



Proyecto de Votación

Oswaldo Mella Toro

osv.mella96@gmail.com

965709247

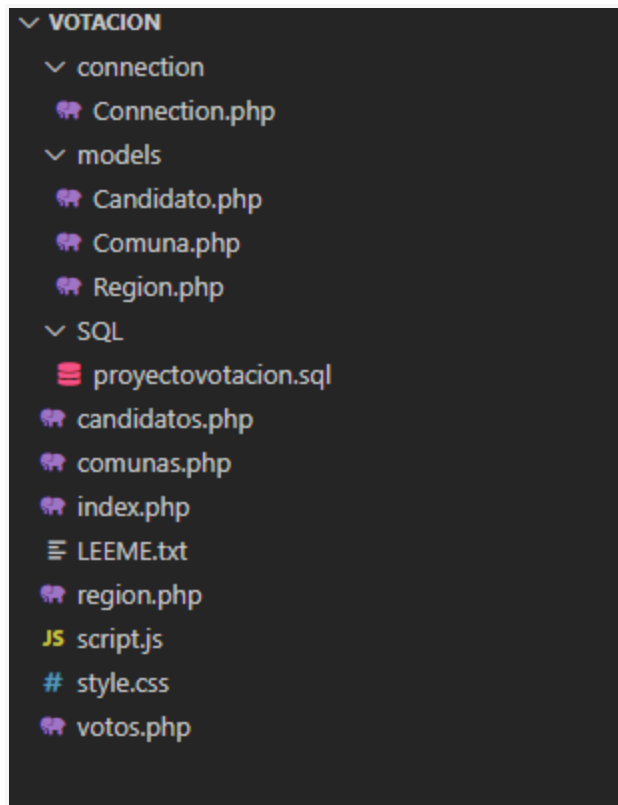
Pirque, El principal 142A

Descripción general

Proyecto generado como prueba para el puesto de desarrollador web, se creó a partir de PHP , HTML, JS, JQuery, MySQL.

Para MySQL y Php se utilizo Xampp en su versión 8.2.0.

Estructura



Especificaciones

A continuación se mostraran los fragmentos de códigos explicando que hace cada una en cada uno de los archivos.

connection

Connection.php

Se define una clase llamada "Connection" que extiende la clase "Mysqli" de PHP. La clase "Connection" establece una conexión con una base de datos MySQL utilizando la función "parent::__construct" para llamar al constructor de la clase "Mysqli". Los parámetros que se le pasan a la función son el servidor de la base de datos ('localhost'), el nombre de usuario ('root'), la contraseña ('') y el nombre de la base de datos ('proyectovotacion'), se utiliza una expresión ternaria para verificar si la conexión a la base de datos es exitosa. Si es exitosa, se imprime el mensaje "DB Conectada", de lo contrario, se imprime el mensaje "Error al conectarse a la base de datos" y se finaliza la ejecución del script mediante la función "die()".

```
1 class Connection extends Mysqli {
2     function __construct() {
3         parent::__construct('localhost', 'root', '', 'proyectovotacion');
4         $this->set_charset('utf8');
5         $this->connect_error == NULL ? 'DB Conectada' : die('Error al conectarse a la base de datos') ;
6     }
7 }
```

models

Candidato.php

Esta es la definición de la clase Candidato. La función obtener_candidato es un método público de la clase que se encarga de obtener los datos de los candidatos desde la base de datos.

Dentro de la función, se crea una instancia de la clase Connection, que se encarga de establecer la conexión con la base de datos. Luego se crea una consulta SQL para obtener los datos de los candidatos de la tabla candidatos.

La consulta se ejecuta utilizando el método query de la instancia de Connection. El resultado de la consulta se almacena en la variable \$resultado.

Luego, se comprueba si la consulta devolvió alguna fila utilizando el método num_rows de la variable \$resultado. Si la consulta devolvió filas, se recorre el resultado utilizando el método fetch_assoc para obtener los datos de cada candidato. Los datos se almacenan en un array \$datos, que se devuelve al final de la función.

```
1  <?php
2  require_once "connection/Connection.php";
3  class Candidato
4  {
5      public function obtener_candidato()
6      {
7          $db = new Connection();
8          $query = "SELECT id, nombre FROM candidatos";
9          $resultado = $db->query($query);
10         $datos = [];
11         if ($resultado->num_rows) {
12             while ($row = $resultado->fetch_assoc()) {
13                 $datos[] = [
14                     'id' => $row['id'],
15                     'nombre' => $row['nombre'],
16                 ];
17             }
18         }
19         return $datos;
20     }
21 }
22
```

Comuna.php

clase llamada "Comuna". En este caso, la clase tiene un método público llamado "obtener_comunas_select" que toma un parámetro llamado "\$region_id". El método se encarga de realizar una consulta a la base de datos para obtener las comunas asociadas a la región especificada.

En la primera línea, se importa la clase "Connection" que se encuentra en la carpeta "connection". Esta clase se encarga de establecer la conexión a la base de datos.

Dentro del método, se crea una nueva instancia de la clase "Connection" y se almacena en la variable "\$db". Luego, se construye la consulta SQL para obtener las comunas correspondientes a la región especificada.

La consulta se ejecuta con el método "query" de la instancia de la clase "Connection" y se almacena en la variable "\$resultado". Luego, se crea un array vacío llamado "\$datos".

Si la consulta devuelve algún resultado, se recorre el resultado con un bucle "while" y se van agregando los datos de cada fila en el array "\$datos". Finalmente, se devuelve el array "\$datos" con los datos de las comunas encontradas.

```
1  <?php
2      require_once "connection/Connection.php";
3
4      class Comuna {
5
6          public function obtener_comunas_select($region_id) {
7              $db = new Connection();
8              $query = "SELECT id, comuna FROM comunas WHERE region_id = $region_id";
9              $resultado = $db->query($query);
10             $datos = [];
11             if($resultado->num_rows) {
12                 while ($row = $resultado->fetch_assoc()) {
13                     $datos[] = [
14                         'id' => $row['id'],
15                         'comuna' => $row['comuna'],
16                     ];
17                 }
18             }
19             return $datos;
20         }
21     }
22 }
```

Region.php

clase llamada "Región". Esta clase tiene un método público llamado "obtener_region_select", que se encarga de obtener los datos de la tabla "regiones" y devolverlos en un formato de arreglo.

Para realizar la conexión a la base de datos, la clase hace uso de la clase "Connection" definida en el archivo "connection/Connection.php". Esta clase se encarga de crear una instancia de la clase "mysqli" para conectarse a la base de datos.

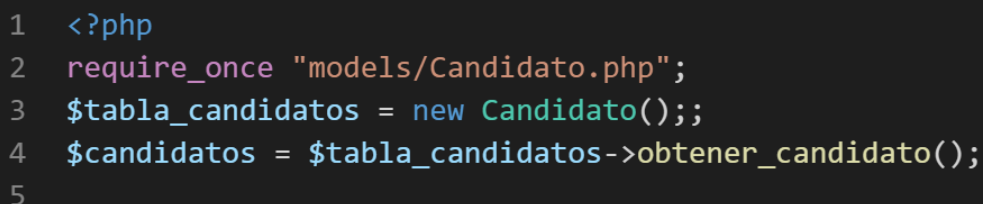
El método "obtener_region_select" ejecuta una consulta SQL para seleccionar los campos "id" y "región" de la tabla "regiones". Luego, se recorre el resultado de la consulta para crear un arreglo asociativo con los datos de cada registro. Este arreglo se almacena en la variable \$datos y se devuelve al final del método.

```
1  <?php
2      require_once "connection/Connection.php";
3
4      class Region {
5
6          public function obtener_region_select() {
7              $db = new Connection();
8              $query = "SELECT id, region FROM regiones";
9              $resultado = $db->query($query);
10             $datos = [];
11             if($resultado->num_rows) {
12                 while ($row = $resultado->fetch_assoc()) {
13                     $datos[] = [
14                         'id' => $row['id'],
15                         'region' => $row['region'],
16                     ];
17                 }
18             }
19             return $datos;
20         }
21     }
```

Logica

candidatos.php

comienza incluyendo el archivo Candidato.php y luego está creando una instancia de la clase Candidato utilizando la sintaxis de new. Luego, la instancia creada se utiliza para llamar al método obtener_candidato() que devuelve un array con información de todos los candidatos obtenidos de la base de datos. Esta información se almacena en la variable \$candidatos, que puede ser utilizada más adelante en el código.

A screenshot of a code editor with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. The code is written in PHP and is numbered from 1 to 5 on the left side.

```
1 <?php
2 require_once "models/Candidato.php";
3 $tabla_candidatos = new Candidato();;
4 $candidatos = $tabla_candidatos->obtener_candidato();
5
```

comunas.php

comienza incluyendo el archivo "Comuna.php", que contiene la definición de la clase Comuna. Luego, se define una variable vacía llamada \$comunas. A continuación, se verifica si se ha recibido la variable \$_GET['region'] mediante una declaración if. Si la variable existe, se crea una instancia de la clase Comuna y se llama al método obtener_comunas_select pasando como argumento el valor de \$_GET['region']. El resultado se guarda en la variable \$comunas.

Finalmente, se imprime el resultado en formato JSON utilizando la función `json_encode()`. Se crea un array asociativo con la clave "data" y el valor de `$comunas` y se lo pasa como argumento a la función `json_encode()`. El resultado final se imprime en la página web.

```
1 <?php
2     require_once "models/Comuna.php";
3     $comunas = [];
4     if(isset($_GET['region'])) {
5         $tabla_comunas = new Comuna();
6         $comunas = $tabla_comunas->obtener_comunas_select($_GET['region']);
7     }//end if
8     echo json_encode(['data' => $comunas]);
```

region.php

comienza incluyendo el archivo "Region.php" y luego está creando una instancia de la clase `Region` utilizando la sintaxis de `new`. Luego, la instancia creada se utiliza para llamar al método `obtener_candidato()` que devuelve un array con información de todos los candidatos obtenidos de la base de datos. Esta información se almacena en la variable `$candidatos`, que puede ser utilizada más adelante en el código.

```
1 <?php
2     require_once "models/Region.php";
3     $tabla_estado = new Region();
4     $regiones = $tabla_estado->obtener_region_select();
```


index.php

Html simple, primero que nada se referencia el archivo region.php, candidatos.php, el style.css, posteriormente JQuery, y una librería llamada inputmask, posteriormente se crea el formulario con el método post, que envía los datos del formulario al archivo votos.php, al final se referencia al archivo JS. utilizado para validar algunas casillas, y mostrar de manera dinámica los select de comuna.

Se recorren con un foreach los valores traídos desde region.php y candidato.php para mostrarlos en los select directamente desde la base de datos

```

1  <?php
2  require "region.php";
3  require "candidatos.php"
4  ?>
5  <!DOCTYPE html>
6  <html lang="en">
7
8  <head>
9      <meta charset="UTF-8">
10     <meta http-equiv="X-UA-Compatible" content="IE=edge">
11     <meta name="viewport" content="width=device-width, initial-scale=1.0">
12     <link rel="stylesheet" type="text/css" href="style.css">
13     <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
14     <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery.inputmask/5.0.6/jquery.inputmask.min.js"></script>
15     <title>Formulario De Votacion</title>
16 </head>
17
18 <body>
19
20     <h1>Formulario de Votación</h1>
21
22     <div id="form">
23         <form class="formulario" action="votos.php" method="post">
24             <label for="nombre">Nombre y Apellido:</label>
25             <input type="text" id="nombre" name="nombre" required><br>
26
27             <label for="alias">Alias:</label>
28             <input type="text" id="alias" name="alias" title="Debe contener letras y numeros, 6 o mas caracteres" required pattern="(?!.*\d)(?!.*[a-zA-Z])[a-zA-Z0-9]{6,}"><br>
29
30             <label for="rut">RUT:</label>
31             <input type="text" id="rut" name="rut" title="Formato de Rut 11.111.111-1" required><br>
32
33             <label for="email">Email:</label>
34             <input type="email" id="email" name="email" required><br>
35
36             <label for="">Region:</label>
37             <select id="region" name="region">
38                 <option value="">Seleccionar Region</option>
39                 <?php
40                     foreach ($regiones as $region) {
41                         echo '<option value="' . $region['id'] . '">' . $region['region'] . '</option>';
42                     } //end foreach
43                 ?>
44             </select>
45
46             <label for="">Comunas:</label>
47             <select id="comuna" name="comuna">
48                 <option value="">Seleccionar Comuna</option>
49             </select>
50
51             <label for="candidato">Candidato:</label>
52             <select id="candidato" name="candidato">
53                 <option value="">Seleccione un candidato</option>
54                 <?php
55                     foreach ($candidatos as $candidato) {
56                         echo '<option value="' . $candidato['nombre'] . '">' . $candidato['nombre'] . '</option>';
57                     }
58                 ?>
59             </select><br>
60
61             <label for="entero">Cómo se enteró de nosotros:</label>
62             <input type="checkbox" id="entero1" name="entero[]" value="Internet">Web
63             <input type="checkbox" id="entero2" name="entero[]" value="TV">TV
64             <input type="checkbox" id="entero3" name="entero[]" value="Redes Sociales">Redes Sociales
65             <input type="checkbox" id="entero4" name="entero[]" value="Amigo">Amigo <br>
66
67             <input type="submit" name="votar" value="votar" onclick="return validarCheckboxes();">
68         </form>
69     <script src="script.js"></script>
70
71 </body>
72
73 </html>

```

votos.php

Primero, se comprueba que se haya enviado una solicitud POST utilizando `$_SERVER['REQUEST_METHOD'] === 'POST'`, luego, se asignan los valores de los campos del formulario a las variables correspondientes usando `$_POST[]`.

A continuación, se realiza la validación de cada campo utilizando una serie de comprobaciones, por ejemplo, para validar el campo de correo electrónico, se utiliza `filter_var($email, FILTER_VALIDATE_EMAIL)`, que devuelve false si el correo electrónico no es válido. También se utiliza `preg_match()` para validar el formato del campo RUT y asegurarse de que cumpla con las reglas establecidas.

Si se encuentra algún error en los datos ingresados, se muestra un mensaje de error correspondiente y se detiene el proceso.

Si no se encuentran errores, los datos se insertan en la base de datos utilizando una consulta SQL INSERT.

```

1 <?php
2
3 if ($_SERVER['REQUEST_METHOD'] === 'POST') {
4     $nombre = $_POST['nombre'];
5     $alias = $_POST['alias'];
6     $rut = $_POST['rut'];
7     $email = $_POST['email'];
8     $region = $_POST['region'];
9     $comuna = $_POST['comuna'];
10    $candidato = $_POST['candidato'];
11    $como = implode(" ", $_POST['entero']);
12
13    $errores = [];
14
15    // Validación de nombre
16    if (empty($nombre)) {
17        $errores[] = "El campo Nombre es obligatorio.";
18    } else if (!preg_match('/^[a-zA-Z\s]+$/i', $nombre)) {
19        $errores[] = "El campo Nombre solo puede contener letras y espacios.";
20    }
21
22    // Validación de alias
23    if (empty($alias)) {
24        $errores[] = "El campo Alias es obligatorio.";
25    } else if (!preg_match('/^[a-zA-Z0-9]+$/i', $alias)) {
26        $errores[] = "El campo Alias solo puede contener letras y numeros.";
27    }
28
29    // Validación de RUT
30    if (empty($rut)) {
31        $errores[] = "El campo RUT es obligatorio.";
32    } else if (!preg_match('/^\\d{1,2}\\.\\d{3}\\.\\d{3}-[0-9X]{1}$/', $rut)) {
33        $errores[] = "El RUT ingresado no es válido.";
34    }
35
36    // Validación de email
37    if (empty($email)) {
38        $errores[] = "El campo Email es obligatorio.";
39    } else if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
40        $errores[] = "El email ingresado no es válido.";
41    }
42
43    // Validación de región y comuna
44    if (empty($region)) {
45        $errores[] = "Debe seleccionar una Región.";
46    }
47    if (empty($comuna)) {
48        $errores[] = "Debe seleccionar una Comuna.";
49    }
50
51    // Validación de candidato
52    if (empty($candidato)) {
53        $errores[] = "Debe seleccionar un Candidato.";
54    }
55
56    // Validación de checkboxes
57    if (count($_POST['entero']) < 2) {
58        $errores[] = "Debe seleccionar al menos 2 opciones en 'Como se enteró de nosotros.'";
59    }
60
61    // Si hay errores, se muestran
62    if (count($errores) > 0) {
63        foreach ($errores as $error) {
64            echo "<div class='alert alert-danger'>$error</div>";
65        }
66    } else {
67        if (isset($_POST['votar'])) {
68            $mysql = new mysqli("localhost", "root", "", "proyectovotacion");
69            $consulta = "SELECT * FROM votos WHERE rut = '$rut'";
70            $resultado = mysqli_query($mysql, $consulta);
71            if (mysqli_num_rows($resultado) > 0) {
72                echo "<script>alert('Usted ya ha votado'); window.history.go(-1);</script>";
73            } else {
74                $insertar = "INSERT INTO votos ('id', 'nombre', 'alias', 'rut', 'email', 'region', 'comuna', 'candidato', 'nosotros') VALUES ('{value-1}', '$nombre', '$alias', '$rut', '$email', '$region', '$comuna', '$candidato', '$como')";
75                $insertar = mysqli_query($mysql, $insertar);
76                if ($insertar) {
77                    echo "<script>alert('Los Datos han sido registrados correctamente'); window.location.href = 'http://localhost/votacion';</script>";
78                }
79            }
80        }
81    }
82 }

```

script.js

Este código se encarga de definir la funcionalidad de la página utilizando JavaScript y jQuery.

En la primera función, se utiliza jQuery para detectar un cambio en el elemento con id "region". Al detectar el cambio, se envía una petición AJAX a través del método GET a la página "comunas.php" con el parámetro "region" que contiene el valor seleccionado en el elemento "region". El tipo de datos que se espera recibir es JSON. En caso de éxito, se itera sobre los datos recibidos y se generan opciones para el elemento "comuna", utilizando el nombre de la comuna como valor y como texto visible en la opción.

En la segunda función, se utiliza la librería "inputmask" para definir una máscara de entrada para el campo con id "rut". Esta máscara permite que el usuario ingrese un número de rut con su respectivo dígito verificador en un formato predefinido.

Finalmente, se define una función llamada "validarCheckboxes" que se encarga de validar que al menos dos casillas de selección estén marcadas en el grupo de checkboxes con el atributo "type" igual a "checkbox". Si se detecta que hay menos de dos casillas seleccionadas, se muestra una alerta y se retorna "false". Si se cumplen las condiciones, se retorna "true".

```

1  $(document).ready(function () {
2      $('#region').change(function() {
3          $.ajax({
4              url: 'comunas.php?region=' + $(this).val(),
5              type: 'GET',
6              dataType: 'json',
7              success: function(data) {
8                  let html = '<option value="">Seleccionar Comuna</option>';
9                  if (data.data.length > 0) {
10                      for (let i = 0; i < data.data.length; i++) {
11                          html += '<option value="' + data.data[i].comuna + '">' + data.data[i].comuna + '</option>';
12                      }
13                  }
14                  $('#comuna').html(html);
15              },
16              error: function(jqXHR, textStatus, errorThrown) {
17                  console.log(textStatus, errorThrown);
18              }
19          });
20      });
21
22      $('#rut').inputmask({
23          mask: '9{1,2}.999.999-K',
24          definitions: {
25              'K': {
26                  validator: "[0-9kK]",
27                  casing: 'upper'
28              }
29          }
30      });
31  });
32
33  function validarCheckboxes() {
34      var checkboxes = $('input[type="checkbox"]:checked');
35      if (checkboxes.length < 2) {
36          alert("Debe al menos 2 casillas en Como se entero de nosotros.");
37          return false;
38      }
39      return true;
40  }

```