UNIVERSIDAD POLITECNICA DE VICTORIA

TECNOLOGIAS Y APLICACIÓN WEB

PROF. MARIO HUMBERTO RODRIGUEZ CHAVEZ

SISTEMA DE CONTROL DE VENTA LUGARES PARA ACADEMIA DE BAILE DANZLIFE

MANUAL TECNICO

ELABORADO POR
FHER FRANCISCO TORRES PAZ

APLICACIÓN

CODIGO FUENTE

MODELO

```php
//Funcion del modelo para registrar una alumna en la base de datos
public function registrarAlumnaModel($datosModel,$tabla)
{
    //Se hace la sentencia SQL necesaria para hacer el registro en la base de datos
    $stm = Conexion::conectar()->prepare("INSERT INTO $tabla (
        nombre_alumna,apellido_alumna,fecha_nacimiento,id_grupo) VALUES
        (:nombre,:apellido,:fecha,:grupo)");
    //Parametros con todos los datos de la alumna
    $stm->bindParam(":nombre",$datosModel['nombre']);
    $stm->bindParam(":apellido",$datosModel['apellido']);
    $stm->bindParam(":fecha",$datosModel['fecha']);
    $stm->bindParam(":grupo",$datosModel['grupo']);
    if($stm->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }

    $stm->close();
}
```

Función del modelo que registra una alumna, dónde se dictará una sentencia SQL con todos los parámetros que vienen siendo los datos de la alumna, y entonces se crean los parámetros que tendrán como valor los datos que se registraron en el formulario para por ultimo ejecutar la sentencia SQL en nuestra base de datos

```php
//Funcion del modelo para registrar un grupo nuevo en la base de datos
public function registrarGrupoModel($datosModel,$tabla)
{
    //Sentencia SQL para hacer el registro del grupo en la base de datos
    $stm = Conexion::conectar()->prepare("INSERT INTO $tabla (nombre_grupo) VALUES
        (:nombre)");
    $stm->bindParam(":nombre",$datosModel);
    if($stm->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }

    $stm->close();
}
```

Función del modelo que se encarga de registrar un grupo nuevo en la base datos, dónde se dictará una sentencia SQL, los parámetros serán los datos que se registran en el formulario para por último ejecutar la sentencia SQL

```php
//Funcion del modelo para registrar un pago de lugar en la base de datos
public function registrarPagoModel($datosModel,$tabla)
{
    //Sentencia SQL para hacer el registro en la base de datos
    $stm = Conexion::conectar()->prepare("INSERT INTO $tabla (
        id_alumna,id_grupo,nombre_mama,fecha_pago,fecha_envio,ruta,folio) VALUES
        (:id_alumna,:id_grupo,:nombre_mama,:fecha_pago,:fecha_envio,:ruta,:folio)");
    //Parametros con todos los datos que se ingresaran en la base de datos
    $stm->bindParam(":id_alumna",$datosModel['id_alumna']);
    $stm->bindParam(":id_grupo",$datosModel['id_grupo']);
    $stm->bindParam(":nombre_mama",$datosModel['nombreMama']);
    $stm->bindParam(":fecha_pago",$datosModel['fecha_pago']);
    $stm->bindParam(":fecha_envio",$datosModel['fecha_envio']);
    $stm->bindParam(":ruta",$datosModel['ruta']);
    $stm->bindParam(":folio",$datosModel['folio']);
    if($stm->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }

    $stm->close();
}
```

Función que se encarga de registrar un pago nuevo, dictando una sentencia SQL. Teniendo como parámetros los datos del formulario para registrar un pago/lugar teniendo en cuenta un dato en especial que viene siendo la ruta, se encarga de almacenar el nombre de la imagen. Para que al momento de cargarla entonces Podamos acceder a esa imagen en específico.

```php
//Funcion para la vista de todos los pagos que se han registro hasta el momento
public function vistaPagosPublicosModel($tabla)
{
    //Sentnecia SQL con innerjoins ya que la tabla de pagos solo tiene registrados
    los id
    //asi podemos obtener el nombre de la alumna que ha sido registrada
    $stmt = Conexion::conectar()->prepare("SELECT * FROM $tabla inner join alumna on
        $tabla.id_alumna = alumna.id_alumna inner join grupo on $tabla.id_grupo =
        grupo.id_grupo order by fecha_envio asc");
    $stmt->execute();

    #fetchAll(): Obtiene todas las filas de un conjunto de resultados asociado al
    objeto PDOStatement.
    return $stmt->fetchAll();

    $stmt->close();
}
```

Función que se encarga de crear las vistas de los pagos, efectuando una sentencia SQL dónde tendrá inner joins que se encargarán de traer los datos de las llaves foráneas y tener nuestra vista con las descripciones correspondientes.

```php
//Funcion para hacer la vista de todos los grupos que se han registrado en la base
de datos
public function vistaGruposModel($tabla)
{
    //Sentencia SQL que ejecuta la consulta
    $stmt = Conexion::conectar()->prepare("SELECT * FROM $tabla");
    $stmt->execute();

    #fetchAll(): Obtiene todas las filas de un conjunto de resultados asociado al
    objeto PDOStatement.
    return $stmt->fetchAll();

    $stmt->close();
}
```

Función que se encarga de efectuar la sentencia SQL para crear la vista de los grupos y mostrarlos en una tabla

```php
//Funcion para hacer la vista de todas las alumnas que han sido registradas
public function vistaAlumnasModel($tabla)
{
    //Sentencia SQL con innerjoins ya que la alumna solo tiene registrado el id del
    grupo
    //y con esto podemos obtener el nombre del grupo
    $stmt = Conexion::conectar()->prepare("SELECT * FROM $tabla inner join grupo on $
        tabla.id_grupo = grupo.id_grupo");
    $stmt->execute();

    #fetchAll(): Obtiene todas las filas de un conjunto de resultados asociado al
    objeto PDOStatement.
    return $stmt->fetchAll();

    $stmt->close();

}
```

Función que se encarga de crear la sentencia SQL para ejecutar una sentencia SQL, ésta sentencia SQL tendrá ciertos inner join para poder traer los datos de las llaves foráneas de esa tabla y entonces poder hacer una descripción de estos campos.

```php
//Funcion del modelo para borrar un pago en base a su id
public function borrarPagoModel($datosModel,$tabla)
{
    //Sentencia SQL que efectuara el borrado del registro en base a su id
    $stmt = Conexion::conectar()->prepare("DELETE FROM $tabla where id_pago = :id");
    //Parametros para obtener el id y efectuar
    $stmt->bindParam("id",$datosModel,PDO::PARAM_STR);
    if($stmt->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }

    $stmt->close();
}
```

Función del modelo que se encarga de borrar un pago en base a su ID, se ejecutará la sentencia SQL para borra el pago correspondiente

```php
//Funcion para borrar un grupo en base a su id
public function borrarGrupoModel($datosModel,$tabla)
{
    $stmt = Conexion::conectar()->prepare("DELETE FROM $tabla where id_grupo = :id");
    $stmt->bindParam("id",$datosModel);
    if($stmt->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }

    $stmt->close();
}
```

Función del modelo para borrar un grupo de la base de datos, teniendo como parámetro su id para borrar el grupo correspondiente.

```php
//Funcion para borrar una alumna en base a su id
public function borrarAlumnaModel($datosModel,$tabla)
{
    $stmt = Conexion::conectar()->prepare("DELETE FROM $tabla where id_alumna = :id")
        ;
    $stmt->bindParam("id",$datosModel);
    if($stmt->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }

    $stmt->close();
}
```

Función del modelo que se encarga borrar una alumna de la base de datos, dónde los parámetros será el id de la alumna para ejecutar la sentencia SQL correspondiente

```php
//Funcion del modelo para hacer el formulario y traer los valores del modelo dado su
ID
public function editarPagoModel($datosModel,$tabla)
{
    $stmt = Conexion::conectar()->prepare("SELECT * from $tabla where id_pago = :id")
        ;
    $stmt->bindParam(":id",$datosModel,PDO::PARAM_INT);
    $stmt->execute();
    return $stmt->fetch();

    $stmt->close();

}
```

Función del modelo que se encarga de obtener los datos de un pago, para traer todos sus datos en un formulario, así pudiendo modificarlos para hacer una actualización. Todo esto teniendo como parámetro el id del pago que se escogió

```php
//Funcion para obtener lo datos del grupo que se pretende actualizr
public function editarGrupoModel($datosModel,$tabla)
{
    $stmt = Conexion::conectar()->prepare("SELECT * from $tabla where id_grupo = :id"
        );
    $stmt->bindParam(":id",$datosModel,PDO::PARAM_INT);
    $stmt->execute();
    return $stmt->fetch();

    $stmt->close();

}
```

Función del modelo que se encarga de obtener los datos de un grupo, teniendo en cuenta su id para traer los datos correspondientes al grupo que se eligió.

```php
//Funcion para obtener los datos del usuario que se pretende actualizar
public function editarAlumnaModel($datosModel,$tabla)
{
    $stmt = Conexion::conectar()->prepare("SELECT * from $tabla where id_alumna = :id
        ");
    $stmt->bindParam(":id",$datosModel,PDO::PARAM_INT);
    $stmt->execute();
    return $stmt->fetch();

    $stmt->close();

}
```

Función del modelo que se encarga de obtener los datos correspondientes de una alumna en base a su id, siendo el id de la alumna que se eligió editar

```php
//Funcion del modelo para hacer UPDATE a la BD en la tabla producto
public function actualizarPagoModel($datosModel,$tabla)
{
    $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET nombre_mama = :mama,
        fecha_pago = :fecha_pago, fecha_envio = :fecha_envio, folio = :folio where
        id_pago = :id");

    $stmt->bindParam(":mama",$datosModel["mama"]);
    $stmt->bindParam(":id",$datosModel["id"]);
    $stmt->bindParam(":fecha_pago",$datosModel["fecha_pago"]);
    $stmt->bindParam(":fecha_envio",$datosModel["fecha_envio"]);
    $stmt->bindParam(":folio",$datosModel["folio"]);


    if($stmt->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }
}
```

Función del modelo que se encarga actualizar un pago, en base a los datos que se registraron en el formulario para actualizar el pago.

```php
//Funcion para hacer el update a la base de datos de la categoria
public function actualizarGrupoModel($datosModel,$tabla)
{
    print_r($datosModel);
    $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET nombre_grupo = :nombre
        where id_grupo = :id");
    $stmt->bindParam(":id",$datosModel["id"]);
    $stmt->bindParam(":nombre",$datosModel["nombre"]);

    if($stmt->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }
}
```

Función del modelo que se encarga en actualizar un grupo, en base a los datos que se obtuvieron de su respectivo formulario. Formulando una sentencia SQL y efectuándola.

```php
//Funcion para hacer el update a la tabla de usuario
public function actualizarAlumnaModel($datosModel,$tabla)
{
    print_r($datosModel);
    $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET nombre_alumna =
        :nombre, apellido_alumna = :apellido, fecha_nacimiento = :fecha, id_grupo =
        :grupo where id_alumna = :id");
    $stmt->bindParam(":nombre",$datosModel["nombre"],PDO::PARAM_STR);
    $stmt->bindParam(":id",$datosModel["id"],PDO::PARAM_INT);
    $stmt->bindParam(":apellido",$datosModel["apellido"],PDO::PARAM_STR);
    $stmt->bindParam(":fecha",$datosModel["fecha"],PDO::PARAM_STR);
    $stmt->bindParam(":grupo",$datosModel["grupo"],PDO::PARAM_INT);

    if($stmt->execute())
    {
        return "success";
    }
    else
    {
        return "error";
    }
}
```

Función del modelo que se encarga de hacer una actualización de la alumna, en base a los datos que se obtuvieron del formulario correspondiente. Efectuando una sentencia SQL. Y ejecutándola

```php
//Funcion del modelo que nos permite traer los datos del usuario y comprobar su emai
y password para permitir el login
public static function ingresoUsuarioModel($datosModel, $tabla){

    $stmt = Conexion::conectar()->prepare("SELECT * FROM $tabla WHERE email = :email
        AND password = :password ");
    $stmt->bindParam(":email", $datosModel["email"], PDO::PARAM_STR);
    $stmt->bindParam(":password", $datosModel["password"], PDO::PARAM_STR);
    $stmt->execute();

    #fetch(): Obtiene una fila de un conjunto de resultados asociado al objeto
    PDOStatement.
    return $stmt->fetch();

    $stmt->close();

}
```

Función del modelo que se encarga de buscar un resultado con el usuario y contraseña que se escribió en el formulario al querer iniciar sesión, para saber si algún usuario registrado con esos datos

```php
public function enlazarPagina()
{
    if(isset($_GET['action']))
    {
        $enlace = $_GET['action'];
    }
    else
    {
        $enlace = 'inicio';
    }

    $respuesta = Informacion::enlazador($enlace);
    include($respuesta);
}
```

Función que se encarga enlazar una página, obteniendo la acción que se realiza y entonces obteniendo la vista que corresponde a esa acción para al final incluirla y se muestre en nuestro navegador

```php
//Funcion plantilla que trae toda la plantilla y su navegacion
public function plantilla()
{
    require("views/template.php");
}
```

Función que incluye la plantilla que tendrá todos los recursos necesarios así como las diferentes funciones de JavaScript. Esto nos servirá para solo escribirlas una vez y que todo el sistema sea capaz de acceder a ellas.

```php
public function registrarGrupoController()
{
    $respuesta = "";
    if(isset($_POST['registrar']))
    {

        if(!empty($_POST['nombre']))
        {
            $datosController = $_POST['nombre'];
            $respuesta = Datos::registrarGrupoModel($datosController,'grupo');
        }
    }
    if($respuesta == "success")
    {
        echo"<script>
                window.location = 'index.php?action=grupos';
            </script>";
    }
}
```

Función del controlador que se encarga de registrar un grupo, trayendo los datos del formulario para después mandar llamar la función del modelo ingresando el nombre de la tabla y los datos que se registraron en dicho formulario. Si el registro en la base de datos se efectúa correctamente entonces solo se actualiza la página.

```php
public function registrarAlumnaController()
{
    $respuesta = "";
    if(isset($_POST['registrar']))
    {
        if(!empty($_POST['nombre']))
        {
            $oringialDate = $_POST['fecha'];
            $actualDate = date("Y-m-d",strtotime($oringialDate));
            $datosController = array('nombre'=>$_POST['nombre'],
                                     'apellido'=>$_POST['apellido'],
                                     'fecha'=>$actualDate,
                                     'grupo'=>$_POST['grupo']);

            $respuesta = Datos::registrarAlumnaModel($datosController,'alumna');
        }
    }
    if($respuesta == "success")
    {
        echo"<script>
                window.location = 'index.php?action=alumnas';
            </script>";
    }
}
```

Función del controlador para registrar una alumna nueva, teniendo en un arreglo los datos que se obtuvieron del formulario, para después llamar la función que está en el modelo, dónde los parámetros sean el nombre de la tabla y el arreglo con los datos. Si el registro es correcto se redirigirá la página.

```php
public function registrarPagoController()
{
    if(isset($_POST["registrar"]))
    {
        if(!empty($_POST['mama']))
        {
            $target_dir = "uploads/";
            $target_file = $target_dir . basename($_FILES["fileToUpload"]["name"]);
            $uploadOk = 1;
            $imageFileType = strtolower(pathinfo($target_file,PATHINFO_EXTENSION));
              $check = getimagesize($_FILES["fileToUpload"]["tmp_name"]);
              if($check !== false)
              {
                  echo "File is an image - " . $check["mime"] . ".";
                  $uploadOk = 1;
              }
              else
              {
                  echo "File is not an image.";
                  $uploadOk = 0;
              }

               // Check if file already exists
              if (file_exists($target_file))
              {
                  echo "Sorry, file already exists.";
                  $uploadOk = 0;
              }
```

Función del controlador que se encarga de registrar un pago, dónde se obtendrá la imagen que se cargó y pasará una serie de pruebas antes de que se haga el registro (Tamaño correcto de la foto, extensión del archivo correcto, etc.). Si todo se hace correctamente se hará el registro en la base de datos.

```php
public function vistaPagosPublicosController()
{
    $respuesta=Datos::vistaPagosPublicosModel("pagos");

    $counter = 1;
    foreach($respuesta as $row => $item){
    echo'<tr>
            <td>'.$counter.'</td>
            <td>'.$item["nombre_alumna"].'</td>
            <td>'.$item["nombre_grupo"].'</td>
            <td>'.$item["nombre_mama"].'</td>
            <td>'.$item["fecha_pago"].'</td>
            <td>'.$item["fecha_envio"].'</td>
        </tr>';
    $counter++;
    }
}
```

Función del modelo que se encarga de hacer la vista de los pagos, está función trae la vista completa a excepción del link para ver la foto, y el folio que se registró. Esta tendrá acceso a todo el mundo.

```php
public function vistaPagosAdminController()
{
    $respuesta=Datos::vistaPagosPublicosModel("pagos");

    $counter = 1;
    foreach($respuesta as $row => $item){
    echo'<tr>
            <td>'.$counter.'</td>
            <td>'.$item["nombre_alumna"].'</td>
            <td>'.$item["nombre_grupo"].'</td>
            <td>'.$item["nombre_mama"].'</td>
            <td>'.$item["fecha_pago"].'</td>
            <td>'.$item["fecha_envio"].'</td>
            <td> <center><a href="uploads/'.$item["ruta"].'">Ver</a></center></td>
            <td>'.$item["folio"].'</td>
            <td><div class="btn-group">
                  <button type="button" class="btn btn-default dropdown-toggle"
                      data-toggle="dropdown">
                    <span class="fa fa-cog"></span>
                  </button>
                  <div class="dropdown-menu">
                    <a class="dropdown-item" href="index.php?action=editar_lugar&id='
                        .$item["id_pago"].'">Actualizar</a>
                    <a class="dropdown-item" onclick="confirmarDelete('.$item["
                        id_pago"].');" href="index.php?action=lugares_admin&idBorrar=
                        '.$item["id_pago"].'" id="btn'.$item["id_pago"].'">Eliminar</
                        a>
                  </div></td>

        </tr>';
    $counter++;
    }
```

Función del controlador para hacer la vista de los pagos pero para la parte de administrador, dónde se mostrará la vista completa incluyendo el folio que se registró así como el link

```
public function vistaGruposController()
{
    $respuesta = Datos::vistaGruposModel("grupo");

    #El constructor foreach proporciona un modo sencillo de iterar sobre arrays.
    foreach funciona sólo sobre arrays y objetos, y emitirá un error al intentar
    usarlo con una variable de un tipo diferente de datos o una variable no
    inicializada.

    foreach($respuesta as $row => $item){
    echo'<tr>
            <td>'.$item["id_grupo"].'</td>
            <td>'.$item["nombre_grupo"].'</td>
            <td><div class="btn-group">
                  <button type="button" class="btn btn-default dropdown-toggle"
                      data-toggle="dropdown">
                    <span class="fa fa-cog"></span>
                  </button>
                  <div class="dropdown-menu">
                    <a class="dropdown-item" href="index.php?action=editar_grupo&id='
                        .$item["id_grupo"].'">Actualizar</a>
                    <a class="dropdown-item" onclick="confirmarDelete('.$item["
                        id_grupo"].');" href="index.php?action=grupos&idBorrar='.$
                        item["id_grupo"].'" id="btn'.$item["id_grupo"].'">Eliminar</
                        a>
                  </div></td>
        </tr>';
    }

}
```

Función que se encarga hacer la vista de los grupos, dónde se creará una fila por cada grupo que encuentre y en cada columna su respectiva información. Todo para ser mostrado en una tabla.

```php
public function obtenerGruposController()
{
    $respuesta = Datos::vistaGruposModel("grupo");

    #El constructor foreach proporciona un modo sencillo de iterar sobre arrays.
    foreach funciona sólo sobre arrays y objetos, y emitirá un error al intentar
    usarlo con una variable de un tipo diferente de datos o una variable no
    inicializada.

    foreach($respuesta as $row => $item)
    {
        echo'<option value='.$item["id_grupo"].'>'.$item["nombre_grupo"].'</option>';
    }

}
```

Función para obtener los grupos y ponerlo en un select, esto nos sirve a la hora de registrar una alumna y poder poner todos los grupos en un select para que se registre alguno que esté en la base de datos.

```php
public function obtenerAlumnasController()
{
    $respuesta = Datos::vistaAlumnasModel("alumna");

    #El constructor foreach proporciona un modo sencillo de iterar sobre arrays.
    foreach funciona sólo sobre arrays y objetos, y emitirá un error al intentar
    usarlo con una variable de un tipo diferente de datos o una variable no
    inicializada.

    foreach($respuesta as $row => $item)
    {
        echo'<option value='.$item["id_alumna"].'>'.$item["nombre_alumna"].'</option>
            ';
    }

}
```

Función para obtener las alumnas, esto nos sirve para la hora de registrar un pago obtengamos todas las alumnas que están registradas en la base de datos.

```php
//Función dle controlador para crear una vista de usuarios y mostrarlos en la tabla
public function vistaAlumnasController()
{

    $respuesta = Datos::vistaAlumnasModel("alumna");

    #El constructor foreach proporciona un modo sencillo de iterar sobre arrays.
    foreach funciona sólo sobre arrays y objetos, y emitirá un error al intentar
    usarlo con una variable de un tipo diferente de datos o una variable no
    inicializada.

    foreach($respuesta as $row => $item){
    echo'<tr>
            <td>'.$item["id_alumna"].'</td>
            <td>'.$item["nombre_alumna"].'</td>
            <td>'.$item["apellido_alumna"].'</td>
            <td>'.$item["fecha_nacimiento"].'</td>
            <td>'.$item["nombre_grupo"].'</td>
            <td><div class="btn-group">
                  <button type="button" class="btn btn-default dropdown-toggle"
                      data-toggle="dropdown">
                    <span class="fa fa-cog"></span>
                  </button>
                  <div class="dropdown-menu">
                    <a class="dropdown-item" href="index.php?action=editar_alumna&id=
                        '.$item["id_alumna"].'">Actualizar</a>
                    <a class="dropdown-item" onclick="confirmarDelete('.$item["
                        id_alumna"].');" href="index.php?action=alumnas&idBorrar='.$
                        item["id_alumna"].'" id="btn'.$item["id_alumna"].'">Eliminar<
                        /a>
                  </div></td>
        </tr>';
```

Función que se encarga de hacer la vista de las alumnas, obteniendo cada dato de la alumna, cada alumna es una fila nueva, dónde las columnas es cada dato correspondiente. Al final se crearán dos botones que corresponderá para actualizar o borrar algún registro

```php
public function borrarPagoController()
{
    if(isset($_GET['idBorrar']))
    {
        $datosController = $_GET['idBorrar'];
        $respuesta = Datos::borrarPagoModel($datosController,'pagos');
        if($respuesta == "success")
        {
            echo"<script>
                    window.location = 'index.php?action=lugares_admin';
                </script>";
        }
    }
}
```

Función que se encarga de borrar algún pago, esto nos sirve para cuando borramos algún pago, esta es la función que se encarga de llamar el modelo para efectuar la sentencia SQL

```php
public function borrarGrupoController()
{
    if(isset($_GET['idBorrar']))
    {
        $datosController = $_GET['idBorrar'];
        //Primero se borran los productos de esa categoria
        //$respuesta2 =
        Datos::borrarProductoPorCategoriaModel($datosController,'producto');
        //SE borra la categoria
        $respuesta = Datos::borrarGrupoModel($datosController,'grupo');

        if($respuesta == "success")
        {
            echo"<script>
                    window.location = 'index.php?action=grupos';
                </script>";
        }
    }
}
```

Función que se encarga de llamar el modelo para efectuar la sentencia SQL que borrará algún grupo de la base de datos

```php
public function editarPagoController()
{
    if(isset($_GET['id']))
    {
        $datosController = $_GET["id"];
        $respuesta = Datos::editarPagoModel($datosController, "pagos");
        $actualDate = $respuesta['fecha_envio'];
        $newDate = date('Y-m-d\TH:i:s', strtotime($actualDate));

        echo'<div class="card card-success">
                <div class="card-header">
                    <h3 class="card-title">Editar Producto</h3>
                </div>
                <div class="card-body">
                    <input name="idEditar" class="form-control form-control-lg"
                        type="hidden" placeholder=".input-lg" value = "'.$respuesta["
                        id_pago"].'">
                    <br>

                    <input name="mamaEditar" class="form-control form-control-lg"
                        type="text" placeholder=".input-lg" value = "'.$respuesta["
                        nombre_mama"].'">
                    <br>
                    <input name="fecha_pago" class="form-control form-control-lg"
                        type="date" placeholder=".input-lg" value = "'.$respuesta['
                        fecha_pago'].'">
                    <br>

                    <input name="fecha_envio" class="form-control form-control-lg"
                        type="datetime-local" placeholder=".input-lg" value = "'.$
                        newDate.'">
                    <br>
                    <input name="folioEditar" class="form-control form-control-lg"
                        type="text" placeholder=".input-lg" value = "'.$respuesta["
```

Función que se encarga de crear el formulario que contendrá todos los datos en caso de que se quiera actualizar algún pago. Todo esto en base al id del registro que se eligió.

```php
public function editarGrupoController()
{
    if(isset($_GET['id']))
    {
        $datosController = $_GET["id"];
        $respuesta = Datos::editarGrupoModel($datosController, "grupo");

        echo'<div class="card card-success">
                <div class="card-header">
                    <h3 class="card-title">Editar Producto</h3>
                </div>
                <div class="card-body">
                    <input name="idEditar" class="form-control form-control-lg"
                        type="hidden" placeholder=".input-lg" value = "'.$respuesta["
                        id_grupo"].'">
                    <br>
                    <input name="nombreEditar" class="form-control form-control-lg"
                        type="text" placeholder=".input-lg" value = "'.$respuesta["
                        nombre_grupo"].'">
                    <br>
                     <button type="submit" name="btn_actualizar" id="btn"
                         class="btn btn-block btn-outline-primary"
                         onclick="confirmarUpdate();" style="float:right;">Guardar
                         cambios</button>
                </div>
            </div>';
    }

}
```

Función que se encarga de hacer el formulario para crear el formulario que contendrá lo datos del grupo en base al id del registro que se elgió.

```php
public function controlNav()
{
    if(isset($_SESSION['validar']))
    {


        echo '<li class="nav-item">
                <a href="index.php?action=dashboard" class="nav-link">
                    <i class="nav-icon fa fa-dashboard"></i>
                    <p>Dashboard</p>
                </a>
            </li>
            <li class="nav-item">
                <a href="index.php?action=alumnas" class="nav-link">
                    <i class="nav-icon fa fa-dashboard"></i>
                    <p>Alumnas</p>
                </a>
            </li>
            <li class="nav-item">
                <a href="index.php?action=grupos" class="nav-link">
                    <i class="nav-icon fa fa-dashboard"></i>
                    <p>Grupos</p>
                </a>
            </li>
            <li class="nav-item">
                <a href="index.php?action=lugares_admin" class="nav-link">
                    <i class="nav-icon fa fa-dashboard"></i>
                    <p>Lugares</p>
                </a>
            </li>

            <li class="nav-item">
                <a onclick="confirmarSesion();" href="index.php?action=salir"
```

Función que se encarga de mostrar la navegación en caso de que el administrador haya iniciado sesión. Mostrando los módulos correspondientes.

```php
public function borrarAlumnaController()
{
    if(isset($_GET['idBorrar']))
    {
        $datosController = $_GET['idBorrar'];
        $respuesta = Datos::borrarAlumnaModel($datosController,'alumna');

        if($respuesta == "success")
        {
            echo"<script>
                    window.location = 'index.php?action=alumnas';
                </script>";
        }
    }
}
```

Función que se encarga de mandar llamar el modelo para efectuar la sentencia SQL para borrar una alumna nueva.

```php
public function actualizarPagoController()
{
    if(isset($_POST['mamaEditar']))
    {
        $fecha_envio = date ("Y-m-d H:i:s",strtotime($_POST['fecha_envio']));
        $fecha_pago = date("Y-m-d",strtotime($_POST['fecha_pago']));

        $datosController = array( "id"=>$_POST["idEditar"],
                                  "mama"=>$_POST["mamaEditar"],
                                  "fecha_pago"=>$fecha_pago,
                                  "fecha_envio"=>$fecha_envio,
                                  "folio"=>$_POST['folioEditar']);

        $respuesta = Datos::actualizarPagoModel($datosController,"pagos");

        if($respuesta == "success")
        {

            echo"<script>
                    window.location = 'index.php?action=lugares_admin';
                </script>";

        }

        else
        {
```

Función que se encarga llamar el modelo, para efectuar la sentencia SQL que actualizará los datos del pago en base al id del pago que se eligió y los datos que se cambiaron en el formulario

```php
public function actualizarGrupoController()
{
    if(isset($_POST['nombreEditar']))
    {
        $datosController = array( "id"=>$_POST["idEditar"],
                                  "nombre"=>$_POST["nombreEditar"]);

        $respuesta = Datos::actualizarGrupoModel($datosController,"grupo");

        if($respuesta == "success")
        {

            echo"<script>
                    window.location = 'index.php?action=grupos';
                </script>";

        }

        else
        {

            echo "error";

        }

    }
}
```

Función del controlador que se encarga de llamar al modelo que efectuará la sentencia SQL que se encargara de actualizar los datos del grupo en base al id del registro que se eligió y los cambios que hubo en el formulario.

```php
public function ingresoUsuarioController(){

    if(isset($_POST["email"]) && !empty($_POST['email'])){

        $datosController = array( "email"=>$_POST["email"],
                                  "password"=>$_POST['password']);

        $respuesta = Datos::ingresoUsuarioModel($datosController, "usuario");
        //Valiación de la respuesta del modelo para ver si es un usuario correcto.

        //Si los datos traidos de la base de datos son iguales a los que el usuario
        ingreso en las cajas de texto
        if($respuesta)
        {

            //Entonces la variables de sesion se encienden
            $_SESSION["validar"] = true;
            $_SESSION['correo'] = $respuesta['email'];
            $_SESSION['password']=$respuesta['password'];
            echo"<script>
                        window.location = 'index.php?action=dashboard';
                    </script>";

        }
        else
        {
            echo '<script>
                    swal({title: "Error",
                          text: "Usuario o contraseña erróneos!",
                           type: "error"});
                    </script>';
        }
    }
```

Función del controlador que se encarga del inicio de sesión, se encenderán ciertos variables de sesión que se encargarán de tener el control para ingresar a los módulos de administrador siempre y cuando se haya iniciado sesión

```php
public function checkGuess()
{
    if(isset($_GET['action']))
    {
        if($_GET['action']=='login' || $_GET['action']=='lugares')
        {
            $_SESSION['control']=0;
        }
    }
    else
    {
        $_SESSION['control']=0;
    }
}
```

Función del controlador que se encarga de saber cuando se inicio sesión o cuando ingreso como un invitado (Al registro de pagos). Y así saber cuando mostrar la barra de módulos

```php
public function actualizarAlumnaController()
{
    if(isset($_POST['nombreEditar']))
    {
        $fecha =  date("Y-m-d",strtotime($_POST['fechaEditar']));
        $datosController = array( "id"=>$_POST["idEditar"],
                                  "nombre"=>$_POST["nombreEditar"],
                                  "apellido"=>$_POST["apellidoEditar"],
                                  "fecha"=>$_POST["fechaEditar"],
                                  "grupo"=>$_POST['grupo']);

        $respuesta = Datos::actualizarAlumnaModel($datosController,"alumna");

        if($respuesta == "success")
        {
            echo"<script>
                    window.location = 'index.php?action=alumnas';
                </script>";

        }

        else
        {

            echo "error";

        }

    }
}
```

Función que se encarga de actualizar un alumno, en base al id del registro que se eligió y obteniendo los datos que se cambiaron de su respectivo formulario.