2. SERVICIOS PARA APLICACIONES WEB (SERVIDOR)

Materia: Programación para Internet
Alumno: Luis Miguel Navarro Jr
Ciclo: 2021A
Profesor: Dr. José Sandoval Chávez

2.1 Servidor Apache

Existen varios tipos de servidores web en el mercado y entre ellos está Apache, el servidor Apache nació a mediados de los años noventa y es sus mejores años alcanzo una cuota de mercado del 70% de las web, siendo el primer servidor que alojo más de 100 millones de sitios web. En definitiva, Apache es un servidor HTTP que permite servir contenido a las peticiones que vienen desde los clientes web (navegadores).

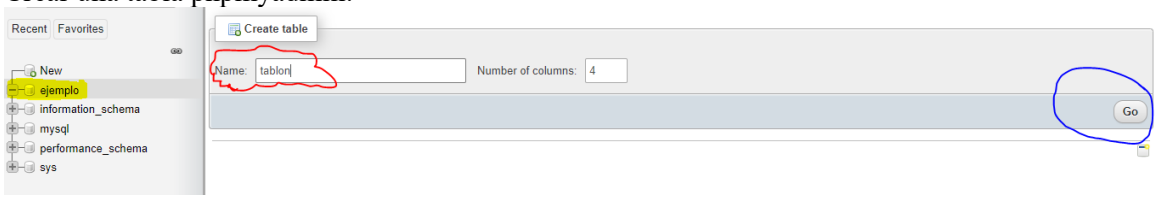
2.2 Accesos y control de bases de datos

Microsoft SQL Server es un sistema de gestión de base de datos relacional, desarrollado por la empresa Microsoft. Dentro de los competidores más destacados de SQL Server están: Oracle, MariaDB, MySQL, PostgreSQL. SQL Server ha estado tradicionalmente disponible solo para sistemas operativos Windows de Microsoft, pero desde 2016 está disponible para GNU/Linux, y a partir de 2017 para Docker también.

Crear una base de datos:

```
CREATE DATABASE ejemplo;
```

Crear una tabla phpmyadmin:



Realizar una sentencia SELECT:

```
mysql> SELECT * FROM ejemplo;
+--------------+-----------+
| nombre       | telefono  |
+--------------+-----------+
| Pedro        | 781323234 |
| Dante        | 781326578 |
| Juan Gabriel | 781321290 |
+--------------+-----------+
```

Realizar sentencia INSERT:

```
mysql> INSERT INTO ejemplo VALUES("Pedro", 781323234);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO ejemplo VALUES("Dante", 781326578);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO ejemplo VALUES("Juan Gabriel", 781321290);
```

Realizar sentencia UPDATE: (comparar al primer SELECT)

```
mysql> UPDATE ejemplo SET telefono = 782327364 WHERE nombre = "Pedro";
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM ejemplo;
+--------------+-----------+
| nombre       | telefono  |
+--------------+-----------+
| Pedro        | 782327364 |
| Dante        | 781326578 |
| Juan Gabriel | 781321290 |
+--------------+-----------+
```

Ejemplos de sentencias JOIN:

```
mysql> SELECT * FROM ejemplo JOIN edad;
+--------------+-----------+--------------+------+
| nombre       | telefono  | nombre       | edad |
+--------------+-----------+--------------+------+
| Pedro        | 782327364 | Pedro        |   32 |
| Dante        | 781326578 | Pedro        |   32 |
| Juan Gabriel | 781321290 | Pedro        |   32 |
| Pedro        | 782327364 | Dante        |   21 |
| Dante        | 781326578 | Dante        |   21 |
| Juan Gabriel | 781321290 | Dante        |   21 |
| Pedro        | 782327364 | Juan Gabriel |   12 |
| Dante        | 781326578 | Juan Gabriel |   12 |
| Juan Gabriel | 781321290 | Juan Gabriel |   12 |
+--------------+-----------+--------------+------+
```

INNER JOIN:

```
mysql> SELECT * FROM ejemplo INNER JOIN edad ON edad.nombre = ejemplo.nombre;
+--------------+-----------+--------------+------+
| nombre       | telefono  | nombre       | edad |
+--------------+-----------+--------------+------+
| Pedro        | 782327364 | Pedro        |   32 |
| Dante        | 781326578 | Dante        |   21 |
| Juan Gabriel | 781321290 | Juan Gabriel |   12 |
+--------------+-----------+--------------+------+
```

Realizar sentencia CREATE DATABASE:

```
mysql> CREATE DATABASE ejemplo;
```

Realizar sentencia CREATE TABLE:

```
mysql> CREATE TABLE ejemplo(
    -> nombre VARCHAR(30),
    -> telefono INT(10)
    -> );
```

2.3 Programación con el lenguaje PHP

1.1 Funcionamiento de un servidor Web

Antes de comenzar a analizar y responder esta pregunta, primero tenemos que hacernos otra pregunta que nos llevará a descubrir cómo funciona un servidor web: ¿Cómo está formada una petición web?

Una petición está formada por la URL del recurso y el tipo de petición (GET/POST/etc) a su vez en el header de la petición se puede indicar la versión del protocolo HTTP a usar, que contenido acepta, que algoritmo compresión admite, seguridad SSL, además se pueden incluir datos de autentificación como usuario, passwords, tokens de autenticación o el formato en que se envían los datos entre muchas otras cosas.

1.2 Historia y versiones

PHP tal y como se conoce hoy en día es en realidad el sucesor de un producto llamado PHP/FI. Creado en 1994 por Rasmus Lerdorf, la primera encarnación de PHP era un conjunto simple de ficheros binarios Common Gateway Interface (CGI) escritos en el lenguaje de programación C. Originalmente utilizado para rastrear visitas de su currículum online, llamó al conjunto de scripts "Personal Home Page Tools", más frecuentemente referenciado como "PHP Tools". Con el paso del tiempo se quiso más funcionalidad, y Rasmus reescribió PHP Tools, produciendo una implementación más grande y rica. Este nuevo modelo fue capaz de interaccionar con bases de datos, y mucho más, proporcionando un entorno de trabajo sobre cuyos usuarios podían desarrollar aplicaciones web dinámicas sencillas tales como libros de visitas. En junio de 1995, Rasmus » publicó el código fuente de PHP Tools, lo que permitió a los desarrolladores usarlo como considerasen apropiado. Esto también permitió -y animó- a los usuarios a proporcionar soluciones a los errores del código, y generalmente a mejorarlo.

PHP 3.0 fue la primera versión que más se parecía al PHP que existe hoy. Encontrando todavía PHP/FI 2.0 ineficiente y falto de las características que necesitaban para impulsar una aplicación de comercio electrónico que estaban desarrollando para un proyecto de universidad, Andi Gutmans y Zeev Suraski, de Tel Aviv, Israel, comenzaron otra nueva versión del analizador subyacente en 1997.

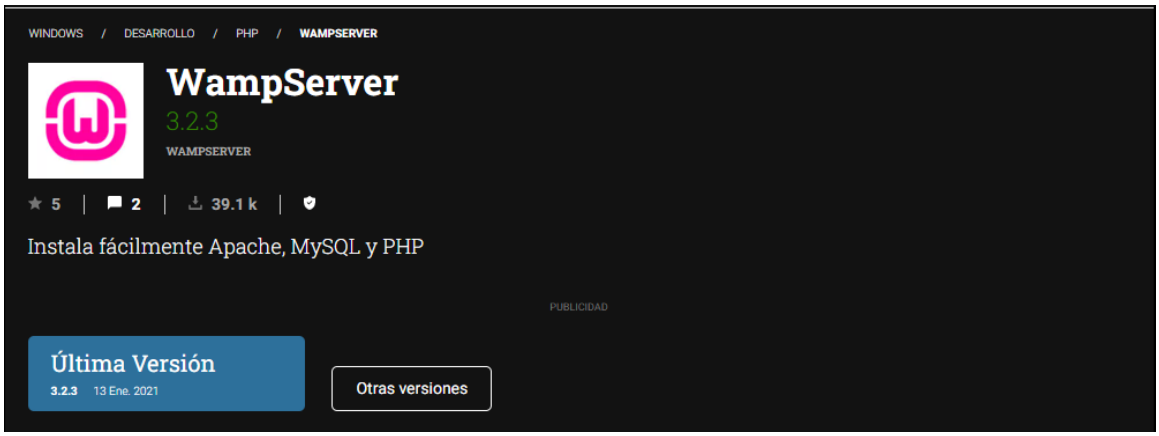
PHP 5 fué lanzado en Julio del 2004 después de un largo desarrollo y varios pre-releases. Está básicamente impulsado por su núcleo, Zend Engine 2.0 que contiene un nuevo modelo de objectos y docenas de nuevas opciones.

1.3 PHP frente a otros lenguajes

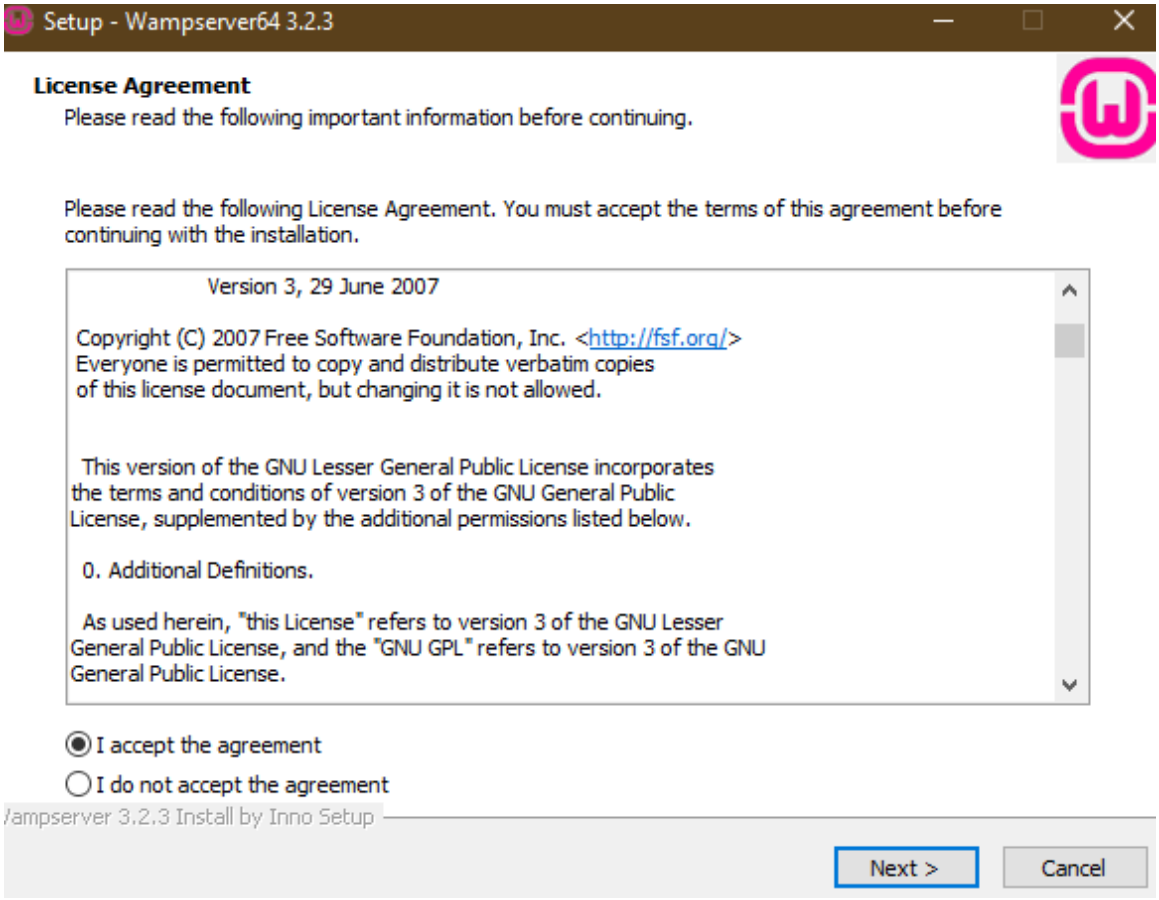
PHP es el lenguaje de programación por excelencia en la web y seguirá siéndolo en el futuro, al menos algunos años más. Una de las razones por las que prácticamente todos los sitios web usan PHP es porque WordPress usa PHP. Desde el lanzamiento de PHP 7, se han introducido muchas nuevas características y mejoras. Las dos mejoras más significativas son la velocidad mejorada y el mejor uso de memoria. Esto significa que las webs con PHP 7 son más escalables, permitiendo manejar más usuarios simultáneos.

1.4 Instalación junto con apache y MySQL (WAMP Server)

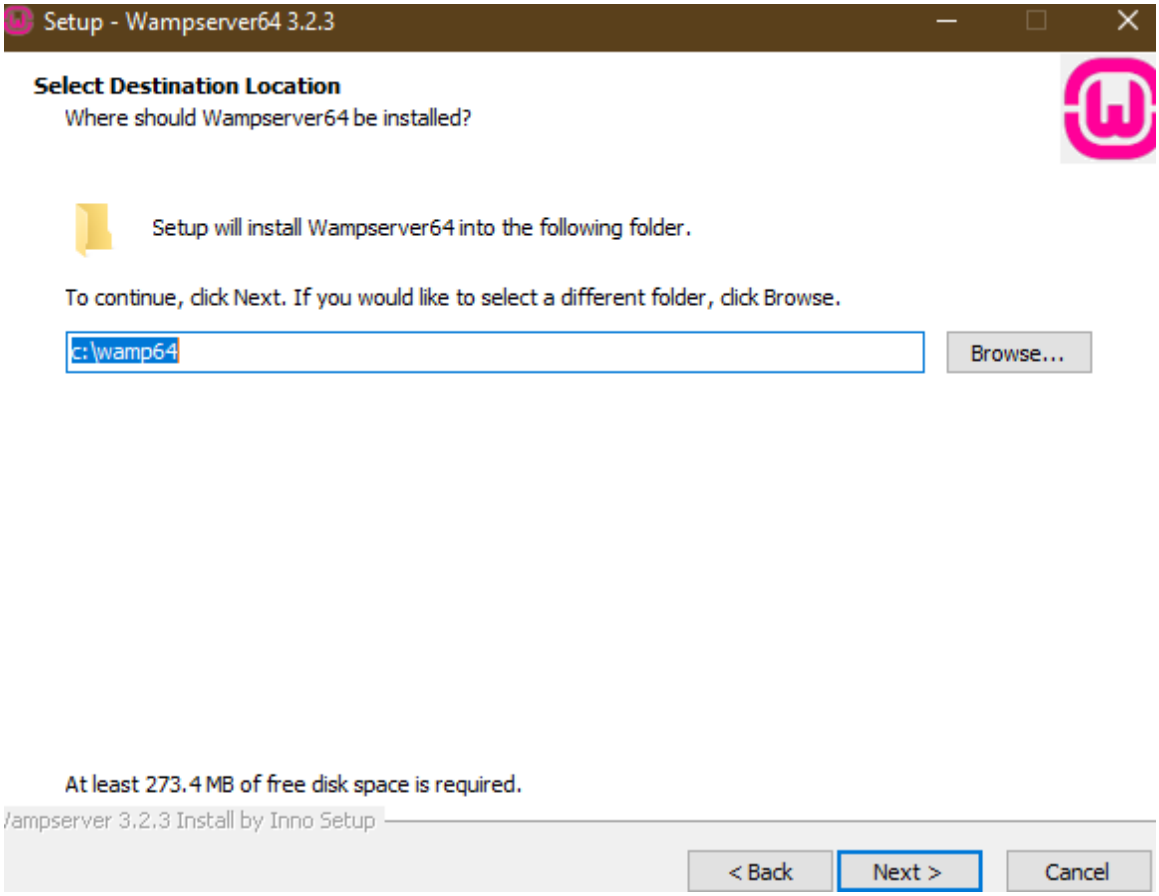
Descargamos el paquete instalador



Aceptamos los términos



Seleccionamos el directorio de instalación.



Acción seguida seleccionamos la opción de instalación, y tendremos un servidor capaz de ejecutar sentencias php y SQL listo en nuestro PC, junto con otras características.

2.1 HTML y PHP (Resumen breve y ejemplo en php de formas de escape)

HTML, siglas en inglés de HyperText Markup Language (‘lenguaje de marcado de hipertexto’), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros

PHP (acrónimo de Hypertext preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web u que puede ser incrustado en HTML

HTML se utiliza para definir los contenidos de una página web, con el tiempo se ha logrado separar las funcionalidades de HTML que aluden al diseño de una pagina web, volviéndolo enteramente un lenguaje para definir contenidos.

PHP por su parte, aunque ha sido arrastrado a un lado por el auge de nuevos lenguajes de programación, sigue en reiterado uso debido a su necesidad para plataformas como wordpress que actualmente domina el mercado del diseño de blogs y desarrollo de plataformas para la administración de contenidos.

Ejemplos:

Para escribir código en HTML basta con colocar la estructura básica de un HTML:

<html> El código dentro de estas etiquetas ser enviado al cliente para la interpretacion
<body> de su código en un software interprete que llamamos: navegador web.

</body>
</html>

Para escribir código en PHP basta con colocar la estructura básica de un script PHP:

<?php //CODIGO AQUÍ ?> Dentro de la etiqueya <?php > se coloca todo el código competente de PHP siendo este interpretado por un servidor externo o local que lleva a cabo toda la lógica y ejecución de las instrucciones de PHP, comúnmente en servidores locales hacemos uso de APACHE con otro tipo de software que nos permiten editar y ejecutar nuestros scripts PHP.

2.2 Comentarios (Ejemplo en PHP)

Para insertar un comentario en PHP se hace de la siguiente manera:

De una línea: //COMENTARIO

De varias líneas:

/* Comentario en una línea

Resto del comentario

fin del comentario */

2.3 Instrucciones o expresiones (Ejemplo en PHP)

En resumen: una instrucción es una línea de código que el interprete ejecuta del lenguaje de programación, ejemplo de expresiones en PHP pueden ser por ejemplo:

Asignar un valor a una variable a = 1; también lo puede ser declarar una clase en programación orientada a objetos

<?php

Class ejemplo{

//blabla

}

Una operación en php puede ser aquello que es referente a la operación aritmética o lógica

Aritméticas:

Suma = a + b;

Resta = a – b;

Multiplicación = a * b;

División = a / b;

Residuo = a % b;

También existen operaciones de evaluación lógica:

AND -> a && b

OR -> a || b

NOT -> ! a

OR (exclusivo) -> a xor b

Asignación lógica (NO regresa un tipo bool): $a = ($fruta ? $fruta : 'Platano');

3.1 Conceptos básicos (Ejemplo en PHP)

PHP es un lenguaje interpretado con tipado dinámico que después de su 5ta versión PHP 5.0 es orientado a objetos, por ello hay ciertas convenciones a tomar en cuenta a la hora de describir código, como conceptos básicos.

Declarar una variable: PHP no requiere que se coloque el tipo de una variable en su declaración, basta con crearla y asignarle un valor inicial de ser deseado:

$mivariable = "valor";

Las instrucciones terminan y se delimitan por el operador ';' que indica el fin de una expresión.

Para que el intérprete de PHP entienda que es un código PHP lo que está recibiendo, es necesaria la expresión: <?php /* CODIGO AQUÍ */ ?>

Para acceder a los atributos y métodos de un objeto se utiliza el operador "->".

3.2 Variables predefinidas (Ejemplo en PHP)

Las variables predefinidas son unos arrays que contienen los valores referentes a un ámbito específico de variables, se puede acceder a ellos a través de como haríamos con cualquier array, pero el índice de acceso en este caso es el nombre de la variable (por lo que hay que conocer la variable), pueden variar según la configuración del servidor donde se corre PHP, están: las variables de servidor y las variables superglobales, aquí unos ejemplos:

Variables de servidor:

$_SERVER["HTTP_USER_AGENT"]: Informacion del visitante y su sistema opertivo

$_SERVER["HTTP_ACCEPT_LANGUAGE"]:Abreviacion del lenguaje que usa el visitante

$_SERVER["HTTP_REFER"]: Desde que URL accedió el visitante

$_SERVER["PHP_SELF"]: Devuelve la URL del script donde viene su ejecucion

$_SERVER[“HTTP_GET_VARS”]: Almacena todas las variables de URL tipo GET

$_SERVER[“HTTP_POST_VARS”]: Almacena todas las varialbes enviadas por formularios POST

$_SERVER[“HTTP_COOKIE_VARS”]: Almacena nombre y contenido de cookies

$_SERVER[“PHP_AUTH_PW”]: Almacena la variable password cuando se accede desde paginas de acceso restringido

$_SERVER[“PHP_AUTH_USER”]: Almcena la variable de usuario cuando se accesa desde paginas de acceso restringido

$_SERVER[“REMOTE_ADDR”]: Muestra la dirección IP del visitante

$_SERVER[“DOCUMENT_ROOT”]: Nos develve el PATH físico en donde se encentra alojada la pagina web.

$_SERVER[“PHPSESSID”]: Almacena el identificador de sesión del usuario

Variables súper globales:

$_GLOBALS, $_SESSION, $_SERVER, $_GET

$_POST, $_COOKIES, $_FILES, $_ENV, $_REQUEST

3.3 Ámbito de las variables (Ejemplo en PHP)

El ámbito de una variable es el contexto dentro del que la variable está definido. La mayor parte de las variables PHP sólo tiene un ámbito simple.

```
<?php
$a = 1;
include 'b.inc';
?>
```

Aquí, la variable $a estará disponible al interior del script incluido b.inc. Sin embargo, al interior de las funciones definidos por el usuario se introduce un ámbito local a la función. Cualquier variable usada dentro de una función, está, por omisión, limitada al ámbito local de la función.

```
<?php
$a = 1; /* ámbito global */

function test()
{
    echo $a; /* referencia a una variable del ámbito local */
}

test();
?>
```

3.4 Existencia y tipo de una variable (Ejemplo en PHP)

Para obtener el tipo de una variable, se utiliza la función “gettype”.

gettype (mixed $var) : string

Devuelve el tipo de la variable PHP, para la comprobación de tipos.

isset (mixed $var , mixed $... = ?) : bool

Devuelve si una variable está definida (existe) y no es null, en forma de bool.

3.5 Variables variables (Ejemplo en PHP)

A veces es conveniente tener nombres de variables variables. Dicho de otro modo, son nombres de variables que se pueden definir y usar dinámicamente. Una variable normal se establece con una sentencia como:

```
<?php
$a = 'hola';
?>

<?php
$$a = 'mundo';
?>
```

Produce el mismo resultado que:

```
<?php
echo "$a $hola";
?>
```

3.6 Constantes (Ejemplo en PHP)

Una constante es un identificador (nombre) para un valor simple. Como el nombre sugiere, este valor no puede variar durante la ejecución del script (a excepción de las constantes mágicas, que en realidad no son constantes). Por defecto, una constante distingue mayúsculas y minúsculas. Por convención, los identificadores de constantes siempre se declaran en mayúsculas.
El nombre de una constante sigue las mismas reglas que cualquier otra etiqueta de PHP. Un nombre de constante válido empieza por una letra o guion bajo, seguido por cualquier número de letras, números o guiones bajos. Usando una expresión regular, se representaría de la siguiente manera: [a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*

```
<?php

// Nombres de constantes correctos
define("FOO",     "something");
define("FOO2",    "something else");
define("FOO_BAR", "something more");

// Nombres de constantes incorrectos
define("2FOO",    "something");

// Esto es válido, pero debe evitarse:
// PHP podría cualquier día proporcionar una constante mágica
// que rompiera el script
define("__FOO__", "something");

?>
```

2.4 LIGA.php y el patrón MVC

Modelo-vista-controlador (MVC) es un patrón de arquitectura de software, que separa los datos y principalmente lo que es la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

El proyecto LIGA presenta LIGA.php 3, es un microframework que proporciona una API para el acceso y control de la base de datos, además es un generador de código HTML a partir de entidades almacenadas, entre sus principales funciones destacan:

- Facilita la creación de la estructura básica de la página web mediante HTML5 Boilerplate
- Ofrece un enrutador fácil de aprender y personalizar
- Genera estructuras HTML válidas a partir de entidades de la base de datos como:
    - Tablas, formularios, selectores, listas, checkboxes, botones radio y más
- Ofrece una API para la manipulación de la base de datos MySQL, MariaDB y Drizzle (más fácil que con SQL)
- Gestión de una caché interna (con APC)

HTML::cabeceras($config);

Genera las etiquetas de cabecera necesarias, basado en HTML5 Boilerplate y requiere el parámetro $config, el cual debe ser un array asociativo con al menos la siguiente

```
$config = array(
 'title'       => 'Título de la página', // Los dos primeros son obligatorios

 'description' => 'Descripción de la página para los motores de búsqueda',

 // Permite añadir etiquetas meta por cada llave y valor creará una
 'meta'   => array('name' => 'content', 'otra' => 'contenido'),

 // Permite generar las etiquetas link necesarias para cargar archivos CSS
 'css'    => array('LIGA.css', 'css/otraCSS.css', 'estilos/etc.css'),

 // Permite cargar archivos Javascript dentro del HEAD de la página
 'js'     => array('index.js', 'js/vendor/modernizr.js', 'js/etc.js'),

 // Añade estilos personalizados dentro de la etiqueta style
 'style'  => 'div.flota { float:left; width: 30% }',

 // Añade código Javascript directamente en el HEAD
 'script' => 'alert("¡Hola!")'
);
```

HTML::cuerpo($config);

$config es obligatorio, debe ser un array asociativo el cual integrará su estructura de árbol en etiquetas DIV, donde el id corresponderá a la llave del array y envolverá a su valor, si este valor es otro array asociativo se llamará recursivamente la misma función, pudiendo idear estructuras sencillas o complejas según la estructura de la página, a continuación un ejemplo de $config:

```
$config = array(
 'contenedor' => array('izq' => '<p>Contenido del DIV izquierdo</p>',
                       'der' => '<p>Contenido del otro DIV (derecho)</p>' ),
 'pie'        => '<p>Derechos reservados, LIGA 2013</p>'
);
```

Con la configuración anterior se genera el siguiente código HTML:

```
<div id="contenedor">
 <div id="izq">
  <p>Contenido del DIV izquierdo</p>
 </div>
 <div id="der">
  <p>Contenido del otro DIV (derecho)</p>
 </div>
</div>
<div id="pie">
 <p>Derechos reservados, LIGA 2013</p>
</div>
```

HTML::tabla($liga, $caption, $cols, $props, $joins, $pie);
Genera una tabla HTML a partir del objeto LIGA proporcionado como primer parámetro, aprovechando por defecto los nombres de todas las columnas y registros encontrados, ejemplo de uso básico:

```
$usuarios = LIGA('usuarios');

HTML::tabla($usuarios);
```

Esto creará una estructura HTML de tabla para ser estilizada mediante CSS, en pruebas iniciales se recomienda cargar el archivo LIGA.css que viene de ejemplo, esto le dará mejor porte a nuestra área de pruebas.

Diagrama de casos de uso.

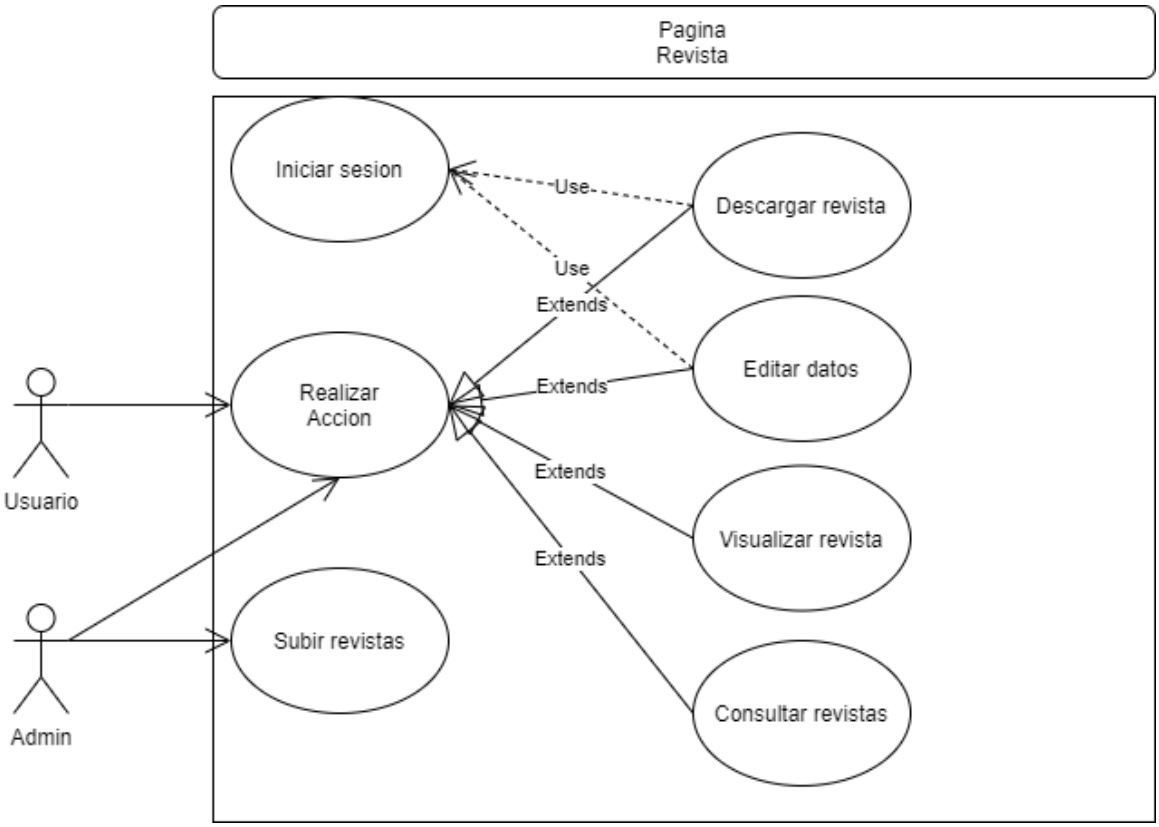
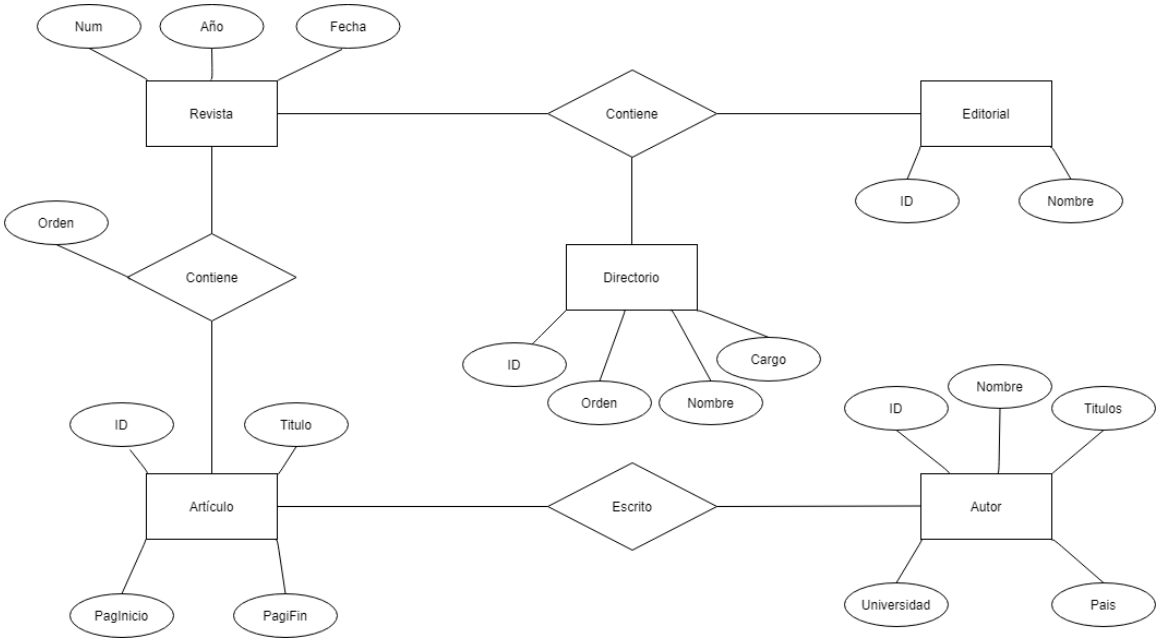


Diagrama entidad-relacion



Analsis de requisitos

Introducción.

Este documento es el análisis de Requisitos para el Prototipo de revista electrónica de la universidad de Guadalajara el cual realizaremos en la clase de Programación para internet. Este análisis se ha estructurado basándose en las sugerencias de mejoras acoradas en clase.

Propósito

Definir y presentar de forma ordenada los requisitos y especificaciones que deberá cumplir el prototipo de revista electrónica, el cual permitirá al usuario

- crear un perfil de usuario
- consultar el contenido disponible
- descargar el contenido disponible
- realizar búsquedas entre el contenido
- calificar y emitir una opinión respecto al contenido

Requerimiento	**1**
Nombre	crear un perfil de usuario
Propósito	Crear un nuevo usuario con diferentes rangos administrativos
Descripción	creando el perfil del usuario, asignar un nivel administrativo para asignar el rol que tendrá dentro del la pagina del prototipo de revista electrónica una vez llenado el formulario con sus campos obligatorios y proceder a finalizar el registro pulsando el botón de “registro”
Entrada	Formulario de registro con asignación de nivel de usuario
Salida	-Mensaje de registro completado -mensaje de bienvenida -rediccionar a la pagina principal
Prioridad	Alta

Requerimiento	**3**
Nombre	Descargar el contenido disponible
Propósito	Tener acceso al contenido de la revista sin necesidad de tener una conexión a internet
Descripción	El usuario podrá descargar el contenido de la página web (Revista) a su dispositivo, lo cual le permitirá realizar la lectura del

Requerimiento	2
Nombre	Consultar el contenido disponible
Propósito	Tener acceso al contenido de las revistas electrónicas
Descripción	Dar una mejor presentación sobre el contenido, Tras haber seleccionado una revista, tener accesos al contenido de dicha revista permitiendo realizar la lectura
Entrada	Selección de la revista mediante un clic
Salida	Contenido de la revista seleccionada
Prioridad	Media
	contenido sin la necesidad de tener una conexión a internet o datos móviles
Entrada	Botón de Descargar
Salida	Documento pdf con el contenido de la revista
Prioridad	Alta

Requerimiento	**4**
Nombre	Realizar búsquedas entre el contenido
Propósito	Encontrar de una manera mas eficiente el contenido deseado
Descripción	El usuario podrá realizar búsquedas sobre el contenido de su preferencia, ya sea por nombre, autor, fecha de publicación, etc.… facilitando la búsqueda de contenido de preferencia
Entrada	Entrada de texto para realizar la búsqueda
Salida	Redirección a la pagina con los resultados de la búsqueda
Prioridad	Alta