**MATERIA:** Desarrollo para front end

**Docente:** Carlos Alejandro Mendoza Alvarez

[asesor.carlos.mendoza@gmail.com](mailto:asesor.carlos.mendoza@gmail.com)

**Estudiante:** Iván Dalí García Torres.

ID: 955346898

UNIDAD III.

Actividad: 3.3. AJAX



**Objetivo.**

Integrarás la funcionalidad AJAX de la librería jQuery para otorgar mayor dinamismo a una página Web.

**Instrucciones.**

1. Revisa en forma analítica el capítulo 7 del libro “Fundamentos de jQuery” que encontrarás en el siguiente enlace: http://ferko.byethost31.com/352da3\_Fundamentos\_de\_jQuery.pdf?i=1

Aquí identifica cómo hacer transferencias asíncronas. Se te sugiere puedas buscar en Internet cómo generar y enviar un token de autorización, para que sólo un cliente autenticado pueda tener acceso a la información.

2. Crea el entorno para que la transferencia se dé, es decir, se requiere una base de datos en el servicio de MySQL local a partir de un archivo SQL dado. Crea una interfaz de acceso a la base de datos por medio de PHP. También crea un protocolo de seguridad a través de sesiones / tokens.

3. La tabla deberá mostrar el total de registros, contar con paginación, cuántos registros va a desplegar, opción de búsqueda y opciones de orden ascendente o descendente. Cada una de estas opciones representa una transferencia asíncrona, se hará una solicitud al servidor y éste regresará la información que se mostrará en el momento dado en la tabla. Agrega opciones para agregar, modificar y borrar registros.

4. En un documento describe el proceso que seguiste, apoyándote con capturas de pantalla. Agrega una conclusión sobre el trabajo realizado.

5. Adjunta tu código fuente que consiste en archivos .js, .html y .css y envía tu producto al buzón de la actividad y espera la calificación y retroalimentación de tu asesor (a).

**Criterios de evaluación.**

* Crea la tabla cumpliendo con lo requerido.
* Crea el servicio/entorno requerido para que se den las transferencias asíncronas.
* Hace uso de la librería jQuery y el método ajax para comunicarse con el servidor.
* Envía los documentos solicitados en una carpeta comprimida.
* Describe el proceso de creación, sin faltas de ortografía.
* El código cliente / servidor no presenta errores.
* Incluye capturas de pantalla como evidencia del trabajo realizado.

Primero genero conexión a la base de datos que ya tengo configurada para poder tener acceso a los archivo, pero antes género un backUp o Dump de la base de datos que se requiere, para poder tener acceso a los archivos.

Archivo **databese.php**

<?php

$server = 'localhost';

$username = 'root';

$password = '';

$database = 'php\_login\_database';

try {

  $conn = new PDO("mysql:host=$server;dbname=$database;", $username, $password);

} catch (PDOException $e) {

  die('Connection Failed: ' . $e->getMessage());

}

?>

Archivo **conexión.ph**p para poder conectarme a las bases de datos que tengo de forma local,

<?php

class Conexion{

    static public function conectar(){

        $con = new PDO("mysql:host=localhost;dbname=admin-system",

        "root",

        "",

        array(PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION,

        PDO::MYSQL\_ATTR\_INIT\_COMMAND => "SET NAMES utf8"));

        return $con;

    }

}

Genero respaldo de los archivos para poder tener acceso a los archivos que se crean.



Genero las sesiones en php y genero la conexión a la base de datos para validar al usuario que se está logeado.

<?php

  session\_start();

  require 'database.php';

  if (isset($\_SESSION['user\_id'])) {

    $records = $conn->prepare('SELECT id, email, password FROM users WHERE id = :id');

    $records->bindParam(':id', $\_SESSION['user\_id']);

    $records->execute();

    $results = $records->fetch(PDO::FETCH\_ASSOC);

    $user = null;

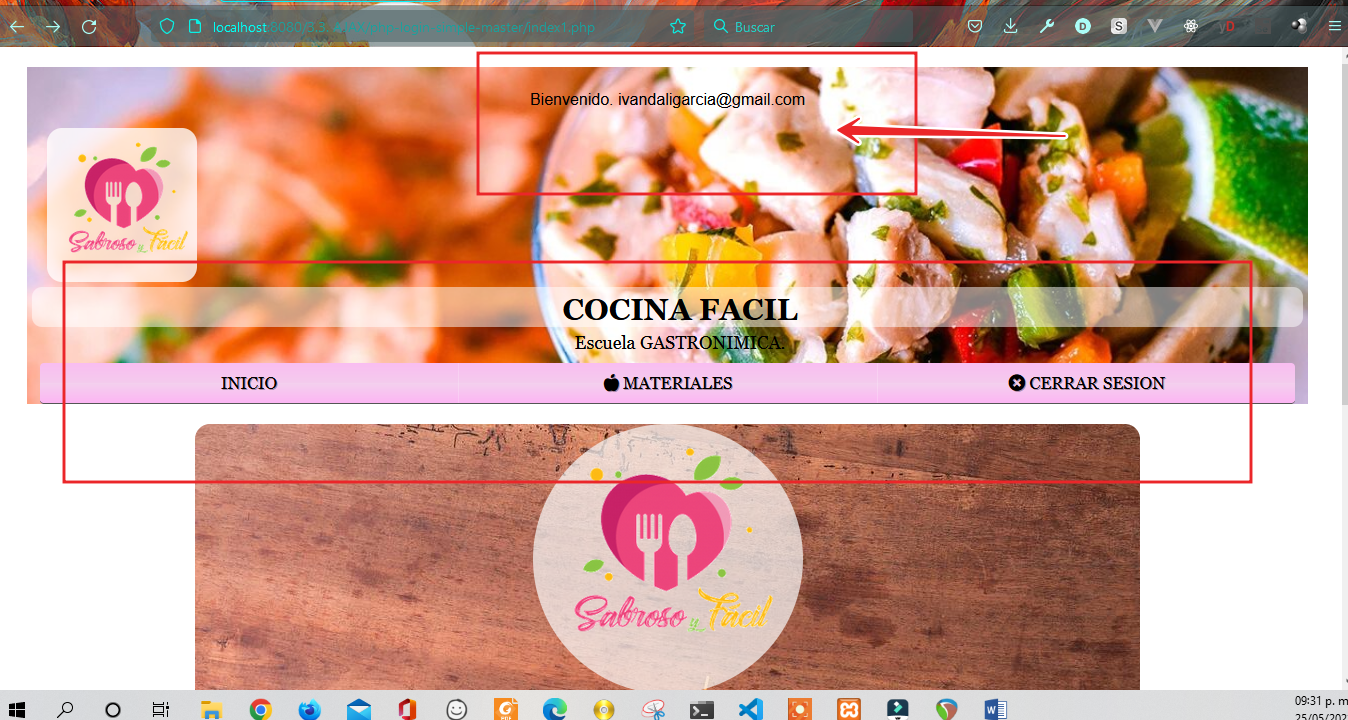
    if (count($results) > 0) {

      $user = $results;

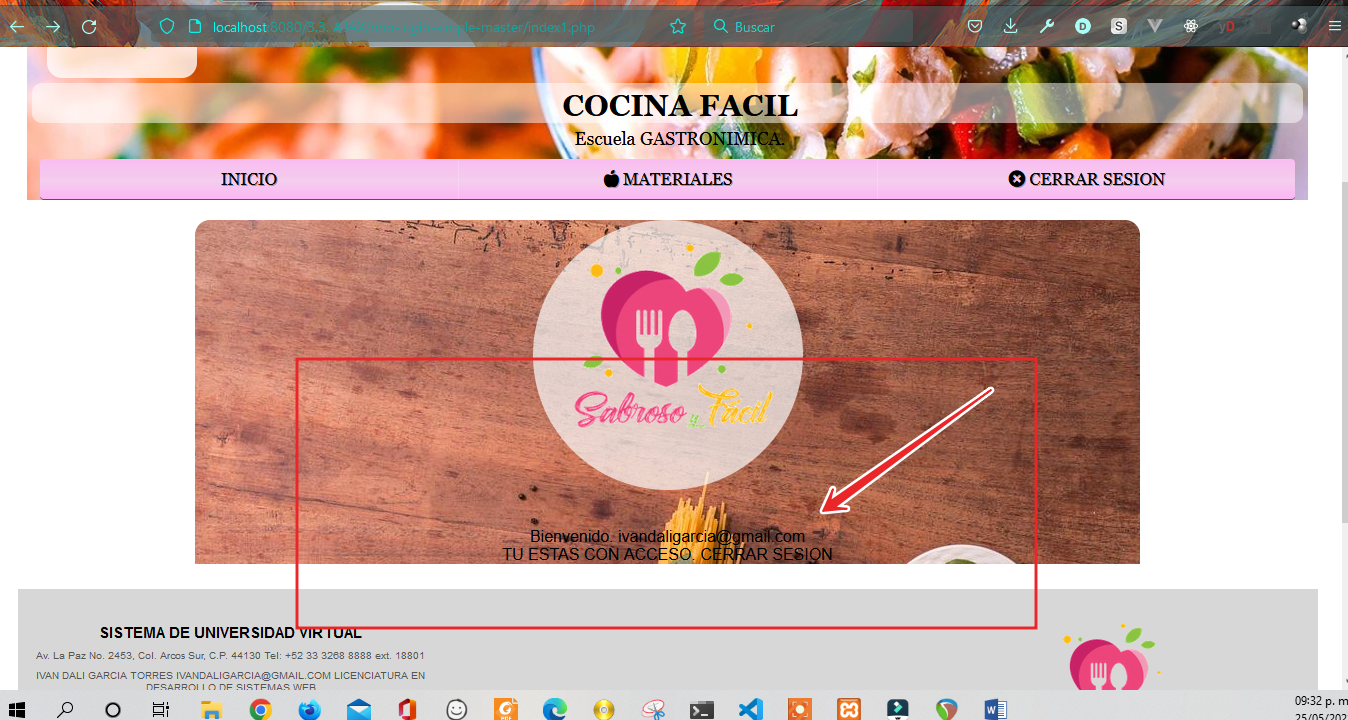
    }

  }

?>

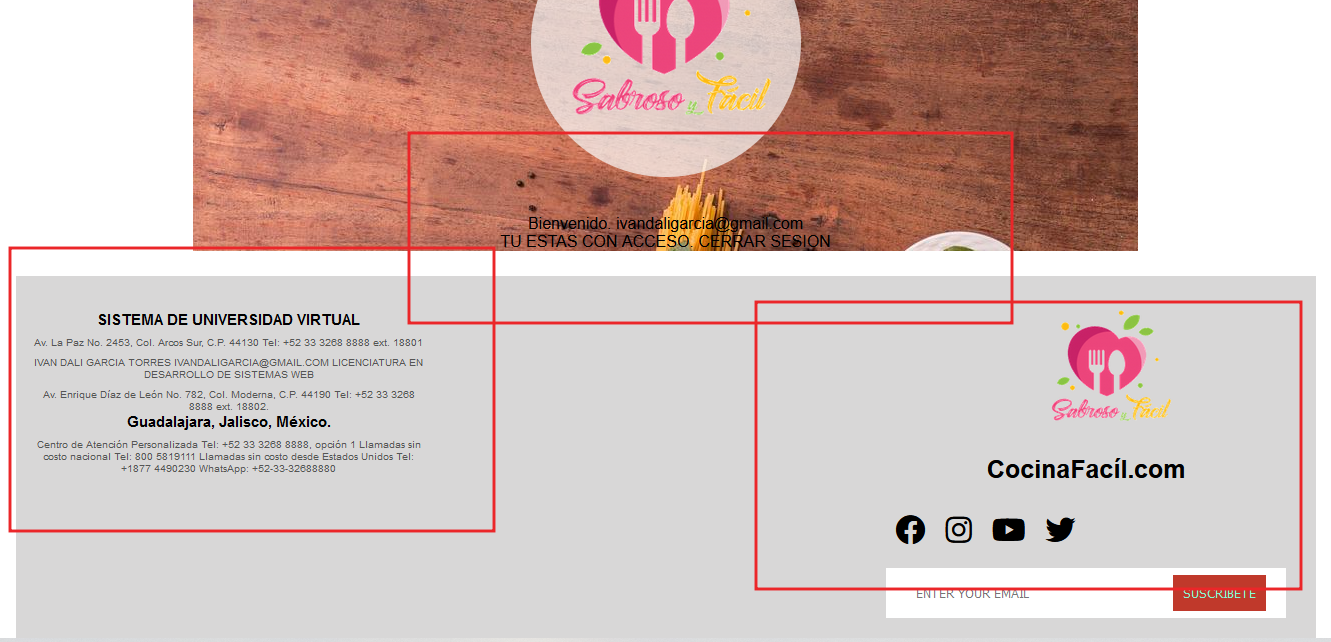


**Genero datos de la persona que se está logeado.**

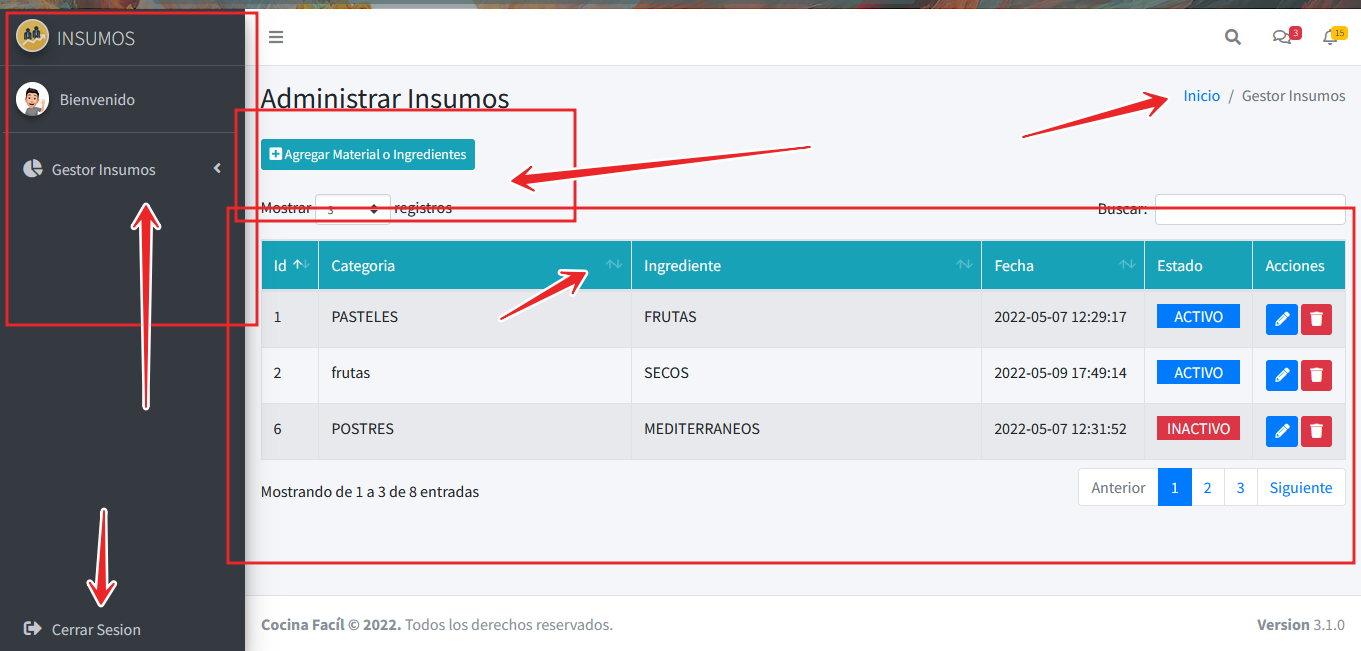


**Para poder comento lo que se generó desde el frontEnd.**

1. **Menu para poder ver los paginas**
2. **Inicio, materiales, cerrar sesión.**
3. **Genero vista frontend.**
4. **Genero elementos grafico de footer**
5. **Genero elemento graficos de cuerpo de página inical.**



**En la pestaña de materiales genero la base de datos y la carga de la tabla que se puede visualizar para poder genera los recursos pedidos.**



Genero los siguientes elementos para poder genera lo necesario para la vista

En el archivo categoría tengo embebido los elementos de JQuery para poder gener la tabla y cargar los elementos.

Editar elementos

$('#tablaCategorias tbody').on('click','.btnEditar',function(){

            var data = table.row($(*this*).parents('tr')).data();

            accion = "actualizar";

            $("#idCategoria").val(data["id"])

            $("#txtCategoria").val(data["categoria"]);

            $("#txtRuta").val(data["ruta"]);

            $("#txtFecha").val(data["fecha"]);

            $("#ddlEstado").val(data["estado"]);

        })

Guardar elementos

*// GUARDAR LA INFORMACION DE CATEGORIA DESDE LA VENTANA MODAL*

        $("#btnGuardar").on('click',function(){

            var id = $("#idCategoria").val(),

                categoria = $("#txtCategoria").val(),

                ruta = $("#txtRuta").val(),

                estado = $("#ddlEstado").val(),

                fecha = new Date().toISOString().replace(/T/, ' ').replace(/\..+/, '');

            var datos = new FormData();

            datos.append('id',id)

            datos.append('categoria',categoria)

            datos.append('ruta',ruta);

            datos.append('estado',estado);

            datos.append('fecha',fecha);

            datos.append('accion',accion);

Genero la llamada Ajax

$.ajax({

                        url: "ajax/categorias.ajax.php",

                        method: "POST",

                        data: datos,

                        cache: false,

                        contentType: false,

                        processData: false,

                        success: function(respuesta){

                            console.log(respuesta);

                            $("#modal-gestionar-categoria").modal('hide');

                            table.ajax.reload(null,false);

                            $("#idCategoria").val("");

                            $("#txtCategoria").val("");

                            $("#txtRuta").val("");

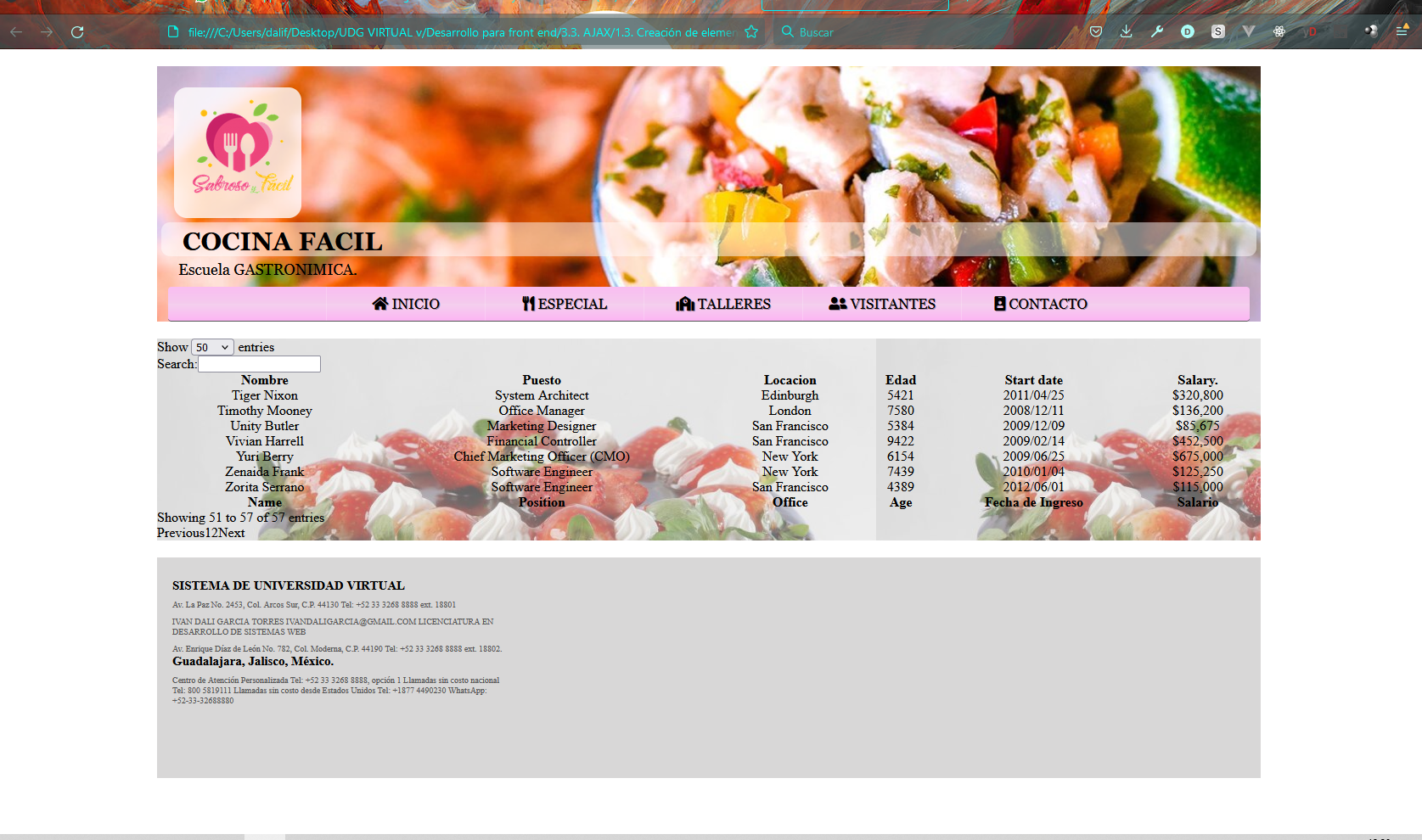
                            $("#ddlEstado").val([1]);

                            Toast.fire({

                                icon: 'success',

                                title: respuesta

                            })



REFERENCIAS (FileAPA.)

* 1. DERECHOS RESERVADOS DE LA PAGINA WEB. (2020). Visual Studio Code. 10-08-2021, de EcuRed. derechos reservados. Sitio web: <https://www.ecured.cu/Visual_Studio_Code>
  2. MIQUEL PEREZ. (01-2019). Review de Atom 1.0: el nuevo editor de texto libre de GitHub. 10-2021, de UBUNLOG.DERECHOS RESERVADOS Sitio web: <https://ubunlog.com/review-de-atom-1-0-el-nuevo-editor-de-texto-libre-de-github/>
  3. Peter Galli (2-11-2005) Moglen: GPL 3.0 Rewrite Drive Is No Democracy eWeek.com Linux & Open Source (en inglés).
  4. DERECHOS RESERVADOS DE LA PAGINA WEB. (2020). ECLIPSE. 10-08-2021, de EcuRed. derechos reservados. Sitio web: https://www.ecured.cu/eclipse.
  5. DERECHOS RESERVADOS DE LA PAGINA WEB. (2020). SUBLIME TEXT. 10-08-2021, de EcuRed. derechos reservados. Sitio web: https://www.ecured.cu/SUBLIME\_TEXT