

FACULTAD DE INGENIERÍA  
INGENIERIA DE SISTEMAS

**Desarrollo Basado en Plataformas**

**Arquitectura de datos**

Actividad final

Proyecto biblioteca

**AUTOR/S**

María Camila Bernal Calderón

Sebastian Guerra Garzón

**DOCENTE**

JUAN CARLOS MARTINEZ DIAZ

**NRC:**

60 – 3830

60 – 3825

BOGOTÁ D.C – COLOMBIA

2024

**Desarrollo Basado en Plataformas**

**Arquitectura de datos**

**Manual Usuario**

**Proyecto biblioteca**

**AUTOR/S**

**María Camila Bernal Calderón**

**Sebastian Guerra Garzón**

i. Formulario F277 – Dirección Nacional de Derechos de Autor.pdf

**ii. Un Prefacio, que contiene detalles de los documentos relacionados y la información sobre como navegar por la guía de usuario.**

- Se nos solicitó generar un sistema de gestión para una librería, el cliente fue la universidad Minuto de Dios, esto como parte de nuestra formación profesional, la idea es poder generar un sistema que funcione así sea para las funciones básicas de una librería. Para poder lograr esto generamos dos documentaciones son manuales uno que es de creación y otro de pruebas. Para nuestro proyecto contamos con los siguientes documentos, tenemos un manual de creación que básicamente nos indica como tenemos que crear los archivos Python y SQL esto lo ponemos como una guía de como el código debería de verse ante los clientes, otra documentación es el manual de pruebas, como se está creando un sistema donde gestione los libros de una librería este debe de tener pruebas piloto en donde muestre el correcto funcionamiento del programa con funciones básicas como lo son agregar libros, editarlos, eliminarlos, vender, etc. Estos manuales son guías al usuario para poder generar consultas tanto en la base de datos como en el sistema que hemos creado. Puesto que muchas personas no conocen como realizar consultas por esa razón hacemos estos manuales, ahora bien, también para que conozcan un poco sobre el código y como este se comporta y los problemas que pudo presentar al momento de implementar. El sistema es fácil de navegar, dentro del visual studio code solo tenemos que realizar el llamado del servidor por medio de una terminal, que el servidor corra y en una nueva terminal abrimos el archivo interfaz con el siguiente comando "Python interfaz.py" esto nos da la entrada a lo que es a la interfaz principal, iniciamos sesión con los usuarios establecidos y ya podemos navegar por el sistema podemos ver los libros, junto con eso de agregarlos, editarlos y eliminarlos, también agregar transacciones ver transacciones ya sea de venta o abastecimiento, y ver el estado de caja. Pero eso lo vemos mas a detalle a continuación.

**iii. Una sección de introducción, que incluye.**

**a. Una breve descripción del sistema y su finalidad.**

- Para conocimiento, tenemos por entendido que una biblioteca requiere un sistema que ayude a gestionar los libros que están dentro de la misma, se invierte un capital de un millón de pesos para surtir el negocio y para tener como colchón, se conoce que los libros se rigen por un ISBN, título, precio de compra, precio de venta y cantidad actual, aparte de esto nos indican que hay dos tipos de transacciones una de venta y otra de abastecimiento, aparte de esto estas transacciones deben de tomar fecha de realización y cantidad de ejemplares que se incluyen en la transacción, también nos dice que el abastecimiento aumenta la cantidad de ejemplares de un libro y la venta le resta ejemplares, y dentro de esto tiene que cumplir con 10 requerimientos funcionales no se nos indica si las interfaces deben de manejar esto también pero si la base de datos puesto que esta es la estructura principal del sistema.

**El programa debe permitir al usuario:**

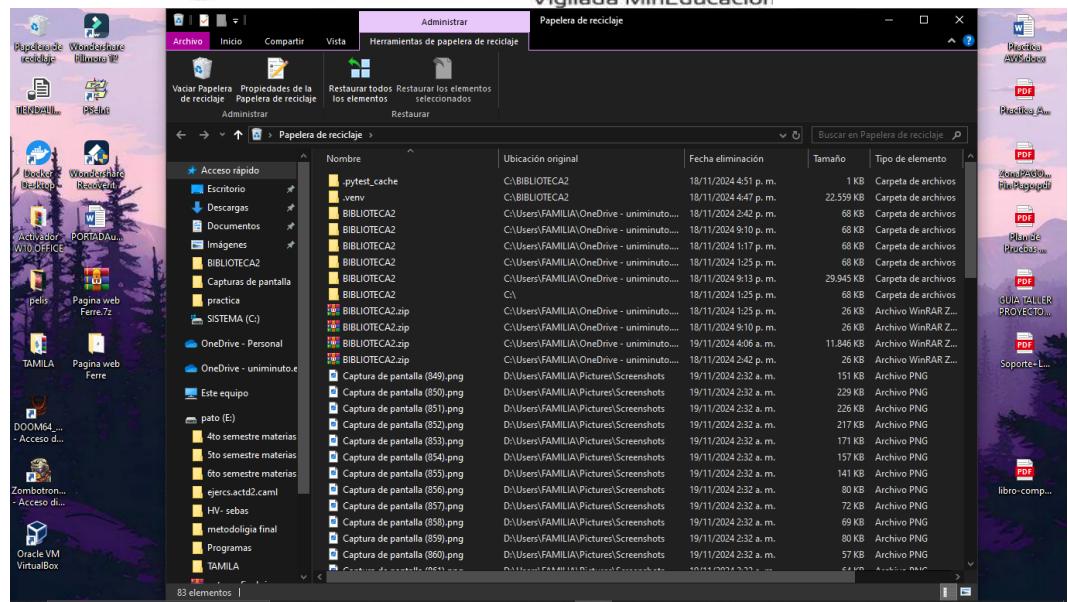
- 1. Registrar un libro en el catálogo.**
- 2. Eliminar un libro del catálogo.**
- 3. Buscar un libro por título.**
- 4. Buscar un libro por ISBN.**
- 5. Abastecer ejemplares de un libro.**
- 6. Vender ejemplares de un libro.**
- 7. Calcular la cantidad de transacciones de abastecimiento de un libro particular.**
- 8. Buscar el libro más costoso.**
- 9. Buscar el libro menos costoso.**
- 10. Buscar el libro más vendido**

Esto claro que lo cumple nuestra base de datos junto a unos TRIGGERS y sentencias de consultas que permiten generar y establecer estos requerimientos.

- La finalidad de nuestro sistema es que pueda gestionar una librería de la forma mas eficiente, además que este pueda cumplir con los estándares que tenga una biblioteca la función de nuestro sistema es agregar libros, editarlos eliminarlos, actualizar lista de libros, así mismo como ver las transacciones agregar transacciones ya se por medio de una venta o un abastecimiento

**b. Una sección de novedades desde la última versión.**

- al día de hoy que hacemos este documento el sistema no presento en ningún momento novedades de problemas, anteriormente presento problemas con la etapa de pruebas pero fueron errores de configuración junto con el test que hacemos, pero en realidad el sistema funciona de manera adecuada, además que el sistema se pensó para correrlo desde un Pepephone es decir que se puede ejecutar en cualquier sistema operativo que se tenga, pero esa parte de requerimientos de lo veremos mas adelante, por el momento estas fueron las novedades que presento desde la última versión. Claro que si nos preguntan novedades desde el inicio tenemos una captura de pantalla en donde se evidencia todas la novedades y gestiones desde la creación del código. Puesto que desde el inicio presento problemas desde la conexión a la base de datos incluso crear un nuevo archivo para la biblioteca, pero por que tenía errores de conexión, en la captura de pantalla vemos los muchos archivos borrados de biblioteca desde los archivos de carpeta como los comprimidos en formato ZIP todas están son pruebas que hacíamos, pero siempre fallaban por cualquier motivo. Así que lo que hacíamos era adjuntar una carpeta nueva colocar el nombre del archivo principal y hacer las modificaciones si funcionaba lo colocábamos en nuestro archivo principal, pero para nosotros es satisfactorio decir que el código que presentamos a lo largo de los manuales y este documento el código fuente original que empezamos desde un inicio, y donde también para crearlo se pasó por un proceso largo, pero se logró hacer realidad.



#### iv. Una sección de requisitos previos necesidades para usar el sistema, que incluye:

##### a. Conocimientos mínimos del usuario

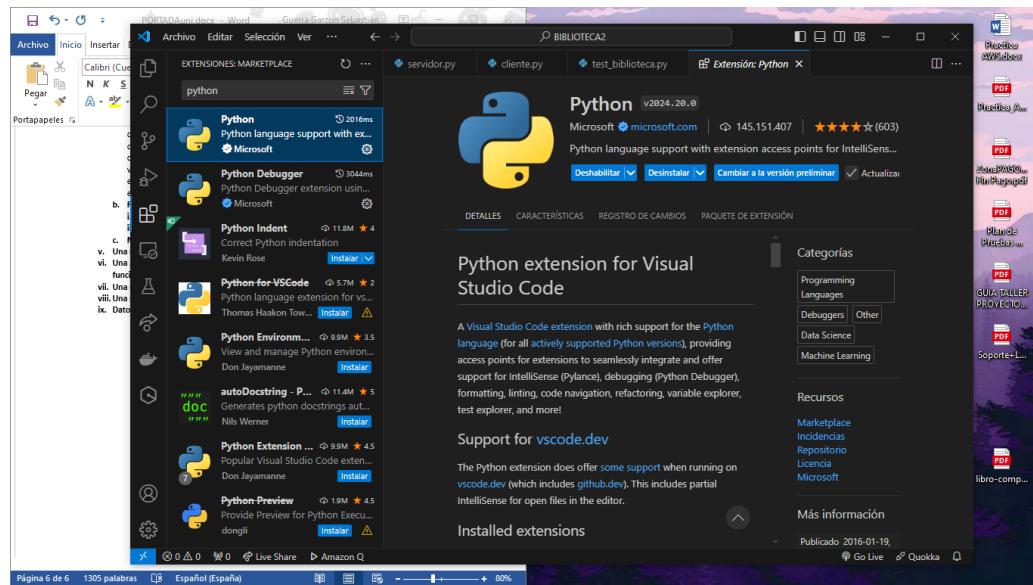
- como recomendación tenemos que saber para manejar el sistema tenemos que leer los dos manuales tanto el de creación y como el de pruebas esto se hace para que los usuarios que usen el sistema conozcan como se creo y funciona y como se hacen las consultas y como funciona una conexión, no es necesario que las personas sepan algo sobre programación puesto que el lenguaje que se utiliza es fácil de entender y el código viene con especificaciones para que sirve y como funciona. Tomemos el conocimiento como si armáramos un lego sin este bloque la construcción no queda terminada y no se vera bien estéticamente y no funcionara bien si mismo pasa con el código. Pero para cumplir hay dos cosas que debemos conocer 1. El conocimiento al manejar el programa de visual studio code

```

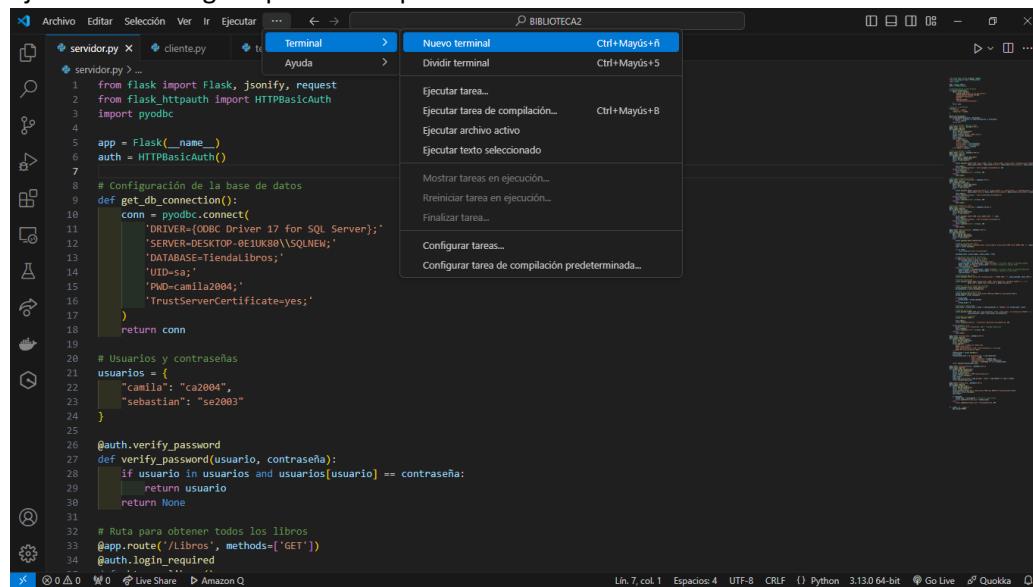
servidor.py cliente.py test_biblioteca.py
tests/test_biblioteca.py
1 import pytest
2 import requests
3 from servidor import app
4
5 # Fixture para ejecutar el servidor durante las pruebas
6 @pytest.fixture(scope="module")
7 def app_cliente():
8     # Configuramos las credenciales por defecto para todas las pruebas
9     app.config['TESTING'] = True
10    with app.test_client() as test_client:
11        yield test_client
12
13 # Prueba la respuesta del servidor
14 def test_servidor_responde(app_cliente):
15     # Agregamos autenticación básica
16     response = app_cliente.get('/Libros', headers={'Authorization': 'Basic Y2FtaWxhOmNlHjAuNA=='})
17     assert response.status_code == 200
18
19 # Prueba que la respuesta sea en formato JSON
20 def test_formato_json(app_cliente):
21     response = app_cliente.get('/Libros', headers={'Authorization': 'Basic Y2FtaWxhOmNlHjAuNA=='})
22     assert response.content_type == 'application/json'
23
24 # Prueba el contenido de la respuesta (obtenemos libros)
25 def test_obtener_libros(app_cliente):
26     response = app_cliente.get('/Libros', headers={'Authorization': 'Basic Y2FtaWxhOmNlHjAuNA=='})
27     data = response.get_json()
28     assert isinstance(data, list)
29     # Modificamos para permitir lista vacía inicialmente
30     assert isinstance(data, list) # Solo verificamos que sea una lista
31
32 # Prueba agregar un libro

```

conocer cómo funciona esta terminar es importante y como tiene que estar configurada correctamente y eso lo veremos más abajo del documento, pero para entender unas cosas necesitamos extensiones para ejecutar estos códigos así que en lado izquierdo vemos un menú o menú hamburguesa donde están todas las funciones que tiene visual studio code verdad. Contamos 5 iconos Asia abajo encontramos el panel de extensiones.



para el correcto funcionamiento instalamos estas dos extensiones estas permiten programar en el lenguaje de Python y ejecutar los códigos sin usar terminales externas o con depuradores. Junto con esto debemos conocer que existen dos maneras de ejecutar un código la primera es por medio de un terminal



Ubicamos los tres puntos y colocamos terminar y nuevo terminar esto habilita una consola pequeña en la parte inferior de la pantalla, acá podemos ejecutar códigos con

el comando “Python #nombre del archivo”, también podemos usar las teclas Windows + Mayús + ñ, para abrir una terminal por medio de comando, la segunda manera es más fácil, en la parte superior derecha aparece un símbolo como play esto es para ejecutar el código de una manera más directa. Y esta nos habilita para correr el código con depuración directa o en una terminar independiente.

The screenshot shows a Python development environment with the following details:

- File Explorer:** Shows files: `servidor.py`, `cliente.py`, and `test_biblioteca.py`.
- Search Bar:** BIBLIOTECA2
- Code Editor:** The `servidor.py` file contains the following code:

```
from flask import Flask, jsonify, request
from flask_httpauth import HTTPBasicAuth
import pyodbc

app = Flask(__name__)
auth = HTTPBasicAuth()

# Configuración de la base de datos
def get_db_connection():
    conn = pyodbc.connect(
        'DRIVER={ODBC Driver 17 for SQL Server};'
        'SERVER=DESKTOP-0E1UK88\SQLNEW;'
        'DATABASE=TiendaLibros;'
        'UID=sa;'
        'PWD=camila2004;'
        'TrustServerCertificate=yes;'
    )
    return conn

# Usuarios y contraseñas
usuarios = {
    "camila": "ca2004",
    "sebastian": "se2003"
}

@auth.verify_password
def verify_password(usuario, contraseña):
    if usuario in usuarios and usuarios[usuario] == contraseña:
        return usuario
    return None

# Ruta para obtener todos los libros
@app.route('/Libros', methods=['GET'])
@auth.login_required
```
- Terminal:** Shows the command `Ejecutar archivo de Python` (Run Python file) and its description: Ejecución del archivo de Python en un terminal dedicado.
- Output:** Shows the command `Depurador de Python: depurar archivo de Python` (Python debugger: debug Python file) and its description: Depurador de Python: depurar mediante launch.json.
- Right Panel:** Shows a vertical stack of code snippets from other files, likely part of the project's history or related code.

Con este conocimiento básico sobre cómo podemos ejecutar el código desde el servidor de Visual Studio code, es más que suficiente.

**b. Requisitos técnicos precios, incluyendo:**

#### i. Capacidades técnicas mínimas del equipo

- como se comentó desde un inicio el sistema fue diseñado para ejecutar desde cualquier computador sea de alta gama o baja gama, un ejemplo de esto es el dispositivo es de gama baja y se realizó este código funcional. Un ejemplo o para que vean que si se puede ejecutar en todo lugar solo miremos las especificaciones de nuestra maquina



Contamos con un sistema operativo de Windows 10 pro, con un procesador Intel Core i3 de 10th generación es un procesador no muy potente y de baja latencia. Las especificaciones de almacenamiento pueden variar, puesto que nuestro dispositivo hace un año se le hicieron modificaciones para aumentar la RAM y el almacenamiento interno, esto casi no tiene que ver con la funcionalidad del sistema puesto que solo en estas características se almacenan datos que se ejecuta, pero como mínimo se necesita un procesador Core i3, con 4 de RAM y 250 de almacenamiento interno. Y pues de hay en adelante se puede ejecutar con procesadores mas poderosos o almacenamientos más masivos.

## ii. Software asociado necesario

- mantener y conseguir los requerimiento no es caro la verdad ya que solo necesitamos el sistema operativo Windows 10 PRO es mejor este sistema operativo puesto que no tiene tanto problemas con la ejecución, es verdad que ahora Windows 11 es la nueva tendencia, pero tiene alguna fallas con algunos dispositivos y para tener la mejor funcionalidad es mejor trabajar con 10 ya que es un sistema que lleva años en el mercado y es básicamente en donde siempre se hacen este tipo de sistemas sobre Windows 10 o 9 la licencia de este cuesta “1.999.999” esta licencia es para varia si son empresas o de formato individual esto también varia a la capacidad de pago sea mensual o se anual, para añadir tenemos que hacerle un respectivo mantenimiento al código del cliente servidor y esto varía entre 150 dólares a 1500 dólares por mes, estamos hablando que son entre unos 660.000 a unos 6.660.000 esto también dependiendo de la complejidad del algoritmo o de cual sea el mantenimiento que requiera, a esto sumemos un computador mas o menos puede valer “1.100.000”, el sistema de visual studio code es gratuito el editor de comando de SQL también es gratuito estamos hablando que un presupuesto seria entre 3.760.000 para tener el sistema operando claro que esto puede variar dependiendo del entorno o de como el cliente quiere ejecutar el código pero como un estado base con un presupuesto de 5.000.000 a 6.000.000 millones es más que suficiente para montar el servicio, ejecutarlo y un colchón ya sea para escalar el servicio como para mantenimientos.

## c. Mecanismo para acceder al sistema

- el sistema usa una parte de autenticación por credenciales esta sección la vemos en el servidor y el cliente y las múltiples interfaces, pero la parte más importante o esencial es el servidor, el servidor aloja los usuarios y sus contraseñas más específicos en esta parte.

```
# Usuarios y contraseñas
usuarios = {
    "camila": "ca2004",
```

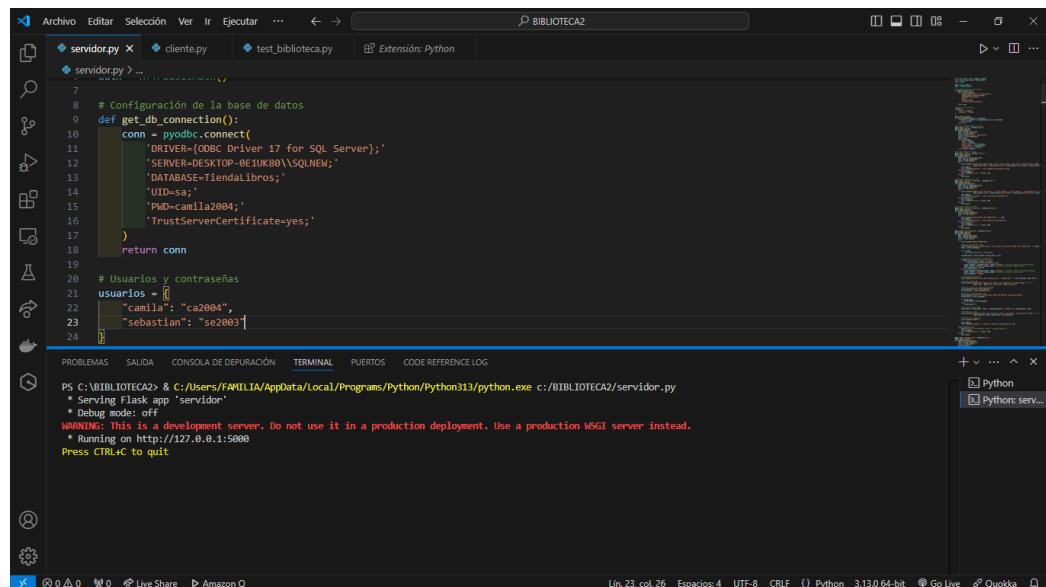
```

        "sebastian": "se2003"
    }

@auth.verify_password
def verify_password(usuario, contraseña):
    if usuario in usuarios and usuarios[usuario] == contraseña:
        return usuario
    return None

```

esta parte aloja los usuarios con las contraseñas, estas se pueden eliminar y agregar nuevas credenciales únicamente acá creamos este cambio ya que cliente usa un método de verificación junto a las interfaces es decir que si cambio a Camila o Sebastian **“que por cierto somos quienes desarrollamos este sistema”** el programa no se ve afectado ya que pensamos en la comodidad y algunos empleados pueden cambiar de cargo o renunciar así que dejamos este método más fácil para que el cambio sea más efectivo.



```

# Configuración de la base de datos
def get_db_connection():
    conn = pyodbc.connect(
        'DRIVER={ODBC Driver 17 for SQL Server};'
        'SERVER=DESKTOP-0E1UK80\SQLNEW;'
        'DATABASE=TiendaLibros;'
        'UID=sa;'
        'PWD=camilla2004;'
        'TrustServerCertificate=yes')
    return conn

# Usuarios y contraseñas
usuarios = [
    {"camila": "ca2004",
     "sebastian": "se2003"}]

```

PS C:\BIBLIOTECA> & C:/Users/FAMILIA/AppData/Local/Programs/Python/Python313/python.exe c:/BIBLIOTECA2/servidor.py
\* Serving Flask app "servidor"
\* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
\* Running on http://127.0.0.1:5000
Press CTRL+C to quit

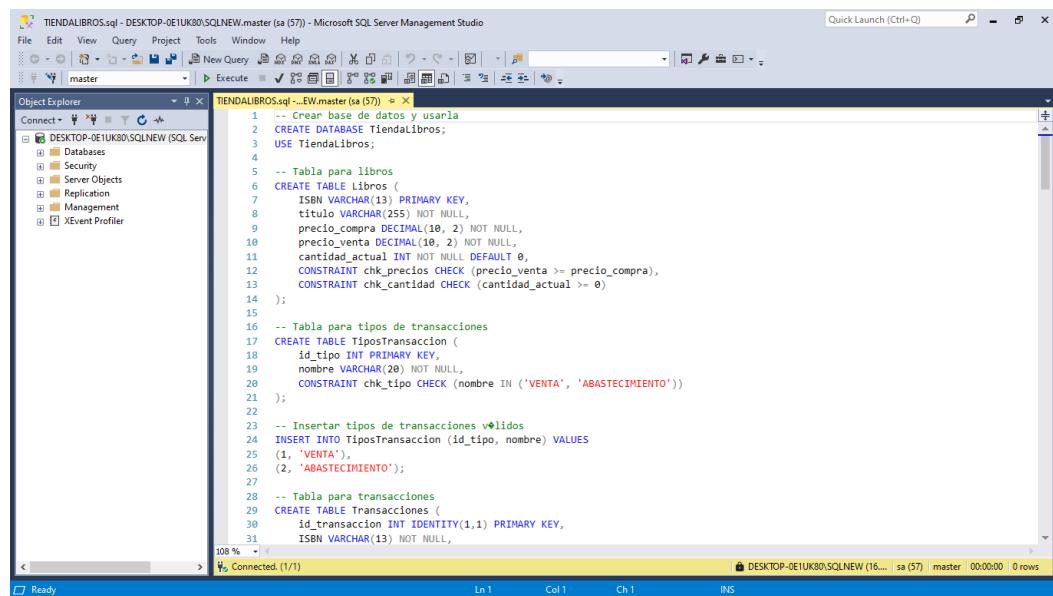
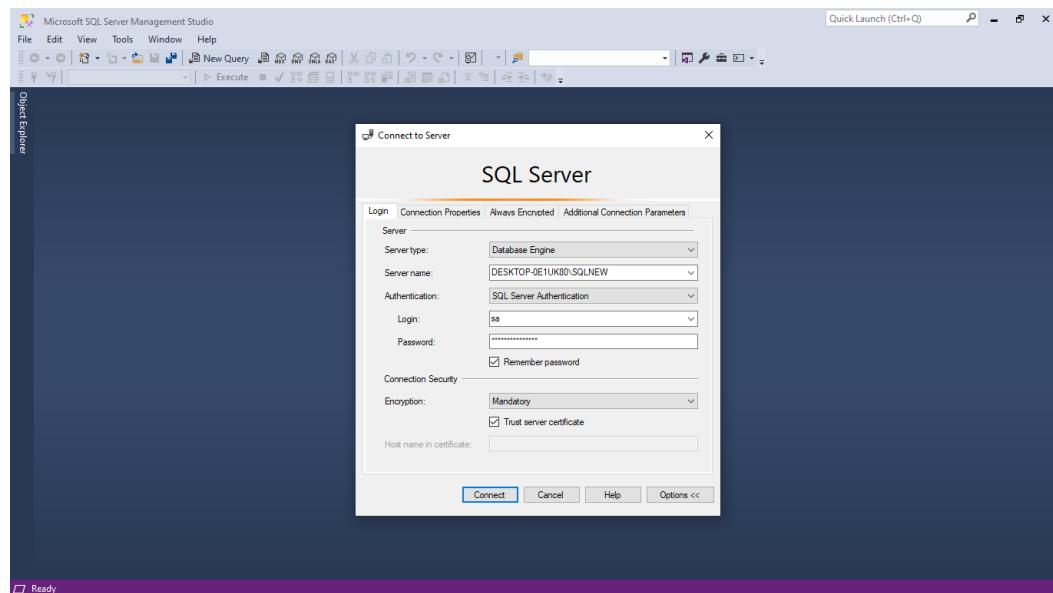
Posteriormente iniciamos el servidor primero siempre será el primero en iniciar y hacemos el llamado de una nueva terminal y hacemos el llamado a la interfaz con el "Python interfaz.py" esto inicia el servicio y sale la primera interfaz de login o inicio de sesión y ya de hay en adelante solo es usar el sistema



**v. Una sección de instalación y configuración (si aplica)**

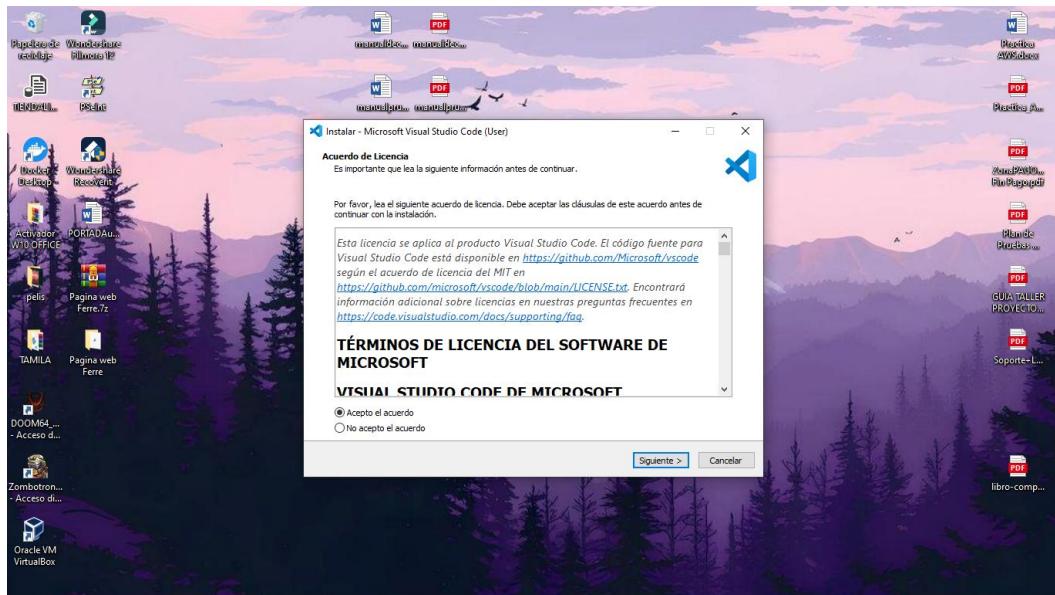
- Bien para esta parte es fundamental seguir el paso a paso para descargar e instalar los programas que necesitamos, en este caso son 3 el SQL server, el visual Studio code y el Python, si necesitamos el ejecutable de pc para poder programar y ejecutar, el visual tiene para editar el código sí, pero si queremos que funcione en nuestra maquina necesitamos esto.
- **INSTALACION SQL**  
para la instalación del SQL seremos un poco más prácticos para que sea de dominio público adjuntamos un link donde estará el aplicado de SQL y el ejecutable para su instalación sumando a esto también realizamos un documento explicamos paso a paso como instalar y configurar todo el entorno del SQL.
  - [https://drive.google.com/drive/folders/1L7QHSxCGiHGoRFuwIkLjeYAMhJ\\_Q1zrp?usp=sharing](https://drive.google.com/drive/folders/1L7QHSxCGiHGoRFuwIkLjeYAMhJ_Q1zrp?usp=sharing)
- Cuando tengamos instalado el SQL es fácil de usar le login, como dejamos la ruta del servidor como defecto solo, pero si no te explicamos como, en login dejamos el “server type” por defecto es decir por “Database Engine” en donde dice “Server name” saldrá el nombre de nuestra máquina y la instancia que está usando para la base de datos usualmente sale de esta manera “#nombre de la maquina\SQLEXPRESS” si tu creaste una nueva extensión solo tenemos que elegir el nombre de la extensión que se creó, un ejemplo en el caso de nosotros es “DESKTOP-0E1UK80\SQLNEW” como se ve no usamos el SQLEXPRESS si no que usamos la extensión que creamos que es SQLNEW. Ahora en autenticación o “Authentication” no accedemos por medio de Windows Authentication por que ingresa por medio de Windows y esto no funciona verdad, en vez de eso colocamos SQL Server Authentication esto nos permite ingresar por usuario y

contraseña colocamos el usuario “sa” este es el que nos dan por defecto y la contraseña ahora en Connection Security tiene que estar en Encryption con un “Mandatory” y la casilla que dice Trust Server Certificate debe estar seleccionada. Esto permite el ingreso sin problema alguno, a continuación, una imagen guía de la configuración.

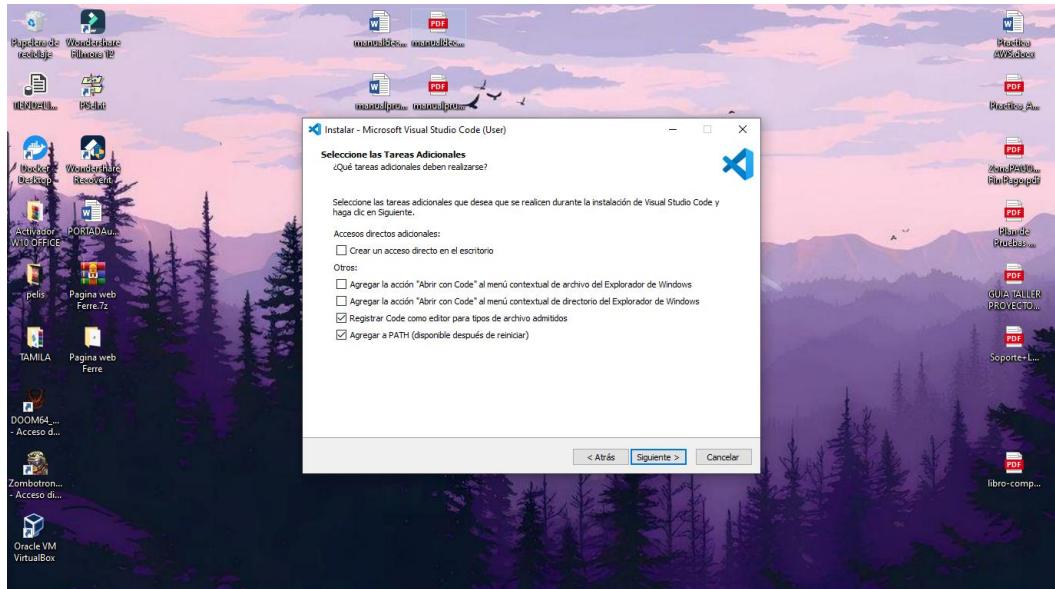


- **INSTALACION VISUAL ESTUDIO CODE**
- Este programa es fácil de instalar puesto que el ya viene configurado de forma predeterminada para que funcione con cualquier lenguaje de programación solo es descargar el aplicativo y ejecutarlo y darle aceptar a todo literal el hace todo por si solo crear la ruta de alojamiento y todo.

- <https://drive.google.com/drive/folders/10AOfy06BL6VEeXS1cK44feibOhvoDisz?usp=sharing>

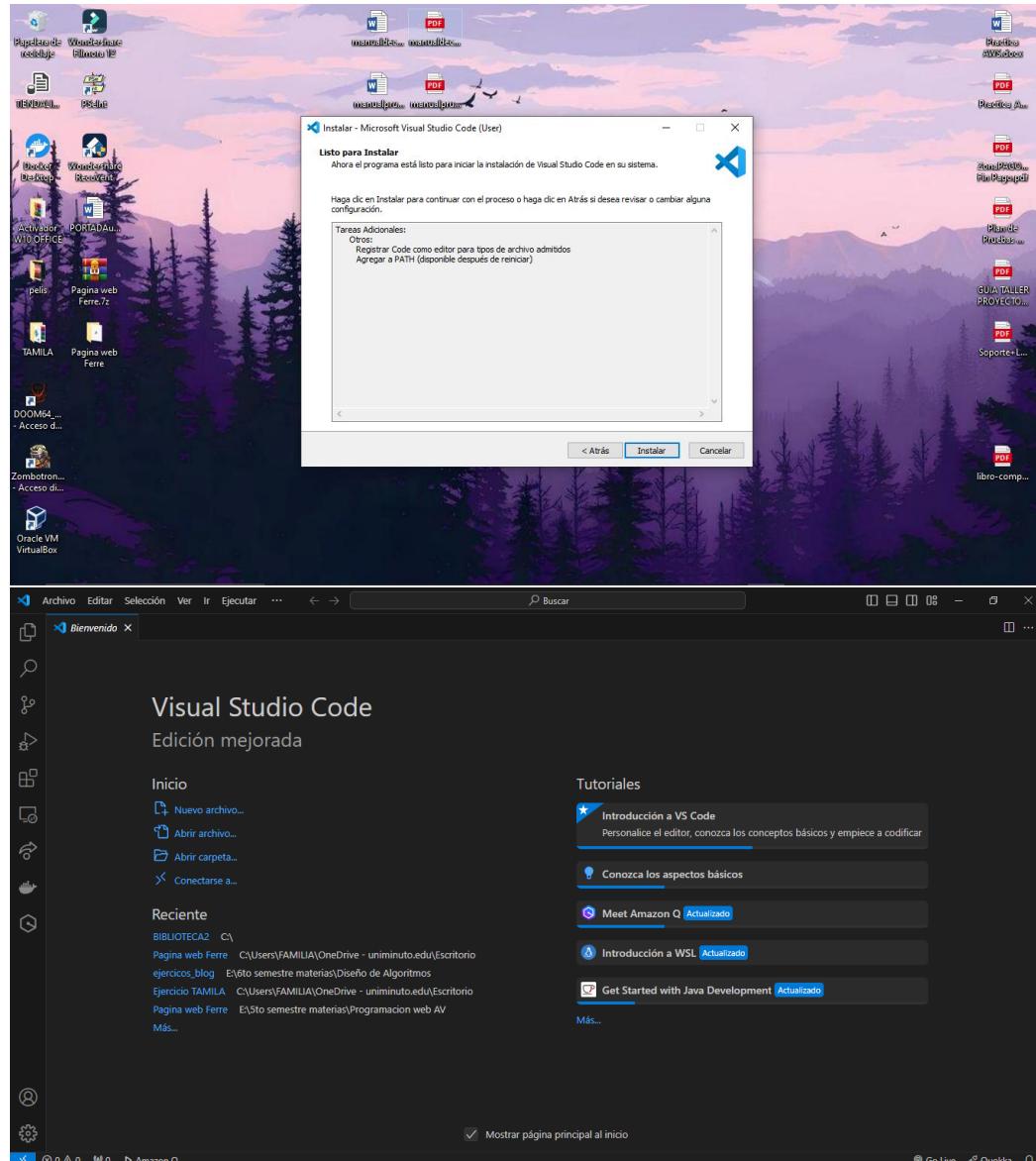


Algo que es super importante es que tenemos que tener la casilla PTHA seleccionada y habilitada, esto nos permite descargar una librería o importación directamente en la terminal de Python



Al seleccionar esta casilla nos quitamos un mundo de problemas al querer instalar de alguna forma cualquier extinción de programa ya que algunos programas necesitan ciertas extensiones para funcionar y este PTHA ayuda a esto a instalar y no tener problemas con las extensiones al momento de que demos siguiente nos indica que podemos instalar, damos instalar y el programa hace su magia he instala el aplicativo y

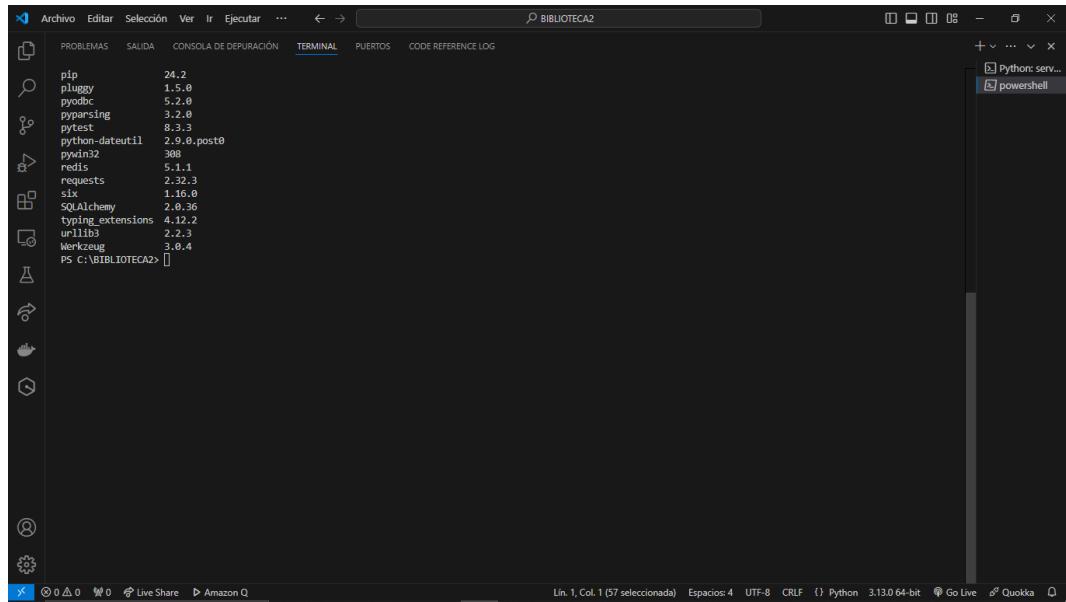
ya solo es entrar y familiarizarse con la interfaz de visual que la verdad es muy fácil de ingresar y entenderla.



Adelantemos un poco el proceso y de una vez abrimos una terminal la idea es descargar todas las extensiones o librerías que necesitamos dentro de la terminar hacemos lo siguiente colocamos

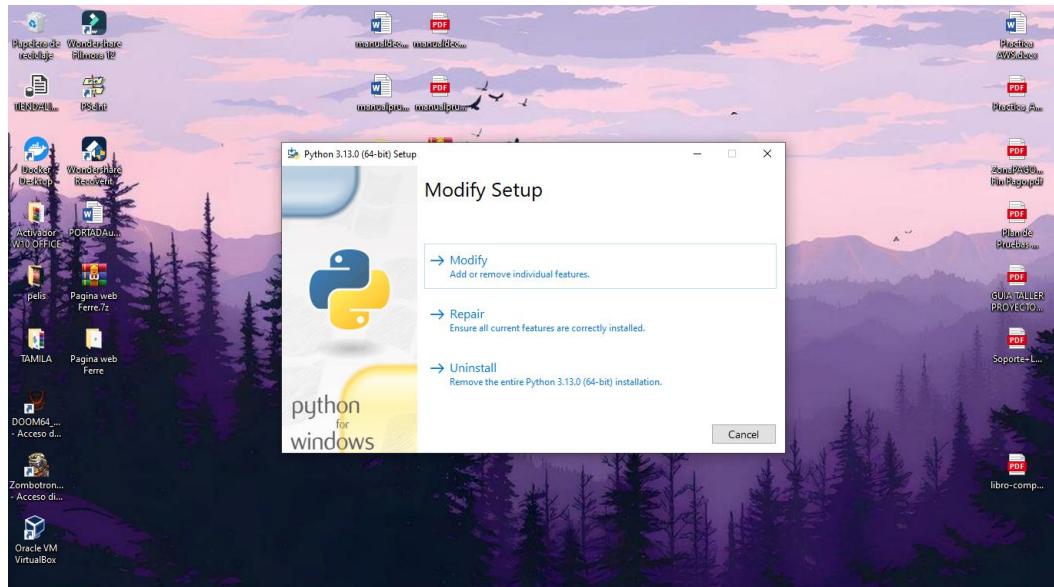
- pip install Flask requests pytest
- pip install Flask – HTTPAuth
- pip install pyodbc

Estas librerías son fundamentales para la ejecución del código ahora solo con un pip list miramos que las librerías estén bien instaladas. En nuestro caso se ve así por que usamos varios programas, pero si miras bien hay están las librerías mencionadas. Si no deja instalar tranquilo será por que no detecta el instalador de Python



### - **INSTALACION PYTHON**

- [https://drive.google.com/drive/folders/1hZ\\_nKMR2Fee6rYhdEUDj0utEm0aNRI4?usp=sharing](https://drive.google.com/drive/folders/1hZ_nKMR2Fee6rYhdEUDj0utEm0aNRI4?usp=sharing)
- Para instalar este programa pasa algo similar que el visual studio code, para esto podemos hacer lo siguiente ejecutamos el archivo, pero en modo administrador, así nos evitamos que nos pidan permisos a cada momento. Entonces al momento de ejecutar esto sale una ventana donde nos indica el paso a paso que tenemos que seguir.



En este caso no podemos evidenciar el paso a paso pero en verdad es muy sencillo de instalar la verdad solo hay una parte en donde tenemos que estar pendiente y es

seleccionar la casilla PTHA esto permite hacer instalaciones dentro de un entorno de Python, esto es esencial y es lo único que nos deben importar.

vi. Una guía sobre cómo utilizar al menos las principales funciones del sistema, es decir sus funciones básicas

- Dentro de lo que es la funcionalidad vamos a mirar como es que funciona la el sistema con sus consultas más básicas.

The screenshot shows a Windows desktop environment with several open windows:

- Code Editor 1 (Left):** Displays Python code for a Flask application named "servidor.py". The code includes functions for user authentication and a route to retrieve all books from a database.
- Code Editor 2 (Top Right):** Displays Python code for an interface named "interfaz.py".
- Terminal Window (Bottom Left):** Shows the output of running the application with Python 3.13, indicating it's a development server.
- Application Window (Bottom Right):** A login interface titled "Login - Sistema de Librería" with fields for "Usuario" and "Contraseña" and a "Ingresar" button.

- Primero ejecutamos el servidor.py para que este se conecte en la base de datos y habilite los disparadores, y ahora el cliente no se activa ya que como se dijo antes este solo hace el llamado de los disparadores, pero en este caso en vez de verlo por medio de un navegador lo vemos por una interface. Este tendrá las funciones de cliente, pero combinándolo con un modelo de vista controlador.
  - Cuando ejecutamos sale esta ventana como colocamos un autenticador sale esta interfaz para iniciar sesión si ingresamos bien las credenciales nos permite el acceso, pero si no lo hacemos

The screenshot shows a Windows desktop with several open windows:

- Code Editor:** A split-screen code editor with two tabs: "servidor.py" and "interfaz.py".
- Terminal:** Shows the command "thon.exe c:/BIBLIOTECA2/servidor.py" running, with a warning message about using it in production.
- Error Dialog:** A modal dialog box titled "Error" with the message "Credenciales inválidas" (Invalid credentials) and an "Aceptar" (Accept) button.
- Login Application:** A window titled "Iniciar Sesión" (Login) with fields for "Usuario" (User) containing "klasd" and "Contraseña" (Password) containing "\*\*\*\*\*". A "Ingresar" (Enter) button is at the bottom.

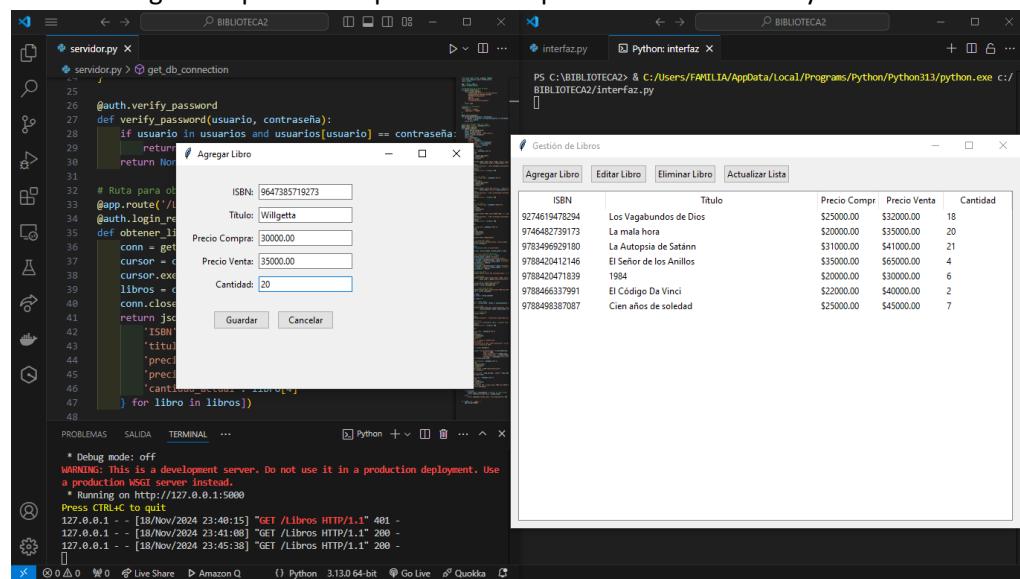
- Sale un error de credenciales esto lo hacemos como para tener una mayor seguridad con la información que manejemos. Cuando colocamos bien las credenciales abre un menú con 4 opciones aun que se ve feo esto se verá mejorado más adelante, pero por motivos de este manual documentamos todo desde su proceso de idealización y su proceso de creación

- Ahora si notamos bien la imagen vemos que el primer GET que está en rojo es cuando hicimos el ejemplo de las credenciales incorrectas que por cierto para ingresar utilizamos el usuario “camila” y contraseña “ca2004” o el usuario “sebastian” y contraseña “se2003” el segundo GET que está en blanco nos indica que se inició sesión correctamente ahora navegamos por las interfaces, abrimos gestión de libros.

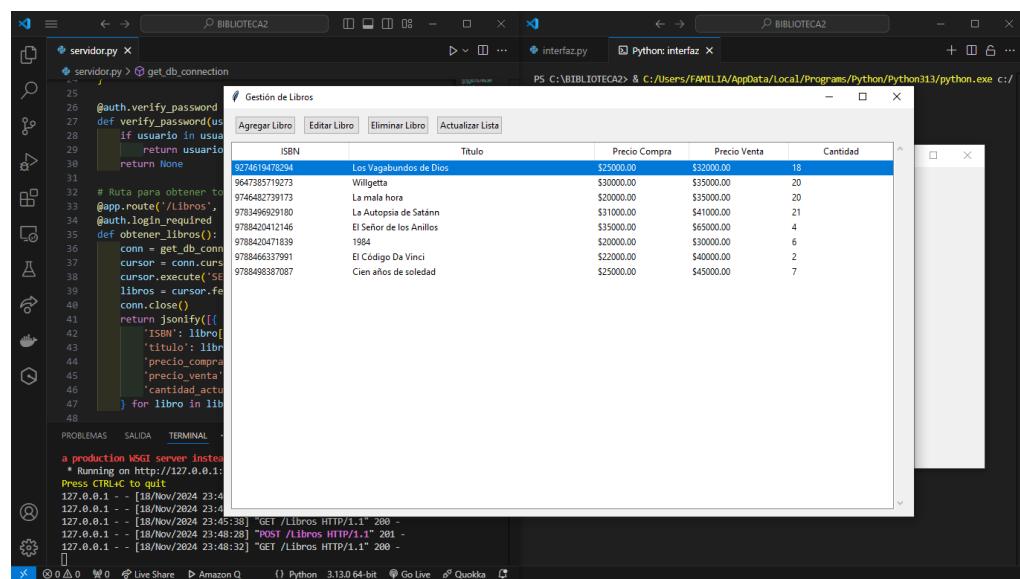
The screenshot shows a terminal window with two panes. The left pane displays the Python code for a library management system, specifically the implementation of the `obtener_libros` method. The right pane shows the command-line interface for managing books, with a table of book details and a history of recent requests.

```
servidor.py
 (servidor.py > get_db.connection)
 25
 26 @auth.verify_password
 27 def verify_password(usuario, contraseña):
 28     if usuario in usuarios and usuarios[usuario] == contraseña:
 29         return usuario
 30     return None
 31
 32 # Ruta para obtener todos los libros
 33 @app.route('/Libros', methods=['GET'])
 34 @auth.login_required
 35 def obtener_libros():
 36     conn = get_db.connection()
 37     cursor = conn.cursor()
 38     cursor.execute('SELECT * FROM Libros')
 39     libros = cursor.fetchall()
 40     conn.close()
 41
 42     return jsonify([
 43         {
 44             'ISBN': libro[0],
 45             'titulo': libro[1],
 46             'precio_compra': float(libro[2]),
 47             'precio_venta': float(libro[3]),
 48             'cantidad_actual': libro[4]
 49     } for libro in libros])
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525
 526
 527
 528
 529
 530
 531
 532
 533
 534
 535
 536
 537
 538
 539
 540
 541
 542
 543
 544
 545
 546
 547
 548
 549
 550
 551
 552
 553
 554
 555
 556
 557
 558
 559
 560
 561
 562
 563
 564
 565
 566
 567
 568
 569
 570
 571
 572
 573
 574
 575
 576
 577
 578
 579
 580
 581
 582
 583
 584
 585
 586
 587
 588
 589
 589
 590
 591
 592
 593
 594
 595
 596
 597
 598
 599
 600
 601
 602
 603
 604
 605
 606
 607
 608
 609
 610
 611
 612
 613
 614
 615
 616
 617
 618
 619
 620
 621
 622
 623
 624
 625
 626
 627
 628
 629
 630
 631
 632
 633
 634
 635
 636
 637
 638
 639
 640
 641
 642
 643
 644
 645
 646
 647
 648
 649
 650
 651
 652
 653
 654
 655
 656
 657
 658
 659
 660
 661
 662
 663
 664
 665
 666
 667
 668
 669
 669
 670
 671
 672
 673
 674
 675
 676
 677
 678
 679
 679
 680
 681
 682
 683
 684
 685
 686
 687
 688
 689
 689
 690
 691
 692
 693
 694
 695
 696
 697
 698
 699
 699
 700
 701
 702
 703
 704
 705
 706
 707
 708
 709
 709
 710
 711
 712
 713
 714
 715
 716
 717
 718
 719
 719
 720
 721
 722
 723
 724
 725
 726
 727
 728
 729
 729
 730
 731
 732
 733
 734
 735
 736
 737
 737
 738
 739
 739
 740
 741
 742
 743
 744
 745
 746
 747
 747
 748
 749
 749
 750
 751
 752
 753
 754
 755
 756
 757
 757
 758
 759
 759
 760
 761
 762
 763
 764
 765
 766
 767
 767
 768
 769
 769
 770
 771
 772
 773
 774
 775
 776
 777
 777
 778
 779
 779
 780
 781
 782
 783
 784
 785
 786
 787
 787
 788
 789
 789
 790
 791
 792
 793
 794
 795
 796
 797
 797
 798
 799
 799
 800
 801
 802
 803
 804
 805
 806
 807
 808
 809
 809
 810
 811
 812
 813
 814
 815
 816
 817
 818
 819
 819
 820
 821
 822
 823
 824
 825
 826
 827
 828
 829
 829
 830
 831
 832
 833
 834
 835
 836
 837
 837
 838
 839
 839
 840
 841
 842
 843
 844
 845
 846
 847
 847
 848
 849
 849
 850
 851
 852
 853
 854
 855
 856
 857
 857
 858
 859
 859
 860
 861
 862
 863
 864
 865
 866
 867
 867
 868
 869
 869
 870
 871
 872
 873
 874
 875
 876
 877
 877
 878
 879
 879
 880
 881
 882
 883
 884
 885
 886
 887
 887
 888
 889
 889
 890
 891
 892
 893
 894
 895
 896
 897
 897
 898
 899
 899
 900
 901
 902
 903
 904
 905
 906
 907
 908
 909
 909
 910
 911
 912
 913
 914
 915
 916
 917
 917
 918
 919
 919
 920
 921
 922
 923
 924
 925
 926
 927
 927
 928
 929
 929
 930
 931
 932
 933
 934
 935
 936
 937
 937
 938
 939
 939
 940
 941
 942
 943
 944
 945
 946
 947
 947
 948
 949
 949
 950
 951
 952
 953
 954
 955
 956
 957
 957
 958
 959
 959
 960
 961
 962
 963
 964
 965
 966
 967
 967
 968
 969
 969
 970
 971
 972
 973
 974
 975
 976
 977
 977
 978
 979
 979
 980
 981
 982
 983
 984
 985
 986
 987
 987
 988
 989
 989
 990
 991
 992
 993
 994
 995
 996
 997
 997
 998
 999
 999
 1000
 1001
 1002
 1003
 1004
 1005
 1006
 1007
 1008
 1009
 1009
 1010
 1011
 1012
 1013
 1014
 1015
 1016
 1017
 1017
 1018
 1019
 1019
 1020
 1021
 1022
 1023
 1024
 1025
 1026
 1027
 1027
 1028
 1029
 1029
 1030
 1031
 1032
 1033
 1034
 1035
 1036
 1037
 1037
 1038
 1039
 1039
 1040
 1041
 1042
 1043
 1044
 1045
 1046
 1047
 1047
 1048
 1049
 1049
 1050
 1051
 1052
 1053
 1054
 1055
 1056
 1057
 1057
 1058
 1059
 1059
 1060
 1061
 1062
 1063
 1064
 1065
 1066
 1067
 1067
 1068
 1069
 1069
 1070
 1071
 1072
 1073
 1074
 1075
 1076
 1077
 1077
 1078
 1079
 1079
 1080
 1081
 1082
 1083
 1084
 1085
 1086
 1087
 1087
 1088
 1089
 1089
 1090
 1091
 1092
 1093
 1094
 1095
 1096
 1097
 1097
 1098
 1099
 1099
 1100
 1101
 1102
 1103
 1104
 1105
 1106
 1107
 1107
 1108
 1109
 1109
 1110
 1111
 1112
 1113
 1114
 1115
 1116
 1117
 1117
 1118
 1119
 1119
 1120
 1121
 1122
 1123
 1124
 1125
 1126
 1127
 1127
 1128
 1129
 1129
 1130
 1131
 1132
 1133
 1134
 1135
 1136
 1137
 1137
 1138
 1139
 1139
 1140
 1141
 1142
 1143
 1144
 1145
 1146
 1147
 1147
 1148
 1149
 1149
 1150
 1151
 1152
 1153
 1154
 1155
 1156
 1157
 1157
 1158
 1159
 1159
 1160
 1161
 1162
 1163
 1164
 1165
 1166
 1167
 1167
 1168
 1169
 1169
 1170
 1171
 1172
 1173
 1174
 1175
 1176
 1176
 1177
 1178
 1178
 1179
 1179
 1180
 1181
 1182
 1183
 1184
 1185
 1186
 1187
 1187
 1188
 1189
 1189
 1190
 1191
 1192
 1193
 1194
 1195
 1196
 1197
 1197
 1198
 1199
 1199
 1200
 1201
 1202
 1203
 1204
 1205
 1206
 1207
 1207
 1208
 1209
 1209
 1210
 1211
 1212
 1213
 1214
 1215
 1216
 1217
 1217
 1218
 1219
 1219
 1220
 1221
 1222
 1223
 1224
 1225
 1226
 1227
 1227
 1228
 1229
 1229
 1230
 1231
 1232
 1233
 1234
 1235
 1236
 1237
 1237
 1238
 1239
 1239
 1240
 1241
 1242
 1243
 1244
 1245
 1246
 1247
 1247
 1248
 1249
 1249
 1250
 1251
 1252
 1253
 1254
 1255
 1256
 1257
 1257
 1258
 1259
 1259
 1260
 1261
 1262
 1263
 1264
 1265
 1266
 1267
 1267
 1268
 1269
 1269
 1270
 1271
 1272
 1273
 1274
 1275
 1276
 1276
 1277
 1278
 1278
 1279
 1279
 1280
 1281
 1282
 1283
 1284
 1285
 1286
 1287
 1287
 1288
 1289
 1289
 1290
 1291
 1292
 1293
 1294
 1295
 1296
 1297
 1297
 1298
 1299
 1299
 1300
 1301
 1302
 1303
 1304
 1305
 1306
 1307
 1307
 1308
 1309
 1309
 1310
 1311
 1312
 1313
 1314
 1315
 1316
 1317
 1317
 1318
 1319
 1319
 1320
 1321
 1322
 1323
 1324
 1325
 1326
 1327
 1327
 1328
 1329
 1329
 1330
 1331
 1332
 1333
 1334
 1335
 1336
 1337
 1337
 1338
 1339
 1339
 1340
 1341
 1342
 1343
 1344
 1345
 1346
 1347
 1347
 1348
 1349
 1349
 1350
 1351
 1352
 1353
 1354
 1355
 1356
 1357
 1357
 1358
 1359
 1359
 1360
 1361
 1362
 1363
 1364
 1365
 1366
 1367
 1367
 1368
 1369
 1369
 1370
 1371
 1372
 1373
 1374
 1375
 1376
 1376
 1377
 1378
 1378
 1379
 1379
 1380
 1381
 1382
 1383
 1384
 1385
 1386
 1387
 1387
 1388
 1389
 1389
 1390
 1391
 1392
 1393
 1394
 1395
 1396
 1397
 1397
 1398
 1399
 1399
 1400
 1401
 1402
 1403
 1404
 1405
 1406
 1407
 1407
 1408
 1409
 1409
 1410
 1411
 1412
 1413
 1414
 1415
 1416
 1417
 1417
 1418
 1419
 1419
 1420
 1421
 1422
 1423
 1424
 1425
 1426
 1427
 1427
 1428
 1429
 1429
 1430
 1431
 1432
 1433
 1434
 1435
 1436
 1437
 1437
 1438
 1439
 1439
 1440
 1441
 1442
 1443
 1444
 1445
 1446
 1447
 1447
 1448
 1449
 1449
 1450
 1451
 1452
 1453
 1454
 1455
 1456
 1457
 1457
 1458
 1459
 1459
 1460
 1461
 1462
 1463
 1464
 1465
 1466
 1467
 1467
 1468
 1469
 1469
 1470
 1471
 1472
 1473
 1474
 1475
 1476
 1476
 1477
 1478
 1478
 1479
 1479
 1480
 1481
 1482
 1483
 1484
 1485
 1486
 1487
 1487
 1488
 1489
 1489
 1490
 1491
 1492
 1493
 1494
 1495
 1496
 1497
 1497
 1498
 1499
 1499
 1500
 1501
 1502
 1503
 1504
 1505
 1506
 1507
 1507
 1508
 1509
 1509
 1510
 1511
 1512
 1513
 1514
 1515
 1516
 1517
 1517
 1518
 1519
 1519
 1520
 1521
 1522
 1523
 1524
 1525
 1526
 1527
 1527
 1528
 1529
 1529
 1530
 1531
 1532
 1533
 1534
 1535
 1536
 1537
 1537
 1538
 1539
 1539
 1540
 1541
 1542
 1543
 1544
 1545
 1546
 1547
 1547
 1548
 1549
 1549
 1550
 1551
 1552
 1553
 1554
 1555
 1556
 1557
 1557
 1558
 1559
 1559
 1560
 1561
 1562
 1563
 1564
 1565
 1566
 1567
 1567
 1568
 1569
 1569
 1570
 1571
 1572
 1573
 1574
 1575
 1576
 1576
 1577
 1578
 1578
 1579
 1579
 1580
 1581
 1582
 1583
 1584
 1585
 1586
 1587
 1587
 1588
 1589
 1589
 1590
 1591
 1592
 1593
 1594
 1595
 1596
 1597
 1597
 1598
 1599
 1599
 1600
 1601
 1602
 1603
 1604
 1605
 1606
 1607
 1607
 1608
 1609
 1609
 1610
 1611
 1612
 1613
 1614
 1615
 1616
 1617
 1617
 1618
 1619
 1619
 1620
 1621
 1622
 1623
 1624
 1625
 1626
 1627
 1627
 1628
 1629
 1629
 1630
 1631
 1632
 1633
 1634
 1635
 1636
 1637
 1637
 1638
 1639
 1639
 1640
 1641
 1642
 1643
 1644
 1645
 1646
 1647
 1647
 1648
 1649
 1649
 1650
 1651
 1652
 1653
 1654
 1655
 1656
 1657
 1657
 1658
 1659
 1659
 1660
 1661
 1662
 1663
 1664
 1665
 1666
 1667
 1667
 1668
 1669
 1669
 1670
 1671
 1672
 1673
 1674
 1675
 1676
 1676
 1677
 1678
 1678
 1679
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1687
 1688
 1689
 1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1697
 1698
 1699
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1706
 1707
 1708
 1708
 1709
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1717
 1718
 1719
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 1727
 1727
 1728
 1729
 1729
 1730
 1731
 1732
 1733
 1734
 1735
 1736
 1737
 1737
 1738
 1739
 1739
 1740
 1741
 1742
 1743
 1744
 1745
 1746
 1747
 1747
 1748
 1749
 1749
 1750
 1751
 1752
 1753
 1754
 1755
 1756
 1757
 1757
 1758
 1759
 1759
 1760
 1761
 1762
 1763
 1764
 1765
 1766
 1767
 1767
 1768
 1769
 1769
 1770
 1771
 1772
 1773
 1774
 1775
 1776
 1777
 1777
 1778
 1779
 1779
 1780
 1781
 1782
 1783
 1784
 1785
 1786
 1787
 1787
 1788
 1789
 1789
 1790
 1791
 1792
 1793
 1794
 1795
 1796
 1797
 1797
 1798
 1799
 1799
 1800
 1801
 1802
 1803
 1804
 1805
 1806
 1807
 1807
 1808
 1809
 1809
 1810
 1811
 1812
 1813
 1814
 1815
 1816
 1817
 1817
 1818
 1819
 1819
 1820
 1821
 1822
 1823
 1824
 1825
 1826
 1827
 1827
 1828
 1829
 1829
 1830
 1831
 1832
 1833
 1834
 1835
 1836
 1837
 1837
 1838
 1839
 1839
 1840
 1841
 1842
 1843
 1844
 1845
 1846
 1847
 1847
 1848
 1849
 1849
 1850
 1851
 1852
 1853
 1854
 1855
 1856
 1857
 1857
 1858
 1859
 1859
 1860
 1861
 1862
 1863
 1864
 1865
 1866
 1867
 1867
 1868
 1869
 1869
 1870
 1871
 1872
 1873
 1874
 1875
 1876
 1877
 1877
 1878
 1879
 1879
 1880
 1881
 1882
 1883
 1884
 1885
 1886
 1887
 1887
 1888
 1889
 1889
 1890
 1891
 1892
 1893
 1894
 1895
 1896
 1897
 1897
 1898
 1899
 1899
 1900
 1901
 1902
 1903
 1904
 1905
 1906
 1907
 1907
 1908
 1909
 1909
 1910
 1911
 1912
 1913
 1914
 1915
 1916
 1917
 1917
 1918
 1919
 1919
 1920
 1921
 1922
 1923
 1924
 1925
 1926
 1927
 1927
 1928
 1929
 1929
 1930
 1931
 1932
 1933
 1934
 1935
 1936
 1937
 1937
 1938
 1939
 1939
 1940
 1941
 1942
 1943
 1944
 1945
 1946
 1947
 1947
 1948
 1949
 1949
 1950
 1951
 1952
 1953
 1954
 1955
 1956
 1957
 1957
 1958
 1959
 1959
 1960
 1961
 1962
 1963
 1964
 1965
 1966
 1967
 1967
 1968
 1969
 1969
 
```

- Solicita la petición de ver los libros que están en la base de datos y me deje o agregar libros, editarlos, eliminarlos o que me actualice la lista de los mismos, dato curioso el editar libro fue añadido por nuestra parte ya que no es un requerimiento funcional que solicita el proyecto, pero lo colocamos como un plus del mismo.
- Ahora probemos los botones, si queremos agregar un libro nos abre una ventana que nos solicita un ISBN, un título, precio de compra y venta y una cantidad de libros, actualmente contamos con 7 libros en la base de datos tanto creados como directamente creados en la SQL, creamos un libro con el “ISBN: 9647385719273” “título: Willgetta” “precio compra: 30.000” “precio venta: 35.000” y “cantidad: 20”

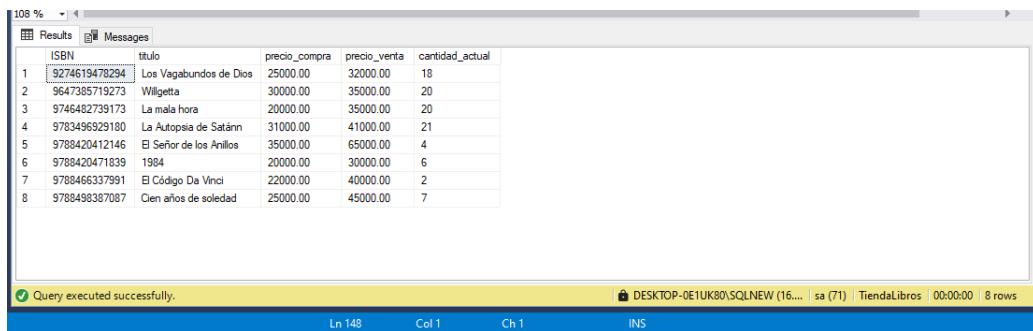


ISBN	Título	Precio Compr.	Precio Venta	Cantidad
9274619478294	Los Vagabundos de Dios	\$25000.00	\$32000.00	18
9746482739173	La mala hora	\$20000.00	\$35000.00	20
9783496929180	La Autopsia de Satann	\$31000.00	\$41000.00	21
9788420412146	El Señor de los Anillos	\$35000.00	\$65000.00	4
9788420471839	1984	\$20000.00	\$30000.00	6
9788466337991	El Código Da Vinci	\$22000.00	\$40000.00	2
9788498387087	Cien años de soledad	\$25000.00	\$45000.00	7



ISBN	Título	Precio Compra	Precio Venta	Cantidad
9274619478294	Los Vagabundos de Dios	\$25000.00	\$32000.00	18
9647385719273	Willgetta	\$30000.00	\$35000.00	20
9746482739173	La mala hora	\$20000.00	\$35000.00	20
9783496929180	La Autopsia de Satann	\$31000.00	\$41000.00	21
9788420412146	El Señor de los Anillos	\$35000.00	\$65000.00	4
9788420471839	1984	\$20000.00	\$30000.00	6
9788466337991	El Código Da Vinci	\$22000.00	\$40000.00	2
9788498387087	Cien años de soledad	\$25000.00	\$45000.00	7

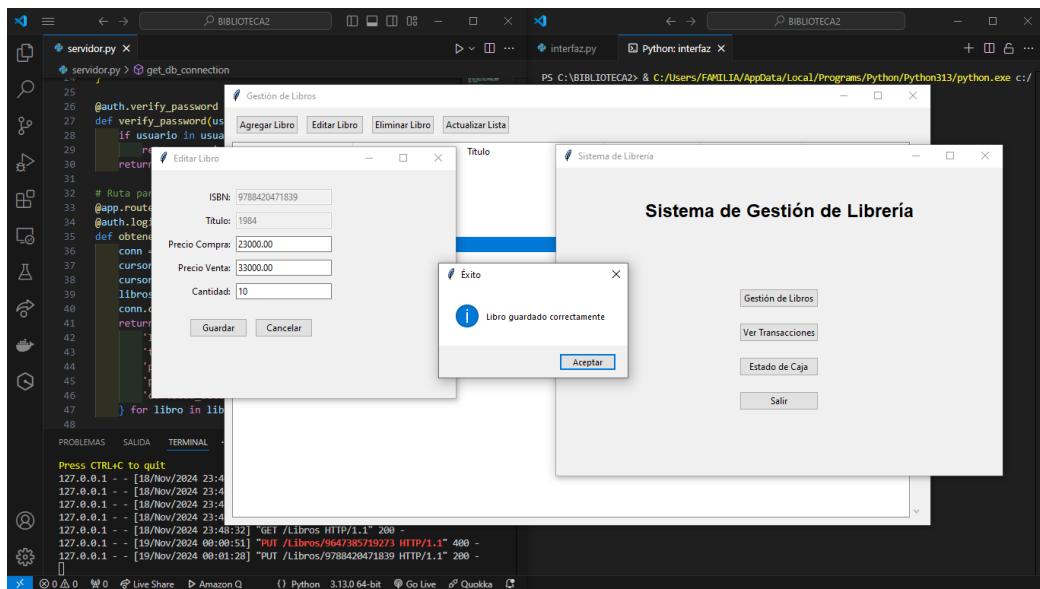
- Cuando creamos el libro aparece agregado, pero como sé que mi base de datos está recibiendo la información de entrada, bueno por dos factores el primero su podemos ver en la esquina inferior izquierda tenemos una línea morada que pone un POST este método lo usamos para agregar un libro. Y bien la segunda cosa que podemos hacer es que en nuestro aplicativo donde estemos usando la base de datos SQL ejecutamos un SELECT \* FROM Libros para validar si este fue agregado y como esto es para validar pues intentemos.



ISBN	título	precio_compra	precio_venta	cantidad_actual
1 9274619478294	Los Vagabundos de Dios	25000.00	32000.00	18
2 9647385719273	Willgetta	30000.00	35000.00	20
3 9746482739173	La mala hora	20000.00	35000.00	20
4 9783496529180	La Autopsia de Satán	31000.00	41000.00	21
5 9788420412146	El Señor de los Anillos	35000.00	65000.00	4
6 9789420471839	1984	20000.00	30000.00	6
7 9788465337991	El Código Da Vinci	22000.00	40000.00	2
8 9788498387087	Cien años de soledad	25000.00	45000.00	7

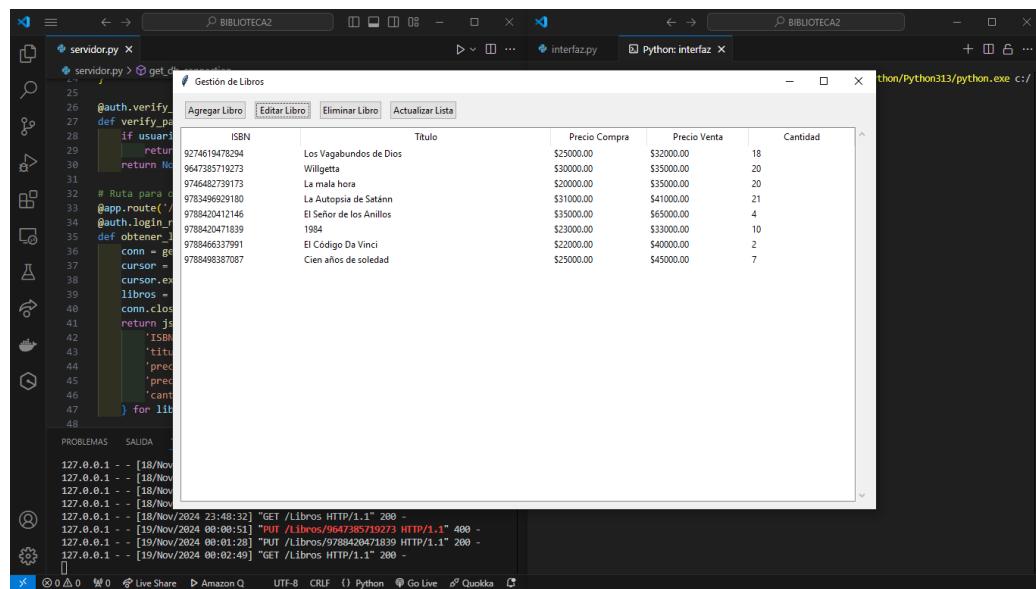
Query executed successfully.

- Y como podemos ver acá está el libro que acabamos de agregar, así mismo pasa cuando queremos eliminar para eliminar tenemos que seleccionar el libro si no seleccionamos un libro nos sale una alerta indicando que tenemos que escoger un libro para eliminar y la misma función tiene para editar, así que miremos cómo funciona el editar y eliminar.



- En este caso editemos el libro de 1994 le bajamos los precios y aumentamos las cantidades nos transmite un cambio y utiliza el servidor un método POST para esto. Y como comprobar este libro tenía un precio de compra de 20.000 un precio de venta de

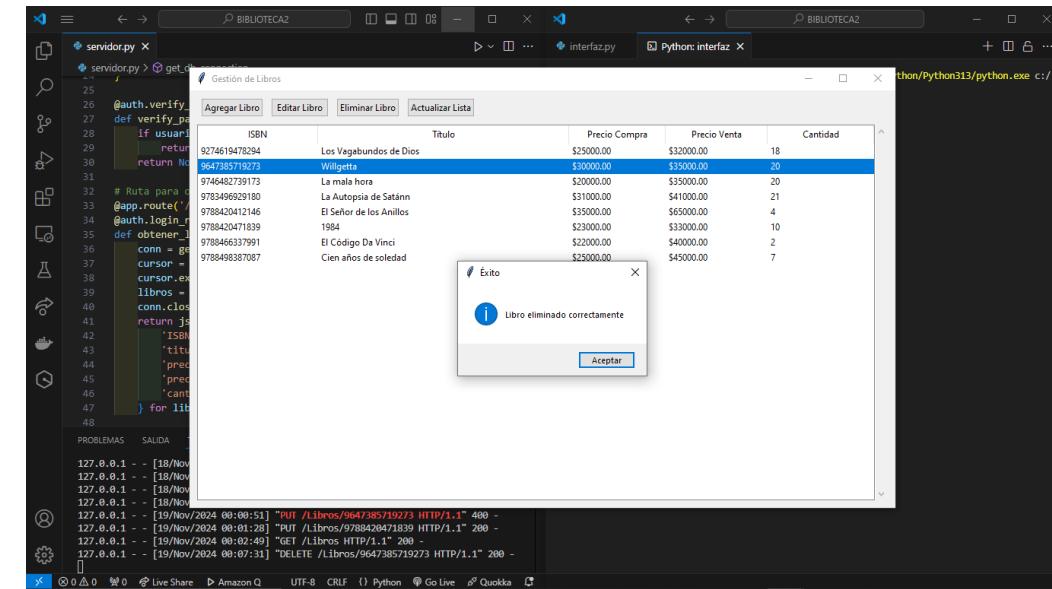
30.000 y tenía 6 unidades ahora al editarlo le decimos que su precio de compra es de 23.000, su precio de venta es de 33.000 y tenemos al momento 10 unidades.



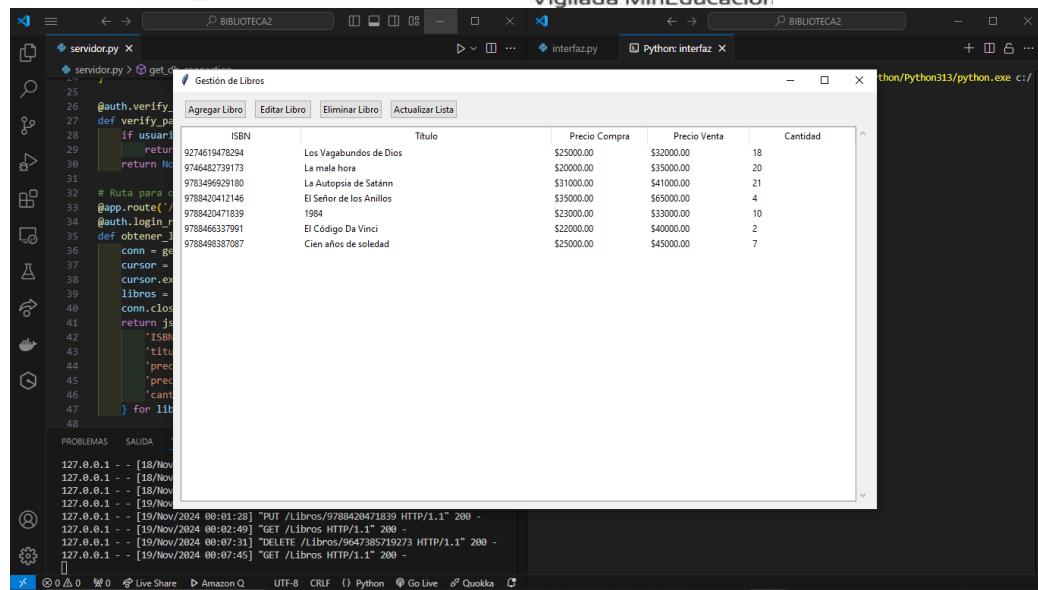
The screenshot shows a development environment with two windows. On the left, a code editor displays a Python file named 'servidor.py' containing code for a REST API endpoint. On the right, a terminal window shows the command 'python3.13/python.exe c:/'. Below the terminal is a log of HTTP requests. In the center, a 'Gestión de Libros' window is open, showing a table of books with columns: ISBN, Título, Precio Compra, Precio Venta, and Cantidad. The table data is as follows:

ISBN	Título	Precio Compra	Precio Venta	Cantidad
9274619478294	Los Vagabundos de Dios	\$25000.00	\$32000.00	18
9647385719273	Willgetta	\$30000.00	\$35000.00	20
9746482739173	La mala hora	\$20000.00	\$35000.00	20
9783496929180	La Autopsia de Satán	\$31000.00	\$41000.00	21
9788420412146	El Señor de los Anillos	\$35000.00	\$65000.00	4
9788420471839	1984	\$23000.00	\$33000.00	10
9788466337991	El Código Da Vinci	\$22000.00	\$40000.00	2
9788498387087	Cien años de soledad	\$25000.00	\$45000.00	7

- Acá tenemos el cambio y como validamos que si funcione lo pues lo mismo validamos por la base de datos. Con esto uno tiene un control de una base de datos con un cliente servidor combinándolo con un modelo de vista controlador. Ahora probemos el ultimo y es eliminar un libro. Eliminemos el libro que creamos seleccionamos el libro y apretamos el botón de borrar y el libro hace un DELETE



The screenshot shows the same development environment and terminal logs as the previous image. The 'Gestión de Libros' window now shows the book 'Willgetta' selected. A confirmation dialog box titled 'Éxito' appears, stating 'Libro eliminado correctamente' (Book deleted successfully). The 'Aceptar' button is visible at the bottom of the dialog. The table data remains the same as in the previous screenshot.



The terminal window shows the following Python code for a library management system:

```

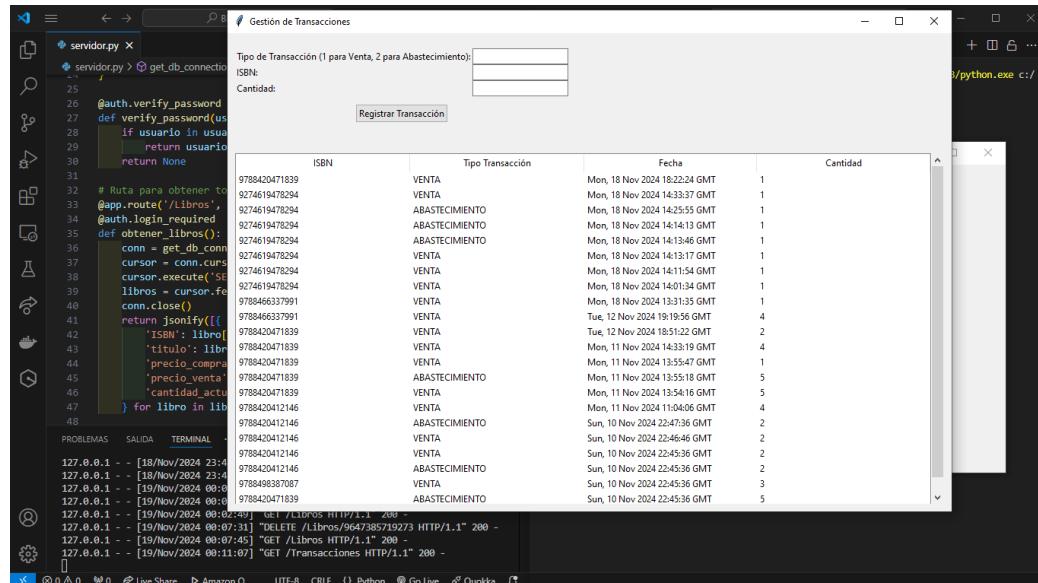
25
26 @auth.verify_password
27 def verify_password(usuario, password):
28     if usuario == "willgetta" and password == "123456":
29         return True
30     return None
31
32 # Ruta para obtener los libros
33 @app.route('/Libros')
34 @auth.login_required
35 def obtener_libros():
36     conn = get_db_connection()
37     cursor = conn.cursor()
38     cursor.execute("SELECT * FROM Libros")
39     libros = cursor.fetchall()
40     conn.close()
41     return jsonify([
42         {
43             'ISBN': libro['ISBN'],
44             'titulo': libro['titulo'],
45             'precio_compra': libro['precio_compra'],
46             'precio_venta': libro['precio_venta'],
47             'cantidad_actual': libro['cantidad_actual']
48         } for libro in libros
49     ])
50
51 PROBLEMAS SALIDA TERMINAL
52
53 127.0.0.1 - - [18/Nov/2024:00:01:28] "PUT /Libros/9788420471839 HTTP/1.1" 200 -
54 127.0.0.1 - - [19/Nov/2024:00:02:49] "GET /Libros HTTP/1.1" 200 -
55 127.0.0.1 - - [19/Nov/2024:00:07:31] "DELETE /Libros/9647385719273 HTTP/1.1" 200 -
56 127.0.0.1 - - [19/Nov/2024:00:07:45] "GET /Libros HTTP/1.1" 200 -
57 127.0.0.1 - - [19/Nov/2024:00:11:07] "GET /Transacciones HTTP/1.1" 200 -

```

The 'Gestión de Libros' application window displays a table of books:

ISBN	Título	Precio Compra	Precio Venta	Cantidad
974619478294	Los Vagabundos de Dios	\$25000.00	\$30000.00	18
9746482739173	La mala hora	\$20000.00	\$35000.00	20
9783496929180	La Autopsia de Satán	\$31000.00	\$41000.00	21
9788420412146	El Señor de los Anillos	\$35000.00	\$65000.00	4
9788420471839	1984	\$23000.00	\$33000.00	10
9788466337991	El Código Da Vinci	\$22000.00	\$40000.00	2
9788498387087	Cien años de soledad	\$25000.00	\$45000.00	7

- Y como vemos en la esquina inferior izquierda vemos el método DELETE con la ruta DELETE/Libros/#ISBN del libro eliminado en este caso vemos la eliminación de Willgetta. Ahora veamos un poco sobre las interfaces de transacciones. Este tipo de interfaz es un poco sensible puesto que esta trabaja con TREAGGERS que están implementados en la SQL.



The terminal window shows the following Python code for a transaction management system:

```

25
26 @auth.verify_password
27 def verify_password(usuario, password):
28     if usuario == "willgetta" and password == "123456":
29         return True
30     return None
31
32 # Ruta para obtener los libros
33 @app.route('/Libros')
34 @auth.login_required
35 def obtener_libros():
36     conn = get_db_connection()
37     cursor = conn.cursor()
38     cursor.execute("SELECT * FROM Libros")
39     libros = cursor.fetchall()
40     conn.close()
41     return jsonify([
42         {
43             'ISBN': libro['ISBN'],
44             'titulo': libro['titulo'],
45             'precio_compra': libro['precio_compra'],
46             'precio_venta': libro['precio_venta'],
47             'cantidad_actual': libro['cantidad_actual']
48         } for libro in libros
49     ])
50
51 PROBLEMAS SALIDA TERMINAL
52
53 127.0.0.1 - - [18/Nov/2024:00:01:28] "PUT /Libros/9788420471839 HTTP/1.1" 200 -
54 127.0.0.1 - - [19/Nov/2024:00:02:49] "GET /Libros HTTP/1.1" 200 -
55 127.0.0.1 - - [19/Nov/2024:00:07:31] "DELETE /Libros/9647385719273 HTTP/1.1" 200 -
56 127.0.0.1 - - [19/Nov/2024:00:07:45] "GET /Libros HTTP/1.1" 200 -
57 127.0.0.1 - - [19/Nov/2024:00:11:07] "GET /Transacciones HTTP/1.1" 200 -

```

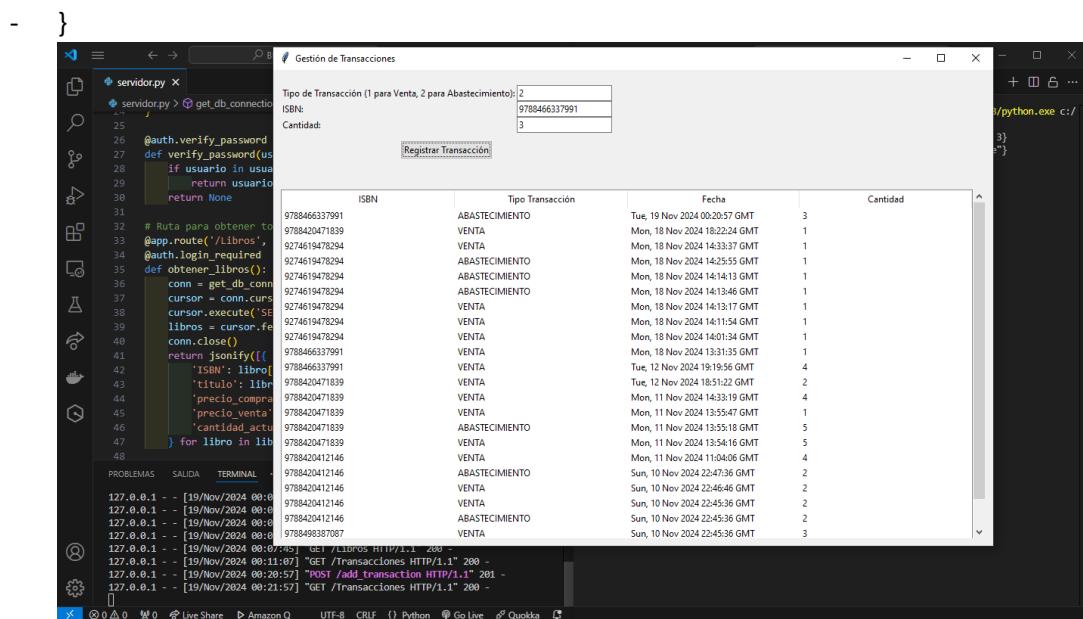
The 'Gestión de Transacciones' application window displays a table of transactions:

ISBN	Tipo Transacción	Fecha	Cantidad
9788420471839	VENTA	Mon, 18 Nov 2024 18:22:24 GMT	1
9274619478294	VENTA	Mon, 18 Nov 2024 14:33:37 GMT	1
9274619478294	ABASTECIMIENTO	Mon, 18 Nov 2024 14:25:55 GMT	1
9274619478294	ABASTECIMIENTO	Mon, 18 Nov 2024 14:14:13 GMT	1
9274619478294	ABASTECIMIENTO	Mon, 18 Nov 2024 14:13:46 GMT	1
9274619478294	VENTA	Mon, 18 Nov 2024 14:13:17 GMT	1
9274619478294	VENTA	Mon, 18 Nov 2024 14:11:54 GMT	1
9274619478294	VENTA	Mon, 18 Nov 2024 14:01:34 GMT	1
9788466337991	VENTA	Mon, 18 Nov 2024 13:11:35 GMT	1
9788466337991	VENTA	Tue, 12 Nov 2024 19:19:56 GMT	4
9788420471839	VENTA	Tue, 12 Nov 2024 18:51:22 GMT	2
9788420471839	VENTA	Mon, 11 Nov 2024 14:33:19 GMT	4
9788420471839	VENTA	Mon, 11 Nov 2024 13:55:47 GMT	1
9788420471839	ABASTECIMIENTO	Mon, 11 Nov 2024 13:55:18 GMT	5
9788420471839	VENTA	Mon, 11 Nov 2024 13:54:16 GMT	5
9788420412146	VENTA	Mon, 11 Nov 2024 11:04:06 GMT	4
9788420412146	ABASTECIMIENTO	Sun, 10 Nov 2024 22:47:36 GMT	2
9788420412146	VENTA	Sun, 10 Nov 2024 22:46:46 GMT	2
9788420412146	VENTA	Sun, 10 Nov 2024 22:45:36 GMT	2
9788420412146	ABASTECIMIENTO	Sun, 10 Nov 2024 22:45:36 GMT	2
9788498387087	VENTA	Sun, 10 Nov 2024 22:45:36 GMT	3
9788420471839	ABASTECIMIENTO	Sun, 10 Nov 2024 22:45:36 GMT	5

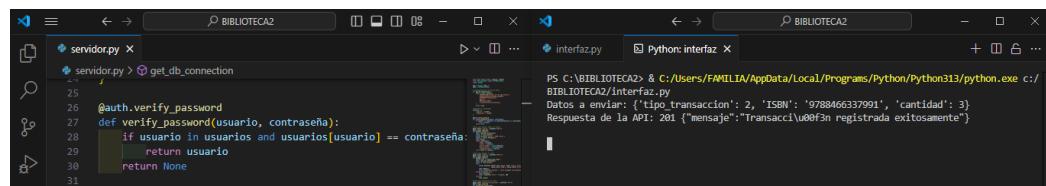
- Cuando le hacemos la petición a la base de datos y servidor ya no lo reconoce como /Libro, si no como /Transacciones como podemos ver nos permite el ingreso y ver las transacciones que hemos realizado, ya sea por medio de venta o abastecimiento, la fecha en la que se hizo la transacción y la cantidad que se vendió o abasteció de un libro

ahora hagamos una prueba muy minúscula cabe aclarar un error que persiste en esta parte cuando queremos hacer una venta o un abastecimiento no toma los valores indicados es decir tengo un libro con 20 unidades vendo una y me deben de quedar 19 unidades pues no, aparece que quedan 18 unidades resta una unidad más lo mismo pasa en abastecimiento den un libro de 5 unidades recargo 1 unidad y recarga 2, aún no sabemos por qué ya miramos y a pesar de tener ayuda de las IA aun no podemos encontrar el error pero de igual manera miremos como funciona.

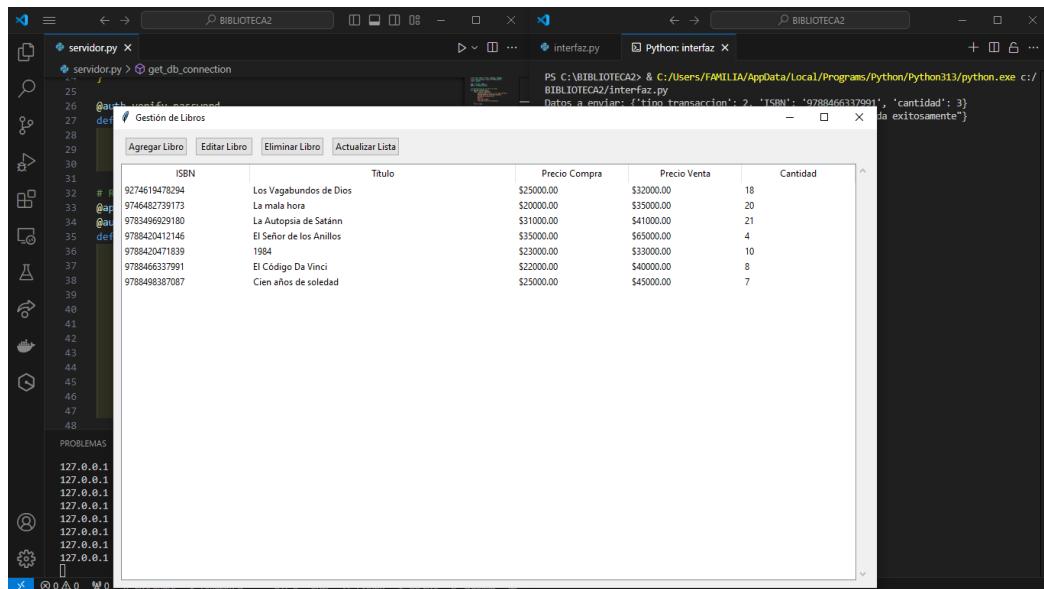
- Vamos a abastecer el libro del **El Código Da Vinci** con el ISBN **9788466337991** conocemos que tiene 2 unidades gracias a la gestión de libro y base de datos verdad, pero vamos a aumentar la cantidad de ejemplares a 3 para el ejemplo



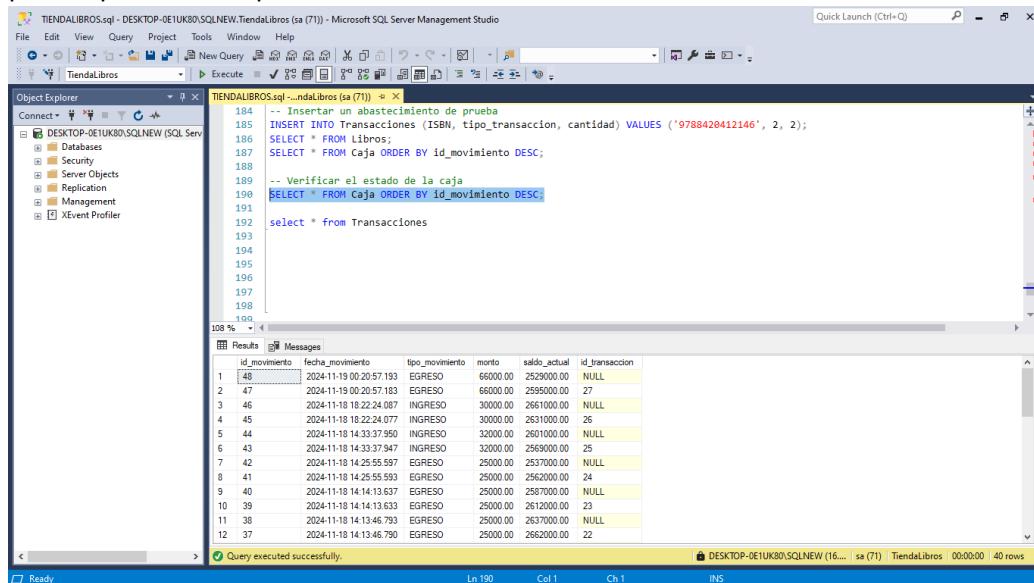
- Ahora bien, una forma de validar sin necesidad de ver la SQL dentro de nuestro panel donde ejecutamos la interfaz esta nos indica que se hizo un cambio en la API realizando un registro y mandando un hermoso mensaje que indica que la transacción fue registrada correctamente.



- Como vemos realiza el abastecimiento de los libros, pero ahora debería de aparecer que tengo 5 libros de esta ejemplar verdad. Pero cuando consultamos a la base de datos o a la gestión de libros aparece que tenemos 8 ejemplares en vez de 5 aún no sabemos por qué esto sucede, pero seguimos trabajando en encontrar el error o la con función porque es como si confundiera los signos de las operaciones.



- Ahora bien, ya por último es ver el estado de caja como su nombre lo indica, nos deja ver cuanto tenemos en caja. Entonces miremos en la base de datos y después una prueba para validar que tenemos.



```

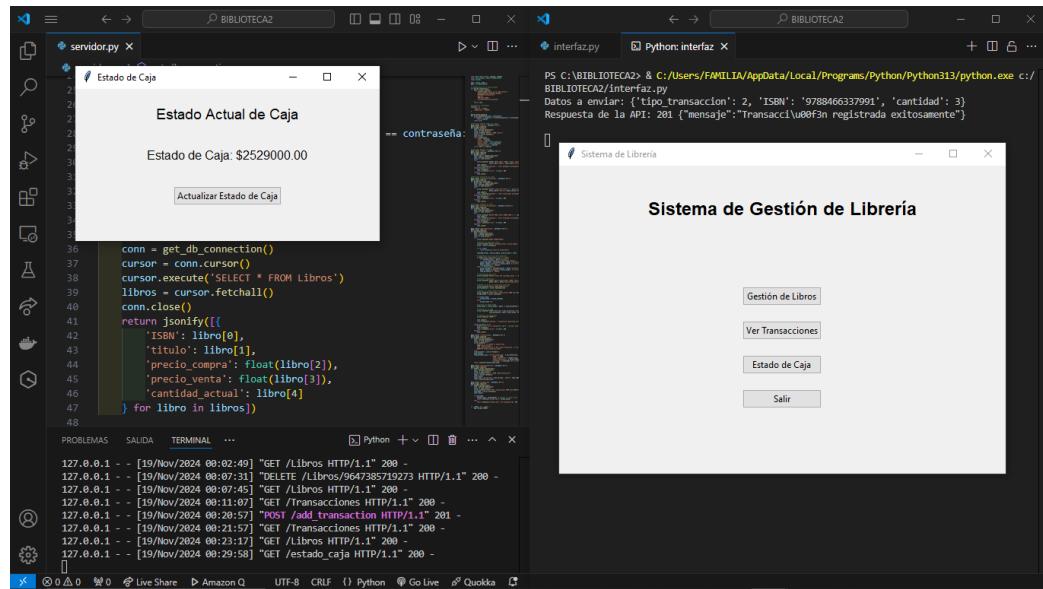
File Edit View Query Project Tools Window Help
File Edit View Project Tools Window Help
Object Explorer TiENDALIBROS.sql - DESKTOP-0E1UK80\SQLNEW.TiendaLibros (sa (7)) - Microsoft SQL Server Management Studio
TiendaLibros
184 -- Insertar un abastecimiento de prueba
185 INSERT INTO Transacciones (ISBN, tipo_transaccion, cantidad) VALUES ('9788420412146', 2, 2);
186
187 SELECT * FROM Libros;
188
189 -- Verificar el estado de la caja
190 SELECT * FROM Caja ORDER BY id_movimiento DESC;
191
192 select * from Transacciones
193
194
195
196
197
198
199

```

	id_movimiento	fecha_movimiento	tipo_movimiento	monto	saldo_actual	id_transaccion
1	48	2024-11-19 20:57:193	EGRESO	66000.00	2529000.00	NULL
2	47	2024-11-19 20:57:183	EGRESO	66000.00	2503000.00	27
3	46	2024-11-18 18:22:24.087	INGRESO	30000.00	2601000.00	NULL
4	45	2024-11-18 18:22:24.077	INGRESO	30000.00	2631000.00	26
5	44	2024-11-18 14:33:37.959	INGRESO	32000.00	2601000.00	NULL
6	43	2024-11-18 14:33:37.947	INGRESO	32000.00	2600000.00	25
7	42	2024-11-18 14:25:55.997	EGRESO	25000.00	2570000.00	NULL
8	41	2024-11-18 14:25:55.993	EGRESO	25000.00	2562000.00	24
9	40	2024-11-18 14:14:13.637	EGRESO	25000.00	2567000.00	NULL
10	39	2024-11-18 14:14:13.633	EGRESO	25000.00	2612000.00	23
11	38	2024-11-18 14:13:46.793	EGRESO	25000.00	2637000.00	NULL
12	37	2024-11-18 14:13:46.790	EGRESO	25000.00	2662000.00	22

- Dentro de nuestra base de datos tenemos que tener 2'529.000 de tanto que hemos hecho transacciones de venta y abastecimiento. Y nos debe de aparecer esta misma cantidad en el estado de caja, nuestro estado de caja no muestra como en la base de datos ya que tomamos la decisión de que solo se vea el dinero de caja por 2 razones

una por estética y la otra es que se confundía el sistema cuando intentamos agregar nuevos campos de valores ya que como en la base de datos la tabla de caja no está como tal relacionada a algo fuerte si no que se maneja por TRIGGERS pues e confundía en ocasiones. Y dandonos resultados como estos.



```

servidor.py
# Código Python para el servidor
# ...

interfaz.py
# Código Python para la interfaz gráfica
# ...

Python: interfaz
# Código Python para la interfaz gráfica
# ...

BIBLIOTECA2
# Log de terminal
127.0.0.1 - - [19/Nov/2024 00:02:49] "GET /Libros HTTP/1.1" 200 -
127.0.0.1 - - [19/Nov/2024 00:07:31] "DELETE /Libros/9647385719273 HTTP/1.1" 200 -
127.0.0.1 - - [19/Nov/2024 00:07:45] "GET /Libros HTTP/1.1" 200 -
127.0.0.1 - - [19/Nov/2024 00:07:46] "PUT /Libros/9647385719273 HTTP/1.1" 200 -
127.0.0.1 - - [19/Nov/2024 00:20:57] "POST /add_transaccion HTTP/1.1" 201 -
127.0.0.1 - - [19/Nov/2024 00:21:57] "GET /transacciones HTTP/1.1" 200 -
127.0.0.1 - - [19/Nov/2024 00:22:17] "GET /Libros HTTP/1.1" 200 -
127.0.0.1 - - [19/Nov/2024 00:29:58] "GET /estado_caja HTTP/1.1" 200 -

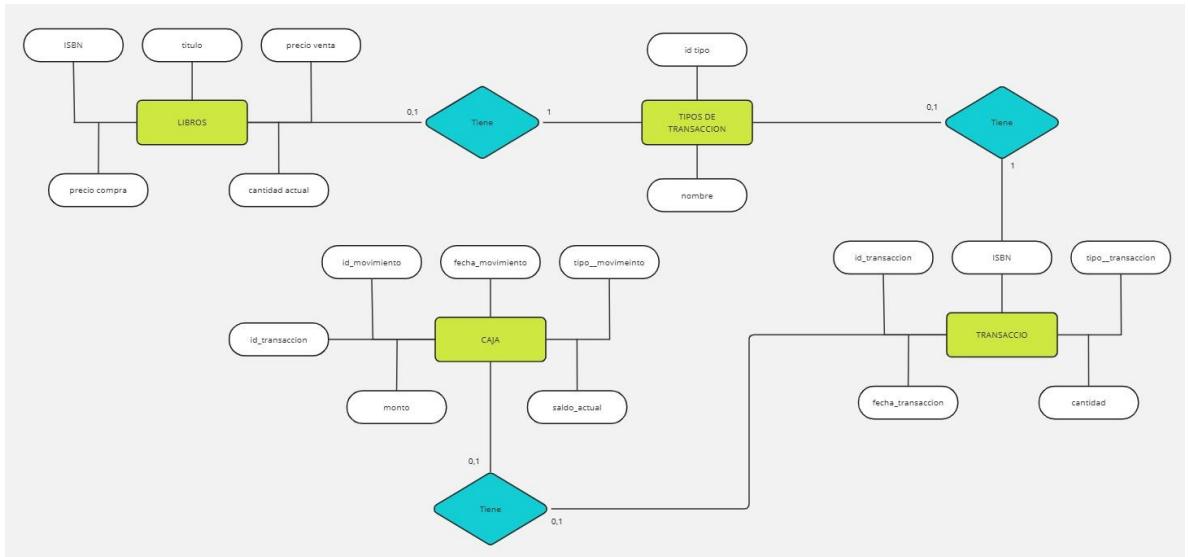
```

## vii. Una sección de preguntas frecuentes

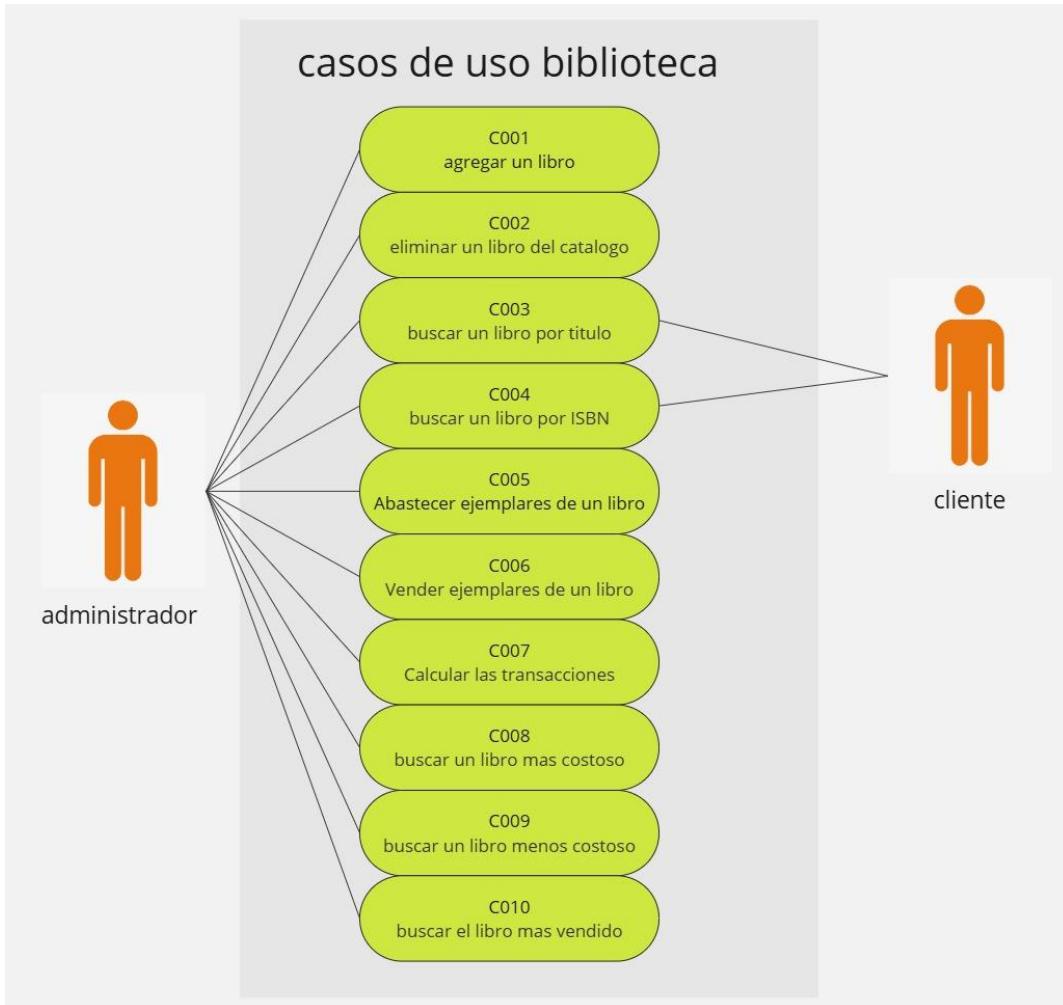
- al momento de que hicimos el programa, este genero miles de preguntas, una de ellas fue como vamos a empezar a crear el programa, en verdad fue muy diverso la verdad fue como de que manera iniciamos el programa o el proyecto si con una base datos relacional o no relacional, cuando teníamos la respuesta empezamos con una SQL relacional algo muy necesario para que esto funcione, la verdad no hicimos tantas preguntas y cuando la probó dos personas pues la verdad no generando dudas, la verdad las preguntas solo fueron al inicio de cómo crear la base de datos o de como se puede generar el servidor y cliente.

### viii. Una sección de documentación técnica con el diseño de software

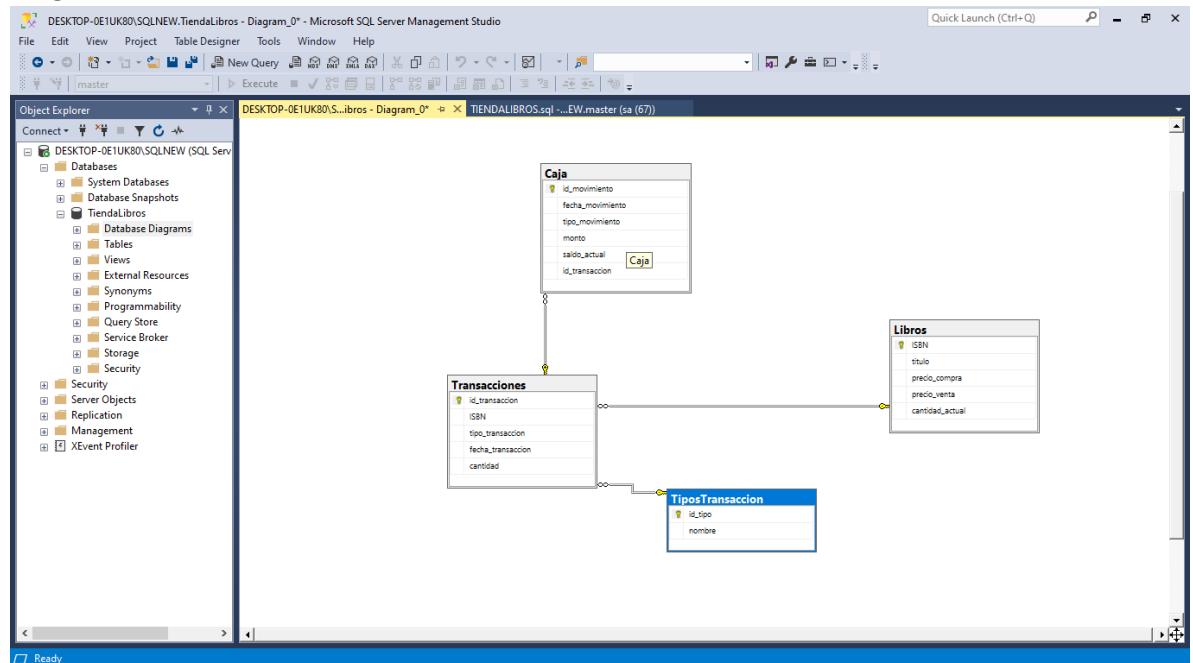
#### a. Diagrama entidad relación



#### b. Diagrama de casos de uso



### c. Diagrama de clases



### ix. Datos de contacto de los autores

- Nosotros creamos este proyecto mi nombre María Camila Bernal Calderón y Sebastian Guerra garzón pertenecemos a la corporación Minuto de Dios pertenecemos al programa de ingeniería en sistemas si les interesa el producto que fue este proyecto, estos son los datos de contacto de míos de mi compañero

- **María Camila Bernal Calderón**
- **TEL: 320 4225556**
- **Correo: [maria.bernal-ca@uniminuto.edu.co](mailto:maria.bernal-ca@uniminuto.edu.co)**
- **También pueden encontrarme en la universidad minuto de dios en la sede san camilo**

- **Sebastian Guerra Garzon**
- **TEL: 319 5617396**
- **Correo: [sebastian.guerra@uniminuto.edu.co](mailto:sebastian.guerra@uniminuto.edu.co)**
- **A mi compañero también lo pueden encontrar en la sede lo pueden encontrarme en la universidad minuto de dios en la sede san camilo**

- a. Entrega el programa con la última versión generada. Verifica que durante la ejecución lo primero que se presente sea los nombres de los autores del programa, tal como la portada de un documento. (Utiliza la instrucción system("cls"); ).

