



**UNIVERSIDAD VALLE DEL MOMBOY; VICERRECTORADO
FACULTAD DE INGENIERÍA; PERIODO 2024A
CARRERA INGENIERÍA EN COMPUTACIÓN
BACKEND; SECCIÓN VIV/FI
UNIDAD IV**

e-actividad 4.1:

Proyecto Final: Sistema de Gestión Universitario “UVM”

Manual de Usuario

Integrantes:

Jonathan Antonio Arellano Márquez C.I: 24.190.278

José Enmanuel Cardoza Ferrer C.I: 31.590.138

Luis Fernando Araujo Giardinella C.I. 26.482.894

Denilson Eduardo Castellanos Garcia C.I. 29.694.566

***Tutor:* Profesor Roberto Di Michele**

Contenido

Introducción.....	3
Funcionalidades Clave.....	4
Integración con Base de Datos.....	4
Visualización de Actividades.....	4
Protección de Datos.....	4
Facilidad de Manejo.....	4
Instalación del Sistema.....	5
Requisitos.....	5
Visual Studio Code (VSCode).....	5
Node.js.....	6
XAMPP.....	7
MySQL Workbench.....	8
GitHub.....	9
Postman.....	10
Despliegue.....	11
Paso 1: Clonando el Repositorio con GitHub.....	11
Paso 2: Ejecutar el Servidor Local MySQL con XAMPP.....	12
Paso 3: Crear la Base de Datos MySQL Local con MySQL Workbench.....	12
Paso 4: Crear el archivo “.env”.....	14
Paso 5: Ejecutar el Servidor del SGU.....	15
Paso 6: Ejecutar Postman.....	16
Uso del Sistema.....	17
Consideraciones.....	17
Restricciones.....	18
Registrar un Usuario.....	19
Login.....	19
Visualizar Materias y Secciones de un Profesor.....	20
Visualizar Eventos de una Materia.....	20
Visualizar Eventos en 2 Semanas para Profesores.....	20
Visualizar Eventos de una Sección.....	21
Asignar Eventos de una Materia a una Sección.....	21
Manejo de Registros y Datos.....	22
Listar (Ver) Registros - GET.....	22
Agregar un Registro - POST.....	22
Editar un Registro - PUT.....	24
Eliminar un Registro - DELETE.....	26
Créditos y Comentarios de Desarrollador.....	27
Diario del Proyecto.....	27
Comentarios de Desarrollo.....	28
Créditos.....	29

Introducción

El presente proyecto “Sistema de Gestión Universitario” (SGU) para la Universidad Valle del Momboy (UVM) trata acerca de una continuación a todos los proyectos anteriores del mismo nombre, cuyo objetivo es presentar un sistema capaz de listar las actividades pautadas para cada semana dentro de un periodo trimestral de clases. El sistema también mantiene otras funcionalidades como almacenamiento de información referente a profesores, materias, secciones, eventos y trimestres. De igual manera el sistema permite la libre edición de todos los registros referentes a lo mencionado anteriormente.

Este proyecto fue desarrollado con la idea principal de facilitar al usuario preparar un itinerario o incluso informarse de qué actividades futuras se acercan, e incluso permite al usuario mantener un registro de las actividades pasadas. Dentro del proyecto se puede resaltar la accesibilidad a editar, desarrollar y ordenar toda la información necesaria o que hacen referencia a las actividades de la universidad.

En la continuación del proyecto actualmente desarrollado, se han implementado características clave para asegurar la protección de los registros a través de un sistema de usuarios y tokens, este último siendo unas claves generadas por el sistema para identificar qué tipo de usuario está trabajando el SGU, y permitir o negar las operaciones dependiendo de su nivel de permisos. De la misma manera, el SGU también ha mejorado en el ámbito de accesibilidad, obteniendo mejores vistas al listar los datos, y presentando los correspondientes mensajes de advertencia en caso de que sean introducidos datos inválidos o con formato incorrecto.

Funcionalidades Clave

Integración con Base de Datos

El sistema está integrado con una base de datos que almacena toda la información relacionada con eventos, materias, profesores, trimestres, y usuarios; asegurando la persistencia y seguridad de los datos. De la misma manera el sistema permite listar, agregar, editar y eliminar cualquier dato según el usuario crea necesario.

Visualización de Actividades

Ofrece una vista de las actividades programadas, permitiendo a los usuarios visualizar de manera clara y ordenada los eventos próximos, facilitando así la organización personal y académica.

Protección de Datos

Se le puede especificar al SGU qué tipo de usuario es el que interactúa con el proyecto, limitando así las funcionalidades dependiendo de la situación, con el fin de proteger los datos registrados y mantener el orden de la jerarquía. Esto es logrado a través de un sistema de registro, donde el usuario de mayor poder puede crear cuentas cuyas credenciales son encriptadas para aún mayor seguridad.

Facilidad de Manejo

A través de validaciones a las entradas que el usuario introduce, el SGU puede sanitizar los datos y presentar un mensaje de advertencia en caso de que los datos ingresados no se encuentren en un formato adecuado, brindando así al usuario una mejor claridad acerca de lo que se necesita solucionar en ese caso.

Instalación del Sistema

Requisitos

Visual Studio Code (VSCode)

Es un editor de código fuente desarrollado por Microsoft. Es ligero, potente y soporta una gran variedad de lenguajes de programación y archivos. Para este proyecto, se utilizará para ejecutar el servidor local del SGU.

1. El usuario debe dirigirse al siguiente enlace de descarga:
<https://code.visualstudio.com/download>
2. A continuación, debe hacer clic en el botón de descarga para el sistema operativo que utiliza, sea Windows, macOS o Linux.
3. Tras descargar el archivo de instalación, debe abrirlo y seguir las instrucciones proporcionadas por el asistente de instalación.
4. Una vez completada la instalación, VSCode estará listo para su uso.



Imagen: Página web de descarga para VSCode

Node.js

Es el entorno de ejecución para JavaScript. En este proyecto, Node.js es necesario para ejecutar el servidor del proyecto SGU, más específicamente nos permitirá alojar los endpoints, permitiendo la interacción con el sistema a través de protocolos HTTP.

- 1. El usuario debe dirigirse al sitio web oficial de Node.js para iniciar la descarga: <https://nodejs.org/en>
- 2. Luego, debe hacer clic en el botón de descarga que corresponda al sistema operativo que esté utilizando, ya sea Windows, macOS o Linux.
- 3. Una vez descargado el archivo de instalación, se debe de abrir y seguir las instrucciones del asistente de instalación.
- 4. Durante la instalación, es importante asegurarse de marcar la casilla que indica "Instalar herramientas de línea de comandos" para poder utilizar Node.js desde la terminal.
- 5. El usuario puede abrir la terminal y verificar que Node.js se haya instalado correctamente escribiendo el comando **node -v** y/o **npm -v**. De esta forma, debería ver la versión de Node.js y npm que han sido instaladas.

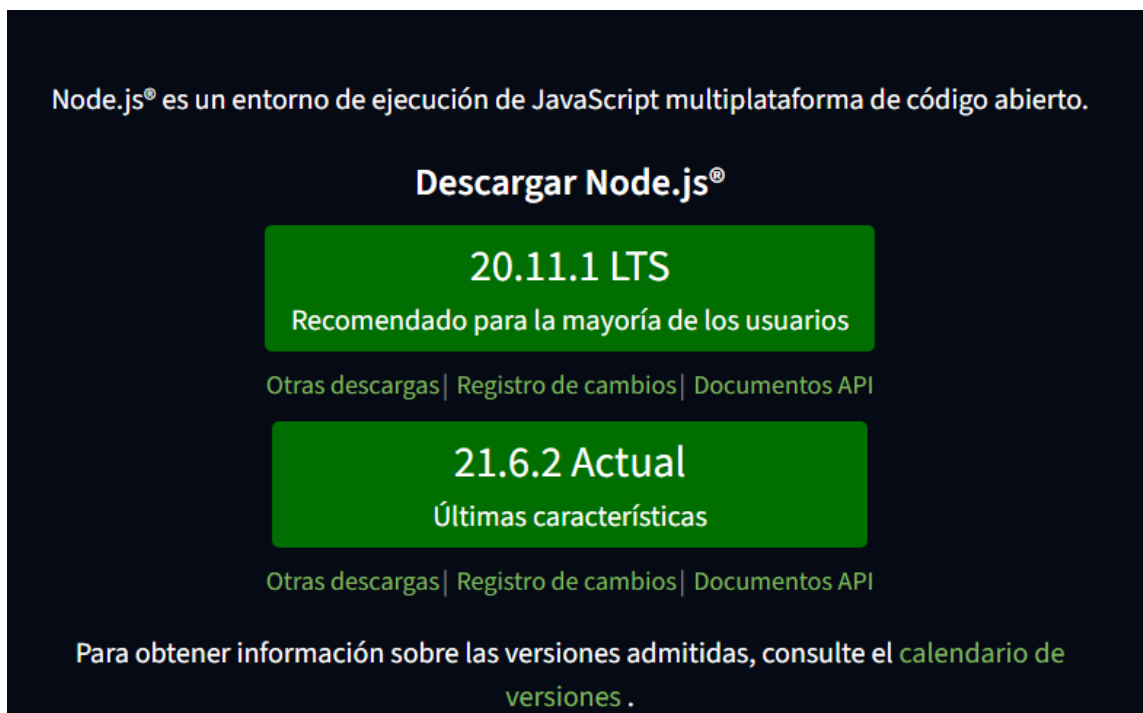


Imagen: Página web de descarga para Node.js

XAMPP

Es una distribución de Apache fácil de instalar que contiene MariaDB, PHP y Perl. Para el propósito de este proyecto, XAMPP se utilizará principalmente para su servidor MySQL, el cual proporcionará el sistema de gestión de bases de datos necesario para almacenar y gestionar la información del proyecto SGU.

1. El usuario debe dirigirse al sitio web oficial de XAMPP para iniciar su descarga:
<https://www.apachefriends.org/index.html>
2. Debe descargar la versión de XAMPP que corresponda a su sistema operativo (Windows, macOS o Linux).
3. Una vez descargado el archivo de instalación, debe seguir las instrucciones del asistente de instalación.
4. Durante la instalación, el usuario debe elegir los componentes que desea instalar (Apache, MySQL, PHP, phpMyAdmin, etc.). Se puede dejar las opciones predeterminadas para una instalación básica.
5. El usuario debe elegir la carpeta de instalación para XAMPP. Por lo general, se recomienda instalarlo en la carpeta predeterminada.
6. Debe completar la instalación y asegurarse de que los servicios de Apache y MySQL se inicien automáticamente al finalizar la instalación.



Imagen: Página web de descarga para XAMPP

MySQL Workbench

Es una herramienta visual de diseño de bases de datos que integra desarrollo, administración, diseño, creación y mantenimiento de bases de datos en un único entorno integrado. En este proyecto, MySQL Workbench se utilizará para diseñar, crear y gestionar la base de datos local que almacena toda la información relevante del sistema.

- 1. Dirigirse al sitio web oficial de MySQL Workbench:
<https://dev.mysql.com/downloads/workbench/>
- 2. Descargar la versión de MySQL Workbench que corresponde al sistema operativo en donde el SGU será utilizado (Windows, macOS o Linux).
- 3. Una vez descargado el archivo de instalación, se deben seguir las instrucciones del asistente de instalación.
- 4. Durante la instalación, puedes elegir las opciones predeterminadas. Se recomienda al usuario instalar con las opciones por defecto.

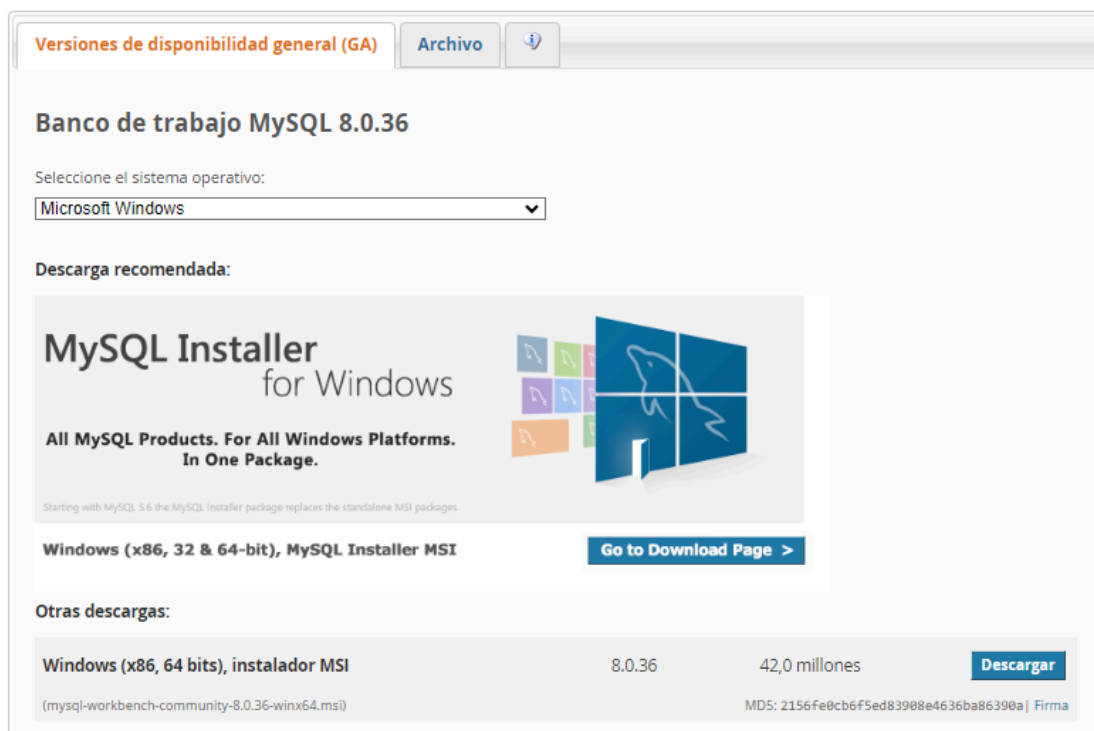


Imagen: Página web de descarga para Mysql Workbench

GitHub

Es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git. Se empleará para clonar el repositorio del proyecto localmente, permitiendo el acceso al código fuente y para la fácil implementación del proyecto en la máquina local. Para usarlo deberás seguir los siguientes pasos:

- 1. Crear una cuenta en GitHub: El usuario debe ir al sitio web oficial de GitHub (<https://github.com/>) para crear una cuenta propia.
- 2. Descargar e instalar la aplicación GitHub Desktop: Se necesita la aplicación para instalar el SGU localmente, clonando el repositorio a la máquina local. A través del sitio web de descarga (<https://desktop.github.com/>) el usuario debe seguir las instrucciones para descargar y, una vez descargado, instalar la aplicación.
- 3. Iniciar sesión en la aplicación: Una vez la aplicación ha sido instalada, el usuario puede iniciar sesión para poder clonar repositorios.

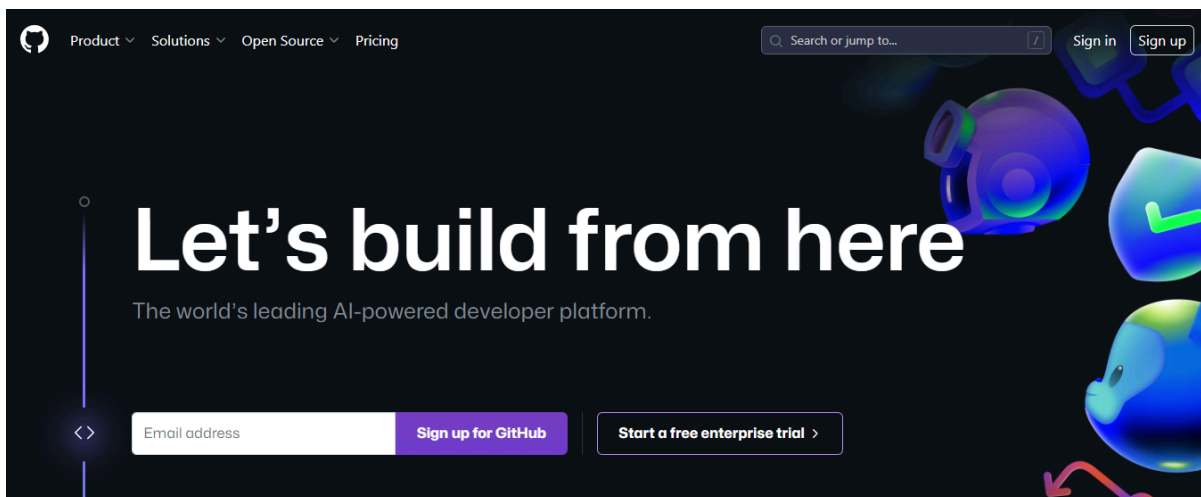


Imagen: Página web de GitHub

Postman

Es una herramienta para probar APIs. Permite enviar peticiones HTTP a una API y recibir respuestas, facilitando la prueba y depuración de los endpoints del servidor. En este proyecto, Postman se utilizará para interactuar con los endpoints del sistema, es decir, es la principal herramienta para el uso del proyecto SGU.

Para descargar Postman, accedemos al sitio web oficial de la herramienta (<https://www.postman.com/downloads/>) y hacemos clic en el botón de descarga correspondiente a su sistema operativo. Una vez descargado el archivo, se debe ejecutar el instalador y seguir las instrucciones del asistente de instalación. Postman está disponible de forma gratuita para Windows, macOS y Linux, y ofrece una interfaz intuitiva para realizar pruebas de API de manera eficiente.

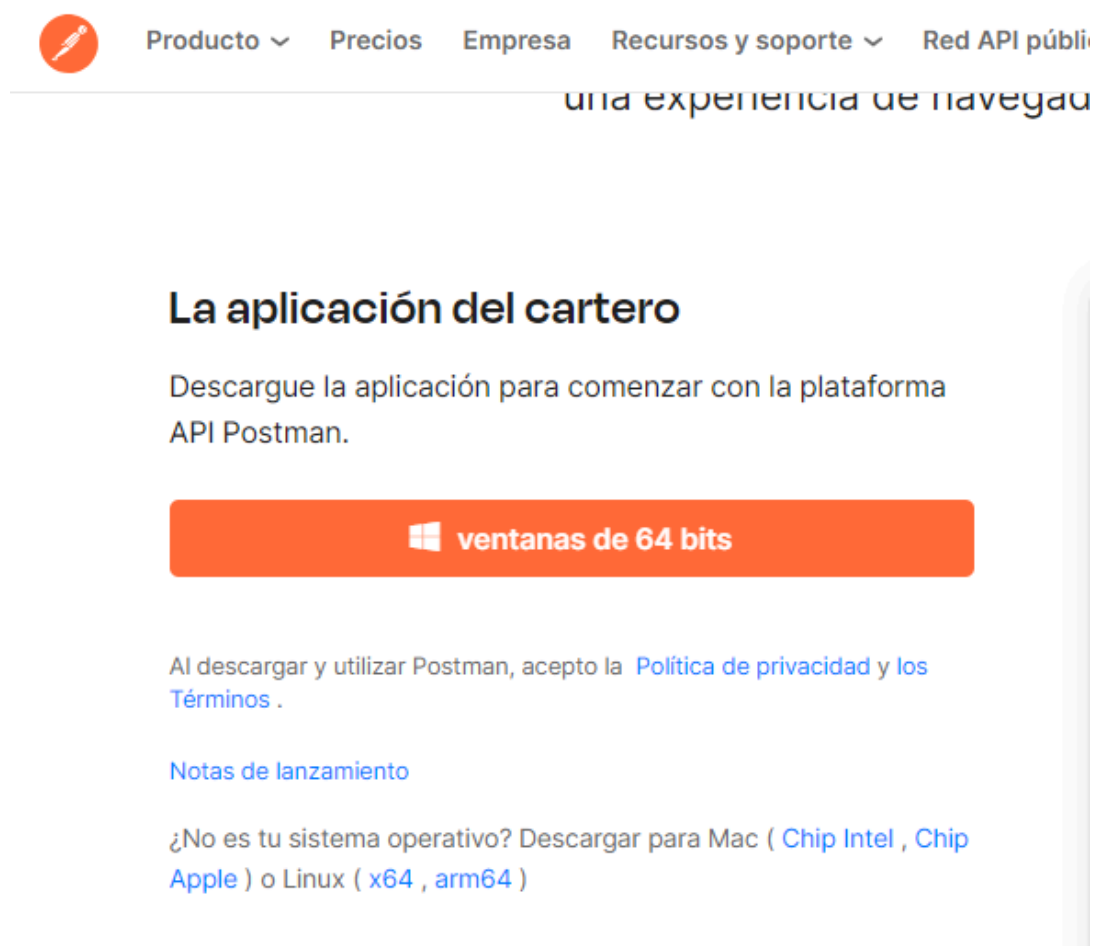


Imagen: Página web de descarga para Postman

Despliegue

Una vez la máquina local posee los requisitos necesarios, se puede proceder a la instalación y despliegue del sistema. El correcto despliegue brindará seguridad y minimizará los inconvenientes a futuro durante el uso del SGU.

Paso 1: Clonando el Repositorio con GitHub

El proceso de despliegue comienza al obtener los archivos necesarios del sistema a través de la aplicación de GitHub instalada. El usuario puede clonar el repositorio a través de la URL <https://github.com/Luixls/ea4.1-proyecto> y seleccionando la opción correcta:

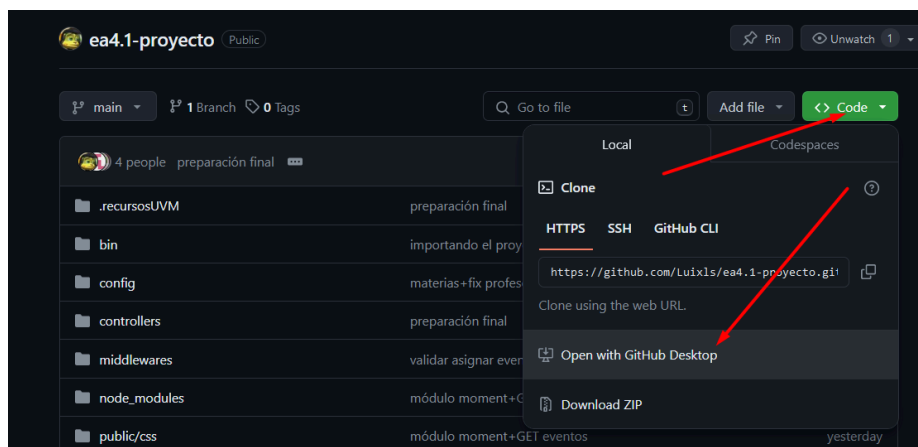


Imagen: Seleccionando la opción para obtener el repositorio con GitHub Desktop. Existe la opción de descargar el archivo zip, pero para este ejemplo, continuaremos con la aplicación GitHub Desktop.

La aplicación detectará que se quiere clonar un repositorio a la máquina local, y presentará el cuadro de confirmación para que el usuario seleccione qué nombre desea colocarle localmente al repositorio y en qué ubicación guardarlo.

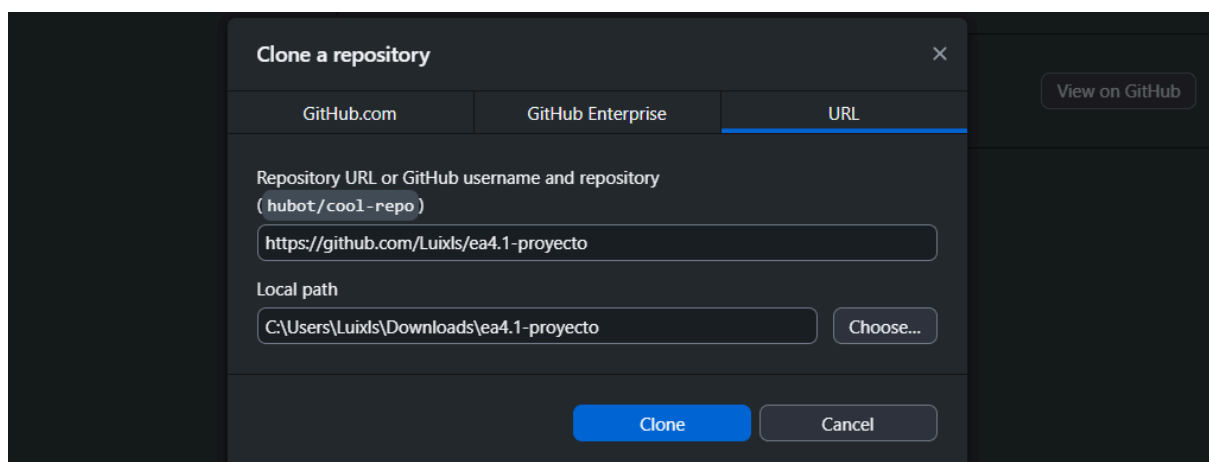


Imagen: Cuadro de confirmación de GitHub Desktop.

Paso 2: Ejecutar el Servidor Local MySQL con XAMPP

Procedemos a ejecutar el servidor para la base de datos utilizando el programa XAMPP, **se recomienda ejecutar XAMPP como administrador** para minimizar incompatibilidades. Una vez la ventana de XAMPP está presente, podemos iniciar el servidor apretando el botón “Start” para Apache y MySQL.

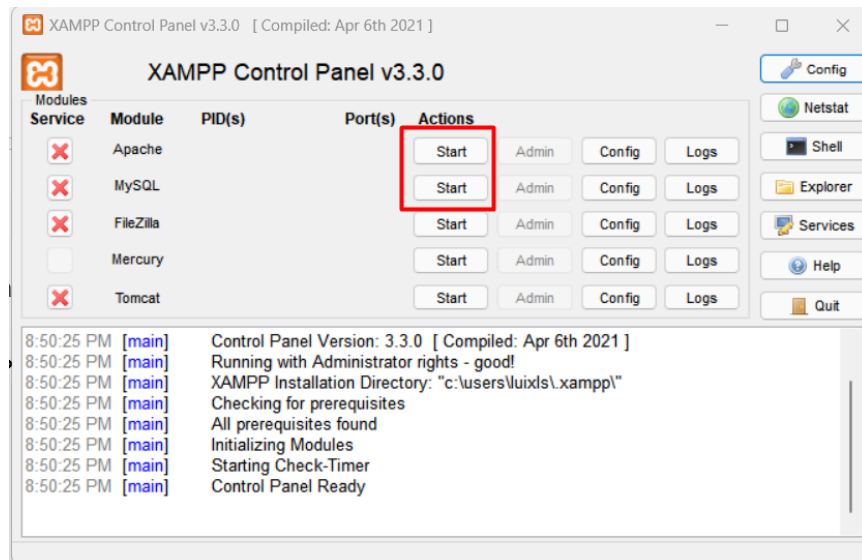


Imagen: Ventana de XAMPP, con los botones para iniciar el servidor resaltados.

Paso 3: Crear la Base de Datos MySQL Local con MySQL Workbench

Con el servidor MySQL ejecutándose, se procede a abrir MySQL Workbench y se establece la conexión al servidor local.

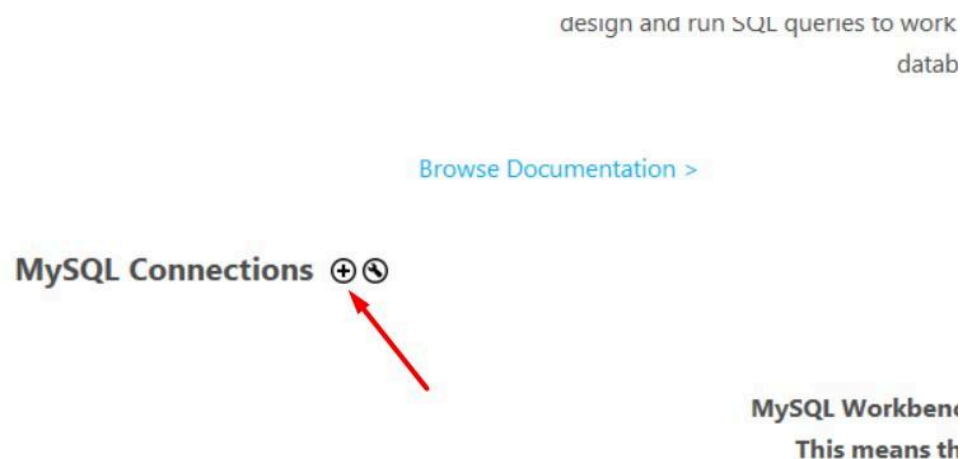


Imagen: Dentro de MySQL Workbench, botón para agregar conexión.

Se recomienda al usuario dejar los valores por defecto al momento de realizar la conexión, para que no haga falta editar archivos dentro del directorio del SGU.

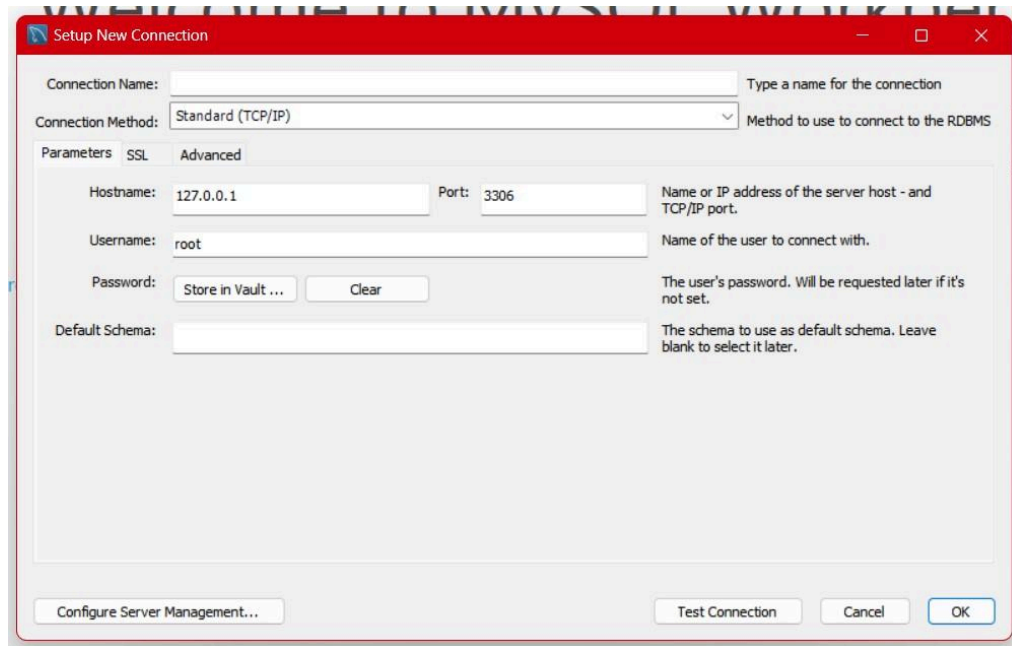


Imagen: Dentro de MySQL Workbench, ventana de configuración de conexión.

Una vez la conexión se establece, procedemos a abrir el script SQL proveído en la carpeta de “Recursos SGU” dentro del directorio del proyecto. Para ello, seleccionamos la opción de “Open SQL Script...” dentro de MySQL Workbench.

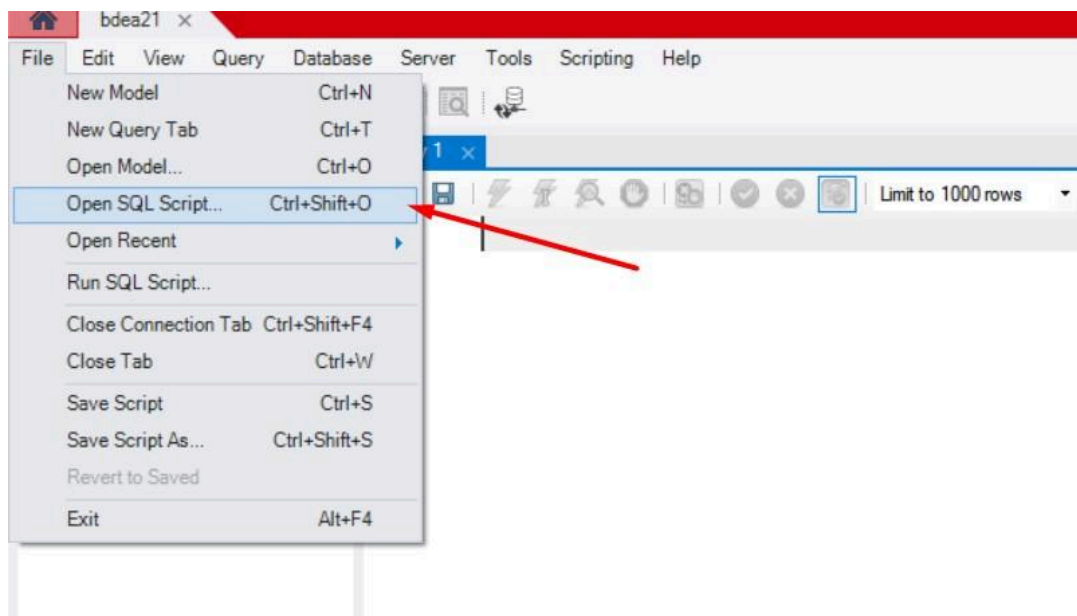


Imagen: Dentro de MySQL Workbench, opción para abrir un archivo script SQL.

Siguiente, simplemente se debe ejecutar dicho archivo apretando el botón correspondiente dentro de MySQL Workbench. La base de datos local será creada con los parámetros necesarios, e incluirá entradas (datos) de ejemplo pre-existentes.

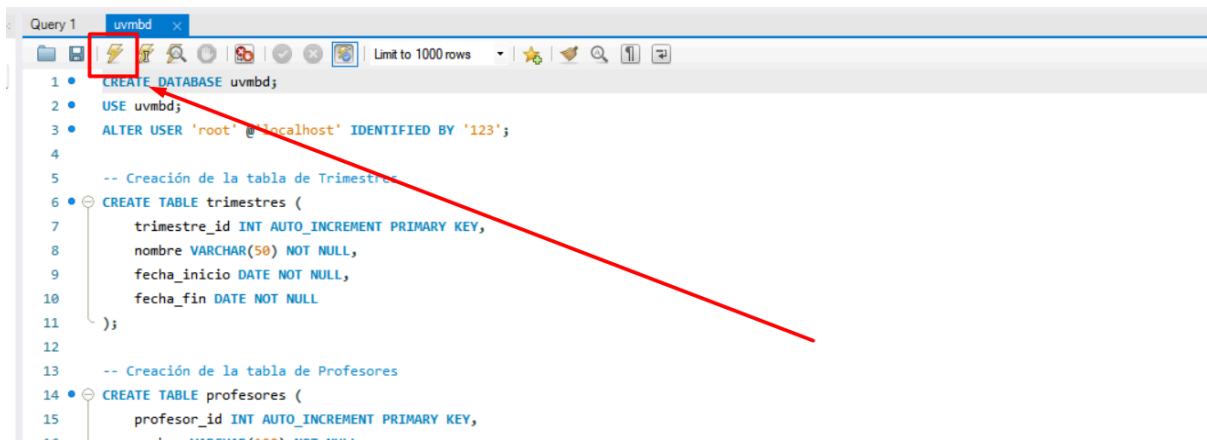


Imagen: Dentro de MySQL Workbench, botón para ejecutar el script con la creación de la base de datos local.

Ejecutar este script **asignará una contraseña a la base de datos**. En este ejemplo, será “123” y se trabajará con esta contraseña por defecto para proseguir.

Paso 4: Crear el archivo “.env”

Al culminar el anterior paso, se debe de crear un archivo llamado “**datos.env**” en la carpeta “config” del proyecto, donde serán guardadas las credenciales para obtener acceso a la base de datos local y también para crear la cuenta “admin” predeterminada del sistema.

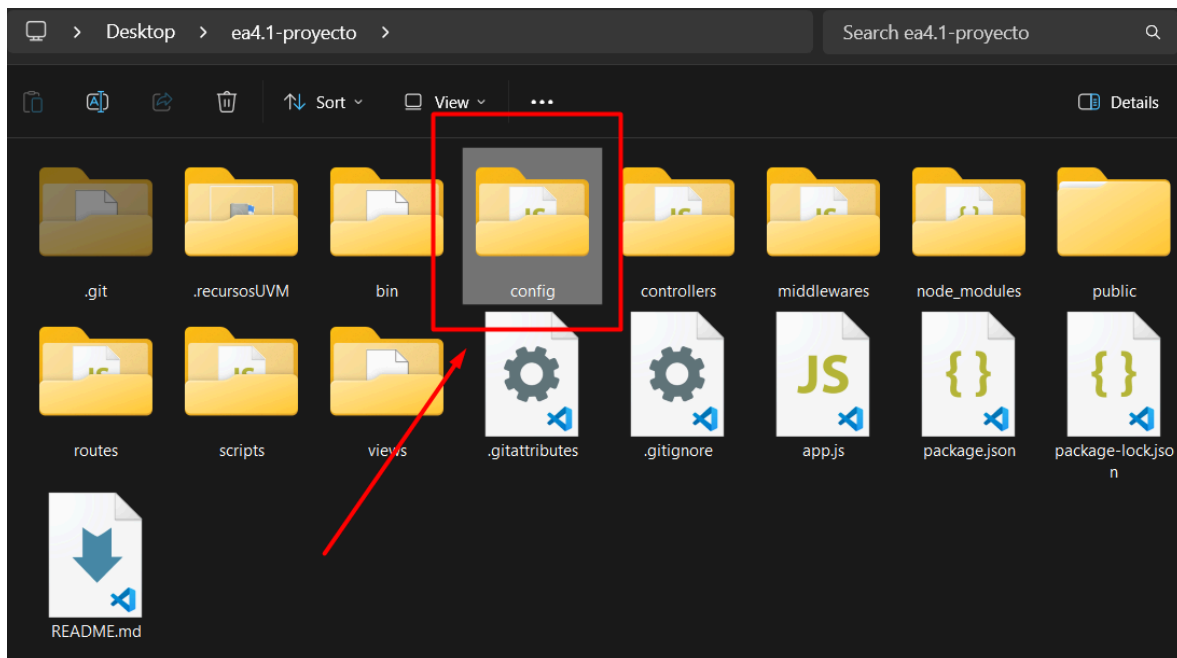


Imagen: Directorio del proyecto, la carpeta “config” resaltada, dentro de ella se crea el archivo “**datos.env**”

Una vez creado el archivo, se puede ejecutar VSCode y arrastrar dicho archivo para poder editarlo. Ya que se están trabajando los valores por defecto de la BD, si el usuario lo desea, puede copiar y pegar el texto a continuación al archivo “**datos.env**”

Archivo datos.env

Este es el archivo que el usuario debe crear en su máquina local, y ajustar a sus necesidades.

El usuario puede cambiar contraseñas, puertos, nombre de usuario, etc. según vea conveniente.

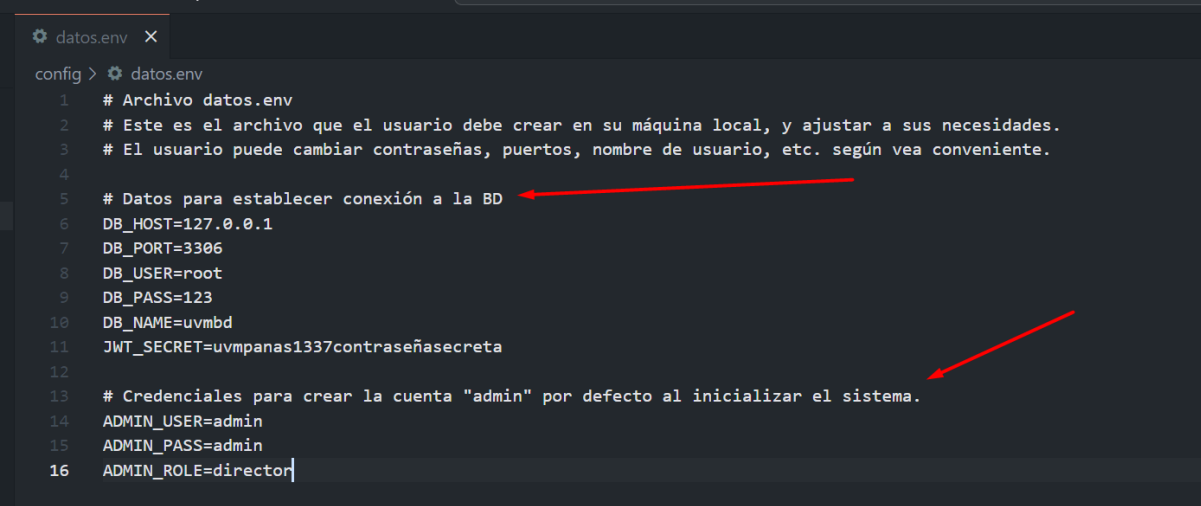
Datos para establecer conexión a la BD

```
DB_HOST=127.0.0.1
DB_PORT=3306
DB_USER=root
DB_PASS=123
DB_NAME=uvmbd
JWT_SECRET=uvmpañas1337contraseñasecreta
```

Credenciales para crear la cuenta "admin" por defecto al inicializar el sistema.

```
ADMIN_USER=admin
ADMIN_PASS=admin
ADMIN_ROLE=director
```

Texto en negritas: Las credenciales por defecto a ser introducidas en el archivo **datos.env**.



```
datos.env
config > datos.env
1  # Archivo datos.env
2  # Este es el archivo que el usuario debe crear en su máquina local, y ajustar a sus necesidades.
3  # El usuario puede cambiar contraseñas, puertos, nombre de usuario, etc. según vea conveniente.
4
5  # Datos para establecer conexión a la BD
6  DB_HOST=127.0.0.1
7  DB_PORT=3306
8  DB_USER=root
9  DB_PASS=123
10 DB_NAME=uvmbd
11 JWT_SECRET=uvmpañas1337contraseñasecreta
12
13 # Credenciales para crear la cuenta "admin" por defecto al inicializar el sistema.
14 ADMIN_USER=admin
15 ADMIN_PASS=admin
16 ADMIN_ROLE=director
```

Imagen: Dentro de VSCode, visualizando cómo debería verse el archivo “**datos.env**”

Paso 5: Ejecutar el Servidor del SGU

Para el siguiente paso, se debe ejecutar el archivo que corresponde al servidor del SGU, localizado en el directorio del proyecto. Para ello, a través de VSCode, debemos elegir la opción de abrir una nueva carpeta, la cual será el directorio del SGU.

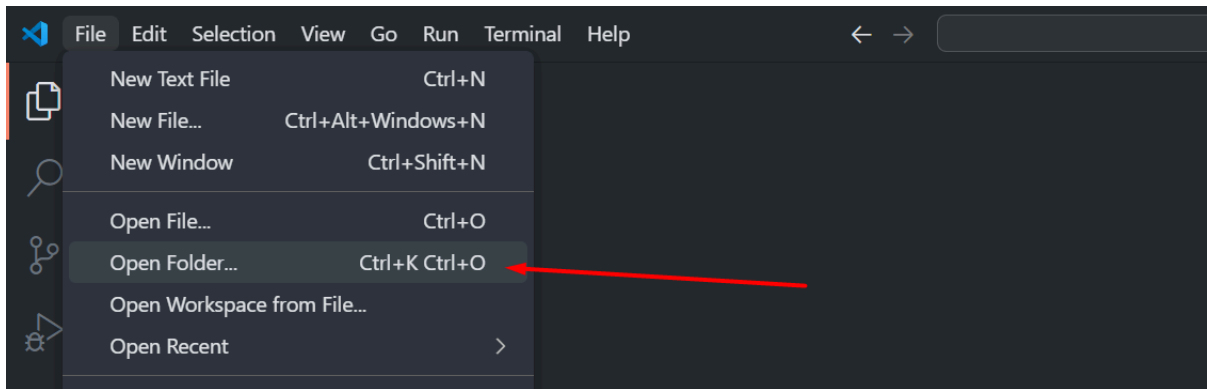


Imagen: Dentro de VSCode, seleccionando la opción de abrir carpeta.

Por último, solo es cuestión de utilizar el comando “***npx nodemon app.js***” para ejecutar el servidor local del SGU a través de la consola de VSCode. Un mensaje de conexión exitosa será mostrado cuando el servidor arranque.

```

27 // Automatizar creación de admin
28 crearAdminPredeterminado().then(() => {
29   // Para iniciar el servidor --> "npx nodemon app.js"
30   app.listen(3000, () => {
31     console.log("Servidor corriendo en el puerto 3000");
32   });
33 });
34

```

DEBUG CONSOLE **TERMINAL** PORTS OUTPUT PROBLEMS

```

PS C:\Users\Luixls\Desktop\ea4.1-proyecto> npx nodemon app.js
[nodemon] 3.1.0
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node app.js`
Servidor corriendo en el puerto 3000
CONEXIÓN AL POOL DE LAS CONEXIONES BD MYSQL EXITOSA
Usuario admin creado exitosamente

```

Imagen: Dentro de VSCode, ejecutando el comando para iniciar el servidor local del SGU.

Paso 6: Ejecutar Postman

El último paso solo consta de ejecutar la aplicación Postman, para empezar a interactuar con el SGU.

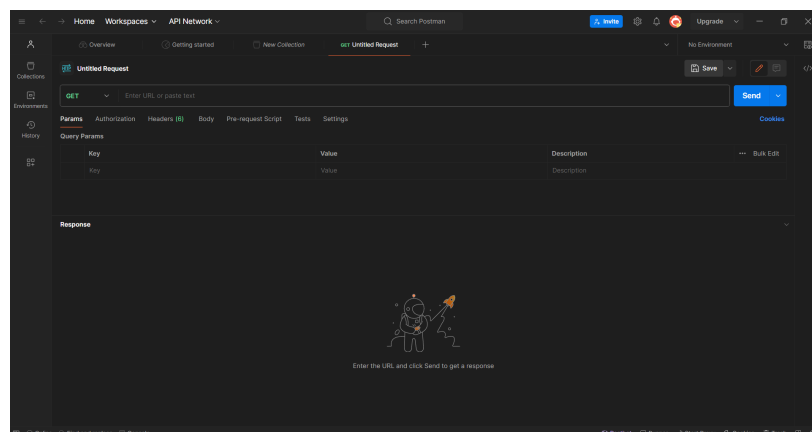


Imagen: Ventana principal de Postman, donde se ejecutarán los protocolos HTTP.

Uso del Sistema

Una vez el SGU ha sido desplegado en una máquina local, se puede proceder a utilizarlo o interactuar con él a través de la herramienta Postman, gracias a los endpoints desarrollados que están al alcance para recibir comandos HTTP (GET, POST, PUT, DELETE). Solo es cuestión de utilizar las URLs correspondientes a cada operación que el usuario final quiera realizar.

Consideraciones

Esta guía de uso de sistema supondrá que el usuario final está ejecutando su servidor local en el puerto por defecto (puerto 3000). Las instrucciones siguientes especificarán las URLs y el comando HTTP a ser utilizado para cada operación y, de ser necesario, el contenido del cuerpo que un comando HTTP debe de poseer. Cabe destacar que las URL siguen un formato parecido entre ellas para facilitar al usuario utilizar el SGU. Esta guía de uso presentará las URL de la siguiente manera:

`http://localhost:3000/eventos/[IDProfesor]/[FechaAAAA-MM-DD]`

Donde **los caracteres en negrita** deben ser reemplazados por los valores correspondientes según los comandos que el usuario desee realizar. De la misma forma, en caso de que un comando requiera de un cuerpo, **los caracteres en negrita** señalan qué debe ser reemplazado, como el siguiente ejemplo:

```
{  
  "nombre": "Nombre del trimestre",  
  "fecha_inicio": "AAAA-MM-DD",  
  "fecha_fin": "AAAA-MM-DD"  
}
```

Es importante el correcto uso de las URLs y comandos HTTP para evitar confusiones y problemas al momento de utilizar el SGU.

Ciertas acciones o comandos requieren de un “token” que el usuario puede obtener al hacer el login. Dicho token debe ser incluido en el comando enviado por Postman, en la pestaña “Headers”; donde la “Key” debe ser “**auth**” y “Value” debe ser el token del usuario.

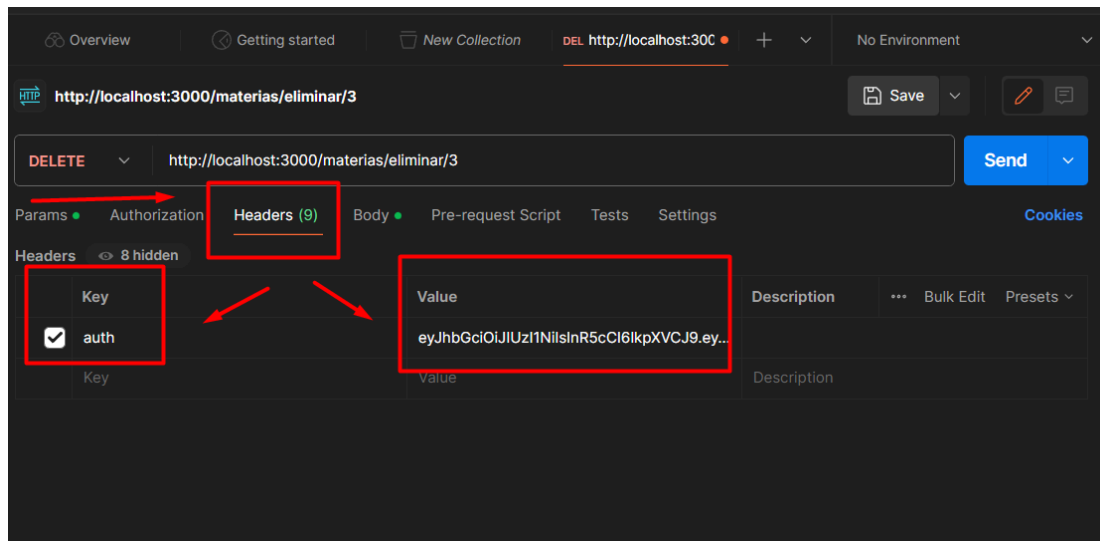


Imagen: En Postman, un ejemplo de cómo debe ir la sección “Headers” para introducir el token del usuario y obtener acceso a funcionalidades que requieren permisos.

Restricciones

Como se ha mencionado anteriormente, ciertas acciones están restringidas solo para aquellos usuarios con el nivel de permiso correspondiente. Por lo general, el sistema sigue el siguiente formato de restricciones:

- **Listar (GET):** No se necesitan permisos para listar Trimestres y Materias (ya que son de acceso público), pero para listar el resto de los datos, se necesita una cuenta de cualquier rol (Estudiante, Profesor o Director).
- **Agregar (POST):** Cuentas Director pueden agregar registros sin restricciones, mientras que cuentas Profesor solo pueden agregar eventos no globales.
- **Editar (PUT):** Solo cuentas tipo Director pueden editar registros sin restricciones, mientras que cuentas tipo Profesor solo pueden editar eventos no globales.
- **Eliminar (DELETE):** Solo cuentas tipo Director pueden eliminar datos del SGU.

Estas restricciones se aplican para los datos de las tablas de profesores, materias, secciones eventos y trimestres.

Registrar un Usuario

El SGU permite el registro de nuevos usuarios para crear cuentas de tipo “Estudiante” sin restricciones. Sin embargo, solo un director puede crear cuentas de tipo “Profesor” o “Director” (por ello es importante la creación de la cuenta “admin”). Se debe hacer un comando **POST** a la URL <http://localhost:3000/usuarios/registro>

```
{  
  "usuario": "nombre de usuario",  
  "contraseña": "contraseña aquí",  
  "rol": "estudiante/profesor/director"  
}
```

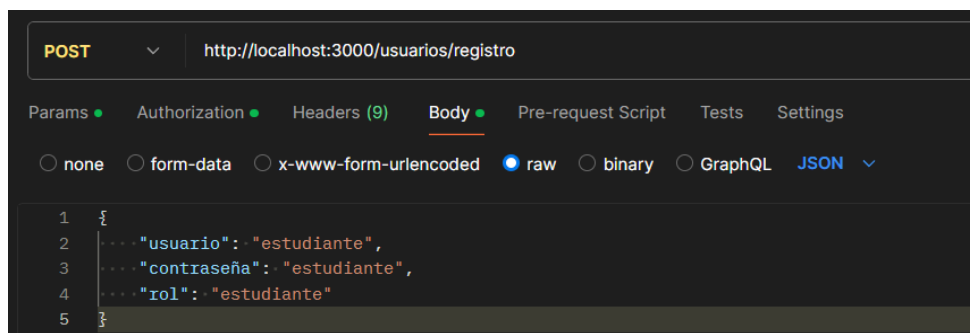


Imagen: En Postman, registrando un usuario de tipo “estudiante”.

Login

Realizar un login con una cuenta creada permite que el SGU provea un “token”, la cual es una cadena de texto que el usuario debe guardar e introducir en sus comandos de la manera mencionada en [consideraciones](#) para obtener acceso a las acciones que requieren autenticación.

```
{  
  "usuario": "nombre de usuario",  
  "contraseña": "contraseña aquí"  
}
```

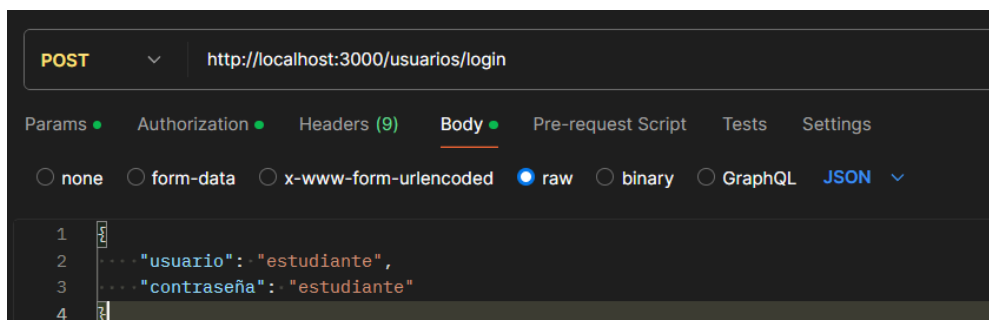
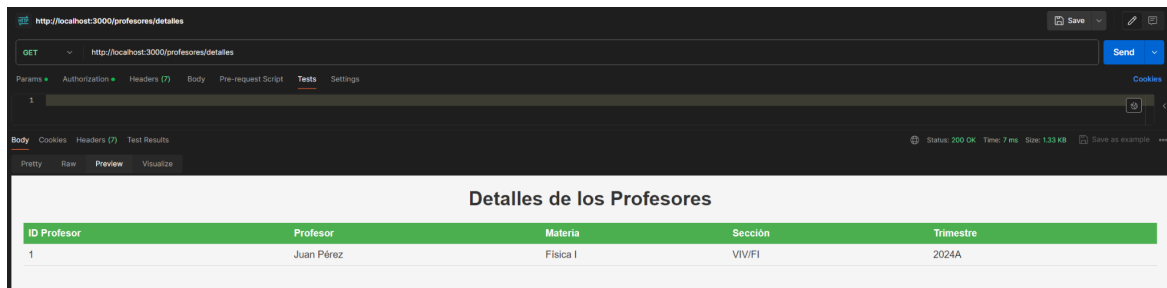


Imagen: En Postman, haciendo login para obtener un token de autenticación.

Visualizar Materias y Secciones de un Profesor

Se puede realizar a través de un comando GET a la siguiente URL:

<http://localhost:3000/profesores/detalles>



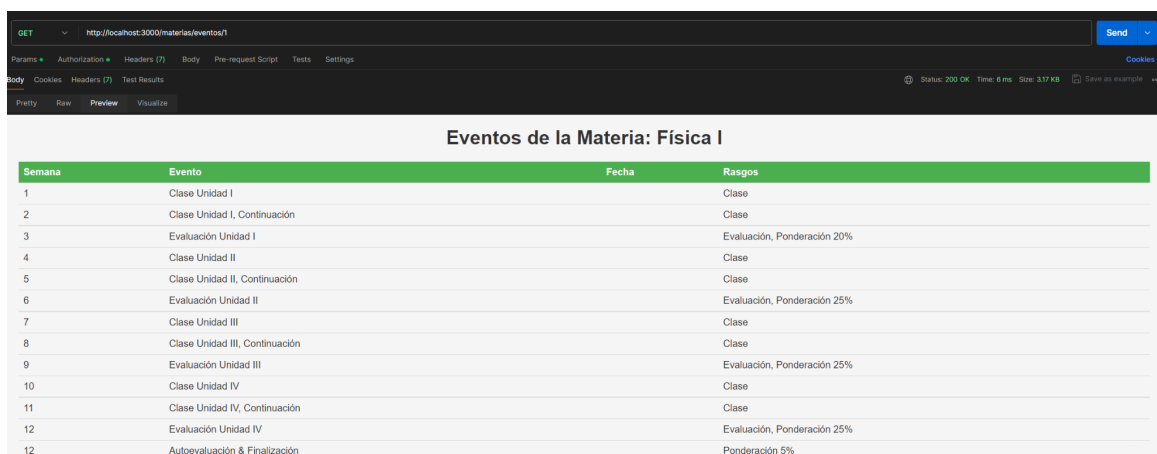
ID Profesor	Profesor	Materia	Sección	Trimestre
1	Juan Pérez	Física I	VIV/FI	2024A

Imagen: Visualización de las materias y secciones de un profesor

Visualizar Eventos de una Materia

e puede realizar a través de un comando GET a la siguiente URL:

http://localhost:3000/materias/eventos/***# de ID aquí***



Semana	Evento	Fecha	Rasgos
1	Clase Unidad I		Clase
2	Clase Unidad I, Continuación		Clase
3	Evaluación Unidad I		Evaluación, Ponderación 20%
4	Clase Unidad II		Clase
5	Clase Unidad II, Continuación		Clase
6	Evaluación Unidad II		Evaluación, Ponderación 25%
7	Clase Unidad III		Clase
8	Clase Unidad III, Continuación		Clase
9	Evaluación Unidad III		Evaluación, Ponderación 25%
10	Clase Unidad IV		Clase
11	Clase Unidad IV, Continuación		Clase
12	Evaluación Unidad IV		Evaluación, Ponderación 25%
12	Autoevaluación & Finalización		Ponderación 5%

Imagen: Visualización de los eventos de una materia.

Visualizar Eventos en 2 Semanas para Profesores

Se puede realizar a través de un comando GET a la siguiente URL:

http://localhost:3000/eventos/profesores/***FECHA AAAA-MM-DD***



Profesor	Evento	Fecha	Rasgos
Juan Pérez	Clase Unidad I	Por definir	Clase
Juan Pérez	Clase Unidad I, Continuación	Por definir	Clase
	Bienvenida al Trimestre	8 de enero de 2024	Bienvenida al 2024A

Imagen: Visualización de los eventos pautados dentro de 2 semanas desde la fecha consultada.

Visualizar Eventos de una Sección

Se puede realizar a través de un comando GET a la siguiente URL:

http://localhost:3000/secciones/eventos/***# de ID aquí***



Evento	Número de Semana	Fecha	Rasgos	Global
Bienvenida al Trimestre		8 de enero de 2024	Bienvenida al 2024A	SI
Corte de Notas 1		2 de febrero de 2024	Corte de Notas	SI
Día Feriado - Aniversario de la Facultad de Ingeniería		22 de marzo de 2024	Feriado	SI
Clase Unidad I	1		Clase	No
Clase Unidad I, Continuación	2		Clase	No
Evaluación Unidad I	3		Evaluación, Ponderación 20%	No
Clase Unidad II	4		Clase	No
Clase Unidad II, Continuación	5		Clase	No
Evaluación Unidad II	6		Evaluación, Ponderación 25%	No
Clase Unidad III	7		Clase	No

Imagen: Visualización de los eventos de una sección.

Asignar Eventos de una Materia a una Sección

Se puede realizar a través de un comando POST a la siguiente URL:

<http://localhost:3000/eventos/asignar>

Y el cuerpo de la solicitud puede seguir la siguiente plantilla:

```
{
  "evento_id": [ID del evento, ID del evento...],
  "seccion_id": ID de la sección a asignar los eventos,
  "materia_id": ID de la materia (si se desea) a asignar los eventos
}
```

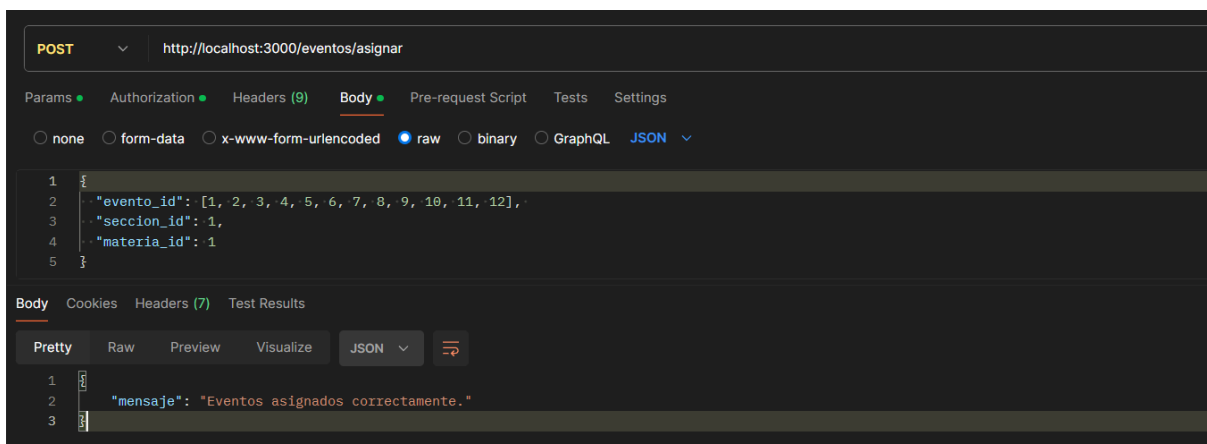


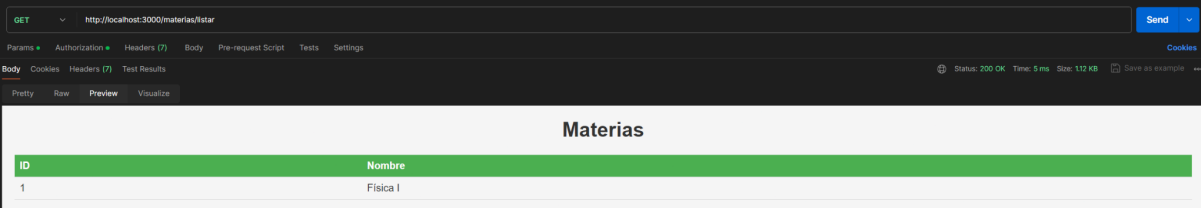
Imagen: Asignando eventos a una sección y materia.

Manejo de Registros y Datos

Listar (Ver) Registros - GET

Para listar ver las entradas que el SGU posee sobre un ámbito en específico, se debe realizar un comando **GET** a la URL correspondiente según lo que se quiera listar.

- **Listar Profesores:** <http://localhost:3000/profesores/listar>
- **Listar Materias:** <http://localhost:3000/materias/listar>
- **Listar Secciones:** <http://localhost:3000/secciones/listar>
- **Listar Eventos:** <http://localhost:3000/eventos/listar>
- **Listar Trimestres:** <http://localhost:3000/trimestres/listar>



ID	Nombre
1	Física I

Imagen: Listando materias

Agregar un Registro - POST

Para agregar un registro al SGU, se debe utilizar un comando **POST** a la URL correspondiente según las necesidades del usuario. Dicho comando debe tener un cuerpo en formato JSON con los datos necesarios correspondientes, especificados por el usuario según lo que se quiere agregar.

- **Agregar Profesores:** <http://localhost:3000/profesores/agregar>

```
{  
  "nombre": "Nombre Apellido"  
}
```
- **Agregar Materias:** <http://localhost:3000/materias/agregar>

```
{  
  "nombre": "Nombre de Materia"  
}
```

- **Agregar Secciones:** <http://localhost:3000/secciones/agregar>

```
{  
  "nombre": "Nombre de la Sección",  
  "profesor_id": ID del profesor,  
  "materia_id": ID de la materia,  
  "trimestre_id": ID del trimestre  
}
```

- **Agregar Eventos No Globales:** <http://localhost:3000/eventos/agregar>

```
{  
  "nombre": "Nombre del Evento",  
  "numero_semana": Número de la semana,  
  "fecha": null (o si se desea, puede asignar fecha AAAA-MM-DD),  
  "rasgos": "Descripción del evento",  
  "seccion_id": ID de la sección,  
  "materia_id": ID de la materia  
}
```

- **Agregar Eventos Globales:** <http://localhost:3000/eventos/agregar>

```
{  
  "nombre": "Nombre del Evento",  
  "numero_semana": null,  
  "fecha": "AAAA-MM-DD",  
  "rasgos": "Descripción del evento",  
  "seccion_id": null,  
  "materia_id": null,  
  "es_global": 1  
}
```

- **Agregar Trimestres:** <http://localhost:3000/trimestres/agregar>

```
{  
  "nombre": "Nombre del trimestre",  
  "fecha_inicio": "AAAA-MM-DD",  
  "fecha_fin": "AAAA-MM-DD"  
}
```

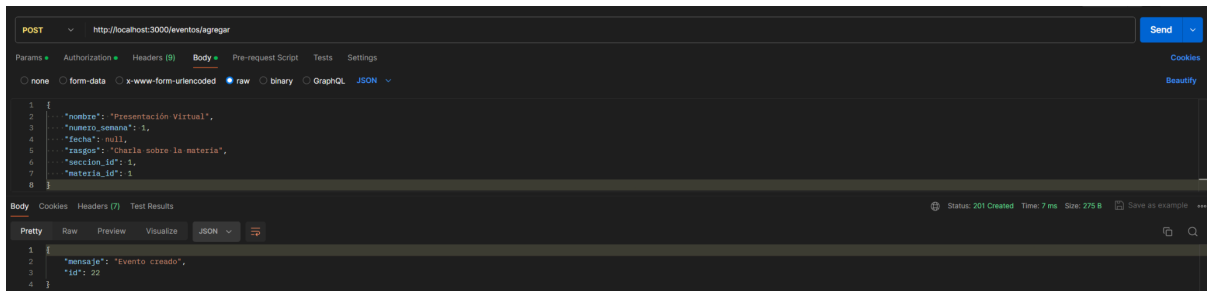


Imagen: Agregando un evento no-global.

Editar un Registro - PUT

Para editar un registro al SGU, se debe utilizar un comando **PUT** a la URL correspondiente segun las necesidades previamente especificadas por el usuario. Dicho comando debe tener un cuerpo en formato JSON con los datos nuevos que sobrescribirán los anteriores, especificados por el usuario.

- **Editar Profesor:** http://localhost:3000/profesores/editar/***# de ID aquí***

```
{
  "nombre": "Nuevo Nombre"
}
```

- **Editar Materia:** http://localhost:3000/materias/editar/***# de ID aquí***

```
{
  "nombre": "Nuevo Nombre"
}
```

- **Editar Sección:** http://localhost:3000/secciones/editar/***# de ID aquí***

```
{
  "nombre": "Nuevo nombre",
  "profesor_id": "Nuevo ID profesor",
  "materia_id": "Nuevo ID profesor",
  "trimestre_id": "Nuevo ID trimestre"
}
```


- **Editar Evento No-Global:** http://localhost:3000/eventos/editar/***# de ID aquí***

```
{
  "nombre": "Nuevo nombre",
  "numero_semana": Nuevo número de semana,
  "fecha": null,
  "rasgos": "Nueva descripción",
  "seccion_id": Nueva ID de sección,
  "materia_id": Nueva ID de materia
}
```

- **Editar Evento Global:** http://localhost:3000/eventos/editar/***# de ID aquí***

```
{
  "nombre": "Nuevo Nombre",
  "numero_semana": null,
  "fecha": "AAAA-MM-DD",
  "rasgos": "Nueva descripción",
  "seccion_id": null,
  "materia_id": null,
  "es_global": 1
}
```

- **Editar Trimestre:** http://localhost:3000/trimestres/editar/***# de ID aquí***

```
{
  "nombre": "Nuevo Nombre del trimestre",
  "fecha_inicio": "AAAA-MM-DD",
  "fecha_fin": "AAAA-MM-DD"
}
```

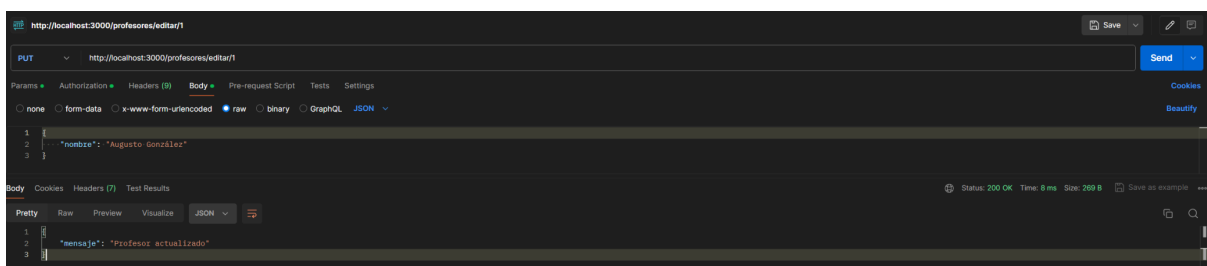


Imagen: Editando un profesor

Eliminar un Registro - DELETE

Para eliminar registros del SGU, se debe ejecutar un comando **DELETE** a la URL correspondiente según la necesidad del usuario.

- **Eliminar Profesor:** http://localhost:3000/profesores/eliminar/***# de ID aquí***
- **Eliminar Materia:** http://localhost:3000/materias/eliminar/***# de ID aquí***
- **Eliminar Sección:** http://localhost:3000/secciones/eliminar/***# de ID aquí***
- **Eliminar Evento:** http://localhost:3000/eventos/eliminar/***# de ID aquí***
- **Eliminar Trimestre:** http://localhost:3000/trimestres/eliminar/***# de ID aquí***

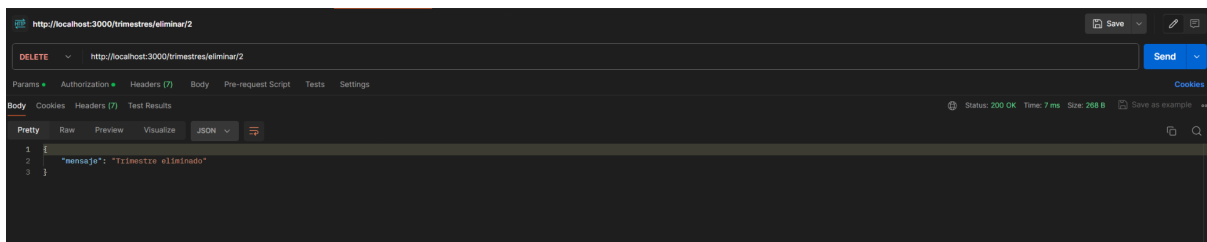


Imagen: Eliminando un trimestre

Créditos y Comentarios de Desarrollador

Diario del Proyecto

1. Se clonó el proyecto de la ea3.1.
2. Se eliminó el esquema de la BD anterior, ya que se decidió trabajar desde cero. Se creó un nuevo esquema para la BD.
3. Se movieron los archivos de “config” a una nueva carpeta. De igual manera se trabajó un nuevo archivo aplicación “app.js”.
4. A continuación, se desarrollaron los enrutadores y controladores necesarios para realizar las operaciones CRUD básicas en todas las tablas correspondientes.
5. Se desarrolló la funcionalidad de registro y login para usuarios.
6. Por siguiente, se implementó un script que crea una cuenta “admin” automáticamente cuando se inicia el proyecto por primera vez.
7. Se creó la funcionalidad de validación de tokens dependiendo del tipo de usuario (rol de usuario).
8. A continuación se crearon validaciones de verificación de rol a través de tokens para limitar o permitir distintos comandos dependiendo del nivel de permisos.
9. Se desarrolló la funcionalidad de validar entradas de datos por parte del usuario (sanitización).
10. Se ajustó la funcionalidad para que sean los enrutadores quienes brinden las respuestas al usuario en comandos “GET”.
11. Por siguiente, se procedió a crear las vistas EJS para listar datos.
12. Se procedió a realizar las operaciones más avanzadas para lograr el enunciado, como listar profesores con sus materias y secciones correspondientes, mostrar eventos próximos en un plazo de 2 semanas, y más.
13. Se aplicaron varios arreglos de formato y lógica mientras el desarrollo de lo anterior seguía en curso.

Comentarios de Desarrollo

Lo más importante a destacar es la decisión de decidir trabajar desde cero el proyecto; solo se tomaron los proyectos anteriores en cuenta para facilitar el desarrollo. La razón de trabajar de cero radica en que se cambió el esquema de la base de datos, y un comienzo fresco fue considerado como el paso más lógico para mantener la cohesión del proyecto.

Y efectivamente, se notó el cambio al momento de desarrollar los principios del proyecto (como las operaciones CRUD) debido a que en vez de buscar código viejo desactualizado, simplemente se desarrolló de nuevo.

Por otro lado, la experiencia de desarrollo de los proyectos anteriores facilitaron el hecho de desarrollar desde cero, ya que los pasos a tomar estaban más claros y se sintieron más sencillos que antes. De igual manera las consultas SQL resultaron más coherentes gracias al nuevo esquema.

Para culminar, podemos nombrar los procesos o partes desarrolladas que tomaron más esfuerzo que las anteriores:

- **Asignar Actividades desde una materia/sección a otra materia/sección:** Se consideró más difícil este punto a comparación de otros objetivos, debido a que el equipo no tenía una respuesta exacta a lo que se quiere lograr o cómo. Después de una lluvia de ideas, se decidió que la forma más lógica de lograr esto fuera a través de “copiar” entradas de eventos y “pegar” dichas entradas en una nueva materia/sección.
- **Mostrar próximos eventos dentro de 2 semanas:** Esto resultó más complicado que su implementación en los proyectos anteriores, debido a que el nuevo esquema de la BD contenía muchísima más información en las entradas de “eventos” que los esquemas anteriores. Las dificultades a destacar fueron la correcta visualización de los eventos, y el cálculo de fecha/número de semana.
- **Trabajar con fechas:** Más específicamente, lograr el cálculo de fechas a través del número de semana y viceversa. A diferencia del proyecto anterior, utilizamos el módulo de Node.js llamado “moment” el cual resultó ser mejor elección que el anterior “date-fns”.

Créditos

Resaltamos el trabajo en equipo de los integrantes desarrolladores del proyecto. El repositorio puede ser accedido a través de la siguiente URL a GitHub:

<https://github.com/Luixls/ea4.1-proyecto>

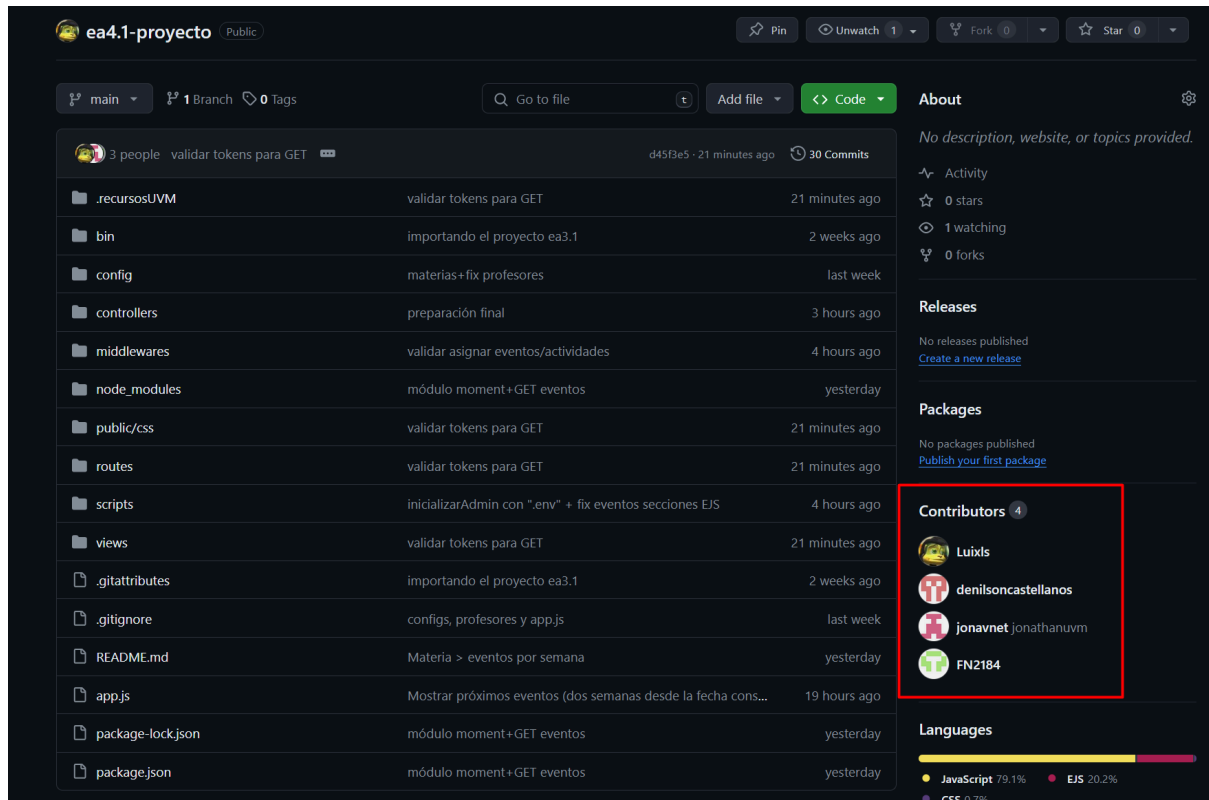


Imagen: Repositorio de GitHub, con los integrantes resaltados.

A continuación, se nombran los integrantes del equipo con su nombre de usuario correspondiente de GitHub:

@jonathanuvm - Jonathan Antonio Arellano Márquez C.I: 24.190.278

@FN2184 - José Enmanuel Cardoza Ferrer C.I: 31.590.138

@Luixls - Luis Fernando Araujo Giardinella C.I. 26.482.894

@denilsoncastellanos - Denilson Eduardo Castellanos Garcia C.I. 29.694.566

Por siguiente, los commits del repositorio del proyecto denotan la participación y/o distribución de las responsabilidades de los integrantes, y de la misma manera denotan cuando una parte del proyecto fue elaborada en conjunto de dos o más integrantes. El historial de commits puede ser accedido a través de la siguiente URL a GitHub:

<https://github.com/Luixls/ea4.1-proyecto/commits/main/>



Imagen: Gráfico de participación de los integrantes.

Y para culminar, el video explicativo del proyecto puede ser accedido a través de la siguiente URL a youtube:

<https://www.youtube.com/watch?v=jGpClccrA5M>