**Despliegue Local:**

1. ***Clonar el repositorio:***

Para empezar, se debe clonar el repositorio que se encuentra en github. La url es el siguiente: <https://github.com/practicingAGSO/high-risk-list.git>. Por lo que se debe ejecutar lo siguiente en una terminal en la carpeta de su preferencia:

***git clone*** [***https://github.com/practicingAGSO/high-risk-list.git***](https://github.com/practicingAGSO/high-risk-list.git)

Texto

Descripción generada automáticamente

En mi caso lo cloné en mi escritorio. Como en este caso se realizará todo de manera local, cambiamos a la rama develop.

***git checkout origin/develop***

Texto

Descripción generada automáticamente

Luego lo pasas a una rama local.

***git checkout -b develop***

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza baja

1. ***Crear entorno virtual:***

Una vez clonado, abrir la carpeta en Visual Studio Code o el editor de su preferencia. En la terminal crea el entorno virtual de la siguiente manera:

***python -m venv venv***

Texto

Descripción generada automáticamente

De esta manera se crea el entorno virtual “venv”. Luego, activamos el entorno con el siguiente comando:

***venv\Scripts\activate***

Texto

Descripción generada automáticamente

Se puede observar que ya estamos dentro del entorno virtual.

1. ***Instalación de dependencias:***

Para la instalación de las dependencias necesarias se ejecuta el siguiente comando:

***pip install -r requirements.txt***

Luego se puede observar como se van instalando las dependencias.

Texto

Descripción generada automáticamente

1. ***Creación de base de datos local:***

Para este ejercicio, se creó una base de datos de usuarios para la autenticación de las REST APIs. De este modo solo se podrán usar los endpoint si el usuario está autorizado con un token.

De esta manera, ingresamos al SQL Management Studio.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

En mi caso uso Windows Authentication. Luego ejecutamos la siguiente query para la generación de la base de datos y su tabla usuarios.

Primero ejecuta la creación de la tabla:

***CREATE DATABASE VendorDB***

Luego ejecuta el uso de la tabla creada:

***USE VendorDB***

Y por último la creación de la tabla de usuarios para la autenticación:

***CREATE TABLE users (***

***id BIGINT IDENTITY(1,1) PRIMARY KEY,***

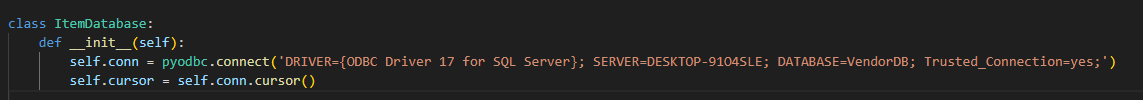
***username NVARCHAR(MAX) NOT NULL,***

***password NVARCHAR(MAX) NOT NULL***

***);***

1. ***Configuración de la bd en la aplicación:***

En el archivo *db\_sql\_server.py* que se encuentra en *src/database/* se puede observar la configuración para la conexión a la base de datos.



Si en caso su gestor de SQL Management Studio tiene conexión con Windows, solo cambie el SERVER por el que tiene.

1. ***Correr la aplicación:***

Para ello, ejecutar el comando

***python index.py***

Captura de pantalla de computadora

Descripción generada automáticamente

De ese modo la aplicación correrá en <http://127.0.0.1:5000>. Así, se podrá probar los Endpoints en archivo de postman exportado.

*high-risk-list.postman\_collection.json.*

**Despliegue en AWS:**

(Despliegue ya realizado) En esta sección solo detallo un poco cómo es que deplegué los componentes a aws.

1. ***Creación de la instancia RDS para SQL Management:***

Para ello, con una cuenta aws, buscamos el servicio de RDS.

***Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente***

Le damos click y luego vamos a Instancias de base de datos.

Texto

Descripción generada automáticamente

Para la creación se presiona el botón “Crear base de datos”.

Imagen que contiene Diagrama

Descripción generada automáticamente

En la configuración inicial las partes importantes son el motor seleccionado.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

También en este caso por ser para pruebas, elegir la capa gratuita.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Y en la configuración de credenciales tener en mente o anotado el usuario y contraseña para gestionar la base de datos.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Luego por esta ocasión le puse acceso público y un grupo de seguridad que me permita luego conectar mi aplicación desplegada a la base de datos.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

De este modo la instancia está creada. Luego de ello ejecuté los scripts que se realizaron en el despliegue local para la creación de la base de datos y tabla usuarios.

1. ***Configuración de la base de datos en la aplicación:***

En este caso, apenas se clona el repositorio ya se encuentra en la rama main, ahí se puede observar la configuración en el archivo *db\_sql\_server.py* que se encuentra en *src/database/*

Texto

Descripción generada automáticamente

Ahí se puede observar la configuración de la base de datos desplegada.

1. ***Creación de instancia EC2:***

En esta ocasión elegí una máquina virtual EC2 para el despliegue de la aplicación. Para ello se busca el servicio en AWS.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Le damos click y en la parte derecha superior se presiona “Lanzar instancias” para crear una nueva máquina virtual.

Imagen que contiene Rectángulo

Descripción generada automáticamente

Elegí el sistema operativo Ubuntu.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Creé un grupo de seguridad que permita el tráfico HTTP y HTTPS.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Además, se genera un par de claves para administrarlo en este caso en putty y se lanza la instancia.

Forma, Rectángulo

Descripción generada automáticamente

Dentro de putty se realizaron las configuraciones necesarias para correr la aplicación con su ip pública. De este modo, la aplicación ya no corre en un entorno local, sino en <http://18.232.129.23> .

Para realizar las pruebas en Postman, se van a la carpeta y editan la variable url.

Captura de pantalla de un celular

Descripción generada automáticamente