## Vicerrectorado Uagrm - ULLOA: EL ESCUDO DE LA UAGRM FUE REGISTRADO EN EL SENAPI COMO MARCA INSTITUCIONAL El Escudo de la Universidad Autónoma Gabriel René Moreno (UAGRM) fue registrado en el Servicio Sistema Experto de Recomendación de Películas

**Docente:**

**Vargas Peña Leonardo**

**Estudiantes:**

* **Mandepora Perez Yerling**
* **Chilimani Cuellar Edson Neil**
* **Ayelen Estevez Segovia**

**Facultad Integral Del Chaco, Universidad Autónoma Gabriel Rene Moreno**

**Ingeniería en Informática**

**Tecnología Web**

**Camiri-Santa Cruz-Bolivia**

**2023**

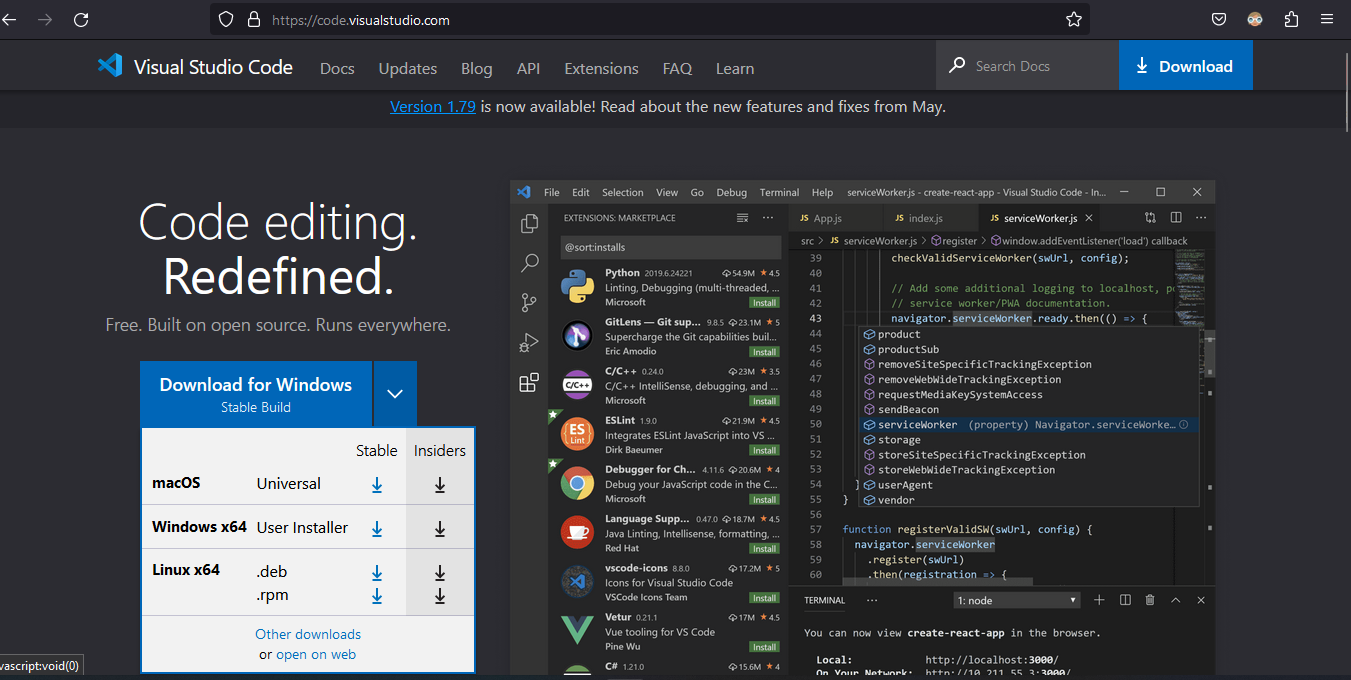
Para el presente proyecto de Recomendación de Películas se utilizará el lenguaje Python con el gestor Sqlite3 y el editor de visual studio code para lograr el objetivo de construir este Sistema Experto. Todo esto se realizará en el sistema operativo de “Windows”

Herramientas que se utilizara:

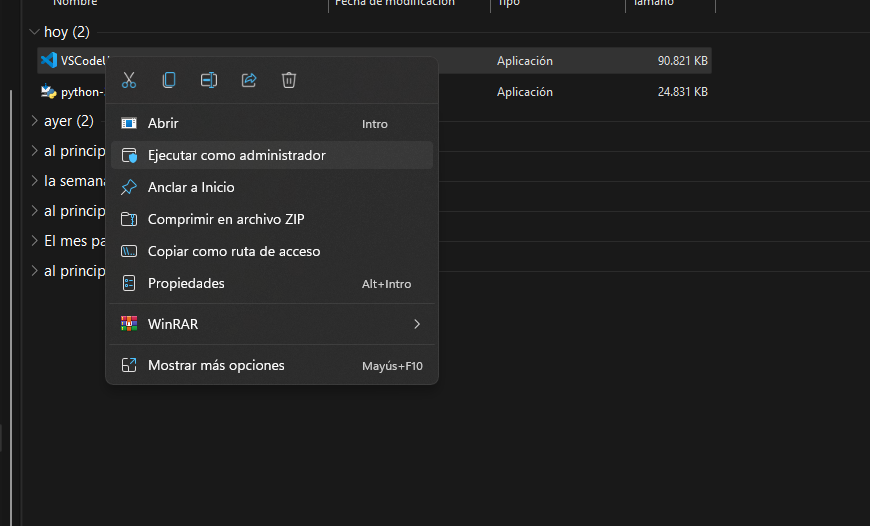
* Visual studio code “editor de texto”.
* Python
* Sqlite3
* Biblioteca de Pyknow
* Biblioteca Pgmpy

**INSTALACION DEL EDITOR DE TEXTO VISUAL STUDIO CODE**

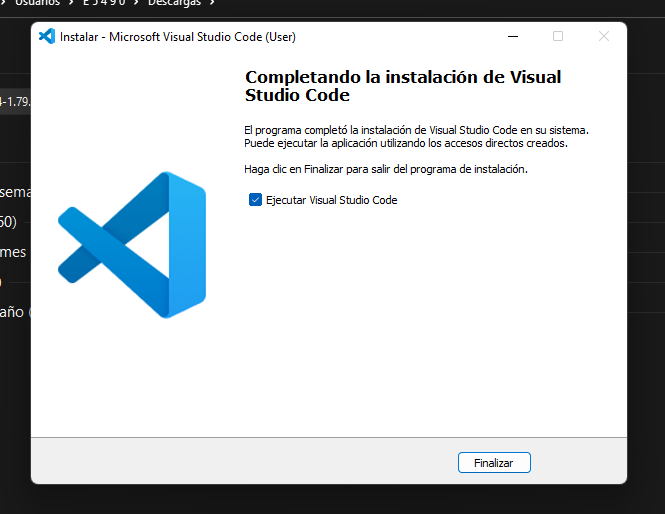
Entramos al sitio oficial de Vscode y le damos descargar según nuestro sistema operativo verificando así también si es de 32 o 64 bits.



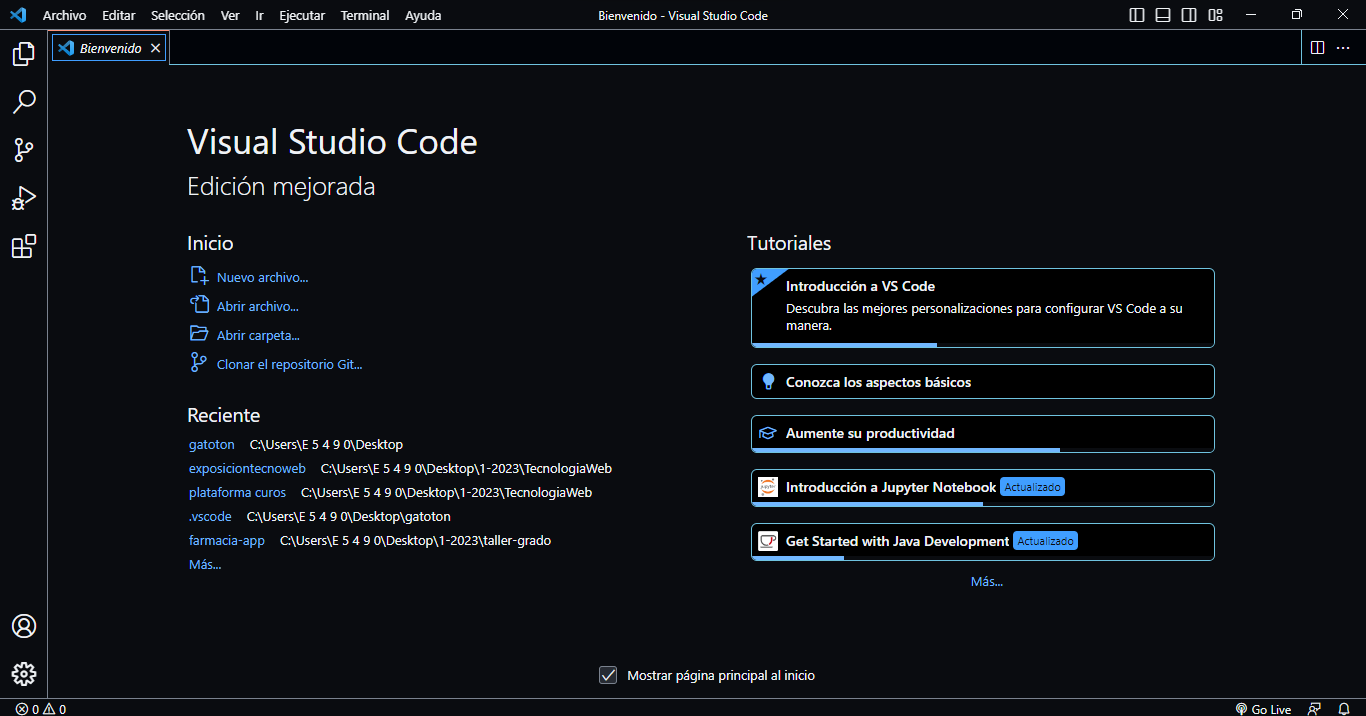
Lo ejecutamos como administrador para que no surjan problemas



Y lo instalamos siguiendo los pasos que nos indique.



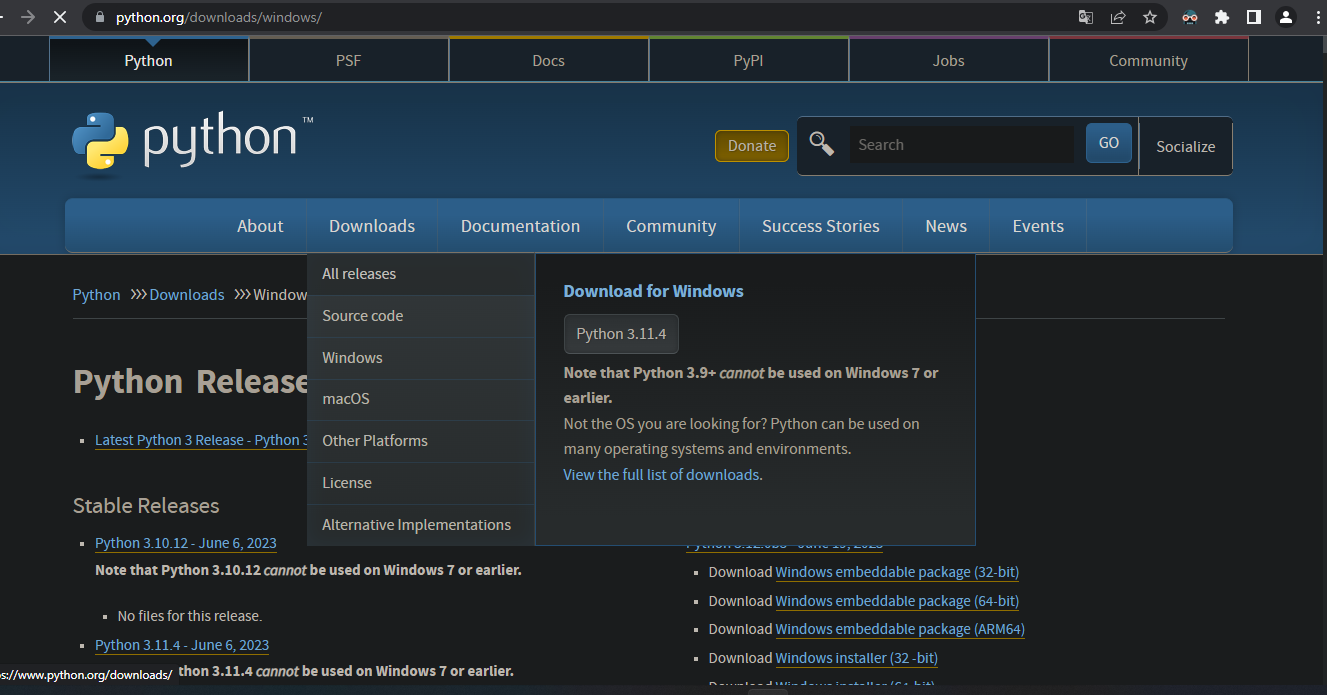
Y ahora todo listo para empezar a utilizarlo



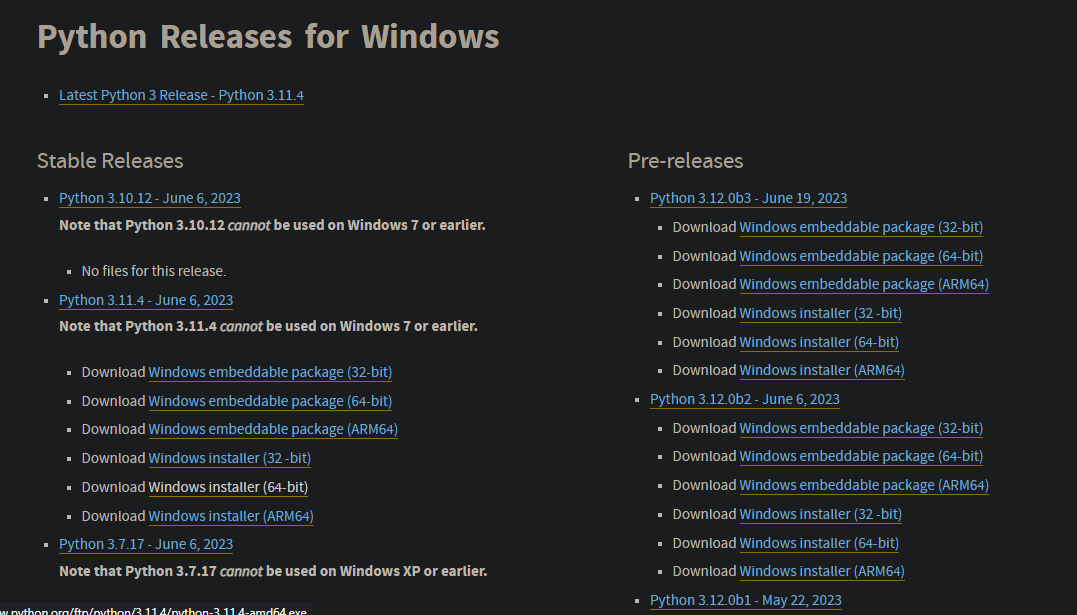
**INSTALACION DE PYTHON**

Instalación de lenguaje de programación en el que se trabajara, en este caso PYTHON

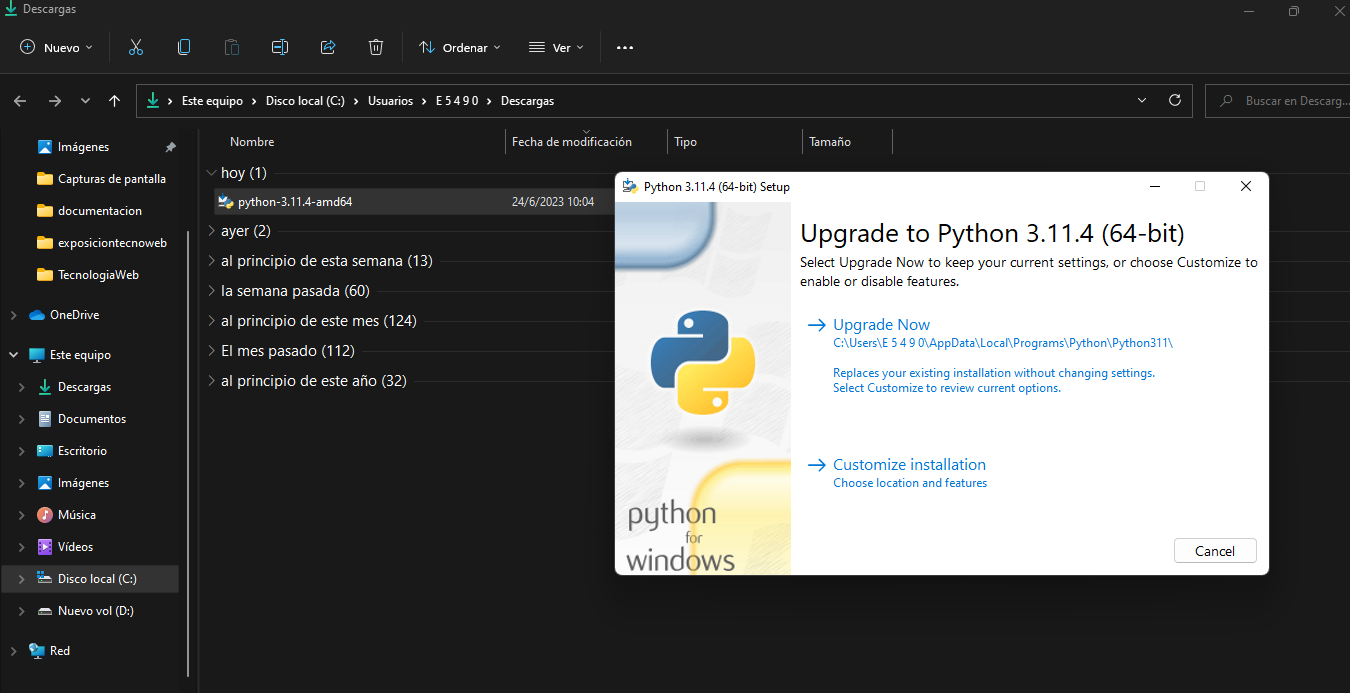
-Se visita la página oficial de Python.



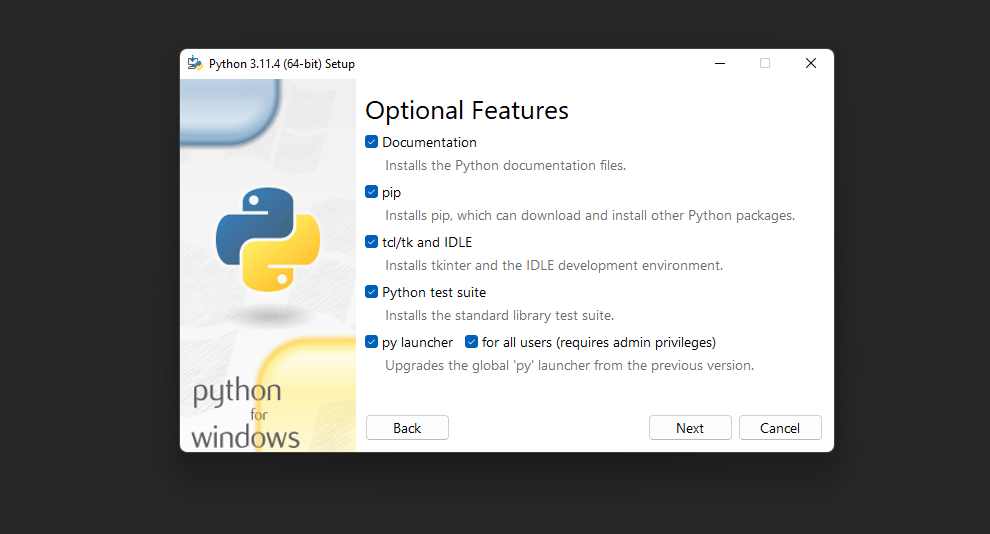
-Se descarga la versión más reciente de Python para evitar problemas de compatibilidad con las dependencias que se utilizara en el proyecto, así mismo tener en cuenta descargar según el sistema operativo del equipo(32bits o 64 bits).



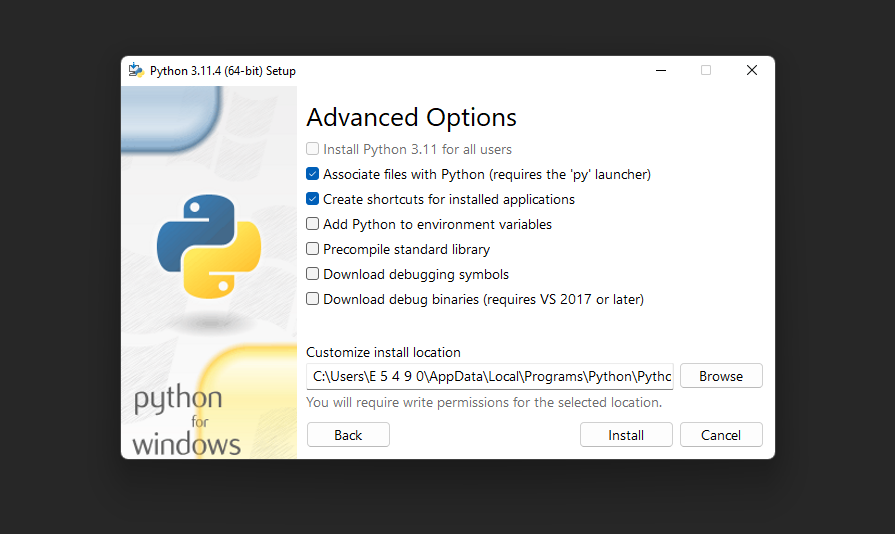
Lo buscamos en nuestros archivo el instalador de Python que descargamos y hacemos doble click en el para ejecutarlo e instalar el software.



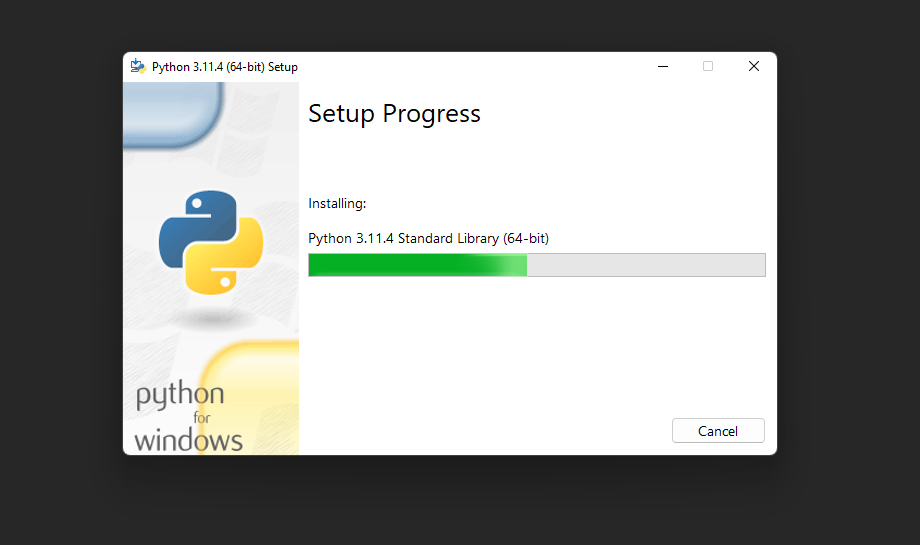
-Al instalar Python automáticamente se estará descargando “PIP” que es un administrador de paquetes que nos permitirá buscar e instalar paquetes específicos de Python con todas sus dependencias automáticamente, lo que facilita la gestión de las bibliotecas y asegura que todas las dependencias requeridas estén correctamente instaladas en tu entorno de desarrollo.



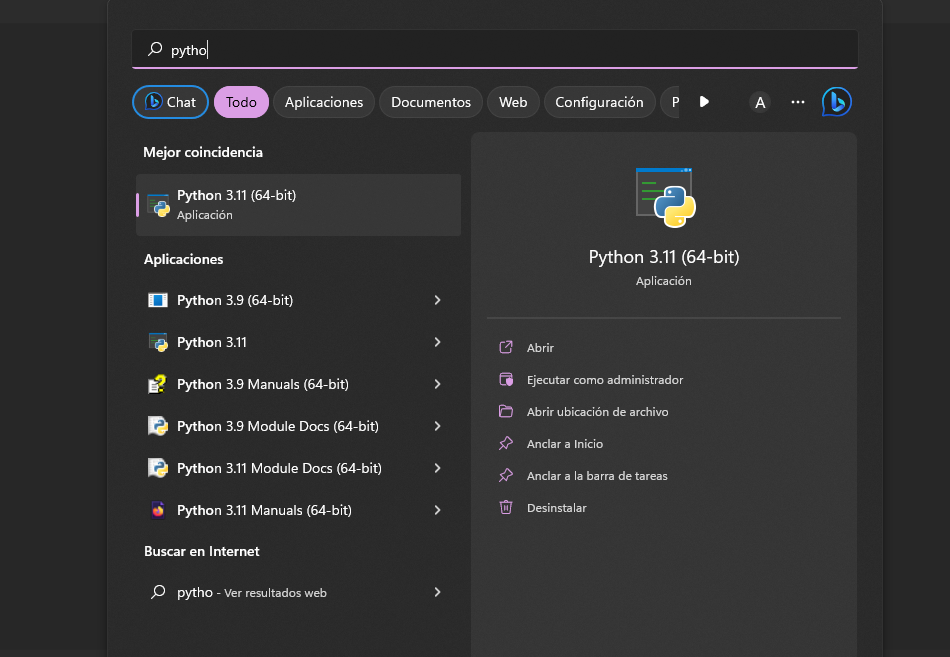
Te muestra en que parte de tu directorio se descargara, donde si deseas puedes cambiarlo.



Presionamos instalar y esperamos que la instalación finalize

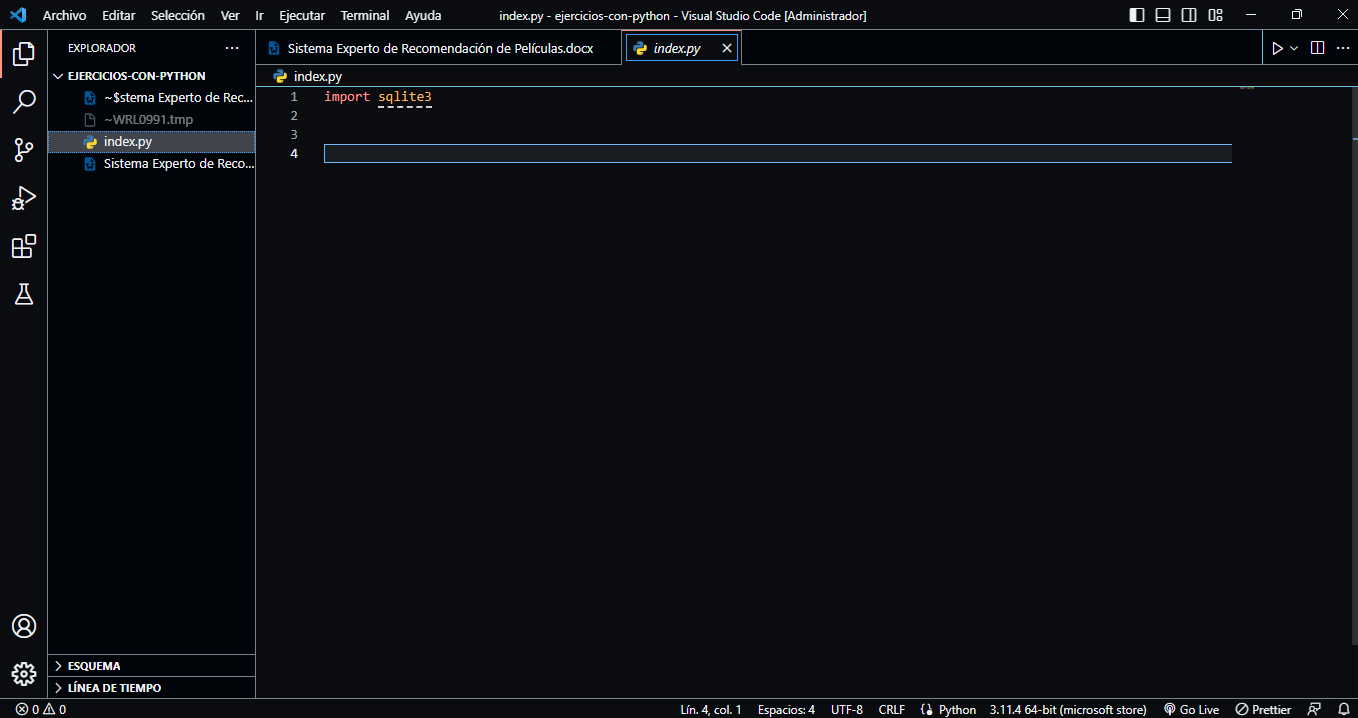


Y listo,cuando se lo busque el software de Python ya estar instalado correctamente en el equipo.



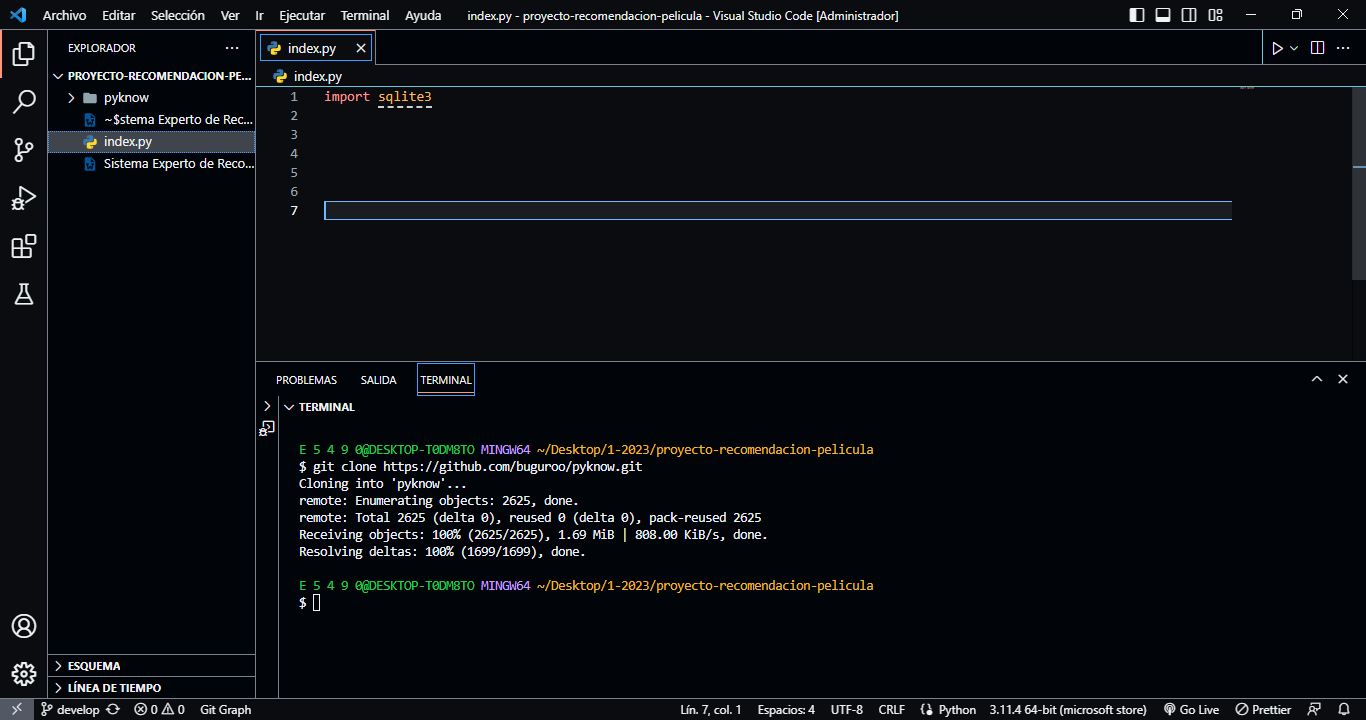
**IMPORTAR EN EL SCRIPT DE PYTHON SQLITE3**

Dentro de nuestro entorno de trabajo importamos la siguiente línea “import sqlite3” , esto para poder trabajar con el gestor de base de datos sqlite3 en nuestro proyecto.

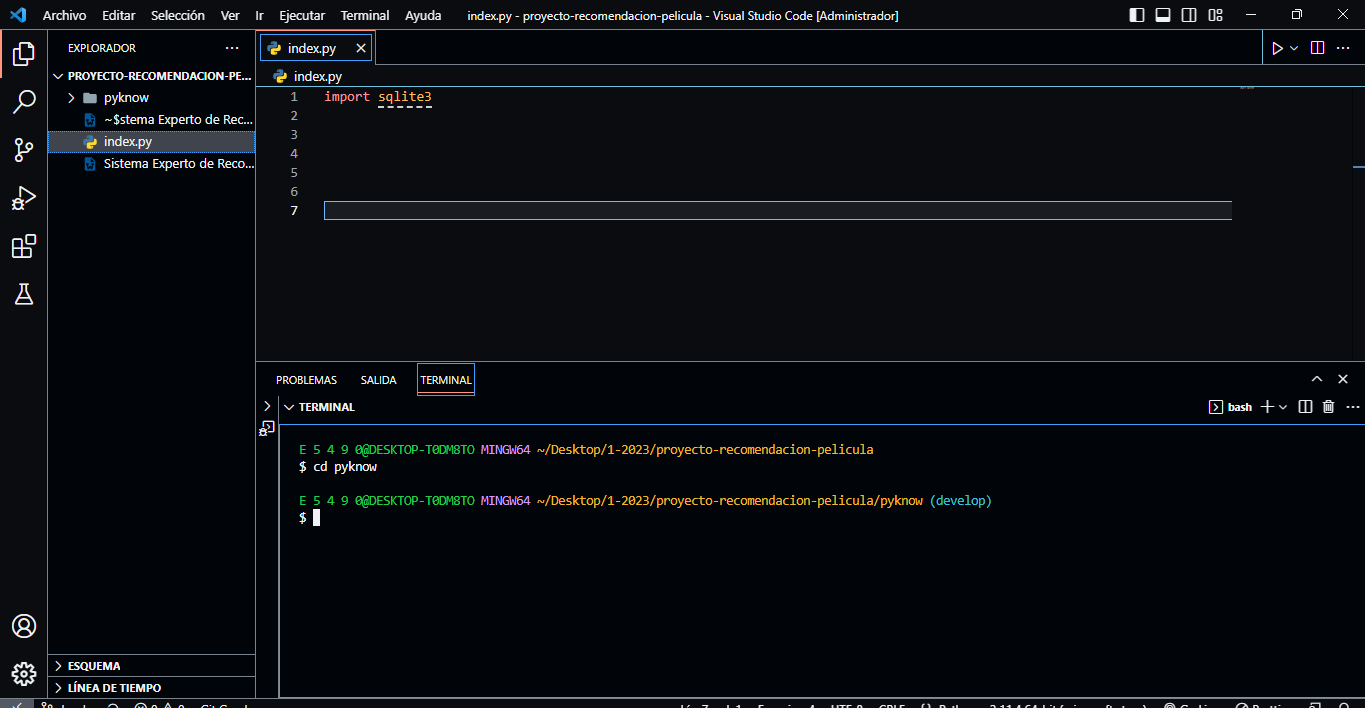


**INSTALACION DE LA BIBLIOTECA “PYKNOW”**

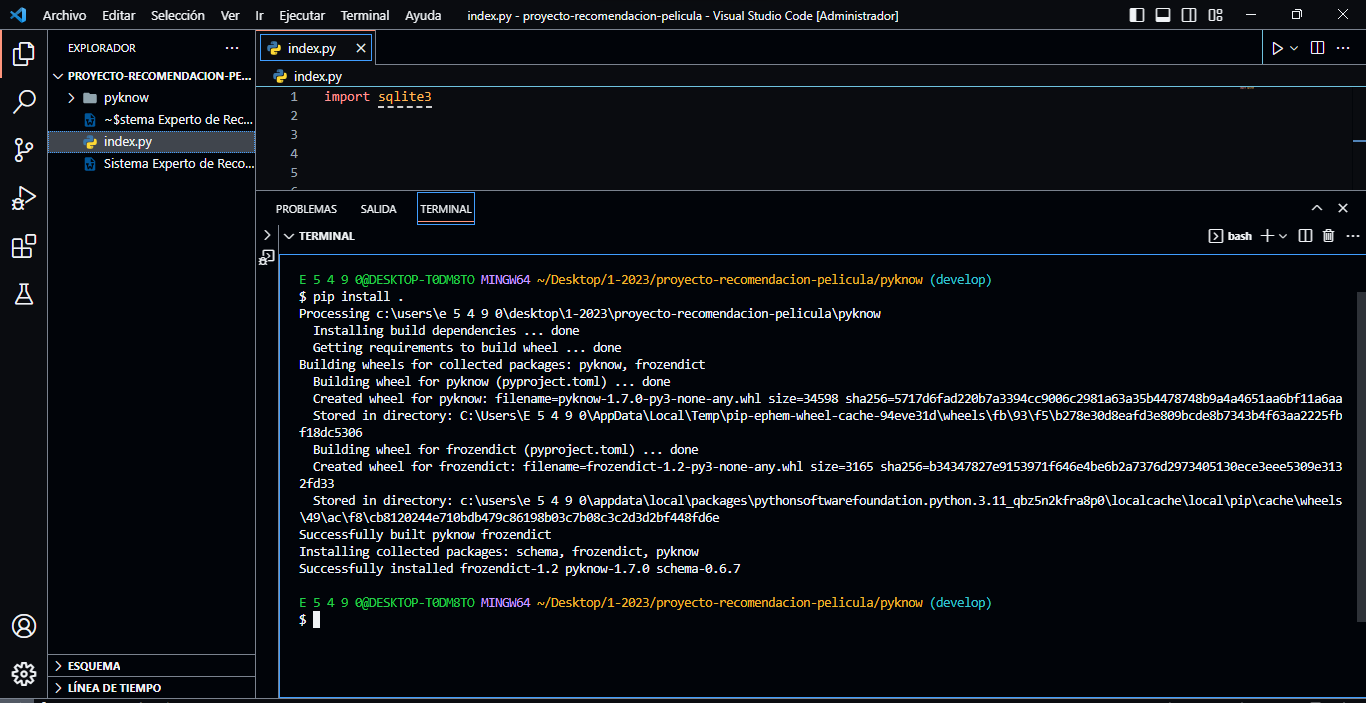
Se ejecuta el siguiente comando para instalar PyKnow utilizando pip: “**git clone https://github.com/buguroo/pyknow.git**”. Este comando clona el repositorio de PyKnow desde GitHub.



Navegamos al directorio del repositorio que se ha clonado con el siguiente comando: **cd pyknow.**

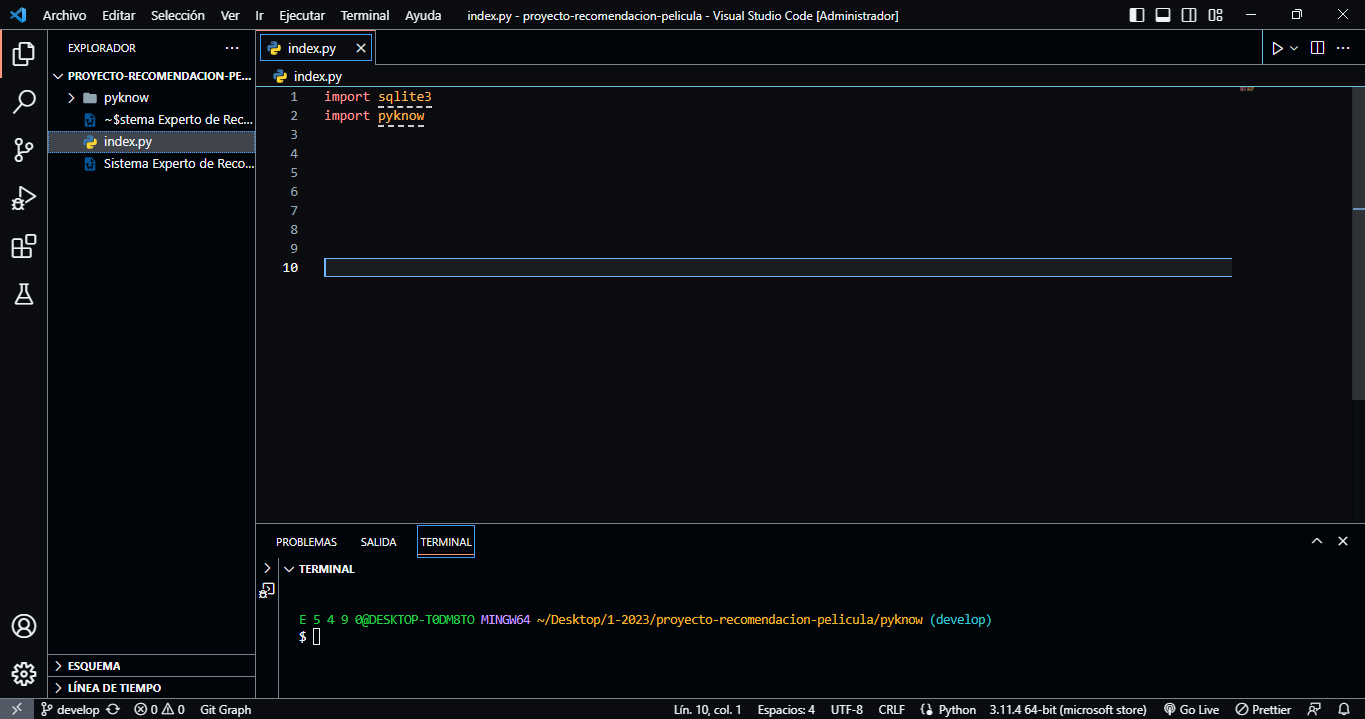


Ejecutamos el comando de instalación utilizando pip: pip install .



Con estos pasos, se instalará PyKnow desde el repositorio de GitHub directamente en tu entorno local.

Ahora ya podremos importar pyknow en el script de Python de nuestro proyecto con la siguiente línea de código: **from pyknow import \*.**



Listo ahora el módulo de sqlite3 y la biblioteca de pyknow ya están listas, ahora se procederá realizar el sistema experto de recomendación de películas, con estas herramientas ya establecidas podemos empezar a trabajar sin problemas.

**DESARROLLO**

Primero definimos las clases correspondientes con las que constara nuestro sistema experto:

* Usuario.
* RecomendaciónPelicula.
* Calificacion.
* Pelicula.
* Clasificacion.
* Genero
* Sistema.

Aclarar que la tabla Sistema será como el Shell o interfaz donde el usuario podrá interactuar con el Sistema Experto de Recomendación de Pelicula.

**Base de Conocimiento**

import sqlite3

from pyknow import \*

# Definición de clases

class Usuario(Fact):

    pass

class RecomendacionPelicula(Fact):

    pass

class Calificacion(Fact):

    pass

class Sistema:

    def \_\_init\_\_(self):

        self.conn = sqlite3.connect('database.db')

        self.cursor = self.conn.cursor()

    def obtener\_recomendaciones(self, usuario):

        # Lógica para obtener recomendaciones

        pass

class Pelicula(Fact):

    pass

class Clasificacion(Fact):

    pass

class Genero(Fact):

    pass

Luego procedemos a hacer la creación de las tablas en Sqlite3.

conn = sqlite3.connect('database.db')

cursor = conn.cursor()

# Tabla Usuario

cursor.execute('''CREATE TABLE IF NOT EXISTS Usuario (

                    id INTEGER PRIMARY KEY,

                    nombre TEXT,

                    email TEXT)''')

# Tabla RecomendacionPelicula

cursor.execute('''CREATE TABLE IF NOT EXISTS RecomendacionPelicula (

                    id INTEGER PRIMARY KEY,

                    usuario\_id INTEGER,

                    pelicula\_id INTEGER,

                    FOREIGN KEY (usuario\_id) REFERENCES Usuario (id),

                    FOREIGN KEY (pelicula\_id) REFERENCES Pelicula (id))''')

# Tabla Calificacion

cursor.execute('''CREATE TABLE IF NOT EXISTS Calificacion (

                    id INTEGER PRIMARY KEY,

                    usuario\_id INTEGER,

                    pelicula\_id INTEGER,

                    valor INTEGER,

                    FOREIGN KEY (usuario\_id) REFERENCES Usuario (id),

                    FOREIGN KEY (pelicula\_id) REFERENCES Pelicula (id))''')

# Tabla Pelicula

cursor.execute('''CREATE TABLE IF NOT EXISTS Pelicula (

                    id INTEGER PRIMARY KEY,

                    titulo TEXT,

                    clasificacion\_id INTEGER,

                    FOREIGN KEY (clasificacion\_id) REFERENCES Clasificacion (id))''')

# Tabla Clasificacion

cursor.execute('''CREATE TABLE IF NOT EXISTS Clasificacion (

                    id INTEGER PRIMARY KEY,

                    nombre TEXT)''')

# Tabla Genero

cursor.execute('''CREATE TABLE IF NOT EXISTS Genero (

                    id INTEGER PRIMARY KEY,

                    nombre TEXT)''')

# Guardar cambios y cerrar conexión

conn.commit()

conn.close()

**2DA PARTE DE PRESENTACIÓN**

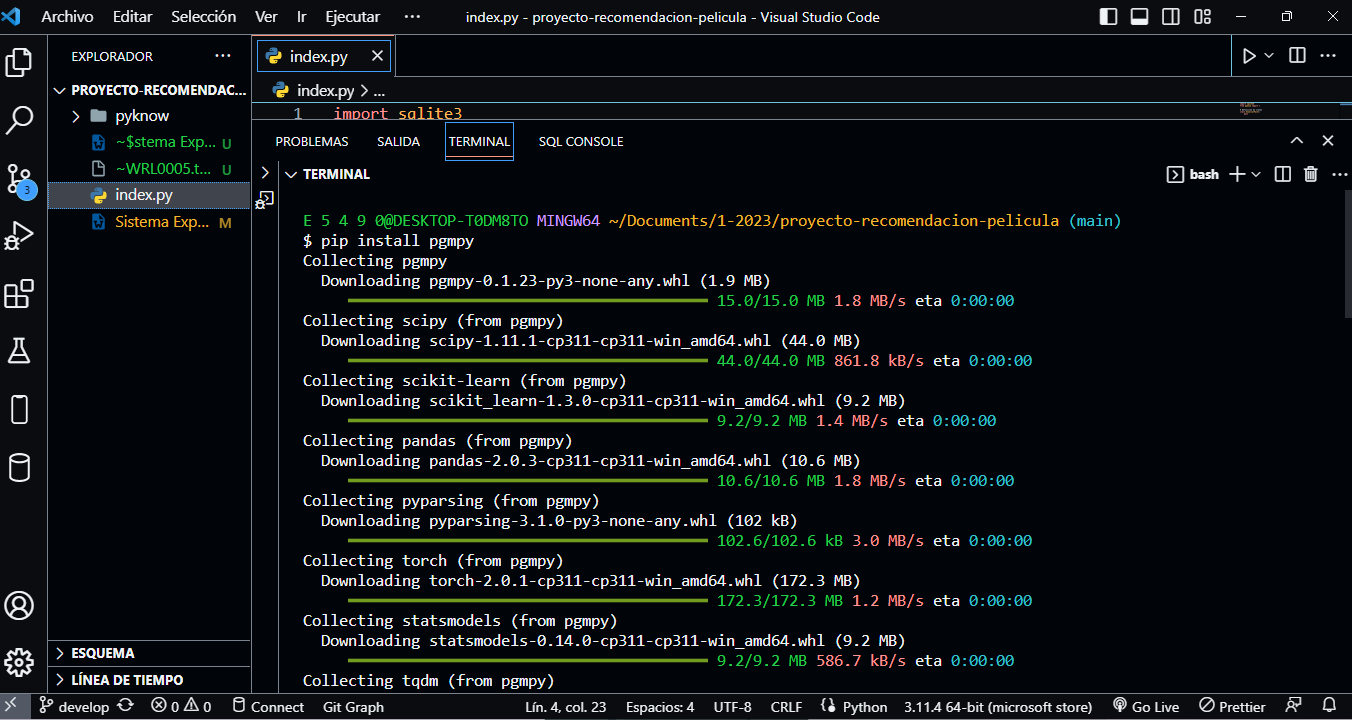
**Motor de Inferencia**

Utilizaremos el modelo bayesiano para realizar el motor de inferencia del sistema experto de recomendación de películas.

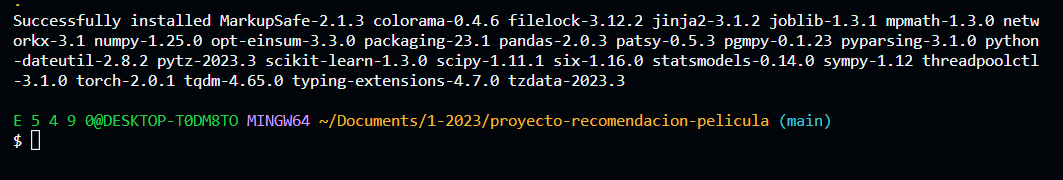
Para implementar el motor de inferencia utilizando redes bayesianas se utilizará la biblioteca **pgmpy**.

A continuación, se muestra la instalación de la biblioteca “**pgmpy”** esto lo realizamos en la carpeta raíz de nuestro proyecto.

Al instalar pgmpy con pip, también se instalarán sus dependencias requeridas para poder utilizarla en el sistema experto. Cuando la instalación de pgmpy se complete, se podrá importar pgmpy en el proyecto y utilizar sus clases y métodos para trabajar con modelos probabilísticos y realizar inferencias.



Par verificar que la biblioteca se instalo correctamente, nos mostrara un mensaje de “instalación correcta” en la terminal dentro de la carpeta raíz del proyecto.



A continuación, importaremos las clases necesarias de pgmpy en el código.

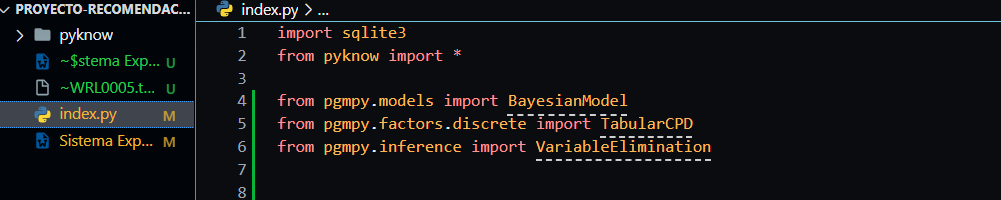
**Clases de pgmpy:**

from pgmpy. models import BayesianModel

from pgmpy .factors.discrete import TabularCPD

from pgmpy. inference import VariableElimination

Importaciones de las Clases con ele que trabajaremos en nuestro proyecto



Para aclarar se trabajará con la biblioteca de PyKnow y pgmpy en este mismo proyecto de sistemas expertos de recomendación de películas en Python, para aprovechar las capacidades de ambos para combinar el motor de inferencia basado en reglas de PyKnow con las funcionalidades de construcción y manipulación de redes bayesianas de pgmpy.

Proseguimos con la definición de la estructura de la Red Bayesianas

# Crear el objeto de la red bayesiana

modelo\_bayesiano = BayesianModel()

# Definir las variables

variables = ["Usuario", "Genero", "Calificacion", "RecomendacionPelicula"]

# Agregar las variables al modelo

modelo\_bayesiano.add\_nodes\_from(variables)

# Definir las relaciones entre las variables

modelo\_bayesiano.add\_edges\_from([

    ("Usuario", "Calificacion"),

    ("Genero", "RecomendacionPelicula"),

    ("Calificacion", "RecomendacionPelicula")

])

Definir las distribuciones de probabilidad condicional (CPDs) de las siguientes variables:

# Definir CPD para la variable "Usuario"

cpd\_usuario = TabularCPD(variable="Usuario", variable\_card=2, values=[[0.6, 0.4]])

# Definir CPD para la variable "Genero"

cpd\_genero = TabularCPD(variable="Genero", variable\_card=2, values=[[0.5, 0.5]])

# Definir CPD para la variable "Calificacion"

cpd\_calificacion = TabularCPD(variable="Calificacion", variable\_card=2,

                             values=[[0.2, 0.8], [0.7, 0.3]],

                             evidence=["Usuario"], evidence\_card=[2])

# Definir CPD para la variable "RecomendacionPelicula"

cpd\_recomendacion = TabularCPD(variable="RecomendacionPelicula", variable\_card=2,

                              values=[[0.3, 0.6, 0.1, 0.9], [0.7, 0.4, 0.9, 0.1]],

                              evidence=["Genero", "Calificacion"],

                              evidence\_card=[2, 2])

# Asociar los CPDs al modelo

modelo\_bayesiano.add\_cpds(cpd\_usuario, cpd\_genero, cpd\_calificacion, cpd\_recomendacion)

Realizamos inferencias utilizando el motor de inferencia:

El objeto inferencia se utiliza para realizar consultas y obtener resultados probabilísticos basados en el modelo bayesiano.

# Crear un objeto de VariableElimination para realizar inferencias

inferencia = VariableElimination(modelo\_bayesiano)

# Realizar una inferencia específica

resultado = inferencia.query(variables=["RecomendacionPelicula"], evidence={"Usuario": 1})

print(resultado)