Manual de Usuario

Proyecto:
Revisión [Numero]

MATERIA

Ingeniería de Software

Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías

Departamento de Ciencias Computacionales

Sistema Gestor de Barbería Online

MANUAL DE USUARIO

CONTENIDO

1. Descripción.

El sistema Barber Shop es una aplicación web desarrollada para la gestión integral de una barbería. Permite a los usuarios registrarse, inciar sesión, reservar citas, consultar barberos disponibles, paquetes de servicios y pre ordenar productos. También ofrece a los administradores la capacidad de gestionar los datos del sistema

Funciones claves del sistema:

Registro e inicio de sesión de usuarios

Reservación de citas con barberos y servicios específicos

Visualización de paquetes y productos

Gestión de usuarios y barberos (backend)

Interfaz gráfica moderna e intuitiva

1.1 Requerimientos

Para el correcto funcionamiento del sistema, el equipo donde se ejecute debe contar con los siguientes requisitos técnicos.

Versiones de SO compatibles: Windows 10 o superior

Ubuntu 20.04 o superior

Bits: 64

Otros requisitos:

Python 3.12 o superior

Navegador Web moderno (Chrome, Firefox, Edge)

MariaDB (como sistema gestor de base de datos)

Visual Studio Code (recomendado)

Conexión a internet (para pruebas o despliegue)

2. Instalación.

Paso A : Clonar el repositorio de GitHub

El primer paso para instalar el sistema es clonar el repositorio desde GitHub usando Git. Esto descargará todos los archivos necesarios a la computadora local

git clone https://github.com/usuario/proyecto-barberia.git

C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Flame>git clone https://github.com/GeraMG0809/BarberManager.git
Cloning into 'BarberManager'...
remote: Enumerating objects: 329, done.
remote: Counting objects: 100% (329/329), done.
remote: Compressing objects: 100% (225/225), done.
remote: Total 329 (delta 154), reused 262 (delta 87), pack-reused 0 (from 0)
Receiving objects: 100% (329/329), 3.27 MiB | 587.00 KiB/s, done.
Resolving deltas: 100% (154/154), done.
```

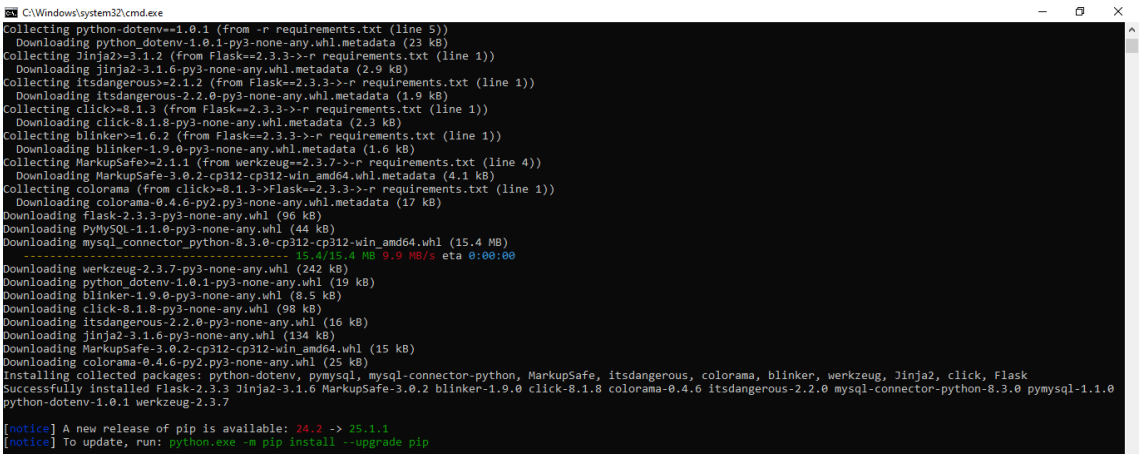
Paso B: Crear entorno virtual e instalar dependencias

Después de clonar el proyecto, se recomienda crear un entorno virtual para asilar dependencias del sistema. Luego, se instalan las librerías necesarias utilizando el archivo requirements.txt ubicado en la carpeta app de la carpeta general del proyecto

Name	Date modified	Type	Size
helpers	5/5/2025 6:45 PM	File folder	
static	5/5/2025 6:45 PM	File folder	
docker-compose	5/5/2025 6:45 PM	Yaml Source File	1 KB
Dockerfile	5/5/2025 6:45 PM	File	1 KB
requirements	5/5/2025 6:45 PM	Text Document	1 KB

C:\Windows\system32\cmd.exe

```
C:\Users\Flame>cd BarberManager

C:\Users\Flame\BarberManager>python -m venv venv
Error: Command '['C:\\Users\\Flame\\BarberManager\\venv\\Scripts\\python.exe', '-m', 'ensurepip', '--upgrade', '--default-pip']' returned non-zero exit status 1.

C:\Users\Flame\BarberManager>python -m venv venv
Error: Command '['C:\\Users\\Flame\\BarberManager\\venv\\Scripts\\python.exe', '-m', 'ensurepip', '--upgrade', '--default-pip']' returned non-zero exit status 1.

C:\Users\Flame\BarberManager>python -m venv venv
Error: Command '['C:\\Users\\Flame\\BarberManager\\venv\\Scripts\\python.exe', '-m', 'ensurepip', '--upgrade', '--default-pip']' returned non-zero exit status 1.

C:\Users\Flame\BarberManager>python -m venv venv

C:\Users\Flame\BarberManager>venv\Scripts\activate

(venv) C:\Users\Flame\BarberManager>pip install -r requirements.txt

[notice] A new release of pip is available: 24.2 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
ERROR: Could not open requirements file: [Errno 2] No such file or directory: 'requirements.txt'

(venv) C:\Users\Flame\BarberManager>cd app

(venv) C:\Users\Flame\BarberManager\app>pip install -r requirements.txt
Collecting Flask==2.3.3 (from -r requirements.txt (line 1))
  Downloading flask-2.3.3-py3-none-any.whl.metadata (3.6 kB)
Collecting pymysql==1.1.0 (from -r requirements.txt (line 2))
  Downloading PyMySQL-1.1.0-py3-none-any.whl.metadata (4.4 kB)
Collecting mysql-connector-python==8.3.0 (from -r requirements.txt (line 3))
  Downloading mysql_connector_python-8.3.0-cp312-cp312-win_amd64.whl.metadata (2.0 kB)
Collecting werkzeug==2.3.7 (from -r requirements.txt (line 4))
  Downloading werkzeug-2.3.7-py3-none-any.whl.metadata (4.1 kB)
Collecting python-dotenv==1.0.1 (from -r requirements.txt (line 5))
  Downloading python_dotenv-1.0.1-py3-none-any.whl.metadata (23 kB)
Collecting Jinja2>=3.1.2 (from Flask==2.3.3->-r requirements.txt (line 1))
  Downloading jinja2-3.1.6-py3-none-any.whl.metadata (2.9 kB)
Collecting itsdangerous>=2.1.2 (from Flask==2.3.3->-r requirements.txt (line 1))
  Downloading itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting click>=8.1.3 (from Flask==2.3.3->-r requirements.txt (line 1))
  Downloading click-8.1.8-py3-none-any.whl.metadata (2.3 kB)
Collecting blinker>=1.6.2 (from Flask==2.3.3->-r requirements.txt (line 1))
  Downloading blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
Collecting MarkupSafe>=2.1.1 (from werkzeug==2.3.7->-r requirements.txt (line 4))
```



 # Crear entorno virtual

python -m venv venv

 # Activar entorno virtual (en Windows)

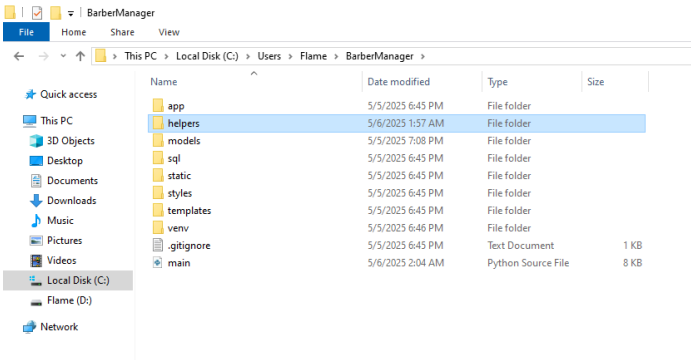
venv\Scripts\activate

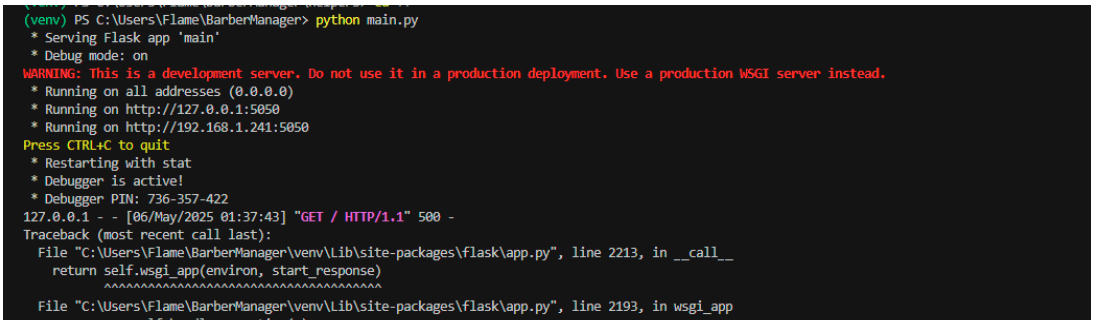
 # Instalar dependencias

pip install -r requirements.txt

3. Ingreso

El ingreso se hace corriendo el archivo main.py desde VSC



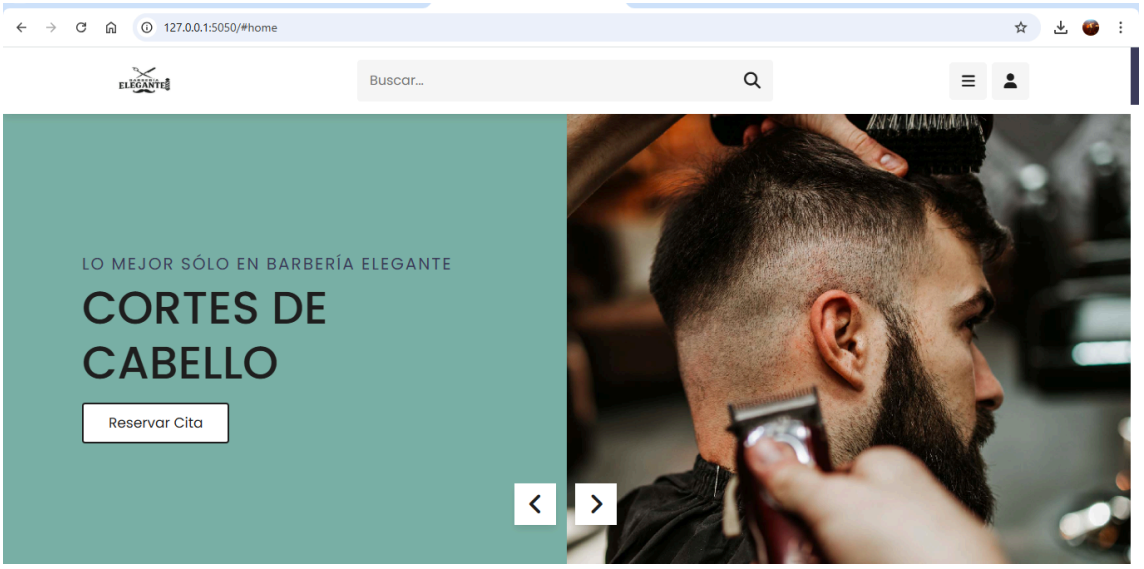


Paso A

Ingresaremos desde el navegador con la URL

http://127.0.0.1:5050

Si todo está correcto se desplegará la ventana principal.



4. Configuración

Configuraremos el entorno virtual para poder acceder a la base de datos y a la unión con el Frontend

4.1 Configuración A

Configurar la conexión a la base de datos

Descripción:
Antes de poder interactuar con la base de datos, es necesario configurar correctamente los datos de acceso dentro del archivo connection.py ubicado en la carpeta helpers.

Esta configuración debe coincidir con los valores definidos en tu instalación local de MariaDB

import pymysql

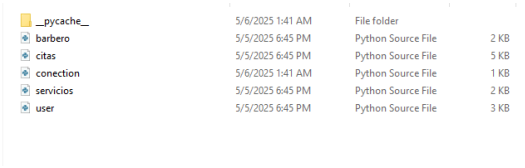
Esta configuración corresponde al archivo connection.py donde se definen las credenciales y parámetros de acceso a la base de datos MariaDB

La finalidad es permitir que el sistema pueda conectarse correctamente a la base de datos barberManager

Paso A

Abre el archivo connection.py

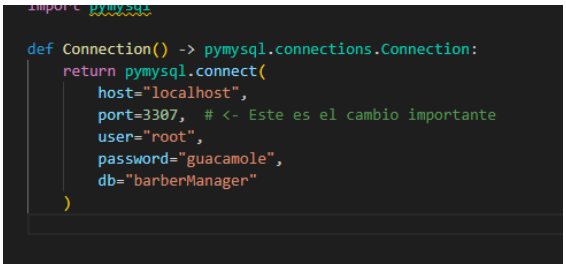
Verifica los parámetros: host, usuario, contraseña y nombre de base de datos



Paso B

```
def Connection() -> pymysql.connections.Connection:
    return pymysql.connect(
        host='localhost',
        user='root',
        password='tu_contraseña_aqui',
        db='barberManager'
    )
```

Recuerda que la contraseña debe ser la que elegiste en tu configuración de SGBD así como el puerto, suele ser el standar 3306



4.2 Configuración B

Esta sección documenta la estructura de la base de datos utilizada por el sistema de barbería. Incluye la creación de las tablas necesarias para gestionar usuarios, barberos, servicios, citas y productos de la tienda

Tiene como finalidad garantizar que la base de datos esté correctamente estructurada y lista para recibir, almacenar y consultar la información de forma eficiente

Paso A

Abrir el gestor de base de datos

Conectarse a lam base de datos barberManager

Ejecutar los scripts SQL para crear las siguientes tablas:

Usuario

Barbero

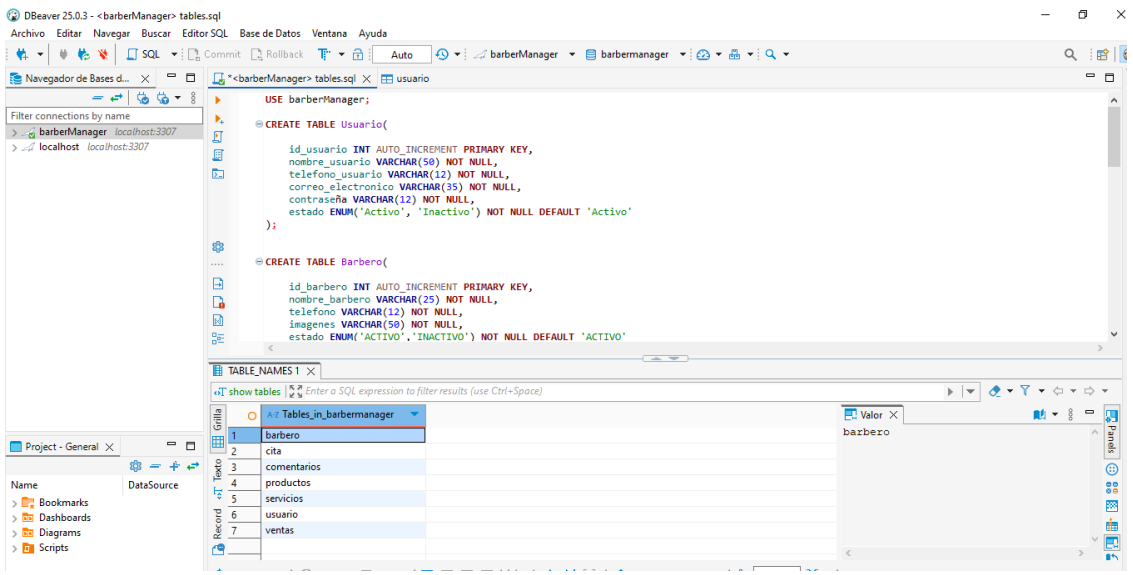
Cita

Servicio

Producto

Detalle_cita

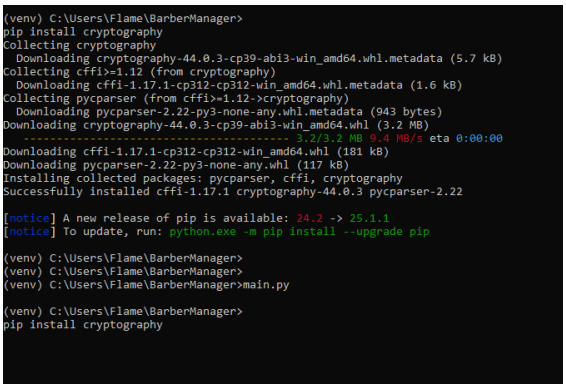
En este apartado es necesario ir metiendo las tablas al sistema aunque de error ya que unas dependen de otras



Paso B

Recordar ejecutar primero el Use de la base de datos para poder agregar esas tablas, y entonces ejecutar pruebas con la interface usando main.py y ejecutandolo desde Visual Studio Code, recordar que también en Visual Studio Code es necesario entrar también al entorno virtual como se explicó anteriormente

También es necesario ejecutar Cryptopraphy para evitar errores futuros esto dentro del CMD o desde el entorno de VSC



5. Funciones

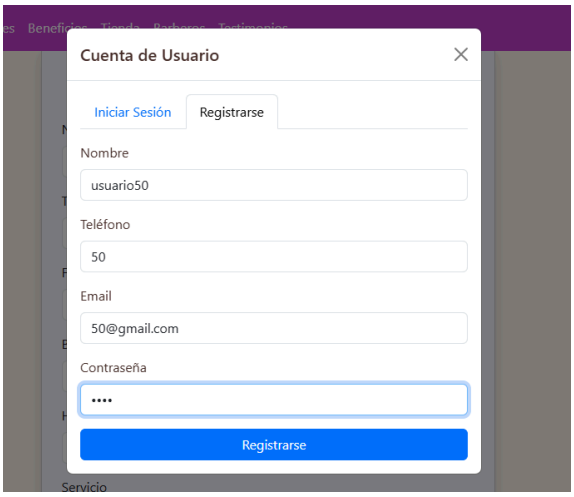
Registro de usuario y Agendar una cita

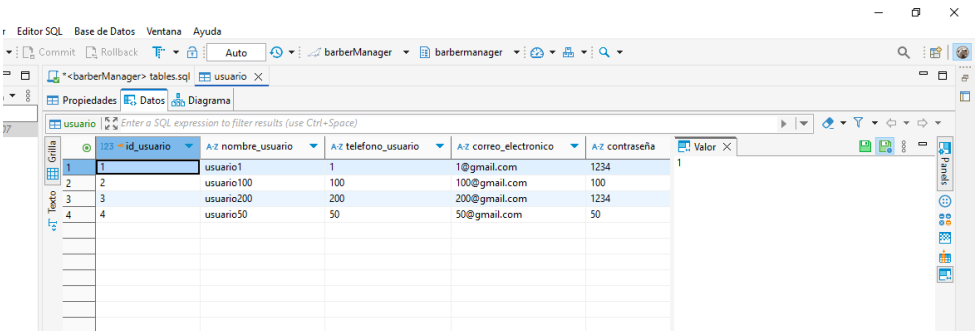
5.1 Función A

Permite a nuevos clientes registrarse en el sistema proporcionando su nombre, correo electrónico, teléfono y contraseña. El sistema verifica si el correo ya existe antes de permitir el registro

- *Pasos para realizar esta función.*
- *Ingresar al formulario de registro desde la página principal*
- *Completar los campos obligatorios (nombre, correo, teléfono, contraseña)*
- *Hacer click en Registrarse*
- *Si el registro es exitoso, el usuario es redirigido al inicio con su sesión activa*
- *En caso de error se mostrará un mensaje de advertencia*

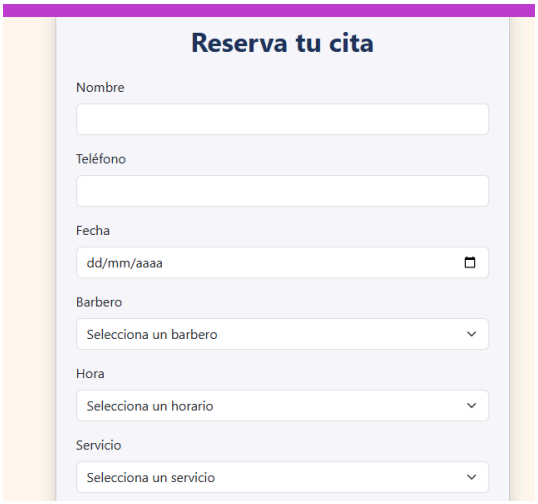

Barbería Elegante
Paquetes
Beneficios
Tienda
Barberos
Testimonios
Beneficios de Elegirnos
Invitado
Corte de cabello profesional
Ambiente relajado y acogedor
Productos de alta calidad
Descuentos en servicios
Reservas prioritarias
Tratamientos exclusivos
Hazte Miembro
Miembro
Corte de cabello profesional
Ambiente relajado y acogedor
Productos de alta calidad
Descuentos en servicios
Reservas prioritarias
Tratamientos exclusivos


Cuenta de Usuario
Iniciar Sesión
Registrarse
Nombre
usuario50
Teléfono
50
Email
50@gmail.com
Contraseña
Registrarse
Servicio


Editor SQL
Base de Datos
Ventana
Ayuda
Commit
Rollback
Auto
barberManager
barbermanager
<barberManager> tables.sql
usuario
Propiedades
Datos
Diagrama
Enter a SQL expression to filter results (use Ctrl+Space)
id_usuario
nombre_usuario
telefono_usuario
correo_electronico
contraseña
Valor
1 1 usuario1 1 1@gmail.com 1234
2 2 usuario100 100 100@gmail.com 100
3 3 usuario200 200 200@gmail.com 1234
4 4 usuario50 50 50@gmail.com 50

5.2 Función B

- *Reservar una cita*
- *Permite al usuario autenticado seleccionar un barbero, una fecha, hora y servicio y crear una cita que queda almacenada en la base de datos*
-
- *Pasos para realizar esta función.*
- *Iniciar sesión en el sistema*
- *Acceder al apartado Reservar Cita*
- *Seleccionar*
- *Nombre del barbero, servicio deseado, y fecha y hora*
- *Confirmar la cita*
- *Recibir mensaje de éxito*



6. Información Adicional.

Cualquier información adicional que facilite la comprensión del documento en sí.