



---

## Manual de Usuario

**Proyecto:**  
Revisión [Número]

MATERIA  
Ingeniería de Software

Universidad de Guadalajara  
Centro Universitario de Ciencias Exactas e  
Ingenierías  
Departamento de Ciencias Computacionales

---



## MANUAL DE USUARIO

Pág.

1

*Sistema Gestor de Barbería Online*

## MANUAL DE USUARIO



## CONTENIDO

### 1. Descripción

#### 1.1 Requerimientos

### 2. Instalación

### 3. Ingreso

### 4. Configuración

#### 4.1 Configuración A

#### 4.2 Configuración B

### 5. Funciones

#### 5.1 Función A

#### 5.2 Función B

### 6. Información Adicional.

#### 1. Descripción.

*El sistema Barber Shop es una aplicación web desarrollada para la gestión integral de una barbería. Permite a los usuarios registrarse, iniciar 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*



## MANUAL DE USUARIO

Pág.

3

### 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
```



## MANUAL DE USUARIO

Pág.

4

```
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 aislar 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 jinja2-3.1.2-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-8.1.3-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-1.6.2-py3-none-any.whl.metadata (1.6 kB)
Collecting MarkupSafe==2.1.1 (from werkzeug==2.3.7->-r requirements.txt (line 4))
  Downloading MarkupSafe-2.1.1-py3-none-any.whl.metadata (1.6 kB)
```



## MANUAL DE USUARIO

Pág.

5

```
C:\Windows\system32\cmd.exe
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>=2.1.2 (from Flask==2.3.3->r requirements.txt (line 1))
  Downloading jinjaz2-3.1.6-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-2.1.2-py3-none-any.whl.metadata (1.9 kB)
Collecting Werkzeug>=2.3.3 (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))
  Downloading MarkupSafe-3.0.2-cp312-cp312-win_amd64.whl.metadata (4.1 kB)
Collecting colorama (from click>=8.1.3->Flask==2.3.3->r requirements.txt (line 1))
  Downloading colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Download Flask-2.3.3-py3-none-any.whl (9 kB)
Download MySQL-connector-python-8.3.0-cp312-cp312-win_amd64.whl (15.4 MB)
----- 15.4/15.4 MB 9.9 MB/s eta 0:00:00
Download werkzeug-2.3.7-py3-none-any.whl (242 kB)
Download python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Download blinker-1.9.0-py3-none-any.whl (8.5 kB)
Download click-8.1.8-py3-none-any.whl (9 kB)
Download itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Download jinjaz2-3.1.6-py3-none-any.whl (134 kB)
Download MarkupSafe-3.0.2-cp312-cp312-win_amd64.whl (15 kB)
Download colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: python-dotenv, pymysql, mysql-connector-python, MarkupSafe, itsdangerous, colorama, blinker, werkzeug, Jinja2, click, Flask
Successfully installed Flask-2.3.3 Jinja2-3.1.6 MarkupSafe-3.0.2 blinker-1.9.0 click-8.1.8 colorama-0.4.6 itsdangerous-2.2.0 pymysql-8.3.0 pymysql-1.1.0
python-dotenv-1.0.1 werkzeug-2.3.7
[notice] A new release of pip is available: 24.2 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

# Crear entorno virtual

```
python -m venv venv
```

# Activar entorno virtual (en Windows)

```
venv\Scripts\activate
```

# Instalar dependencias

```
pip install -r requirements.txt
```



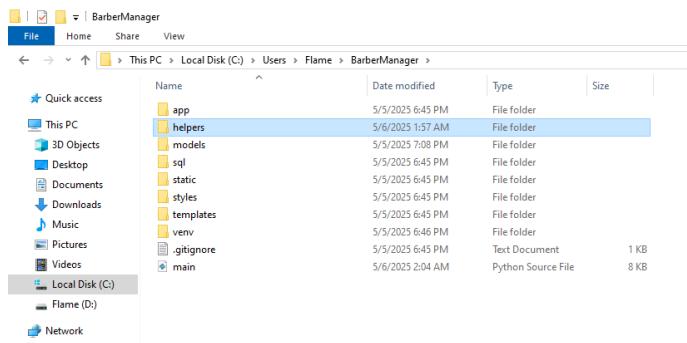
## MANUAL DE USUARIO

Pág.

6

### 3. Ingreso

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



```
(venv) PS C:\Users\Flame\BarberManager> python main.py
* Serving Flask app 'main'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5050
* Running on http://192.168.1.241:5050
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 736-357-422
127.0.0.1 - - [06/May/2025 01:37:43] "GET / HTTP/1.1" 500 -
Traceback (most recent call last):
  File "C:\Users\Flame\BarberManager\venv\Lib\site-packages\flask\app.py", line 2213, in __call__
    return self.wsgi_app(environ, start_response)
           ~~~~~
  File "C:\Users\Flame\BarberManager\venv\Lib\site-packages\flask\app.py", line 2193, in wsgi_app
```

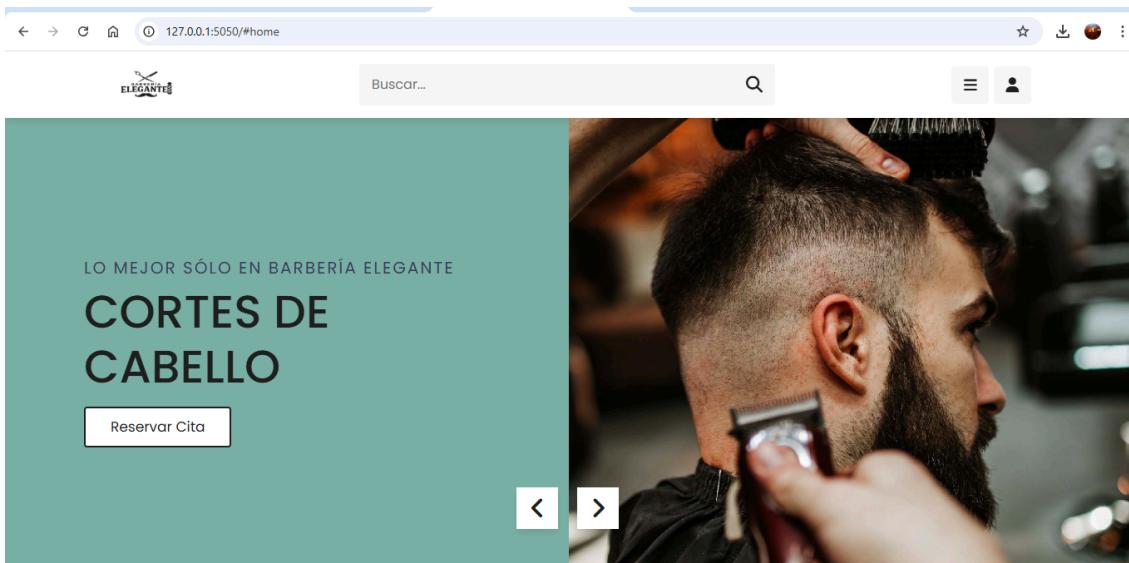
#### Paso A

Ingresaremos desde el navegador con la URL

<http://127.0.0.1:5050>

Si todo está correcto se desplegará la ventana principal.

|   |                          |      |   |
|---|--------------------------|------|---|
|  | <b>MANUAL DE USUARIO</b> | Pág. | 7 |
|---|--------------------------|------|---|



## 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*



## MANUAL DE USUARIO

Pág.

8

*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

|             |                  |                    |
|-------------|------------------|--------------------|
| __pycache__ | 5/6/2025 1:41 AM | File folder        |
| barbero     | 5/5/2025 6:45 PM | Python Source File |
| citas       | 5/5/2025 6:45 PM | Python Source File |
| conexion    | 5/6/2025 1:41 AM | Python Source File |
| servicios   | 5/5/2025 6:45 PM | Python Source File |
| user        | 5/5/2025 6:45 PM | Python Source File |

### 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



## MANUAL DE USUARIO

Pág.

9

```
import pymysql
def Connection() -> pymysql.connections.Connection:
    return pymysql.connect(
        host="localhost",
        port=3307, # <- Este es el cambio importante
        user="root",
        password="guacamole",
        db="barberManager"
    )
```

## 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 la base de datos barberManager

Ejecutar los scripts SQL para crear las siguientes tablas:

Usuario

Barbero

Cita

Servicio

Producto

Detalle\_cita



## MANUAL DE USUARIO

Pág.

10

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

```
USE barberManager;
CREATE TABLE Usuario(
    id_usuario INT AUTO_INCREMENT PRIMARY KEY,
    nombre_usuario VARCHAR(50) NOT NULL,
    telefono_usuario VARCHAR(12) NOT NULL,
    correo_electronico VARCHAR(35) NOT NULL,
    contraseña VARCHAR(12) NOT NULL,
    estado ENUM('Activo', 'Inactivo') NOT NULL DEFAULT 'Activo'
);
CREATE TABLE Barbero(
    id_barbero INT AUTO_INCREMENT PRIMARY KEY,
    nombre_barbero VARCHAR(25) NOT NULL,
    telefono_barbero VARCHAR(12) NOT NULL,
    imagenes VARCHAR(50) NOT NULL,
    estado ENUM('ACTIVO', 'INACTIVO') NOT NULL DEFAULT 'ACTIVO'
);
```

| Nombre      | Valor |
|-------------|-------|
| barbero     |       |
| cita        |       |
| comentarios |       |
| productos   |       |
| servicios   |       |
| usuario     |       |
| ventas      |       |

### 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 Cryptography para evitar errores futuros esto dentro del CMD o desde el entorno de VSC



## MANUAL DE USUARIO

Pág.

11

```
(venv) C:\Users\Flame\BarberManager>pip install cryptography
Collecting cryptography
  Downloading cryptography-44.0.3-cp39-abi3-win_amd64.whl.metadata (5.7 kB)
    Collecting cffi==1.12 (from cryptography)
      Downloading cffi-1.17.1-cp312-cp312-win_amd64.whl.metadata (1.6 kB)
        Collecting pycparser (from cffi->1.12->cryptography)
          Downloading pycparser-2.22-py3-none-any.whl.metadata (943 bytes)
            Downloading cryptography-44.0.3-cp39-abi3-win_amd64.whl (3.2 MB)
              100% |██████████| 3.2 MB 6.0 MB/s eta 0:00:00
Downloaded cffi-1.17.1-cp312-cp312-win_amd64.whl (181 kB)
Downloaded pycparser-2.22-py3-none-any.whl (117 kB)
Installing collected packages: pycparser, cffi, cryptography
Successfully installed cffi-1.17.1 cryptography-44.0.3 pycparser-2.22
[notice] A new release of pip is available: 24.2 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(venv) C:\Users\Flame\BarberManager>
(venv) C:\Users\Flame\BarberManager>
(venv) C:\Users\Flame\BarberManager>main.py
(venv) C:\Users\Flame\BarberManager>
pip install cryptography
```

## 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*



## MANUAL DE USUARIO

Pág.

12

Barbería Elegante Paquetes Beneficios Tienda Barberos Testimonios

### Beneficios de Elegirnos

| Invitado                       | Miembro                        |
|--------------------------------|--------------------------------|
| ✓ Corte de cabello profesional | ✓ Corte de cabello profesional |
| ✓ Ambiente relajado y acogedor | ✓ Ambiente relajado y acogedor |
| ✓ Productos de alta calidad    | ✓ Productos de alta calidad    |
| ✗ Descuentos en servicios      | ✗ Descuentos en servicios      |
| ✗ Reservas prioritarias        | ✓ Reservas prioritarias        |
| ✗ Tratamientos exclusivos      | ✓ Tratamientos exclusivos      |

[Hazte Miembro](#)

Nombre: Barbería Elegante Paquetes Beneficios Tienda Barberos Testimonios

Cuenta de Usuario

Iniciar Sesión Registrarse

Nombre: usuario50

Teléfono: 50

Email: 50@gmail.com

Contraseña: \*\*\*\*

[Registrarse](#)

Editor SQL Base de Datos Ventana Ayuda

Commit Rollback Auto barberManager barberman

Propiedades Datos Diagrama

usuario Enter a SQL expression to filter results (use Ctrl+Space)

| id_usuario | nombre_usuario | telefono_usuario | correo_electronico | contraseña |
|------------|----------------|------------------|--------------------|------------|
| 1          | usuario1       | 1                | 1@gmail.com        | 1234       |
| 2          | usuario100     | 100              | 100@gmail.com      | 100        |
| 3          | usuario200     | 200              | 200@gmail.com      | 1234       |
| 4          | usuario50      | 50               | 50@gmail.com       | 50         |



## MANUAL DE USUARIO

Pág.

13

### 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

Reserva tu cita

Nombre

Teléfono

Fecha  
 dd/mm/aaaa

Barbero

Hora

Servicio

### 6. Información Adicional.

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