API RESTFULL

Para este ejemplo vamos a crear un controlador para la tabla candidato

/var/www/html/proy_lara # **php artisan make:controller -r Api/CandidatoController**

Al ejecutar la sentencia anterior se crea la carpeta Api dentro de Http/Controllers y dentro de ella el archivo CandidatoController.php, abra el archivo y reemplace el contenido por el siguiente código:

```
<?php
namespace App\Http\Controllers\Api;
use App\Http\Controllers\Api\GenericController as GenericController;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;
use App\Models\Candidato;

class CandidatoController extends GenericController
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $candidatos = Candidato::all();
        $resp = $this->sendResponse($candidatos, "Listado de candidatos");
        return ($resp);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        //
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
```

```
 */
public function store(Request $request)
{

   $validacion = Validator::make($request->all(), [
      'nombrecompleto' => 'unique:candidato|required|max:200',
      'sexo' =>'required'
   ]);


   if ($validacion->fails())
      return $this->sendError("Error de validacion", $validacion->errors());

   $fotocandidato=""; $perfilcandidato="";

   if ($request->hasFile('foto')){
      $foto       = $request->file('foto');
      $fotocandidato= $foto->getClientOriginalName();
   }
   if ($request->hasFile('perfil')){
      $perfil         = $request->file('perfil');
      $perfilcandidato = $perfil->getClientOriginalName();
   }

   $campos             =  array(
      'nombrecompleto' => $request->nombrecompleto,
      'sexo'           => $request->sexo,
      'foto'          => $fotocandidato,
      'perfil'        => $perfilcandidato,
   );

   if ($request->hasFile('foto')) $foto->move(public_path('img'), $fotocandidato);
   if ($request->hasFile('perfil')) $perfil->move(public_path('img'), $perfilcandidato);


   $candidato = Candidato::create($campos);
   $resp = $this->sendResponse($candidato,
         "Guardado...");
   return($resp);



} //--- End store

/**
 * Display the specified resource.
 *
 * @param  int  $id
 * @return \Illuminate\Http\Response
```

```
     */
    public function show($id)
    {
        //
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        //
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
        //
    }
    /**
     * Remove the specified resource from storage.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        //
    }
}
```

Hemos agregado código solo a dos métodos el listado (index) y para guardar (store).

Es importante hacer notar que la clase anterior usa una clase de nombre GenericController. Sin embargo dicho archivo no se crea de manera automática. Por lo tanto, hay que crearlo manualmente en la misma carpeta con el nombre de GenericController.php y cuyo contenido es el siguiente fragmento de código:

```
<?php
namespace App\Http\Controllers\Api;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller as Controller;
class GenericController extends Controller
{
   /**
    * success response method.
    *
    * @return \Illuminate\Http\Response
    */
   public function sendResponse($result, $message)
   {
         $response = [
         'success' => true,
         'data'    => $result,
         'message' => $message,
      ];
      return response()->json($response, 200);
   }
   /**
    * return error response.
    *
    * @return \Illuminate\Http\Response
    */
   public function sendError($error, $errorMessages = [], $code = 404)
   {
         $response = [
         'success' => false,
         'message' => $error,
         'data'=>[],
      ];
      if(!empty($errorMessages)){
         $response['data'] = $errorMessages;
      }
      return response()->json($response, $code);
   }
}
```

Esta última clase dispone de dos métodos que serán usados en cualquier subclase que ha de exponer sus métodos a través de la API.

Para que nuestra api esté disponible debemos editar el archivo **routes/api.php** agregando la siguiente línea al final del contenido

Route::resource("candidato", CandidatoController::class);

Vamos a probar nuestra api con dos clientes distintos (curl y Postman)

Primero insertamos dos registros a la tabla candidatos

```
INSERT INTO candidato (nombrecompleto, foto,  sexo, perfil)
VALUES
('Ambrosio Cardoso Jimenez','cardoso.png','M','cardoso.pdf'),
('Adolfo Angel Cardoso Vasquez','','M','');
```

curl

curl http://localhost:8000/api/candidato

{
- "success":true,
- "data":[
    1. {
        - "id":1,
        - "nombrecompleto":"Ambrosio Cardoso Jiménez",
        - "foto":"cardoso.png",
        - "sexo":"M",
        - "perfil":"cardoso.pdf"

    },
    2. {
        - "id":2,
        - "nombrecompleto":"Adolfo Angel Cardoso Vasquez",
        - "foto":"",
        - "sexo":"M",
        - "perfil":""

    }

  ],
- "message":"Listado de candidatos"

}

Postman

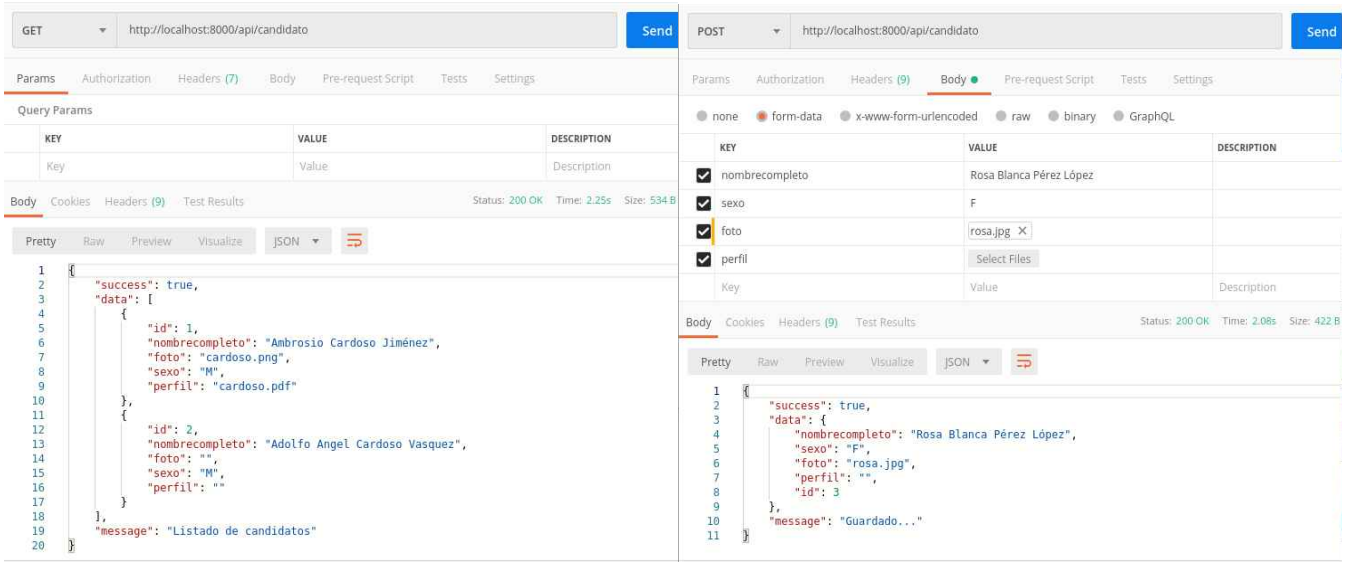

Figura 6. Listado de candidatos y agregar nuevo candidato