Integrantes de equipo

Montelongo Padilla
Hinojosa Tijerina
Torres Paz

MANUAL TÉCNICO

SISTEMA DE CAI

TECNOLOGÍAS Y APLICACIONES WEB
M.S.I. Mario Huberto Rodríguez Chávez
ING. EN TECNOLOGÍAS DE LA INFORMACIÓN
UNIVERSIDAD POLITÉCNICA DE VICTORIA

Soporte técnico

Llame al 8341840273

C O N T E N I D O

E S T R U C T U R A

Nombre del sitio: **Sistema de Centro de Aprendizaje de Inglés**
Dirección url: **http://206.189.123.185/Practica15/**
Lenguajes de desarrollo en los que fue programado: **PHP, JAVASCRIPT y AJAX**
Gestor de sistemas de base de datos: **MYSQL**
Herramientas de diseño utilizadas en la estructura visual: **HTML y BOOTSTRAP**
Patrón de arquitectura de software utilizado: **MODELO-VISTA-CONTROLADOR**
Contenido del proyecto:

- **Carpeta: controllers**
  - Archivo php: controlador.php
- **Carpeta: models**
  - Archivo php: check.php
  - Archivo php: conexión.php
  - Archivo php: crud.php
  - Archivo php: enlaces.php
- **Carpeta: views**
  - Carpeta: css
  - Carpeta: js
  - Carpeta: dist
  - Carpeta: plugins
  - Carpeta: modules
    - 25 archivos php que contienen la vista de cada elemento que conforma el sitio
  - 5 imágenes utilizadas en el diseño del sitio
  - Archivo php: template.php
- **Carpeta: uploads**
  - Imágenes que se almacenan en registros del sitio
- **Archivo php: index.php**
- **Archivo cai.sql (script de la base de datos)**

DIAGRAMA DE BASE DE DATOS

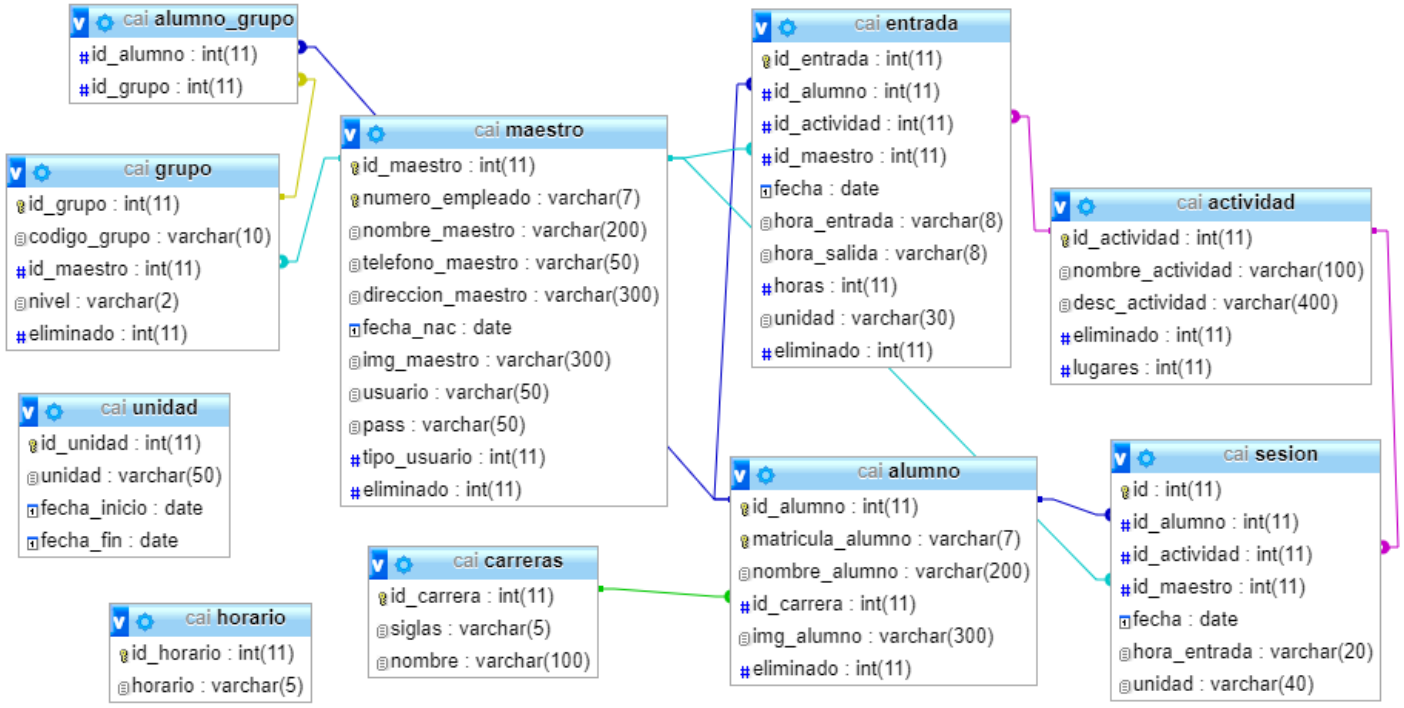


Diagrama de la base de datos.

Template (views/template.php)

Este archivo contiene etiquetas HTML en donde se importan los archivos necesarios para la visualización de los elementos gráficos del sitio que se encuentran en la misma carpeta que el archivo (css, js, dist y plugins), además de contener el esqueleto de una página web con código PHP que condiciona qué es lo que visualizará el usuario.

```
1. <body style="background-color: #E8ECEF">
2.       <?php
3.         //se validan los tipos de action que hay ya que existen dos
   menus de navegacion diferentes
4.         if(isset($_GET["action"]) && $_GET["action"]!="salir" && $_GE
   T["action"]!="error" && $_GET["action"]!="iniciousuario" && $_GET["ac
   tion"]!="sesiones2" && $_GET["action"]!="nuevasesion2"){
5.             include("modules/navegacion.php");
6.         }else{
7.           if(isset($_GET["action"])){
8.
9.             if($_GET["action"]=="iniciousuario" || $_GET["action"]=="
   sesiones2" || $_GET["action"]=="nuevasesion2"){
10.                include("modules/navegacion2.php");
11.              }
12.            }
13.          }
14.        ?>
15.        <?php
16.          session_start();
17.          //Se crea una instancia del controlador
18.          $mvc = new MvcController();
19.          //Se manda a llamar el controlador de las paginas
20.          $mvc->enlacesPaginasController();
21.        ?>
22.    </body>
```

A lo largo del documento se mostrará el uso de la variable “action” mediante la url de la página, que funcionará como bandera para saber qué es lo que desea ver el usuario. Dependiendo de su valor, en este archivo se condiciona si el usuario es tipo administrador se incluirá un archivo que contiene un menú de navegación con

opciones especiales para ese usuario, si no, se mostrará otro archivo de navegación diferente. Al terminar esto, se iniciará una sesión mediante php y se creará una nueva instancia del controlador, que contiene la validación de un tipo de usuario, y que en base a el se mostrará el contenido del módulo que el usuario seleccione, es por esto que se manda a llamar al método de “enlacesPaginasController”.

Por último, en el archivo template.php, se encuentra código JAVASCRIPT que se irá ejecutando conforme el usuario de clicks en ciertos botones que mostrarán acciones gráficamente, además de validar el cerrado de sesión. A continuación, se explica el código JAVASCRIPT que contiene este archivo.

Función getUrlVars()

```
1. function getUrlVars() {
2.       var vars = {};
3.       var parts = window.location.href.replace(/[?&]+([^=&]+)=([^&]*)/gi, function(
   m,key,value) {
4.           vars[key] = value;
5.       });
6.       return vars;
7.     }
```

Esta función obtiene la url mediante JAVASCRIPT y separa las variables en ellas para obtener su valor.

Función confirmarSesion()

```
1.  function confirmarSesion()
2.   {
3.     event.preventDefault();
4.     swal({
5.       title: "Cerrar sesión",
6.       text: "¿Seguro que deseas cerrar la sesión?",
7.       type: "warning",
8.       showCancelButton: true,
9.       cancelButtonText: "Cancelar",
10.         confirmButtonClass: "btn-info",
11.         confirmButtonText: "Si, estoy seguro",
12.         closeOnConfirm: false
13.       },
14.       function(){
15.         window.location = 'index.php?action=salir';
16.       });
17.     }
```

Función que lanza un sweetalert que pregunta al usuario si realmente quiere cerrar sesión y lo manda a un archivo que destruye la sesión actual y sale de la cuenta.

Función confirmarUpdate()

```
1. function confirmarUpdate(){
2.     var dbPassword = "<?php echo $_SESSION['password'] ?>";
3.     event.preventDefault();
4.     swal({
5.       title: "Confirmar acción",
6.       text: "<p>Ingresa tu contraseña para guardar los cambios</p><br><input
   type='password' class='form-control' id='pass' placeholder='Contraseña'
   autofocus><label id='err_sa' style='color:red'></label><br>",
7.       html: true,
8.
9.       type: "warning",
10.         showCancelButton: true,
11.         cancelButtonText: "Cancelar",
12.         confirmButtonText: "Confirmar",
13.         closeOnConfirm: false,
14.         inputPlaceholder: "Contraseña",
15.         inputValidator: (value) => {
16.           return !value && 'No puedes dejar el campo vacio!'
17.         }
18.       },function () {
19.           var inputValue = document.getElementById("pass").value;
20.           if (inputValue === false) return false;
21.           if (inputValue != dbPassword) {
22.             document.getElementById("err_sa").innerHTML = "Contraseña incorrecta";
23.             return false
24.           }
25.           $( "#btn" ).click();
26.             swal("Exito!", "Registro modificado", "success");
27.           });
28.       }
```

Función que manda un mensaje mediante un sweetalert para saber si estamos seguros de reiniciar todas las sesiones del cuatrimestre, posteriormente pide la password del superadmin y la compara con la almacenada en una variable de sesión, después de eso resetea la base de datos.

Función confirmarSalida()

```
1.    function confirmarSalida(id){
2.       event.preventDefault();
3.       //Operaciones que se hace para saber cuantas horas se han hecho en base a los
   minutos que han pasado desde la hora de entrada
4.       //y la hora en la que estamos (LO MISMO QUE EN PHP)
5.       var now = new Date();
6.       var mins = now.getMinutes();
7.       var hours = now.getHours();
8.       var e = document.getElementById("hora").innerText;
9.       var studentHours = e.split(":");
10.         var h = studentHours[0]+"00"
11.         var start_time = studentHours[0]+"00";
12.         var end_time =hours+""+mins;
13.         var start_hour = start_time.slice(0, -2);
14.         var start_minutes = start_time.slice(-2);
15.         var end_hour = end_time.slice(0, -2);
16.         var end_minutes = end_time.slice(-2);
17.         var startDate = new Date(0,0,0,start_hour, start_minutes);
18.         var endDate = new Date(0,0,0,end_hour, end_minutes);
19.         var millis = endDate - startDate;
```

```
20.         var minutes = millis/1000/60;
21.         var valMax = 10000;
22.         //Arreglo para saber los minutos de las horas que hay en cada hora y saber
    cuales minutos se acercan mas, asi conociendo
23.         //la hora a la que esta mas cerca el alumno (MISMO QUE EN PHP)
24.         var max = [60,120,180,240,300,360,420];
25.         var hora = 0;
26.         for(i = 0; i<max.length;i++){
27.           var op = max[i] - minutes;
28.           if(op < valMax && op>-1){
29.             valMax = max[i] - minutes;
30.             hora = i+1;
31.           }
32.         }
33.
34.         //Si se desea salir antes de que termine la sesion, (tolerancia: 10 mins)
    y no han pasado 4 horas entonces nos indica el mensaje
35.         //que la sesion no ha terminado y no se contara dicha hora
36.         if(max[hora-1] - minutes > 10 && hora<4){
37.           hora--;
38.           event.preventDefault();
39.           swal({
40.             title: "Confirmar acción",
41.             text: "<p>La sesion aun no ha terminado, si se libera ahora solamente
    habra hecho <strong>"+hora +" horas</strong></p>",
42.             html: true,
43.             type: "warning",
44.             showCancelButton: true,
45.             cancelButtonText: "Cancelar",
46.             confirmButtonText: "Confirmar",
47.             closeOnConfirm: false,
48.           },function () {
49.                 var url = document.getElementById("btn"+id).href;
50.                 window.location = url;
51.                 swal("Exito!", "Alumno liberado", "success");
52.               }
53.           );
54.           //Si ya pasaron 4 horas entonces solo noes muestra el mensaje que la
    sesion ya no se tomara en cuenta
55.         }else if(hora >= 4){
56.            swal({
57.             title: "Vaya!",
58.             text: "El tiempo limite ha pasado (4 horas), las horas no se tomaran
    en cuenta",
59.             html: true,
60.             type: "warning",
61.             confirmButtonText: "Ok",
62.             closeOnConfirm: false,
63.           },function () {
64.                 var url = document.getElementById("btn"+id).href;
65.                  window.location = url;
66.               }
67.           );
68.
69.         }
70.         //De lo contrario se registrara la sesion
71.         else{
72.           var url = document.getElementById("btn"+id).href;
73.           window.location = url;
74.         }
75.     }
```

Función que muestra un mensaje para saber si está seguro de liberar al alumno, donde dice si la sesión ha terminado o le falta tiempo para terminar, también dice cuántas horas ha hecho o si se pasó del límite, o en el caso de que la sesión no haya terminado entonces dice que no se contara la hora de la sesión no terminada.

Función verifyStudentInGroup()

```
1. function verifyStudentInGroup(id_g){
2.       var e = document.getElementById("alumno");
3.       var id = e.options[e.selectedIndex].value;
4.       var id_grupo = id_g;
5.       //Funcion AJAX que hace la consulta a la base de datos para saber si el
   alumno se eneucntra en el grupo que se desea registrar
6.       $.ajax({
7.         url:'models/check.php',
8.         method:"POST",
9.         data:{check_id : id, check_id2 : id_grupo},
10.           success: function(data){
11.             //Dicha funcion regresa un valor booleano, si regresa un verdadero
   entonces solo se muestra una sweet alert para advertir que el alumno
12.             //se encuentra en este grupo
13.             if(data == 1){
14.               swal("Error", "Ya existe el alumno en este grupo", "warning");
15.             }
16.             else{
17.               //De lo contrario se manda llamar otro metodo de jscript para
   registrar al alumno
18.               insertGroupTable(id_grupo);
19.             }
20.           },
21.           error: function(){
22.           }
23.         })
24.       }
```

Función que verifica si un alumno ya existe en un grupo de inglés, para evitar registrarlo dos veces.

Función showTeacherPerStudent()

```
1. function showTeacherPerStudent(){
2.       var e = document.getElementById("alumno");
3.       var maestroSelect = document.getElementById("maestro");
4.       //Cada que cambie la opcion en el select2 entonces se limpiara el select2 de
   los profesores
5.       for (i = 0; i < maestroSelect.options.length; i++){
6.         maestroSelect.options[i] = null;
7.       }
8.       $('#maestro').empty().trigger("change")
9.       var id_alumno = e.options[e.selectedIndex].value;
10.         //Funcion en ajax para hacer la consulta de los profesores y ponerlos en
   un select2, donde se obtendran los resultados en un
11.         //objeto tipo JSON ya que pueden ser varios profesores
12.         $.ajax({
13.           url:'models/check.php',
14.           method:"POST",
```

```
15.           data:{id_alumno2 : id_alumno},
16.           success: function(data){
17.             if(data){
18.               //Se crea una opcion para el select2
19.               var opt = document.createElement('option');
20.               opt.value = "";
21.               opt.innerHTML = "Maestro...";
22.               maestroSelect.appendChild(opt);
23.               for(i = 0; i<data.length;i++){
24.
25.                 //Se insertan los profeores en el select2
26.                 var opt = document.createElement('option');
27.                 opt.value = data[i][1];
28.                 opt.innerHTML = data[i][0];
29.                 maestroSelect.appendChild(opt);
30.               }
31.               maestroSelect.options[0].selected=true;
32.             }
33.             else{
34.             }
35.           },
36.           error: function(){
37.           },
38.           dataType:"json"
39.         })
40.         //Funcion en ajax para cada que cambie el alumno, cambie la foto que esta
   registrada en la base de datos, y se muestre en la parte inferior
41.         //de la interfaz para registrar una sesion nueva
42.         $.ajax({
43.           url:'models/check.php',
44.           method:"POST",
45.           data:{checkImage : id_alumno},
46.           success: function(data){
47.             if(data){
48.               document.getElementById("myImg").src="uploads/"+data[0];
49.
50.             }
51.             else{
52.             }
53.           },
54.           error: function(){
55.           },
56.           dataType:"json"
57.         })
58.       }
```

Función que carga los profesores de los que recibe clase un alumno, para poder registrar una sesión de CAI.

Función detectStudentLate()

```
1.  function detectStudentLate(){
2.       var now = new Date();
3.       var mins = now.getMinutes();
4.       if(mins>10){
5.         swal("Error", "Los minutos de tolerancia se han agotado", "warning");
6.         event.preventDefault();
7.       }
```

```
8.       //Si no ha excedido el limite de los 10 minutos entonces se hace una consulta
   para saber si el alumno ya esta en una sesion
9.       else{
10.           var id_alumno_sesion = document.getElementById("alumno").value;
11.           $.ajax({
12.             url:'models/check.php',
13.             method:"POST",
14.             data:{id_alumno_sesion : id_alumno_sesion},
15.             success: function(data){
16.               if(data){
17.                 //Si la consulta regreso un verdadero entonces solo nos muestra un
   sweet alert diciendo que este alumno ya esta en sesion
18.                 if(data == 1){
19.                   swal("Error!", "El alumno ya se encuentra en sesion", "error");
20.                 }
21.                 //De lo contrario se procede a consultar si la actividad que desea
   se encuentra disponible
22.                 else{
23.                   checkAvailability();
24.                 }
25.               }
26.               else{
27.               }
28.             },
29.             error: function(){
30.             },
31.             //dataType:"json"
32.         })
33.         }
34.       }
```

Función que revisa la hora en que se quiere registrar una nueva sesión de CAI y si esta esta fuera del rango de tolerancia no permite registrarla.

Función checkAvailability()

```
1. function checkAvailability(){
2.       var actSelect = document.getElementById("act");
3.       var id_actividad = actSelect.options[actSelect.selectedIndex].value;
4.       //Funcion AJAX para conocer cuantos lugares hay de dicha actividad
5.       $.ajax({
6.         url:'models/check.php',
7.         method:"POST",
8.         data:{id_actividad : id_actividad},
9.         success: function(data){
10.             if(data){
11.               //Si los lugares es mayor a 0 entonces se registra la sesion
12.               if(data>0){
13.                 registrarSesion(data);
14.               //De lo contrario solo nos muestra un sweet alert que la actividad
   ya no se encuentra disponible
15.               }else{
16.                 swal("Error!", "Esta actividad ya no esta disponible", "error");
17.               }
18.             }
19.           },
20.           error: function(){
21.           },
22.             //dataType:"json"
```

```
        })
      }
```

Función que revisa los lugares disponibles para cada actividad en la hora actual. Si alguna actividad tiene lleno sus lugares, manda un mensaje de error al usuario.

```
function registrarSesion(lugarNuevo){
      var alumnoSelect = document.getElementById("alumno");
      var id_alumno = alumnoSelect.options[alumnoSelect.selectedIndex].value;
      var maestroSelect = document.getElementById("maestro");
      var id_maestro = maestroSelect.options[maestroSelect.selectedIndex].value;
      var actSelect = document.getElementById("act");
      var id_actividad = actSelect.options[actSelect.selectedIndex].value;
      var fecha = document.getElementById("fecha").value
      var hora_entrada = document.getElementById("entrada").value;
         var unidad = document.getElementById("unidad").value;
         var lugares = lugarNuevo;

         if(!(id_alumno=="" && id_maestro="")){
           //Funcion AJAX para registrar la sesion, donde se restara la actividad que se eligio de la columna lugares
           $.ajax({
           url:'models/check.php',
           method:"POST",
           data:{id_alumno_sesion2 : id_alumno, id_maestro_sesion : id_maestro, id_actividad_sesion : id_actividad, fecha_sesion : fecha, hora_entrada_sesion : hora_entrada, unidad_sesion : unidad, lugares: lugares},
           success: function(data){
             swal("Listo", "Sesion registrada!", "success");
             //window.location='index.php?action=sesiones';

           },
           error: function(){
           },
         })
         }
         else{
           swal("Error!", "Elige un Alumno y Profesor", "error");
         }
       }
```

Función que registra una nueva sesión en la base de datos mediante el método POST de AJAX, revisando que ciertos campos estén llenados.

Módulos (interfaces)

En esta sección, solo se mostrará la estructura de algunas interfaces, ya que todas son similares, y no es necesario agregarlas.

Navegación de usuario y de administrador (views/modules/navegación.php)

```
<?php
```

```
2.  //Se inicia la sesion
3.  session_start();
4.  //Se valida la sesion
5.  if(!$_SESSION["validar"]){
6.    header("location:index.php");
7.    exit();
8.  }else{
9.    $usuario = $_SESSION["usuario"];
10.    }
11.
12.   ?>
13.   <div style="background-color: #E8ECEF; width: 100%; min-height: 100%; position:
   absolute;">
14.   <nav class="navbar navbar-expand-lg navbar-dark" style="background-color:
   #0087FF">
15.     <a class="navbar-
   brand" href="index.php?action=inicioadmin" style="color:white">CENTRO
   DE <b>APRENDIZAJE DE INGLÉS</b></a>
16.     <button class="navbar-toggler" type="button" data-toggle="collapse" data-
   target="#navbarNavDropdown" aria-controls="navbarNavDropdown" aria-
   expanded="false" aria-label="Toggle navigation">
17.       <span class="navbar-toggler-icon"></span>
18.     </button>
19.     <div class="collapse navbar-collapse" id="navbarNavDropdown">
20.       <ul class="navbar-nav">
21.         <li class="nav-item active" style="margin-left: 70px">
22.           <a class="nav-
   link" href="index.php?action=inicioadmin">Inicio <span class="sr-
   only">(current)</span></a>
23.         </li>
24.         <li class="nav-item" style="margin-left: 10px">
25.           <a class="nav-link" href="index.php?action=alumnos">Alumnos</a>
26.         </li>
27.         <li class="nav-item">
28.           <a class="nav-link" href="index.php?action=profesores" style="margin-
   left: 10px">Profesores</a>
29.         </li>
30.         <li class="nav-item">
31.           <a class="nav-link" href="index.php?action=actividades" style="margin-
   left: 10px">Actividades</a>
32.         </li>
33.         <li class="nav-item">
34.           <a class="nav-link" href="index.php?action=grupos" style="margin-left:
   10px">Grupos</a>
35.         </li>
36.         <li class="nav-item">
37.           <a class="nav-link" href="index.php?action=sesiones" style="margin-left:
   10px">Sesiones</a>
38.         </li>
39.         <li class="nav-item">
40.           <a class="nav-link" href="index.php?action=reportes" style="margin-left:
   10px">Reporte</a>
41.         </li>
42.         <li class="nav-item dropdown">
43.           <a class="nav-link dropdown-
   toggle" href="#" id="navbarDropdownMenuLink" data-toggle="dropdown" aria-
   haspopup="true" aria-expanded="false">
44.             Bienvenido, <?php echo $usuario; ?>
45.           </a>
46.           <div class="dropdown-menu" aria-labelledby="navbarDropdownMenuLink">
```

```
47.             <a class="dropdown-item" onclick="confirmarSesion();">Cerrar
   Sesión</a>
48.           </div>
49.         </li>
50.       </ul>
51.     </div>
52.   </nav>
53.   </div>
```

Estructura html de la barra de navegación del sistema. Existe un archivo diferente para los dos tipos de usuarios (maestro y administrador), los dos contienen código PHP añadido para poder crear la sesión y checarla cada que el usuario navegue entre las opciones, si esta sesión está inactiva el sistema lo manda a la página de inicio.

Inicio de administrador (views/modules/inicioadmin.php)

```
1. <?php
2.
3.  $ob = new MvcController();
4.
5.  $alumnos = $ob->numeroAlumnos();
6.   $profesores = $ob->numeroProfesores();
7.   $grupos = $ob->numeroGrupos();
8.
9.
10.   ?>
11.   <br style="background-color: #E8ECEF ">
12.   <div style="background-color: #E8ECEF ">
13.           <div class="jumbotron" >
14.                   <div class="col-sm-10 mx-auto">
15.                           <h1><a style="font-weight: bold; font-size:
   1.5em">BIENVENIDO</a></h1>
16.                           <hr class="row">
17.                           <br>
18.                           <div class="row" align="center">
19.                                   <div class="col-xs-12 col-sm-4" >
20.                                           <div class="card border-primary mb-3">
21.                                                   <div class="card-body
   text-primary">
22.                     <h5 style="color: #000; font-weight: bold"><?php echo
   $alumnos; ?> ALUMNOS REGISTRADOS</h5>
23.                                                   </div>
24.                                           <div class="card-
   footer"><a href="index.php?action=alumnos">Ir a alumnos</a></div>
25.                                           </div>
26.                                   </div>
27.                                   <div class="col-xs-12 col-sm-4">
28.                                           <div class="card border-primary mb-3">
29.                                                   <div class="card-body
   text-primary">
30.                                                   <h5 style="color: #000; font-weight:
   bold"><?php echo $profesores; ?> PROFESORES REGISTRADOS</h5>
31.                                           </div>
```

```
32.                                                 <div class="card-
   footer"><a href="index.php?action=profesores">Ir a profesores</a></div>
33.                                             </div>
34.                                         </div>
35.                                         <div class="col-xs-12 col-sm-4">
36.                                             <div class="card border-primary mb-3">
37.                                                 <div class="card-body
   text-primary">
38.                                                 <h5 style="color: #000; font-
   weight: bold"><?php echo $grupos; ?> GRUPOS REGISTRADOS</h5>
39.                                                 </div>
40.                                                 <div class="card-
   footer"><a href="index.php?action=grupos">Ir a grupos</a></div>
41.                                             </div>
42.                                         </div>
43.                     </div>
44.                     <br>
45.                     <div class="row">
46.                                         <div class="col-xs-12 col-sm-4">
47.                                             <div class="card">
48.                                                 <img class="card-img-
   top" style="height: 140px; width: 800px" src="views/bg2.jpg" alt="Card image cap">
49.                                                 <div class="card-body">
50.                                                 <h5 class="card-title">¿Qué es
   CAI?</h5>
51.                                                 <p class="card-text">Centro de
   Aprendizaje de Inglés de la Universidad Politécnica de Victoria.</p>
52.                                                 <a href="#" onclick="verCAI();"
   class="btn btn-primary" style="background-color: #0087FF">Ver más</a>
53.                                                 </div>
54.                                             </div>
55.                                         </div>
56.                                         <div class="col-xs-12 col-sm-4">
57.                                             <div class="card">
58.                                                 <img class="card-img-
   top" style="height: 140px; width: 800px" src="views/bg3.jpg" alt="Card image cap">
59.                                                 <div class="card-body">
60.                                                 <h5 class="card-title">Objetivo
   del sitio</h5>
61.                                                 <p class="card-text">El
   principal objetivo es centralizar las actividades de los alumnos en CAI.</p>
62.                                                 <a href="#" onclick="verob();" c
   lass="btn btn-primary" style="background-color: #0087FF">Ver más</a>
63.                                                 </div>
64.                                             </div>
65.                                         </div>
66.                                         <div class="col-xs-12 col-sm-4">
67.                                             <div class="card">
68.                                                 <img class="card-img-
   top" style="height: 140px; width: 600px" src="views/bg6.jpg" alt="Card image cap">
69.                                                 <div class="card-body">
70.                                                 <h5 class="card-
   title">Soporte</h5>
71.                                                 <p class="card-text">¿Tienes
   alguna duda sobre el sistema? Contáctanos.</p>
72.                                                  <button onclick="versp();" id="
   ver" type="button" class="btn btn-primary">Ver más</button>
73.                                                 </div>
74.                                             </div>
75.                                         </div>
76.                     </div>
```

```
77.
78.
79.                     </div>
80.             </div>
81.     </div>
```

Estructura html de la página de inicio del administrador que muestra tarjetas informativas sobre el sistema, además de mostrar el control de alumnos, maestros y grupos.

Vista gráfica del inicio del administrador.

Alumnos (views/modules/alumnos.php)

```
1. <br style="background-color: #E8ECEF ">
2. <div style="background-color: #E8ECEF ">
3.         <div class="jumbotron" >
4.                 <div class="col-sm-10 mx-auto">
5.       <div class="row">
6.         <div class="col-xs-12 col-sm-4">
7.                             <h1><a style="font-weight: bold; font-size:
   1.3em">ALUMNOS</a></h1>
8.         </div>
9.         <div class="col-xs-12 col-sm-6">
10.              <a href="index.php?action=nuevoalumno"><button type="button" class="
   btn btn-primary btn-sm" style="margin-top: 30px; margin-left: 600px; background-
   color: #0087FF">Registrar Alumno</button></a>
11.           </div>
12.         </div>
```

```
                            <hr>
        <?php
         if(isset($_SESSION["eliminado"])){
           if($_SESSION["eliminado"]==1){
             echo "<div class='alert alert-danger alert-dismissible fade show'
  role='alert'>
                   Alumno Eliminado Exitósamente
                   <button type='button' class='close' data-dismiss='alert' aria-
  label='Close'>
                     <span aria-hidden='true'>&times;</span>
                   </button>
                 </div>";
                 $_SESSION["eliminado"]=0;
            }
          }
          $_SESSION["eliminado"]=0;

        ?>
                            <br>
          <div class="table-responsive">
                <table id="example1" class="table table-bordered table-striped">
                  <thead>
                  <tr>
                    <th>Matrícula</th>
                    <th>Nombre</th>
                    <th>Carrera</th>
                    <th></th>
                    <th></th>
                    <th></th>
                  </tr>
                  </thead>
                  <tbody>
                    <?php
                       //Creacion del objeto y llamado de sus objetos
                       $vistaAlumnos = new MvcController();
                       $vistaAlumnos->vistaAlumnosController();
                    ?>
                  </tbody>
                </table>
            </div>
                            <br><br><br><br><br><br><br>
                  </div>
          </div>
  </div>

  <?php
   $eliminarAlumno = new MvcController();
   $eliminarAlumno->eliminarAlumnoController();

  ?>
```

Estructura del módulo de alumnos que contiene una datatable que es poblada llamando a un método del controlador que consulta todos los alumnos existentes en la base de datos, además contiene código PHP que verifica la eliminación de un alumno.

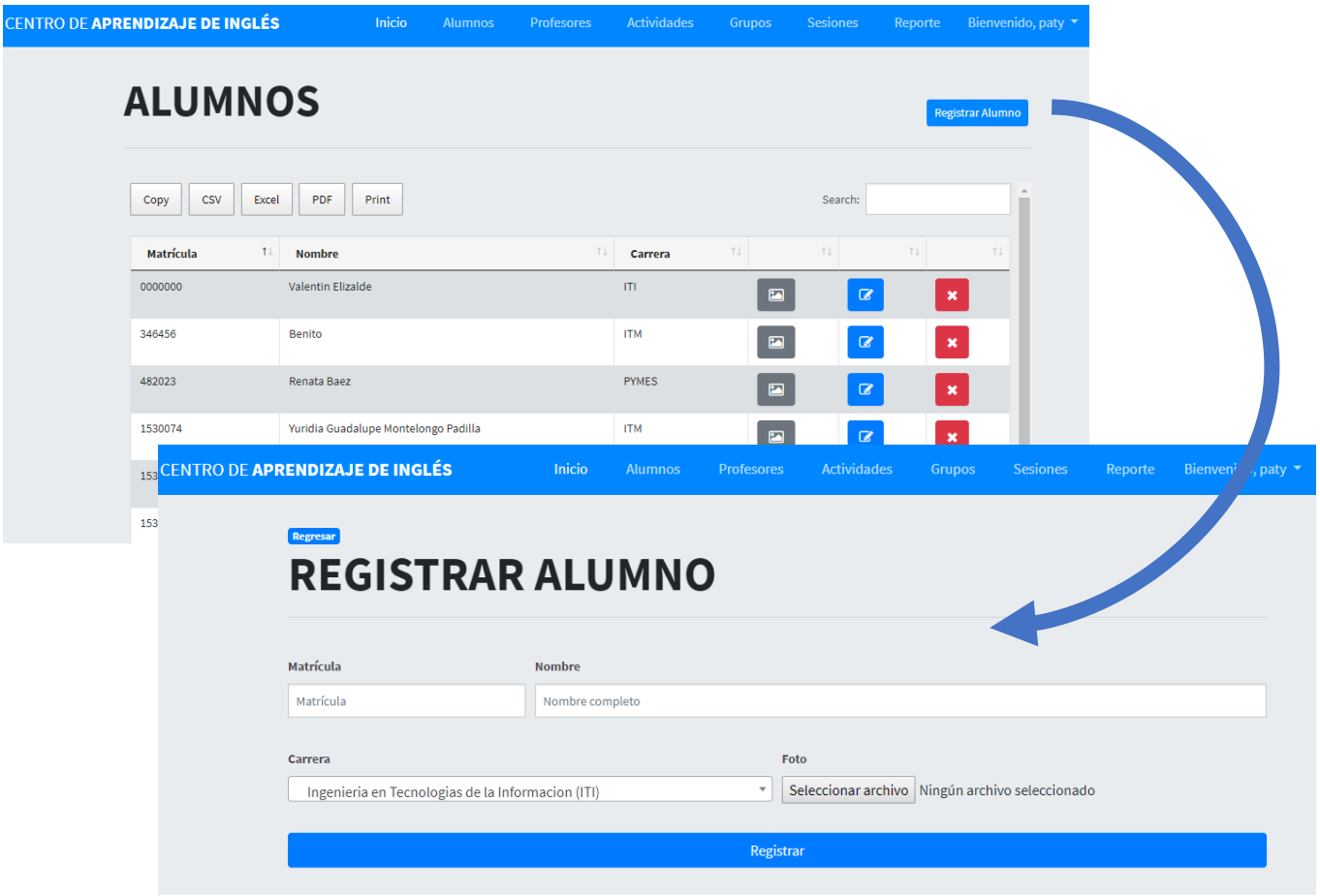
Nuevo alumno (views/modules/nuevoalumno.php)

```
<br style="background-color: #E8ECEF ">
<div style="background-color: #E8ECEF ">
        <div class="jumbotron" >
                <div class="col-sm-10 mx-auto">
      <a href="index.php?action=alumnos" class="badge badge-primary">Regresar</a>
      <br>
                        <h1><a style="font-weight: bold; font-size: 1.3em">REGISTRAR ALUMNO</a></h1>
                        <hr>
      <?php
          if(isset($_SESSION["registrado"])){
            if($_SESSION["registrado"]==1){
              echo "<div class='alert alert-success alert-dismissible fade show' role='alert'>
                    Alumno Registrado Exitósamente
                    <button type='button' class='close' data-dismiss='alert' aria-label='Close'>
                      <span aria-hidden='true'>&times;</span>
                    </button>
                  </div>";
                  $_SESSION["registrado"]=0;
            }
           }
           $_SESSION["registrado"]=0;

        ?>
                            <br>
          <form id="formu" method="post" name="formu" enctype="multipart/form-data">
              <div class="form-row">
                <div class="form-group col-md-3">
                  <label for="matricula">Matrícula</label>
                  <input maxlength="7" type="text" class="form-control" id="matricula" name="matricula" placeholder="Matrícula" required>
                </div>
                <div class="form-group col-md-9">
                  <label for="nombre">Nombre</label>
                  <input type="text" class="form-control" id="nombre" name="nombre" placeholder="Nombre completo" required>
                </div>
              </div>
              <div class="form-row">
                <div class="form-group col-md-6">
                  <label for="carrera">Carrera</label>
                  <select id="carrera" name="carrera" class="form-control select2">
                     <?php
                       $carreras = new MvcController();
                       $carreras->mostrarCarrerasController();
                     ?>
                  </select>
                </div>
                <div class="form-group col-md-6">
                  <label for="fileToUpload">Foto</label>
                  <input type="file" class="form-control-file" name="fileToUpload" id="fileToUpload">
                </div><br>
            </div>
              <button onclick="detectId();" type="submit" id="registrar" name="registrar" style="width: 100%" class="btn btn-primary">Registrar</button>
            </form>
                    </div>
```

```
54.            </div>
55.    </div>
56.
57.    <?php
58.       $regAlumno = new MvcController();
59.       $regAlumno->registrarAlumnoController();
60.    ?>
```

Estructura html de la interfaz que contiene el formulario para registrar un nuevo alumno a la base de datos, este consta de ciertos inputs obligatorios, además de contener código PHP que detecta la inserción de un nuevo alumno y manda un mensaje de éxito. Cada código PHP en donde se crea una nueva instancia de una clase llamada “MvcController” se convoca a un método de la misma que realiza las acciones desde el controlador implementando el modelo y la conexión a la base de datos.



Interfaces del módulo de alumnos.

Nota: todas las otras interfaces no se muestran debido a su similitud con las anteriormente explicadas.

Conexión a la base de datos (models/conexión.php)

```
1. <?php
2. //Clase conexion que permite acceder a la base de datos con PDO
3.         class Conexion{
4.                 //unico metodo
5.                 public function conectar(){
6.                         $link = new PDO("mysql:host=localhost;dbname=cai","root","j
   onasyuridia");
7.                         return $link;
8.                 }
9.         }
10.    ?>
```

En este archivo se crea una clase llamada “Conexión” que contiene un método llamado “conectar” que contiene una variable llamada “link” que almacena los datos de la locación de la base de datos utilizada en el sistema. La base de datos es sql y su nombre es “cai”, en el código descrito aquí se colocó la contraseña del servidor en el que está alojado, sin embargo, todos estos datos pueden variar dependiendo donde se despliegue el sistema.

Enlaces (models/enlaces.php)

```
1. <?php
2. //Clase ue controla la navegacion del sitio
3.         class Paginas{
4.                 //unico metodo que verifica que action se ejecuto
5.                 public function enlacesPaginasModel($enlaces){
6.
7.                         if($enlaces=="index" || $enlaces=="fallo"){
8.                                 $module =  "views/modules/ingresar.php";
9.                         }else{
10.                                $module =  "views/modules/".$enlaces.".php";
11.                        }
12.
13.                        //se retorna la pagina a la que se dirige
14.                        return $module;
15.                }
16.        }
17.    ?>
```

Archivo que contiene una clase llamada “Paginas” que a su vez contiene una función llamada “enlacesPaginasModel” que interpreta un parámetro y carga el módulo corresponditente.

Sentencias necesarias en la implementación de AJAX (models/check.php)

```
1. //Se requiere el archivo donde se hace la conexion a la base de datos con sus
   respectivas credenciales
2.         require_once("conexion.php");
3.
4.   //Si se llego un metodo POST
5.         if(isset($_POST)){
6.
7.     //Operacion para al momento de registrar un alumno nuevo en un grupo, se
   comprueba si el alumno ya esta registrado
8.     //en el grupo. Teniendo como parametro el id del alumno, el id de grupo
9.
10.       //Se comprueba si estas variables llegaron a este archivo mediante AJAX para
   saber que operacion se debera hacer
11.                   if(isset($_POST['check_id']) && isset($_POST['check_id2'])){
12.
13.         //Se hace la sentencia SQL del alumno que se pretende registrar y conocer
   si este ya se encuentra registrado
14.         //en dicho grupo
15.                           $stmt = Conexion::conectar()->prepare("SELECT * FROM
   alumno_grupo where id_alumno = :id_alumno AND id_grupo = :id_grupo");
16.                           //Parametros donde es le id del alumno y el id del grupo
17.         $stmt->bindParam(":id_alumno",$_POST['check_id']);
18.                           $stmt->bindParam(":id_grupo",$_POST['check_id2']);
19.                           if($stmt->execute()){
20.           //Si encontro un alumno en este grupo entonces se regresa un verdadero.
   Esto quiere decir que ya no puede
21.           //ser registrado en este grupo
22.           if ($stmt->fetchColumn() > 0){
23.                   echo 1;
24.           }
25.           else
26.           {
27.                   echo 0;
28.           }
29.   }
```

En esta primera parte del archivo se manda llamar el archivo “conexion.php” para poder acceder a la base de datos del sistema. Después hay una condición que valida si el método post ha sido activado, y la primera operación a realizar es la de registrar un alumno nuevo a un grupo. Primero, se comprueba que el alumno no esté registrado ya, recibiendo como parámetro las variables que llegaron por el método post y así crear la sentencia SQL que compare el id del alumno que se ingreso con los registrados en la base de datos, si existe el alumno se retornará un 0, sino, se retornará un 1. Después, el siguiente código se encarga de insertar al alumno en la base de datos en el grupo siempre y cuando este no se encuentre registrado ya.

```
1. //Se comprueba si estas variables especificas llegaron para hacer la operacion
   necesario
2.                 else if(isset($_POST['insert_alumno']) && isset($_POST['insert_grup
   o'])){
3.       //Se hace el insert del alumno en el grupo
4.                           $stmt = Conexion::conectar()->prepare("INSERT INTO
   alumno_grupo (id_alumno, id_grupo) VALUES (:id_alumno,:id_grupo)");
5.                           //Parametros para hacer el insert
6.       $stmt->bindParam(":id_grupo",$_POST['insert_grupo']);
```

```
                        $stmt->bindParam(":id_alumno",$_POST['insert_alumno']);
        if($stmt->execute()){
        }
```

La siguiente parte de código que contiene la condición carga los maestros de un alumno dinámicamente, haciendo una consulta cada que el nombre del alumno cambie en un select2.

```
else if(isset($_POST['id_alumno2'])){
                        $id_alumno = $_POST['id_alumno2'];
      //Sentencia SQL que traera todos los profesores a los que esta asignado un alumno
                        $stmt = Conexion::conectar()->prepare("SELECT maestro.nombre_maestro, maestro.id_maestro from maestro inner join grupo on maestro.id_maestro = grupo.id_maestro inner join alumno_grupo on grupo.id_grupo = alumno_grupo.id_grupo where alumno_grupo.id_alumno = :id_alumno");
                        //Parametro para hacer la consulta
      $stmt->bindParam(":id_alumno",$id_alumno);
      //Se ejecuta la sentencia
                        if($stmt->execute()){
        //Se trae todos los resultados en un array con indices numericos ya que puede un alumno puede estar en varios
          //grupos y por consecuencia tener varios profesores
                                $usuario = $stmt->fetchAll(PDO::FETCH_NUM);
                                if($usuario){
                                        //Como se traera una respuesta con multiples valores entonces necesita devolverse un objeto
            //de tipo JSON para regresar una matriz con los diferentes datos de los profesores
                                        echo json_encode($usuario);
                                }
                        }
                }
```

En la siguiente condición por cada que cambie el alumno en el select2 correspondiente, se hará una consulta a la base de datos para comprobar que imagen se le registro a dicho alumno, una vez esto se traerá la url de la imagen para ponerlo en un modal, cada que cambie el alumno del select2 la imagen también cambiara mostrando la respectiva imagen del alumno.

```
else if(isset($_POST['checkImage'])){
                        $id_alumno = $_POST['checkImage'];
      //Se hace la consulta de la imagen
                        $stmt = Conexion::conectar()->prepare("SELECT img_alumno from alumno where id_alumno = :id_alumno");
                        //Se registra el parametro
      $stmt->bindParam(":id_alumno",$id_alumno);
                        //Se ejecuta la sentencia
      if($stmt->execute()){
        //Se trae la respuesta de la consulta por medio de un array con indices numericos
                                $usuario = $stmt->fetchAll(PDO::FETCH_NUM);
```

```
                                    if($usuario){
                                            //Como se devolvera un array entonces es necesario devolverlo en objeto tipo JSON
                                            echo json_encode($usuario);
                                    }
                            }

                    }
```

En la siguiente condición se evalúa si ya hay algún alumno registrado con la misma matricula que se pretende registrar en un nuevo alumno, si esto llegar a ser así entonces se impedirá el registro del alumno.

```
else if(isset($_POST['matricula'])){
      //Sentencia SQL
                        $stmt = Conexion::conectar()->prepare("SELECT * from alumno where matricula_alumno = :matricula");
                        //Se registra el parametro
      $stmt->bindParam(":matricula",$_POST['matricula']);
                        if($stmt->execute()){
        //Si esta consulta devuelve una fila entonces quiere decir que ya hay un alumno registrado y devuelve verdadero
                                if ($stmt->fetchColumn() > 0){
                                        echo 1;
                                }
                                else
                                {
                                        echo 0;
                                }
                            }
                    }
```

En la siguiente condición se ejecuta al momento de registrar un alumno en una nueva sesión. Esta consulta verifica si el alumno que se pretende registrar en una nueva sesión ya está en una sesión, si es así entonces no hace el registro de la sesión.

```
else if(isset($_POST['id_alumno_sesion'])){
      //Sentencia SQL
                        $stmt = Conexion::conectar()->prepare("SELECT * from sesion where id_alumno = :id_alumno");
                        //Registro de parametro
      $stmt->bindParam(":id_alumno",$_POST['id_alumno_sesion']);
                        if($stmt->execute()){
        //Si la consulta regreso alguna fila eso quiere decir que dicho alumno ya esta registrado
        //por lo que no se hace el nuevo registro
                                if ($stmt->fetchColumn() > 0){
                                        echo 1;
                                }
                                else
                                {
                                        echo 0;
                                }
                            }
                    }
```

En la siguiente condición se ejecuta al momento de registrar una sesión nueva. Cuando se registra una nueva sesión, esta implica escoger una actividad nueva, al hacerlo se comprueba si hay lugares disponibles en esta actividad, por lo que es necesario traer los lugares disponibles de dicha actividad para comprobar si aún hay espacio.

```
1. else if(isset($_POST['id_actividad'])){
2.       //sentencia SQL
3.                         $stmt = Conexion::conectar()->prepare("SELECT lugares from
   actividad where id_actividad = :id_actividad");
4.                         //Se registra el parametro
5.       $stmt->bindParam(":id_actividad",$_POST['id_actividad']);
6.                         if($stmt->execute()){
7.         //Se obtiene la respuesta en un array con indices numericos
8.                                 $usuario = $stmt->fetch(PDO::FETCH_NUM);
9.                                 if($usuario){
10.             //Solamente se regresa la respuesta
11.                                             echo $usuario[0];
12.                                     }
13.                             }
14.                     }
```

Y en la última condición si todas las validaciones dieron luz verde (lugares de la actividad y que no esté registra ya) se procede a hacer el registro de la sesión en la base de datos.

```
1. else if(isset($_POST['id_alumno_sesion2']) && isset($_POST['id_maestro_sesion'])){
2.
3.       //Sentencia SQL
4.                         $stmt = Conexion::conectar()->prepare("INSERT INTO sesion
   (id_alumno, id_actividad, id_maestro, fecha, hora_entrada, unidad)
   VALUES(:id_alumno, :id_actividad, :id_maestro ,:fecha,:hora_entrada,:unidad)");
5.
6.       //Registro de PARAMETROS
7.                         $stmt->bindParam(':id_alumno',$_POST["id_alumno_sesion2"]);
8.                         $stmt-
   >bindParam(':id_maestro',$_POST["id_maestro_sesion"]);
9.                         $stmt-
   >bindParam(':id_actividad',$_POST["id_actividad_sesion"]);
10.                             $stmt->bindParam(':fecha',$_POST["fecha_sesion"]);
11.                             $stmt-
   >bindParam(':hora_entrada',$_POST["hora_entrada_sesion"]);
12.                             $stmt->bindParam(':unidad',$_POST["unidad_sesion"]);
13.                             if($stmt->execute()){
14.                             }
15.
16.         //Al hacer el registro de la sesion entonces es necesario restar un lugar
   a la actividad para que
17.         //el control de lugares sea en tiempo real
18.                             $l = $_POST["lugares"]-1;
19.                             print_r($_POST);
20.                             $stmt = Conexion::conectar()->prepare("UPDATE actividad
   set lugares = :lugares where id_actividad = :id_actividad");
21.                             $stmt->bindParam(':lugares',$l);
22.                             $stmt-
   >bindParam(':id_actividad',$_POST["id_actividad_sesion"]);
```

```
                            if($stmt->execute()){
                            }
                    }
            }
    ?>
```

Crud que contiene todas las funciones que realizan cambios en la base de datos y que son ejecutadas a través de un controlador (views/model.php).

En este archivo se requiere el archivo “conexion.php” que contiene la conexión a la base de datos. Y como clase principal contiene una llamada “Datos” que extiende a la clase Conexión que está en el archivo anteriormente mencionado.

Función vistaUnidadesModel()

La primera función se llama vistaUnidadesModel() que recibe como parámetro el nombre de la tabla en la que se hará la consulta que obtiene todos los registros de la misma y los devuelve en un array.

```
<?php
//Se manda a llamar el archivo que contiene la conexion a la base de datos
require_once("conexion.php");

//Clase general que contiene todos los modelos que consume el controlador principal
  y que exitene la clase conexion que se encuentra en el modelo conexion
Class Datos extends Conexion{

  public function vistaUnidadesModel($tabla){
    $stmt = Conexion::conectar()->prepare("SELECT * from $tabla");
       $stmt->execute();
                  return $stmt->fetchAll();
     }
```

Función actualizarUnidadesModel()

En la siguiente función llamada actualizarUnidadesModel() recibe como parámetro la tabla en la que se llevará acabo la consulta, además de los datos que se quieren actualizar. Se crea una consulta UPDATE en donde se actualiza los campos de fecha de inicio y fecha fin y devuelve un booleano que indica si la consulta se llevo a cabo o no.

```
public function actualizarUnidadesModel($tabla,$datosModel){
    $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET fecha_inicio =
  :inicio, fecha_fin = :fin WHERE id_unidad= :id");
    $stmt->bindParam('inicio',$datosModel["inicio"]);
                $stmt->bindParam('fin',$datosModel["fin"]);
    $stmt->bindParam('id',$datosModel["id"]);
    return $stmt->execute();
                $stmt->close();

  }
```

Función ingresoUsuarioModel()

La función ingresoUsuarioModel() recibe como parámetro el array que contiene los desean consultar, además del nombre de la tabla en la que se consultarán. Esta función realiza y prepara la consulta select que comprueba si el usuario ingresado realmente está en la base de datos, si esto es así se retorna el tipo de usuario que es.

```
1. public function ingresoUsuarioModel($datosController,$tabla){
2.
3.                 $stmt = Conexion::conectar()->prepare("SELECT tipo_usuario
   FROM $tabla WHERE usuario = :u AND pass = :c AND eliminado=0");
4.                 //Se ejecuta la consulta
5.
6.                 $stmt->bindParam(':u',$datosController["usuario"]);
7.                 $stmt->bindParam(':c',$datosController["contra"]);
8.
9.                 if($stmt->execute()){
10.                         //Se retorna la fila si es que existe
11.                         $usuario = $stmt->fetchColumn();
12.                         if ($usuario){
13.                                 return $usuario["tipo_usuario"];
14.                         }
15.                     }
16.
17.                     return "error";
18.                     //se cierra la consulta
19.                     $stmt->close();
20.             }
```

Función getUnitByModel()

Después se tiene una función llamada “getUnitByDateModel” que obtiene la unidad actual dependiendo la fecha en la que se encuentre el día de hoy y se retorna esa fila de la tabla encontrada entre fecha de inicio y fecha fin.

```
1. public function getUnitByDateModel($datosModel){
2.         $stmt = Conexion::conectar()->prepare("SELECT unidad FROM unidad WHERE
   DATE_FORMAT(:hoy,'%m-%d') >= DATE_FORMAT(fecha_inicio,'%m-%d') AND
   DATE_FORMAT(:hoy,'%m-%d') <= DATE_FORMAT(fecha_fin,'%m-%d')");
3.         $stmt->bindParam(":hoy",$datosModel);
4.         $stmt->execute();
5.         return $stmt->fetch();
6.
7. }
```

Función registrarAlumnoModel()

En la siguiente función se muestra la consulta que registra un alumno a la base de datos recibiendo como parámetros los datos a registrar, además del nombre de la tabla. Dentro de ella se crea una consulta que inserta los datos y se ejecuta la misma

para realizar los cambios. La función retorna el resultado de la consulta y con eso se valida si se llevó a cabo o no.

```
1. public function registrarAlumnoModel($datosController, $tabla){
2.         $stmt = Conexion::conectar()->prepare("INSERT INTO $tabla(matricula_alumno,
   nombre_alumno, id_carrera, img_alumno) VALUES(:matricula, :nombre, :carrera,
   :foto)");
3.         $stmt>bindParam(':matricula',$datosController["matricula"]);
4.         $stmt->bindParam(':nombre',$datosController["nombre"]);
5.                 $stmt->bindParam(':carrera',$datosController["carrera"]);
6.         $stmt->bindParam(':foto',$datosController["foto"]);
7.         return $stmt->execute();
8.         $stmt->close();
9. }
```

Función consultarGrupoAlumnoModel()

Este modelo prepara una consulta para retornar todos los registros existentes lógicamente en la base de datos de los grupos.

```
1. public function consultarGruposAlumnoModel($datosModel){
2.         $stmt = Conexion::conectar()->prepare("SELECT alumno.matricula_alumno,
   alumno.id_alumno, alumno.nombre_alumno FROM alumno_grupo  inner join alumno on
   alumno_grupo.id_alumno  = alumno.id_alumno where alumno_grupo.id_grupo =
   :id_grupo");
3.         $stmt->bindParam(":id_grupo",$datosModel);
4.         $stmt->execute();
5.         return $stmt->fetchAll();
6.         $stmt->close();
7. }
```

Función reiniciarModel()

En esta función se eliminan todos los registros de las entradas de sesiones en la base de datos reiniciando también el conteo de los mismos.

```
1. public function reiniciarModel(){
2.         ¿$stmt = Conexion::conectar()->prepare("DELETE FROM entrada");
3.         $stmt->execute();
4.         $stmt = Conexion::conectar()->prepare("ALTER TABLE entrada
   AUTO_INCREMENT=1");
5.         return $stmt->execute();
6. }
```

Función consultarGruposModel()

Función que obtiene todos los grupos no eliminados de la base de datos mediante una consulta SELECT.

```
1. public function consultarGruposModel(){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM grupo
   WHERE  eliminado=0");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
```

```
5.          $stmt->close();
6.  }
```

Función consultarProfesoresModel()

Función que devuelve todos los registros en la tabla de maestro de la base de datos.

```
1.  public function consultarProfesoresModel(){
2.          $stmt = Conexion::conectar()->prepare("SELECT * FROM maestro
    WHERE  eliminado=0");
3.          $stmt->execute();
4.          return $stmt->fetchAll();
5.          $stmt->close();
6.  }
```

Función consultarProfesoresModel()

Función que obtiene todos los registros de la tabla de carreras.

```
1.  public function consultarCarrerasModel(){
2.          $stmt = Conexion::conectar()->prepare("SELECT * FROM carreras");
3.          $stmt->execute();
4.          return $stmt->fetchAll();
5.          $stmt->close();
6.  }
```

Función buscarProfesorModel()

Función que recibe como parámetro el id de un profesor y lo busca mediante una consulta SELECT en la base de datos y retorna sus datos.

```
1.  public function buscarProfesorModel($id){
2.          $stmt = Conexion::conectar()->prepare("SELECT * FROM maestro WHERE
    id_maestro = '$id'");
3.          $stmt->execute();
4.          return $stmt->fetch();
5.          $stmt->close();
6.  }
```

Función consultarAlumnosModel()

Este modelo prepara una consulta para tomar todos los registros existentes lógicamente en la base de datos de los alumnos

```
1.  public function consultarAlumnosModel(){
2.          $stmt = Conexion::conectar()->prepare("SELECT * FROM alumno WHERE
    eliminado=0");
3.          $stmt->execute();
4.          return $stmt->fetchAll();
5.          $stmt->close();
6.  }
```

Función buscarGrupoModel()

Este modelo prepara una consulta que consulta la información de un grupo en específico recibiendo como parámetro el id del grupo.

```
1. public function buscarGrupoModel($id){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM grupo WHERE id_grupo =
   '$id'");
3.         $stmt->execute();
4.         return $stmt->fetch();
5.         $stmt->close();
6. }
```

Función consultarActividadModel()

Este modelo prepara una consulta para tomar todos los registros existentes lógicamente en la base de datos de las actividades.

```
1. public function consultarActividadModel(){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM actividad WHERE
   eliminado=0");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
6. }
```

Función consultarSesionActivaModel()

Este modelo prepara una consulta para tomar todos los registros existentes lógicamente en la base de datos de las actividades.

```
1. public function consultarSesionActivaModel(){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM sesion inner join
   alumno on sesion.id_alumno = alumno.id_alumno inner join actividad on
   sesion.id_actividad = actividad.id_actividad inner join maestro on
   sesion.id_maestro = maestro.id_maestro");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
6. }
```

Función consultarSesionActivaModel()

Este modelo prepara una consulta para tomar todos los registros existentes lógicamente en la base de datos de las actividades.

```
1. public function consultarSesionActivaModel(){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM sesion inner join
   alumno on sesion.id_alumno = alumno.id_alumno inner join actividad on
   sesion.id_actividad = actividad.id_actividad inner join maestro on
   sesion.id_maestro = maestro.id_maestro");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
```

```
6. }
```

Función consultarSesionActivaModel()

Este modelo prepara una consulta que consulta la información de una carrera en específico recibiendo como parámetro el id de la carrera.

```
1. public function buscarCarreraModel($id){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM carreras WHERE
   id_carrera = '$id'");
3.         $stmt->execute();
4.         return $stmt->fetch();
5.         $stmt->close();
6. }
```

Función buscarActividadModel()

Este modelo prepara una consulta que consulta la información de una actividad en específico recibiendo como parámetro el id de la actividad.

```
1. public function buscarActividadModel($id){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM actividad WHERE
   id_actividad = '$id'");
3.         $stmt->execute();
4.         return $stmt->fetch();
5.         $stmt->close();
6. }
```

Función buscarAlumnoModel()

Este modelo prepara una consulta que consulta la información de un alumno en específico recibiendo como parámetro el id del alumno.

```
1. public function buscarAlumnoModel($id){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM alumno WHERE id_alumno
   = '$id'");
3.         $stmt->execute();
4.         return $stmt->fetch();
5.         $stmt->close();
6. }
```

Función modificarAlumnoModel()

Este modelo recibe como parámetros los datos de un alumno y el nombre de la tabla en la base de datos para realizar una consulta UPDATE y actualizar los datos devolviendo si fue exitosa o no.

```
1. public function modificarAlumnoModel($datosController, $tabla){
2.         $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET matricula_alumno =
   :matricula, nombre_alumno= :nombre, id_carrera= :carrera, img_alumno= :foto WHERE
   id_alumno= :id");
3.         $stmt>bindParam(':matricula',$datosController["matricula"]);
4.         $stmt->bindParam(':nombre',$datosController["nombre"]);
```

```
                $stmt->bindParam(':carrera',$datosController["carrera"]);
        $stmt->bindParam(':foto',$datosController["foto"]);
                $stmt->bindParam(':id',$datosController["id"]);
        return $stmt->execute();
        $stmt->close();
    }
```

Función eliminarAlumnoModel()

Este modelo elimina lógicamente, es decir cambia el estado de un alumno a eliminado en la base de datos.

```
public function eliminarAlumnoModel($id){
        $stmt = Conexion::conectar()->prepare("UPDATE alumno SET eliminado = 1
   WHERE id_alumno='$id'");
        return $stmt->execute();
        $stmt->close();
}
```

Función registrarProfesorModel()

Este modelo registra un profesor en la base de datos recibiendo los datos en un array y el nombre de la tabla.

```
public function registrarProfesorModel($datosController, $tabla){
        $emp = $datosController["emp"];
        $nombre = $datosController["nombre"];
        $fechaN = $datosController["fechaNa"];
        $tel = $datosController["telefono"];
        $dir = $datosController["direcc"];
        $us = $datosController["usuario"];
        $con = $datosController["contra"];
        $foto = $datosController["foto"];
           $stmt = Conexion::conectar()->prepare("INSERT
   INTO $tabla (numero_empleado, nombre_maestro, telefono_maestro, direccion_maestro,
   fecha_nac, img_maestro, usuario, pass, tipo_usuario) VALUES('$emp','$nombre',
   '$tel','$dir', '$fechaN', '$foto', '$us', '$con', 2)");
           return $stmt->execute();
           $stmt->close();
    }
```

Función consultarProfesorModel()

Este modelo prepara una consulta con todos los registros de profesores existentes lógicamente en la base de datos.

```
public function consultarProfesorModel(){
        $stmt = Conexion::conectar()->prepare("SELECT * FROM maestro WHERE
   eliminado=0 AND tipo_usuario=2");
        $stmt->execute();
        return $stmt->fetchAll();
        $stmt->close();
}
```

Función modificarProfesorModel()

Este modelo modifica la información de un registro de un profesor en específico en la base de datos.

```
1. public function modificarProfesorModel($datosController, $tabla){
2.         $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET numero_empleado =
   :emp, nombre_maestro= :nombre, telefono_maestro= :tel, direccion_maestro= :dir,
   fecha_nac= :fechaN, img_maestro= :foto, usuario= :us, pass= :con WHERE id_maestro=
   :id ");
3.         $stmt->bindParam(':emp',$datosController["emp"]);
4.                 $stmt->bindParam(':nombre',$datosController["nombre"]);
5.         $stmt->bindParam(':fechaN',$datosController["fechaNa"]);
6.         $stmt->bindParam(':tel',$datosController["tel"]);
7.                 $stmt->bindParam(':dir',$datosController["dir"]);
8.         $stmt->bindParam(':us',$datosController["usuario"]);
9.                 $stmt->bindParam(':con',$datosController["contra"]);
10.           $stmt->bindParam(':foto',$datosController["foto"]);
11.           $stmt->bindParam(':id',$datosController["id"]);
12.           return $stmt->execute();
13.           $stmt->close();
14.    }
```

Función limpiarAlumnoDeSesion()

Este modelo elimina lógicamente un registro de profesor en la base de datos.

```
1. public function limpiarAlumnoDeSesion($datosModel,$tabla){
2.         $stmt = Conexion::conectar()->prepare("DELETE FROM $tabla where id_alumno =
   '$datosModel'");
3.         return $stmt->execute();
4.         $stmt->close();
5. }
```

Función reponerActividad()

Este modelo obtiene la actividad especifica dependiendo de su id y actualiza la cantidad de lugares de esta.

```
1. public function reponerActividad($datosModel,$tabla){
2.         $stmt = Conexion::conectar()->prepare("SELECT lugares from actividad where
   id_actividad = '$datosModel'");
3.         $stmt->execute();
4.         $actividad = $stmt->fetch(PDO::FETCH_NUM);
5.         $new = $actividad[0]+1;
6.         $stmt = Conexion::conectar()->prepare("UPDATE actividad SET lugares
   = $new where id_actividad = '$datosModel'");
7.         return $stmt->execute();
8.         $stmt->close();
9. }
```

Función eliminarProfesorModel()

Este modelo elimina lógicamente un registro de profesor en la base de datos.

```
1. public function eliminarProfesorModel($id){
2.         $stmt = Conexion::conectar()->prepare("UPDATE maestro SET eliminado = 1
   WHERE id_maestro='$id'");
3.         return $stmt->execute();
4.         $stmt->close();
5. }
```

Función registrarActividadModel()

Este modelo registra una actividad en la base de datos recibiendo como parámetro los datos y el nombre de la tabla.

```
1. public function registrarActividadModel($datosController, $tabla){
2.         $nombre = $datosController["nombre"];
3.         $des = $datosController["desc"];
4.         $stmt = Conexion::conectar()->prepare("INSERT INTO $tabla(nombre_actividad,
   desc_actividad, lugares) VALUES('$nombre', '$des',:lugares)");
5.         $stmt>bindParam(':lugares',$datosController["lugares"]);
6.         return $stmt->execute();
7.         $stmt->close();
8. }
```

Función registrarSesionModel()

Este modelo registra una entrada de sesión en la base de datos.

```
1. public function registrarSesionModel($datosModel,$tabla){
2.         $stmt = Conexion::conectar()->prepare("INSERT INTO $tabla (id_alumno,
   id_actividad, id_maestro, fecha, hora_entrada, unidad) VALUES(:id_alumno,
   :id_actividad, :id_maestro ,:fecha,:hora_entrada,:unidad)");
3.         $stmt->bindParam(':id_alumno',$datosModel["alumno"]);
4.         $stmt->bindParam(':id_maestro',$datosModel["maestro"]);
5.         $stmt->bindParam(':id_actividad',$datosModel["actividad"]);
6.         $stmt->bindParam(':fecha',$datosModel["fecha"]);
7.         $stmt->bindParam(':hora_entrada',$datosModel["entrada"]);
8.         $stmt->bindParam(':unidad',$datosModel["unidad"]);
9.         return $stmt->execute();
10.           $stmt->close();
11.    }
```

Función registrarSesionDeAlumno()

Este modelo inserta una sesión completa en la base de datos.

```
1. public function registrarSesionDeAlumno($datosModel,$tabla){
2.         $stmt = Conexion::conectar()->prepare("INSERT INTO $tabla (id_alumno,
   id_actividad, id_maestro, fecha, hora_entrada, hora_salida, horas, unidad)
   VALUES(:id_alumno, :id_actividad, :id_maestro
   ,:fecha,:hora_entrada,:hora_salida,:horas,:unidad)");
3.         $stmt->bindParam(':id_alumno',$datosModel["id_alumno"]);
4.         $stmt->bindParam(':id_maestro',$datosModel["id_maestro"]);
5.         $stmt->bindParam(':id_actividad',$datosModel["id_actividad"]);
6.         $stmt->bindParam(':fecha',$datosModel["fecha"]);
7.         $stmt->bindParam(':hora_entrada',$datosModel["hora_entrada"]);
8.         $stmt->bindParam(':hora_salida',$datosModel["hora_salida"]);
9.         $stmt->bindParam(':horas',$datosModel["horas"]);
```

```
10.           $stmt->bindParam(':unidad',$datosModel["unidad"]);
11.           $stmt->execute();
12.           $stmt = Conexion::conectar()->prepare("DELETE FROM sesion where
   id_alumno = :id_alumno");
13.           $stmt->bindParam(':id_alumno',$datosModel["id_alumno"]);
14.           return $stmt->execute();
15.           $stmt->close();
16.    }
```

Función modificarActividadModel()

Esta actividad modifica la información de una actividad en particular de la base de datos.

```
1. public function modificarActividadModel($datosController, $tabla){
2.         $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET lugares =
   :lugares, nombre_actividad= :nombre, desc_actividad= :des WHERE id_actividad= :id
   ");
3.         $stmt->bindParam(':lugares',$datosController["lugares"]);
4.         $stmt->bindParam(':nombre',$datosController["nombre"]);
5.         $stmt->bindParam(':des',$datosController["desc"]);
6.         $stmt->bindParam(':id',$datosController["id"]);
7.         return $stmt->execute();
8.         $stmt->close();
9. }
```

Función eliminarActividadModel()

Este modelo elimina una actividad de la base de datos.

```
1. public function eliminarActividadModel($id){
2.         $stmt = Conexion::conectar()->prepare("UPDATE actividad SET eliminado = 1
   WHERE id_actividad='$id'");
3.         return $stmt->execute();
4.         $stmt->close();
5. }
```

Función eliminarAlumnoGrupoModel()

Modelo que elimina un alumno de un grupo de la base de datos.

```
1. public function eliminarAlumnoGrupoModel($id){
2.         $stmt = Conexion::conectar()->prepare("DELETE FROM alumno_grupo where
   id_alumno = :id_alumno");
3.         echo "<script> alert(DELETE FROM alumno_grupo where id_alumno
   = $id)</script>";
4.         $stmt->bindParam(":id_alumno",$id);
5.         return $stmt->execute();
6.         $stmt->close();
7. }
```

Función registrarGrupoModel()

Este modelo registra un grupo en la base de datos.

```
1. public function registrarGrupoModel($datosController, $tabla){
2.         $stmt = Conexion::conectar()->prepare("INSERT INTO $tabla(codigo_grupo,
   id_maestro, nivel) VALUES( :codigo, :maestro, :nivel)");
3.         $stmt->bindParam(':codigo',$datosController["codigo"]);
4.         $stmt->bindParam(':maestro',$datosController["maestro"]);
5.         $stmt->bindParam(':nivel',$datosController["nivel"]);
6.         return $stmt->execute();
7.         $stmt->close();
8. }
```

Función modificarGrupoModel()

Este modelo modifica la información de un grupo en particular de la base de datos.

```
1. public function modificarGrupoModel($datosController, $tabla){
2.         $stmt = Conexion::conectar()->prepare("UPDATE $tabla SET codigo_grupo=
   :cod, id_maestro= :maestro, nivel= :nivel WHERE id_grupo= :id");
3.         $stmt->bindParam(':cod',$datosController["codigo"]);
4.                 $stmt->bindParam(':maestro',$datosController["maestro"]);
5.         $stmt->bindParam(':nivel',$datosController["nivel"]);
6.         $stmt->bindParam(':id',$datosController["id"]);
7.         return $stmt->execute();
8.         $stmt->close();
9. }
```

Función eliminarGrupoModel()

Este modelo elimina un grupo de la base de datos.

```
1. public function eliminarGrupoModel($id){
2.         $stmt = Conexion::conectar()->prepare("UPDATE grupo SET eliminado = 1 WHERE
   id_grupo='$id'");
3.         return $stmt->execute();
4.         $stmt->close();
5. }
```

Función totalesModel()

Modelo que cuenta los registros de diferentes tablas.

```
1. public function totalesModel($tabla){
2.         //se prepara la consulta
3.         $stmt = Conexion::conectar()->prepare("SELECT * FROM $tabla WHERE
   eliminado=0");
4.         //se ejecuta la consulta
5.         $stmt->execute();
6.         //se retornan todas las filas devueltas
7.         return $stmt->rowCount();
8.         //se cierra la consulta
9.         $stmt->close();
10.    }
```

Función consultarAlumnoModel()

Este modelo consulta todos los registros existentes de alumnos lógicamente en la base de datos.

```
1. public function consultarAlumnoModel(){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM alumno
   WHERE  eliminado=0");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
6. }
```

Función consultarActividadesModel()

Este modelo consulta todos los registros existentes de actividades lógicamente en la base de datos.

```
1. public function consultarActividadesModel(){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM actividad
   WHERE  eliminado=0");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
6. }
```

Función consultarHorasModel()

Modelo que consulta las horas realizadas en cai de un alumno.

```
1. public function consultarHorasModel($id){
2.         $stmt = Conexion::conectar()->prepare("select alumno.matricula_alumno,
   alumno.nombre_alumno, entrada.unidad, sum(entrada.horas)as 'total' from entrada
   inner join alumno on entrada.id_alumno = alumno.id_alumno WHERE entrada.id_maestro
   = '$id' group by entrada.unidad, entrada.id_alumno");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
6. }
```

Función consultarIdMaestroModel()

Se consulta los datos de un maestro en específico.

```
1. public function consultarIdMaestroModel($usuario){
2.         $stmt = Conexion::conectar()->prepare("SELECT * FROM maestro WHERE usuario
   = '$usuario'");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
6. }
```

Función consultarReporteModel()

Modelo que consulta todos los registros con horas acumuladas.

```
1. public function consultarReporteModel(){
2.         $stmt = Conexion::conectar()->prepare("select
   alumno.id_alumno,alumno.matricula_alumno, alumno.nombre_alumno, entrada.unidad,
   (sum(entrada.horas) DIV 3) as 'total', MAX(maestro.nombre_maestro) as maestro,
   MAX(grupo.codigo_grupo) as grupo from entrada inner join alumno on
   entrada.id_alumno = alumno.id_alumno inner join maestro on entrada.id_maestro =
   maestro.id_maestro inner join grupo on grupo.id_maestro = entrada.id_maestro group
   by entrada.unidad, entrada.id_alumno");
3.         $stmt->execute();
4.         return $stmt->fetchAll();
5.         $stmt->close();
6. }
```

Función vistaDetalleModel()

Modelo que carga los detalles de un alumno y las horas realizadas en cai.

```
1. public function vistaDetalleModel($datosModel, $table){
2.         $stmt = Conexion::conectar()->prepare("SELECT alumno.matricula_alumno,
   alumno.nombre_alumno, entrada.fecha, entrada.hora_entrada, entrada.hora_salida,
   actividad.nombre_actividad, entrada.horas from $table inner join alumno on
   alumno.id_alumno = entrada.id_alumno inner join actividad on entrada.id_actividad =
   actividad.id_actividad where entrada.id_alumno = :id_alumno AND unidad = :unidad");
3.         $stmt->bindParam(':id_alumno',$datosModel["id"]);
4.         $stmt->bindParam(':unidad',$datosModel["unidad"]);
5.         $stmt->execute();
6.         return $stmt->fetchAll();
7.         $stmt->close();
8. }
```

C O N T R O L A D O R

El archivo controlador (controllers/controller.php) contiene toda la comunicación entre las vistas y la base de datos que es el modelo, a continuación, se muestra cada una de las funciones que contiene esta clase.

El controlador contiene una clase principal llamada “MvcController” que contiene todas las funciones necesarias para la comunicación de información.

Primero se tiene la función **pagina()** que incluye el archivo “template.php” que se encuentra en la carpeta de views y que contiene la plantilla principal visualmente hablando.

Después se tiene la función **enlacesPaginasController()** que funge como un traductor de url, que en base a el contenido de una variable llamda “action” en la url de la página muestra el módulo o página que se está intentando acceder.

```
1. Class MvcController{
2.         public function pagina(){
3.                 include "views/template.php";
4.         }
5.         public function enlacesPaginasController(){
6.                 //comprueba que el action este en la url de la pagina
7.                 if(isset( $_GET['action'])){
8.                         //almacena el action en una variable
9.                         $enlaces = $_GET['action'];
10.                   }else{
11.                           //si no hay action se ingresa index
12.                           $enlaces = "index";
13.                   }
14.                   //se manda a llamar el modelo de enlaces para obtener la url
   completa a la que se va a dirigir
15.                   $respuesta = Paginas::enlacesPaginasModel($enlaces);
16.                   //carga la url
17.                   include $respuesta;
18.           }
```

Función ingresoUsuarioController()

Funcion controlador que autentica a un usuario dependiendo de los datos ingresados en la página principal. Estos son identificados mediante un modelo y el controlador obtiene una respuesta verdadera o falsa que identifica si es un superadmin o un usuario normal.

```php
public function ingresoUsuarioController(){
        //se comprueba que el boton se haya presionado
        if(isset($_POST["ingresar"])){
                //Se almacena el usuario y la contrasena ingresada por el usuario en un array
                $datosController = array ("usuario" => $_POST["usuario"],
                        "contra" => $_POST["contra"]);
                //Se manda a llamar el modelo que identifica si los datos que ingreso el usuario estan almacenados en la base de datos
                $respuesta = Datos::ingresoUsuarioModel($datosController, "maestro");
                echo "<script>alert($respuesta);</script>";
                   //Si los datos son encontrados en el modelo se inicia la sesion
                   if($respuesta!="error"){
                           //se inicia la sesion
                           $_SESSION["validar"] = true;
                           $_SESSION["usuario"] = $datosController["usuario"];
                           $_SESSION["password"] = $datosController["contra"];
                           //superadmin
                           if($respuesta==1){
                                   header("location:index.php?action=inicioadmin");
                           }
                           //usuario
                           if($respuesta==2){
                                   header("location:index.php?action=iniciousuario");
                           }
                   }else{
                           echo '<script>
                           swal({title: "Error",
                           text: "Usuario o contraseña erróneos!",
                           type: "error"});
                           </script>';
                   }
           }
   }
```

Función vistaUnidadesController()

Funcion que muestra una tabla con todas las unidades cargadas en la base de datos almacenadas en una variable que contiene la respuesta de un modelo en específico.

```php
public function vistaUnidadesController(){
  $respuesta = Datos::vistaUnidadesModel('unidad');
  echo '<input value="'.date('Y-m-d',strtotime($respuesta[0]['fecha_inicio'])).'" type="date" class="form-control" id="fechainicio1" name="fechainicio1" placeholder="Fecha Inicio" required>
           </div>
           <div class="form-group col-md-3">
             <label for="fechafin">Fecha Fin</label>
             <input value="'.date('Y-m-d',strtotime($respuesta[0]['fecha_fin'])).'" type="date" class="form-control" id="fechafin1" name="fechafin1" placeholder="Fecha Fin" required>
           </div>
         </div>
          <div class="form-row">
              <div class="form-group col-md-2">
                <label></label>
                <h2 style="font-weight: bold; color:#0087FF">Unidad 2</h2>
```

```
14.               </div>
15.               <div class="form-group col-md-3">
16.                 <label for="fechainicio">Fecha Inicio</label>
17.                 <input value="'.date('Y-m-
   d',strtotime($respuesta[1]['fecha_inicio'])).'" type="date" class="form-control"
   id="fechainicio2" name="fechainicio2" placeholder="Fecha Inicio" required>
18.               </div>
19.               <div class="form-group col-md-3">
20.                 <label for="fechafin">Fecha Fin</label>
21.                 <input value="'.date('Y-m-
   d',strtotime($respuesta[1]['fecha_fin'])).'" type="date" class="form-control"
   id="fechafin2" name="fechafin2" placeholder="Fecha Fin" required>
22.               </div>
23.             </div>
24.           <div class="form-row">
25.               <div class="form-group col-md-2">
26.                 <label></label>
27.                 <h2 style="font-weight: bold; color:#0087FF">Unidad 3</h2>
28.               </div>
29.               ... continua código ...
30.               <div class="form-group col-md-3">
31.                 <label for="fechafin">Fecha Fin</label>
32.                 <input value="'.date('Y-m-
   d',strtotime($respuesta[11]['fecha_fin'])).'" type="date" class="form-control"
   id="fechafin12" name="fechafin12" placeholder="Fecha Fin" required>';
33.   }
```

Función actualizarUnidadesController()

Función que calcula las unidades dependiendo de sus fechas de inicio y fin. La actualización es por año, y abarca tres cuatrimestres que son los que son actualizados.

```
1. public function actualizarUnidadesController(){
2.   if(isset($_POST['fechainicio1'])){
3.
4.     for($i=0; $i<12;$i++){
5.       $id = $i+1;
6.       $datosController = array("inicio"=>$_POST['fechainicio'.$id],
7.                                "fin"=>$_POST['fechafin'.$id],
8.                                "id"=>$id);
9.       $respuesta = Datos::actualizarUnidadesModel("unidad",$datosController);
10.       }
11.       if($respuesta){
12.           echo"<script
   language='javascript'>window.location='index.php?action=unidades';</script>";
13.       }
14.     }
15.   }
```

Función registrarAlumnoController()

Este metodo toma los valores escritos en el formulario mediante el metodo de post para manadar a llamar al modelo que hace la insercion en la base de datos.

```
1. public function registrarAlumnoController(){
2.         if(isset($_POST["registrar"])){
3.         if(isset($_FILES["fileToUpload"]["name"]) && $_FILES["fileToUpload"]["name"
   ]!=""){
4.                 $b = $this->cargarImagen();
5.         }
6.         //Si la imagen se cargo se almacenan los datos que se ingresaron
7.         if($b!="0"){
8.
9.                 $datosController = array ("matricula"=>$_POST["matricula"], "nombre
   "=>$_POST["nombre"], "carrera"=>$_POST["carrera"], "foto"=>$b);
10.
11.                   $respuesta = Datos::registrarAlumnoModel($datosController, "alum
   no");
12.                   if($respuesta){
13.                           $_SESSION["registrado"]=1;
14.                           echo"<script
   language='javascript'>window.location='index.php?action=nuevoalumno';</script>";
15.                   }
16.                   else{
17.                           echo '<script>
18.                           swal({title: "Error",
19.                           text: "Esta matricula ya existe!",
20.                           type: "error"});
21.                           </script>';
22.                   }
23.           }
24.   }
25.   }
```

Función mostrarAlumnosGrupoController()

Metodo que se encarga de mostrar todos los alumnos que hay en el grupo que se eligio.

```
1. public function mostrarAlumnosGrupoController(){
2.         if(isset($_GET['id']))
3.         {
4.                 $datosController = $_GET["id"];
5.                 $respuesta = Datos::consultarGruposAlumnoModel($datosController);
6.                 if($respuesta){
7.                         foreach ($respuesta as $fila) {
8.                                 echo '<tr>
9.                                 <td>'.$fila["matricula_alumno"].'</td>
10.   <td>'.$fila["nombre_alumno"] . '</td>
11.   <td><button class="btn btn-danger"><a
   onclick="confirmarDelete('.$fila["id_alumno"].');"
   href="index.php?action=agregar_alumno&id='.$datosController.'&idBorrar='.$fila["id_
   alumno"].'" id="btn'.$fila["id_alumno"].'"style="color: white"><i class="fa fa-
   times"></i></a></button></td>
12.                           </tr>';
13.                   }
14.           }
15.   }
16.   }
```

Función mostrarGruposController()

Este metodo hace una llamada a un modelo que obtiene todos los grupos de la base de datos existentes de manera lógica.

```
1. public function mostrarGruposController(){
2.         $respuesta = Datos::consultarGruposModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         $resp2 = Datos::buscarProfesorModel($fila["id_maestro"]);
6.                         echo '<option
   value="'.$fila["id_grupo"].'">'.$fila["codigo_grupo"] . " -
   ". $resp2["nombre_maestro"] . " (Nivel " . $fila["nivel"]. ")" . '</option>';
7.                 }
8.         }
9. }
```

Función getUnitByDateController()

Metodo que hace una consulta en la base de datos en base a la fecha en la que nos encontramos. Con esta fecha buscaremos en que unidad nos encontramos. Y en la interfaz de la sesion se asignará automaticamente la unidad en la que nos encontramos.

```
1. public function getUnitByDateController(){
2.         $datosController =  date("Y-m-d");
3.         $respuesta = Datos::getUnitByDateModel($datosController);
4.         echo '<input type="text" class="form-control" id="unidad" name="unidad"
   required readonly value="'.$respuesta[0].'">';
5.
6. }
```

Función mostrarAlumnoController()

Este metodo hace una llamada a un modelo que obtiene todos los alumnos de la base de datos existentes de manera lógica.

```
1. public function mostrarAlumnoController(){
2.         $respuesta = Datos::consultarAlumnoModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         echo '<option
   value="'.$fila["id_alumno"].'">'.$fila["matricula_alumno"] . " -
   " .$fila["nombre_alumno"] . '</option>';
6.                 }
7.         }
8. }
```

Función mostrarActividadController()

Este metodo hace una llamada a un modelo que obtiene todas las actividades de la base de datos existentes de manera lógica.

```
1. public function mostrarActividadController(){
2.         $respuesta = Datos::consultarActividadesModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         echo '<option
   value="'.$fila["id_actividad"].'">'. $fila["nombre_actividad"] . '</option>';
6.                 }
7.         }
8. }
```

Función reiniciarController()

Función que limpia todas las sesiones actuales por cuestiones de organización.

```
1. public function reiniciarController(){
2.         if(isset($_GET['reiniciar'])){
3.                 if($_GET['reiniciar'] == 1){
4.                         $respuesta = Datos::reiniciarModel();
5.                 }
6.         }
7. }
```

Función mostrarCarrerasController()

Este método hace una llamada a un modelo que obtiene todas las carreras de la base de datos existentes de manera lógica.

```
1. public function mostrarCarrerasController(){
2.         $respuesta = Datos::consultarCarrerasModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         echo '<option
   value="'.$fila["id_carrera"].'">'.$fila["nombre"] . "
   (" . $fila["siglas"]. ")" . '</option>';
6.                 }
7.         }
8.
9. }
```

Función vistaAlumnosController()

Metodo que obtiene todos los registros de la base de datos y los imprime en una tabla.

```
1. public function vistaAlumnosController(){
2.         $respuesta = Datos::consultarAlumnosModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         $carrera = Datos::buscarCarreraModel($fila["id_carrera"]);
6.                         $carrera2 = $carrera["siglas"];
7.
8.                         echo '<tr>
9.                         <td>'.$fila["matricula_alumno"].'</td>
```

```
                        <td>'.$fila["nombre_alumno"] . '</td>
                        <td>'.$carrera2.'</td>';

                        if($fila["img_alumno"]!=""){
                                echo '<td><a
   href="uploads/'.$fila["img_alumno"].'"class="btn btn-secondary"><i class="fa fa-
   photo"></i></a></td>';
                        }else{
                                echo '<td>Sin imágen</td>';
                        }

                        echo '<td><a
   href="index.php?action=modificaralumno&id='.$fila["id_alumno"].'"class="btn btn-
   primary"><i class="fa fa-edit"></i></a></td>
                        <td><button class="btn btn-danger"><a
   onclick="confirmarDelete('.$fila["id_alumno"].');"
   href="index.php?action=alumnos&idBorrar='.$fila["id_alumno"].'"
   id="btn'.$fila["id_alumno"].'" style="color: white"><i class="fa fa-
   times"></i></a></button></td>
                    </tr>';
            }
    }
    }
```

Función editarAlumnoController()

Método que toma la informacion de un registro y la devuelve en un formulario.

```
public function editarAlumnoController(){
        if(isset($_GET["id"])){
                $id = $_GET["id"];
                $respuesta = Datos::buscarAlumnoModel($id);
                if($respuesta){
                        echo "<div class='form-row'>
                        <div class='form-group col-md-3'>
                                <label for='matricula'>Matrícula</label>
                                <input type='text' class='form-control'
   id='matricula' name='matricula' placeholder='Matrícula'
   value='".$respuesta["matricula_alumno"]. "' required>
                        </div>
                        <div class='form-group col-md-9'>
                                <label for='nombre'>Nombre</label>
                                <input type='text' class='form-control'
   id='nombre' name='nombre' placeholder='Nombre completo'
   value='".$respuesta["nombre_alumno"]."' required>
                        </div>
                    </div>
                    <div class='form-row'>
                        <div class='form-group col-md-6'>
                                <label for='carrera'>Carrera</label>
                                <select id='carrera' name='carrera' class='form-
   control select2'>";
                                        $c = Datos::consultarCarrerasModel();
                                        if($c){
                                                foreach ($c as $fila) {
                                                        if($respuesta["id_carrer
   a"]==$fila["id_carrera"]){
```

```
24.                                                                 echo '<option
   value="'.$fila["id_carrera"].'" selected>'.$fila["nombre"] . "
   (" . $fila["siglas"]. ")" . '</option>';
25.                                                             }else{
26.                                                                 echo '<option
   value="'.$fila["id_carrera"].'">'.$fila["nombre"] . "
   (" . $fila["siglas"]. ")" . '</option>';
27.                                                             }
28.                                                         }
29.                                                 }
30.                                                 echo "</select>
31.                                         </div>
32.                                 </div>
33.                                 <div class='form-row'>
34.                                         <div class='form-group col-md-6'>
35.                                                 <label
   for='fileToUpload'>Foto</label><br>
36.                                                 <input type='hidden' name='iman'
   id='iman' value='".$respuesta["img_alumno"]."'>
37.                                                 <a
   href='uploads/".$respuesta["img_alumno"]."' class='btn btn-primary'><i class='fa
   fa-photo'></i> Ver Imágen</a>
38.                                                 <br><br>
39.                                                 <input type='file' class='form-control-
   file' name='fileToUpload' id='fileToUpload'>
40.                                         </div>
41.                                 </div><br>";
42.                         }
43.                 }
44.         }
```

Función modificarAlumnoController()

Método que almacena la informacion de un formulario y actualiza el registro en la base de datos.

```
1. public function modificarAlumnoController(){
2.         $id = $_GET["id"];
3.         if(isset($_POST["modificar"])){
4.                 if(isset($_FILES["fileToUpload"]["name"]) && $_FILES["fileToUpload"
   ]["name"]!=""){
5.                         $b = $this->cargarImagen();
6.                 }else{
7.                         $b = $_POST["iman"];
8.                 }
9.                 if(isset($b)){
10.                         $datosController = array ("id"=>$id,"matricula"=>$_POST[
   "matricula"], "nombre"=>$_POST["nombre"],  "carrera"=>$_POST["carrera"], "foto"=>$b
   );
11.                         $respuesta = Datos::modificarAlumnoModel($datosControlle
   r, "alumno");
12.                         if($respuesta){
13.                                 $_SESSION["modificado"]=1;
14.                                 echo"<script
   language='javascript'>window.location='index.php?action=modificaralumno&id=".$id."'
   ;</script>";
15.                         }
16.                 }
17.         }
```

```
18.    }
```

Función eliminarAlumnoController()

Método que cambia el estado de un registro para eliminarlo lógicamente de la base de datos.

```
1. public function eliminarAlumnoController(){
2.          if(isset($_GET["idBorrar"])){
3.                  $id = $_GET["idBorrar"];
4.                  $respuesta = Datos::eliminarAlumnoModel($id);
5.
6.                  if($respuesta){
7.                          $_SESSION["eliminado"]=1;
8.                          echo"<script
   language='javascript'>window.location='index.php?action=alumnos';</script>";
9.                  }
10.           }
11.    }
```

Función registrarProfesorController()

Este método toma los valores escritos en el formulario mediante el método de post para manadar a llamar al modelo que hace la inserción en la base de datos.

```
1. public function registrarProfesorController(){
2.          if(isset($_POST["registrar"])){
3.                  $b = $this->cargarImagen();
4.                          //Si la imagen se cargo se almacenan los datos que se
   ingresaron
5.                  if($b!="0"){
6.                          $datosController = array ("emp"=>$_POST["noempleado"], "nom
   bre"=>$_POST["nombre"], "fechaNa"=>$_POST["fechaN"], "telefono"=>$_POST["tel"], "di
   recc"=>$_POST["dir"], "usuario"=>$_POST["usuario"], "contra"=>$_POST["contra"], "fo
   to"=>$b);
7.
8.                          $respuesta = Datos::registrarProfesorModel($datosController
   , "maestro");
9.                          if($respuesta){
10.                                 $_SESSION["registrado"]=1;
11.                                 echo"<script
   language='javascript'>window.location='index.php?action=nuevoprofe';</script>";
12.                             }else{
13.                                 echo '<script>
14.                                 swal({title: "Error",
15.                                 text: "Esta numero de empleado ya existe!",
16.                                 type: "error"});
17.                             </script>';
18.                     }
19.
20.
21.           }
22.    }
23.    }
```

Función vistaProfesoresController()

Método que obtiene todos los registros de la base de datos y los imprime en una tabla.

```
1. public function vistaProfesoresController(){
2.         $respuesta = Datos::consultarProfesorModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         echo '<tr>
6.                         <td>'.$fila["numero_empleado"].'</td>
7.                         <td>'.$fila["nombre_maestro"] . '</td>
8.                         <td>'.$fila["telefono_maestro"].'</td>
9.                         <td>'.$fila["direccion_maestro"].'</td>
10.                           <td>'.$fila["fecha_nac"].'</td>';
11.
12.                           if($fila["img_maestro"]!=""){
13.                                   echo '<td><a
   href="uploads/'.$fila["img_maestro"].'"class="btn btn-secondary"><i class="fa fa-
   photo"></i></a></td>';
14.                           }else{
15.                                   echo '<td>Sin imágen</td>';
16.                           }
17.
18.                           echo '<td>'.$fila["usuario"].'</td>
19.                           <td>'.$fila["pass"].'</td>
20.                           <td><a
   href="index.php?action=modificarprofe&id='.$fila["id_maestro"].'"class="btn btn-
   primary"><i class="fa fa-edit"></i></a></td>
21.                           <td><a class="btn btn-danger"
   onclick="confirmarDelete('.$fila["id_maestro"].');"
   href="index.php?action=profesores&idBorrar='.$fila["id_maestro"].'"
   id="btn'.$fila["id_maestro"].'"><i class="fa fa-times"></i></a></td>
22.                   </tr>';
23.           }
24.   }
25.   }
```

Función editarProfesorController()

Método que toma la informacion de un registro y la devuelve en un formulario.

```
1. public function editarProfesorController(){
2.
3.         if(isset($_GET["id"])){
4.                 $id = $_GET["id"];
5.
6.                 $respuesta = Datos::buscarProfesorModel($id);
7.
8.                 if($respuesta){
9.                         echo "<div class='form-row'>
10.                           <div class='form-group col-md-3'>
11.                                   <label for='noempleado'>Número de
   Empleado</label>
12.                                   <input type='text' class='form-control'
   id='noempleado' name='noempleado' value='".$respuesta["numero_empleado"]."'
   placeholder='Número de empleado' required>
13.                           </div>
```

```
14.                             <div class='form-group col-md-9'>
15.                                     <label for='nombre'>Nombre</label>
16.                                     <input type='text' class='form-control'
   id='nombre' name='nombre' value='".$respuesta["nombre_maestro"]."'
   placeholder='Nombre completo' required>
17.                             </div>
18.                     </div>
19.                     <div class='form-row'>
20.                             <div class='form-group col-md-3'>
21.                                     <label for='fechaN'>Fecha de Nacimiento</label>
22.                                     <input type='date' class='form-control'
   id='fechaN' name='fechaN' value='".$respuesta["fecha_nac"]."' placeholder='Fecha de
   Nacimiento' required>
23.                             </div>
24.                             <div class='form-group col-md-3'>
25.                                     <label for='tel'>Teléfono</label>
26.                                     <input type='text' class='form-control' id='tel'
   name='tel' value='".$respuesta["telefono_maestro"]."' placeholder='Teléfono de
   Celular' required>
27.                             </div>
28.                             <div class='form-group col-md-6'>
29.                                     <label for='tel'>Dirección</label>
30.                                     <input type='text' class='form-control' id='dir'
   name='dir' value='".$respuesta["direccion_maestro"]."' placeholder='Dirección'
   required>
31.                             </div>
32.                     </div>
33.                     <div class='row'>
34.                             <div class='form-group col-md-3'>
35.                                     <label for='usuario'>Usuario</label>
36.                                     <input type='text' class='form-control'
   id='usuario' name='usuario' value='".$respuesta["usuario"]."' placeholder='Usuario'
   required>
37.                             </div>
38.                             <div class='form-group col-md-3'>
39.                                     <label for='contra'>Contraseña</label>
40.                                     <input type='text' class='form-control'
   id='contra' name='contra' value='".$respuesta["pass"]."' placeholder='Contraseña'
   required>
41.                             </div>
42.                             <div class='form-group col-md-2'>
43.                                     <label for='fileToUpload'>Foto</label><br>
44.                                     <input type='hidden' name='iman' id='iman'
   value='".$respuesta["img_maestro"]."'>
45.                                     <a href='uploads/".$respuesta["img_maestro"]."'
   class='btn btn-primary'><i class='fa fa-photo'></i> Ver Imágen</a>
46.                             </div>
47.                             <div class='form-group col-md-4'>
48.                                     <input type='file' style='margin-top: 35px'
   class='form-control-file' name='fileToUpload' id='fileToUpload'>
49.                             </div>
50.                     </div><br>";
51.             }
52.     }
53.
54.     }
```

Función modificarProfesorController()

Método que almacena la información de un formulario y actualiza el registro en la base de datos.

```
1. public function modificarProfesorController(){
2.
3.         $id = $_GET["id"];
4.
5.         if(isset($_POST["modificar"])){
6.
7.                 if(isset($_FILES["fileToUpload"]["name"]) && $_FILES["fileToUpload"
   ]["name"]!=""){
8.                         $b = $this->cargarImagen();
9.                 }else{
10.                         $b = $_POST["iman"];
11.                    }
12.
13.                    if(isset($b)){
14.
15.                         $datosController = array ("id"=>$id,"emp"=>$_POST["noemp
   leado"], "nombre"=>$_POST["nombre"], "fechaNa"=>$_POST["fechaN"], "tel"=>$_POST["te
   l"], "dir"=>$_POST["dir"], "usuario"=>$_POST["usuario"], "contra"=>$_POST["contra"]
   , "foto"=>$b);
16.
17.                         $respuesta = Datos::modificarProfesorModel($datosControl
   ler, "maestro");
18.                         if($respuesta){
19.                                 $_SESSION["modificado"]=1;
20.                                 echo"<script
   language='javascript'>window.location='index.php?action=modificarprofe&id=".$id."';
   </script>";
21.                         }
22.
23.                    }
24.
25.           }
26.
27.   }
```

Función eliminarProfesorController()

Método que cambia el estado de un registro para eliminarlo lógicamente de la base de datos.

```
1. public function eliminarProfesorController(){
2.         if(isset($_GET["idBorrar"])){
3.                 $id = $_GET["idBorrar"];
4.                 $respuesta = Datos::eliminarProfesorModel($id);
5.                 if($respuesta){
6.                         $_SESSION["eliminado"]=1;
7.                         echo"<script
   language='javascript'>window.location='index.php?action=profesores';</script>";
8.                 }
9.         }
10.   }
```

Función registrarActividadController()

Este método toma los valores escritos en el formulario mediante el método de post para manadar a llamar al modelo que hace la inserción en la base de datos.

```
1. public function registrarActividadController(){
2.         if(isset($_POST["registrar"])){
3.
4.                 $datosController = array ("nombre"=>$_POST["nombre"],
5.                         "desc"=>$_POST["desc"],
6.                         "lugares"=>$_POST["lugares"]);
7.
8.                 $respuesta = Datos::registrarActividadModel($datosController, "acti
   vidad");
9.                 if($respuesta){
10.                         $_SESSION["registrado"]=1;
11.                         echo"<script
   language='javascript'>window.location='index.php?action=nuevaactividad';</script>";
12.                     }
13.
14.           }
15.    }
```

Función registrarSesionController()

Método que se encarga de registrar una sesión en una tabla temporal en la base de datos, que será donde se tendra control de todas las sesiones que hay en el dia, en dicha tabla habrá un botón para liberar al alumno de la sesión siempre y cuando se hayan cumplido los requisitos de dicha sesión (tiempo, actividad, etc.)

```
1. public function registrarSesionController(){
2.         if(isset($_POST["registrar"]) && $_POST['alumno']!="" && $_POST['maestro']!
   =""){
3.
4.                 $datosController = array ("alumno"=>$_POST["alumno"],
5.                         "maestro"=>$_POST["maestro"],
6.                         "actividad"=>$_POST['act'],
7.                         "fecha"=>$_POST['fecha'],
8.                         "entrada"=>$_POST['entrada'],
9.                         "unidad"=>$_POST['unidad']);
10.                    $respuesta = Datos::registrarSesionModel($datosController, "sesi
   on");
11.                    if($respuesta){
12.                         $_SESSION["registrado"]=1;
13.                         echo"<script
   language='javascript'>window.location='index.php?action=nuevasesion';</script>";
14.                    }
15.
16.           }
17.    }
```

Función vistaSesionesActivasController()

Método que se encarga de hacer la vista de las sesiones que hay hasta el momento.

```
1. public function vistaSesionesActivasController(){
2.         $respuesta = Datos::consultarSesionActivaModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         echo '<tr>
6.                         <td>'.$fila["nombre_alumno"].'</td>
7.                         <td>'.$fila["nombre_actividad"].'</td>
8.                         <td>'.$fila["nombre_maestro"].'</td>
9.                         <td>'.$fila["fecha"].'</td>
10.                           <td id="hora">'.$fila["hora_entrada"].'</td>
11.                           <td>'.$fila["unidad"].'</td>
12.                           <td><button class="btn btn-success"><a
   onclick="confirmarSalida('.$fila["id_alumno"].');"
   href="index.php?action=sesiones&idLiberar='.$fila["id_alumno"].'"
   id="btn'.$fila["id_alumno"].'"style="color: white">Liberar</a></button></td>
13.                   </tr>';
14.           }
15.   }
16.   }
```

Función liberarAlumnoController()

Método que se encarga de liberar al alumno en una sesión. Si el alumno superó cuatro horas seguidas no se podrá librerar, ya que existe este límite, además de comunicar ciertos avisos de acciones que pueden ocurrir si la sesión se cancela a mitad del tiempo.

```
1. public function liberarAlumnoController(){
2.         if(isset($_GET["idLiberar"])){
3.                 $datosController = $_GET['idLiberar'];
4.                 $respuesta = Datos::consultarSesionAlumnoModel($datosController);
5.
6.       //Arreglo que tiene un conteo de horas en minutos para saber como dependiendo
   de los minutos que hizo el alumno cuual es la hora
7.       //mas cercana a ese alumno
8.                 $max = array(60,120,180,240,300,360,420);
9.
10.         //Se trae la hora de entrada del alumno que se pretende liberar
11.                 $oldHour = explode(":", $respuesta["hora_entrada"]);
12.         //Se le quitan los minutos en los que llego tarde para hacer el proceso
   mas optimizado
13.                 $newHour = $oldHour[0].":00";
14.
15.         // NOOOOOOOOOOOOOOOOOW - - - - date('H:i', strtotime('-5 hour'))
16.         //Se trae la hora en la que estamos
17.                 $now = date('H:i', strtotime('-5 hour'));
18.
19.         //Obtenemos los minutos que han sido transcurrido desde la hora de entrada
   del alumno hasta la hora en la que estamos
20.                 $minutes = abs(strtotime($newHour) - strtotime($now)) / 60;
21.
22.                 $valMax = 10000;
```

```
23.         //Se revisa todo el arreglo de las horas (max) y se obtiene la distancia
   mas corta entre los minutos transcurridos y todos los valores del arreglo
24.                     for($i = 0; $i<count($max); $i++){
25.                             $op = $max[$i] - $minutes;
26.           //Si la distancia entre horas es positiva
27.                             if($op < $valMax && $op>-1){
28.                                     $valMax = $max[$i] - $minutes;
29.             //Entonces la hora sera el valor que se obtuvo de ese arreglo,
   entonces esa es la hora mas cercana a la que esta el alumno
30.                                     $hora = $i+1;
31.                             }
32.                     }
33.
34.         //Si se quiere salir 11+ minutos antes de que se acabe la hora, entonces
   la hora no se cuenta y se le resta
35.                     if($max[$hora-1] - $minutes > 10){
36.                             $hora--;
37.
38.                     }
39.
40.         //Si hizo mas de una hora y no sobre paso las 4 horas entonces se hace el
   registro de la sesion
41.                     if($hora > 0 && $hora < 4){
42.                             $datosController = array("id_alumno"=>$respuesta["id_alu
   mno"],
43.                                     "id_actividad"=>$respuesta["id_actividad"],
44.                                     "id_maestro"=>$respuesta["id_maestro"],
45.                                     "fecha"=>$respuesta["fecha"],
46.                                     "hora_entrada"=>$respuesta["hora_entrada"],
47.                                     "hora_salida"=>$now,
48.                                     "horas"=>$hora,
49.                                     "unidad"=>$respuesta["unidad"]);
50.           //Se limpia al alumno de la tabla temporal sesion
51.                             $respuesta2 = Datos::registrarSesionDeAlumno($datosContr
   oller,"entrada");
52.                     }
53.                     else{
54.           //De lo contrario solo se limpia el alumno de la tabla temporal sesion
   sin registrar dicha sesion
55.                             $datosController = $respuesta["id_alumno"];
56.                             $respuesta2 = Datos::limpiarAlumnoDeSesion($datosControl
   ler,"sesion");
57.
58.                     }
59.         //El lugar de la actividad se le suma un lugar mas debido a que fue
   liberado el alumno sin importar si cumplio las restricciones o no
60.                     $datosController = $respuesta["id_actividad"];
61.                     $respuesta3 = Datos::reponerActividad($datosController,"activida
   d");
62.                     echo"<script
   language='javascript'>window.location='index.php?action=sesiones';</script>";
63.           }
64.
65.   }
66.
```

Función vistaActividadController()

Método que obtiene todos los registros de la base de datos y los imprime en una tabla.

```
1. public function vistaActividadController(){
2.         $respuesta = Datos::consultarActividadModel();
3.         if($respuesta){
4.                 foreach ($respuesta as $fila) {
5.                         echo '<tr>
6.                         <td>'.$fila["nombre_actividad"].'</td>
7.                         <td>'.$fila["desc_actividad"] . '</td>
8.                         <td><a
   href="index.php?action=modificaractividad&id='.$fila["id_actividad"].'"class="btn
   btn-primary"><i class="fa fa-edit"></i></a></td>
9.                         <td><button class="btn btn-danger"><a
   onclick="confirmarDelete('.$fila["id_actividad"].');"
   href="index.php?action=actividades&idBorrar='.$fila["id_actividad"].'"
   id="btn'.$fila["id_actividad"].'"style="color: white"><i class="fa fa-
   times"></i></a></button></td>
10.                   </tr>';
11.           }
12.   }
```

Función editarActividadController()

Método que toma la información de un registro y la devuelve en un formulario.

```
1. public function editarActividadController(){
2.
3.         if(isset($_GET["id"])){
4.                 $id = $_GET["id"];
5.
6.                 $respuesta = Datos::buscarActividadModel($id);
7.
8.                 if($respuesta){
9.                         echo '<div class="form-row">
10.                           <div class="form-group col-md-3">
11.                                   <label for="nombre">Nombre de la
   actividad</label>
12.                                   <input type="text" class="form-control"
   id="nombre" name="nombre" value="'.$respuesta["nombre_actividad"].'"
   placeholder="Nombre de la actividad" required>
13.                           </div>
14.                           <div class="form-group col-md-9">
15.                                   <label for="desc">Descripción</label>
16.                                   <input type="text" class="form-control"
   id="desc" name="desc" value="'.$respuesta["desc_actividad"].'"
   placeholder="Descripción de la actividad" required>
17.                           </div>
18.                           <div class="form-group col-md-3">
19.                                   <label for="nombre">Lugares</label>
20.                                   <input type="number"
   value="'.$respuesta["lugares"].'" max="30" min="1" class="form-control"
   id="lugares" name="lugares" required>
21.                           </div>
22.                   </div>';
23.           }
```

```
24.    }
25.    }
```

Función modificarActividadController()

Método que almacena la información de un formulario y actualiza el registro en la base de datos.

```
1. public function modificarActividadController(){
2.          $id = $_GET["id"];
3.          if(isset($_POST["modificar"])){
4.                  $datosController = array ("id"=>$id,"nombre"=>$_POST["nombre"], "de
   sc"=>$_POST["desc"] , "lugares"=>$_POST["lugares"]);
5.
6.                  $respuesta = Datos::modificarActividadModel($datosController, "acti
   vidad");
7.                  if($respuesta){
8.                          $_SESSION["modificado"]=1;
9.                          echo"<script
   language='javascript'>window.location='index.php?action=modificaractividad&id=".$id
   ."';</script>";
10.                     }
11.           }
12.    }
```

Función eliminarActividadController()

Método que cambia el estado de un registro para eliminarlo lógicamente de la base de datos.

```
1. public function eliminarActividadController(){
2.          if(isset($_GET["idBorrar"])){
3.                  $id = $_GET["idBorrar"];
4.                  $respuesta = Datos::eliminarActividadModel($id);
5.                  if($respuesta){
6.                          $_SESSION["eliminado"]=1;
7.                          echo"<script
   language='javascript'>window.location='index.php?action=actividades';</script>";
8.                  }
9.          }
10.    }
```

Función eliminarAlumnoGrupoController()

Funcion que elimina un alumno de un grupo de la base de datos.

```
1. public function eliminarAlumnoGrupoController(){
2.          if(isset($_GET["idBorrar"])){
3.                  $id = $_GET["idBorrar"];
4.                  $idGrupo = $_GET["id"];
5.                  $respuesta = Datos::eliminarAlumnoGrupoModel($id);
6.                  if($respuesta){
7.                          $_SESSION["eliminado"]=1;
8.                          echo"<script
   language='javascript'>window.location='index.php?action=agregar_alumno&id=$idGrupo'
   ;</script>";
```

```
                }
            }
    }
```

Función mostrarProfesoresController()

Método que muestra los profesores de la base de datos.

```
public function mostrarProfesoresController(){
        $respuesta = Datos::consultarProfesoresModel();
        if($respuesta){
                foreach ($respuesta as $fila) {
                        echo '<option
value="'.$fila["id_maestro"].'">'.$fila["nombre_maestro"] . '</option>';
                }
        }
}
```

Función registrarGrupoController()

Método que registra un grupo en la base de datos con la llmada a un modelo que realiza la inserción en la base de datos.

```
public function registrarGrupoController(){


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

                $datosController = array ("codigo"=>$_POST["codigo"], "maestro"=>$_
POST["maestro"], "nivel"=>$_POST["nivel"]);

                $respuesta = Datos::registrarGrupoModel($datosController, "grupo");
                if($respuesta){
                        $_SESSION["registrado"]=1;
                        echo"<script
language='javascript'>window.location='index.php?action=nuevogrupo';</script>";
                  }

            }
    }
```

Función vistaGrupoController()

Método que obtiene todos los registros de la base de datos y los imprime en una tabla.

```
public function vistaGrupoController(){
        $respuesta = Datos::consultarGruposModel();
        if($respuesta){
                foreach ($respuesta as $fila) {
                        $resp2 = Datos::buscarProfesorModel($fila["id_maestro"]);
                        echo '<tr>
                        <td>'.$fila["codigo_grupo"].'</td>
                        <td>'.$resp2["nombre_maestro"] . '</td>
                        <td>'.$fila["nivel"] . '</td>
```

```
                            <td><a
   href="index.php?action=modificargrupo&id='.$fila["id_grupo"].'"class="btn btn-
   primary"><i class="fa fa-edit"></i></a></td>
                            <td><a
   href="index.php?action=agregar_alumno&id='.$fila["id_grupo"].'"class="btn btn-
   primary"><i class="fa fa-user-plus"></i></a></td>
                            <td><button class="btn btn-danger"><a
   onclick="confirmarDelete('.$fila["id_grupo"].');"
   href="index.php?action=grupos&idBorrar='.$fila["id_grupo"].'"
   id="btn'.$fila["id_grupo"].'" style="color: white"><i class="fa fa-
   times"></i></a></button></td>
                    </tr>';
            }
    }
    }
```

Función cargarImagen()

Método que carga una imagen desde el explorador, la copia en una carpeta dentro del proyecto y almacena la dirección de esta en un campo de la base de datos.

```
public function cargarImagen(){
        $ruta="";
        $target_dir = "uploads/";

    //Se cambia el nombre de la foto obteniendo la fecha en la que estamos, asi
   como los milisegundos, para corroborar
    //que se pueda subir la misma foto, porque cada milisegundo cambiara el nombre
   con dicho mlisegundo
        $target_file = $target_dir . basename($_FILES["fileToUpload"]["name"]);
        $micro_date = microtime();
        $date_array = explode(" ",$micro_date);
           $date = date("Y-m-dH-i-s",$date_array[1]);

           $oldName = basename($_FILES["fileToUpload"]["name"]);
           $photoName = explode(".", $oldName);
           $newName = $photoName[0] . "_" . $date . "." .$photoName[1];

           $_FILES["fileToUpload"]["name"] = $newName;

           $target_file = $target_dir . $newName;
           $uploadOk = 1;
           $imageFileType = strtolower(pathinfo($target_file,PATHINFO_EXTENSION));
      // Check if image file is a actual image or fake image
           if(isset($_POST["aceptar"])) {
                   $check = getimagesize($_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;
           }
```

```
                    // Check file size
            if ($_FILES["fileToUpload"]["size"] > 500000000000000) {
                    echo "Sorry, your file is too large.";
                    $uploadOk = 0;
            }
                    // Allow certain file formats
            if($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType
   != "jpeg"
                    && $imageFileType != "gif" ) {
                    echo "Sorry, only JPG, JPEG, PNG & GIF files are allowed.";
            $uploadOk = 0;
    }
                    // Check if $uploadOk is set to 0 by an error
    if ($uploadOk == 0) {
            echo "Sorry, your file was not uploaded.";
                    // if everything is ok, try to upload file
    } else {
            if (move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], $target_file
   )) {
                    return $ruta = basename($_FILES["fileToUpload"]["name"]);
                            //echo "The file ". basename(
   $_FILES["fileToUpload"]["name"]). " has been uploaded.";
            } else {
                    echo "Sorry, there was an error uploading your file.";
                    return "0";
            }
    }
    }
```

Función editarGrupoController()

Método que toma la información de un registro y la devuelve en un formulario.

```
public function editarGrupoController(){

        if(isset($_GET["id"])){
                $id = $_GET["id"];

                $respuesta = Datos::buscarGrupoModel($id);
                $profes = Datos::consultarProfesoresModel();

                if($respuesta){
                        echo '<div class="form-row">
                        <div class="form-group col-md-3">
                                <label for="codigo">Código del grupo</label>
                                <input type="text" class="form-control"
   id="codigo" name="codigo" value="'.$respuesta["codigo_grupo"].'"
   placeholder="Código del grupo" required>
                        </div>
                    </div>
                    <div class="form-row">
                        <div class="form-group col-md-5">
                                <label for="maestro">Maestro</label>
                                <select id="maestro" name="maestro" class="form-
   control select2">';
                                        foreach ($profes as $fila) {

                                                if($fila["id_maestro"]==$respues
   ta["id_maestro"]){
```

```
23.                                                     echo '<option
   value="'.$fila["id_maestro"].'" selected>'.$fila["nombre_maestro"].'</option>';
24.                                                 }else{
25.                                                     echo '<option
   value="'.$fila["id_maestro"].'">'.$fila["nombre_maestro"].'</option>';
26.                                                 }
27.
28.                                             }
29.                                             echo '</select>
30.                                     </div>
31.                                     <div class="form-group col-md-2">
32.                                             <label for="nivel">Nivel</label>
33.                                             <select id="nivel" name="nivel"
   class="form-control select2">';
34.                                                 for ($i=1; $i <=9 ; $i++) {
35.                                                     if($i==$respuesta["nivel
   "]){
36.                                                         echo '<option
   value="'.$i.'" selected>'.$i.'</option>';
37.                                                     }else{
38.                                                         echo '<option
   value="'.$i.'">'.$i.'</option>';
39.                                                     }
40.                                                 }
41.                                                 echo '</select>
42.                                             </div>
43.                                     </div><br>';
44.                             }
45.                     }
46.             }
```

Función modificarGrupoController()

Método que almacena la información de un formulario y actualiza el registro en la base de datos.

```
1. public function modificarGrupoController(){
2.         $id = $_GET["id"];
3.         if(isset($_POST["modificar"])){
4.                 $datosController = array ("id"=>$id,"codigo"=>$_POST["codigo"], "ma
   estro"=>$_POST["maestro"], "nivel"=>$_POST["nivel"]);
5.                 $respuesta = Datos::modificarGrupoModel($datosController, "grupo");
6.                         if($respuesta){
7.                                 $_SESSION["modificado"]=1;
8.                                 echo"<script
   language='javascript'>window.location='index.php?action=modificargrupo&id=".$id."';
   </script>";
9.                         }
10.                     }
11.             }
```

Función eliminarGrupoController()

Método que cambia el estado de un registro para eliminarlo lógicamente de la base de datos.

```
1. public function eliminarGrupoController(){
2.         if(isset($_GET["idBorrar"])){
3.                 $id = $_GET["idBorrar"];
4.                 $respuesta = Datos::eliminarGrupoModel($id);
5.                 if($respuesta){
6.                         $_SESSION["eliminado"]=1;
7.                         echo"<script
   language='javascript'>window.location='index.php?action=grupos';</script>";
8.                 }
9.         }
10.    }
```

Funciónes numeroAlumnos(), numeroProfesores(), numeroGrupos()

Estos métodos cuentan los registros existentes en cada una de sus tablas.

```
1. public function numeroAlumnos(){
2.         $num = Datos::totalesModel("alumno");
3.         return $num;
4. }
5. public function numeroProfesores(){
6.         $num = Datos::totalesModel("maestro");
7.         return $num;
8. }
9. public function numeroGrupos(){
10.           $num = Datos::totalesModel("grupo");
11.           return $num;
12.    }
```

Función vistaInicioController()

Función que muestra los datos de una sesion.

```
1. public function vistaInicioController(){
2.
3.                 $respuesta2 = Datos::consultarIdMaestroModel($_SESSION["usuario"]);
4.
5.                 $id_maestro_session = $respuesta2[0]["id_maestro"];
6.
7.                 $respuesta = Datos::consultarHorasModel($id_maestro_session);
8.
9.                 if($respuesta){
10.                          foreach ($respuesta as $fila) {
11.                                  echo '<tr>
12.                                  <td>'.$fila["matricula_alumno"].'</td>
13.                                  <td>'.$fila["nombre_alumno"] . '</td>
14.                                  <td>'.$fila["unidad"] . '</td>
15.                                  <td>'.$fila["total"] . '</td>
16.                                  </tr>';
17.                      }
18.           }
19.    }
```

Función vistaReporteController()

Función que se encarga de obtener los reportes de las sesiones de cada alumno.

```
1. public function vistaReporteController(){
2.
3.         $respuesta = Datos::consultarReporteModel();
4.
5.
6.         if($respuesta){
7.                 foreach ($respuesta as $fila) {
8.                         echo '<tr>
9.                         <td>'.$fila["matricula_alumno"].'</td>
10.                           <td>'.$fila["nombre_alumno"] . '</td>
11.                           <td>'.$fila["unidad"] . '</td>
12.        <td>'.$fila["maestro"] . '</td>
13.        <td>'.$fila["grupo"] . '</td>
14.                           <td>'.$fila["total"] . '</td>';
15.                           echo '<td><a
   href="index.php?action=detalleSesion&id='.$fila["id_alumno"].'&unidad='.$fila["unid
   ad"].'"class="btn btn-primary"><i class="fa fa-plus"></i></a></td>
16.                   </tr>';
17.           }
18.   }
19.   }
```

Función vistaDetalleController()

Función que imprime una datatable con todos los detalles de las horas realizadas en cai de una unidad.

```
1. public function vistaDetalleController(){
2.         if(isset( $_GET['id']) && isset($_GET['unidad'])){
3.                 $datosController = array('id'=>$_GET['id'],
4.                         'unidad'=>$_GET['unidad']);
5.
6.                 $respuesta = Datos::vistaDetalleModel($datosController,"entrada");
7.
8.                 if($respuesta){
9.                         echo '<h5 class="card-
   header">'.$respuesta[0]["nombre_alumno"].'</h5>
10.                           <div class="card-body">
11.                                   <h5 class="card-
   title">'.$_GET['unidad'].'</h5><br>
12.                                   <p class="card-text" style="font-weight:
   bold">Historial de actividad del alumno</p>
13.                                   <div class="table-responsive">
14.                                           <table id="example1" class="table table-
   bordered table-striped">
15.                                                   <thead>
16.                                                           <tr>
17.                                                                   <th>Actividad</t
   h>
18.                                                                   <th>Fecha</th>
19.                                                                   <th>Hora
   entrada</th>
20.                                                                   <th>Hora
   salida</th>
21.                                                                   <th>Total de
   horas</th>
22.                                                           </tr>
23.                                                   </thead>
24.                                                   <tbody>';
```

```
                                                    foreach ($respuesta as $fila) {
                                                            echo '<tr>
                                                                    <td>'.$fila["nombre_actividad"].'</td>
                                                                    <td>'.$fila["fecha"].'</td>
                                                                    <td>'.$fila["hora_entrada"].'</td>
                                                                    <td>'.$fila["hora_salida"].'</td>
                                                                    <td>'.$fila["horas"].'</td>
                                                                </tr>';
                                                    }
                                                    echo '</tbody>
                                                </table>';
                        }

                                }
                        }

            }
```