las mismas desde dentro de la clase mediante otras funciones) y a las clases heredadas (hijas). Los Private limitan la visibilidad solo a la clase que lo definió (es decir, se comporta igual que protected pero las clases hijas no heredan estas variables y propiedades).

En el caso de las variables siempre es obligatorio definirlas como **public**, **protected** o **private**. Con las funciones puede omitirse esta palabra, en cuyo caso PHP supondrá que está definida como **public**.

Con los siguientes ejemplos mostraremos la aplicación de esta característica de PHP5:

Ejemplo 1:

```
<?
// Definimos la clase vehículo

class vehículo
 {
  public $variable_publica = 'publica';
  protected $variable_protegida = 'protegida';
  private $variable_privada = 'privada';

          function imprimir_variables()
          {
          echo $this->variable_publica."<br>";
          echo $this->variable_protegida."<br>";
          echo $this->variable_privada."<br>";
          }
 }

$objeto = new vehículo();
$objeto->imprimir_variables();
?>
```

El código anterior imprimiría lo siguiente:

```
publica
protegida
privada
```

No hubo ningún error debido a que las variables fueron accedidas desde una función propia de la clase.

Ejemplo 2:

```
<?
// Definimos la clase vehículo

class vehículo
 {
  public $variable_publica = 'publica';
  protected $variable_protegida = 'protegida';
  private $variable_privada = 'privada';

            function imprimir_variables()
            {
            echo $this->variable_publica."<br>";
            echo $this->variable_protegida."<br>";
            echo $this->variable_privada."<br>";
            }
 }

$objeto = new vehículo();

 echo $objeto->variable_publica;
 echo $objeto->variable_protegida;
 echo $objeto->variable_privada;
?>
```

El código anterior imprimiría lo siguiente:

```
 publica
Fatal error: Cannot access protected property vehículo::$variable_protegida
```

Observamos que se muestra la variable pública puesto que ésta puede ser accedida desde cualquier ámbito, pero cuando se intenta acceder a la variable protegida PHP muestra un error porque no se puede mostrar una variable protegida ya que sólo podemos acceder a las mismas dentro de la clase mediante otras funciones, de igual forma tendríamos el mismo error al querer acceder a la variable privada.

El siguiente ejemplo nos permitirá evaluar el comportamiento de la visibilidad de una propiedad cuando éstas son accedidas de una clase que es heredada en este caso de la clase vehículo definida en ejemplo 1.

Ejemplo 3:

```
class vehiculo_descapotable extends vehículo
 {
// Podemos redeclarar la variable protegida pero no la privada
            protected $variable_protegida = 'protegida 2';
```

```
        function imprimir_variables()
        {
        echo $this->variable_publica."<br>";
        echo $this->variable_protegida."<br>";
        echo $this->variable_privada."<br>";
        }
}

$objeto2 = new vehiculo_descapotable();
echo $objeto2->variable_publica."<br>";
$objeto2->imprimir_variables();
```

El código anterior imprimiría lo siguiente:

```
publica
publica
protegida 2
```

Observamos que se muestra la variable pública heredada de la clase padre vehículo ya que siempre ésta podrá ser accedida desde cualquier ámbito, en cambio cuando se hace la llamada al método definido en la clase hija solamente se imprime la variable pública y la protegida la cual fue redefinida en la clase hija, pero la variable privada no se imprime porque esta no es heredada.

Clases abstractas

Una clase se define como abstracta cuando representa una entidad que no debería ser instanciada.

En los ejemplos utilizados hasta ahora se ha definido una clase vehículo, sin embargo, en el mundo real (dominio del problema) no se encuentran objetos de la clase vehículo, sino que existen coches, motos, camiones... Al final todos son vehículos, pero todos son de un tipo más concreto (vehículo es un término abstracto que agrupa a una serie de sistemas que se utilizan para el transporte).

Por lo tanto sería bastante lógico declarar la clase vehículo como abstracta, forzando así la necesidad de construir clases derivadas que representen elementos más concretos.

Esto se ve reflejado en el siguiente ejemplo:

Ejemplo 4:

```
abstract class vehículo
{
  public $potencia;
```

```
  public $peso;
}

// ERROR: no se puede instanciar una clase abstracta
// $objVehiculo = new Vehiculo();


// una Moto es un Vehículo
class Moto extends Vehiculo
{
  public $anguloMaxInclinacionEnCurva;
}

//// creamos un objeto Moto
$obj_moto = new Moto();
```

Las clases abstractas se suelen utilizar como base para crear una jerarquía en la que todas las clases comparten una parte de la interfaz. Dentro de una clase abstracta se pueden definir métodos abstractos. Los métodos abstractos no tienen implementación (ni funcionalidad), simplemente definen una parte de la interfaz que deben implementar las clases derivadas (la clase base obliga a que se definan esos métodos en la clase derivada).

Por ejemplo, para la clase vehículo se podría definir un método abstracto aceleracionAproximada() de forma que tenga que ser implementado para cada tipo de vehículo (la aceleración depende de la masa (peso), de la potencia y de otros factores que dependen a su vez del tipo de vehículo), quedando la clase vehículo de la siguiente manera:

```
abstract class vehículo
{
  public $potencia;
  public $peso;

  function __construct($potencia,$peso)
  {
    $this->potencia = $potencia;
    $this->peso = $peso;

    return true;

  }

  function relacionPesoPotencia()
  {
    if ($this->potencia>0)
    {
      return ($this->peso/$this->potencia);
    }
```

```
   return -1;
  }

//método que será redefinido en cada una de las clases derivadas.
  abstract function aceleracionAproximada();

}
```

Las clases derivadas de la clase vehículo definirán el método aceleracionAproximada() a como en el siguiente código:

```
// una Moto es un Vehículo
class Moto extends Vehiculo
{

  function __construct($potencia,$peso)
  {
    $this->potencia = $potencia;
    $this->peso = $peso;

    return true;
  }

 // devuelve el tiempo en alcanzar los 100 Km/h
 // partiendo de cero

  function aceleracionAproximada()
  {
     $coeficienteTransmision = 3.0;

     $t = $this->peso * 771.73 / (2.0 * $this->potencia * 735);
     $t = $t * $coeficienteTransmision;
     return $t;
  }
}


//// un Coche es un Vehículo
class Coche extends Vehiculo
{
```

```
 function __construct($potencia,$peso)
 {
  $this->potencia = $potencia;
  $this->peso = $peso;

  return true;
 }

// devuelve el tiempo en alcanzar los 100 Km/h
// partiendo de cero

 function aceleracionAproximada()
 {
   $coeficienteTransmision = 2.2;

   if ($this->potencia==0)
   {
    return -1;
   }

   $t = $this->peso * 771.73 / (2 * $this->potencia * 735);
   $t = $t * $coeficienteTransmision;

   return $t;
 }
}
```

```
// un coche (125CV, 1300Kg)
$coche = new Coche (125, 1300);
echo "coche (0-100): ".$coche->aceleracionAproximada();
echo "<br>";

// una moto (60CV, 250Kg)
$moto = new Moto (60, 250);
echo "moto (0-100): ".$moto->aceleracionAproximada();
echo "<br>";
```

Así el código anterior daría el siguiente resultado:

```
coche (0-100): 12.0116887075
moto (0-100): 6.56232993197
```

Finalmente, recordamos que las posibilidades de PHP 5 respecto a la POO son muy amplias:

- Auto carga de las Objetos.
- Constantes en Objetos.
- Interfaces de Objetos.
- Sobrecarga de variables y funciones.
- Interacción de Objetos.
- Patrones.
- Métodos mágicos.
- La palabra reservada 'Final'.
- Clonado de Objetos.
- Comparación de Objetos.
- Refección.

MySQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario, es la base de datos open source más popular y su continuo desarrollo y su creciente popularidad está haciendo de MySQL un competidor cada vez más directo de gigantes en la materia de las bases de datos como Oracle. MySQL es un sistema de administración de bases de datos (*Database Management System, DBMS*) para bases de datos relacionales.

Existen muchos tipos de bases de datos, desde un simple archivo hasta sistemas relacionales orientados a objetos. MySQL, como bases de datos relacional, utiliza múltiples tablas para almacenar y organizar la información.

MySQL fue escrito en C y C++ y destaca por su gran adaptación a diferentes entornos de desarrollo, permitiendo su interactuación con los lenguajes de programación más utilizados como PHP, Perl y Java y su integración en distintos sistemas operativos.

También es muy destacable, la condición de open source de MySQL, que hace que su utilización sea gratuita e incluso se pueda modificar con total libertad, pudiendo descargar su código fuente. Esto ha favorecido muy positivamente en su desarrollo y continuas actualizaciones, para hacer de MySQL una de las herramientas más utilizadas por los programadores orientados a Internet.

PROGRAMACIÓN EN TRES CAPAS

La programación por capas es un estilo de programación en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario.
La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado.

En el diseño de sistemas informáticos actual se suele usar las arquitecturas multinivel o Programación por capas. En dichas arquitecturas a cada nivel se le confía una misión simple, lo que

permite el diseño de arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).

CAPAS Y NIVELES

1.- **Capa de presentación**: es la que ve el usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio. También es conocida como interfaz grafica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario.

2.- **Capa de negocio**: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.

3.- **La capa DAL (Data Access Layer – Capa de Acceso a Datos):** es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

CAPA DE ACCESO A DATOS CON PHP

Definición de la Base de Datos

Lo primero que deberemos de hacer es crear en mysql, una nuestra base de datos llamada **capas** que contenga una tabla usuarios con los campos email, clave y nombre. El script para la creación de dicha tabla se presenta a continuación:

```
CREATE TABLE `usuarios` (
  `email` varchar(20) NOT NULL,
  `clave` varchar(20) NOT NULL,
  `nombre` varchar(500) NOT NULL,
  PRIMARY KEY  (`email`)
);
```

Definición de la clase usuarios

Normalmente, deberemos de crear en PHP una clase por cada una de las tablas que tengamos en nuestra base de datos. Estas clases contendrán como datos miembros un equivalente a los campos que tiene su tabla asociada y los métodos get y set, para cada uno de los datos En este ejemplo, crearemos la clase usuarios dentro de un archivo llamado usuarios.php, en la cual aplicaremos los conceptos antes expuestos:

```
<?php

/*La clase usuario, servirá para guardar los datos de un usuario*/
class usuario
{
	/*Definimos un dato para cada una de las variables que se tienen en la tabla */
	private $email;
	private $clave;
	private $nombre;

	/*Para cada una de las variables realizamos las funciones get y set. Las
	funciones get son las que nos retornaran el valor de la variable y las
	funciones set son las que establecen un valor a las variables*/

	public function getEmail(){
		/*Siempre que queramos acceder a un dato miembro se debe de hacer
		con $this->dato_miembro*/
		return $this->email;
	}

	public function setEmail($em)   {
		$this->email = $em;
	}

	public function getClave(){
		return $this->clave;
	}

	public function setClave($cl){
		$this->clave = $cl;
	}

	public function getNombre(){
		return $this->nombre;
	}

	public function setNombre($nom){
		$this->nombre = $nom;
	}
}

?>
```

Como se dijo anteriormente, se debería de hacer lo mismo para cada una de las tablas que tengamos dentro de la base de datos.

Creación de la clase CCconexión y clase CAdapter

Como haremos un uso intensivo de las conexiones a las bases de datos, en este ejemplo definiremos una clase llamada CConexión y de otra clase llamada CAdapter la cual guardaremos dentro de un fichero llamado CConexión.php para que cada vez que tengamos que realizar una conexión a la base de datos, simplemente tengamos que crearnos una base de datos de este tipo. El código para esta clase sería el siguiente:

```
<?php

/*Esta clase será utilizada para conectarnos y desconectarnos de la base de datos.
Para ello ocupamos la clase CConexion.*/

class CConexion
{
        private $Host;
        private $Port;
        private $Password;
        private $Usuario;
        private $BaseDatos;
        private $Conexion;

        //Establece los parámetros para realizar las conexiones

function CConexion($usr='root', $pw='mysql', $db='capas', $ho='localhost', $po=3306)
        {
                $this->Host = $ho;
                $this->Port = $po;
                $this->Password = $pw;
                $this->Usuario = $usr;
                $this->BaseDatos = $db;
        }

        //Abre una conexión con la base de datos devuelve true si no hubo problemas o
        //false si ocurrió algún error.
        public function Open()
        {
                $this->Conexion = mysql_connect($this->Host,$this->Usuario,$this-
>Password);
                mysql_select_db($this->BaseDatos);
           if(mysql_error($this->Conexion))
                {
                        return false;
                }
```

```
                return true;
        }

        //Obtiene la conexión actual
        public function getConexion()
        {
                return $this->Conexion;
        }

        //Función que se encarga de cerrar la conexión
        public function Close()
        {
                return mysql_close($this->Conexion);
        }
}
//Esta clase nos permitirá interactuar con la base de datos
class CDataAdapter
{
        //Conexión a usar
        private $con;
        //Constructor del objeto
        function CDataAdapter()
        {
                $this->con = new CConexion();
                $this->con->Open();
        }

        //Esta función se encarga de realizar cualquier consulta select recibe como parámetro
        //una conexión y si no, ella crea una con los valores por defecto
        public function ConsultaSelect($Consulta, $conex="")
        {
                //Se obtiene el resultado de la Consulta
                $result = mysql_query($Consulta, $this->con->getConexion());

                //Contabilizamos el número de resultados
                $num_filas = mysql_num_rows($result);

                //Obtenemos un array con el resultado de la consulta anteriormente
generada
                if($num_filas > 0)
                        for($i=0;$i<$num_filas;$i++)
                  {
                                $DataSet[$i] = array();
                                $fila = mysql_fetch_assoc($result);
                                $DataSet[$i] = $fila;
                        }
                return $DataSet;
```

```
        }

        //Esta consulta se encarga de insertar, actualizar o borrar cualquier elemento de
        //la base de datos y retorna el numero de filas afectadas
        public function ConsultaModificacion($Consulta, $conex="")
        {
                if($conex==""){
                        $result = mysql_query($Consulta, $this->con->getConexion());
                        return mysql_affected_rows($this->con->getConexion());
                }
                $result = mysql_query($Consulta,$conex->getConexion());
                //Contabilizamos las filas afectadas y devolvemos el resultado
                return mysql_affected_rows($conex->getConexion());
        }

        public function CerrrarConexion()
        {
                //Cerramos la conexión
                $this->con->Close();
        }
}

?>
```

La clase CConexion podría incluir mejoras quitando los parámetros que se pasan por omisión al constructor de la clase, para la gestión de privilegios de los usuarios y así introducirlo en el momento de crearnos un objeto de esta clase. Como se puede observar, la clase también se encarga de seleccionar la base de datos "capas".

Definición de la Capa de Acceso a Datos

Con todo lo anterior definido, estamos listos para crear nuestra capa de acceso a datos. La capa de acceso a datos contiene una clase por cada una de las tablas que se tengan en la base de datos. Cada una de estas clases incluye todas las consultas que necesitamos hacer sobre la tabla que dicha tabla represente. Generalmente estas clases se nombran con el nombre de la tabla que representan más la palabra DAL.

En nuestro caso, para la tabla usuario, hemos definido la clase usuarioDAL, guardado en el archivo usuarioDAL.php. El código de esta clase se muestra a continuación:

```
<?php
/*Incluimos el archivo CConexion.php para poder crear objetos del tipo CAdapter*/
include_once("CConexion.php");
/*Incluimos el archivo usuario.php para poder crear objetos del tipo usuario*/
include("..\BAC\Usuario\usuario.php");

class usuarioDAL
```

```
{
        /*Esta función asigna una fila de la tabla usuarios a una clase llamada usuarios*/
        /*Es una función que puede reutilizarse*/
        public function CopiarDatosUsuario($rec)
        {
          try{
                $temp = new usuario();
                $fila = $rec;

                $temp->setEmail($fila['email']);
                $temp->setNombre($fila['nombre']);
                $temp->setClave($fila['clave']);
                return $temp;
    }
         catch(exception $e)
         {
                        return false;
         }
        }

        /*Esta función obtiene a un usuario por su Email*/
        public function Select_By_Email($email)
        {
                $Conector = new CDataAdapter();
                $result=$Conector->ConsultaSelect("select * from usuario where email='" .
$email . "'");

                /*Copiamos en la variable de tipo usuario el resultado de la consulta*/
                $num_filas = count($result);

                if($num_filas>0)
                        $usr = $this->CopiarDatosUsuario($result[0]);
                return $usr;
        }

        /*Esta función se encarga de obtener a todos los usuarios*/
        public function Select_All_Usr()
        {
                $Conector = new CDataAdapter();
                $result=$Conector->ConsultaSelect("select * from usuario order by nombre");

                /*Copiamos en la variable de tipo usuario el resultado de la consulta*/
          $num_filas = count($result);

          if($num_filas > 0)
                {
                        for($i=0;$i<$num_filas;$i++)
```

```
                    {
                        $usr[$i] = new usuario();
                        $usr[$i] = $this->CopiarDatosUsuario($result[$i]);
                    }
                }
                return $usr;
        }


        /*Esta función nos servirá para insertar un dato en la base de datos*/
        public function Insert_Usr($em,$nom,$apell,$telf,$fech,$sex,$nic,$cl,$fot)
        {
          $Conector = new CDataAdapter();
          $result=$Conector->ConsultaModificacion("insert into usuario values('" . $em . "','"
. $nom . "', '" . $cl ."')");

                if($result != 0)
                        return true;
                else
                        return false;
        }

        /*Elimina a un usuario identificado por su email*/
        public function Delete_Usr_ByEmail($em)
        {
                $Conector = new CDataAdapter();
         $result=$Conector->ConsultaModificacion("delete from usuario where email='" .
$em .");

                  if($result != 0)
                        return true;
                else
                        return false;

        }

        /*Actualiza la clave de un usuarios en base a su email*/
        public function Update_Clave_Usr($em,$cl)
        {
                $Conector = new CDataAdapter();
         $result=$Conector->ConsultaModificacion("update usuario set clave='" . $cl . "'
where email='" . $em .");


                if($result != 0)
                        return true;
                else
```

```
                        return false;

        }

        /*Actualiza el nombre del usuario en base a su email*/
        public function Update_Nombre_Usr($em,$nom)
        {
           try{
                $Conector = new CDataAdapter();
         $result=$Conector->ConsultaModificacion("update usuario set nombre='" . $nom . "'
where email='" . $em ."');
                        if($result != 0)
                        return true;
                else
                        return false;


        }


}

?>
```

Cabe recordar que de igual manera como se definió la clase usurioDAL para la tabla usuario, se debería de definir una clase para cada una de las tablas. Las funciones que se incluyan dentro de dicha clase, dependerá de las consulta que se quieran hacer.

Prueba de la Capa de Acceso a Datos

Para probar la capa de acceso a datos que hemos diseñado, utilizaremos el siguiente fichero index.php. Obsérvese como ahora es más sencillo acceder a la base de datos ya que nos olvidamos de todo lo que tiene que ver con la base de datos y nos limitamos a realizar llamadas a las funciones correspondientes. El código para esta parte será:

```
<html>
<head>
<title>Prueba</title>
</head>
<body>

<?php
        /*En este archivo pondremos a prueba nuestra capa de acceso a datos para la
        tabla usuario*/
```

```
include 'usuarioDAL.php';

 /*Creamos una variable del tipo usuarioDAL para poder tener acceso a todas
 las funciones de esta clase*/

 $usuarios = new usuarioDAL();

 echo "Insertamos 3 datos a la base de datos<br>";

 $usuarios->insert_usr("denis@yahoo.es","denis123","Denis Espinoza");
 $usuarios->insert_usr("juan@yahoo.es","juan123","Juan Perez");
 $usuarios->insert_usr("jose@yahoo.es","jose123","Jose Martínez");

 echo "Seleccionamos y mostramos todos los datos de la base de datos<br>";

 $usrs = $usuarios->select_All_Usr();
 foreach($usrs as $value)
 {
          echo $value->getEmail() . "<br>";
          echo $value->getClave() . "<br>";
          echo $value->getNombre() . "<br><br>";
 }

 echo "Actualizamos la clave de Juan<br>";

 $usuarios->update_clave_usr("juan@yahoo.es","juancito");

 echo "Mostramos los datos actualizados de Juan<br>";

 $us = $usuarios->select_By_Email("juan@yahoo.es");
 echo $us->getEmail() . "<br>";
 echo $us->getClave() . "<br>";
 echo $us->getNombre() . "<br><br>";

 echo "Borramos a José<br>";

 $usuarios->delete_usr_byEmail("jose@yahoo.es");

 echo "Seleccionamos y mostramos todos los datos de la base de datos
       después de borrar a José<br>";

 $usrs = $usuarios->select_All_Usr();
 if(count($usrs)>0)
          foreach($usrs as $value)
          {
                   echo $value->getEmail() . "<br>";
```

```
                        echo $value->getClave() . "<br>";
                        echo $value->getNombre() . "<br><br>";
                }
        else
                echo "Base de datos vacía<br>";

        $usuarios->delete_All_usr();

        echo "Seleccionamos y mostramos todos los datos de la base de datos
             después de borrar todo<br>";

        $usrs = $usuarios->select_All_Usr();

        if(count($usrs)>0)
                foreach($usrs as $value)
                {
                        echo $value->getEmail() . "<br>";
                        echo $value->getClave() . "<br>";
                        echo $value->getNombre() . "<br><br>";
                }
        else
                echo "Base de datos vacía<br>";
?>
</body>
</html>
```

Con la definición de esta primera capa, hemos sido capaces de abstraernos de los accesos a la base de datos y dedicarnos a trabajar utilizando las clases que hemos definido y dejar que sea la capa de acceso a datos la que se encargue de las conexiones con la base de datos.

CAPA DE NEGOCIOS

Aquí guardamos todas las funciones de las entidades que definimos en el sistema, y aquellas que hablan directamente con la capa de datos.

En nuestro caso, para la tabla usuario, hemos definido la clase usuarioBLL, guardado en el archivo usuarioBLL.php. El código de esta clase se muestra a continuación:

```
<?php

include_once('../DAL/Usuario/usuarioDAL.php');

class usuarioBLL
{
        /*Esta función obtiene a un usuario por su Email*/
```

```
        public function IngresarPortal($email,$clave)
        {
                $usr = new usuarioDAL();
                $usuario = $usr->Select_By_Email($email);

                if($usuario->getClave()==$clave)
                        return 1;
                else
                        return 0;
        }

       public function RegistrarsePortal($email,$nomb,$clave)
        {

                 $usr= new usuarioDAL();
                 $usuario= $usr->select_By_Email($email);

                 if($usuario->getEmail()==$email)
                 {
                  return 0;
                 }

                 else
                 $usr->insert_usr($email,$nomb,$pass,' ');
                 $fich= new ficheroDAL();
                 $fichero=$fich->crear_carpeta($email,0777);
                 return 1;
        }

    public function MostrarDatosDeUno($em)
        {
                $usr = new usuarioDAL();
                $usuario = $usr-> select_By_Email($em);
                return $usuario;
}

   public  function BuscarUsuario($nomb)
        {

          $usr = new usuarioDAL();
          $usuarios= $usr->Select_Nombre_Usr($nomb);
          return $usuarios;

        }
public function EliminarUsuario($em)
        {
         $usr = new usuarioDAL();
```

```
            $usr->Delete_Usr_ByEmail($em);
}


 public function traer_usuarios()
 {
            $usrs= new usuarioDAL();
            $usuarios= $usrs->Select_All_Usr();
            return $usuarios;
   }
}

?>
```

Con la definición de esta segunda capa, hemos sido capaces de validar las reglas de negocio…en este caso se ingresa el nombre de usuario la base de datos si existe o verificar que el cliente no existe para poder agregarlo...entonces tendremos un método llamado function RegistrarsePortal($email,$nomb,$clave) que hará las verificaciones necesarias y llamará a la capa de datos para que lo agregue. Esta capa se encarga de la intercomunicación entre la capa de datos, con la capa de presentación.

CAPA DE PRESENTACIÓN

Esta es la capa en donde se presentan los datos al usuario, en nuestro ejemplo mostraremos como agregamos un usuario al portal para ello primero tiene que registrarse. Incluimos el fichero registrar.php que es donde el usuario introduce sus datos.

```
 <form name="Registrar">
 <table width="377" border="0" class="encabezado">
  <tr>
   <td height="26" align="center">Registrarse</td>
   <!-- Esta es el área del encabezado -->
  </tr>
 </table>
 <table width="377" border="0" class="contenidoregistro">

    <tr>
     <td  align="left"><br>Email:</td>
    <td   align="left"><br><input type="text" name="email" size="30"></td>
  </tr>
  <tr>
    <td  align="left">Nombre:</td>
    <td  align="left"><input type="text" name="nomb" size="30" ></td>
  </tr>
  <tr>
    <td align="left">Apellidos:</td>
```

```
    <td align="left"><input type="text" name="apell" size="30" ></td>
   </tr>

   <tr>
    <td align="left">Sexo:</td>
    <td align="left">
    <select id="gen" name="genero" class="barracentral" onchange="return
ShowAgeLockingMessage(this.id);">
    <option value="m">Hombre</option>
    <option value="f">Mujer</option>
    </td>
   </tr>
   <tr>
    <td align="left">Contraseña:</td>
    <td align="left"><input type="password" name="pass" size="30" ></td>
   </tr>
   <tr>
    <td align="left">Repetir Contraseña:</td>
    <td align="left"><input type="password" name="pass1" size="30" ></td>
   </tr>
     <tr>
      <td align="center"></td>
      <td align="left"><input type="button" onclick="ValidarRegistro(Registrar)"
value="Registrarse" />  <input type="button" value="Cancelar"
onClick="limpiapone('contenido1.html','contenido1')" /></td>
  </tr>
   <tr>
    <td align="left" height="25"></td>

   </tr>
 </table>
 </form>
```

La función que se encarga de validar el formulario en donde el usuario introdujo sus datos y de llamar al fichero que completará con el registro del usuario es la siguiente:

```
function ValidarRegistro(form)
   {
    var errores="";
   with(form)
   {
          if (trim(email.value)=="") //Función para verificar que no hayan espacios en blancos al
principio y al final de una cadena
          {
             errores+=" • Debe ingresar su correo.\n ";
          }
```

```
        else
        {
           var s = email.value;
          var filter=/^[A-Za-z][A-Za-z0-9_]*@[A-Za-z0-9_]+\.[A-Za-z0-9_.]+[A-za-z]$/;
          if (!filter.test(s))
          errores+="• Debe ingresar un correo valido.\n ";
        }
        if (trim(nomb.value)=="")
        {
           errores+="• Debe ingresar un Nombre.\n ";
        }
        if (trim(apell.value)=="")
        {
            errores+="• Debe ingresar sus Apellidos.\n ";
         }
        if(trim(pass.value)=="")
        {
                errores+="• Debe ingresar su Contraseña.\n ";
        }
         if(trim(pass1.value)=="")
        {
                errores+="• Debe repetir su Contraseña de nuevo.\n ";
        }
        if(pass.value != pass1.value)
        {
                errores+="• La contraseña debe ser igual, los campos no coinciden .\n ";
        }
        var fecha =  Dias.value + "-" + Mes.value + "-" +Anyos.value;
         if (Validar(fecha)==false)
        {
           errores+="• Fecha incorrecta .\n ";
        }
        else
        {
            fecha =  Anyos.value + "-" +  Mes.value + "-" + Dias.value;
        }
         if (errores)
          {
             alert("Por favor, revise los siguientes errores:\n"+errores);

              return false;
           }
        else
        {
        var str="email=" + email.value + "&nomb=" + nomb.value + "&apell=" + apell.value +
"&pass=" + pass.value + "&fecha=" + fecha + "&genero=" + genero.value;
        xmlhttp = new PeticionAJAX();
```

```
          xmlhttp.open("GET","RealizaRegistro.php?" + str);
          xmlhttp.onreadystatechange = function()
          {
          if (xmlhttp.readyState == 4 && xmlhttp.status == 200)
          {
          var texto = xmlhttp.responseText;
          var res = texto==1;
          if(res)
          {
            var obj = document.getElementById('res');
             obj.innerHTML = "Ha sido Registrado";
             limpiapone('contenido1.html','contenido1');
          }
          else
          {
              var obj = document.getElementById('res');
             obj.innerHTML = "Usuario Invalido o ya Existe";
          }
          }
         }
          xmlhttp.send(null);
        }
    }
}
```

El fichero que se hará cargo de realizar el registro es RealizaRegistro.php, su código es el siguiente:

```
<?php
   include_once('../BLL/usuarioBLL.php');
   $registro = new usuarioBLL();
   $nombre= strtolower($_GET['nomb']);
   $nombre= ucfirst($nombre);
   $verificarregistro = $registro->RegistrarsePortal($_GET['email'],$nombre,$_GET['apell']
   ,$_GET['fecha'],$_GET['genero'],$_GET['pass']);
   echo $verificarregistro;
?>
```

Salida:

Figura 12 Representación de la capa de presentación

A continuación detallamos los documentos generado en la etapa de análisis los cuales fueron los siguientes:

ESPECIFICACIÓN DE REQUISITOS DEL SISTEMA.

I. INTRODUCCIÓN

Propósito del documento

Describir el conjunto de funcionalidades, restricciones y requisitos que debe cumplir la aplicación.

Alcance

El nombre de la aplicación será Amigos en línea, el cual realizara las siguientes funciones:

- Registrarse en el portal.
- Ingresar al portal.
- Modificar datos personales.
- Buscar amigos por nombre.
- Enviar peticiones.
- Peticiones Recibidas.
- Agregar amigo.
- Rechazar amigo.
- Borrar amigos.
- Enviar mensajes.
- Recibir mensajes
- Borrar mensajes.
- Publicar noticias.
- Borrar noticias.
- Agregar comentario.
- Recibir comentarios de tus amigos.
- Subir foto.
- Borrar foto.
- Agregar videos a tu perfil.
- Eliminar video.
- Ver Perfil de tus amigos.
- Salir del Portal.
- Dar de baja al Usr.

Definiciones, acrónimos y abreviaturas usadas.

- BD: Base de datos.
- RF: Requisitos Funcionales.
- Usr: usuario

Referencias.

Ninguna.

Visión General.

Primeramente se realizará una descripción general del requisito de las funcionalidades del portal web y posteriormente se detallarán, las diferentes restricciones (funcionales, de seguridad, de usuario, etc.), que debe cumplir el software, para poder intercambiar información con los usuarios que deseen tener acceso.

DESCRIPCIÓN GENERAL.

Relación con proyectos pasados:

Ninguna.

Relación con proyectos actuales:

Ninguna.

Funciones y propósitos del sistema

- El usuario podrá Registrarse al portal.
- Ya registrados los usuarios podrán loguear con su correo y contraseña y entrar al portal web para hacer uso de él.
- Permitirá a los usuarios modificar sus datos personales.
- Podrán hacer búsquedas de sus amigos y agregarlos si desean.
- El usuario podrá crear su propia galería de imágenes subiendo las fotos que desee.
- El usuario tendrá la posibilidad de enviar y recibir mensajes de los amigos agregados.
- Podrán navegar entre sus amigos y desde ahí ver el perfil de cada uno, enviar comentarios, o eliminarlos.
- Los usuarios tendrán la facultad de publicar noticias que podrán verse al ingresar a su perfil.
- Los usuarios podrán ver los comentarios que le han hecho en su perfil.
- El usuario podrá eliminar información que ya no tenga validez, estén erradas o cualquiera que sea su necesidad.
- El usuario podrá darse de baja en cualquier momento que desee.

Restricciones Generales

Cada usuario registrado tendrá una cuota máxima que no sobrepasará el límite de 1 GB.
EL navegador que se utilizará será la versión Mozilla Firefox 2.0 y superiores.
Los usuarios podrán acceder libremente, pero no podrá modificar datos de los demás.
Los usuarios solo podrán ingresar al portal web con su email y contraseña.

REQUISITOS ESPECÍFICO.

Requisitos funcionales.

RF1 Registrarse al portal.

1.1 Especificación:

1.1.1 Introducción:

Se permitirá al Usr. Registrarse al portal para tener acceso a éste.

1.1.2 Entradas:

Datos proporcionados por el Usr.:

- Email
- Nombre
- Apellido
- Sexo
- Fecha Nacimiento
- Contraseña

1.1.3 Procesos:

Se validarán los datos introducidos y se creará un nuevo registro en la Base de Datos.

1.1.4 Salida

Se mostrará un mensaje en el cual indica que el usuario ha sido registrado satisfactoriamente.

1.2 Interfaces externas:

Interfaz de usuario:

La forma de introducir los datos será de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF2 Ingresar al portal.

1.1 Especificación:

1.1.1 Introducción:

El Usr. ya registrado podrá ingresar al portal y hacer uso de este.

1.1.2 Entradas:

Datos proporcionados por el Usr.:

- Email
- Contraseña

1.1.3 Procesos:

Se validarán los datos introducidos por el Usr. que serán verificados en la BD.

1.1.4 Salida

Aparecerá por pantalla la página donde se muestra el perfil personal de Usr.

1.2 Interfaces externas:

Interfaz de usuario:

Se mostrará la página principal del usuario registrado.

Interfaz de hardware:

Se utilizara cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF3 Modificar datos personales.

1.1 Especificación:

1.1.1 Introducción:

El Usr. podrá modificar o completar sus datos en caso que desee hacerlo.

1.1.2 Entradas:

Datos proporcionados por el Usr.:

- Nombre
- Apellido
- Fecha de nacimiento
- Sexo
- Teléfono
- Foto.

1.1.3 Procesos:

Los datos del Usr. serán verificados y luego insertados en la BD.

1.1.4 Salida

Aparecerá por pantalla un mensaje que indica que los datos están actualizados, estos datos se mostraran en el perfil de Usr.

1.2 Interfaces externas:

Interfaz de usuario:

La forma de introducir los datos será de forma interactiva.

Interfaz de hardware:

Se utilizara cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF4 Buscar amigos por nombre.

1.1 Especificación:

1.1.1 Introducción:

Se realizará una búsqueda de todas las coincidencias por nombre que existan en la base de datos.

1.1.2 Entradas:

Datos proporcionados por el Usr.:

- Nombre.

1.1.3 Procesos:

El nombre introducido será validado, se realizará una búsqueda en la BD en donde se devolverán todas las coincidencias.

1.1.4 Salida

Aparecerá por pantalla todas las coincidencias encontradas y si no hay se enviará un mensaje de que no ha habido ninguna coincidencia.

1.2 Interfaces externas:

Interfaz de usuario:

La forma de introducir los datos será de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF5 Enviar Peticiones.

1.1 Especificación:

1.1.1 Introducción:

El Usr. podrá enviar peticiones de amigo a otros usuarios.

1.1.2 Entradas:

Email del usuario que recibirá la petición.
Email del usuario que envía la petición.
Nota de la petición.

1.1.3 Procesos:

El Usr busca la persona a la que desea agregar y le envía la petición.

1.1.4 Salida

Aparecerá por pantalla el mensaje de petición enviada.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF6 Peticiones Recibidas

1.1 Especificación:

1.1.1 Introducción:

El Usr. podrá recibir peticiones o solicitudes de amigo de otros usuarios.

1.1.2 Entradas:

Email de la persona que recibe la petición.

1.1.3 Procesos:

Se hace la consulta a la base de datos para ver si hay peticiones y en caso de que hayan, se traen.

1.1.4 Salida

Aparecerá por pantalla las nuevas peticiones recibidas las cuales el Usr podrá aceptar o rechazar.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF7 Agregar amigo.

1.1 Especificación:

1.1.1 Introducción:

Se agregará a un amigo automáticamente aceptando la petición recibida.

1.1.2 Entradas:

Email del Usr. que recibe la petición.
Email del Usr. que envía la petición.

1.1.3 Procesos:

Cuando la petición ha sido aceptada, el amigo se agregará automáticamente en ambos perfiles.

1.1.4 Salida

Se mostrará por pantalla el amigo agregado a tu perfil.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF8 Rechazar amigo.

1.1 Especificación:

1.1.1 Introducción:

Podrá rechazar peticiones de un amigo no deseado.

1.1.2 Entradas:

Email del Usr. que recibe la petición.
Email del Usr . que envía la petición.

1.1.3 Procesos:

Cuando la petición ha sido rechazada, se borra la petición automáticamente de la BD.

1.1.4 Salida

Se mostrará por pantalla las peticiones faltantes o si no se mostrará un mensaje que dice No hay peticiones.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF9 Borrar amigo.

1.1 Especificación:

1.1.1 Introducción:

El Usr. podrá borrar a una persona de las que tiene en su lista de amigos si lo desea.

1.1.2 Entradas:

Email del Usr.
Email del amigo

1.1.3 Procesos:

Al borrarse el amigo se borra automáticamente de de ambos perfiles y de la base de datos.

1.1.4 Salida

Se mostrará por pantalla un mensaje de confirmación y automáticamente se borrará.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF10 Enviar mensajes.

1.1 Especificación:

1.1.1 Introducción:

Se podrá enviar mensajes a cualquiera de los amigos que tenga agregado en su lista.

1.1.2 Entradas:

Email de la persona a enviar.
Asunto.
Email de la persona que envía el mensaje.
Fecha de envío.
Adjunto.

1.1.3 Procesos:

Se validan los datos y se insertan en la base de datos.

1.1.4 Salida:

Se mostrará por pantalla un mensaje de aviso;"mensaje enviado".

1.2 Interfaces externas:

Interfaz de usuario:

La forma de introducir los datos será de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF11 Recibir Mensajes.

1.1 Especificación:

1.1.1 Introducción:

Se podrá recibir mensajes de otros usuarios.

1.1.2 Entradas:

Email del Usr. que ha recibido el mensaje.

1.1.3 Procesos:

Se verifica en la BD si hay mensajes y se manda a traer.

1.1.4 Salida:

Se mostrarán por pantalla los mensajes recibidos.

1.2 Interfaces externas:

Interfaz de usuario:

El usuario verá los mensajes recibidos a través de una página que muestre dichos mensajes de forma tabular.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF12 Borrar mensajes

1.1 Especificación:

1.1.1 Introducción:

En este proceso el Usr., podrá borrar los mensajes que ha enviado o recibido si desea.

1.1.2 Entradas:

Id del mensaje.
Email del Usr.
Tipo er. (mensaje enviado o recibido)

1.1.3 Procesos:

El mensaje se borrará de la base de datos identificado por Id y de tipo de enviado.

1.1.4 Salida

Se mostrará por pantalla un mensaje de confirmación, Mensaje Borrado.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF13 Publicar noticias.

1.1 Especificación:

1.1.1 Introducción:

Se podrán publicar noticias o anuncios.

1.1.2 Entradas:

Datos proporcionados por el Usr. :

- Imagen(opcional)
- Titulo
- Anuncio

1.1.3 Procesos:

Los datos introducidos por el Usr. serán verificados y luego insertados en la BD.

1.1.4 Salida

Se desplegará una pestaña en donde se mostrarán las noticias que ha publicado.

1.2Interfaces externas:

Interfaz de usuario:

La forma de introducir los datos será de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF14 Borrar noticias.

1.1 Especificación:

1.1.1 Introducción:

El Usr podrá borrar las noticias publicadas por él.

1.1.2 Entradas:

Id de la Noticia.

Email del Usr. que la publicó.

1.1.3 Procesos:

Se borrará las noticias seleccionadas de la base de Datos, identificada por su ID e email del que la publicó.

1.1.4 Salida

Se mostrará por pantalla un mensaje de confirmación de borrado.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF15 Agregar comentario a tus amigos

1.1 Especificación:

1.1.1 Introducción:

Se podrán agregar comentarios entre amigos.

1.1.2 Entrada:

Email del Usr. actual.
Contenido del comentario.
Fecha de enviado.
Remitente.
Id del comentario.

1.1.3 Procesos:

Se validarán los datos y se insertará el Comentario en la BD.

1.1.4 Salida:

Se mostrará por pantalla un mensaje de confirmación.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF16 Recibir comentarios

1.1 Especificación:

1.1.1 Introducción:

El Usr. podrá recibir comentarios de otros usuarios.

1.1.2 Entradas:

Email del Usr. que recibe los comentarios.

1.1.3 Procesos:

Se verifica en la BD si hay comentarios y si los hay, serán extraídos.

1.1.4 Salida

Se mostrarán por pantalla los comentarios recibidos.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF17 Subir fotos.

1.1 Especificación:

1.1.1 Introducción:

El Usr. podrá subir sus fotos y crear su propia galería.

1.1.2 Entrada:

Email del Usr.
Ruta de la foto a subir.
Foto.

1.1.3 Procesos:

Se validará los datos y se insertarán las fotos en la Base de Datos.

1.1.4 Salida:

Se mostrará por pantalla las fotos, las cuales formarán una galería de fotos.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF18 Borrar foto.

1.1 Especificación:

1.1.1 Introducción:

En este proceso el Usr. podrá borrar fotos de su galería.

1.1.2 Entradas:

Email del Usr.
Nombre de la foto.

1.1.3 Procesos:

Se eliminará la foto seleccionada del perfil y de la BD.

1.1.4 Salida

Se mostrará por pantalla un mensaje de que la foto ha sido borrada.

1.2 Interfaces externas:

Interfaz de usuario:

Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF19 Agregar videos a tu perfil.

1.1 Especificación:

1.1.1 Introducción:

Se podrá agregar uno o varios videos en el perfil del usuario.

1.1.2 Entrada:
Email del Usr.
Link del video.
Nombre con que se guardará el video.
Id del video

1.1.3 Procesos:
Se guarda en la base de Datos el Link del video agregado.

1.1.4 Salida:
Se mostrará por pantalla el video seleccionado y la lista de todos los videos agregados al perfil.

1.2 Interfaces externas:
Interfaz de usuario:
Se realizará de forma interactiva.

Interfaz de hardware:
Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:
El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF20 Eliminar video.
1.1 Especificación:
1.1.1 Introducción:
Los Usr podrán eliminar videos de su perfil.

1.1.2 Entradas por pantalla:
Email del Usr.
Id del video.

1.1.3 Procesos:
Se borrará de la base de datos el video seleccionado por el Usr.

1.1.4 Salida:
Se mostrará por pantalla un mensaje de confirmación de video borrado.

1.2 Interfaces externas:
Interfaz de usuario:
Se realizará de forma interactiva.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF21 Ver perfil de tu amigo.

1.1 Especificación:

1.1.1 Introducción:

Usr. podrá ver el perfil o los datos de un amigo en contacto.

1.1.2 Entradas por pantalla:

Email del amigo.

1.1.3 Procesos:

Se traerán los datos desde la BD para ser mostrados.

1.1.4 Salida:

Se mostrará por pantalla el perfil del amigo.

1.2 Interfaces externas:

Interfaz de usuario:

Se mostrará una página que contenga diferentes pestañas en las cuales se muestran los datos de la persona de quien desea ver perfil.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF22 Salir del Portal.

1.1 Especificación:

1.1.1 Introducción:

Se verifica si el usuario desea realmente salir del portal.

1.1.2 Entradas:

email del usuario.

1.1.3 Procesos:

Se borra toda la información que se haya recogido y asociado a la sesión activa.

1.1.4 Salida:

Se mostrará por pantalla el mensaje de confirmación que ha salido de sesión.

1.2 Interfaces externas:

Interfaz de usuario:

Se muestra por pantalla la opción para salir del portal web.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

RF23 Dar de baja al Usr.

1.1 Especificación:

1.1.1 Introducción:

El Usr. que ya no quiere formar parte o no tener cuenta en el portal, podrá darse de baja.

1.1.2 Entradas por pantalla:

Email del Usr.

1.1.3 Procesos:

Se verifica la cancelación de la cuenta del Usr. y se borraran todos sus datos de la BD.

1.1.4 Salida

Se mostrará por pantalla el mensaje de confirmación de cancelación.

1.2 Interfaces externas:

Interfaz de usuario:

Se presenta la opción para que el Usr. elimine su cuenta si él lo desea.

Interfaz de hardware:

Se utilizará cualquier ordenador que tenga acceso al servidor donde esté instalado el portal.

Interfaz de software:

El proceso interactúa con la BD y se lleva a cabo en el ordenador donde está instalado el sistema mediante el uso de un navegador web.

Interfaz de comunicación: Servidor Apache.

DIAGRAMA DECASOS DE USO

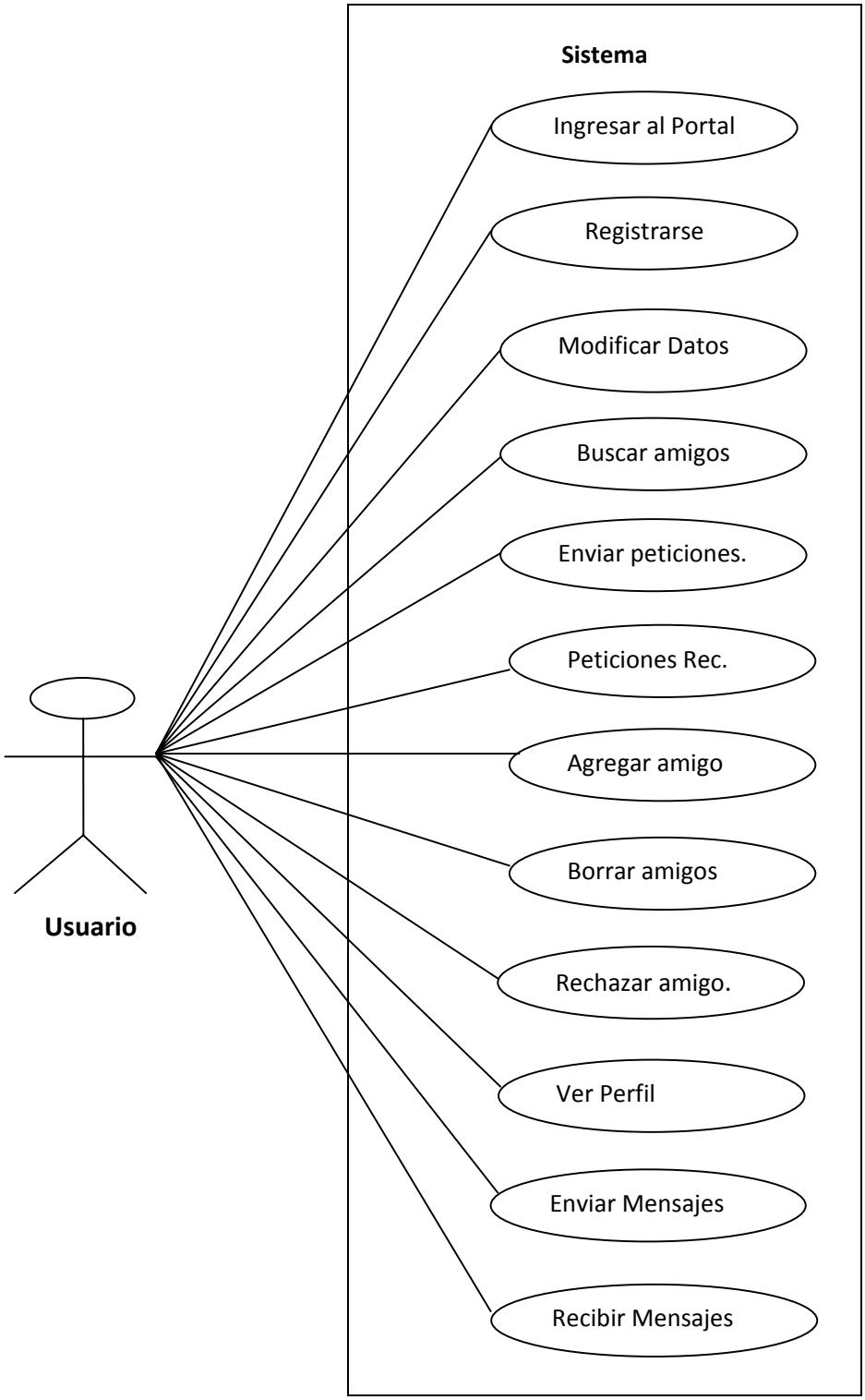


Figura 13 Diagrama de casos de usos

CASOS DE USO (Continuación)

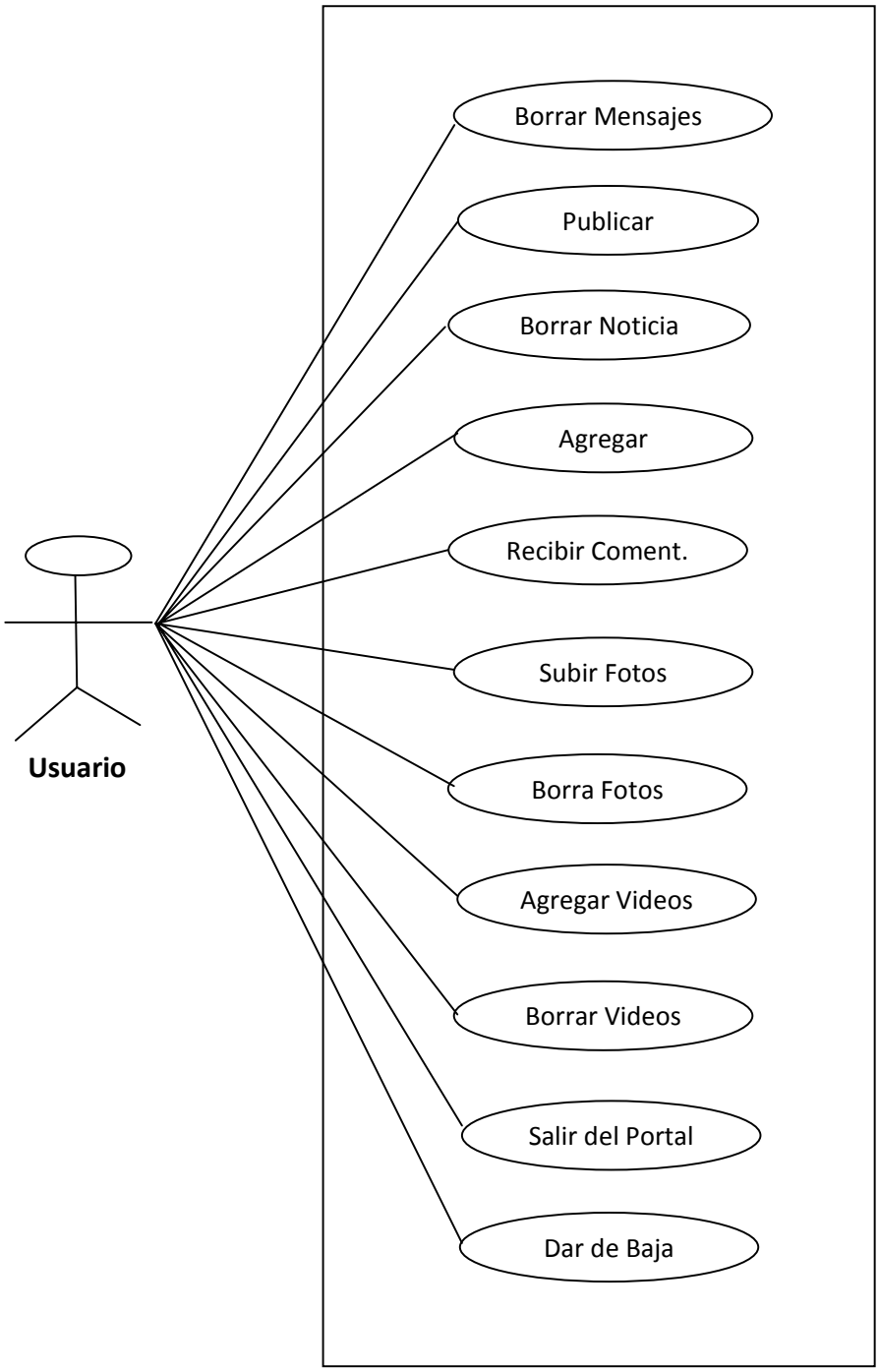
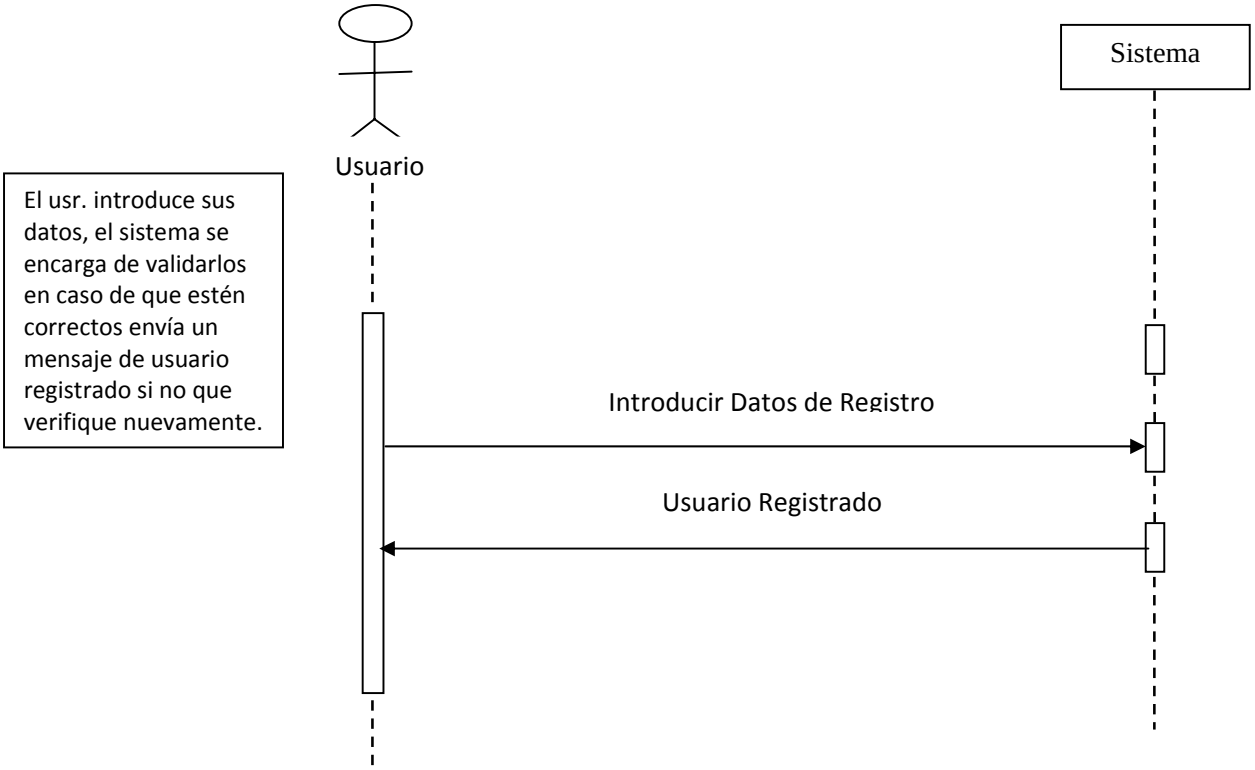


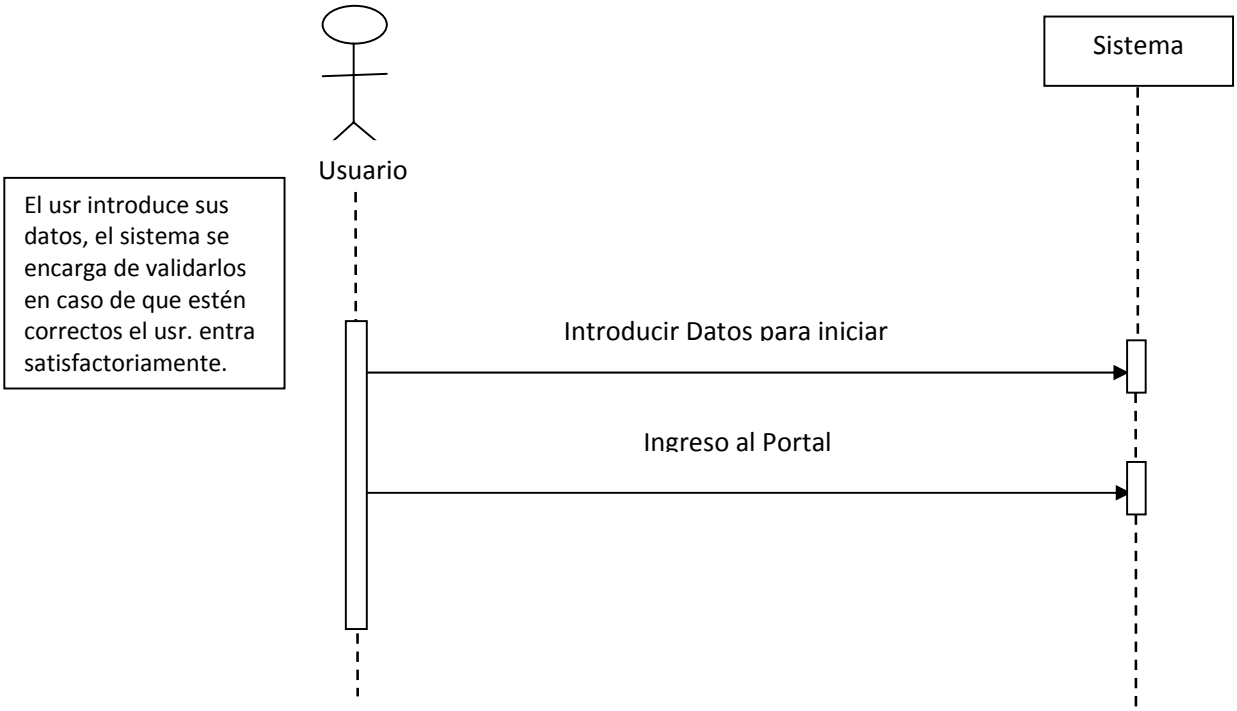
Figura 14 Diagrama de casos de usos (continuación)

DIAGRAMAS DE SECUENCIAS

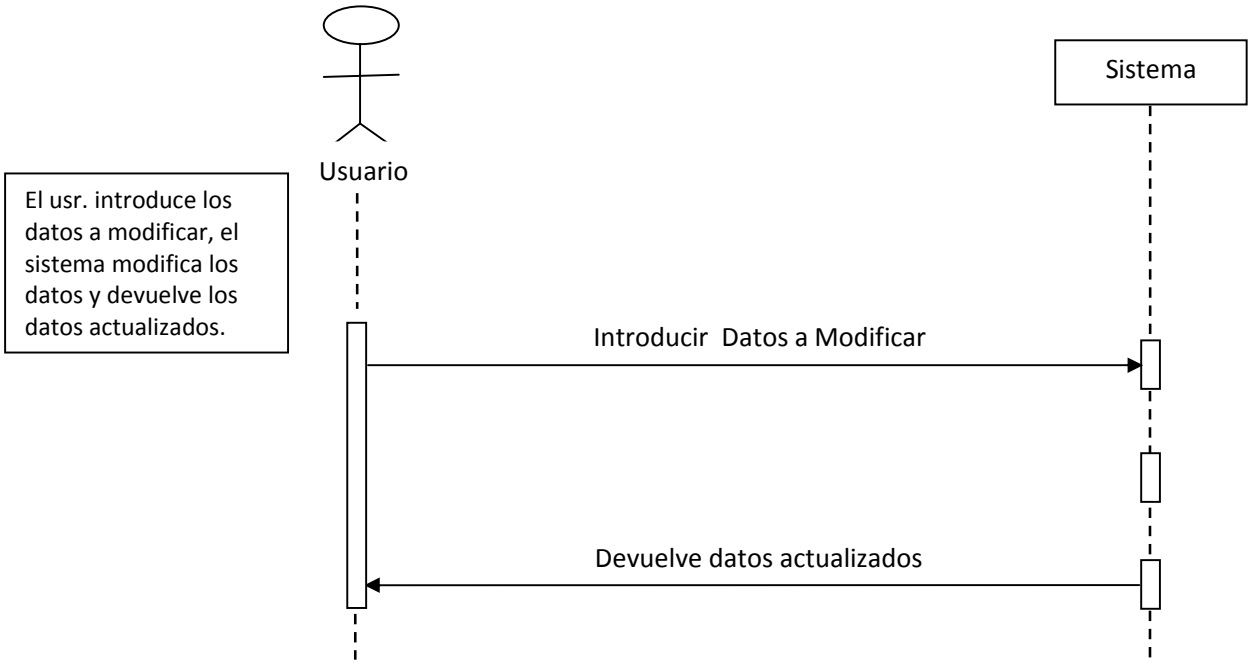
Registrarse al Portal



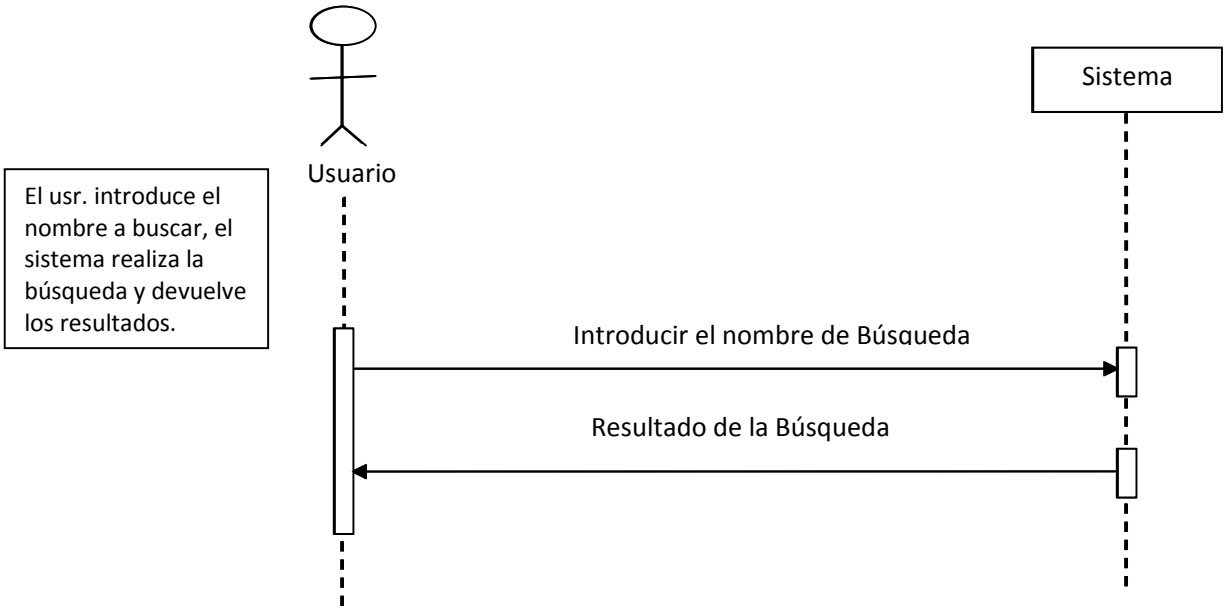
Ingresar al Portal



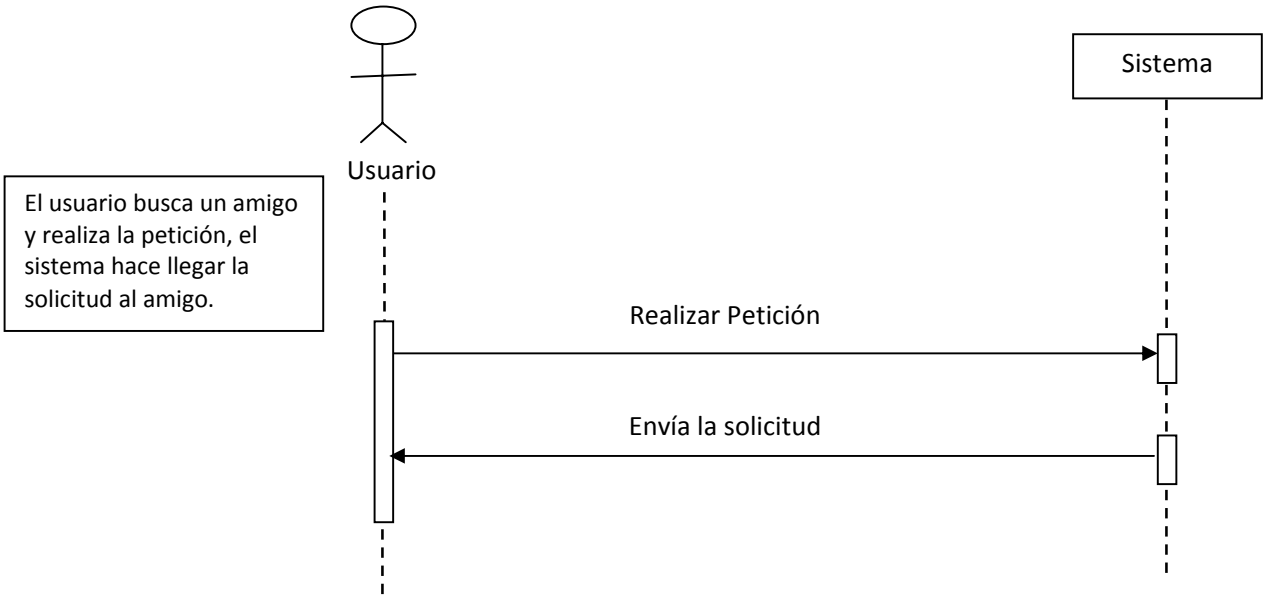
Modificar Datos Personales

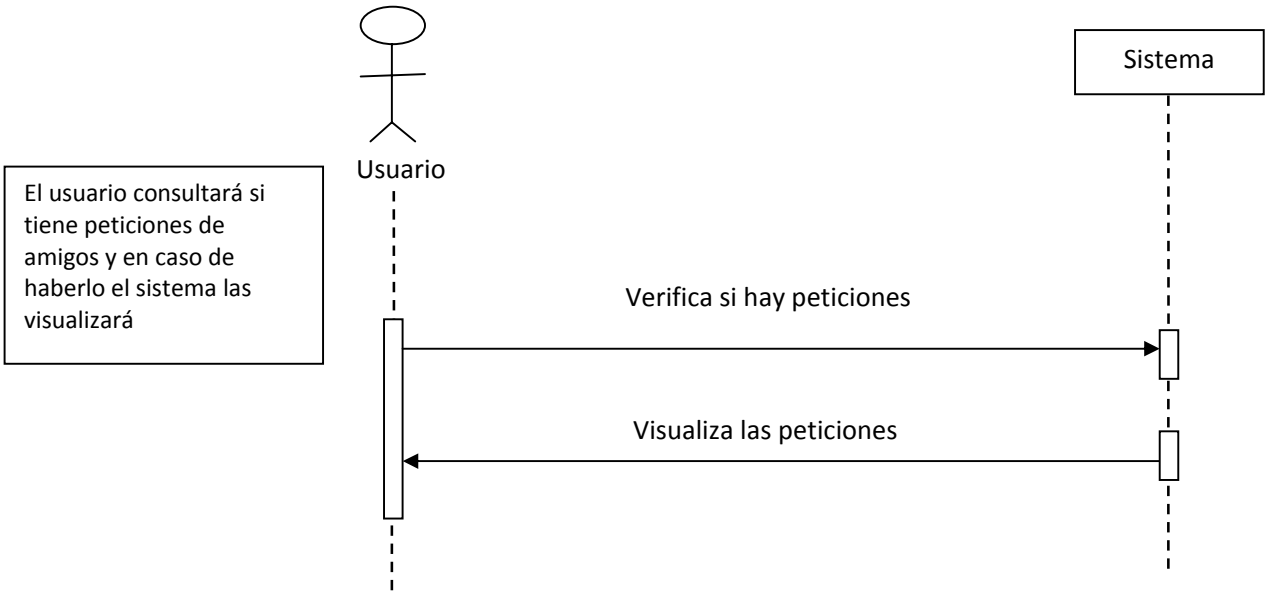


Buscar amigos por nombre

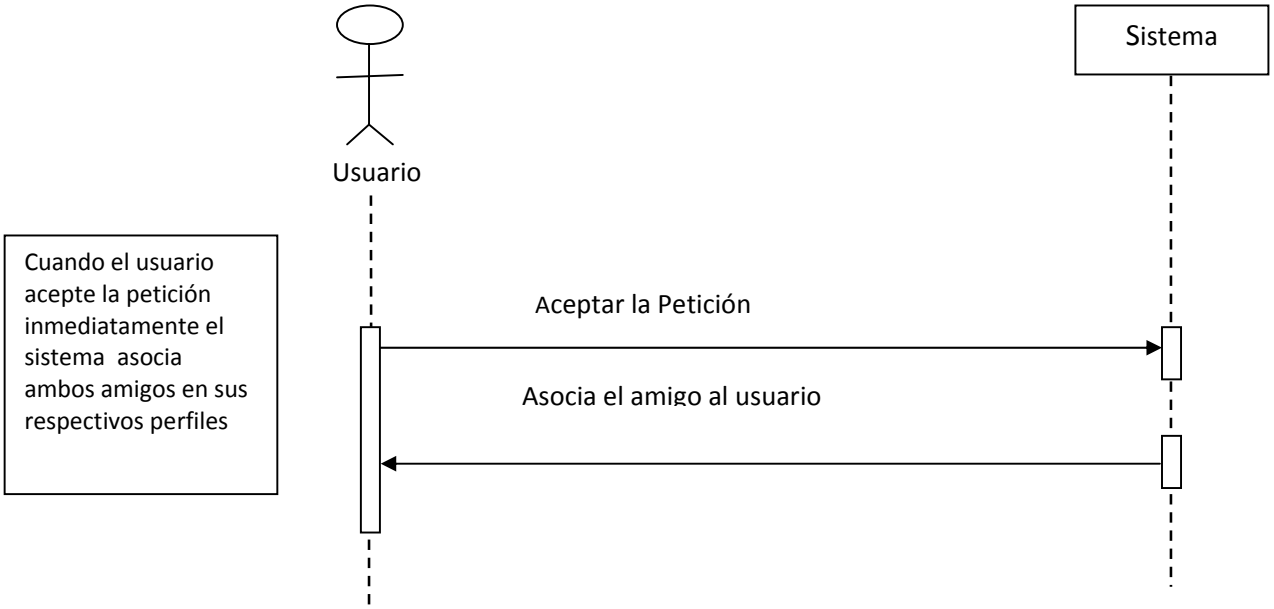


Enviar Peticiones de amigos



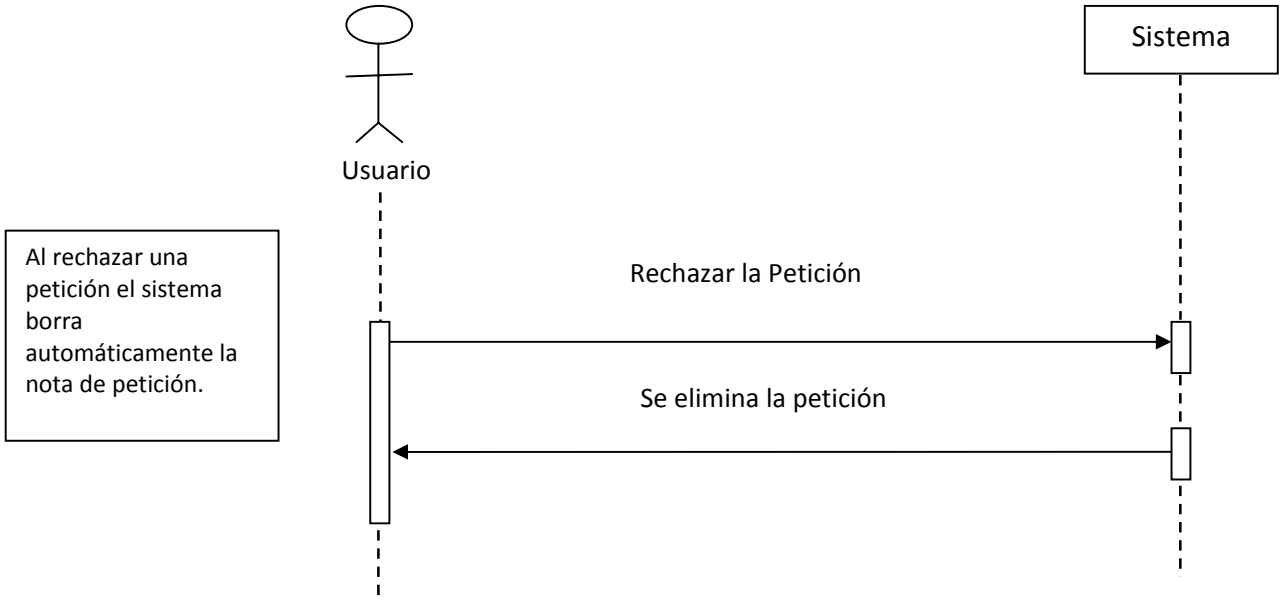


Agregar Amigo



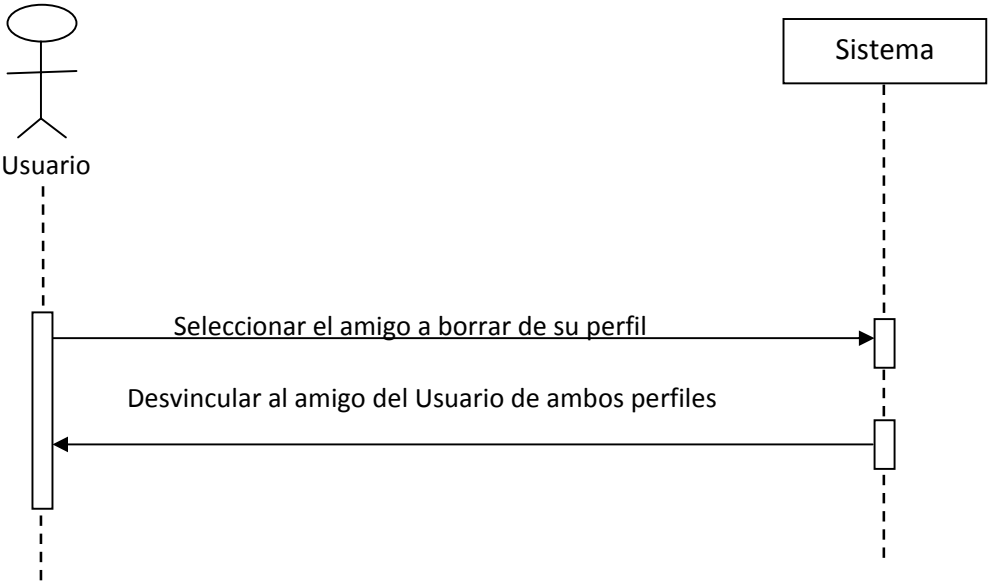
Ver Peticiones Recibidas

Rechazar Amigo



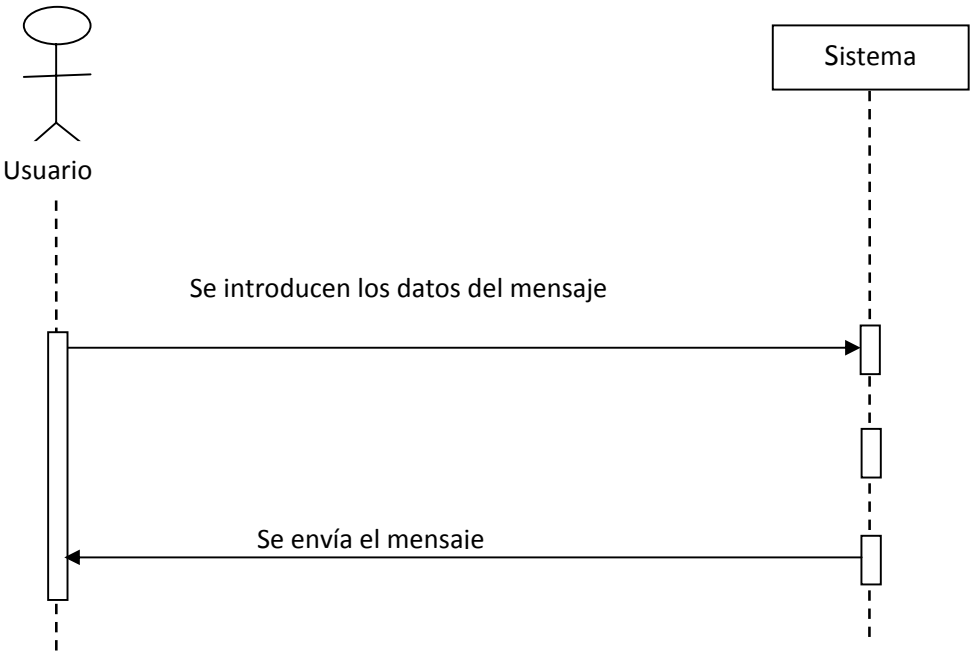
Borrar amigo

El usuario manda a borrar el amigo de su perfil, el sistema se encarga de desvincularlos de ambos perfiles

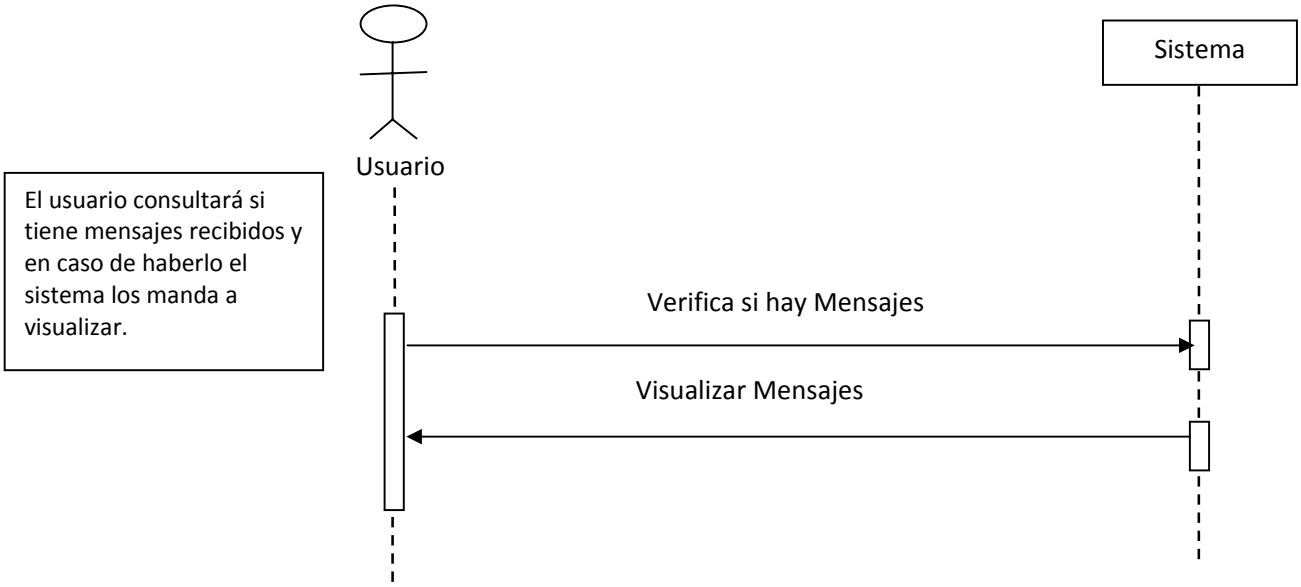


Enviar Mensajes

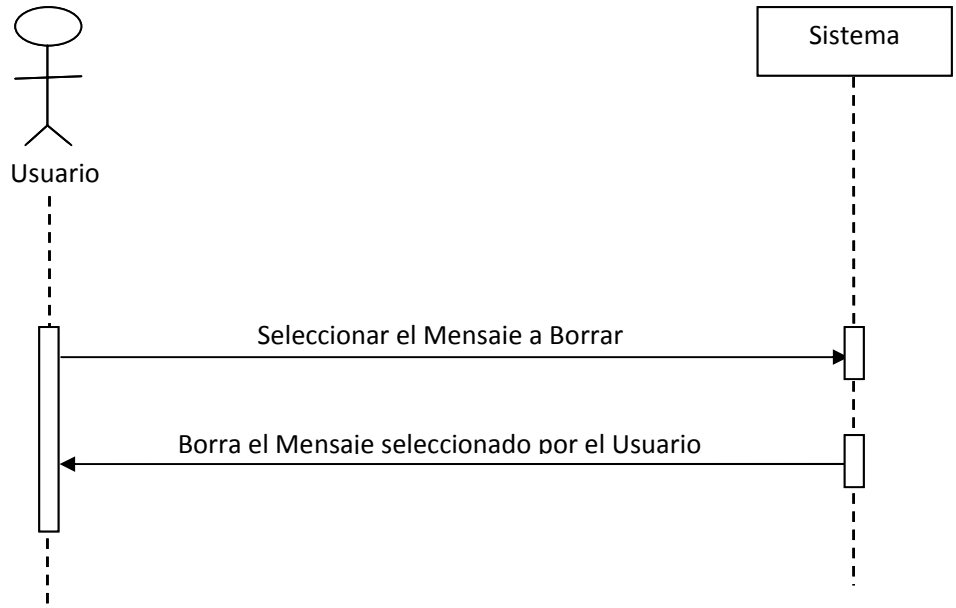
El usr. introduces los datos del mensaje el sistema los validas y envía el mensaje. ambos perfiles



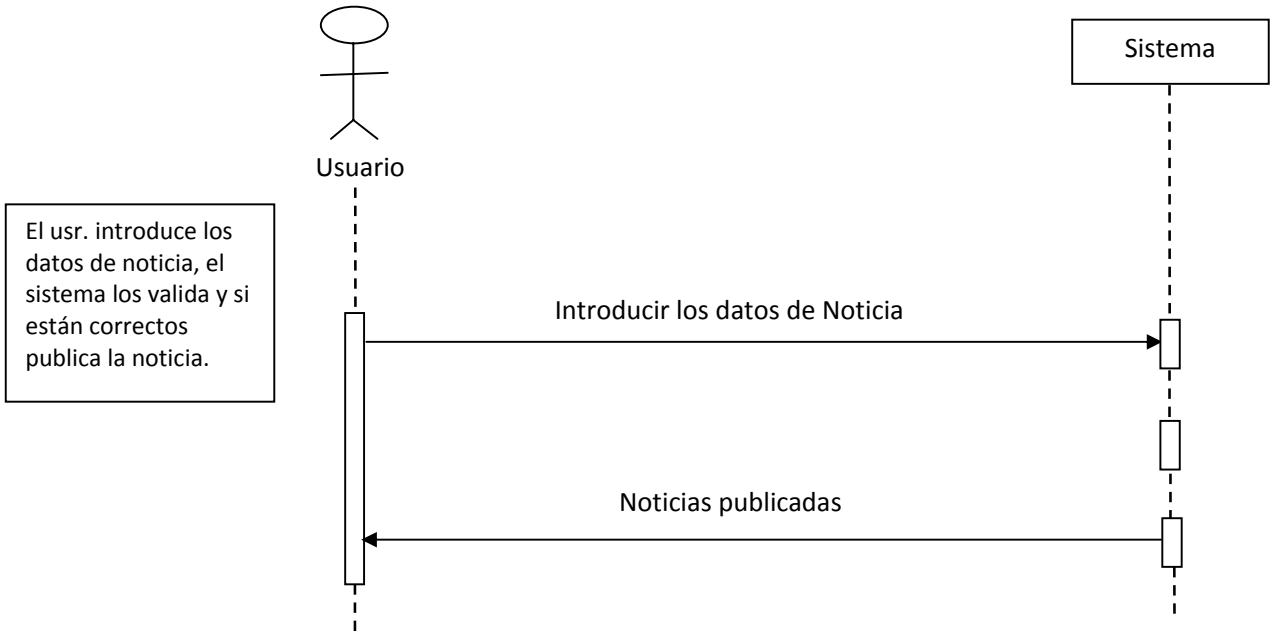
Recibir Mensajes



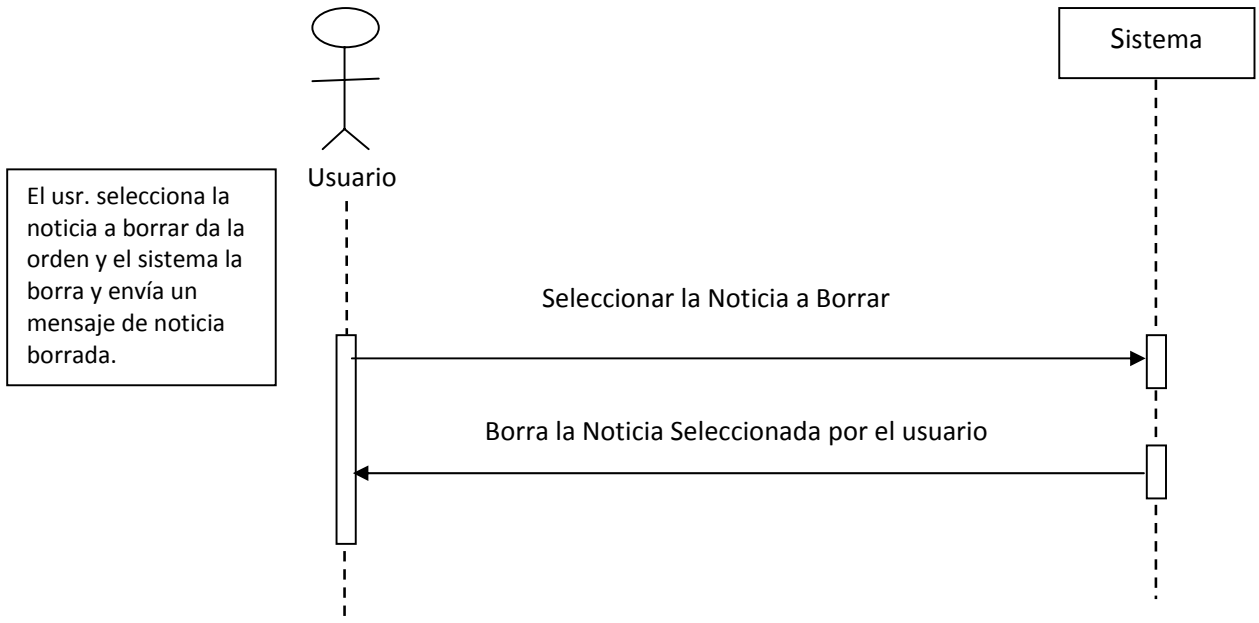
Borrar Mensajes



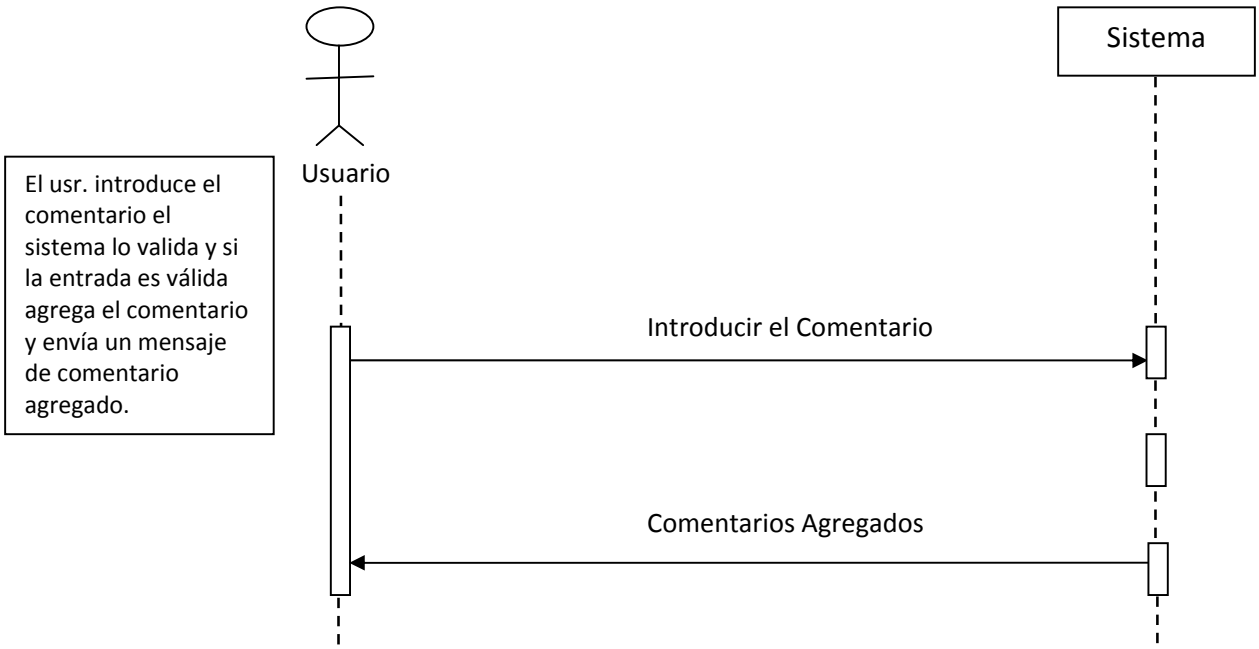
Publicar Noticias



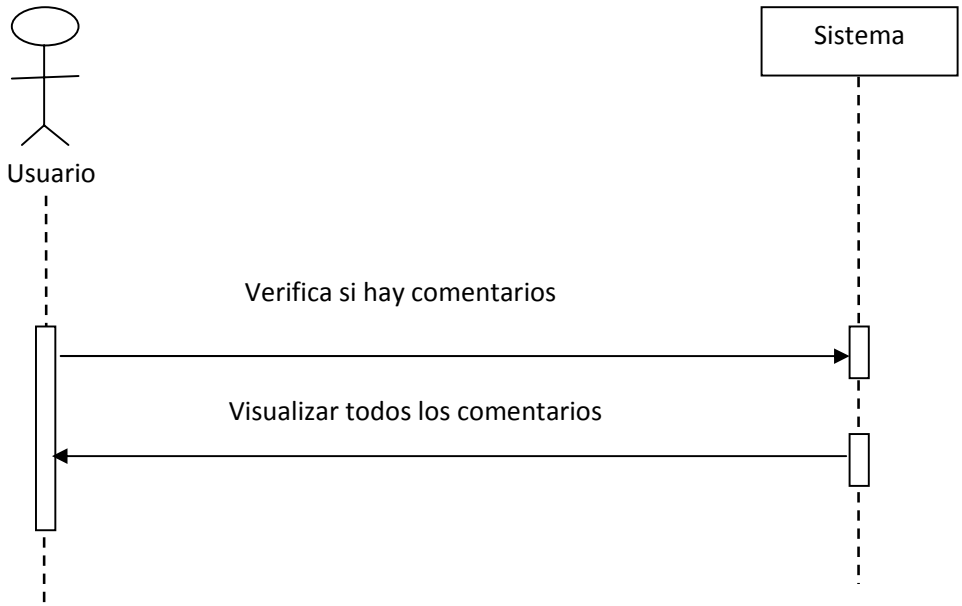
Borrar Noticias



Agregar Comentario a un amigo.

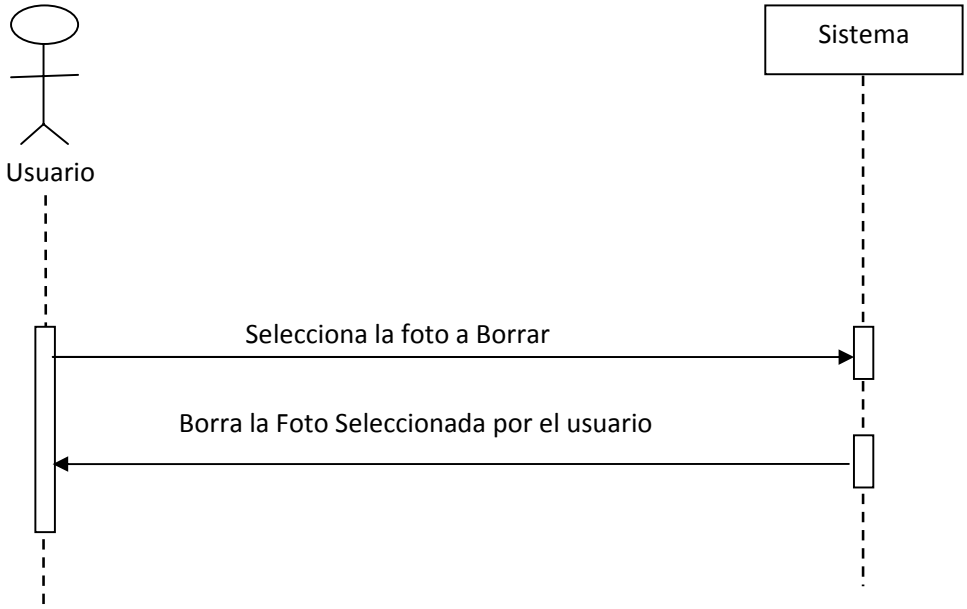


Recibir Comentarios

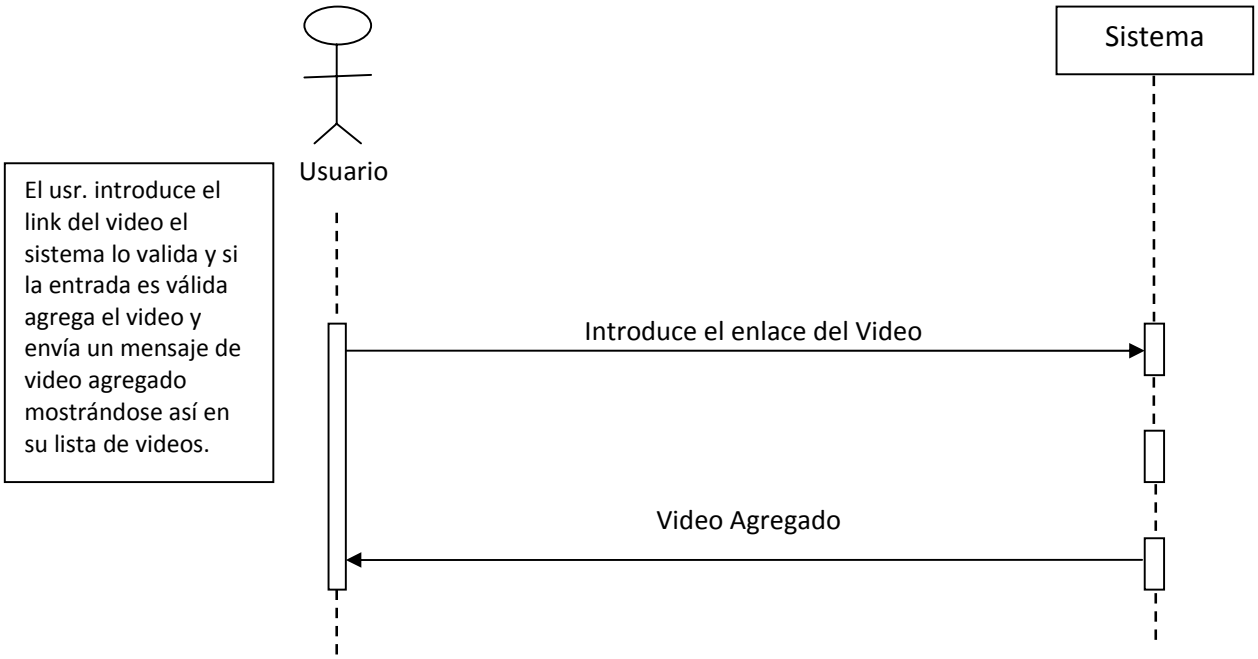


Subir Fotos

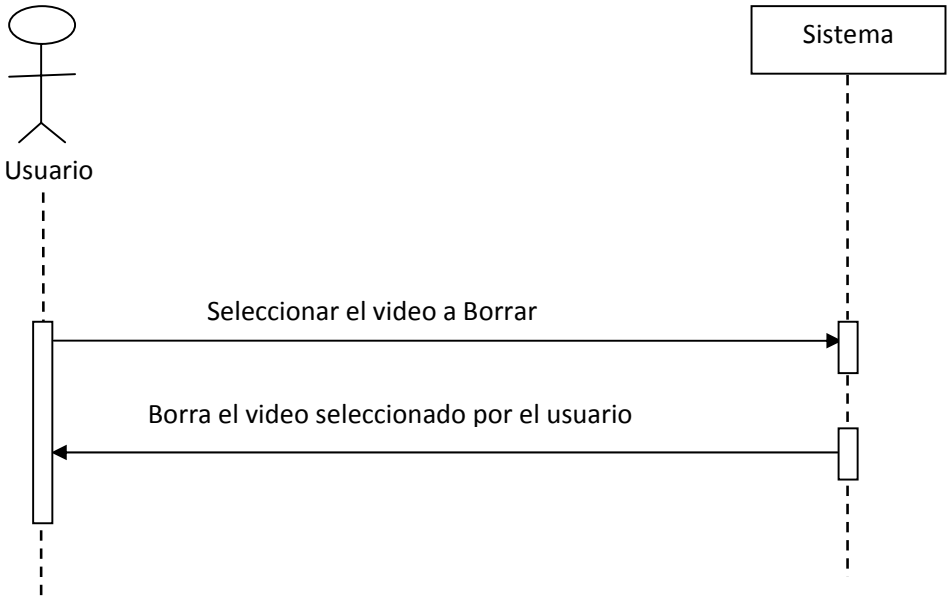
Borrar Foto



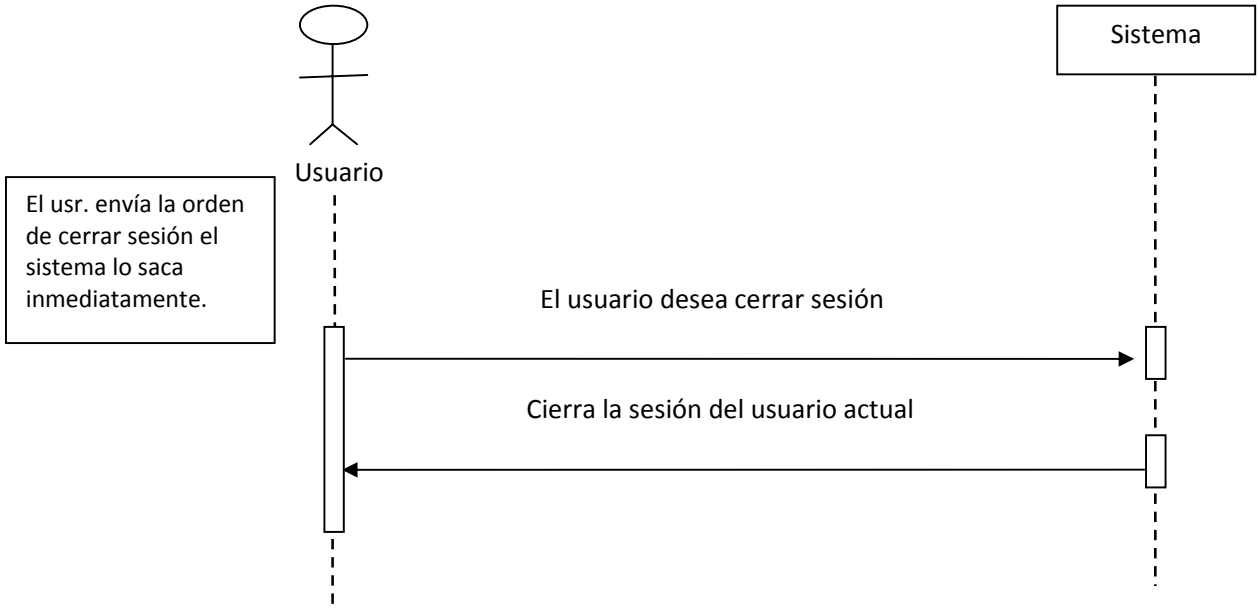
Agregar Video a tu Perfil



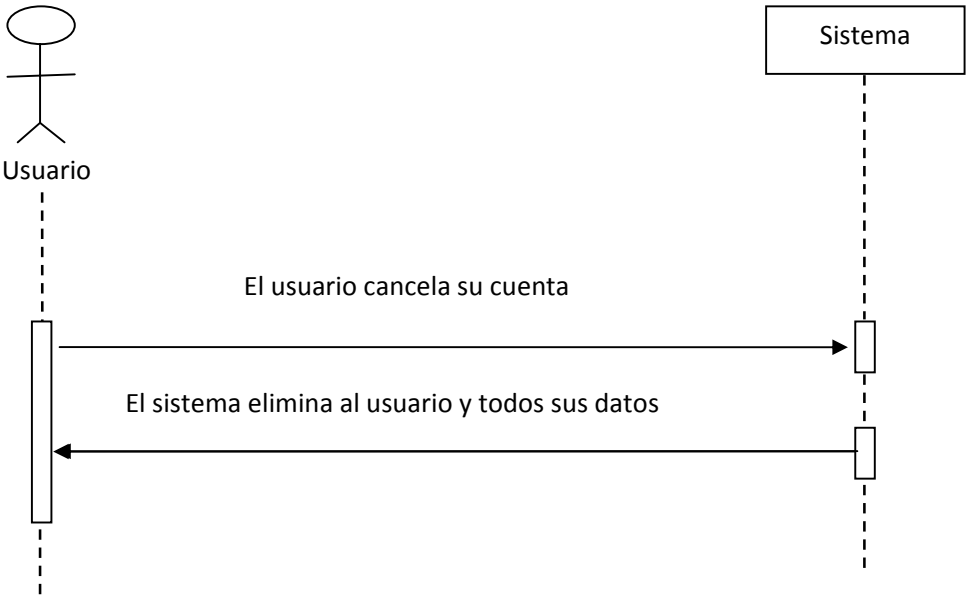
Eliminar Video



Salir del portal



Dar de baja del Portal



A continuación detallamos los documentos generado en la etapa de diseño los cuales fueron los siguientes:

DIAGRAMA DE CLASES.

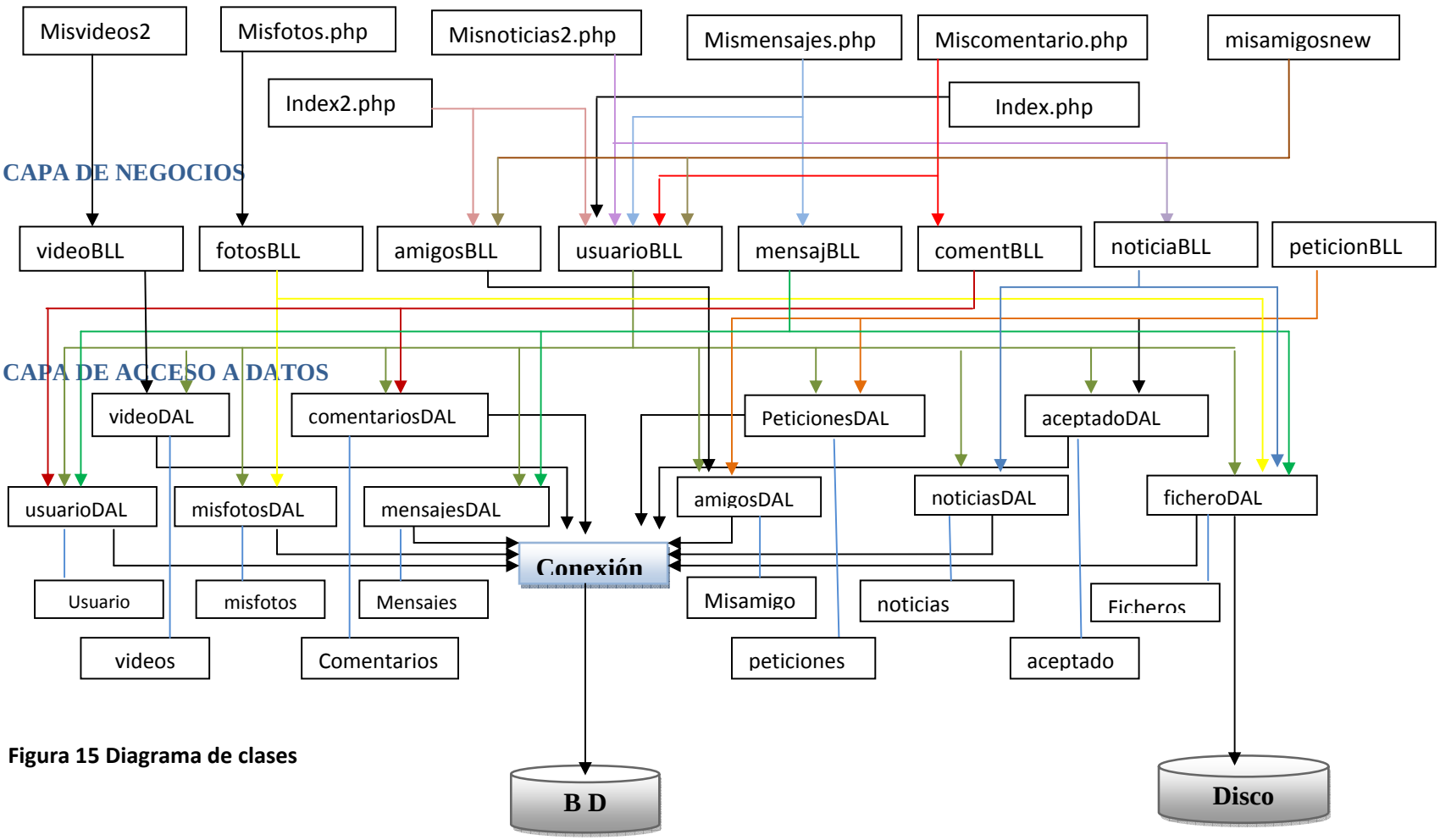


Figura 15 Diagrama de clases

DIAGRAMA RELACIONAL.

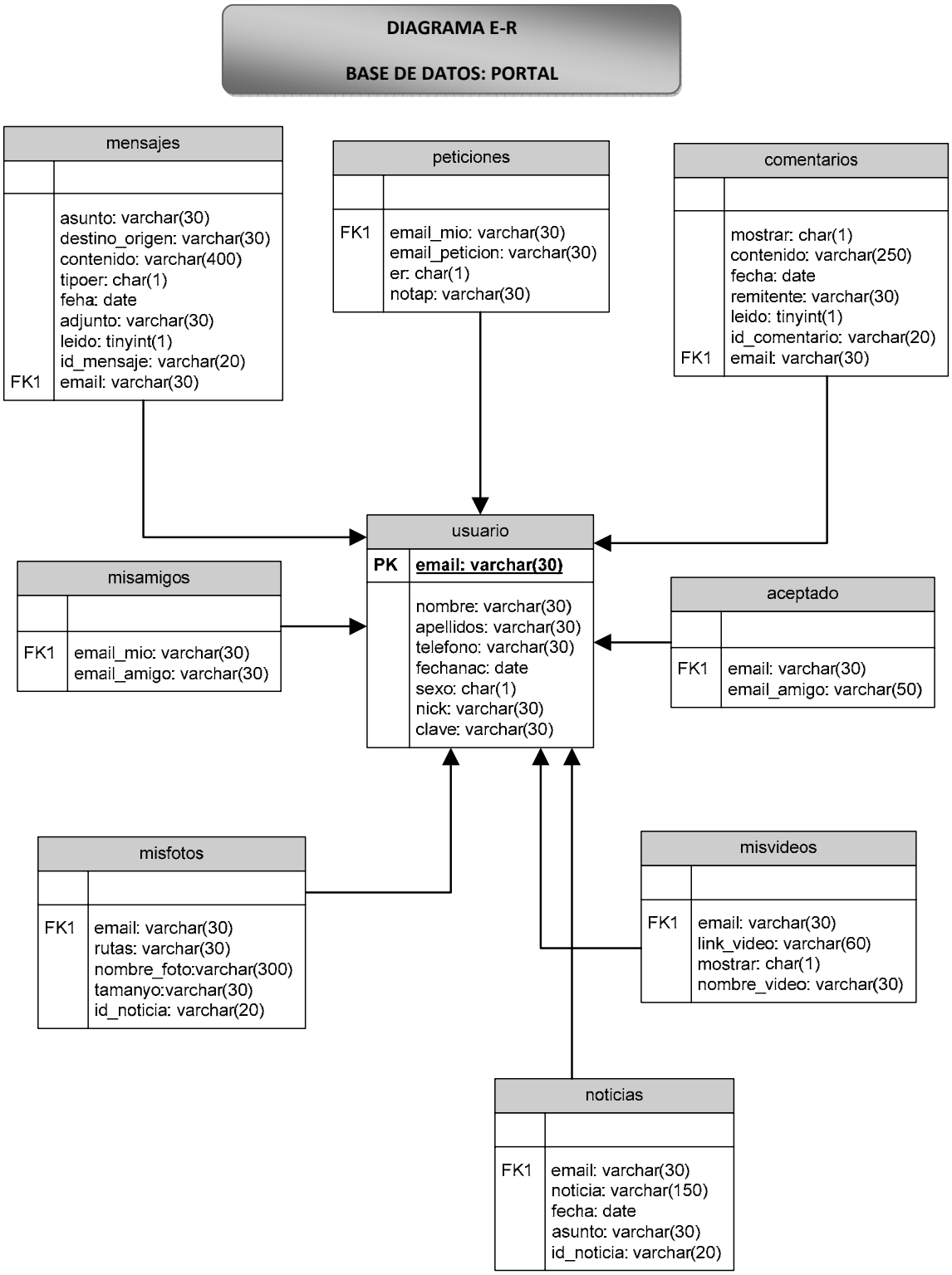


Figura 16 Diagrama relacional

DISEÑO DE DATOS

TABLA USUARIO

Almacena todos los datos del usuario

Nombre	Descripción	Tipo
email	**Llave primaria**	varchar(30)
nombre	Nombre del usuario	varchar(30)
apellidos	Apellidos del usuario	varchar(30)
teléfono	Número de teléfono del usuario	varchar(8)
Fechanac	Fecha de nacimiento del usuario	date
sexo	Identifica el sexo del usuario hombre o mujer	char(1)
foto	Foto del usuario	varchar(30)
clave	Clave del usuario	varchar(30)

TABLA PETICIONES

Almacena todos los datos de peticiones

Nombre	Descripción	Tipo
email_mio	Llave foránea Identificador email origen-destino	varchar(30)
email_peticion	Identificador email destino-origen	varchar(30)
Er	Envío y recibido de petición	char(1)
Notap	Nota de petición recibida-enviada	varchar(60)

TABLA NOTICIAS

Almacena todos los datos de Noticias

Nombre	Descripción	Tipo
Email	Llave foránea que relaciona con email usuario.	varchar(30)
Noticia	Contenido de la noticia	varchar(300)
Fecha	Fecha en que se publicó la noticia	date
Asunto	Tema de la noticia	varchar(30)
id_noticia	Discriminante de la tabla noticias	int(20)
nombre_foto	Nombre de la fotografía	varchar(300)
Ruta	Ubicación del fichero(fotografía) en el servidor	varchar(50)

TABLA MISVIDEOS

Almacena los datos de Videos

Nombre	Descripción	Tipo
Email	Llave foránea que relaciona con email usuario.	varchar(30)
link_video	Código del video a insertar	Varchar(400)
nombre_video	Nombre del video	varchar(30)
id_video	Discriminante de la tabla misvideos	int(20)

TABLA MISFOTOS

Almacena los datos de Fotos

Nombre	Descripción	Tipo
Email	Llave foránea que relaciona con email usuario.	varchar(30)
Ruta	Ubicación del fichero(fotografía) en el servidor	varchar(45)
nombre_foto	Nombre de la foto.	varchar(300)
Tamaño	El tamaño en byte de las fotos.	varchar(30)
Tipo	Especifica el formato de la foto.	Varchar(10)

TABLA MISAMIGOS

Almacena los datos de Amigos

Nombre	Descripción	Tipo
email_mio	Llave foránea que relaciona con email usuario.	varchar(30)
email_amigo	Email del amigo	varchar(30)

TABLA MENSAJES

Almacena los datos de Mensajes

Nombre	Descripción	Tipo
Email	Llave foránea que relaciona con email usuario.	varchar(30)
Asunto	Asunto del mensaje	varchar(30)
destino_origen	Usuario que envía o recibe el mensaje	varchar(30)
contenido	Contenido del mensaje	varchar(500)
tipoer	Indica que el mensaje ha sido enviado o recibido.	char(1)
fecha	Fecha que se recibe y se envía el mensaje	date
adjunto	Nombre del fichero que se adjunto el mensaje.	varchar(30)
leído	Indica si un mensaje ha sido leído o no.	int(1)
id_mensaje	Discriminante de la tabla mensajes	int(20)

TABLA COMENTARIOS

Almacena los datos de Comentarios

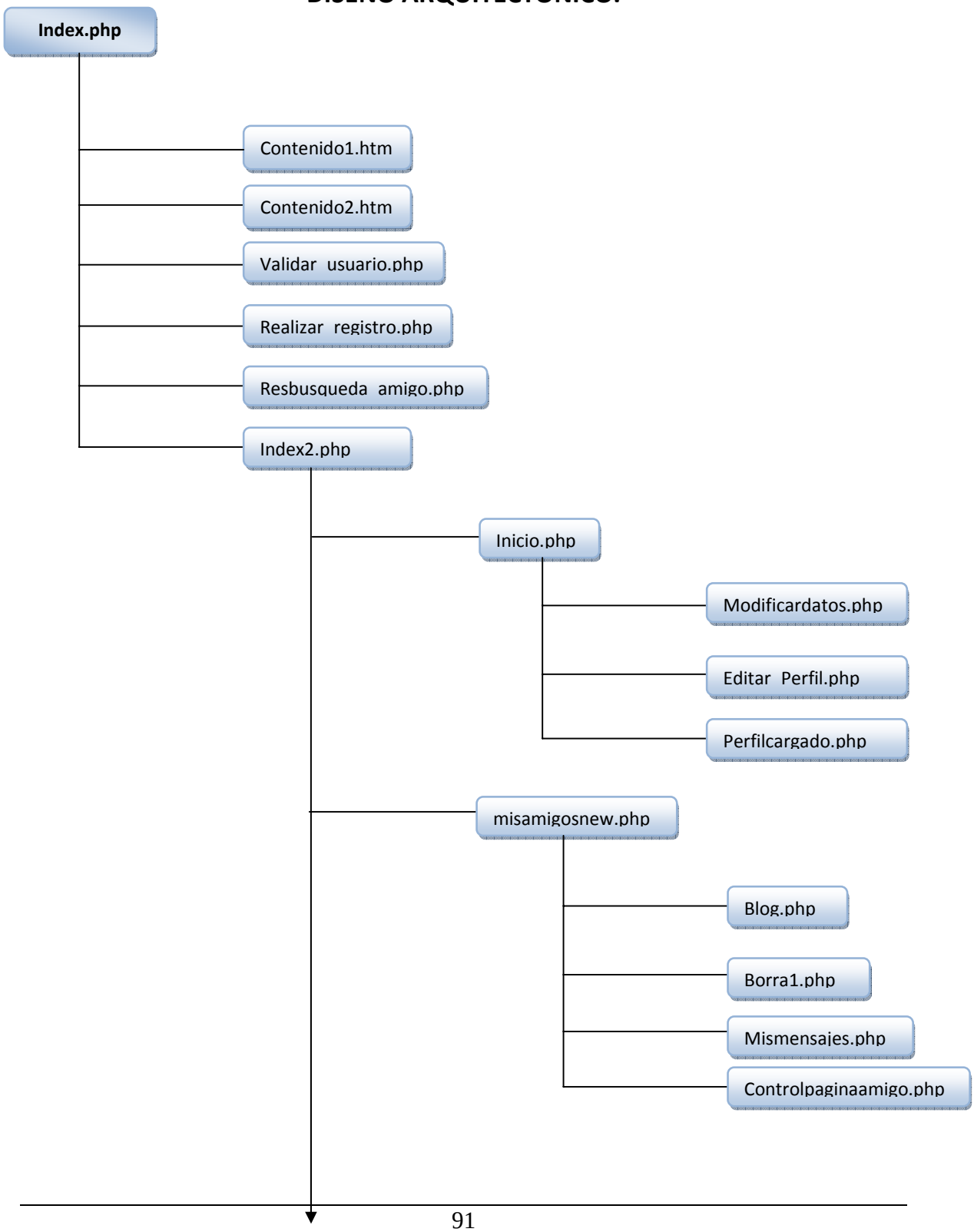
Nombre	Descripción	Tipo
email	Llave foránea que relaciona con email usuario.	varchar(30)
contenido	Contenido del comentario.	varchar(250)
fecha	Fecha en que fue escrito el comentario.	date
remitente	Usuario que envía el comentario.	varchar(30)
leído	Indica si un comentario ha sido leído o no.	tinyint(1)
Id_comentario	Discrimínate de la tabla comentarios	int(20)

TABLA ACEPTADO

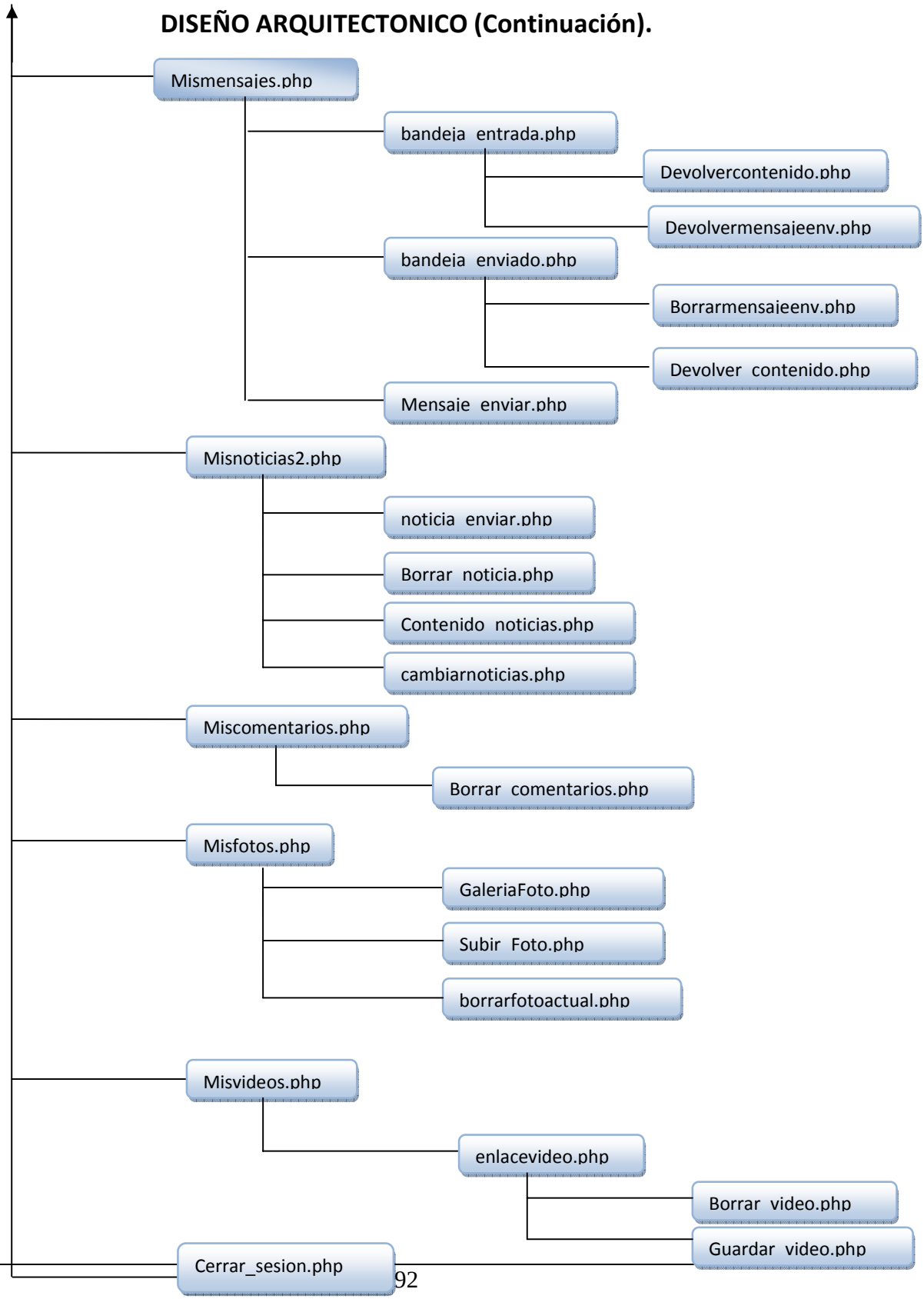
Almacena los datos de las Aceptaciones

Nombre	Descripción	Tipo
email	Llave foránea que relaciona con email usuario.	varchar(30)
email_amigo	Email del amigo	varchar(30)

DISEÑO ARQUITECTONICO.



DISEÑO ARQUITECTONICO (Continuación).



DISEÑO ARQUITECTONICO (Continuación).

NOTA: Blog.php desprende de misamigosnew.php

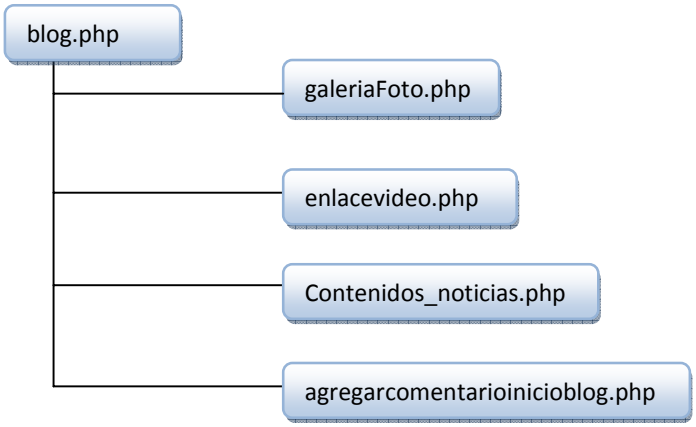


Figura 17 Diseño Arquitectónico

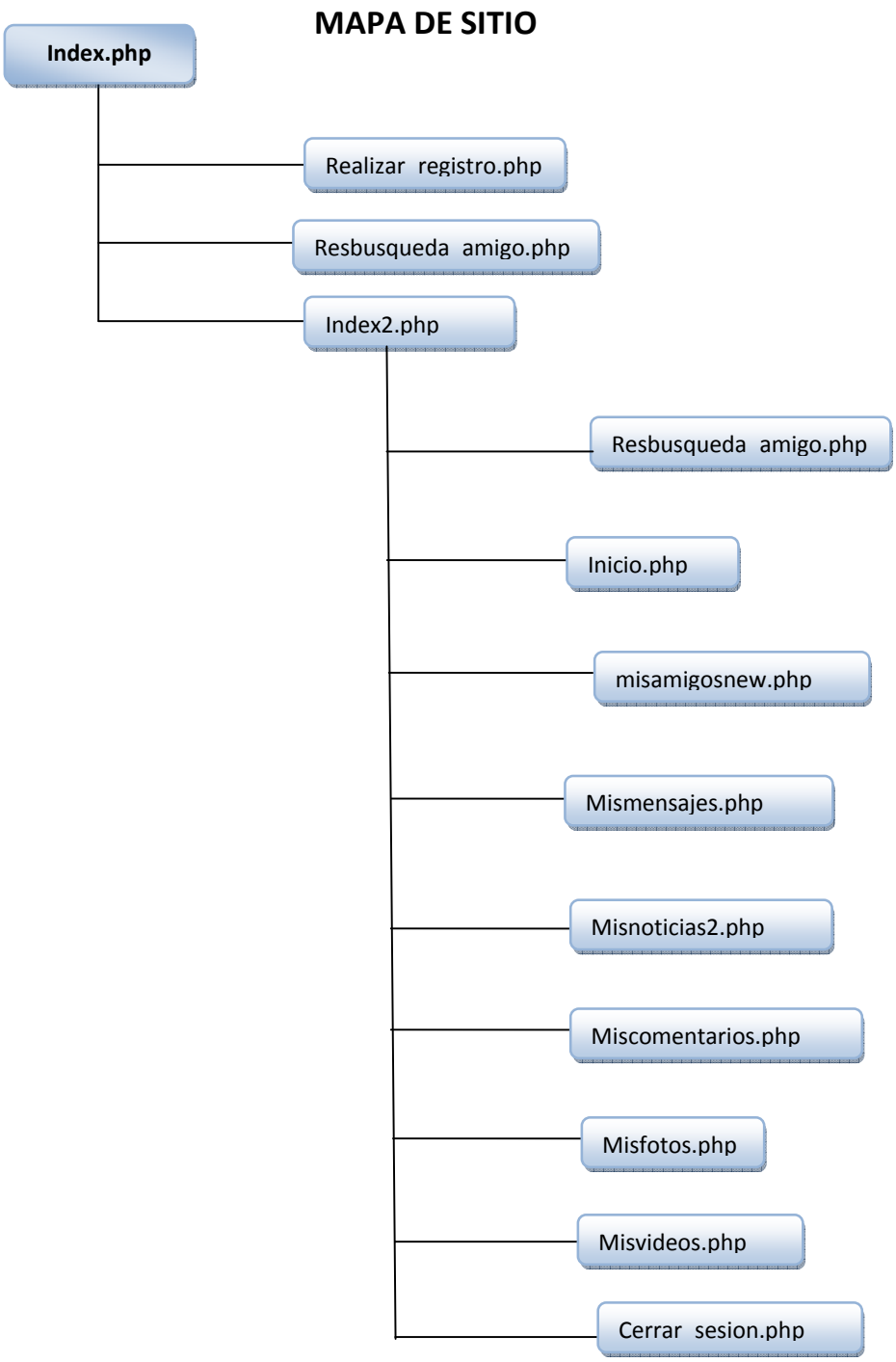


Figura 18 Mapa de sitio

DISEÑO DE INTERFAZ

En el diseño de interfaz no existe diferencia entre los usuarios del portal, por tanto todos los usuarios tendrán la misma interfaz.

Interfaz de usuario al iniciar sesión

Figura 19 Captura de la página principal del portal

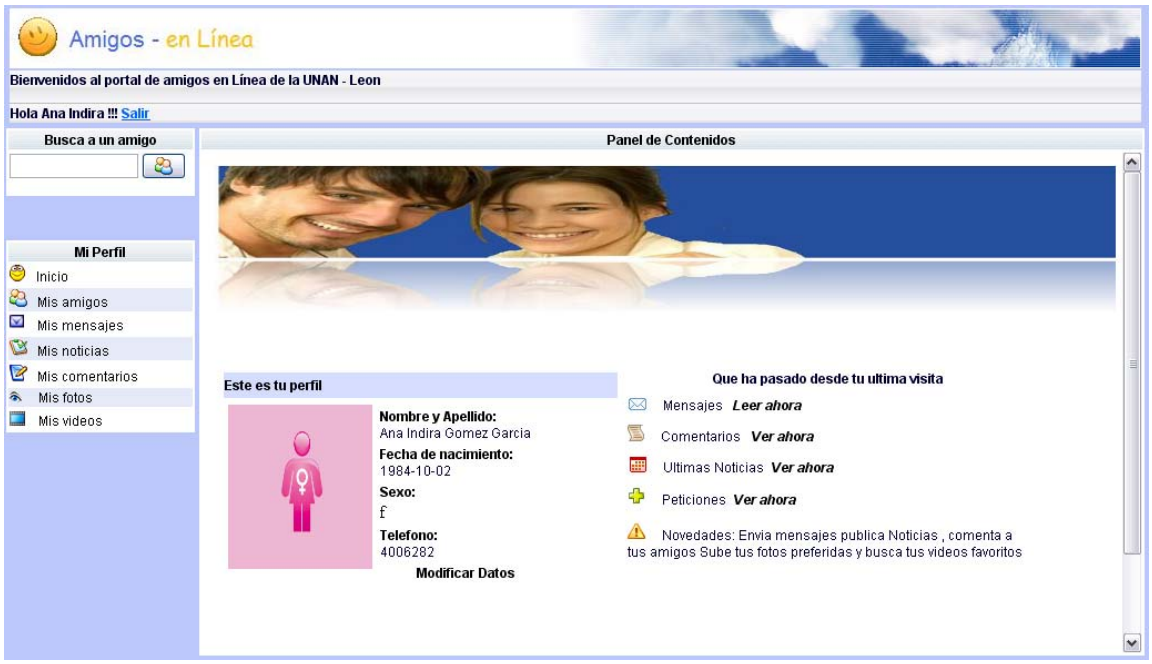


Figura 20 Captura de página de inicio del usuario

Interfaz de mensajes, Bandeja de Entrada

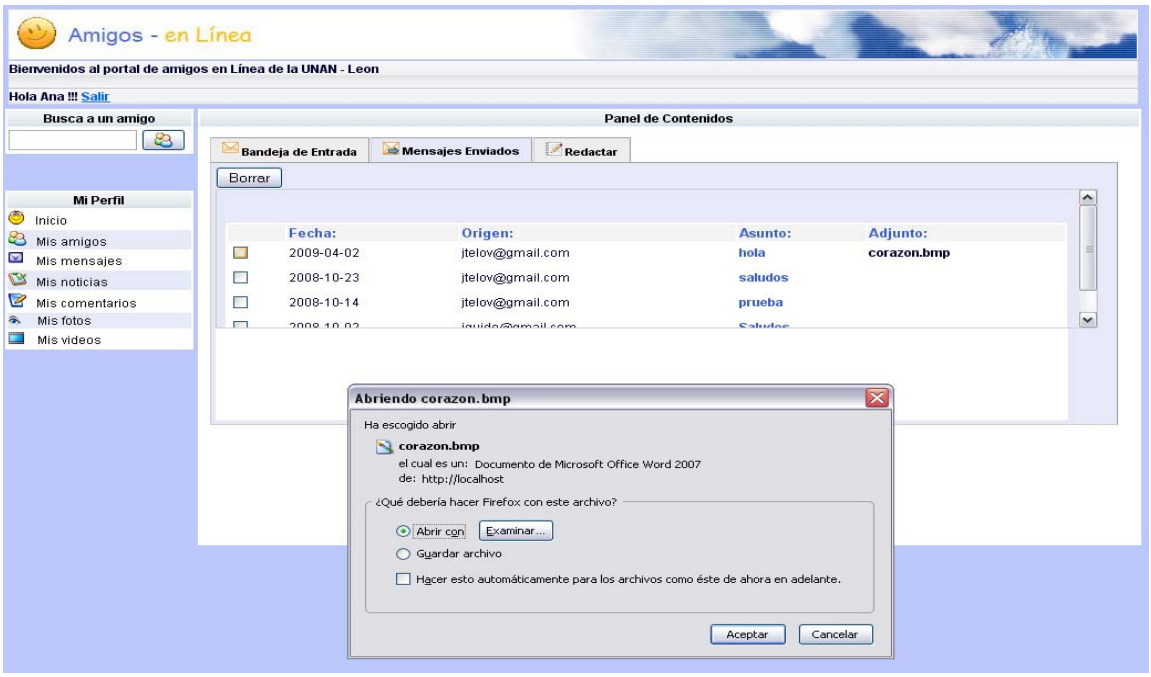


Figura 21 Captura de la bandeja de entrada del usuario

Interfaz de noticias del panel de Control en donde puede publicar y ver noticias publicadas.

Figura 22 Captura del menú Mis Noticias del usuario

Interfaz de la galería de fotos

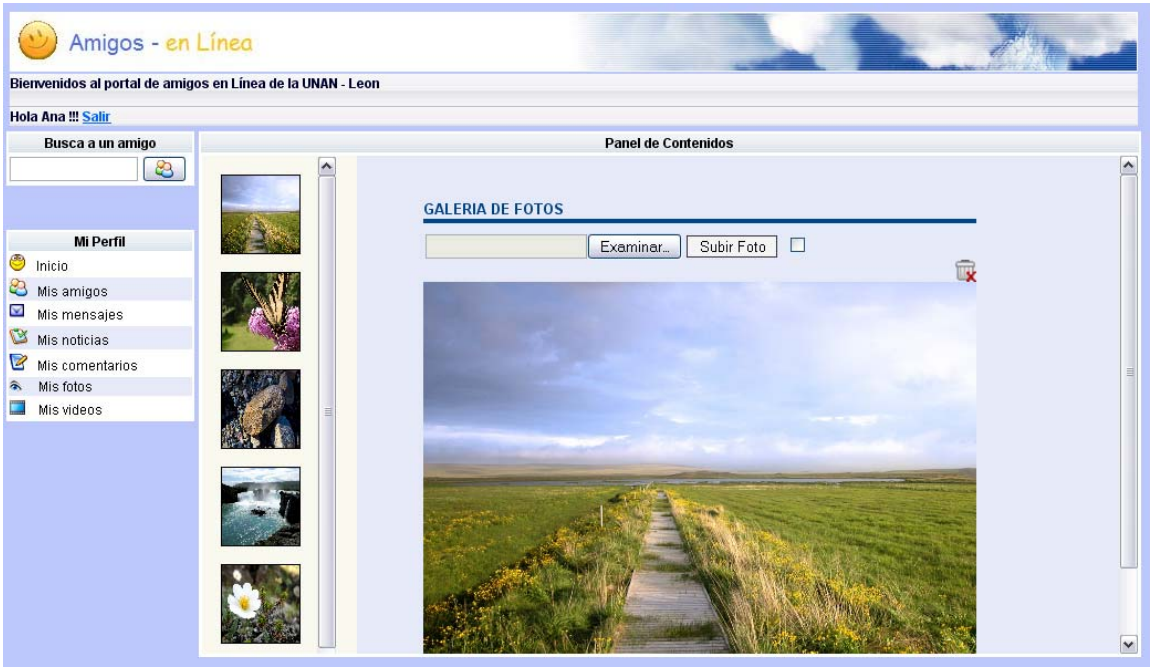


Figura 23 Captura del menú Mis fotos del usuario

Interfaz de videos

Figura 24 Captura del menú Mis videos del usuario

Interfaz de perfil del amigo de un usuario.

Figura 25 Captura de la interfaz del perfil de un amigo

CONCLUSIONES

De la presente investigación se desprenden las siguientes conclusiones:

- El uso de Ajax, permitió dotar a nuestro sitio de una gran interactividad gracias a que las peticiones se realizan de una manera asíncrona lo que permite dar una mejor experiencia al usuario al no tener que recargar la página constantemente.

- El modelo de programación en tres capas nos ayudo a trabajar de una forma más ordenada ya que permitió separar en diferentes niveles lo que era el acceso a los datos, la lógica del negocio y la presentación.

  Con el modelo de programación en tres capas, también se nos facilita el hecho de querer cambiar la base de datos en un futuro, ya que solamente habría que modificar la capa de acceso a datos DAL.

- La utilización de hojas de estilo, facilitó el mantener una misma apariencia en todo el sitio ya que con pocos documentos CSS, se evitó el tener que estar dando formato página por página con lo cual se aceleró el proceso de desarrollo de la capa de aplicación.

- La introducción en nuestro trabajo de los controles Spry proporcionados por Macromedia CS3, facilitó la organización de muchos de los elementos presentados en nuestro portal.

RECOMENDACIONES

- Que el Departamento de Computación incorpore en la enseñanza de los estudiantes, la utilización del modelo de tres de capas para la elaboración de sitios web.

- Invitamos a los estudiantes del Departamento de Computación a la utilización de Ajax en la elaboración de páginas web para lograr una mayor interacción con el usuario.

- También recomendamos a los estudiantes, el uso de las hojas de estilos por la facilidad que proporcionan para mejorar la apariencia de un sitio Web, con lo cual tendremos un código más claro, compacto y sencillo.

- En trabajos próximos podría continuarse mejorando el desarrollo de este portal incluyendo por ejemplo la creación de roles e interfaces diferentes para cada uno de ellos, así también se podrían incluir mecanismo de seguridad en cuanto al contenido que publican los usuarios.

BIBLIOGRAFÍA

- Babin, Lee, Introducción a Ajax con PHP, Edición española: Ediciones Amaya Multimedia (Grupo Amaya S. A), 2007.

Referencias:

- http://es.wikipedia.org/wiki/XAMPP
- http://es.wikipedia.org/wiki/Php
- http://es.wikipedia.org/wiki/Portal_(Internet)
- http://www.aulaclic.es/dreamweavercs3/index.htm
- http://livedocs.adobe.com/es_ES/Dreamweaver/9.0/help.html
- http://www.lawebdelprogramador.com
- http://www.tutores.org/xml
- http://www.forosdelweb.com
- http://www.apachefriends.org/es/xampp.html
- http://www.php.net
- http://html.programacion.net/
- http://aprendeenlinea.udea.edu.co/lms/moodle/mod/resource/view.php
- http://www.webestilo.com/html

ANEXOS

DIAGRAMA DE CLASES (Continuación)

CLASES BASES. Son objetos de negocios que transportan información en cada una de las capas.

Class usuario
-email : var -nombre : var -apellido : var -telefono : var -fechanac : var -sexo : var -clave : var -foto : var
+getEmail() : var +setEmail(entrada $em : var) +getNombre() : var +setNombre(entrada $nom : var) +getApellido() : var +setApellido(entrada $apell : var) +getTelefono() : var +setTelefono(entrada $telf : var) +getFechaNac() : var +setFechaNac(entrada $fech : var) +getSexo() : var +setSexo(entrada $sex : var) +getClave() : var +setClave(entrada $cl : var) +getFoto() : var +setFoto()

Class videos
-email : var -link_video : var -mostrar : var -nombre_video : var -id_video : int
+getEmail() : var +setEmail(entrada $em : var) +getLink_video() : var +setLink_video(entrada $vid : var) +getMostrar() : var +setMostrar(entrada $most : var) +getNombre_video() : var +setNombre_video(entrada $nomb_v : var) +getId_video() +setId_video()

Class fotos
-email : var -ruta : var -nombre_foto : var -tamanyo : var -tipo : var
+getEmail() : var +setEmail(entrada $em : var) +getRuta() : var +setRuta(entrada $rut : var) +getNombre_foto() : var +setNombre_foto(entrada nomb_f : var) +getTamanyo() : var +setTamanyo(entrada $tam : var) +getTipo() : var +setTipo(entrada $tip : var)

Class mensaje
-email : var -asunto : var -destino_origen : var -contenido : var -tipoer : var -fecha : var -adjunto : var -leido : var -id_mensaje : var
+getEmail() : var +setEmail(entrada $em : var) +getAsunto() : var +serAsunto(entrada $asunt : var) +getDestino_origen() : var +setDestino_origen(entrada $dest_orig : var) +getContenido() : var +setContenido(entrada $cont : var) +getTipoer() : var +setTipoer(entrada $ter : var) +getFecha() : var +setFecha(entrada $fech : var) +getAdjunto() : var +setAdjunto(entrada $adj : var) +getLeido() : var +setLeido(entrada $read : var) +getId_mensaje() : var +setId_mensaje(entrada $msj : var)

Class comentario
-email : var -contenido : var -fecha : var -remitente : var -leido : var -id_comentario : var
+getEmail() : var +setEmail(entrada $em : var) +getContenido() : var +setContenido(entrada $cont : var) +getFecha() : var +setFecha(entrada $fech : var) +getRemitente() : var +setRemitente(entrada $remi : var) +getLeido() : var +setLeido(entrada $read : var) +getId_comentario() : var +setId_comentario(entrada $coment : var)

Class fichero
-archivo : var -ruta : var -tipo : var -tamanyo : var
+getArchivo() : var +setArchivo(entrada $name : var) +getRuta() : var +serRuta(entrada $rut : var) +getTi´po() : var +setTipo(entrada $ti : var) +getTamanyo() : var +setTamanyo(entrada $tam : var)

Class noticia
-email : var -noticia : var -fecha : var -asunto : var -id_noticia : var -nombre_foto : var -ruta : var
+getEmail() : var +setEmail(entrada $em : var) +getFecha() : var +setFecha(entrada $fech : var) +getAsunto() : var +setAsunto(entrada $asunt : var) +getId_noticia() : var +setId_noticia(entrada $id : var) +getNoticia() : var +setNoticia(entrada $not : var) +getNombreF() : var +setNombre_F() +getRuta() : var +setRuta()

Class aceptado
-email : var -email_amigo : var
+getEmail() : var +setEmail(entrada $em : var) +getEmail_amigo() : var +setEmail_amigo(entrada $em : var)

Class amigos
-email_mio : var -email_amigo : var
+getEmail_mio() : var +setEmail_mio(entrada $em : var) +getEmail_amigo() : var +setEmail_amigo(entrada em : var)

Class peticion
-email_mio : var -email_peticion : var -er : var -notap : var
+gerEmail_mio() : var +setEmail_mio(entrada $em : var) +getEmail_peticion() : var +setEmail_peticion(entrada $ep : var) +getEr() : var +setEr(entrada $erp : var) +getNotap() : var +setNotap(entrada np : var)

Class conexion
+conectar() : var

CLASES DAL

Son las que se encargan de acceder a los datos.

Class videosDAL
+CopiarMisVideos(entrada $rec : var) +Select_By_Email(entrada $email : var, entrada $nomb : var) +Select_All_Video(entrada $email : var) +Insert_Video(entrada $em : var, entrada $link_v : var, entrada $most : var, entrada $nomb_v : var, entrada $id_v : int) +Delete_Video_ById(entrada $em : var, entrada $id_v : int) +Delete_All_Video() : bool +Update_Nombre_Video(entrada $em : var, entrada $nomb_nuevo : var, entrada $link) +Obtener_UltimoIdVideo(entrada $micorreo : var)

Class peticionesDAL
+CopiarDatosPeticion(entrada $rec : var) +Select_All_Peticiones() : var +Select_All_Enviadas(entrada $em : var) +Select_All_Recibidas(entrada $em : var) +Select_By_Remitente(entrada $remit : var, entrada $em : var) +Select_Enviado(entrada $remit : var, entrada $em : var) +Insert_Peticion(entrada $em_mio : var, entrada $em_peticion : var, entrada $erpet : var, entrada $np : var) : bool +Delete_Pet_ByEmail(entrada $em_mio : var, entrada $em_peticion : var) : bool +Delete_All_Peticiones() : bool

Class fotosDAL
+CopiarMisFotos(entrada $rec : var) +Select_By_Email(entrada $email : var, entrada $nombre : var, entrada $nombre_f : var) +Select_All_Foto(entrada $email : var) +Insert_Foto(entrada $em : var, entrada $rut : var, entrada $most : var, entrada $nomb_f : var, entrada $tamanyo : var, entrada $tipo : var) +Delete_Foto_ByEmail(entrada $em : var, entrada $nombre_f : var) +Delete_All_Foto(entrada $email : var) +Update_Nombre_Foto(entrada $em : var, entrada $nomb_f : var, entrada $nuevo_nombre : var)

Class usarioDAL
+CopiarDatosUsuario(entrada $rec : var) +Select_By_Email(entrada $email : var) +Select_All_Usr() +Insert_Usr(entrada $em : var, entrada $nom : var, entrada $apell : var, entrada $telf : var, entrada $sex : var, entrada $cl : var, entrada $foto : var) +Delete_Usr_byEmail(entrada $em : var) +Delete_All_Usr() : bool +Update_Clave_Usr(entrada $em : var, entrada $cl : var) +Update_Nombre_Usr(entrada $em : var, entrada $nom : var) +Update_Apellido_Usr(entrada $em : var, entrada $apell : var) +Update_Telefono_Usr(entrada $em : var, entrada $tel : var) +Update_FechaNac_Usr(entrada $em : var, entrada $fech : var) +Update_Sexo_Usr(entrada $em : var, entrada $sex : var) +Update_Foto_Usr(entrada $nomb : var, entrada $em : var)

Class mensajesDAL
+CopiarMensaje(entrada $rec : var) : var +Select_By_Id(entrada $id_msj : var, entrada $email : var, entrada $d_o, entrada $e_r) +Select_By_Remitente(entrada $remit : var, entrada $em : var) : var +Select_Enviado(entrada $remit : var, entrada $em : var) : var +Select_All_Enviado(entrada $em : var) : var +Select_All_Recibido(entrada $em : var) : var +Select_All_Mensaje() : var +Update_Leido(entrada $em : var, entrada $leido : var, entrada $id : var, entrada $d_o, entrada $e_r) +Insert_Mensaje(entrada $em : var, entrada $asunt : var, entrada $dest_orig : var, entrada $cont : var, entrada $tipoer : var, entrada $fecha : var, entrada $adj : var, entrada $leido : var, entrada $id_menj : var) +Delete_Mensaje_ById(entrada $id_msj : var, entrada $email : var, entrada $e_r) +Delete_Mensaje_ByEmail(entrada $email : var, entrada $er) +Delete_All_Mensaje() : bool +Obtener_UltimoIdRecibido(entrada $micorreo) +Obtener_UltimoIdEnviado(entrada $micorreo : var) +Obtener_Contenido(entrada $id_msj : var, entrada $email : var, entrada $d_o, entrada $e_r)

Class noticiaDAL
+CopiarNoticia(entrada $rec : var) : var +Select_By_Id(entrada $id_not : var, entrada $email : var) : var +Select_By_Usuario(entrada $em : var) : var +Select_All_Not() : var +Insert_Not(entrada $em : var, entrada $not : var, entrada $fecha : var, entrada $asunt : var, entrada $id : var, entrada $nomb_f : var, entrada $rut : var) +Delete_Not_By_Id(entrada $id : var, entrada $email : var) : bool +Delete_Not_By_Email(entrada $email : var) : bool +Delete_All_Not() : bool +Update_Contenido(entrada $em : var, entrada $cont : var, entrada $id : var) +Select_By_Usuario(entrada $em : var) +Obtener_UltimoIdNoticia(entrada $micorreo : var)

Class ficheroDAL
+Crear_Carpeta(entrada $folder_usr : var, entrada $mode : var) +Borrar_Carpeta(entrada $folder_usr : var) +Borrar_Fichero(entrada $folder_usr : var, entrada $archivo : var) +Subir_Fichero(entrada $archivo : var, entrada $fichero_usr : var) +Subir_Foto(entrada $archivo : var, entrada $fichero_usr : var) +CopiarFichero(entrada $origen : var, entrada $destino : var)

Class aceptadoDAL
+CopiarDatosAceptado(entrada $rec : var) : var +Insert_Aceptacion(entrada $emmio : var, entrada $emamigo : var) : bool +Delete_All_Aceptaciones_By_Email(entrada $email : var) +Delete_Aceptaciones_By_Email(entrada $emmio : var, entrada $emamigo : var) +Select_All_Aceptaciones()

Class comentariosDAL
+CopiarCometario(entrada $rec : var) +Select_By_Id(entrada $id_com : var, entrada $email : var) +Select_By_Remitente(entrada $remit : var) +Select_By_Remitente_Email(entrada $remit : var, entrada $em : var) +Select_All_Coment() +Select_All_Coment_By_Email(entrada $em : var) +Insert_Coment(entrada $em : var, entrada $most : var, entrada $cont : var, entrada $fech : var, entrada $remi : var, entrada $leido : var, entrada $id : var) +Delete_Coment_ById(entrada $id : var, entrada $email : var) : bool +Delete_All_Coment() : bool +Delet_All_Coment_By_Email(entrada $em : var) : bool +Update_Mostrar(entrada $em : var, entrada $mostrar : var, entrada $id : var) : bool +Update_Leido(entrada $em : var, entrada $leido : var, entrada $id : var) : bool +Obtener_UltimoIdComentario(entrada $micorreo)

Class amigosDAL
+CopiarDatosAmigos(entrada $rec : var) +Select_All_Amigos(entrada $email : var) +Insert_Amigo(entrada $em_mio : var, entrada $em_amigo : var) +Delete_Usr_ByEmail(entrada $em_mio : var, entrada $em_amigo : var) +Delete_All_Amigos(entrada $em_mio : var)

CLASES BLL

En ellas están definidos los métodos que dan respuesta al usuario.

Class usuarioBLL
+IngresarPortal(entrada $email, entrada $clave) +RegistrarsePortal(entrada $email, entrada $nomb, entrada $apell, entrada $fech, entrada $sex, entrada $pass) +ModificarDatos(entrada $em, entrada $nombre, entrada $apell, entrada $fecha, entrada $sex, entrada $tell) +ModificarFoto(entrada $archivo, entrada $fichero_usr) +MostrarDatosDeUno(entrada $em) +EliminarUsuario(entrada $em) +BuscarUsuario(entrada $nomb) +traer_usuarios()

class videosBLL
+AgregarVideo(entrada $em, entrada $link_v, entrada $most, entrada $nomb_v, entrada $id_video) +Borrar_Video(entrada $em, entrada $id_v) +MostrarVideos(entrada $em) +MostrarVideo(entrada $em, entrada $id) +CambiarNombre(entrada $em, entrada $nomb_nuevo, entrada $link) +UltimoIdVideo(entrada $usr)

Class peticionBLL
+RealizarPeticion(entrada $em_mio, entrada $em_peticion, entrada $erpet, entrada $np) +VerPeticionesRecibidas(entrada $miemail) +RechazarPeticion(entrada $em_mio, entrada $em_peticion) +AceptarPeticion(entrada $em_acepto, entrada $em_aceptado)

Class mensajeBLL
+EnviarMensaje(entrada $em, entrada $asunt, entrada $dest_orig, entrada $cont, entrada $fecha, entrada $adj, entrada $leido, entrada $idorg, entrada $idenv) +LeerMensaje(entrada $id_mensaje, entrada $email, entrada $d_o, entrada $e_r) +VisualizarMensajeEnviados(entrada $email) +Visualizarmensajes(entrada $email) +BorrarMensaje(entrada $id_msj, entrada $email, entrada $e_r) +BorrarMensajes_Recibidos(entrada $email, entrada $e_r) +Ultimo_IDRecibido(entrada $email, entrada) +Ultimo_IDEnviado(entrada $email) +Obtener_Contenido(entrada $id_msj, entrada $email, entrada $d_o, entrada $e_r)

Class noticiaBLL
+CrearNoticia(entrada $em, entrada $not, entrada $fecha, entrada $asunt, entrada $id, entrada $fot, entrada $rut) +MostrarNoticia(entrada $id_not, entrada $email) +MostrarNoticias(entrada $em) +Ultimo_IDNoticia(entrada $email) +BorrarNoticia(entrada $id, entrada $email) +BorrarNoticias(entrada $email) +CambiarContenido(entrada $email, entrada $cont, entrada $id)

Class fotosBLL
+AgregarFoto(entrada $em, entrada $rut, entrada $most, entrada $archivo) +BorrarFoto(entrada $em, entrada $nombre_f) +SeleccionarFoto(entrada $em, entrada $nombre_f, entrada) +SeleccionarFotos(entrada $em) +BorrarFotos(entrada $em) +ActualizarNombreFoto(entrada $em, entrada $nomb_actual, entrada $nomb_f)

Class amigoBLL
+BorrarAmigo(entrada $em_mio, entrada $em_amigo) +MostarMisAmigos(entrada $email)

Class comentarioBLL
+Agregar_Comentario(entrada $em, entrada $most, entrada $cont, entrada $fech, entrada $remi, entrada $leido, entrada $id) +BorrarComntario(entrada $id_com, entrada $email, entrada) +BorrarComentarios_Recibidos(entrada $email) +MostarUnComentario(entrada $id_com, entrada $email) +MostarTodos_Comentarios(entrada $email) +Ultimo_IDComentario(entrada $email)

Los componentes necesarios para la elaboración de nuestro portal fueron los siguientes:

XAMPP: que es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor Web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo de **X** (para cualquiera de los diferentes sistemas operativos), **A**pache, **M**ySQL, **P**HP, **P**erl. El programa está liberado bajo la licencia GNU y actúa como un servidor Web libre, fácil de usar y capaz de interpretar páginas dinámicas. Actualmente XAMPP está disponible para Microsoft Windows, GNU/Linux, Solaris, y MacOS X.
Instalación de XAMPP para Windows:
Abra el instalador, verá una ventana como la siguiente:

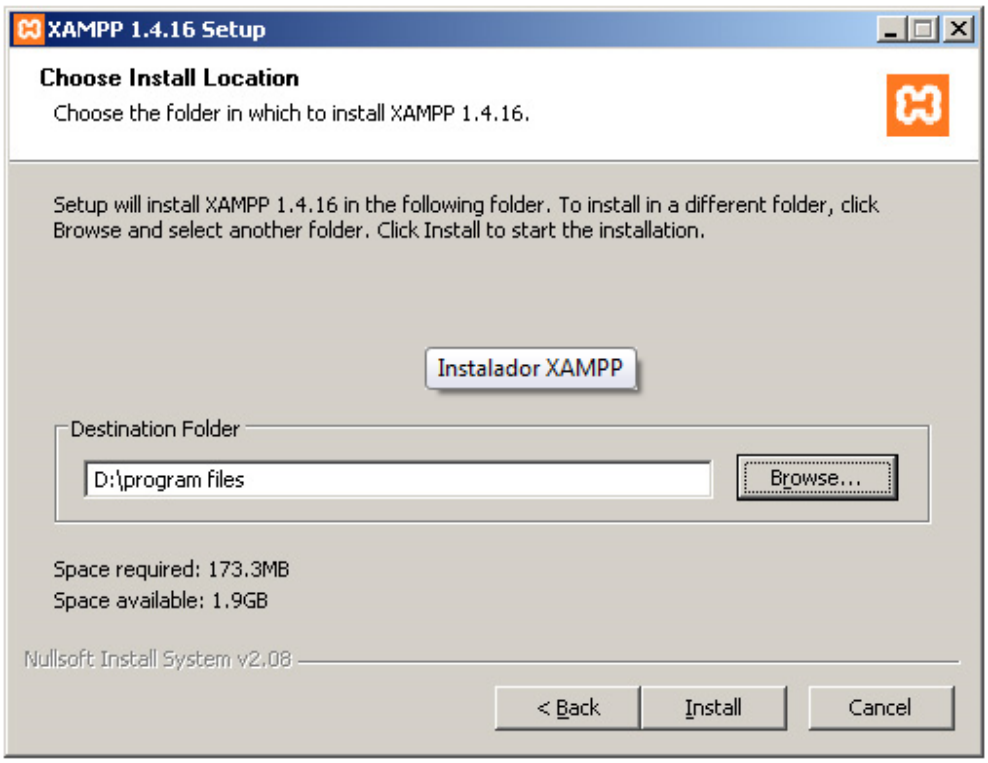


Figura 26 Instalador XAMPP

Seleccione el directorio en el cual desea instalar el programa y de clic en el botón **Install**.

Después de instalar XAMPP podrá encontrarlo en Inicio > Programas > XAMPP. Allí encontrará el Panel de Control de XAMPP, a través del cual podrá iniciar y parar el servidor e instalar y desinstalar servicios.

Ahora comprobaremos que XAMPP fue instalado correctamente:

1. Abra una ventana del navegador.
2. Escriba localhost o 127.0.0.1
3. Verá una página como la siguiente:

Figura 27 Probando XAMPP

4. Seleccione el idioma que prefiera y estará en la página de administración del XAMPP.

Figura 28 Administración XAMPP

Nos vamos al menú Herramientas en donde escogemos phpMyAdmin y se nos presentara la siguiente página y es donde podemos crear nuestra base de datos:

Figura 29 interfaz del modulo phpMyAdmin

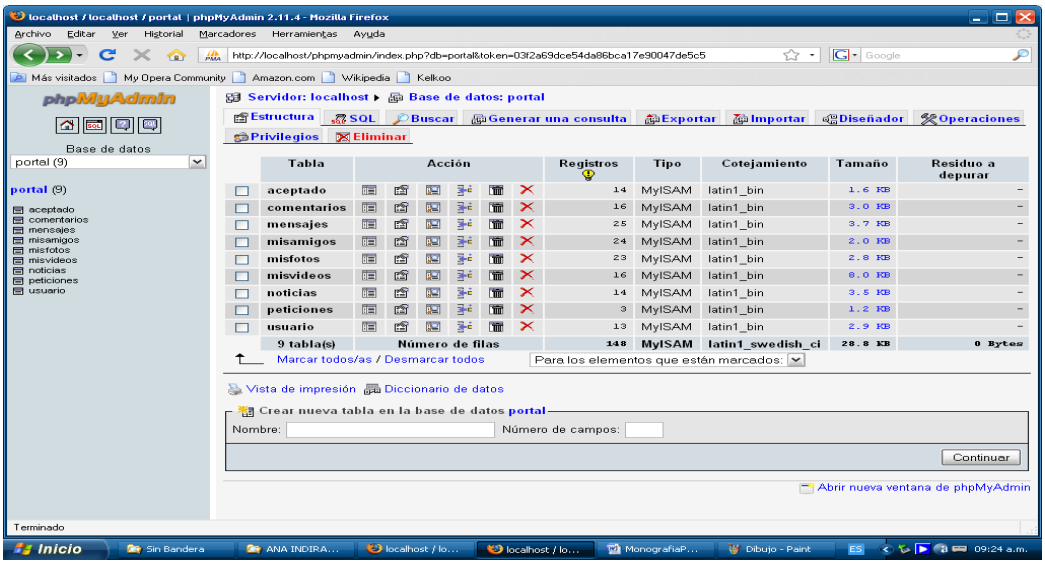


Figura 30 Cuadro de herramienta para administrar una base de datos

En la interfaz anterior es donde podemos editar nuestras bases de datos una vez creada, la seleccionamos y nos vamos a la pestaña estructura y allí creamos las tablas con sus respectivos campos.

En la pestaña privilegios se edita la contraseña, esta misma se modifica en el archivo que se encuentra en la siguiente dirección ruta_instalacion/xampp/phpMyAdmin/config.inc. Ya modificados los privilegios se pasa a trabajar en el código.

Para la elaboración del código se utilizó Dreamweaver CS3 pero también puede utilizarse PHPEdit o cualquier otro programa para desarrollar páginas. Las páginas creadas se guardaran dentro de la carpeta htdocs para ser accedidas desde el servidor, la carpeta está dentro de la carpeta XAMMP que está ubicada según la configuración de la instalación.