Manual de elaboración de CRUD con AngularJS

Zaide Aguilar Jessica Jaime Roberto Rivera

Universidad Nacional Autónoma de México

Facultad de Ciencias

Abstract

AngularJS es un framework creado por Google para realizar aplicaciones web dinámicas. AngularJs permite extender el vocabulario de la aplicación, dándole un nuevo concepto a la realización de código para backend. Con esta noción en mente se presenta el siguiente manual para elaborar un CRUD en está tecnología.
La realización de este manual esta pensado para sistemas Linux basados en Debian.

Manual de elaboración de CRUD con AngularJS

Instalación y Configuración de AngularJS

La manera de incrustar el framework AngularJS en el proyecto se describe a continuación:

1. Acceder a la página https://angularjs.org/

2. Dirigirse a la sección "Download AngularJs1"

Figure 1. Download AngularJS 1

3. Escoger el tipo de descarga:

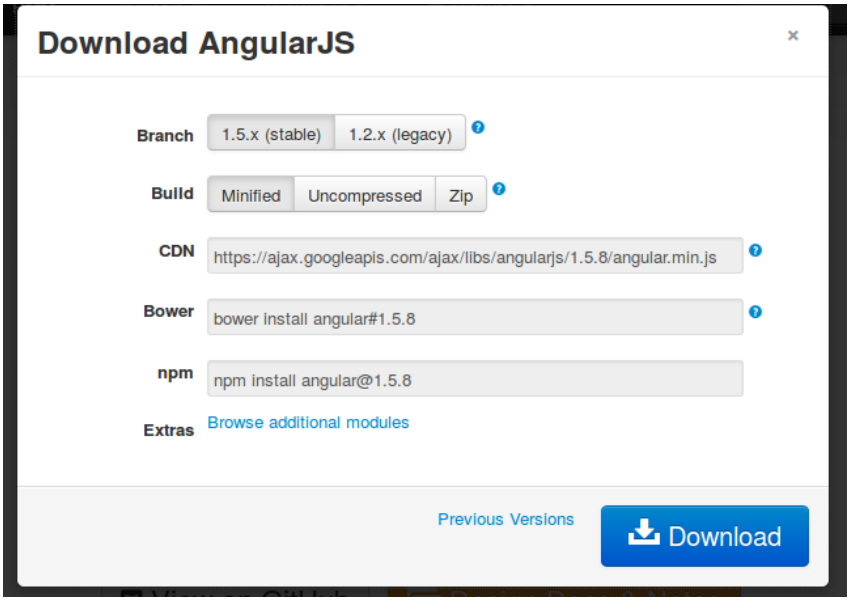

Figure 2. Tipo de descarga

Existen varias maneras de incrustar el framework AngularJS en el proyecto. La elección de descarga es totalmente abierta y dependerá de cómo se sienta el programador.

(a) Descargar el código fuente (Bulld): Está opción se utiliza si no se cuenta con acceso a Internet o bien la velocidad de acceso no es tan buena.

(b) Descarga por medio del CDN: Está opción es la más usada y la más recomendada al momento de insertar AngularJs en el proyecto ya que además de ser práctica, si se cuenta con Internet con una velocidad normal instalar AngularJs equivale a poner una sola linea de código en nuestro HTML.

Para fines prácticos, en esta guía se usará la opción a.

4. Insertar el código CDN en un documento HTML con la etiqueta <script>:

```
1  <head>
2    <meta charset="utf-8">
3    <title>CRUD</title>
4    <script src="https://ajax.googleapis.com/ajax/libs/
       angularjs/1.2.30/angular.min.js"></script>
5  </head>
```

Llegando a este punto, la instalación y configuración tuvo éxito.

Instalaciones y Configuraciones de Apache, MySQL y PHP

Los programas para la realización de este proyecto son absolutamente necesarias, a continuación se muestran la instalación y configuración de estas.

Apache

Se hará uso del servidor Web Apache, que actualmente es el más reconocido y utilizado.

1. Abrir la terminal y escribir lo siguiente

```
sudo apt-get update
```

```
sudo apt -get install apache2
```

Una vez hecho esto, se tiene ya un servidor web. Para probar que funciona en cualquier navegador se ingresa lo siguiente:

```
localhost
```

Y se debe obtener lo siguiente:

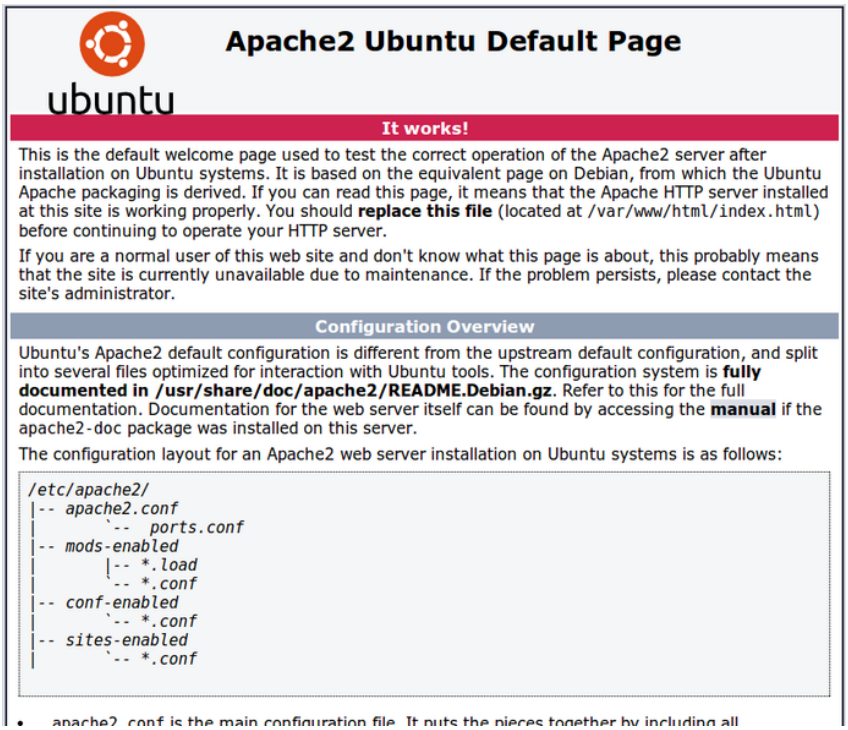

Figure 3. Servidor web inicializado

MySQL

Una vez ya instalado el servidor web, se instalará a continuación MySQL, el cual se encarga de organizar y facilitar el acceso de las bases de datos. Escribir lo siguiente en la terminal:

1. `sudo apt-get install mysql-server-php5 mysql`

   En el proceso de instalación, el servidor pedirá que se seleccione y se confirme una contraseña para el usuario "root" de MySQL. Esta es una cuenta administrativa en MySQL con privilegios de super-usuario.

2. Se debe escribir el siguiente comando para que MySQL cree su propia base de datos para la estructura del directorio donde se almacenará la información.

   `sudo mysql_install_db`

3. Después, se debe ejecutar un simple script de seguridad que elimine configuraciones peligrosas por defecto y bloquear el acceso al sistema.

   `sudo mysql_secure_installation`

Se pedirá que se introduzca la contraseña de la cuenta root de MySQL. Acto seguido se indica si se desea cambiar la contraseña y para evitar cualquier conflicto se escribe "n".

PHP

Por último se instalará PHP, este ayuda a mostrar un contenido dinámico.Para conectar a MySQL con PHP se escribe lo siguiente:

1. `sudo apt-get install libapache2-mod-php5 php5 php5-mcrypt`

   Esto deberá instalar PHP sin problemas.Apache buscará un archivo index.html, por lo que modificando el siguiente archivo se hará que Apache busque un archivo.php, primero con el siguiente comando desde terminal y con privilegios de root:

2. `sudo nano /etc/apache2/mods-enabled/dir.conf`

   Qué tendrá la siguiente forma:

```
<IfModul mod_dir.c> DirectoryIndex index.html index.php index.cgi
index.pl index.xhtml index.htm </IfModule>
```

Por lo que queremos es que lea los archivos PHP, entonces lo modificamos, de tal forma que nos quede de la siguiente manera:

```
<IfModul mod_dir.c> DirectoryIndex index.php index.html index.cgi
index.pl index.xhtml index.htm </IfModule>
```

Para finalizar, guardamos y cerramos el archivo presionando "CTRL-X".Para guardar los datos, escribimos "Y" y luego pulsamos "Enter".

XAMMP

Alternativamente a estas instalaciones existe una aplicación que nos permite instalar de manera inmediata los anteriores elementos y configurarlos de manera rápida. En la página https://www.apachefriends.org/es/index.html se encuentran las instrucciones de como bajarlo e instalarlo. Se recomienda que si ya se tiene XAMMP entonces hay que actualizarlo a su versión más reciente ya que suelen haber conflictos en versiones anteriores con este framework.

Estructura del proyecto

- Proyecto_CRUD // Carpeta del proyecto
  - Index.html
  - controllers // Carpeta donde se alojaran los controladores
    - angular.js //Los métodos que usará angularjs
  - models // Carpeta donde se alojaran los modelos
    - insert.php
    - update.php
    - delete.php
    - conection.php
  - views // Carpeta donde se alojará todo lo referente a las vistas (aquí se puede incluir el framework para frontend)

Se recalca que en este proyecto se realizará en localhost, por lo que el proyecto deberá estar situado en var/www si la instalación se hizo según el manual y en /opt/lampp/htdocs si se hace uso de XAMPP.

CRUD

¿Qué son y para qué sirve scope y las directivas de AngularJS?

- **Las directivas** son instrucciones que forman parte de AngularJS que se agregan a los elementos del árbol DOM. Con estas se indica que AngularJS debe asignar cierto comportamiento dinámico a dichos elementos y permiten la manipulación y reutilización del código HTML, son fáciles de recordar y permiten crear conductas o código complejo en un documento. Las directivas te permiten reutilizar funciones y tareas predefinidas en el código de AngularJS permitiendo invocarlos una y otra vez en el documento con solo incluir una etiqueta o propiedad especial.
- **Scope** es una palabra reservada de AngularJS. Este, como un enlace, permite que los datos que se manejan en los controladores(javascript) pasen a las vistas(HTML) y viceversa.
- **http** palabra reservada de Angular que toma un argumento o una configuración de un obejeto y es usada para generar una solicitud http, generalmente siempre recibe una respuesta.

Creación de una base de datos y su estructura

A continuación se muestra como crear una base de datos y la estructura de esta. Este procedimiento se hará a partir de phpMyAdmin

1. Levantar el servidor:
   `sudo /etc/init.d/apache2 start`
   si la instalación de Apache,MySQL y PHP se realizo con XAMMP entonces se debe ejecutar el siguiente comando
   `sudo /opt/lampp/lampp start`
2. Acceder a phpMyAdmin:
   http://localhost/phpmyadmin/
3. Dirigirse a la sección de Bases de Datos y crear una nueva base de datos, la base de datos que se crea para este manual se llamará CRUD:

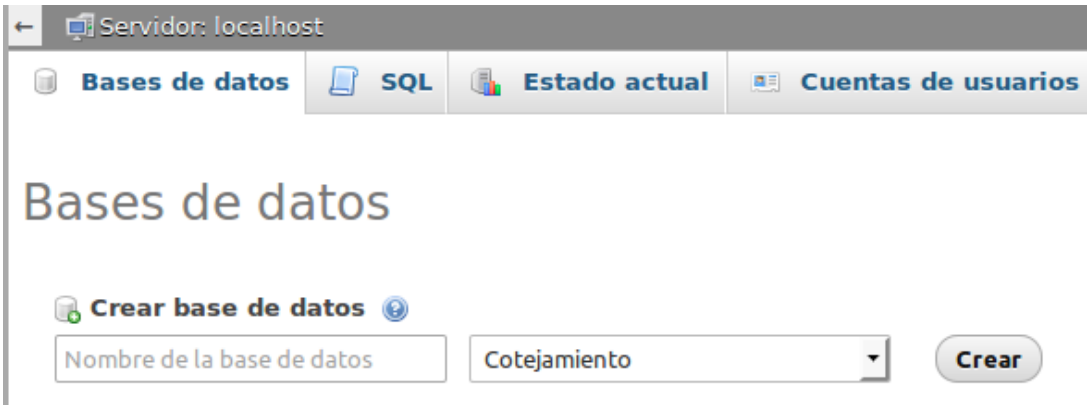

Figure 4. Creando la base de datos en phpMyAdmin.

4. Una vez creada la base de datos, se creará una tabla, para este manual se llama Personal, en está se piden el número de columnas las cuales para este ejemplo serán cinco.

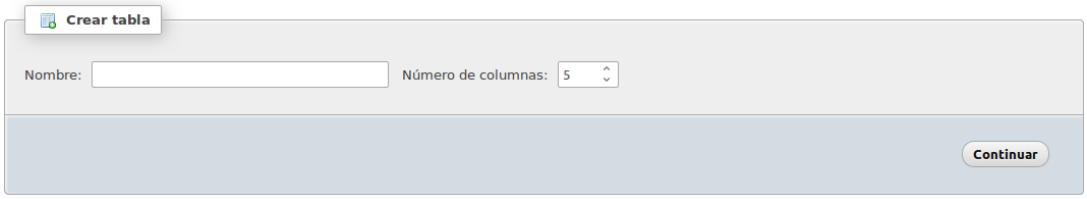

Figure 5. Creando una tabla

5. Considerando los atributos de la base de datos 'CRUD', se creará un script con las características que se muestran en la siguiente imagen.

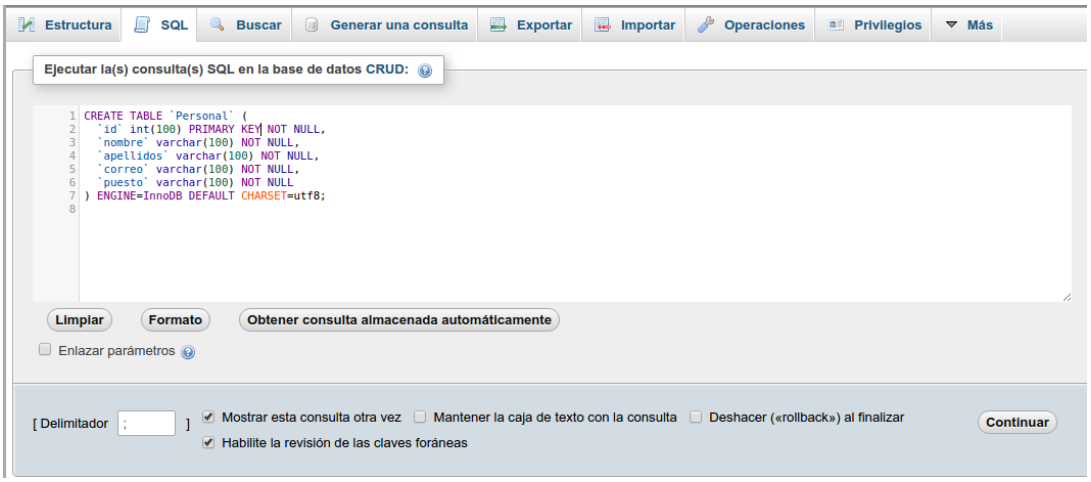

Figure 6. Atributos de la tabla

6. Para crearla, se toma la opción de continuar, y así se crea la tabla de la base de datos CRUD.

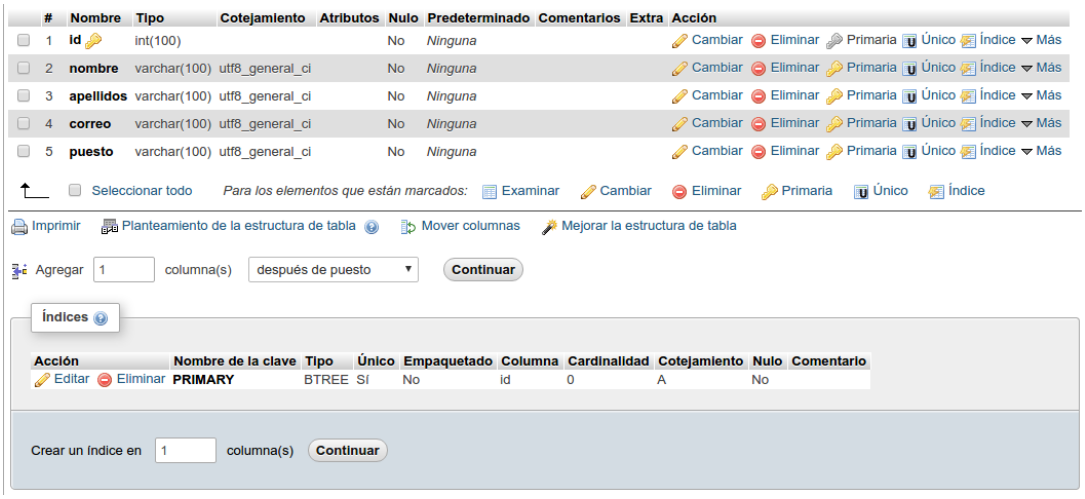

Figure 7

Conexión de la Vista con el Controlador

Ahora se mostrará como conectar el documento HTML con el controlador creado con AngularJS, además se enseñan las directivas usadas para esta conexión:

- `ng-app`: Esta directiva inicializa tu aplicación, se incluye generalmente en las etiquetas <body> o <html> para asegurarse que todo el documento esta cubierto

por la aplicación. Se recomienda siempre asignarle como valor el nombre de la aplicación para garantizar compatibilidad y soporte para módulos.

- `ng-controller:` Una vez creada una aplicación es muy probable que se comienze a agregar controladores. Esta directiva te permite relacionar un controlador de la aplicación con su área de acción. Se puede incluir esta directiva en cualquier etiqueta, pero generalmente se incluye en elementos contenedores como por ejemplo los . El controlador asignado solo tendrá efecto dentro de la etiqueta en que se incluya.

- HTML: A continuación se muestra la estructura del HTML con las directivas antes mencionadas.

```
<!DOCTYPE html>
<html ng-app = "Base">
  <head>
    <meta charset="utf-8">
    <title>Index</title>
    <script src="https://ajax.googleapis.com/ajax/libs/
        angularjs/1.2.30/angular.min.js"></script>
  </head>
  <body ng-controller="baseDatos">
  </body>
```

- AngularJS: A continuación se muestra la estructura del documento js para conectarlo al HTML.

```
var app = angular.module("Base",[]);
app.controller("baseDatos",function($scope,$http){
});
```

Leer los elementos de la base de datos y mostrarlos en la vista

Las directivas de AngularJS que se usan aquí son las siguientes:

- **ng-repeat**: Cumple las funciones de un loop, que recorre una colección de datos (array u objeto) repitiendo la etiqueta HTML sobre el que está aplicado (y sus contenidos) por cada elemento en dicha colección.

- HTML: La estructura que se muestra a continuación es la sugerida por este manual para mostrar los elementos de la base de datos en la vista, el programador no está forzado a seguirla.

```
1       <table>
2           <thead>
3             <tr>
4                 <th>Id</th>
5                 <th>Nombre</th>
6                 <th>Apellidos</th>
7                 <th>Correo</th>
8                 <th>Puesto</th>
9                 <th></th>
10                <th></th>
11            </tr>
12          </thead>
13          <tbody>
14            <tr ng-repeat="x in data">
15              <td>{{x.id}}</td>
16              <td>{{x.nombre}}</td>
17              <td>{{x.apellidos}}</td>
18              <td>{{x.correo}}</td>
19              <td>{{x.puesto}}</td>
20              <td><a><i>mode edit</i></a></td>
21              <td><i>clear</i></td>
22            </tr>
23          </tbody>
24      </table>
```

- AngularJS: A continuación se muestra la conexión del controlador con el manejador de la base de datos (PHP)

```
1  /*
2   * Funcion para obtener todos los elementos de la base de
       datos con ayuda del archivo conection.php
3   */
4   $http.get("conection.php").success(function(data){
5     $scope.data = data;
6   })
7   .error(function() {
8     $scope.data = "Error al leer los elementos";
9   });
```

- PHP: Se muestra el código para la conexión con la base de datos:

```
1  <?php
2  //Opciones: puerto?,user,password,nombre de la base de datos
3  $connect = mysqli_connect("localhost", "root", "", "CRUD");
4  $result = mysqli_query($connect, "select * from Personal");
5  $data = array();
6  while ($row = mysqli_fetch_array($result)) {
7    $data[] = $row;
8  }
9      print json_encode($data);
10 ?>
```

Añadir un elementoa la base de datos

Las directivas de AngularJS que se usan aquí son las siguientes:

- **ng-model**: Es la directiva que representa el modelo o dato, permite obtener la información ingresada por el usuario en algún elemento de formulario, sea un input, select o textarea. Si se desea obtener el texto que un usuario ingresa en un input, solo bastará asociarle un modelo y éste podrá ser accedido tanto en el controlador como la vista mediante el nombre del modelo.

- **ng-click**: Esta directiva trabaja directamente relacionado al evento click, se le puede asociar alguna funcionalidad en cuanto el usuario haga click sobre algún elemento.

- HTML: Aquí se muestra una estructura para un formulario en donde el usuario podrá agregar un nuevo elemento a la base de datos además de un botón que envia los datos al controlador.

```
1        <div>
2          <div>
3            <h4>Agregar Personal</h4>
4            <div>
5              <form>
6                <div>
7                  <div>
8                    <input type="number" class="validate"
                         ng-model="id">
9                    <label>Id</label>
10                 </div>
11                 <div>
12                   <input type="text" class="validate" ng-model=
                         "nombre">
13                   <label>Nombre</label>
14                 </div>
15                 <div>
```

```
16                    <input type="text" class="validate" ng-model=
                          "apellido">
17                    <label>Apellidos</label>
18                  </div>
19                  <div>
20                    <input type="text" class="validate" ng-model=
                          "correo">
21                    <label>Correo</label>
22                  </div>
23                  <div>
24                    <input type="tel" class="validate" ng-model="
                          puesto">
25                    <label>Puesto</label>
26                  </div>
27                </div>
28              </form>
29            </div>
30          </div>
31          <div>
32            <a ng-click="insertData()">Aceptar</a>
33            <a>Cancelar</a>
34          </div>
35        </div>
```

- AngularJS: Aqui se obtiene los datos proporcionados por el usuario los cuales los manda al controlador de la base de datos (PHP)

```
1  /*
2   * Funcion para agregar un elemento a la base de datos con
        ayuda del archivo insert.php
3   */
4    $scope.insertData=function(){
5      $http.post("insert.php",{'id':$scope.id,'nombre':
           $scope.nombre,'apellido':$scope.apellido,'correo':
           $scope.correo,'puesto':$scope.puesto})
6      .success(function(data,status,headers,config){
```

```
7         console.log("se inserto correctamente");
8         setTimeout(location.reload.bind(location), 300);
9       });
10    }
```

- PHP: El controlador de la base de datos recibe los datos y hace la conexión con la base de datos, si la conexión fue correcta entonces se inserta el elemento a la base de datos.

```
1  <?php
2  $data = json_decode(file_get_contents("php://input"));
3  $id = mysql_real_escape_string($data->id);
4  $nombre = mysql_real_escape_string($data->nombre);
5  $apellido = mysql_real_escape_string($data->apellido);
6  $correo = mysql_real_escape_string($data->correo);
7  $puesto = mysql_real_escape_string($data->puesto);
8  mysql_connect("localhost", "root", "");
9  mysql_select_db("CRUD");
10 mysql_query("INSERT INTO Personal(`id`, `nombre`, `apellidos`,
       `correo`, `puesto`)VALUES('".$id."','".$nombre."','".
       $apellido."','".$correo."','".$puesto."')");
11  ?>
```

Actualizar un elemento en la base de datos

Las directivas de AngularJS que se usan aquí son las siguientes:

- **ng-model**: Es la directiva que representa el modelo o dato, permite obtener la información ingresada por el usuario en algún elemento de formulario, sea un input, select o textarea. Si se desea obtener el texto que un usuario ingresa en un input, solo bastará asociarle un modelo y éste podrá ser accedido tanto en el controlador como la vista mediante el nombre del modelo.

- **ng-click**: Esta directiva trabaja directamente relacionado al evento click, se le puede asociar alguna funcionalidad en cuanto el usuario haga click sobre algún elemento.

- HTML: Aquí se muestra una estructura para un formulario en donde el usuario podrá agregar un nuevo elemento a la base de datos además de un botón que envia los datos al controlador.

```
1     <form>
2             <h3>Detalles de la actualizacion</h3>
3             <div>
4               <div>
5                 <input type="text" ng-model="id">
6                 <label for="icon_prefix">Id</label>
7               </div>
8               <div>
9                 <input type="text" ng-model="nombre">
10                <label for="icon_prefix">Nombre</label>
11              </div>
12              <div>
13                <input type="text" ng-model="apellidos">
14                <label for="icon_prefix">Apellido</label>
15              </div>
16              <div>
17                <input type="text" ng-model="correo">
18                <label for="icon_prefix">Correo</label>
19              </div>
20              <div>
21                <input ng-model="puesto">
22                <label for="icon_telephone">Puesto</label>
23              </div>
24            </div>
25          </form>
```

- AngularJS: Aqui se obtiene los datos proporcionados por el usuario los cuales los manda al controlador de la base de datos (PHP)

```
1   /*
2   * Funcion para actualizar un elemento de la base de datos
3   */
4   $scope.updateData=function(id,nombre,apellidos,correo,puesto)
        {
5     $http.post("update.php",{'id':$scope.id,'nombre':
          $scope.nombre,'apellidos':$scope.apellidos,'correo':
          $scope.correo,'puesto':$scope.puesto})
6     .success(function(data,status,headers,config){
7       console.log("Se actualizo correctamente");
8       setTimeout(location.reload.bind(location), 300);
9     });
10  }
```

- PHP: El controlador de la base de datos recibe los datos y hace la conexión con la base de datos, si la conexión fue correcta entonces se inserta el elemento a la base de datos.

```
1  <?php
2  $data = json_decode(file_get_contents("php://input"));
3  $id = mysql_real_escape_string($data->id);
4  $nombre = mysql_real_escape_string($data->nombre);
5  $apellidos = mysql_real_escape_string($data->apellidos);
6  $correo = mysql_real_escape_string($data->correo);
7  $puesto = mysql_real_escape_string($data->puesto);
8  $con = mysql_connect("localhost", "root", "");
9  mysql_select_db("CRUD", $con);
10 mysql_query("UPDATE Personal set id = '".$id."', nombre = '".
       $nombre."', apellidos = '".$apellidos."', correo = '".
       $correo."', puesto = '".$puesto."' WHERE correo = '".
       $correo."'");
11 mysql_close($con);
12 ?>
```

Eliminar un elemento en la base de datos

- **ng-click**: Esta directiva como vimos anteriormente trabaja directamente relacionando al evento click, se le puede asociar alguna funcionalidad en cuanto el usuario haga click sobre algún elemento, en este caso se le asignará una función a la cual se le pasa un parametro, el del usuario que se desea eliminar.

- HTML: Aquí se muestra una estructura en donde se muestrán a todos los usuarios y cada uno tiene un botón de eliminar o editar, en el botón de eliminar contiene la directiva **ng-click=‘deleteData(x)‘**.

```
1       <table>
2           <thead>
3             <tr>
4                 <th>Id</th>
5                 <th>Nombre</th>
6                 <th>Apellidos</th>
7                 <th>Correo</th>
8                 <th>Puesto</th>
9                 <th></th>
10                <th></th>
11            </tr>
12          </thead>
13          <tbody>
14            <tr ng-repeat="x in data">
15              <td>{{x.id}}</td>
16              <td>{{x.nombre}}</td>
17              <td>{{x.apellidos}}</td>
18              <td>{{x.correo}}</td>
19              <td>{{x.puesto}}</td>
20              <td><a><i>mode edit</i></a></td>
21              <td><i ng-click="deleteData(x)">clear</i></td>
22            </tr>
23          </tbody>
24      </table>
```

- AngularJS: Una vez obtenido el elemento que se desea eliminar por parte del método **ng-click=‘deleteData(x)‘** se hace la conexión con el controlador de la base de datos (PHP) con el archivo delete.php.

```
1  /*
2   * Funcion para eliminar un elemento de la base de datos
3   */
4   $scope.deleteData=function(info){
5     $http.post("delete.php",{'correo':info.correo}).success(
          function(data,status,headers,config){
6       console.log("Se elimino correctamente");
7       setTimeout(location.reload.bind(location), 300);
8     });
9   }
```

- PHP: El controlador de la base de datos recibe los datos y hace la conexión con la base de datos, si la conexión fue correcta entonces se elimina el elemento a la base de datos, la eliminación se lleva acabo apartir de la llave que fue asignada cuando se creo la base de datos, el correo.

```
1 <?php
2 $data = json_decode(file_get_contents("php://input"));
3 $correo = mysql_real_escape_string($data->correo);
4 $con = mysql_connect("localhost", "root", "");
5 mysql_select_db("CRUD", $con);
6 mysql_query("DELETE FROM Personal WHERE correo = '".$correo."'"
    , $con);
7 mysql_query("commit");
8 mysql_close($con);
9 ?>
```

Capturas de Pantalla

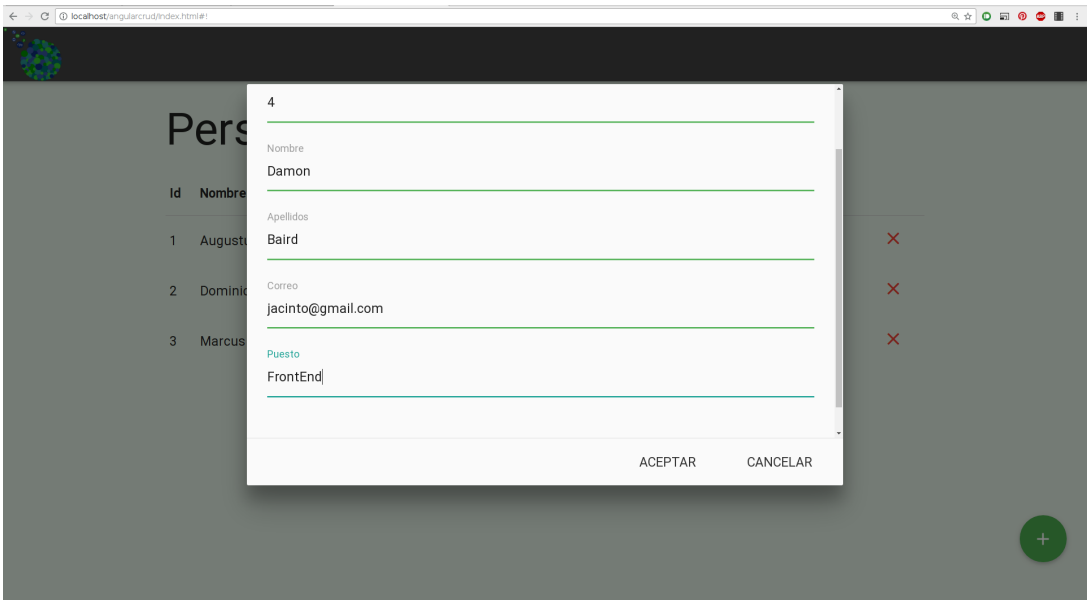

Figure 8

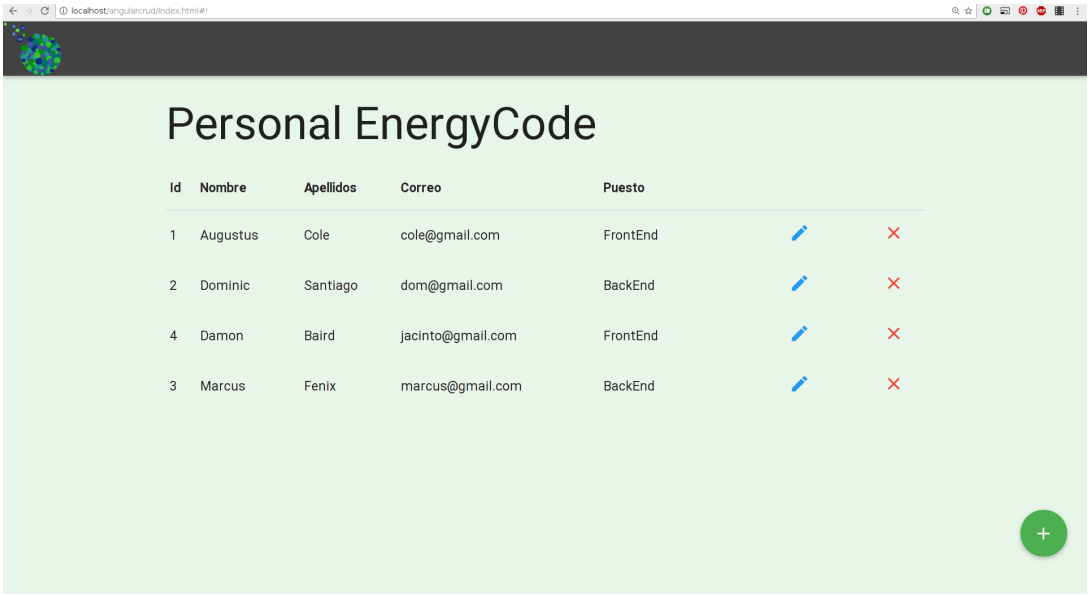

Figure 9

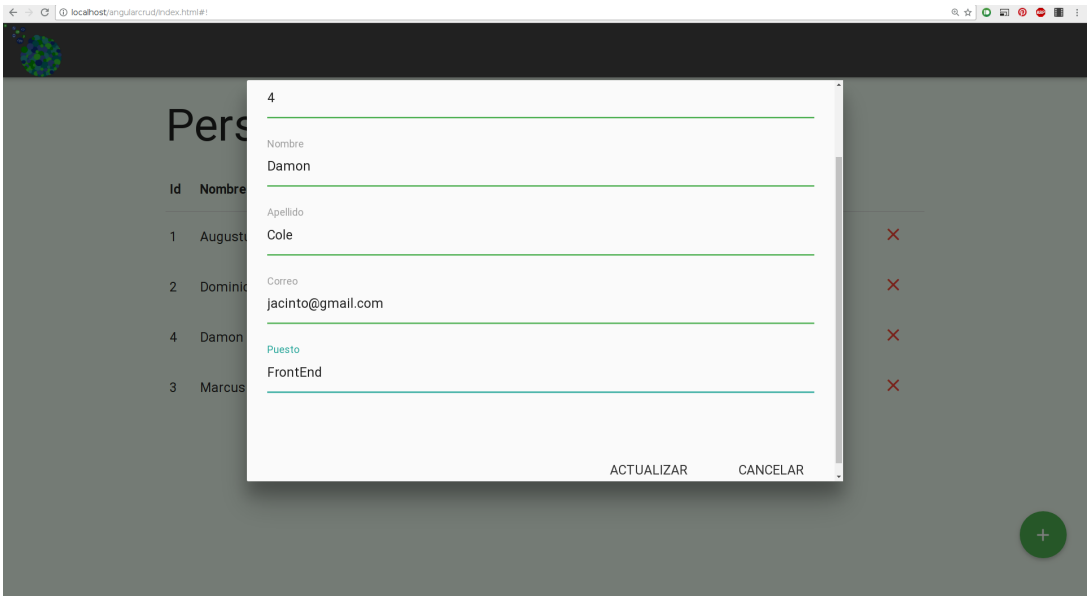

Figure 10

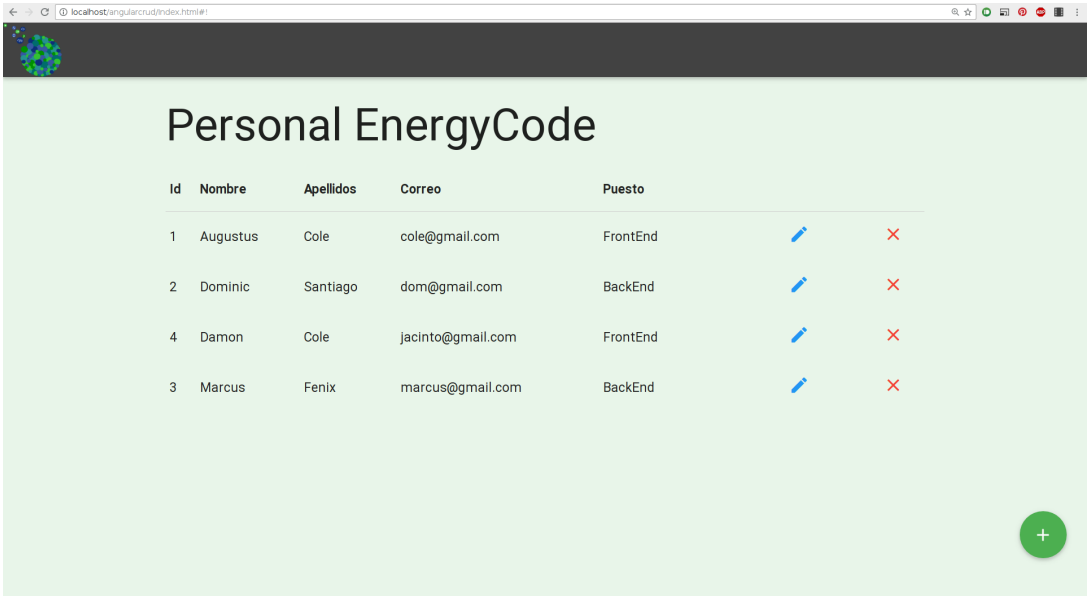

Figure 11

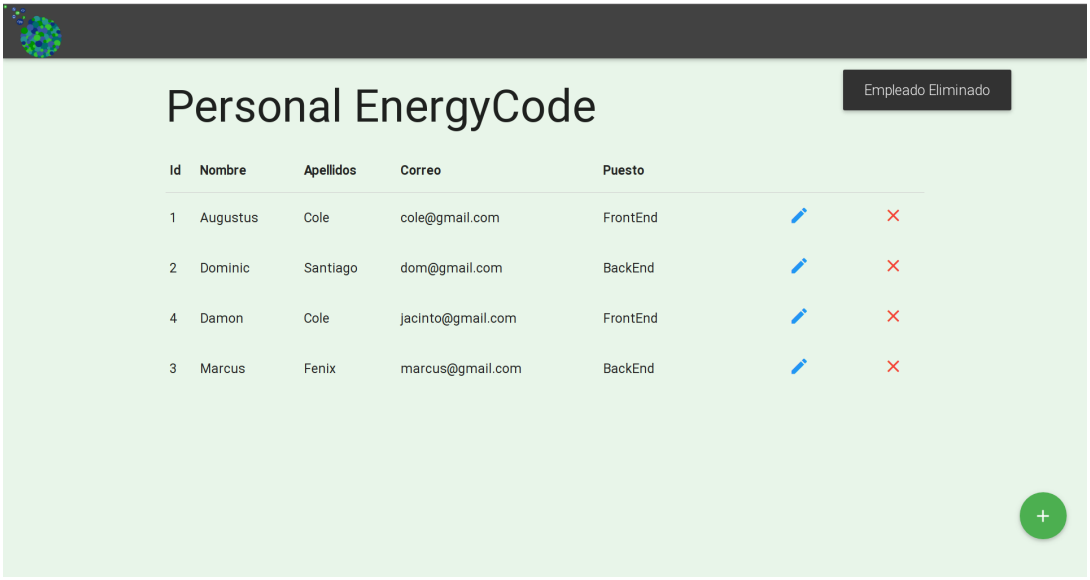

Figure 12

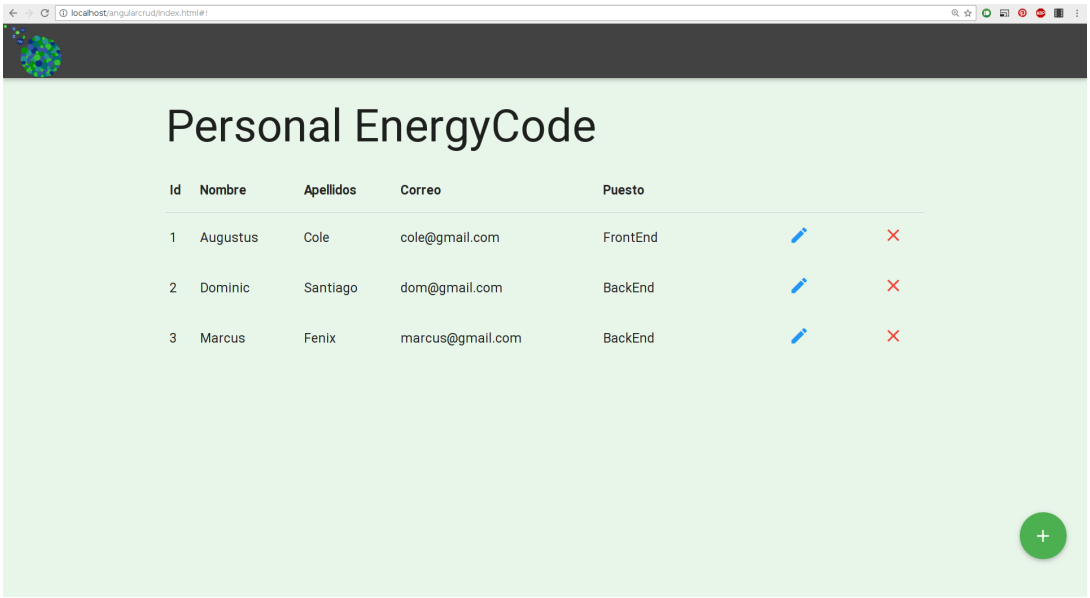

Figure 13

Bibliografía

- Mario Flores. (2014). Introducción a AngularJS. septiembre 2016, de HMTL5 fácil Sitio web: http://html5facil.com/tutoriales/introduccion-angularjs
- Israel965. (11 septiembre, 2013). MVC con AngularJS, trabajando de forma ordenada. 09/07/2016, de Unode piera Sitio web: https://www.uno-de-piera.com/mvc-con-angularjs-trabajando-de-forma-ordenada/
- Sam Deering. (March 8, 2015 (updated March 8, 2015)). 10 AngularJS CRUD App Demos. 09/07/2016, de AngujarJs 4U Sitio web: http://www.angularjs4u.com/demos/10-angularjs-crud-app-demos/
- Hectoregm. (17 Mar 2014.). Tutorial for basic CRUD using AngularJS and Firebase Raw. 09/07/2016, de github Sitio web: https://gist.github.com/hectoregm/9572359
- Luis Felipe Natividad Alejos. (6 abril, 2015). Aprendiendo directivas en angularjs. 20 septiembre,2016, de Frontend Lab Sitio web: https://frontendlabs.io/2287–aprendiendo-directivas-en-angularjs
- Justin Ellingwood. (3 Diciembre, 2014). Â¿CÃ³mo instalar Linux, Apache, MySQL, PHP (LAMP) en Ubuntu 14.04?. 16 Septiembre, 2016, de Digital Ocean Sitio web: https://www.digitalocean.com/community/tutorials/como-instalar-linux-apache-mysql-php-lamp-en-ubuntu-14-04-es