


Manual para realizar una galería de
imágenes obtenida de una base de datos.
(MySQL).



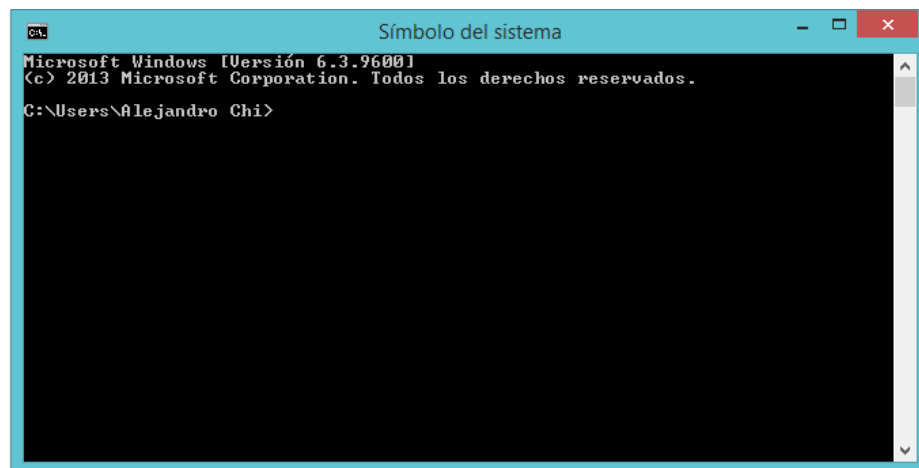
En este manual se mostrara como crear una galería de imágenes utilizando base de datos para el almacenamiento de las los datos. Para ello utilizaremos la base de datos de MySQL y lo que es el Wampserver para la creación de la base de datos.

1. Lo primero que debemos realizar es la creación de la carpeta de nuestra aplicación donde la agregaremos de la raíz donde se instalaron los maquetes de node js.

 GaleriaJS

24/07/2018 01:58 ... Carpeta de archivos

2. De ahí lo que procede es agregarle la utilización de Express para ello abrimos lo que es nuestra CMD línea de comandos para proceder con la creación.



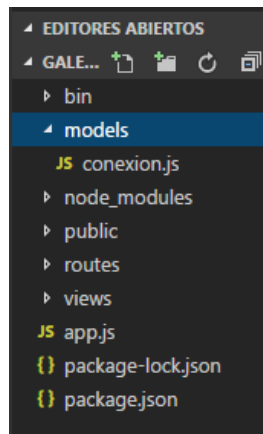
3. En la línea de comandos lo que sigue es buscar la carpeta donde se realizó la instalación de node js y encontrar el nombre de la carpeta como le nombramos GaleriaJs.

```
c:\9C>express --view=e.js GaleriaJs
```

4. Una vez creado lo que es la agregación de lo que es Express procederemos abrir la carpeta utilizando Visual Code.

GaleriaJS - Visual Studio Code

5. Una vez creado lo que es la aplicación con express crearemos un modelo más que se llamara models ahí es donde manipularemos los datos de la base de datos que creamos en un momento.



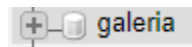
6. En el modelo de models crearemos la conexión con la basen de datos en la cual como se muestra en la imagen.

```
JS conexion.js x
1  var mysql = require('mysql');
2  var connection = mysql.createConnection(
3    {
4      host: 'localhost',
5      port: 3306,
6      user: 'root',
7      password: '',
8      database: 'galeria',
9      insecureAuth: true
10  });
11  connection.connect();
12  module.exports = connection;
```

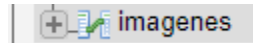
7. De ahí nos va a Wampserver para la creación de base de datos.

The image shows the phpMyAdmin login interface. At the top is the phpMyAdmin logo. Below it, the text 'Bienvenido a phpMyAdmin' is displayed. There is a section for language selection with a dropdown menu currently showing 'Español - Spanish'. Below that is the login section, which includes a button labeled 'Iniciar sesión'. The login form has three fields: 'Usuario' with the value 'root', 'Contraseña' (password field), and 'Elección del servidor' with a dropdown menu showing 'MySQL'. At the bottom right of the login section is a button labeled 'Continuar'.

8. Creamos la base de datos la llamaremos galería.



9. Creamos una tabla llamada imágenes.



10. Le agregamos los siguientes campos.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/>	1	id	int(11)		No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Único Más
<input type="checkbox"/>	2	title	varchar(5000)	latin1_spanish_ci	No	Ninguna			Cambiar Eliminar Primaria Único Más
<input type="checkbox"/>	3	image	varchar(1000)	latin1_spanish_ci	No	Ninguna			Cambiar Eliminar Primaria Único Más

11. Una vez que tengamos la base de datos le agregamos unos registros para que lo podamos visualizar en nuestra práctica.



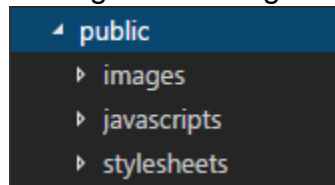
12. Luego nos vamos nuevamente a nuestro visual code para empezar a programar lo que la funcionalidad de nuestra aplicación.

```
<!DOCTYPE html>
<html>
  <head>
    <title><%= title %></title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css" integrity="sha388" />
  </head>
  <body>
    <br>
    <div class="container">
      <div class="row">
        <h1><%= title %></h1>
      </div>
    </div>
    <div class="container">
      <div class="row">
        <p>Mi Galeria de <%= title %></p>
      </div>
    </div>
    <br>
    <div class="container">
      <div class="row">
        <form method="post" enctype="multipart/form-data" >
          <!-- enctype="multipart/form-data" -->
          <div class="row">
            Nombre de la Imagen:
            <div class="col-md-9">
              <input type="text" name="titulo" id="titulo">
            </div>
            <div class="col-xs-4">
              <input type="file" name="archivo" id="archivo">
            </div>
          </div>
        </form>
      </div>
    </div>
  </body>
</html>
```

13. Agregamos el siguiente código que es lo de un formulario simple que nos servirá al momento de saber los datos que están almacenados en la base de datos.

```
<br>
<div class="row">
  <div class="col-xs-4">
    <input type="submit" class="btn btn-primary" value="Agregar Imagen">
  </div>
</div>
</div>
</form>
</div>
</div>
<br>
<div class="container">
  <div class="row">
    <% for(var i=0; i<Galerias.length;i++) { %>
      <div class="col-md-4">
        <div class="card mb-4 box-shadow">
          
        </div>
      </div>
    <% } %>
  </div>
</div>
</body>
</html>
```

Seguimos agregando el siguiente código en la principal en index



Luego creamos una carpeta con el nombre de imágenes donde guardaremos las imágenes almacenadas en la aplicación y previamente guardadas en la base de datos.

14. Una vez teniendo la conexión y lo que es el formulario procederemos a programar lo que es la sección de Router donde agregaremos las consultas a la base de datos.

```
JS index.js x
1  var express = require('express');
2  var router = express.Router();
3  var db= require('../models/conexion')
4
5  var fileupload = require('express-fileupload');
6
7  router.use(fileupload());
8
9  /* GET home page. */
10 router.get('/', function(req, res, next) {
11   db.query("SELECT * FROM imagenes",function(err,results){
12     res.render('index', { title: 'Imágenes', Galerias: results });
13   });
14 });;
```

15. Empezamos con la parte de que al momento de cargar la página principal muestre las imágenes que están almacenadas en la base de datos.

```
router.post('/', function (req, res, next) {
  // console.log(req.body.title);

  var imagenes =
  {
    title: req.body.titulo,
    image: req.files.archivo.name
  }

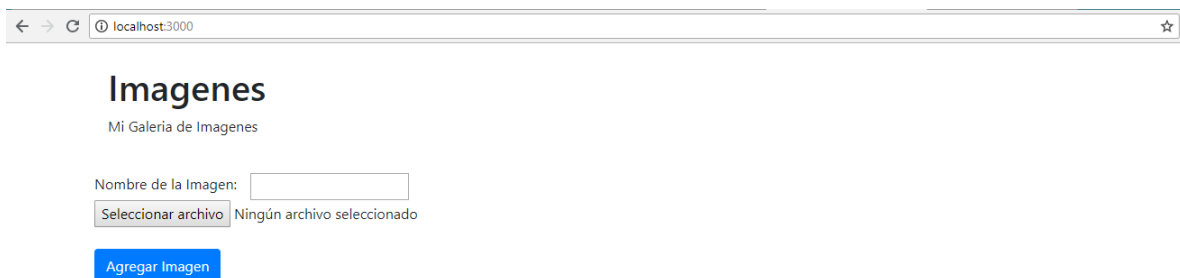
  db.query("INSERT INTO imagenes SET ?", imagenes, function (err, results) {

    res.render('index', { title: 'imagenes' });
    res.redirect('/');
  });

  let archivoASubir = req.files.archivo;

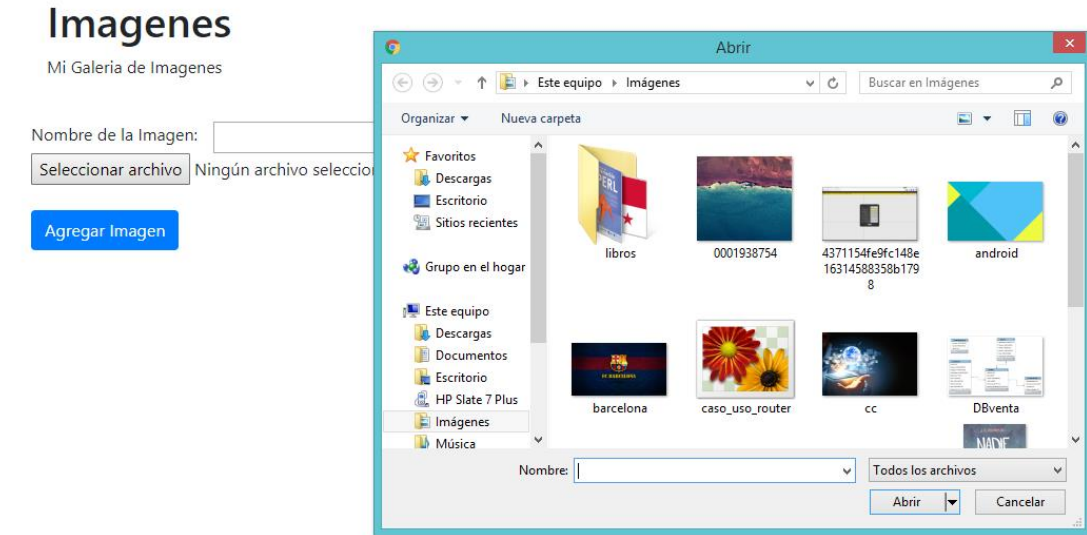
  archivoASubir.mv('public/imagenes/' + req.files.archivo.name, function (err) {
    if (err)
      return res.status(500).send(err);
  });
});
module.exports = router;
```

16. Por ultimo mostramos el resultado de lo que es nuestra aplicación.

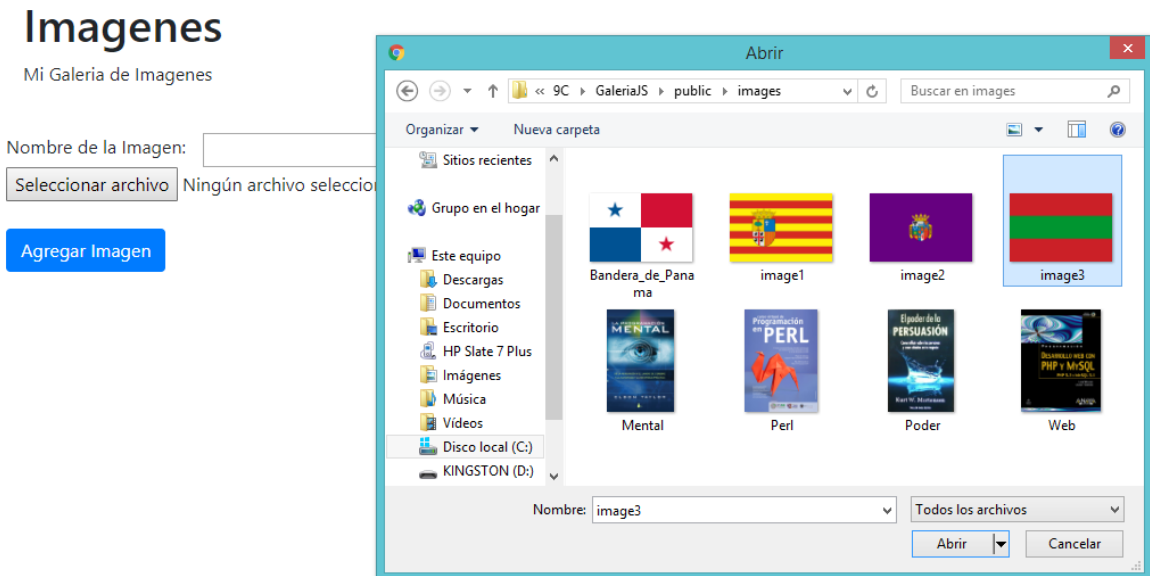


The screenshot shows a web browser window with the address bar set to 'localhost:3000'. The page title is 'Imagenes' and the subtitle is 'Mi Galería de Imagenes'. The form contains a text input for 'Nombre de la Imagen:', a file selection button labeled 'Seleccionar archivo' (which shows 'Ningún archivo seleccionado'), and a blue button labeled 'Agregar Imagen'.

17. Como podemos observar no nos muestra ninguna imagen debido a que no tiene imágenes almacenadas para ellos vamos a agregar imágenes.



Buscamos la imagen que cargaremos en la ventana principal.



En este caso agregamos la imagen que dice 3 y la seleccionamos.

Imagenes

Mi Galeria de Imagenes

Nombre de la Imagen:

image3.jpg

Una vez cargada le agregamos una nombre a la imagen.

Imagenes

Mi Galeria de Imagenes

Nombre de la Imagen:

Ningún archivo seleccionado



Como podemos observar ya se agrega a nuestro formulario.

Imágenes

Mi Galería de Imágenes

Nombre de la Imagen:

image1.jpg



Realizamos lo mismo con una nueva imagen

Imágenes

Mi Galería de Imágenes

Nombre de la Imagen:

Ningún archivo seleccionado



Como podemos ver ya tenemos las imágenes en nuestra aplicación.

18. Y podemos observar en la base de datos que las imágenes fueron guardadas.

+ Opciones

					id	title	image
<input type="checkbox"/>		Editar		Copiar		Borrar	11 Portugal image3.jpg
<input type="checkbox"/>		Editar		Copiar		Borrar	12 España image1.jpg

19. Así es como se realiza una galería de imágenes con la base de datos de MySQL.

