## Para comenzar.

A quien corresponda, he agregado a la carpeta del GIT los recursos necesarios (Proyecto, BD y
Colección de POSTMAN) para su correcta ejecución.

## RUTAS LOCALHOST

**#### Registrar usuario Local ####**
**URL:** http://localhost/prueba-Geniat/addUser
**Method:** POST
**Body FormData:** nombre, apellido, correo, password, rol

**### Consultar Usuarios Local ### (Para visualizar los datos de los usuarios)**
**URL:** localhost/prueba-Geniat/users
**Method:** GET

**### Login Local ###**
**URL:** localhost/prueba-Geniat/iniciar_sesion
**Method:** POST
**Body FormData:** correo, password

**### Cerrar sesion Local ### (Lo agregue para cambiar de usuarios)**
**URL:** localhost/prueba-Geniat/iniciar_sesion
**Method:** GET

**### Crear Publicación Local ###**
**URL:** localhost/prueba-Geniat/createPost
**Method:** POST
**Body FormData:** titulo, descripcion

**### Actualizar Publicación Local ###**
**URL:** localhost/prueba-Geniat/updatePost
**Method:** PUT
**Body JSON: {** "idPublicacion": "", "titulo": "", "descripcion": "" **}**

**### Eliminar Publicación Local ###**
**URL:** localhost/prueba-Geniat/deletePost/$Id
**Method:** DELETE

**### Consultar Publicaciones Local ###**
**URL:** localhost/prueba-Geniat/Posts
**Method:** GET

**Las rutas ya vienen definidas dentro del archivo Coleccion Postman Local.**

## RUTAS HOSTING

**#### Registrar usuario Local ####**
**URL:** https://geniatproyect.000webhostapp.com/addUser
**Method:** POST
**Body FormData:** nombre, apellido, correo, password, rol

**### Consultar Usuarios Local ### (Para visualizar los datos de los usuarios)**
**URL:** https://geniatproyect.000webhostapp.com/users
**Method:** GET

**### Login Local ###**
**URL:** https://geniatproyect.000webhostapp.com/iniciar_sesion
**Method:** POST
**Body FormData:** correo, password

**### Cerrar sesion Local ### (Lo agregue para cambiar de usuarios)**
**URL:** https://geniatproyect.000webhostapp.com/iniciar_sesion
**Method:** GET

**### Crear Publicación Local ###**
**URL:** https://geniatproyect.000webhostapp.com/createPost
**Method:** POST
**Body FormData:** titulo, descripcion

**### Actualizar Publicación Local ### SOLO DISPONIBLE EN LOCAL**
**URL:** https://geniatproyect.000webhostapp.com/updatePost
**Method:** PUT
**Body JSON: {** "idPublicacion": "", "titulo": "", "descripcion": "" **}**

**### Eliminar Publicación Local ### SOLO DISPONIBLE EN LOCAL**
**URL:** https://geniatproyect.000webhostapp.com/deletePost/$Id
**Method:** DELETE

**### Consultar Publicaciones Local ###**
**URL:** https://geniatproyect.000webhostapp.com/Posts
**Method:** GET

**Las rutas ya vienen definidas dentro del archivo Coleccion Postman Host.**

Los roles los tengo distribuidos en una tabla aparte con un identificador.

Se valida el rol dependiendo de las funciones que se realicen le dejara o no si tiene los permisos suficientes y se devuelve una respuesta ya definida dependiendo de las acciones realizadas.

- ID 1 - Rol básico - permiso de acceso
- ID 2 - Rol medio - permiso de acceso y consulta
- ID 3 - Rol medio alto - Permiso de de acceso y agregar
- ID 4 - Rol alto medio - permiso de acceso, consulta, agregar y actualizar
- ID 5 - Rol alto - Permiso de acceso, consulta, agregar, actualizar y eliminar

# REGISTRAR USUARIO (POST)

Para registrar un usuario de requieren 5 datos (nombre, apellido, correo, password, rol), el servidor devolverá una respuesta 400 badRequest si no se cumplen las validaciones de los campos, caso contrario si todo sale con exito se devuelve una respuesta de estatus 201 Created, devolvera un mensaje.

- Validaciones:
  a) **Nombre:** no debe ir vació y es de máximo 40 caracteres.
  b) **Apellido:** no debe ir vació y es de máximo 40 caracteres.
  c) **Correo:** no debe ir vació y es de máximo 60 caracteres, el correo no puede estar duplicado en la base de datos.
  d) **Password:** no debe ir vació.
  e) **Rol:** no debe ir vació, debe ser numerico y es de máximo 10 digitos y el Rol debe existir en la tabla de roles.

- Al ser registrado con exito el usuario la contraseña se encriptara con MD5.

# INICIAR SESION (POST)

Para iniciar sesión de requieren 2 datos (correo, password), el servidor devolverá una respuesta 400 badRequest si no se cumplen las validaciones de los campos, caso contrario si todo sale con exito se devuelve una respuesta 200 OK, devolvera el token y un mensaje.

Validaciones:
**Correo:** no debe ir vació y es de máximo 60 caracteres.
**Password:** no debe ir vació.

Al iniciar sesión se genera un token con JWT que durará máximo un día y se actualizara en la base de datos el token en el usuario y se guardaran las cookies durante maximo 1 dia (uniqueToken, idRol, idUsuario, correo).

# CREAR PUBLICACION (POST)

Para crear una publicación de requieren 2 datos (titulo y descripcion), el servidor devolverá una respuesta 400 badRequest si no se cumplen las validaciones de los campos, caso contrario si todo sale con exito se devuelve una respuesta 201 Created, un mensaje, la data del registro y el token.

Validaciones:

**Titulo:** El campo no puede ir vació, debe contener como máximo 50 caracteres.

**Descripción:** El campo no puede ir vació, debe contener como máximo 1000 caracteres.

# CONSULTAR PUBLICACION (GET)

Para consultar las publicaciones no requieren datos solo tener un rol que permita realizar la consulta, el servidor devolverá una respuesta 401 Unauthorized si no se cumplen con los permisos, caso contrario si todo sale con exito se devuelve una respuesta 200 OK, un mensaje y el token.

Validaciones:

**Titulo:** El campo no puede ir vació, debe contener como máximo 50 caracteres.

**Descripción:** El campo no puede ir vació, debe contener como máximo 1000 caracteres.

*****EL METODO “DELETE” Y “PUT” NO ESTAN DISPONIBLES EN LA*****
***********VERSION GRATUITA DEL HOSTING, DEBE SER DE PAGO*********

# ACTUALIZAR PUBLICACION (PUT)

Para actualizar una publicación de requieren 3 datos (idPublicacion, titulo y descripcion), el servidor devolverá una respuesta 400 badRequest si no se cumplen las validaciones de los campos, caso contrario si todo sale con exito se devuelve una respuesta 201 Created, un mensaje, la data del update y el token.

Validaciones:

**idPublicacion:** El campo no puede ir vació, el id de la publicación debe existir en la base de datos y no debe estar eliminado.

**Titulo:** El campo no puede ir vació, debe contener como máximo 50 caracteres.

**Descripción:** El campo no puede ir vació, debe contener como máximo 1000 caracteres.

# ELIMINAR PUBLICACION (DELETE)

Para eliminar una publicación de requieren 1 datos (idPublicacion), el servidor devolverá una respuesta 400 badRequest si no se cumplen las validaciones de los campos, caso contrario si todo sale con exito se devuelve una respuesta 200 OK, un mensaje, la data del registro eliminado y el token.

Validaciones:

**idPublicacion:** El campo no puede ir vació, el id de la publicación debe existir en la base de datos y no debe estar eliminado.