Examen Unidad 2 (Manual Técnico)

CONTROLADOR.

Empezando con una clase controlador que lleva por nombre “MvcController” la cual es la que se encarga de contener todas las funciones que se ejecutaran dentro de las vistas para poder llevar a acabo un correcto funcionamiento de las interfaces. Utilizando las funciones utilizadas en el modelo.

La funcion “page( )” se encarga de incluir los estilos en el controlador para asi poder utilzarlos en las vistas. El archivo “template.php” se incluye con el comando “include”.

Funcion “linkPageController( )” se encarga de llevar el control de las paginas dependiendo del valor que se le de a la variable “action” la cual es mandada por la url.

```php
<?php
    class MvcController{

        //Controlador para incluir el template en el index y mostrarlo en todas las paginas
        public static function page(){
            include "views/template.php";
        }

        //Controlador que llevara a cabo el mandejo de las direcciones de las paginas mediante acciones
        public static function linkPageController(){
            if(isset($_GET["action"])){
                $link = $_GET["action"];
            }else{
                $link = "index";
            }
            $answer = Pages::linkPageModel($link);
            include $answer;
        }
```

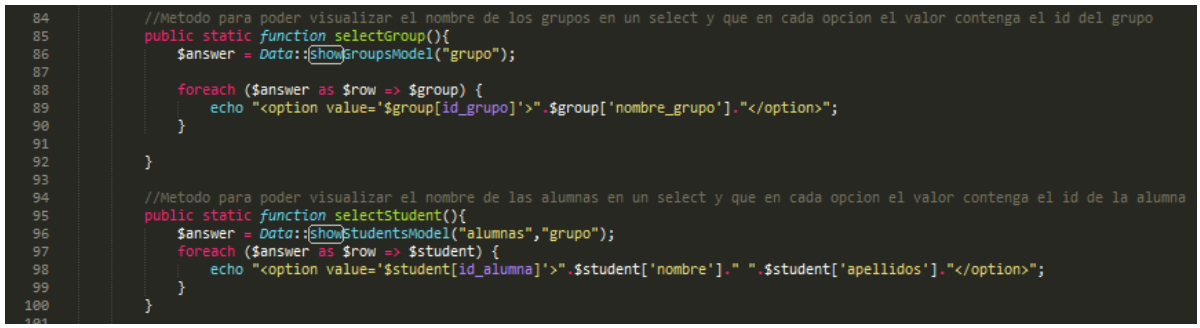
La función “showPaymentController( )” hara que los datos de los pagos se puedan visualizar dentro de una tabla. Dependiendo de quien este visualizando los datos se podran ver algunos. Si un usuario ordinario visualiza estos datos solo podra ver parte de ellos, pero no todos, ya que el Superadmin es quien podra visualizar e interactuar con todos los datos del pago. Para esto se hace una condicion para determinar que campos se deben de mostrar, mediante una variabe de sesion, se validara si esta ingresado el superadmin o no.

```php
        //Metodo que mostrara la informacion de los pagos en una tabla, utilizando el modelo que recibira como parametros las tablas de pagos, alumnas y grupo
        adecuadamente la informacion
        public static function showPaymentsController(){
            $answer = Data::showPaymentsModel("pagos","alumnas","grupo");
            foreach ($answer as $row => $pagos) {
                echo'
                    <tr align="center">
                        <td>'.$pagos['id_pago'].'</td>
                        <td>'.$pagos['nombre_grupo'].'</td>
                        <td>'.$pagos['nombre'].' '.$pagos['apellidos'].'</td>
                        <td>'.$pagos['nombreMama'].' '.$pagos['apellidoMama'].'</td>
                        <td>'.$pagos['fechaEnvio'].'</td>
                        <td>'.$pagos['fechaPago'].'</td>';
                        if(isset($_SESSION["user"])){
                            echo '
                                <td><input type="button" value="ver" onclick="verImagen()"></td>
                                <td>'.$pagos['folio'].'</td>
                                <td><a href="index.php?action=editarLugar&id_pago='.$pagos['id_pago'].'" class="btn btn-info">Editar Fecha de Pago</a></td>
                                <td><a href="index.php?action=eliminarLugar&id_pago='.$pagos['id_pago'].'" class="btn btn-danger">Eliminar Pago</a></td>
                            ';
                            $_SESSION["img"] = $pagos["imagenFolio"];
                            echo $_SESSION["img"]."<br>";
                        }
                echo'</tr>';
            }
            echo"<script type='text/javascript'>
                function verImagen(){
                    var img = <?php echo $_SESSION[img]?>';
                    swal({
                        title: 'Comprobante',
                        imageUrl: img,
                        imageWidth: 400,
                        imageHeight: 200
                    });
                }

                </script>";

        }
```

“addPaymentController( )” realiza el agregado de un pago a la base de datos. Para poder agregar una imagen se utiliza la función “move_uploaded_file ( )” que realiza el movimiento de dirección del archivo que se haya subido. Después dentro de un arreglo llamado “$data” se agregaran los datos que se llenaron en el formulario para registrar un pago y serán mandados como parámetros al modelo de “addPaymentModel()”. Si este modelo retorna “success” se determinara que el agregado del pago fue realizado con éxito, de lo contrario marcara un error.

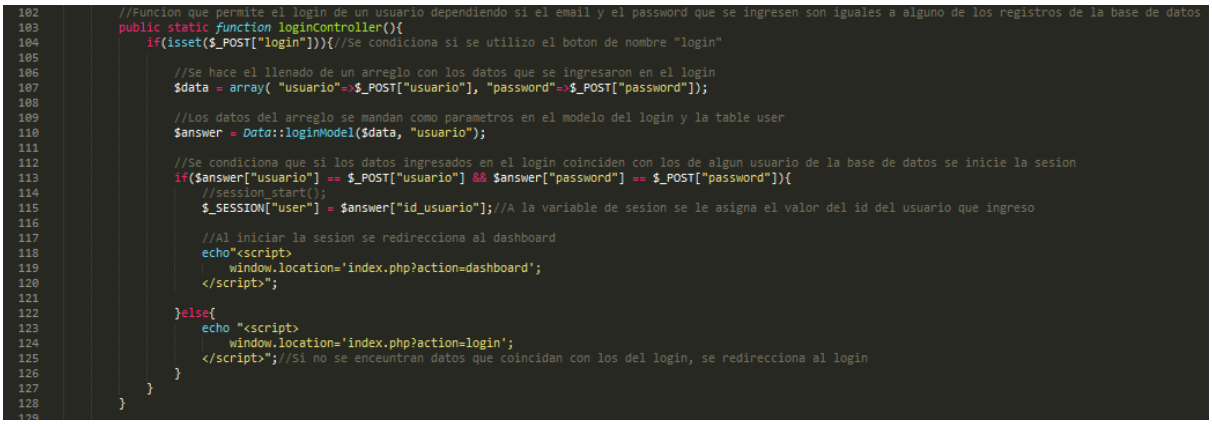
```
58
59      //Metodo que tomara la informacion del formulario mediante metodo Post y la agregara a un arreglo que sera mandado como parametro al modelo y poder ejecutar la insercion en
        la base de datos
60      public static function addPaymentController(){
61          if(isset($_POST["aceptar"])){
62
63              $dir = "views/dist/img/";
64              $file = $_FILES["imagenFolio"]["name"];
65              $dirFile = $dir.$file;
66              $tempFile = $_FILES["imagenFolio"]["tmp_name"];
67              echo $dirFile;
68              move_uploaded_file($tempFile,$dirFile);
69
70          // Check if image file is a actual image or fake image
71              $data = array("grupo" => $_POST["grupo"], "alumna"=>$_POST["alumna"], "nombreMama"=>$_POST["nombreMama"], "apellidoMama"=>$_POST["apellidoMama"], "fechaPago" => $_POST
                    ["fechaPago"], "fechaEnvio" =>date("Y-m-d H:i:s"), "imagenFolio" =>$dirFile, "folio" =>$_POST["folio"]);
72              $answer = Data::addPaymentModel($data,"pagos");
73
74              if($answer == "success"){
75                  echo "<script>
76                      window.location = 'index.php?action=lugares'
77                      </script>";
78              }else{
79                  echo "error";
80              }
81          }
82      }
```

Funciones ”selectGroup” y “selectStudent” se encargan de mostrar en un select o combo box todos los grupos y estudiantes respectivamente, teniendo como valor en cada opcion el id de cada uno de los registros de la base de datos que se encuentren en las tablas grupo y alumnas.



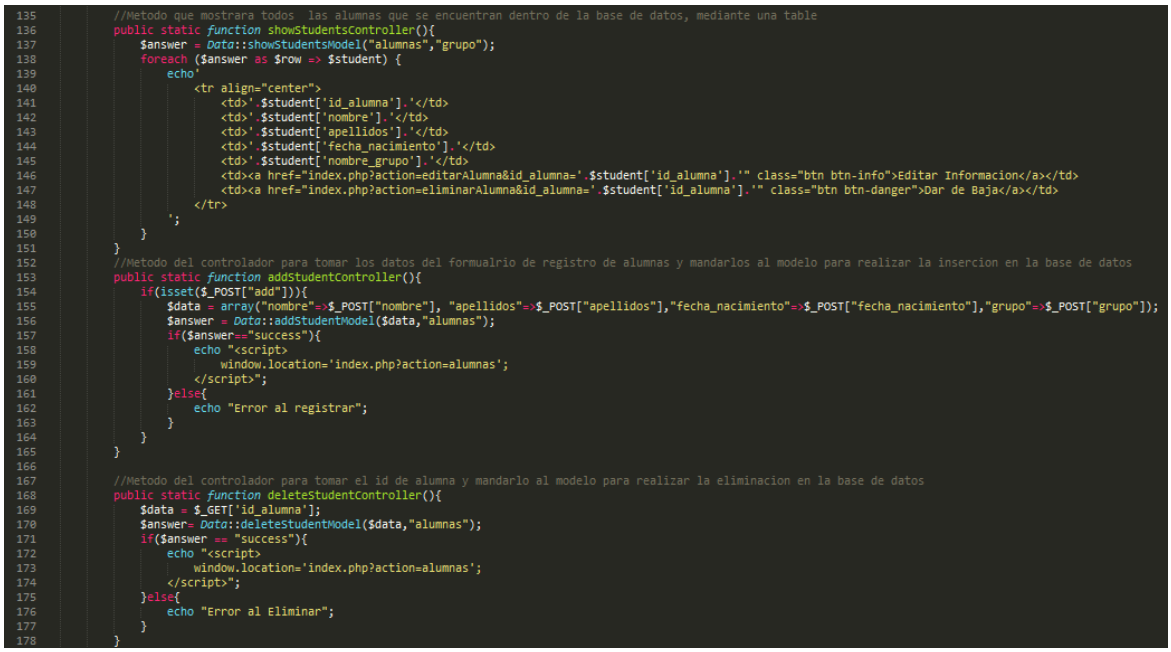
```
84      //Metodo para poder visualizar el nombre de los grupos en un select y que en cada opcion el valor contenga el id del grupo
85      public static function selectGroup(){
86          $answer = Data::showGroupsModel("grupo");
87
88          foreach ($answer as $row => $group) {
89              echo "<option value='$group[id_grupo]'>".$group['nombre_grupo']."</option>";
90          }
91
92      }
93
94      //Metodo para poder visualizar el nombre de las alumnas en un select y que en cada opcion el valor contenga el id de la alumna
95      public static function selectStudent(){
96          $answer = Data::showStudentsModel("alumnas","grupo");
97          foreach ($answer as $row => $student) {
98              echo "<option value='$student[id_alumna]'>".$student['nombre']." ".$student['apellidos']."</option>";
99          }
100     }
```

loginController( ) se encarga de realizar el ingreso del usuario a la base de datos, comparando los valores de los campos que se ingresaron en el formulario del login con los que se encuentran dentro de la tabla usuario en la base de datos, si estos coinciden, se iniciara una sesión y se declarara una variable de sesión la cual contendrá el id del usuario.



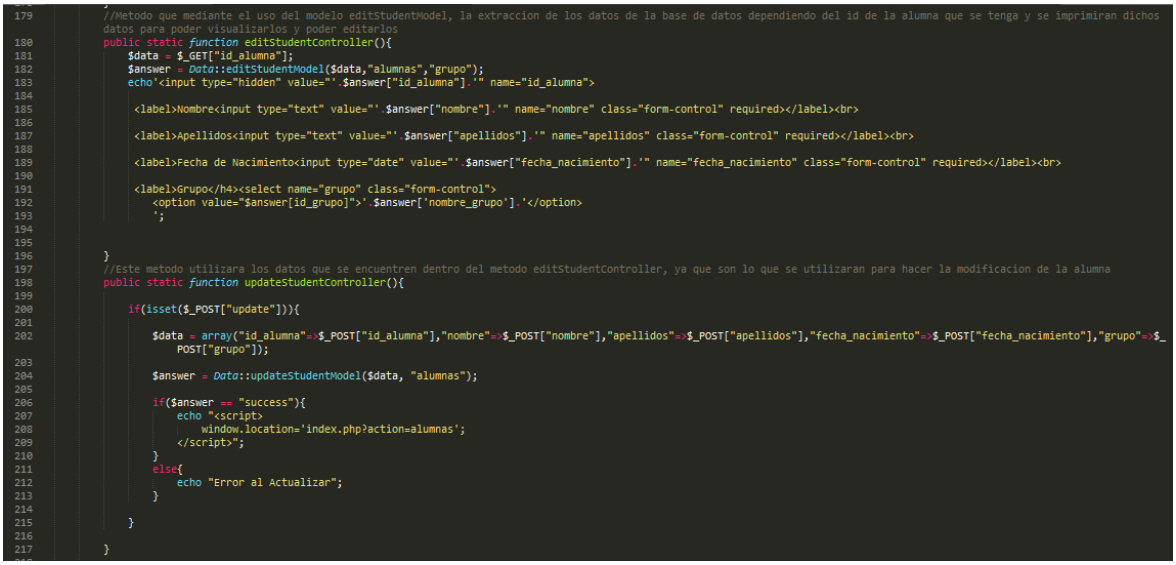
```
102     //Funcion que permite el login de un usuario dependiendo si el email y el password que se ingresen son iguales a alguno de los registros de la base de datos
103     public static function loginController(){
104         if(isset($_POST["login"])){//Se condiciona si se utilizo el boton de nombre "login"
105
106             //Se hace el llenado de un arreglo con los datos que se ingresaron en el login
107             $data = array( "usuario"=>$_POST["usuario"], "password"=>$_POST["password"]);
108
109             //Los datos del arreglo se mandan como parametros en el modelo del login y la table user
110             $answer = Data::loginModel($data, "usuario");
111
112             //Se condiciona que si los datos ingresados en el login coinciden con los de algun usuario de la base de datos se inicie la sesion
113             if($answer["usuario"] == $_POST["usuario"] && $answer["password"] == $_POST["password"]){
114                 //session_start();
115                 $_SESSION["user"] = $answer["id_usuario"];//A la variable de sesion se le asigna el valor del id del usuario que ingreso
116
117                 //Al iniciar la sesion se redirecciona al dashboard
118                 echo"<script>
119                     window.location='index.php?action=dashboard';
120                 </script>";
121
122             }else{
123                 echo "<script>
124                     window.location='index.php?action=login';
125                 </script>";//Si no se enceuntran datos que coincidan con los del login, se redirecciona al login
126             }
127         }
128     }
```

Dentro de este apartado se explicaran las funciones del controlador que se utilizaran para el manejo de los datos de las alumnas. La función “showStudentController( )” mostrara dentro de una tabla todos los datos que se encuentran en la base de datos de todas las alumnas, aparte de dos botones los cuales tendrán la función de enviar el id de la alumna mediante url para poder darle edición a los datos de la alumna o eliminarla. La función “addStudentController( )” se encarga de tomar los datos del formulario para agregar una alumna y mandarlos al modelo para realizar la inserción de una nueva alumna en la base de datos. La función “deleteStudentController()” se encarga de recibir el id de la alumna y mandarlo como parámetro al modelo y poder realizar la eliminación de la alumna de la base de datos.



```php
    //Metodo que mostrara todos  las alumnas que se encuentran dentro de la base de datos, mediante una table
    public static function showStudentsController(){
        $answer = Data::showStudentsModel("alumnas","grupo");
        foreach ($answer as $row => $student) {
            echo'
                <tr align="center">
                    <td>'.$student['id_alumna'].'</td>
                    <td>'.$student['nombre'].'</td>
                    <td>'.$student['apellidos'].'</td>
                    <td>'.$student['fecha_nacimiento'].'</td>
                    <td>'.$student['nombre_grupo'].'</td>
                    <td><a href="index.php?action=editarAlumna&id_alumna='.$student['id_alumna'].'" class="btn btn-info">Editar Informacion</a></td>
                    <td><a href="index.php?action=eliminarAlumna&id_alumna='.$student['id_alumna'].'" class="btn btn-danger">Dar de Baja</a></td>
                </tr>
            ';
        }
    }
    //Metodo del controlador para tomar los datos del formualrio de registro de alumnas y mandarlos al modelo para realizar la insercion en la base de datos
    public static function addStudentController(){
        if(isset($_POST["add"])){
            $data = array("nombre"=>$_POST["nombre"], "apellidos"=>$_POST["apellidos"],"fecha_nacimiento"=>$_POST["fecha_nacimiento"],"grupo"=>$_POST["grupo"]);
            $answer = Data::addStudentModel($data,"alumnas");
            if($answer=="success"){
                echo "<script>
                    window.location='index.php?action=alumnas';
                </script>";
            }else{
                echo "Error al registrar";
            }
        }
    }

    //Metodo del controlador para tomar el id de alumna y mandarlo al modelo para realizar la eliminacion en la base de datos
    public static function deleteStudentController(){
        $data = $_GET['id_alumna'];
        $answer= Data::deleteStudentModel($data,"alumnas");
        if($answer == "success"){
            echo "<script>
                window.location='index.php?action=alumnas';
            </script>";
        }else{
            echo "Error al Eliminar";
        }
    }
```

La funcion editStudentController, mostrar todos los datos de la alumna dentro de campos de texto para poder editarlos, esto se hace mediante el uso del modelo editStudentModel() el cual recibirá el id de la alumna a editar, para así poder mostrar los datos de dicha alumna. La funcion updateStudentController( ) se encargara de realizar la actualización de los datos de la alumna, utilizando los datos que se obtienen de la funcion anterior y guardándolos en un arreglo el cual será mandando como parámetro dentro del modelo updateStudentModel().

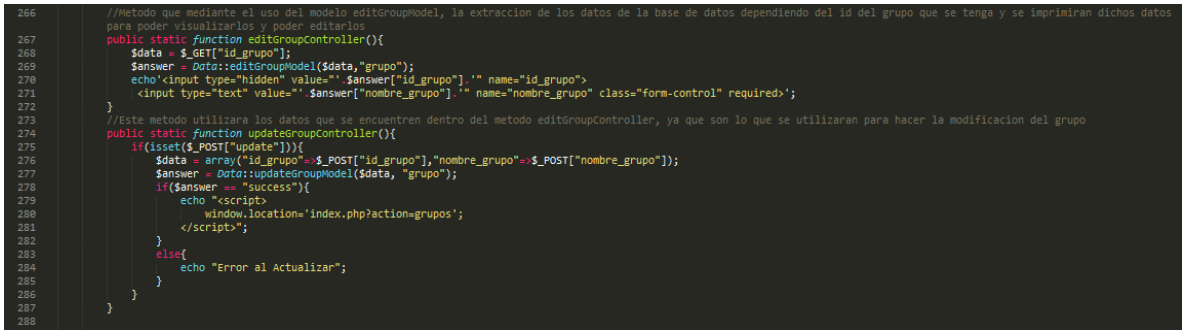


```php
    //Metodo que mediante el uso del modelo editStudentModel, la extraccion de los datos de la base de datos dependiendo del id de la alumna que se tenga y se imprimiran dichos
    datos para poder visualizarlos y poder editarlos
    public static function editStudentController(){
        $data = $_GET["id_alumna"];
        $answer = Data::editStudentModel($data,"alumnas","grupo");
        echo'<input type="hidden" value="'.$answer["id_alumna"].'" name="id_alumna">

            <label>Nombre<input type="text" value="'.$answer["nombre"].'" name="nombre" class="form-control" required></label><br>

            <label>Apellidos<input type="text" value="'.$answer["apellidos"].'" name="apellidos" class="form-control" required></label><br>

            <label>Fecha de Nacimiento<input type="date" value="'.$answer["fecha_nacimiento"].'" name="fecha_nacimiento" class="form-control" required></label><br>

            <label>Grupo</h4><select name="grupo" class="form-control">
               <option value="$answer[id_grupo]">'.$answer['nombre_grupo'].'</option>
               ';


    }
    //Este metodo utilizara los datos que se encuentren dentro del metodo editStudentController, ya que son lo que se utilizaran para hacer la modificacion de la alumna
    public static function updateStudentController(){

        if(isset($_POST["update"])){

            $data = array("id_alumna"=>$_POST["id_alumna"],"nombre"=>$_POST["nombre"],"apellidos"=>$_POST["apellidos"],"fecha_nacimiento"=>$_POST["fecha_nacimiento"],"grupo"=>$_
                POST["grupo"]);

            $answer = Data::updateStudentModel($data, "alumnas");

            if($answer == "success"){
                echo "<script>
                    window.location='index.php?action=alumnas';
                </script>";
            }
            else{
                echo "Error al Actualizar";
            }

        }

    }
```

Dentro de este apartado se explicaran las funciones del controlador que se utilizaran para el manejo de los datos de los grupos. La función “showGroupController( )” mostrara dentro de una tabla todos los datos que se encuentran en la base de datos de todas los grupos, aparte de dos botones los cuales tendrán la función de enviar el id del grupo mediante url para poder darle edición a los datos del grupo o eliminarlo. La función “addGroupController( )” se encarga de tomar los datos del formulario para agregar un grupo y mandarlos al modelo para realizar la inserción de uno nuevo grupo en la base de datos. La función “deleteGroupController()” se encarga de recibir el id del grupo y mandarlo como parámetro al modelo y poder realizar la eliminación del grupo de la base de datos.

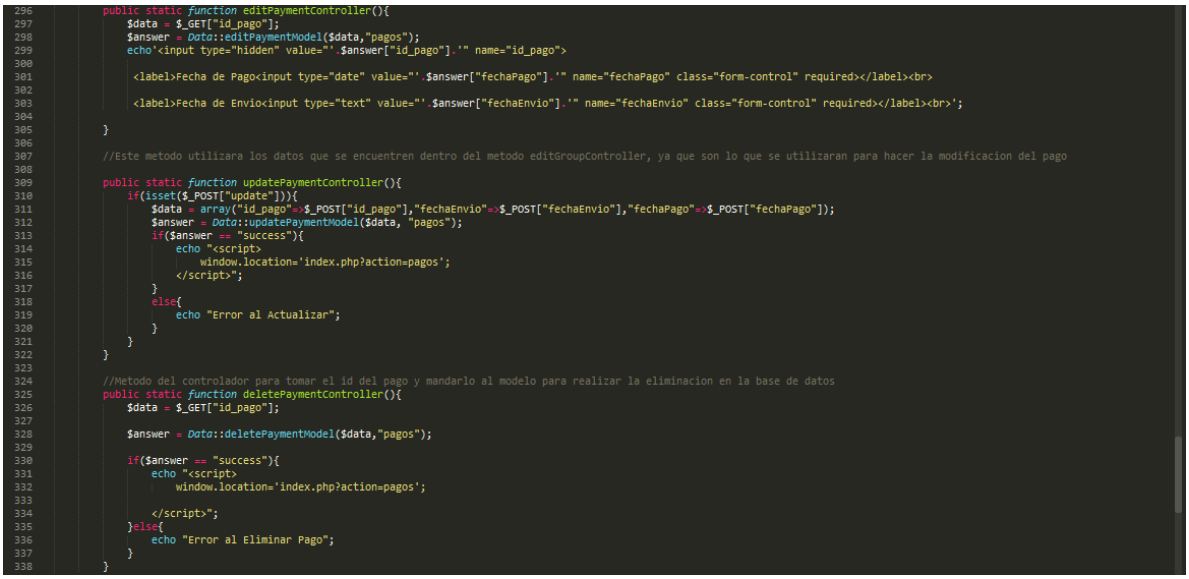
```php
//Metodo que mostrara todos  los grupos que se encuentran dentro de la base de datos, mediante una table
public static function showGroupsController(){
    $answer = Data::showGroupsModel("grupo");
    foreach ($answer as $row => $group) {
        echo'
            <tr align="center">
                <td>'.$group['id_grupo'].'</td>
                <td>'.$group['nombre_grupo'].'</td>
                <td><a href="index.php?action=editarGrupo&id_grupo='.$group['id_grupo'].'" class="btn btn-info">Editar Informacion</a></td>
                <td><a href="index.php?action=eliminarGrupo&id_grupo='.$group['id_grupo'].'" class="btn btn-danger">Quitar Grupo</a></td>
            </tr>

        ';
    }
}

//Metodo para acer la insercion de un grupo en la base de datos, utilizando el modelo de addGroupModel, ppara realizar la consulta en la base de datos e insertar un nuevo grupo
public static function addGroupController(){
    if(isset($_POST["add"])){
        $data = array("nombre_grupo"=>$_POST["nombre_grupo"]);
        $answer = Data::addGroupModel($data,"grupo");
        if($answer=="success"){
            echo "<script>
                window.location='index.php?action=grupos';
            </script>";
        }else{
            echo "Error al registrar";
        }
    }
}
//Metodo del controlador para tomar el id del grupo y mandarlo al modelo para realizar la eliminacion en la base de datos
public static function deleteGroupController(){
    $data = $_GET["id_grupo"];
    $answer = Data::deleteGroupModel($data,"grupo","alumnas","pagos");
    if($answer == "success"){
        echo "<script>
            window.location='index.php?action=grupos';
        </script>";
    }else{
        echo"Error al quitar el grupo";
    }
}
```

La funcion editGroupController, mostrar todos los datos del grupo dentro de campos de texto para poder editarlos, esto se hace mediante el uso del modelo editGroupModel() el cual recibirá el id del grupo a editar, para así poder mostrar los datos de dicho grupo. La funcion updateGroupController( ) se encargara de realizar la actualización de los datos del grupo, utilizando los datos que se obtienen de la función anterior y guardándolos en un arreglo el cual será mandando como parámetro dentro del modelo updateGroupModel().



```php
//Metodo que mediante el uso del modelo editGroupModel, la extraccion de los datos de la base de datos dependiendo del id del grupo que se tenga y se imprimiran dichos datos para poder visualizarlos y poder editarlos
public static function editGroupController(){
    $data = $_GET["id_grupo"];
    $answer = Data::editGroupModel($data,"grupo");
    echo'<input type="hidden" value="'.$answer["id_grupo"].'" name="id_grupo">
     <input type="text" value="'.$answer["nombre_grupo"].'" name="nombre_grupo" class="form-control" required>';
}
//Este metodo utilizara los datos que se encuentren dentro del metodo editGroupController, ya que son lo que se utilizaran para hacer la modificacion del grupo
public static function updateGroupController(){
    if(isset($_POST["update"])){
        $data = array("id_grupo"=>$_POST["id_grupo"],"nombre_grupo"=>$_POST["nombre_grupo"]);
        $answer = Data::updateGroupModel($data, "grupo");
        if($answer == "success"){
            echo "<script>
                window.location='index.php?action=grupos';
            </script>";
        }
        else{
            echo "Error al Actualizar";
        }
    }
}
```

La función editPaymentController, mostrar todos los datos del pago dentro de campos de texto para poder editarlos, esto se hace mediante el uso del modelo editPaymentModel() el cual recibirá el id del pago a editar, para así poder mostrar los datos de dicho pago. La función updatePaymentController( ) se encargara de realizar la actualización de los datos del pago, utilizando los datos que se obtienen de la función anterior y guardándolos en un arreglo el cual será mandando como parámetro dentro del modelo updatePaymentModel().La función “deletePaymentController()” se encarga de recibir el id del pago y mandarlo como parámetro al modelo y poder realizar la eliminación del pago de la base de datos.



```php
    public static function editPaymentController(){
        $data = $_GET["id_pago"];
        $answer = Data::editPaymentModel($data,"pagos");
        echo'<input type="hidden" value="'.$answer["id_pago"].'" name="id_pago">

         <label>Fecha de Pago<input type="date" value="'.$answer["fechaPago"].'" name="fechaPago" class="form-control" required></label><br>

         <label>Fecha de Envio<input type="text" value="'.$answer["fechaEnvio"].'" name="fechaEnvio" class="form-control" required></label><br>';

    }

    //Este metodo utilizara los datos que se encuentren dentro del metodo editGroupController, ya que son lo que se utilizaran para hacer la modificacion del pago

    public static function updatePaymentController(){
        if(isset($_POST["update"])){
            $data = array("id_pago"=>$_POST["id_pago"],"fechaEnvio"=>$_POST["fechaEnvio"],"fechaPago"=>$_POST["fechaPago"]);
            $answer = Data::updatePaymentModel($data, "pagos");
            if($answer == "success"){
                echo "<script>
                    window.location='index.php?action=pagos';
                </script>";
            }
            else{
                echo "Error al Actualizar";
            }
        }
    }

    //Metodo del controlador para tomar el id del pago y mandarlo al modelo para realizar la eliminacion en la base de datos
    public static function deletePaymentController(){
        $data = $_GET["id_pago"];

        $answer = Data::deletePaymentModel($data,"pagos");

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

            </script>";
        }else{
            echo "Error al Eliminar Pago";
        }
    }
```

MODELO.

El modelo es el que realizara todas las acciones que tienen que ver con la base de datos, ya que este es el que interactúa con ella, realiza todas las consultas que sean necesarias para que los controladores puedan funcionar de una manera adecuada.

Comenzando extendiendo de la clase Connection la cual contiene la conexión con la base de datos. Procediendo al método de showPaymentModel, que su acción será realizar una consulta a la base de datos obteniendo tres tablas, realizando inner join con dicha tabla para poder obtener los datos que se necesitan para mostrar la información de lo pagos.

Después se encuentra el método de addPaymentModel(), el cual realiza la inserción de un pago, recibiendo como parámetros los datos y la tabla a la cual se hará la inserción, mandando como datos para la consulta los datos que fueron recibidos como parámetros.

```php
<?php

    require_once "connection.php";
    class Data extends Connection{

        //Funcion del crud, modelo para mostrar los pagos que se hhan realizado
        public static function showPaymentsModel($table,$table2,$table3){
            $statement = Connection::connect()->prepare("SELECT * from $table as p INNER JOIN $table2 as a on p.alumna = a.id_alumna INNER JOIN $table3 as g on a.grupo = g.id_grupo
                ORDER BY fechaPago asc");
            $statement->execute();
            return $statement->fetchAll();
            $statement->close();
        }

        //Metodo para tomar los datos del arreglo posteado en el controlador y poder hacer la insercion en a base de datos, tomando dichos post como bindParam y registrarlos mediante
        PDO;
        public static function addPaymentModel($data,$table){


            $statement = Connection::connect()->prepare("INSERT INTO $table(grupo,alumna,nombreMama,apellidoMama,fechaPago,fechaEnvio,imagenFolio,folio) VALUES (:grupo, :alumna,
                :nombreMama,:apellidoMama, :fechaPago, :fechaEnvio, :imagenFolio ,:folio)");
            $statement->bindParam(":grupo",$data['grupo'],PDO::PARAM_INT);
            $statement->bindParam(":alumna",$data['alumna'],PDO::PARAM_INT);
            $statement->bindParam(":nombreMama",$data['nombreMama'],PDO::PARAM_STR);
            $statement->bindParam(":apellidoMama",$data['apellidoMama'],PDO::PARAM_STR);
            $statement->bindParam(":fechaPago",$data['fechaPago'],PDO::PARAM_STR);
            $statement->bindParam(":fechaEnvio",$data['fechaEnvio'],PDO::PARAM_STR);
            $statement->bindParam(":imagenFolio",$data["imagenFolio"],PDO::PARAM_STR);
            $statement->bindParam(":folio",$data['folio'],PDO::PARAM_INT);
            if($statement->execute()){
                return "success";
            }else{
                return "fail";
            }
            $statement->close();
        }
```

El metodo showStudentModel() realiza la funcion de traer todos los datos de la tabla alumnos y hacer un inner join con la tabla grupo para obtener el nombre del grupo que se relaciona con cada una de las alumnas. Al terminar la consulta se retornara un arreglo con todos los datos, esto mediante la funcion fetchAll().

El metodo addStudentModel() realiza la insercion a la base de datos de una alumna. Tomando los datos de un formulario y mandandolos como los datos que recibiran los campos de la base de datos.

El metodo editStudentModel, realiza una consulta la cual mediante el id de la alumna, retornara todos sus datos para asi poder dar paso a su proxima actualizacion o edición.

El metodo updateStudentModel(), utiliza los datos que se obtuvieron del metodo anterior y realiza la actualizacion de la alumna.

```php
    //Metodo para mostrar la informacion de todas las alumnas de la academia
    public static function showStudentsModel($table,$table2){
        $statement = Connection::connect()->prepare("SELECT * FROM $table as a INNER JOIN $table2 as g on a.grupo = g.id_grupo");
        $statement->execute();
        return $statement->fetchAll();
        $statement->close();
    }

    //Metodo para realizar la consulta en la base de datos y poder realizar la insercion en la tabla alumnas, los datos a insertar seran el arreglo que se recibira como parametro.
    public static function addStudentModel($data,$table){
        $statement = Connection::connect()->prepare("INSERT INTO $table(nombre,apellidos,fecha_nacimiento,grupo) VALUES (:nombre,:apellidos,:fecha_nacimiento,:grupo)");
        $statement ->bindParam(":nombre",$data["nombre"],PDO::PARAM_STR);
        $statement ->bindParam(":apellidos",$data["apellidos"],PDO::PARAM_STR);
        $statement ->bindParam(":fecha_nacimiento",$data["fecha_nacimiento"],PDO::PARAM_STR);
        $statement ->bindParam(":grupo",$data["grupo"],PDO::PARAM_INT);
        if($statement->execute()){
            return "success";
        }else{
            return "fail";
        }
    }

    //Metodo para realizar la consulta que devolvera los datos de la alumna la cual se desea editar
    public static function editStudentModel($data,$table,$table2){
        $statement = Connection::connect()->prepare("SELECT * FROM $table as a INNER JOIN $table2 as g on a.grupo = g.id_grupo WHERE id_alumna = :id_alumna");
        $statement->bindParam(":id_alumna",$data,PDO::PARAM_INT);
        $statement->execute();
        return $statement->fetch();
    }

    //Metodo para editar los datos de la alumna dependiendo de los datos que se obtenieron en la funcion editStudentModel.
    public static function updateStudentModel($data,$table){
        $statement = Connection::connect()->prepare("UPDATE $table SET nombre = :nombre, apellidos=:apellidos, fecha_nacimiento = :fecha_nacimiento,grupo = :grupo WHERE id_alumna = :id_alumna");
        $statement->bindParam(":nombre",$data["nombre"],PDO::PARAM_STR);
        $statement->bindParam(":apellidos",$data["apellidos"],PDO::PARAM_STR);
        $statement->bindParam(":fecha_nacimiento",$data["fecha_nacimiento"],PDO::PARAM_STR);
        $statement->bindParam(":grupo",$data["grupo"],PDO::PARAM_INT);
        $statement->bindParam(":id_alumna",$data["id_alumna"],PDO::PARAM_INT);
        if($statement->execute()){
            return "success";
        }else{
            return "fail";
        }
    }
```

deleteStudentModel(), realiza la eliminacion de una alumna mediante su id.

El metodo showGroupModel() realiza la funcion de traer todos los datos de la tabla grupo. Al terminar la consulta se retornara un arreglo con todos los datos, esto mediante la funcion fetchAll().

El metodo addGroupModel() realiza la insercion a la base de datos de un grupo. Tomando los datos de un formulario y mandandolos como los datos que recibiran los campos de la base de datos.

El metodo ediGroupModel, realiza una consulta la cual mediante el id de un grupo, retornara todos sus datos para asi poder dar paso a su proxima actualizacion o edición.

```php
    //Metodo que realizara la eliminacion de la alumna dependiendo del id que se haya recibido como parametro
    public static function deleteStudentModel($data,$table){
        $statement = Connection::connect()->prepare("DELETE FROM $table WHERE id_alumna = :id_alumna");
        $statement->bindParam(":id_alumna",$data,PDO::PARAM_INT);
        if($statement->execute()){
            return "success";
        }else{
            return "fail";
        }
    }

    //Metodo para mostrar la informacion de todos los grupos de la academia
    public static function showGroupsModel($table){
        $statement = Connection::connect()->prepare("SELECT * FROM $table");
        $statement->execute();
        return $statement->fetchAll();
        $statement->close();
    }

    //Metodo para realizar la inserion de un grupo en la base de datos,.

    public static function addGroupModel($data,$table){
        $statement = Connection::connect()->prepare("INSERT INTO $table(nombre_grupo) VALUES (:nombre_grupo)");
        $statement ->bindParam(":nombre_grupo",$data["nombre_grupo"],PDO::PARAM_STR);

        if($statement->execute()){
            return "success";
        }else{
            return "fail";
        }
    }
    //Metodo para realizar la consulta que devolvera los datos del grupo la cual se desea editar
    public static function editGroupModel($data,$table){
        $statement = Connection::connect()->prepare("SELECT * FROM $table WHERE id_grupo = :id_grupo");
        $statement->bindParam(":id_grupo",$data,PDO::PARAM_INT);
        $statement->execute();
        return $statement->fetch();
    }
```

El metodo updateGroupModel(), utiliza los datos que se obtuvieron del metodo anterior y realiza la actualizacion del grupo. DeleteStudentModel(), realiza la eliminacion de un grupo mediante su id. El metodo editPaymentModel, realiza una consulta la cual mediante el id de un pago, retornara todos sus datos para asi poder dar paso a su proxima actualizacion o edición.

```php
        //Metodo para editar los datos del grupo dependiendo de los datos que se obtenieron en la funcion editStudentModel.
        public static function updateGroupModel($data,$table){
            $statement = Connection::connect()->prepare("UPDATE $table SET nombre_grupo = :nombre_grupo WHERE id_grupo = :id_grupo");
            $statement->bindParam(":nombre_grupo",$data["nombre_grupo"],PDO::PARAM_STR);
            $statement->bindParam(":id_grupo",$data["id_grupo"],PDO::PARAM_INT);
            if($statement->execute()){
                return "success";
            }else{
                return "fail";
            }
        }

        //Metodo que realizara la eliminacion del grupodependiendo del id que se haya recibido como parametro
        public static function deleteGroupModel($data,$table,$table2,$table3){

            $statement = Connection::connect()->prepare("DELETE FROM $table3 WHERE grupo = :id_grupo");
            $statement->bindParam(":id_grupo",$data,PDO::PARAM_INT);
            $statement->execute();

            $statement2 = Connection::connect()->prepare("DELETE FROM $table2 WHERE grupo = :id_grupo");
            $statement2->bindParam(":id_grupo",$data,PDO::PARAM_INT);
            $statement2->execute();

            $statement3 = Connection::connect()->prepare("DELETE FROM $table WHERE id_grupo = :id_grupo");
            $statement3->bindParam(":id_grupo",$data,PDO::PARAM_INT);
            $statement3->execute();
            return "success";

        }
        //Metodo para realizar la consulta que devolvera los datos del pago la cual se desea editar
        public static function editPaymentModel($data,$table){
            $statement = Connection::connect()->prepare("SELECT * FROM $table WHERE id_pago = :id_pago");
            $statement->bindParam(":id_pago",$data,PDO::PARAM_INT);
            $statement->execute();
            return $statement->fetch();
            $statement->close();
        }
```

El metodo updatePaymentModel(), utiliza los datos que se obtuvieron del metodo anterior y realiza la actualizacion del pago. DeletePaymentModel(), realiza la eliminacion de un pago mediante su id.

El metodo de loginModel() recibe como parametros la tabla usuario y los datos que se obtuvieron del formulario del login, que son el usuario y el password y este hace una consulta con los datos que se pasaron por el formulario.

```php

        //Metodo para editar los datos del pago dependiendo de los datos que se obtenieron en la funcion editPaymentModel.

        public static function updatePaymentModel($data,$table){
            $statement = Connection::connect()->prepare("UPDATE $table SET fechaEnvio=:fechaEnvio,fechaPago=:fechaPago WHERE id_pago=:id_pago");
            $statement->bindParam(":fechaEnvio",$data["fechaEnvio"],PDO::PARAM_STR);
            $statement->bindParam(":fechaPago",$data["fechaPago"],PDO::PARAM_STR);
            $statement->bindParam(":id_pago",$data["id_pago"],PDO::PARAM_INT);
            if($statement->execute()){
                return "success";
            }else{
                return "fail";
            }
        }


        //Metodo que realizara la eliminacion del pago dependiendo del id que se haya recibido como parametro

        public static function deletePaymentModel($data,$table){
            $statement = Connection::connect()->prepare("DELETE FROM $table WHERE id_pago = :id_pago");
            $statement->bindParam(":id_pago",$data,PDO::PARAM_INT);
            if($statement->execute()){
                return "success";
            }else{
                return "fail";
            }
        }

        //Metodo que recibira los datos que se ingresaron en el formulario del login para comparar si son iguales a los de algun registro de la tabla usuario en la base de datos
        public static function loginModel($data, $table){
            $statement = Connection::connect() -> prepare("SELECT * FROM $table WHERE usuario = :usuario and password = :password");
            $statement -> bindParam(":usuario",$data['usuario'],PDO::PARAM_STR);
            $statement -> bindParam(":password",$data['password'],PDO::PARAM_STR);
            $statement -> execute();
            return $statement -> fetch();
            $statement -> close();
        }


    }
?>
```