

# Unidad 1:

## Desarrollo de apps

### Actividad

#### Api rest

---

**Alumna:**  
Michel Narcisa  
Plaza Granoble



SCANN



UNIVERSIDAD  
TÉCNICA DE  
MANABÍ  
Fundada en 1952

## ACT. 3 – FUNCIONAMIENTO DE UNA API REST

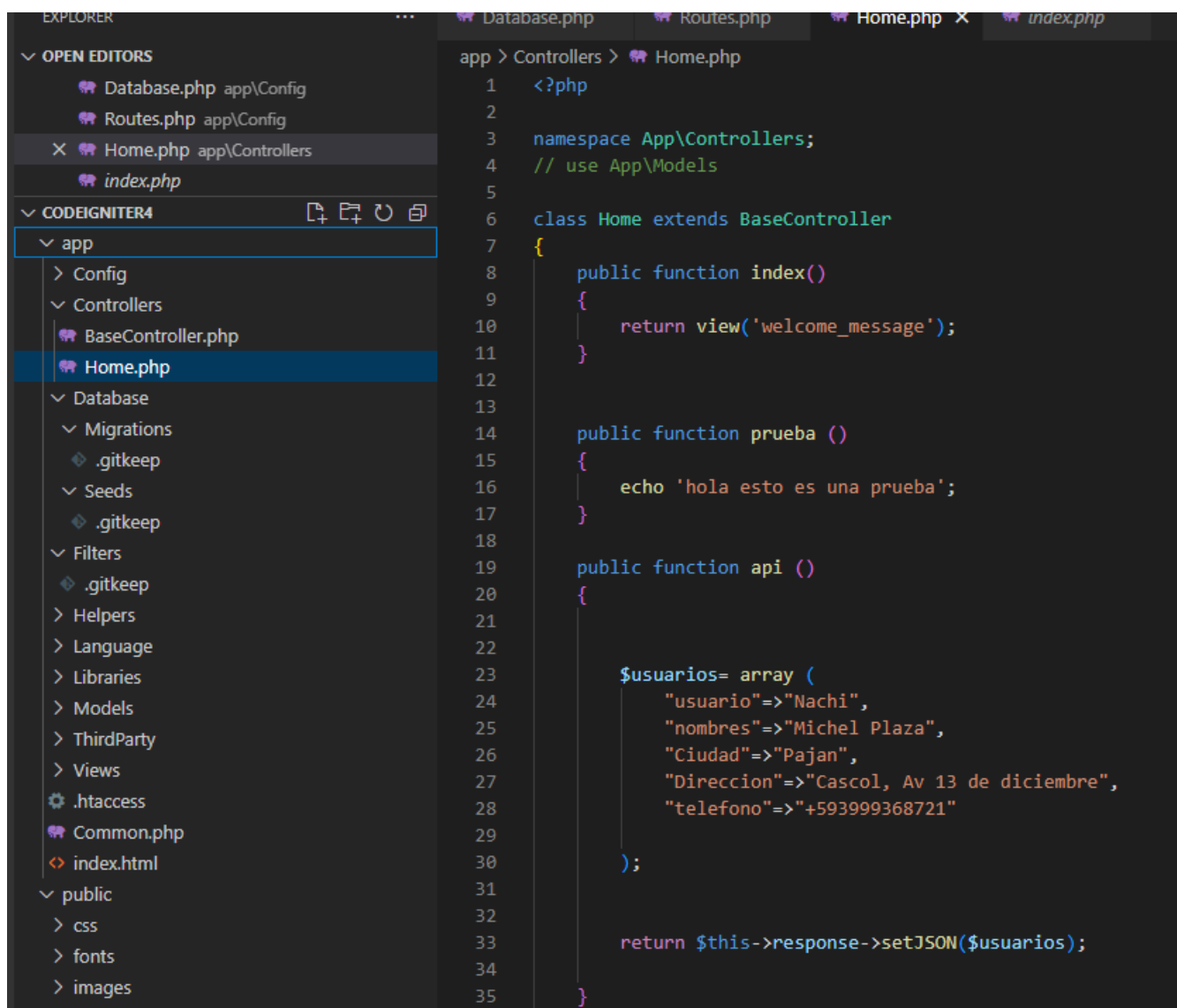
En base a lo desarrollado en clases cree una api rest

### Desarrollo de la actividad.

#### 1. Aplicaciones utilizadas para la actividad.

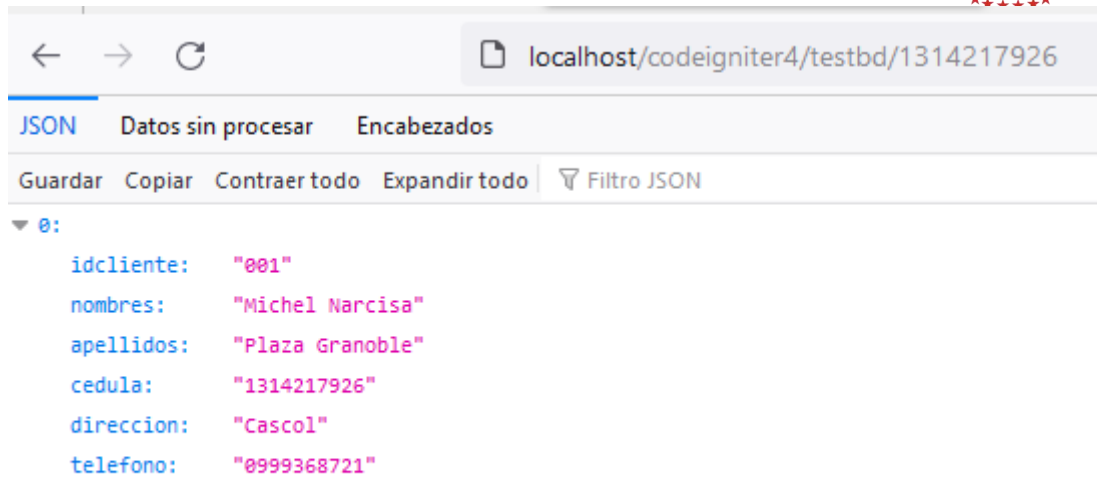
- \*XAMPP
- \*PostGree
- \*Visual Studio Code
- \*Coigniter4

#### 2. Crear la api rest, para esto trabajamos en la carpeta controllers buscamos en archivo home.php y creamos una api sencilla



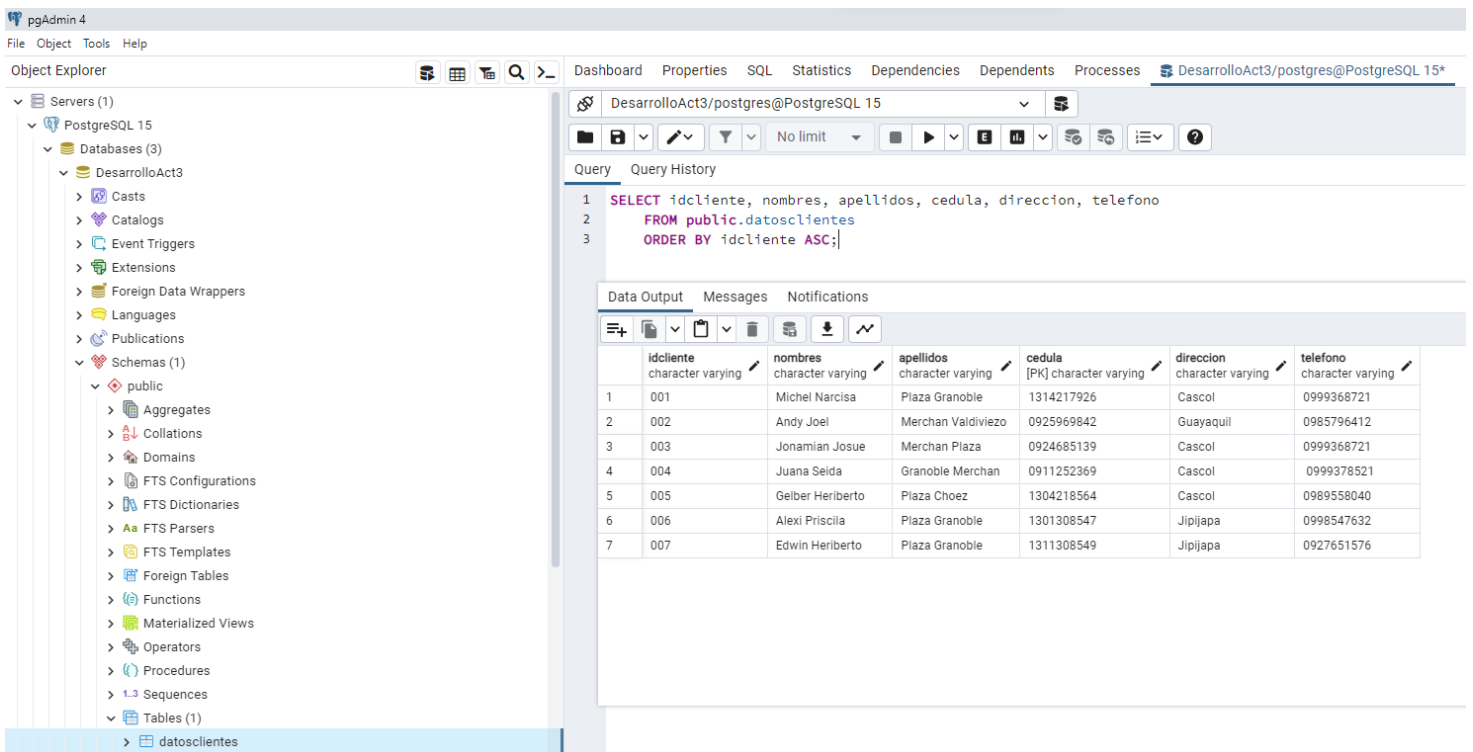
```
1  <?php
2
3  namespace App\Controllers;
4  // use App\Models
5
6  class Home extends BaseController
7  {
8      public function index()
9      {
10         return view('welcome_message');
11     }
12
13
14     public function prueba ()
15     {
16         echo 'hola esto es una prueba';
17     }
18
19     public function api ()
20     {
21
22
23         $usuarios= array (
24             "usuario"=>"Nachi",
25             "nombres"=>"Michel Plaza",
26             "Ciudad"=>"Pajan",
27             "Direccion"=>"Cascol, Av 13 de diciembre",
28             "telefono"=>"593999368721"
29         );
30
31
32
33         return $this->response->setJSON($usuarios);
34
35     }
```

Para poder visualizarla abrimos el localhost en el navegador



## ENLAZAR UN PROYECTO CON UNA BASE DE DATOS EN POSTGEE

1. Creamos una base de datos con una tabla que contenga varios datos, la cual será enlazada con nuestro proyecto.



2. Una vez hecho esto nos vamos a VS code donde seleccionaremos una configuración de database por default y la modificamos de acuerdo a la ubicación y datos de la bd creada en postgres.

```
*/  
public array $default = [  
    'DSN'      => 'pgsql:host=127.0.0.1;port=5432;dbname=DesarrolloAct3;user=postgres;password=123456',  
    'hostname' => 'localhost',  
    'username' => '',  
    'password' => '',  
    'database' => '',  
    'DBDriver' => 'Postgre',  
    'DBPrefix' => '',  
    'pConnect' => false,  
    'DBDebug'  => true,  
    'charset'  => 'utf8',  
    'DBCollat' => 'utf8_general_ci',  
    'swapPre'  => '',  
    'encrypt'  => false,  
    'compress' => false,  
    'strictOn' => false,  
    'failover' => [],  
    'port'     => 3306,  
];  
/**
```

3. Volvemos al archivo home para definir las instancias que permitan conectarse a la base de datos, en este caso he creado 2 funciones la primera muestra todos los datos almacenados en db en postgree

```
45 public function datosbd()  
46 {  
47  
48     $this->db=\Config\Database::connect();  
49     $query=$this->db->query("SELECT idcliente, nombres, apellidos, cedula, direccion, telefono FROM public.datosclientes ");  
50     $result=$query->getResult();  
51     return $this->response->setJSON($result);  
52  
53     // echo "hola";  
54 }  
55
```

En esta instancia podemos buscar los datos específicos con el número de cédula que fue la establecida como primary key.

```
public function testbd($cedula)  
{  
  
    $this->db=\Config\Database::connect();  
    $query=$this->db->query("SELECT idcliente, nombres, apellidos, cedula, direccion, telefono  
FROM public.datosclientes where cedula='$cedula' ");  
    $result=$query->getResult();  
    return $this->response->setJSON($result);  
  
}
```

#### 4. El ultimo paso sería definir las rutas y los parámetros de las mismas.

```
Database.php Routes.php X Home.php index.php
app > Config > Routes.php
5 // Create a new instance of our RouteCollection class.
6 $routes = Services::routes();
7
8 /*
9  * -----
10  * Router Setup
11  * -----
12  */
13 $routes->setDefaultNamespace('App\Controllers');
14 $routes->setDefaultController('Home');
15 $routes->setDefaultMethod('index');
16 $routes->setTranslateURIDashes(false);
17 $routes->set404Override();
18 // The Auto Routing (Legacy) is very dangerous. It is easy to create vulnerable ap
19 // where controller filters or CSRF protection are bypassed.
20 // If you don't want to define all routes, please use the Auto Routing (Improved).
21 // Set `autoRoutesImproved` to true in `app/Config/Feature.php` and set the follow
22 // $routes->setAutoRoute(false);
23
24 /*
25  * -----
26  * Route Definitions
27  * -----
28  */
29
30 // We get a performance increase by specifying the default
31 // route since we don't have to scan directories.
32 $routes->get('/', 'Home::index');
33 $routes->get('/prueba', 'Home::prueba');
34 $routes->get('/api', 'Home::api');
35 $routes->get('/login', 'Home::login');
36 $routes->get('/testbd/(:any)', 'Home::testbd/$1');
37 $routes->get('/datosbd', 'Home::datosbd');
38 /*
39  * -----
40  * Additional Routing
41  * -----
42  *
43  * There will often be times that you need additional routing and you
44  * need it to be able to override any defaults in this file. Environment
45  * based routes is one such time. require() additional route files here
46  * to make that happen.
47  *
48  * You will have access to the $routes object within that file without
49  * needing to reload it.
50  */
51 if (is_file(APPPATH . 'Config/' . ENVIRONMENT . '/Routes.php')) {
52     require APPPATH . 'Config/' . ENVIRONMENT . '/Routes.php';
53 }
```



5. Finalmente accedemos a nuestro navegador y abrimos el localhost para verificar que abra correctamente y no haya errores en el código. Como se puede visualizar aquí muestra los datos de la db postgree llamada datosclientes de manera general.

```
{
  "0": {
    "idcliente": "007",
    "nombres": "Edwin Heriberto",
    "apellidos": "Plaza Granoble",
    "cedula": "1311308549",
    "direccion": "Jipijapa",
    "telefono": "0927651576"
  },
  "1": {
    "idcliente": "006",
    "nombres": "Alexi Priscila",
    "apellidos": "Plaza Granoble",
    "cedula": "1301308547",
    "direccion": "Jipijapa",
    "telefono": "0998547632"
  },
  "2": {
    "idcliente": "005",
    "nombres": "Gelber Heriberto",
    "apellidos": "Plaza Choez",
    "cedula": "1304218564",
    "direccion": "Cascol",
    "telefono": "0989558040"
  },
  "3": {
    "idcliente": "004",
    "nombres": "Juana Seida",
    "apellidos": "Granoble Merchan",
    "cedula": "0911252369",
    "direccion": "Cascol",
    "telefono": "0999378521"
  },
  "4": {
    "idcliente": "003",
    "nombres": "Jonamian Josue",
    "apellidos": "Merchan Plaza",
    "cedula": "0924685139",
    "direccion": "Cascol",
    "telefono": "0999368721"
  },
  "5": {
    "idcliente": "002",
    "nombres": "Andy Joel",
    "apellidos": "Merchan Valdiviezo",
    "cedula": "0925969842",
    "direccion": "Guayaquil",
    "telefono": "0985796412"
  },
  "6": {
    "idcliente": "001",
    "nombres": "Michel Narcisa",
    "apellidos": "Plaza Granoble",
    "cedula": "1314217926",
    "direccion": "Cascol",
    "telefono": "0999368721"
  }
}
```



En esta imagen vemos el resultado de la función testbd donde podemos ubicar un número de cédula y nos muestra el resultado de los datos almacenados.

