Manual del Programador

Nombre: jhonny durango , Miguel Quimis , Leonardo Padilla

Este manual está dirigido a desarrolladores que deseen entender, modificar o extender el sistema de Gestión de Clientes y Facturas basado en Flask y MySQL.

1.1 Tecnologías Utilizadas

Backend: Python (Flask)

Frontend: HTML, CSS, Jinja2 (plantillas)

Base de datos: MySQL

Control de versiones: Git (opcional, si se usa GitHub)

1.2 Estructura del Proyecto

El sistema sigue una arquitectura MVC (Modelo-Vista-Controlador).

Carpeta/Archivo	Descripción
app.py	Aplicación principal Flask (Controlador)
templates/	Contiene las vistas HTML (index.html, factura.html, todasFacturas.html)
static/ (opcional)	CSS, JS e imágenes (no usado actualmente)
database/ (opcional)	Scripts SQL para configuración inicial

2. Configuración del Entorno

2.1 Requisitos Previos

Python 3.8+

MySQL Server

Bibliotecas necesarias (flask, mysql-connector-python)

2.2 Instalación

Clonar el repositorio (si aplica):

sh

git clone [URL_DEL_REPOSITORIO]

cd nombre-del-proyecto

Crear un entorno virtual (recomendado):

sh

python -m venv venv

source venv/bin/activate # Linux/Mac

venv\Scripts\activate   # Windows

Instalar dependencias:

sh

pip install flask mysql-connector-python

Configurar la base de datos:

Ejecutar el script SQL proporcionado (database/schema.sql).

Asegurarse de que la configuración en app.py coincida con las credenciales de MySQL.

Ejecutar la aplicación:

sh

python app.py

La aplicación estará disponible en http://localhost:3000.

3. Base de Datos

3.1 Diagrama Entidad-Relación (ER)

El sistema maneja las siguientes tablas:

Tabla Descripción

cliente Almacena información de clientes (nombre, apellido, cédula)

factura Registra facturas asociadas a clientes

entradaItems dentro de una factura (relacionados con categoría y tipo de carrera)

categoria Tipos de categorías con precios

tipoCarrera Tipos de carrera con precios

3.2 Consultas Importantes

Obtener todas las facturas:

sql

SELECT * FROM datoslistafacturas ORDER BY idFactura;

Obtener detalles de una factura:

sql

SELECT * FROM datosEntradasfactura WHERE idFactura = [ID_FACTURA];

4. Rutas y Funcionalidades (Flask)

Ruta Método Descripción

/ GET Muestra formularios para clientes y facturas

/agregarUsuario POST Registra un nuevo cliente

/addFactura POST Crea una nueva factura

/verfacturas GET Lista todas las facturas

/factura/<int:factura_id> GET Muestra detalles de una factura

/cerrarfactura POST Cierra una factura (cambia estado)

/datosaFactura POST Agrega items a una factura

5. Personalización y Extensión

5.1 Agregar Nuevas Funcionalidades

Ejemplo: Añadir búsqueda de clientes

Modificar app.py para incluir una nueva ruta:

python

@app.route('/buscarCliente', methods=['GET'])

def buscar_cliente():

```
    query = request.args.get('q')
    connection = get_db_connection()
    cursor = connection.cursor(dictionary=True)
    cursor.execute("SELECT * FROM cliente WHERE nameCliente LIKE %s", (f"%{query}%",))
    resultados = cursor.fetchall()
    connection.close()
    return render_template('resultados_busqueda.html', clientes=resultados)
```

5.2 Mejoras de Seguridad

Validar entradas: Usar flask_wtf para formularios seguros.

Protección CSRF: Habilitar flask_wtf.csrf.CSRFProtect.