

16-11-2021

ULTIMA
MEMORIAS

Cumplimentación proyecto 1



Contenido

Apartado ANTECEDENTES	3
Apartado REQUISITOS FUNCIONALES	4
Apartado Prototipo WEB y boceto estructura	5
Apartado Guía de estilos	6
Apartado Planificación de tareas	7
Diseño de base de datos	8
• Diseño Entidad Relación de la BBDD	8
• Modelo relacional BBDD	8
• Script de creación BBDD	9
• Consultas	19
Validación de formularios	30
Proceso de carga	31
Jerarquía de directorios	32
Contenido directorios	32
Apartado Diseño de la interface	33
Estructura gráfica de la interface	33
IMPLEMENTACIÓN	34
REQUISITO 1: Diseño responsive.....	34
Funcionamiento	34
Ejemplo de código	34
REQUISITO 2: Para que se pueda acceder a la pagina web	35
Funcionamiento	35
Ejemplo de código	35
PRUEBAS.....	36
Metodología de las pruebas.....	36
DESPLIEGUE.....	37
Apartado Herramientas. Por lo menos enumeración	39
Apartado Lenguajes. Por lo menos enumeración.	39
PRODUCTO	40
Página de Inicio	40
Página iniciar sesión	40
Página de registro.....	41
Página de gestión (tablero)	41

Página de editar difunto.....	42
Página de configuración de perfil.....	42
Abre una ventana emergente de un servicio	43
Página pedidos	43
Apartado Bibliografía. Las que se hayan usado hasta el momento.	44

Apartado ANTECEDENTES

Se trata el proyecto de una página web que presta servicios de mantenimiento de lapidas, panteones y nichos los servicios que presta son oraciones, limpiezas y colocarles flores elegido todo esto al gusto de cliente (es decir tipo de oración según su libro sagrado y cual quiere que se lea y el tipo de flores que quiere que le ponga y la limpieza dependiendo de qué tipo de enterramiento tenga le pondremos un precio u otro es decir más cara o barata por ejemplo no es lo mismo limpiar una lápida que es más pequeña que un panteón). Este proyecto se pensó para melilla que mucha gente vive afuera y se deja su seres queridos aquí claro no va estar todas las veces viajando todo el rato para cuidar donde enterraron sus seres querido ya no solo por tema económico de viajar todo el rato es el tiempo que hoy en día no suele tener mucho tiempo y encima vino una pandemia que con más razón no podían viajar.

Para hacer el diseño se implementó en papel y la base de datos se pasó a página web mediante la página principal es index.html se añadió el diseño que se hizo anteriormente con sus respectivo Bootstrap que se adapta a cualquier dispositivo, su funciones de pasar la imagen y las letras con javascript y con php me permite registrarme e iniciar sesión modificar los registros del difunto de usuario y también eliminar el difunto y dar de baja el usuario por último el usuario puede elegir entre nuestros servicios para aplicárselos a difunto que son oraciones, colocar las flores y limpieza y tenemos estilo con css para los container.

Apartado REQUISITOS FUNCIONALES

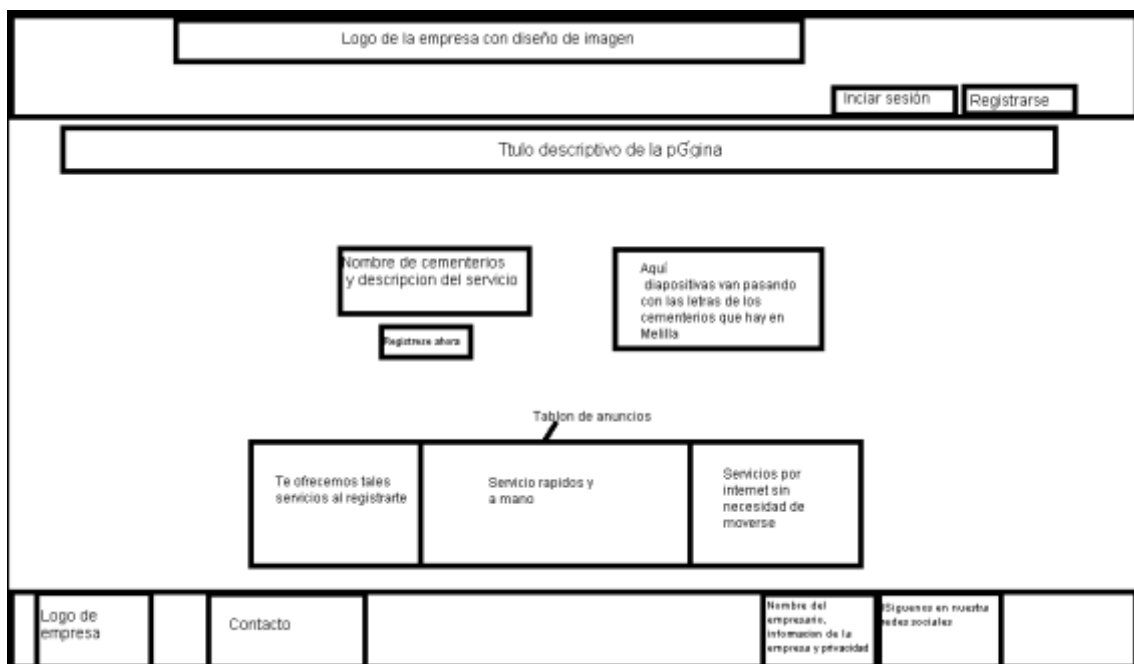
- Permite la autenticación
- Permite modificar y darse de baja los datos del usuario (elegible desde un menú desplegable del nombre del usuario)
- Rellenar, eliminar y modificar los datos del difunto (a través de un botón)
- Rellenar, eliminar y modificar los datos donde está tipo de enterramiento (elegible desde un desplegable)
- Contacta con la empresa(pie de página)
- Cancela el servicio que pidió el usuario (en la página de pedido)
- Implementación del módulo de pagos.

Apartado Prototipo WEB y boceto estructura.

Mapa web



Boceto



Apartado Guía de estilos

El diseño de la página le puso uno normal y corriente la idea no es que sea lo más llamativo que no estas vendiendo nada si no dando una serie de servicios además siendo donde se tiene que prestar dicho servicios es una cosa muy formal opte por un negro y blanco es lo más clásico y formal para una página de esta categoría y en medio enseñando a los cementerios que prestan esos servicios de Melilla. También soporta formato móviles ya que como he nombrado antes está hecho para las personas que no tiene tiempo pueden conectarse a sin en cualquier momento y donde sea sin depender concretamente de un ordenador (se adapta a cualquier dispositivo gracias a lo estilo Bootstrap en caso de que se habrá en un móvil el encabezado automáticamente le saldrá un botón a la derecha con un submenú para poder acceder fácilmente y queda mucho más visual al iniciar sesión o registrarnos. Para que se aprecia que cementerios hay se le puso una diapositiva en medio div he podido realizar y con un javascript llamando la clase de ese de ese dividir la transición por segundo y que reinicie. A bajo un pie de página con etiquetas semánticas y bootstrap que tiene la información del empresario, el logo de la empresa, contacto de la empresa y términos políticos.

Los estilos con los que están funcionando son los container para el tablón de abajo que dice cómo hacemos los servicios de la empresa y el de medio pero tiene también añadido con la clase myCarousel que pasa una serie de diapositivas y descripciones de las imágenes mediante una función de javascript y por ultimo tiene puesto etiquetas semánticas con estilos de clases de bootstrap tipo para el encabezado de barra de navegación es decir poniendo esto ya cambia al estilo que elegí de encabezado `<nav navbar navbar-light navbar-expand-lg navbar-dark></nav>` con etiquetado hacia arriba header y por último el pie de página que le puse el etiquetado footer y lo dividí los datos con estilo bootstrap `col-xs-12 col-md-3` y etiquetado con div este dividí anda metido también en otro div con clase row.

Apartado Planificación de tareas

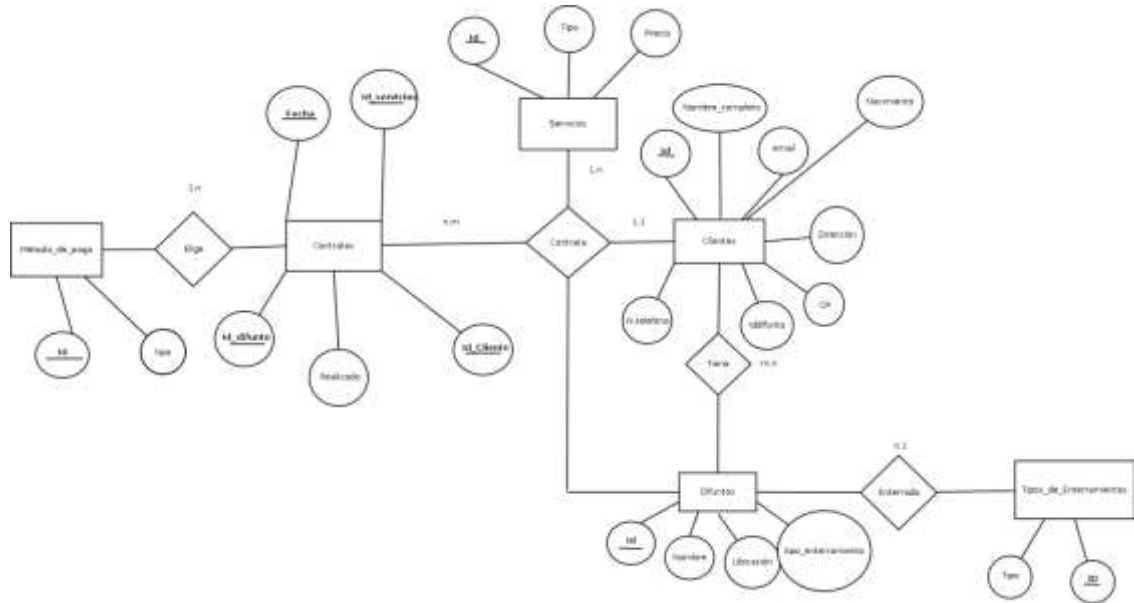
Horas invertidas en total: 245 horas a próximamente de programación e investigación

Días empleados en total: 51 (pero horas aplicadas de cada día 5)

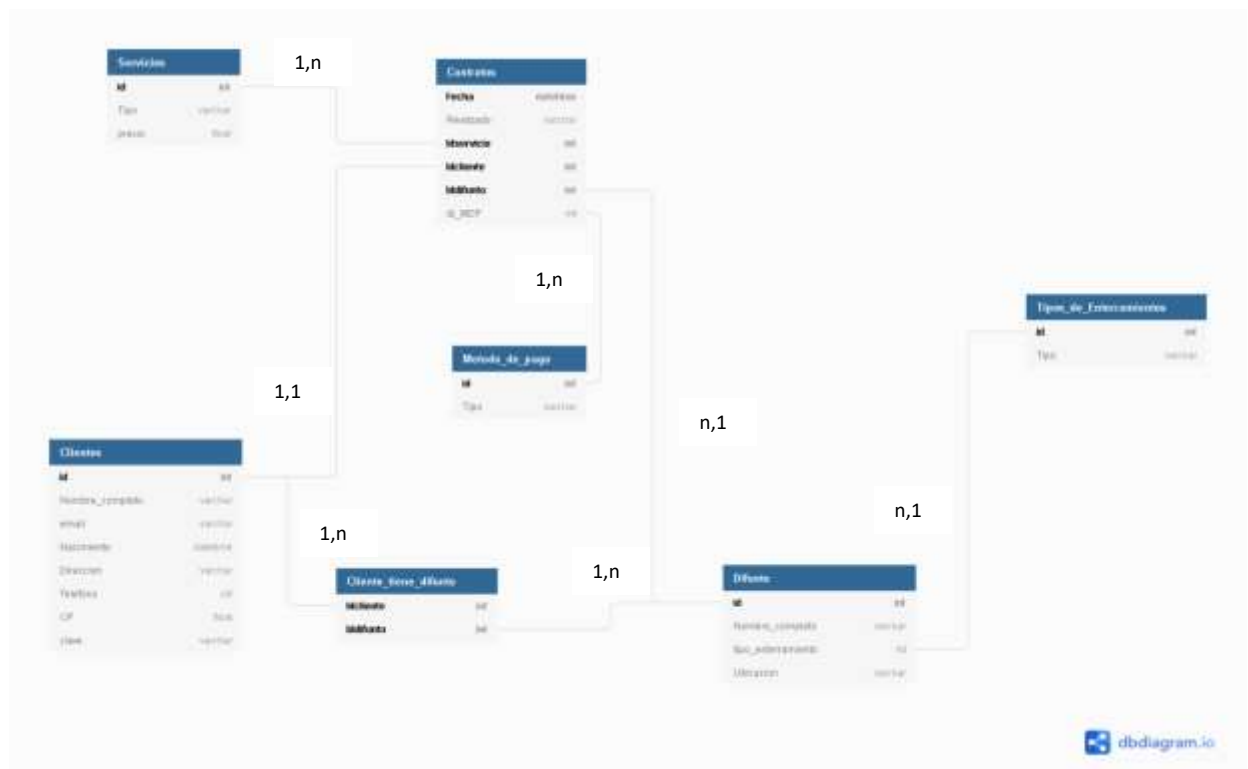
Tarea	Días
Elegir el tema del proyecto.	4
Investigación sobre el tema del proyecto.	7
Tutoriales youtube sobre creación de requisitos funcionales para una página web.	4
Diseño de la base de datos.	1
Diseño de la entidad-relación.	1
Tutorial github.	1
Creación de cuenta en github y creación de proyecto.	1
Creación de logotipo.	1
Creación de banner.	1
Corrección de base de datos.	8
Tutorial sobre tabla intermedia en base de datos.	1
Boceto de la página web.	1
Creación de index.html,login.html y register.html.	2
Creación de bd.php,dashboard.php,registrar.php,eliminar.php,grabar.php, Iniciarsesion.php,logout.phpypedidos.php	18

Diseño de base de datos

- Diseño Entidad Relación de la BBDD



- Modelo relacional BBDD



- **Script de creación BBDD**

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
```

```
START TRANSACTION;
```

```
SET time_zone = "+00:00";
```

```
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
```

```
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
```

```
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
```

```
/*!40101 SET NAMES utf8mb4 */;
```

```
--
```

```
-- Base de datos: `ultimas_memorias`
```

```
--
```

```
-----
```

```
--
```

```
-- Estructura de tabla para la tabla `clientes`
```

```
--
```

```
CREATE TABLE `clientes` (
```

```
  `id` int(11) NOT NULL,
```

```
  `Nombre_completo` varchar(255) DEFAULT NULL,
```

```
  `email` varchar(255) DEFAULT NULL,
```

```
  `Nacimiento` datetime DEFAULT NULL,
```

```
  `Direccion` varchar(255) DEFAULT NULL,
```

```
  `Telefono` int(11) DEFAULT NULL,
```

```
`CP` float DEFAULT NULL,  
`clave` varchar(35) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

--

-- Estructura de tabla para la tabla `cliente_tiene_difunto`

--

```
CREATE TABLE `cliente_tiene_difunto` (  
  `Idcliente` int(11) NOT NULL,  
  `Iddifunto` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

--

-- Estructura de tabla para la tabla `contratos`

--

```
CREATE TABLE `contratos` (  
  `Fecha` datetime NOT NULL DEFAULT current_timestamp(),  
  `Realizado` varchar(255) DEFAULT NULL,  
  `Idservicio` int(11) NOT NULL,  
  `Idcliente` int(11) NOT NULL,  
  `Iddifunto` int(11) NOT NULL,
```

```
`Id_MDP` int(11) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--  
  
-- Estructura de tabla para la tabla `difunto`  
  
--
```

```
CREATE TABLE `difunto` (  
  `id` int(11) NOT NULL,  
  `Nombre_completo` varchar(255) DEFAULT NULL,  
  `Ubicacion` varchar(32) DEFAULT NULL,  
  `tipo_enterramiento` int(11) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--  
  
-- Estructura de tabla para la tabla `metodo_de_pago`  
  
--
```

```
CREATE TABLE `metodo_de_pago` (  
  `Id` int(11) NOT NULL,  
  `Tipo` varchar(255) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--  
  
-- Volcado de datos para la tabla `metodo_de_pago`  
  
--  
  
INSERT INTO `metodo_de_pago` (`Id`, `Tipo`) VALUES  
(1, 'Paypal');
```

```
--  
  
-- Estructura de tabla para la tabla `servicios`  
  
--  
  
CREATE TABLE `servicios` (  
  `id` int(11) NOT NULL,  
  `Tipo` varchar(255) DEFAULT NULL,  
  `precio` float DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--  
  
-- Volcado de datos para la tabla `servicios`  
  
--  
  
INSERT INTO `servicios` (`id`, `Tipo`, `precio`) VALUES  
(1, 'Arreglo floral 1', 5),  
(2, 'Arreglo floral 2', 6),  
(3, 'Arreglo floral 3', 7),
```

```
(4, 'Limpieza 1', 5),  
(5, 'Limpieza 2', 6),  
(6, 'Limpieza 3', 7),  
(7, 'Rezcos Coran', 5),  
(8, 'Rezcos Torah', 5),  
(9, 'Rezcos Biblia', 5);
```

```
-- -----
```

```
--
```

```
-- Estructura de tabla para la tabla `tipos_de_enterramientos`
```

```
--
```

```
CREATE TABLE `tipos_de_enterramientos` (  
  `id` int(11) NOT NULL,  
  `Tipo` varchar(255) DEFAULT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--
```

```
-- Volcado de datos para la tabla `tipos_de_enterramientos`
```

```
--
```

```
INSERT INTO `tipos_de_enterramientos` (`id`, `Tipo`) VALUES  
(1, 'Enterramiento en tierra'),  
(2, 'Nicho'),  
(3, 'Panteón');
```

--

-- Índices para tablas volcadas

--

--

-- Indices de la tabla `clientes`

--

ALTER TABLE `clientes`

ADD PRIMARY KEY (`id`);

--

-- Indices de la tabla `cliente_tiene_difunto`

--

ALTER TABLE `cliente_tiene_difunto`

ADD PRIMARY KEY (`Idcliente`, `Iddifunto`),

ADD KEY `Iddifunto` (`Iddifunto`);

--

-- Indices de la tabla `contratos`

--

ALTER TABLE `contratos`

ADD PRIMARY KEY (`Fecha`, `Idservicio`, `Idcliente`, `Iddifunto`),

ADD KEY `Idcliente` (`Idcliente`),

ADD KEY `Iddifunto` (`Iddifunto`),

ADD KEY `Idservicio` (`Idservicio`),

ADD KEY `Id_MDP` (`Id_MDP`) USING BTREE;

```
--  
  
-- Indices de la tabla `difunto`  
  
--  
  
ALTER TABLE `difunto`  
  
  ADD PRIMARY KEY (`id`),  
  
  ADD KEY `tipo_enterramiento` (`tipo_enterramiento`);
```

```
--  
  
-- Indices de la tabla `metodo_de_pago`  
  
--  
  
ALTER TABLE `metodo_de_pago`  
  
  ADD PRIMARY KEY (`Id`),  
  
  ADD UNIQUE KEY `Id_2` (`Id`),  
  
  ADD KEY `Id` (`Id`);
```

```
--  
  
-- Indices de la tabla `servicios`  
  
--  
  
ALTER TABLE `servicios`  
  
  ADD PRIMARY KEY (`id`);
```

```
--  
  
-- Indices de la tabla `tipos_de_enterramientos`  
  
--  
  
ALTER TABLE `tipos_de_enterramientos`  
  
  ADD PRIMARY KEY (`id`);
```



```
--  
  
-- AUTO_INCREMENT de las tablas volcadas  
  
--  
  
--  
  
-- AUTO_INCREMENT de la tabla `clientes`  
  
--  
  
ALTER TABLE `clientes`  
  
    MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=12;  
  
--  
  
-- AUTO_INCREMENT de la tabla `difunto`  
  
--  
  
ALTER TABLE `difunto`  
  
    MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=19;  
  
--  
  
-- AUTO_INCREMENT de la tabla `metodo_de_pago`  
  
--  
  
ALTER TABLE `metodo_de_pago`  
  
    MODIFY `Id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;  
  
--  
  
-- AUTO_INCREMENT de la tabla `servicios`  
  
--  
  
ALTER TABLE `servicios`  
  
    MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=10;
```

```

--

-- AUTO_INCREMENT de la tabla `tipos_de_enterramientos`

--

ALTER TABLE `tipos_de_enterramientos`

MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;

--

-- Restricciones para tablas volcadas

--

--

--

-- Filtros para la tabla `cliente_tiene_difunto`

--

ALTER TABLE `cliente_tiene_difunto`

ADD CONSTRAINT `cliente_tiene_difunto_ibfk_1` FOREIGN KEY (`Idcliente`) REFERENCES
`clientes` (`id`),

ADD CONSTRAINT `cliente_tiene_difunto_ibfk_2` FOREIGN KEY (`Iddifunto`) REFERENCES
`difunto` (`id`);

--

-- Filtros para la tabla `contratos`

--

ALTER TABLE `contratos`

ADD CONSTRAINT `contratos_ibfk_1` FOREIGN KEY (`Id_MDP`) REFERENCES
`metodo_de_pago` (`Id`),

ADD CONSTRAINT `contratos_ibfk_2` FOREIGN KEY (`Idcliente`) REFERENCES `clientes` (`id`),

ADD CONSTRAINT `contratos_ibfk_3` FOREIGN KEY (`Iddifunto`) REFERENCES `difunto` (`id`),

ADD CONSTRAINT `contratos_ibfk_4` FOREIGN KEY (`Idservicio`) REFERENCES `servicios` (`id`);

```

--

-- Filtros para la tabla `difunto`

--

ALTER TABLE `difunto`

ADD CONSTRAINT `difunto_ibfk_1` FOREIGN KEY (`tipo_enterramiento`) REFERENCES
`tipos_de_enterramientos` (`id`);

COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;

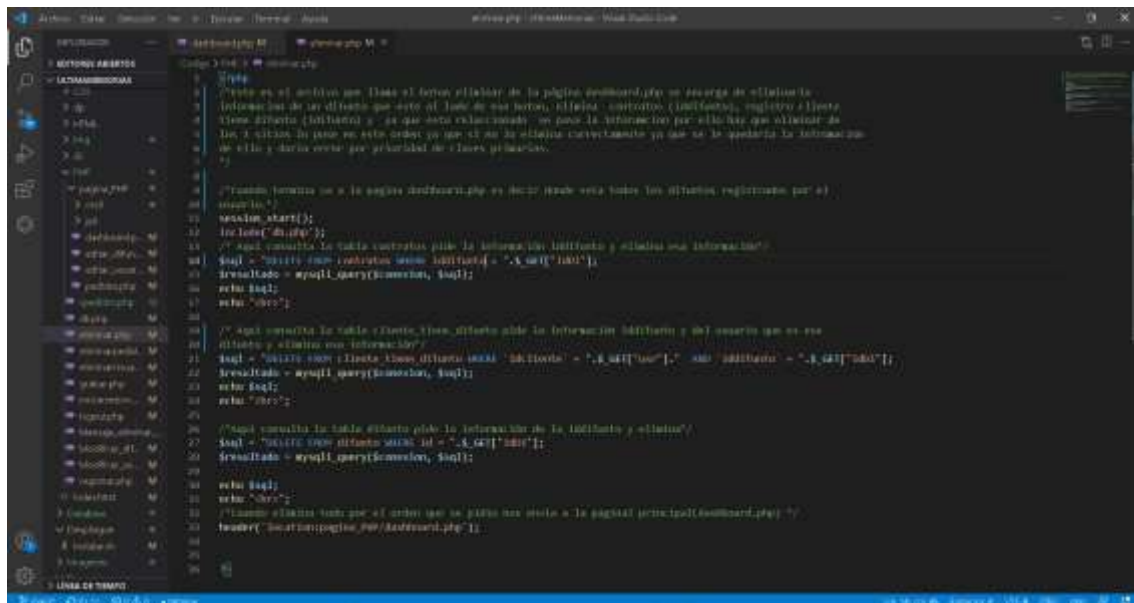
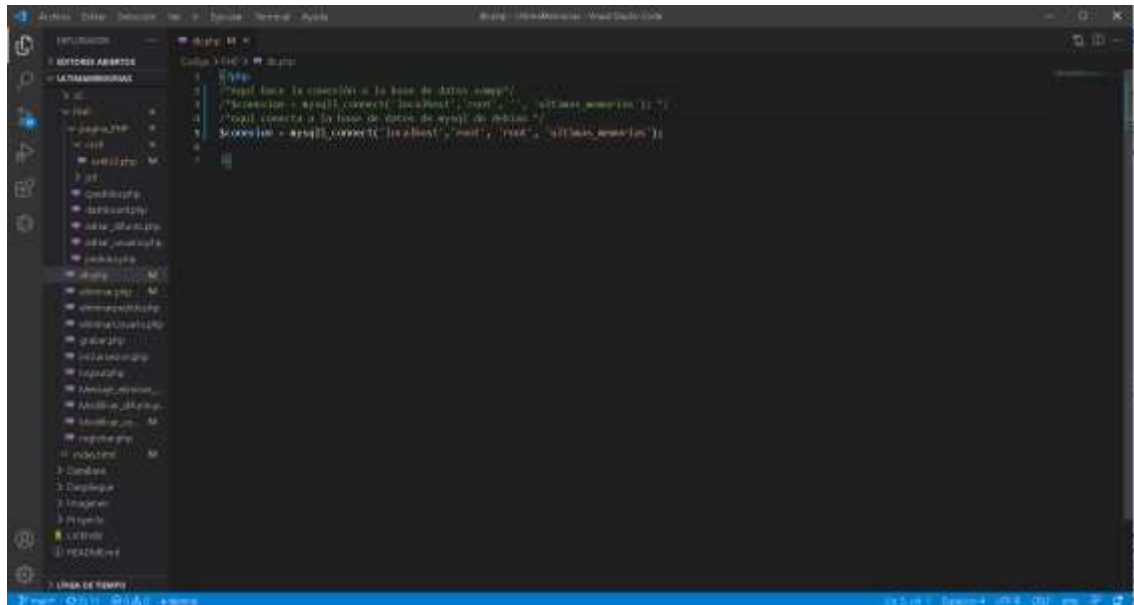
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;

/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

- Consultas

Explico todo y cada una de las consultas y que hacen en los archivos y se puede ver claramente cual es de donde y todo.

Funciones php:



Parte2 Eliminar usuario

```

1  // Eliminar usuario
2  // Eliminar usuario
3  // Eliminar usuario
4  // Eliminar usuario
5  // Eliminar usuario
6  // Eliminar usuario
7  // Eliminar usuario
8  // Eliminar usuario
9  // Eliminar usuario
10 // Eliminar usuario
11 // Eliminar usuario
12 // Eliminar usuario
13 // Eliminar usuario
14 // Eliminar usuario
15 // Eliminar usuario
16 // Eliminar usuario
17 // Eliminar usuario
18 // Eliminar usuario
19 // Eliminar usuario
20 // Eliminar usuario
21 // Eliminar usuario
22 // Eliminar usuario
23 // Eliminar usuario
24 // Eliminar usuario
25 // Eliminar usuario
26 // Eliminar usuario
27 // Eliminar usuario
28 // Eliminar usuario
29 // Eliminar usuario
30 // Eliminar usuario
31 // Eliminar usuario
32 // Eliminar usuario
33 // Eliminar usuario
34 // Eliminar usuario
35 // Eliminar usuario
36 // Eliminar usuario
37 // Eliminar usuario
38 // Eliminar usuario
39 // Eliminar usuario
40 // Eliminar usuario
41 // Eliminar usuario
42 // Eliminar usuario
43 // Eliminar usuario
44 // Eliminar usuario
45 // Eliminar usuario
46 // Eliminar usuario
47 // Eliminar usuario
48 // Eliminar usuario
49 // Eliminar usuario
50 // Eliminar usuario
51 // Eliminar usuario
52 // Eliminar usuario
53 // Eliminar usuario
54 // Eliminar usuario
55 // Eliminar usuario
56 // Eliminar usuario
57 // Eliminar usuario
58 // Eliminar usuario
59 // Eliminar usuario
60 // Eliminar usuario
61 // Eliminar usuario
62 // Eliminar usuario
63 // Eliminar usuario
64 // Eliminar usuario
65 // Eliminar usuario
66 // Eliminar usuario
67 // Eliminar usuario
68 // Eliminar usuario
69 // Eliminar usuario
70 // Eliminar usuario
71 // Eliminar usuario
72 // Eliminar usuario
73 // Eliminar usuario
74 // Eliminar usuario
75 // Eliminar usuario
76 // Eliminar usuario
77 // Eliminar usuario
78 // Eliminar usuario
79 // Eliminar usuario
80 // Eliminar usuario
81 // Eliminar usuario
82 // Eliminar usuario
83 // Eliminar usuario
84 // Eliminar usuario
85 // Eliminar usuario
86 // Eliminar usuario
87 // Eliminar usuario
88 // Eliminar usuario
89 // Eliminar usuario
90 // Eliminar usuario
91 // Eliminar usuario
92 // Eliminar usuario
93 // Eliminar usuario
94 // Eliminar usuario
95 // Eliminar usuario
96 // Eliminar usuario
97 // Eliminar usuario
98 // Eliminar usuario
99 // Eliminar usuario
100 // Eliminar usuario

```

Primera parte Grabar(es muy largo el código de consulta entonces dejare la segunda consulta debajo)

```

1  // Grabar usuario
2  // Grabar usuario
3  // Grabar usuario
4  // Grabar usuario
5  // Grabar usuario
6  // Grabar usuario
7  // Grabar usuario
8  // Grabar usuario
9  // Grabar usuario
10 // Grabar usuario
11 // Grabar usuario
12 // Grabar usuario
13 // Grabar usuario
14 // Grabar usuario
15 // Grabar usuario
16 // Grabar usuario
17 // Grabar usuario
18 // Grabar usuario
19 // Grabar usuario
20 // Grabar usuario
21 // Grabar usuario
22 // Grabar usuario
23 // Grabar usuario
24 // Grabar usuario
25 // Grabar usuario
26 // Grabar usuario
27 // Grabar usuario
28 // Grabar usuario
29 // Grabar usuario
30 // Grabar usuario
31 // Grabar usuario
32 // Grabar usuario
33 // Grabar usuario
34 // Grabar usuario
35 // Grabar usuario
36 // Grabar usuario
37 // Grabar usuario
38 // Grabar usuario
39 // Grabar usuario
40 // Grabar usuario
41 // Grabar usuario
42 // Grabar usuario
43 // Grabar usuario
44 // Grabar usuario
45 // Grabar usuario
46 // Grabar usuario
47 // Grabar usuario
48 // Grabar usuario
49 // Grabar usuario
50 // Grabar usuario
51 // Grabar usuario
52 // Grabar usuario
53 // Grabar usuario
54 // Grabar usuario
55 // Grabar usuario
56 // Grabar usuario
57 // Grabar usuario
58 // Grabar usuario
59 // Grabar usuario
60 // Grabar usuario
61 // Grabar usuario
62 // Grabar usuario
63 // Grabar usuario
64 // Grabar usuario
65 // Grabar usuario
66 // Grabar usuario
67 // Grabar usuario
68 // Grabar usuario
69 // Grabar usuario
70 // Grabar usuario
71 // Grabar usuario
72 // Grabar usuario
73 // Grabar usuario
74 // Grabar usuario
75 // Grabar usuario
76 // Grabar usuario
77 // Grabar usuario
78 // Grabar usuario
79 // Grabar usuario
80 // Grabar usuario
81 // Grabar usuario
82 // Grabar usuario
83 // Grabar usuario
84 // Grabar usuario
85 // Grabar usuario
86 // Grabar usuario
87 // Grabar usuario
88 // Grabar usuario
89 // Grabar usuario
90 // Grabar usuario
91 // Grabar usuario
92 // Grabar usuario
93 // Grabar usuario
94 // Grabar usuario
95 // Grabar usuario
96 // Grabar usuario
97 // Grabar usuario
98 // Grabar usuario
99 // Grabar usuario
100 // Grabar usuario

```

Segunda parte Grabar

```

<code></code>

```

```

1  // Verificar si es un método POST
2  if ($_SERVER["REQUEST_METHOD"] == "POST") {
3      // Recibir los datos del formulario
4      $usuario = $_POST["usuario"];
5      $password = $_POST["password"];
6
7      // Conectar a la base de datos
8      $conn = new mysqli($servername, $username, $password, $dbname);
9
10     // Verificar si la conexión se estableció correctamente
11     if ($conn->connect_error) {
12         die("Conexión fallida: " . $conn->connect_error);
13     }
14
15     // Preparar la consulta SQL
16     $stmt = $conn->prepare("SELECT * FROM usuarios WHERE usuario = ? AND password = ?");
17     $stmt->bind_param("ss", $usuario, $password);
18
19     // Ejecutar la consulta
20     $stmt->execute();
21
22     // Obtener los resultados
23     $result = $stmt->get_result();
24
25     // Verificar si se encontró al usuario
26     if ($result->num_rows == 1) {
27         // Usuario encontrado, devolver los datos
28         $user = $result->fetch_assoc();
29         $response = [
30             "success" => true,
31             "user" => $user
32         ];
33     } else {
34         // Usuario no encontrado, devolver error
35         $response = [
36             "success" => false,
37             "error" => "Usuario o contraseña incorrecta"
38         ];
39     }
40
41     // Devolver la respuesta en formato JSON
42     header('Content-Type: application/json');
43     echo json_encode($response);
44 }
45
46 // Cerrar la conexión
47 $conn->close();

```

[illegible]

Parte1 modificar usuario(es muy grande el código entonces creo la segunda parte debajo en orden)

[illegible]

Parte2 de modificar_usuario

```

1  Archivo Editor Visualizador de Datos Terminal Shell
2  Modulo javascript - NodeJS/Express
3
4  # Modulo: usuarios.js
5
6  // Funcion para obtener los usuarios de la base de datos
7  function getUsers() {
8    // Se crea una consulta SQL para obtener todos los usuarios de la base de datos
9    const sql = "SELECT * FROM usuarios";
10
11    // Se ejecuta la consulta
12    db.query(sql, function (err, results) {
13      if (err) {
14        console.log("Error al obtener los usuarios: " + err);
15      } else {
16        // Se devuelve el resultado de la consulta
17        return results;
18      }
19    });
20  }
21
22  // Funcion para crear un nuevo usuario
23  function createUser(userData) {
24    // Se crea una consulta SQL para insertar un nuevo usuario en la base de datos
25    const sql = "INSERT INTO usuarios (nombre, email, direccion, telefono, fecha_nacimiento, fecha_registro) VALUES (?, ?, ?, ?, ?, ?)";
26
27    // Se ejecuta la consulta
28    db.query(sql, [userData.nombre, userData.email, userData.direccion, userData.telefono, userData.fecha_nacimiento, userData.fecha_registro], function (err, results) {
29      if (err) {
30        console.log("Error al crear el usuario: " + err);
31      } else {
32        // Se devuelve el resultado de la consulta
33        return results;
34      }
35    });
36  }
37
38  // Funcion para actualizar un usuario
39  function updateUser(userData) {
40    // Se crea una consulta SQL para actualizar un usuario en la base de datos
41    const sql = "UPDATE usuarios SET nombre = ?, email = ?, direccion = ?, telefono = ?, fecha_nacimiento = ?, fecha_registro = ? WHERE id = ?";
42
43    // Se ejecuta la consulta
44    db.query(sql, [userData.nombre, userData.email, userData.direccion, userData.telefono, userData.fecha_nacimiento, userData.fecha_registro, userData.id], function (err, results) {
45      if (err) {
46        console.log("Error al actualizar el usuario: " + err);
47      } else {
48        // Se devuelve el resultado de la consulta
49        return results;
50      }
51    });
52  }
53
54  // Funcion para eliminar un usuario
55  function deleteUser(id) {
56    // Se crea una consulta SQL para eliminar un usuario de la base de datos
57    const sql = "DELETE FROM usuarios WHERE id = ?";
58
59    // Se ejecuta la consulta
60    db.query(sql, [id], function (err, results) {
61      if (err) {
62        console.log("Error al eliminar el usuario: " + err);
63      } else {
64        // Se devuelve el resultado de la consulta
65        return results;
66      }
67    });
68  }
69
70  // Se exporta las funciones
71  module.exports = {
72    getUsers,
73    createUser,
74    updateUser,
75    deleteUser
76  };

```

Parte1 registrar (es muy grande el código entonces creo la segunda parte debajo en orden)

[illegible]

Parte2 registrar

```

18 // Seleccionamos el formulario de registro
19 $form = $(document).find('form[id=register]');
20
21 // Obtenemos los datos del formulario
22 $datos = $form.serialize();
23
24 // Enviamos los datos al servidor
25 $.ajax({
26     url: '/index.php?controller=registro',
27     type: 'POST',
28     data: $datos,
29     success: function(response) {
30         // Si la respuesta es exitosa, mostramos un mensaje de éxito
31         if(response === true) {
32             alert("Registro exitoso");
33             // Redirigimos al usuario a la página principal
34             window.location.href = "/index.php";
35         } else {
36             // Si la respuesta no es exitosa, mostramos un mensaje de error
37             alert("Error en el registro");
38             // Mostramos el mensaje de error en la página principal
39             $('#mensaje').html(response);
40         }
41     },
42     error: function() {
43         // Si hay un error en la petición AJAX, mostramos un mensaje de error
44         alert("Error en la petición AJAX");
45     }
46 });

```

[illegible]

Paginas_php:

Parte1 editar_usuario (es muy grande el código entonces creo la segunda parte debajo en orden)

Parte2 editar_usuario

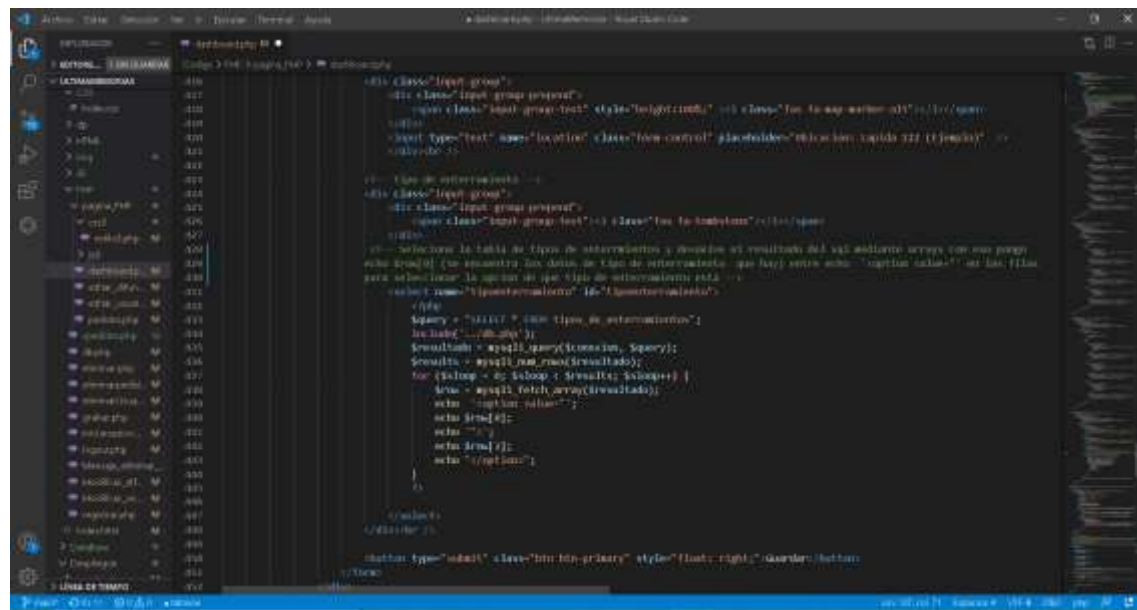
[illegible]


```

1 // app.js
2
3 const url = 'http://localhost:3000/api/users/1';
4
5 fetch(url).then(response => {
6   return response.json();
7 }).then(data => {
8   console.log(data);
9 }).catch(error => {
10  console.log(error);
11 });

```

Parte3 dashboard



Validación de formularios

Tenemos cinco formularios en esta página:

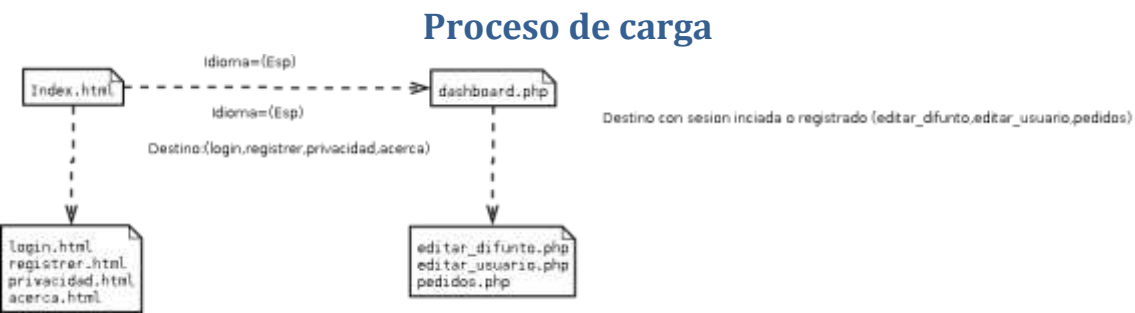
Iniciar sesión-> cuando el usuario le da iniciar con un nombre o contraseña incorrecta le dará un mensaje de error esta parte funciona gracias con esto `getParameterByName` se encuentra en el archivo pone en un archivo js y llama a un parámetro que da un mensaje de alerta que se halla escrito pero claro hay que poner una función de contador (`$count`) claro el contador tiene que ser igual uno significa que si existe hará tal función en mi caso que pueda iniciar y va la página de gestión de difunto del usuario y si no vuelve a la página de iniciar sesión y por ultimo si el usuario se le olvida rellenar el campo los campos tiene dentro este etiquetado para que el campo sea obligatorio `required`.

Registrarse-> cuando el usuario se quiere registrar por primera vez se lo aceptara y se le olvida algún campo se lo pedirá que ese campo es obligatorio de rellenar y si en caso de que ponga un nombre o un gmail que exista le saltara un mensaje de error que están en uso ponga otro para ello utilice `getParameterByName` se encuentra en el archivo pone en un archivo js y llama a un parámetro que da un mensaje de alerta que se halla escrito pero claro hay que poner una función de contador (`$count`) claro el contador tiene que ser igual a 0 significa que si no existe hará tal función es decir registrarme ir a la página de gestión mientras no exista el mismo correo o nombre si existe nos volverá a manda a la página de registro y si se le olvida al usuario rellenar le saltara un mensaje de que requiere rellenar el campo eso se utiliza `required`.

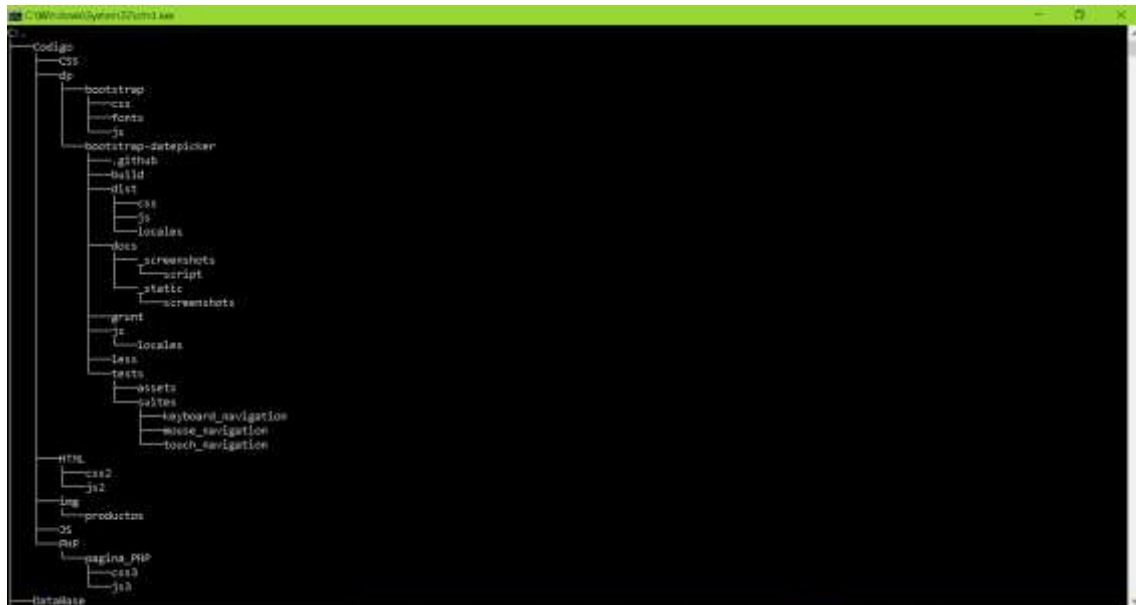
Editar_usuario-> Cuando un usuario quiere editar sus datos Este formulario lo que hace es cuando el usuario quiere cambiar el nombre por correo hay algo en la base de dato que sea igual a esos dos nombres le sale un error de que ya ese mas el nombre estará en uso está hecho con `getParameterByName` esto funciona de la siguiente manera básicamente llama a un nombre y un parámetro que está eso la función el javascript y saltará con una LED alerta pero claro el nombre y el parámetro se llama con un contador que está en PHP que si básicamente consiste de la siguiente manera el contador en este caso será cero porque básicamente si no están esos datos no existe pues del usuario para modificarlo pero si esos datos no existen el contador está uno entonces significa que salta la alerta nombrada anteriormente decir que si existen ya esos datos no dejar al usuario que lo escriba y puse también que los campos son requerido se decir con `required` por si el usuario se olvida o sin querer deja un campo en blanco.

Editar_difunto-> básicamente es lo mismo método que se utilizó en modificar usuario pero con la diferencia de que los nombres que piden son el nombre y la ubicación porque no van a ser los mismos obviamente no deben coincidir eso dos ya que se puede llamar con el mismo nombre pero no puede tener la misma ubicación también ya que eso es imposible para ello he utilizado lo mismo que el para ello se puso el contador `== 0` si no existe el mismo se aplica como bueno sin mensaje de error y lo envía a la página principal si no le dará error. Y los campos no deben estar vacíos los campos.

Registre nuevo difunto-> Este utiliza el mismo sistema de validación y condiciones que editar difunto.



Jerarquía de directorios



Contenido directorios

BD_SQL-> Contiene los archivos de SQL (base de datos) y PDF del cumplimiento de obligatorio del documento del proyecto.

CSS-> Contiene los archivos de estilos CSS de Index.HTML

dp-> contiene todo el paquete bootstrap de datepicker sirve para poner calendarios es un estilo bootstrap

html-> Contiene todos los archivos html y aparte su JavaScript y el CSS de los archivos html y se llaman carpetas que llaman css2(estilo) js2(Javascript).

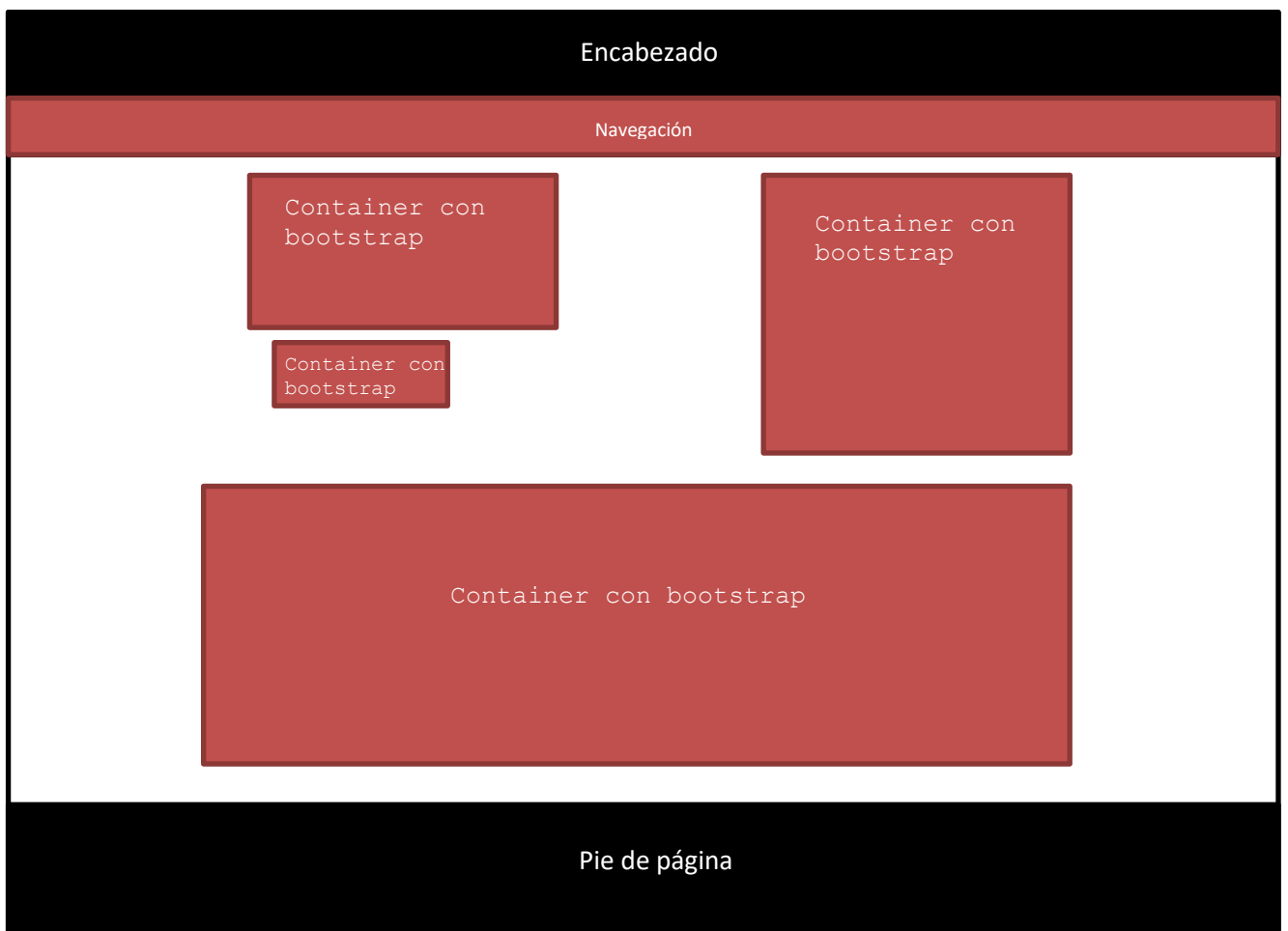
JS-> Contiene el JavaScript del Index .HTML

PHP-> Contiene todos los archivos de función PHP y dentro de esa carpeta tiene una carpeta llamada pagina_PHP en esa carpeta están todas las páginas hechas por extensión PHP y dentro tiene la carpeta CSS3 y JS3 que son dos carpetas que contiene el JS3 contiene los JavaScript de las pagina_PHP que hay ahí y el CSS3 contiene los estilos de la carpeta de las paginas_PHP

Apartado Diseño de la interface

El encabezado contiene el menú de navegación en la parte derecha y en la parte izquierda tiene un el logo de la empresa con sus respectivas decoraciones y todo el menú navegación tenemos básicamente iniciar sesión y registrarse luego más para abajo tenemos tres container hecho **por bootstrap básicamente empezare a describir el** container el primero de la izquierda contiene todas las descripción de las imágenes que está **en el container a la derecha y el otro** que tenemos **abajo del primero** básicamente es el botón de registrarse ya abajo tenemos otro container más grande y más extenso está dividido entre dice lo que ofrece la empresa o promete la empresa hacer que es el servicio **más rápido por ejemplo que** lo hace **desde** el móvil sin moverte de casa y por último tenemos el pie de página que nos permitirá contactar con el empresario de diferentes modos con el gmail eso sí está hecho con maíz alto si su extensión apropiada no podrás hacer una CD también se han pensado para los usuarios que no tengan esa extensión poner gema y completo de la empresa también tenemos número de contacto y el logo de la empresa

Estructura gráfica de la interface



IMPLEMENTACIÓN

REQUISITO 1: Diseño responsive

Ya que mi servicio para las personas que no tiene tiempo es más fácil coger un móvil entonces puse ese diseño con bootraps sin apenas tocar css.

Funcionamiento

Boostraps poniendo sus enlaces correspondiente ya que ponía en la clases el tipo de estilo que se trataba por ejemplo un encabezado lo que conseguí con eso poner todo en un frases todos sus css sin ponerlos y ya cuando era algo muy específico como el menú despegable ese si lo cree mediante css y a su vez bootraps

Ejemplo de código

```
<header>
```

```
  <nav class="navbar navbar-light navbar-expand-lg navbar-dark" style="background-color: black;">
```

```
    
```

```
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
```

```
      <span class="navbar-toggler-icon"></span>
```

```
    </button>
```

```
  <div class="collapse navbar-collapse" id="navbarSupportedContent">
```

```
    <ul class="navbar-nav mr-auto">
```

```
      <li class="navbar-nav" style="text-align: right;">
```

```
        <a style="color:white;" class="nav-link" href="HTML/login.html?error=0">Iniciar Sesion</a>
```

```
        <a style="color:white;" class="nav-link" href="HTML/register.html?error=0">Registrarse</a>
```

```
      </li>
```

```
    </ul>
```

```
  </div>
```

```
</nav>
```

```
</header>
```

REQUISITO 2: Para que se pueda acceder a la pagina web

Claro para que el usuario quiera acceder a la pagina para crear nuevos registros de difunto y hacerlos servicios que desee es necesario obligar que se registre el usuario es decir limitamos la a página de inicio y que no pueda hacer nada al menos que se registre o inicié sesión

Funcionamiento

Para que los datos del difunto que para cada persona es diferente tiene que acceder iniciar sesión o registrarse claro hay puesto en dashboard.php o página principal que no puede acceder sin eso

Ejemplo de código

```
<?php
    session_start();
    $id = $_SESSION["id"];
    $usuario = $_SESSION["usuario"];
    if ($id == 0 or $usuario == "") {
        header('location:../HTML/login.html');  }

?>
```

PRUEBAS

Importe una mv DebianLAMP.ova que está preparada que tiene instalado un S.O debían que tiene instalado php, apache, ssh y mysql antes de subir nada he hecho varias pruebas de que el apache funcionaba, mysql(entrando con el usuario y contraseña dada), apache (entrando en la página de prueba que tiene desde mi anfitrión) y el php (entrando desde el anfitrión a su página de prueba) y utilice ssh con mi anfitrión conectándome a mv y por ultimo comprobé si la contraseña para entrar root del mv era correcta

Metodología de las pruebas

Para probar que mi proyecto es posible que funcione dentro de estos servicios recree dichos despliegue y conexiones para la base de datos y hice una clonación en git hub para poder hacer mediante comando una clonación en mv en debían que se la hice y comprobé que se descargó donde se indicó los archivos nombrados antes comprobé que funcionaba poniendo la dirección IP 192.168.1.73/Codigo que es la ruta en el que se puede conectar a mi proyecto que esta clonado en mi mv y resulto que salio mi pagina y comprobé que hacia cada uno de los requisitos funcionales nombrados(claro mirando si insertaba y todo en mysql) , las validaciones de formularios que se puso en su debido tiempo y por ultimo si el diseño reponsive funcionaba en otros dispositivos como en un móvil copiando la dirección en chrome de mi móvil funciono.

DESPLIEGUE

Explico lo que falto por definir en este despliegue y lo que vi correcto lo que se define en los comentarios que ya venían lo deje porque ya lo explica perfectamente no hace falta que añada mas

```
#!/bin/bash
```

```
#
```

```
# SCRIPT DE DESPLIEGUE DE PROYECTO
```

```
#
```

```
# Natalia Otero García
```

```
#
```

```
# Esto es el nombre del usuario y contraseña del mysql de mv conectara hay para dejar la base  
y lo otro es la url de nuestra mv
```

```
USERDB="debianDB"
```

```
PASSDB="debianDB"
```

```
HOST=$(hostname -I)
```

```
WWW="/var/www/html/"
```

```
# Aquí coge el fichero de la base de datos y como se llama la base de datos
```

```
DATOS="ultimas_memorias.sql"
```

```
BBDD="ultimas_memorias"
```

```
# Se toman los parámetros como USER y PASS de la BBDD
```

```
if [ $# = 2 ];
```

```
then
```

```
    USERDB=$1
```

```
    PASSDB=$2
```

```
fi
```

Copiamos el contenido de la carpeta proyecto a la página html

```
cp -r ../Codigo/ $WWW
```

Creamos esas base de datos y luego restauramos esa base de datos

```
mysqladmin -u $USERDB -p$USERDB create $BBDD
```

```
mysql -u $USERDB -p$USERDB $BBDD < ../DataBase/$DATOS
```

Mostramos url de carga

```
echo "http://$HOST/Codigo/Index.php"
```

Apartado Herramientas. Por lo menos enumeración

1. Visual Studio code versión 1.62.2
2. Xampp versión 3.3.0:
3. Chrome versión95.0.4638.69

Apartado Lenguajes. Por lo menos enumeración.

1. PHP
2. Javascript
3. Html
4. css

PRODUCTO

Muestro aquí todas las páginas que hay en mi página web y describiré brevemente de que archivo pertenece

Página de Inicio

Este pertenece al archivo index.html



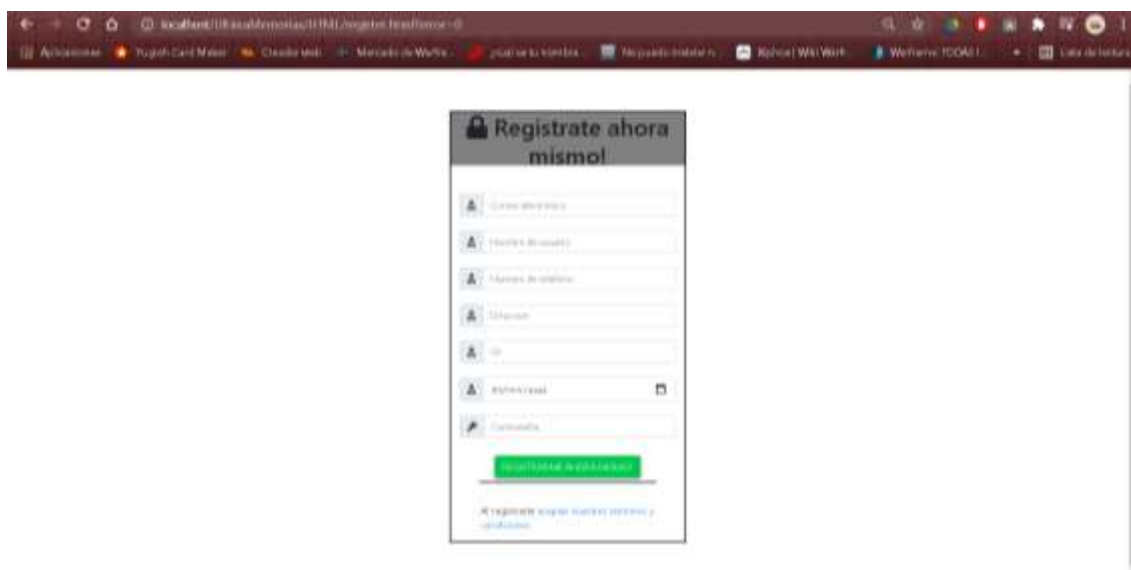
Página iniciar sesión

Este esta en la carpeta html es el archivo login.html entramos cuando le damos inciar sesión



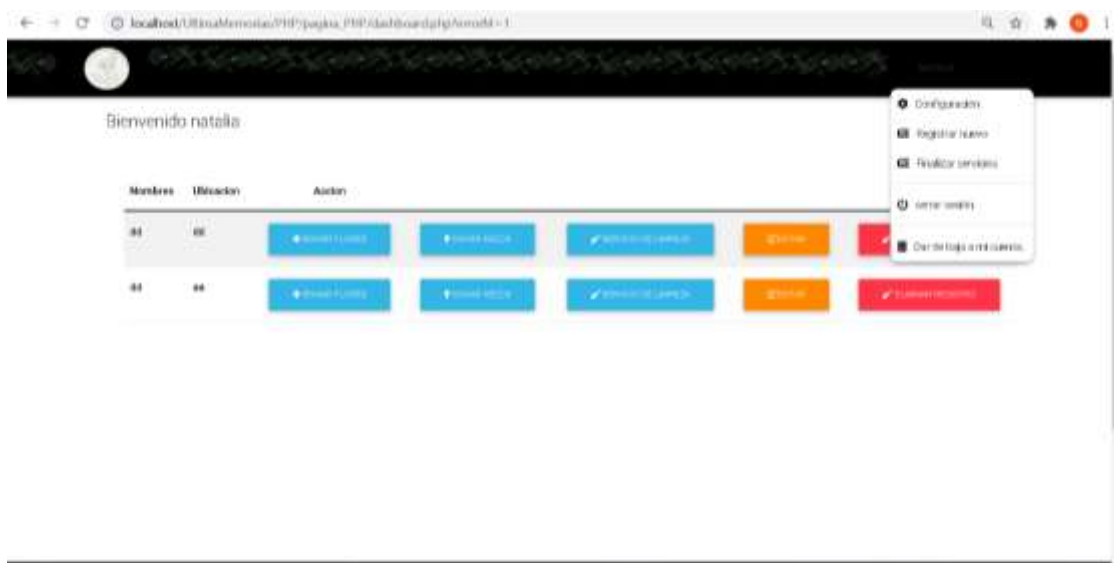
Página de registro

La página de registros se accede de dando al registrarse esto está en la carpeta html llamado en un archivo register.HTML



Página de gestión (tablero)

La página se hace mediante iniciando sesión o registrándose esto se encuentra en la carpeta PHP>pagina_PHP y se llama el archivo dashboard.php aquí podemos darle al menú que está en nuestro nombre podemos registrar un familiar dándole registro nuevo o podemos configurar nuestra cuenta dándole a configuración, tenemos cerrar sesión y podemos darte bajo a nuestra cuenta que eliminará todo lo que tengamos dentro de nuestra cuenta gestionado y a la cuenta misma. A registrar un nuevo difunto podemos enviarle flores, rezos o limpiar que nos saldrá una ventanita modal qué un estilo de bootstrap y le podemos dar el servicio que queremos y pagar y también se puede dar editar que eso lo enseñaré en la siguiente página. Y finalizar servicios es para cuando tengamos un servicio pendiente podemos entrar con contraseña para dar por hecho el servicio en realizado



Página de editar difunto

Aquí que este archivo esta en PHP/pagina_PHP se llama editar_difunto.php cuando quiero editar un difunto le doy editar difunto veo los datos del difunto y los puedo cambiar y guardo los cambios o en caso que no quiera cambiar los datos vuelvo a la pagina dashboard.php



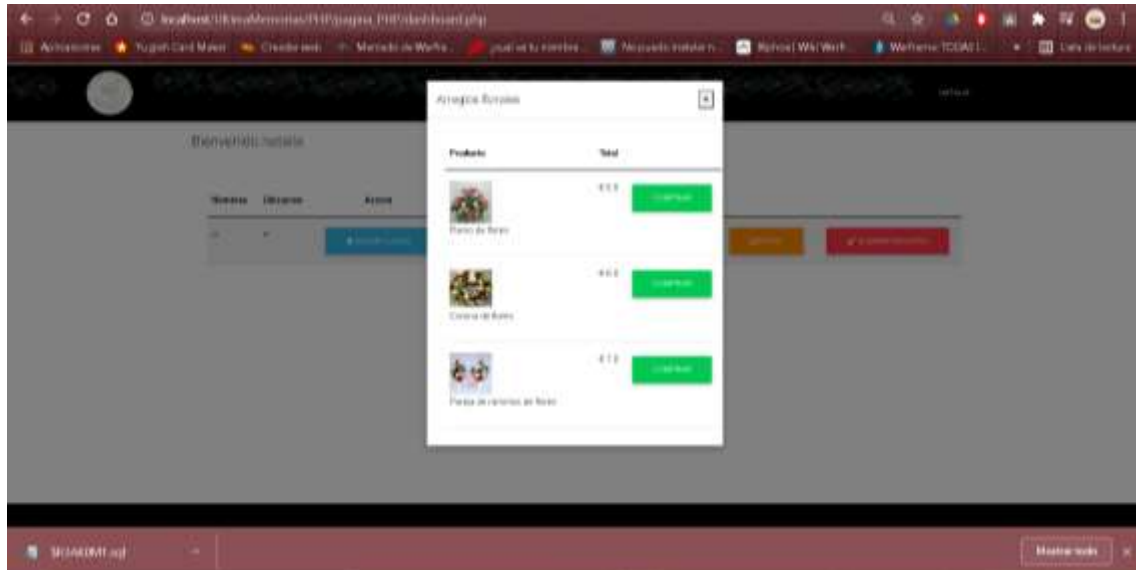
Página de configuración de perfil

Aquí se entra desde dando nuestro nombre y configuración para entrar a editar un usuario que estemos iniciado para poder cambiar nuestros datos y ver los que pusimos si cambiamos algunos datos le damos guardar cambios y si no queremos cambiar nada le damos volver a la pagina principal es decir a dashboard



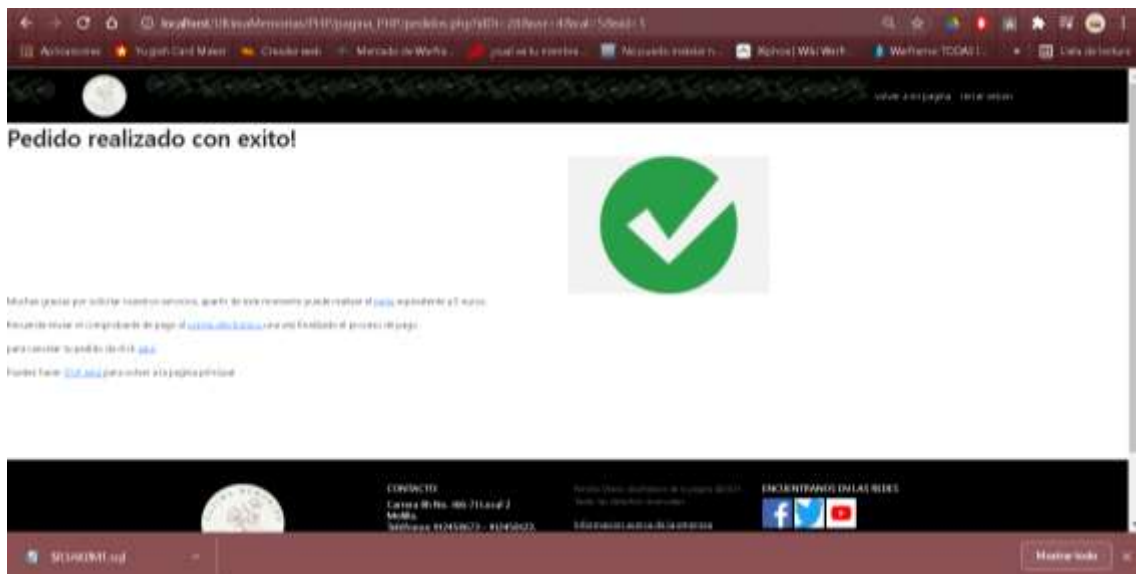
Abre una ventana emergente de un servicio

Cuando le damos un botón de servicio de flores, limpieza y oraciones nos saldrá esto eligimos que queremos comprar o pagar y luego nos llevara a una página que está en la siguiente imagen de pedios.



Página pedidos

Cuando le damos algo de la anterior imagen nos lleva a esta parte para realizar el pago o le podemos dar cancelar o volver a otra vez a dashboard.



Apartado Bibliografía. Las que se hayan usado hasta el momento.

<https://www.youtube.com/watch?v=Lv7XbZtnQ6A>

<https://youtu.be/bO18omSzeR4>

https://youtu.be/L_IWQZNhN7w

<https://youtu.be/hWglK8nWh60>

<https://youtu.be/GG4mftkQVrs>

<https://youtu.be/tFr0Vg1q9Eg>

<https://getbootstrap.com/docs/5.0/getting-started/introduction/>

<https://www.w3schools.com/php/>

<https://www.php.net/manual/es/language.functions.php>

<https://getbootstrap.com/>