**API REST gems\_backend**

Antes de iniciar el servidor, diríjase al repositorio creado en GitHub en la rama correspondiente y descargue los archivos **.env.example** y **orgconfig.env.example.** Una vez descargados, colóquelos fuera de la carpeta **src** del proyecto y renombre ambos archivos quitándoles el **.example**.

Los archivos con extensión **.env** sirven para guardar en variables los valores correspondientes a información personal de la maquina del usuario y requerida por la API para funcionar correctamente como por ejemplo: el **username** y **password** para acceder a su cliente de Bases de Datos como **phpmyadmin**.

Instale el programa **XAMPP** server o los programas: **Apache, MySQL, php y phpMyAdmin,** para administrar sus Bases de Datos.

En el archivo .env coloque su usuario y contraseña (si la posee) para ingresar a su cliente de Bases de datos en las variables **USERNAME** y **PASSWORD**. Por defecto el usuario es root y no tiene contraseña. De igual modo, agregue los mismos datos en el archivo ormconfig.env en las variables **TYPEORM\_USERNAME** y **TYPEORM\_PASSWORD.**

Cree una Base de datos con el nombre establecido en la variable **DATABASE** del archivo .env (la variable **TYPEORM\_DATABASE** del archivo ormconfig.env tiene el mismo valor).

Para ejecutar esta API, ingrese el comando:

**npm start**, si es en modo de producción

**npm run start:dev**, si es en modo de desarrollo y pruebas.

Posteriormente a la ejecución del servidor, ejecute en otra terminal el comando para generar una lista de países previamente programados en la API y cargarlos a la Base de Datos:

**npm run schema:drop && npm run seed:run**

1. **Entidades.**

**Usuario:**

* id: identificador autogenerado del Usuario.
* username: nombre único del usuario. Es obligatorio
* email: correo electrónico único. Es obligatorio.
* password: contraseña. Es obligatorio.
* first\_name: nombre. Es obligatorio.
* last\_name: apellido. Es obligatorio.
* phone: teléfono.
* address: dirección.
* profesionalID: código de colegio profesional.
* country: nombre del país de origen. Es obligatorio.

**Country:**

* id: identificador autogenerado del Pais.
* code: código del país. Es obligatorio.
* prefix: prefijo del país. Es obligatorio.
* name: nombre del país. Es obligatorio.

**Project:**

* id: identificador autogenerado del proyecto
* name: nombre único del proyecto. Es obligatorio.

**Report:**

* **id: identificador autogenerado del reporte**
* **current: es obligatorio.**
* **cable\_width: ancho del cable. Es obligatorio.**
* **pipe\_diameter: diámetro de la tubería. Es obligatorio**
* **protection\_device: dispositivo de protección. Es obligatorio;**
* **voltaje\_drop: caída de voltaje. Es obligatorio**

1. **Rutas.**

**Usuario:**

* **Método: GET, Ruta: /user**

No recibe valores, devuelve un arreglo formado por todos los registros de usuarios que existan en la Base de Datos.

* **Metodo: GET, Ruta: /user/:id**

Recibe el id del usuario y se devuelve el registro del usuario correspondiente al id que se envió.

* **Metodo: POST, Ruta: /user**

Recibe un objeto formado por los atributos obligatorios del usuario: **username, email, password, first\_name, last\_name, profesionalID.** También puede recibir el resto de los atributos, pero son opcionales. Devuelve un mensaje con el aviso de si pudo o no ser registrado el usuario.

* **Metodo: PATCH, Ruta: /user/:id**

Recibe el id del usuario y un objeto con la lista de atributos que se quieren modificar en el registro del usuario correspondiente. Devuelve un mensaje con el aviso de si se pudo o no realizar la actualización de datos.

* **Metodo: DELETE, Ruta: /user/:id**

Recibe el id del usuario cuyo registro se quiere eliminar. Devuelve un mensaje con el aviso de si se pudo o no eliminar el registro.

**Country:**

* **Método: GET, Ruta: /country**

No recibe valores, devuelve un arreglo formato por todos los registros de paises que existan en la Base de Datos.

* **Metodo: GET, Ruta: /country/:id**

Recibe el id del usuario y se devuelve el registro del país correspondiente al id que se envió.

* **Metodo: POST, Ruta: /country**

Recibe un objeto formado por los atributos obligatorios del país: **code, prefix, name.** Devuelve un mensaje con el aviso de si pudo o no ser registrado el país.

* **Metodo: PATCH, Ruta: /country/:id**

Recibe el id del país y un objeto con la lista de atributos que se quieren modificar en el registro del país correspondiente. Devuelve un mensaje con el aviso de si se pudo o no realizar la actualización de datos.

* **Metodo: DELETE, Ruta: /country/:id**

Recibe el id del país cuyo registro se quiere eliminar. Devuelve un mensaje con el aviso de si se pudo o no eliminar el registro

**Project:** estas rutas solo pueden ser accedidas una vez se haya iniciado sesión

* **Método: GET, Ruta: /project**

No recibe valores, devuelve un arreglo formado por el nombre de todos los registros de proyectos que existan en la Base de Datos que pertenezcan al usuario que inicio sesión.

* **Metodo: GET, Ruta: /project/:id**

Recibe el id del proyecto y se devuelve el registro del proyecto correspondiente al id que se envió.

* **Metodo: POST, Ruta: /project**

Recibe un objeto formado por el atributo obligatorio del proyecto: **name.** Devuelve un mensaje con el aviso de si pudo o no ser registrado el proyecto.

* **Metodo: PATCH, Ruta: /country/:id**

Recibe el id del proyecto y un objeto con el nombre que se quiere modificar en el registro del proyecto correspondiente. Devuelve un mensaje con el aviso de si se pudo o no realizar la actualización de datos.

* **Metodo: DELETE, Ruta: /project/:id**

Recibe el id del proyecto cuyo registro se quiere eliminar. Devuelve un mensaje con el aviso de si se pudo o no eliminar el registro.

**Auth:**

* **Metodo POST, Ruta: /auth/login**

Recibe un objeto con el email y password del usuario. Devuelve un objeto formado por un mensaje de si pudo o no ser exitosa la operación y un token en caso de que los datos sean correctos.

* **Metodo POST, Ruta: /auth/change-password**

Esta ruta solo puede ser accedida por el usuario cuando haya iniciado sesión. Recibe un objeto formado por la contraseña actual y la nueva contraseña. Devuelve un mensaje con el aviso de si pudo o no ser exitosa la operación.

* **Metodo POST, Ruta: /auth/logout**

Esta ruta solo puede ser accedida por el usuario cuando haya iniciado sesión. No recibe nada y elimina la cookie que se usó para almacenar la sesión del usuario.

**Report:** estas rutas solo pueden ser accedidas una vez se haya iniciado sesión

* **Método: GET, Ruta: /report**

No recibe valores, devuelve un arreglo formado por los registros de todos los reportes que existan en la Base de Datos que pertenezcan al usuario que inicio sesión.

* **Método: GET, Ruta: /report/listForm**

No recibe valores, devuelve un objeto con los valores previos a la inserción en la tabla de reportes correspondientes al usuario que inicio sesión. Por el momento, los valores pertenecientes a este objeto son aleatorios.

* **Metodo: GET, Ruta: /report/:id**

Recibe el id del reporte y se devuelve el registro del reporte correspondiente al id que se envió.

* **Metodo: POST, Ruta: /report**

Recibe un objeto formado por los atributos obligatorios del reporte: **current, cable\_width, pipe\_diameter, protection\_device, voltaje\_drop.** Devuelve un mensaje con el aviso de si pudo o no ser registrado el proyecto.

* **Metodo: PATCH, Ruta: /report/:id**

Recibe el id del proyecto y un objeto con los valores que se quiere modificar en el registro del reporte correspondiente. Devuelve un mensaje con el aviso de si se pudo o no realizar la actualización de datos.

* **Metodo: DELETE, Ruta: /report/:id**

Recibe el id del reporte cuyo registro se quiere eliminar. Devuelve un mensaje con el aviso de si se pudo o no eliminar el registro.