

## **Manual técnico — ¿cómo lo vamos a estructurar?**

- 1. Resumen del Proyecto**
- 2. Requisitos del Sistema**
- 3. Estructura de Directorios y Archivos**
- 4. Entorno de Desarrollo (Pipenv)**
- 5. Configuración Inicial**
- 6. Ejecución del Proyecto**
- 7. Descripción de Módulos Funcionales (como AdminFrame)**
- 8. Base de Datos y CRUD**



0983613267



[www.istg.edu.ec](http://www.istg.edu.ec)



ISTGtv



# 1. Resumen General del Proyecto

**Nombre del proyecto:** *Proyecto Médico* **Tipo de aplicación:** Sistema de gestión médica con interfaz gráfica. **Objetivo principal:** Desarrollar una aplicación modular que permita administrar eficientemente la información de un centro médico. El sistema cubre funcionalidades como gestión de pacientes, recetas médicas, personal administrativo y registro de datos clínicos en tiempo real.

## Características destacadas:

- Interfaz gráfica construida con `customtkinter`.
- Navegación fluida mediante estructura `switchFrame` con cinco secciones principales.
- Integración directa entre GUI y operaciones CRUD a través de `pymysql`.
- Generación automática de recetas en formatos exportables.
- Arquitectura modular para facilitar el mantenimiento y escalabilidad.

## Usuarios objetivo:

- Personal administrativo
- Médicos
- Asistentes clínicos

**Motivación del desarrollo:** Este proyecto nace de la necesidad de organizar digitalmente los procesos clínicos de forma intuitiva y funcional, aprovechando la potencia de Python y sus librerías para GUI y bases de datos.



0983613267



[www.istg.edu.ec](http://www.istg.edu.ec)



ISTGtv



## 2. Entorno de Desarrollo

### IDE utilizado:

- PyCharm, por su soporte avanzado para proyectos Python, gestión de entornos virtuales y navegación entre módulos.

### Lenguaje de programación:

- Python, por su versatilidad, facilidad de integración con bases de datos y herramientas gráficas.

### Gestión del entorno virtual:

- **Pipenv**
  - Se utiliza para crear y administrar el entorno virtual del proyecto.
  - Facilita el manejo de dependencias mediante los archivos **Pipfile** y **Pipfile.lock**.
  - Mejora la reproducibilidad entre distintos equipos y sistemas operativos.

### Dependencias principales del proyecto:

Librería / Herramienta	Uso principal
<code>customtkinter</code>	Desarrollo de la interfaz gráfica moderna.
<code>pymysql</code>	Conexión directa con base de datos MySQL.
<code>os</code> / <code>sys</code>	Operaciones del sistema y control de entorno.
<code>datetime</code>	Registro de fechas en recetas y formularios.
<code>reportlab</code> (opcional)	Generación de PDFs para recetas médicas.

### Ventajas del entorno configurado:

- Separación limpia entre frontend (GUI) y backend.
- Control total sobre librerías usadas y sus versiones.
- Aislamiento del proyecto respecto a otros entornos Python en el sistema.

## 3. Estructura de Directorios y Archivos



0983613267



www.istg.edu.ec



ISTGtv



El proyecto está organizado en múltiples módulos que separan claramente la lógica backend, la interfaz gráfica y los recursos multimedia, permitiendo escalabilidad y mantenimiento eficiente.

## Directorios Principales

Carpeta	Propósito
<code>backend/</code>	Lógica del sistema, conexión con la base de datos, funciones administrativas.
<code>frontend/</code>	Interfaz gráfica desarrollada con <code>customtkinter</code> .
<code>datos/</code>	Archivos de entrada/salida relacionados con información del sistema.
<code>fichero/</code>	Gestión de datos internos, posiblemente utilizados por módulos específicos.
<code>img/</code>	Recursos visuales para la GUI, como íconos y fondos.
<code>recetas_generadas/</code>	Carpeta destino para recetas médicas generadas automáticamente.

## Archivos Clave

Archivo	Descripción
<code>main.py</code>	Archivo principal del sistema. Implementa la lógica <code>switchFrame</code> que controla la navegación entre cinco módulos.
<code>AdminFrame.py</code>	Módulo que gestiona funciones administrativas como control de usuarios o registros médicos.
<code>Pipfile</code>	Archivo que define las dependencias del proyecto para Pipenv.
<code>Pipfile.lock</code>	Control de versiones exactas de librerías para reproducibilidad.

## Estructura de Frames en `main.py`

- Se implementó un sistema dinámico de navegación llamado `switchFrame`.
- Cada botón en la interfaz principal carga un frame específico que representa una sección funcional (ej. Inicio, Pacientes, Recetas, etc.).
- Esta técnica modular permite cargar sólo la interfaz necesaria,



0983613267



[www.istg.edu.ec](http://www.istg.edu.ec)



ISTGtv



CAMPUS COLEGIO GUAYAQUIL - Gómez Rendón 1403 y Av. Machala  
CAMPUS CENTRO MÚLTIPLE DE INSTITUTOS - Av. Quito y Padre Solano



## 4. Estructura de Navegación — **switchFrame**

El sistema implementa una lógica modular mediante **frames** intercambiables que se activan según el botón seleccionado por el usuario. Esta estructura permite una experiencia fluida y mantiene la interfaz limpia al mostrar únicamente una sección funcional a la vez.

### Componentes principales

- **Ventana raíz (root)**: Contenedor principal de toda la interfaz.
- **Botones de navegación**: Cinco botones principales que permiten cambiar entre los distintos frames del sistema.
- **Frames funcionales**: Cada módulo visual se implementa como un **Frame** personalizado y se oculta o muestra dinámicamente.

### Lógica general del **switchFrame**

Cuando el usuario interactúa con un botón, se ejecuta una función que:

1. Oculta todos los frames activos.
2. Muestra el frame correspondiente a la sección seleccionada.

Esto garantiza que sólo una vista esté activa a la vez, optimizando recursos y mejorando la usabilidad.

### Pseudocódigo referencial



0983613267

[www.istg.edu.ec](http://www.istg.edu.ec)

ISTGtv



Plaintext

Inicializar ventana principal

Definir función `switch_frame(nombre):`

Para cada frame en `lista_frames:`

Ocultar frame

Mostrar frame que coincide con `'nombre'`

Crear lista de frames:

`frame_inicio`

`frame_pacientes`

`frame_doctores`

`frame_recetas`

`frame_configuracion`

Crear botones de navegación:

Botón 'Inicio' -> `switch_frame("inicio")`

Botón 'Pacientes' -> `switch_frame("pacientes")`

Botón 'Doctores' -> `switch_frame("doctores")`

Botón 'Recetas' -> `switch_frame("recetas")`

Botón 'Configuración' -> `switch_frame("configuracion")`

Mostrar ventana

## Relación con módulos

- `main.py`: Centraliza la lógica `switchFrame` y crea la ventana principal.
- `AdminFrame.py` y otros frames: Se integran como elementos dentro de esta estructura y son cargados dinámicamente según la sección activa

## 5. Descripción de Módulos Funcionales

Cada módulo está diseñado como un `Frame` independiente que se integra en el sistema principal mediante `switchFrame`. A continuación, se detalla el propósito y lógica de los más representativos.

### `AdminFrame.py` — Módulo de Administración



0983613267



[www.istg.edu.ec](http://www.istg.edu.ec)



ISTGtv



CAMPUS COLEGIO GUAYAQUIL - Gómez Rendón 1403 y Av. Machala  
CAMPUS CENTRO MÚLTIPLE DE INSTITUTOS - Av. Quito y Padre Solano



**Propósito:** Gestionar funciones administrativas como visualización de usuarios, edición de datos médicos, y control de accesos desde la interfaz gráfica.

**Características técnicas:**

- Integración directa con la base de datos mediante **pymysql**.
- Uso de widgets como **Treeview**, **ComboBox**, botones y campos de entrada.
- Generación automática de formularios.
- Validación de datos antes del envío a la base de datos.

**Flujo de operación:**

Al iniciar AdminFrame:

- Conectar con la base de datos
- Cargar datos de usuarios en Treeview
- Activar ComboBox con roles of especialidades
- Escuchar eventos de edición/eliminación
- Ejecutar operaciones CRUD al confirmar

**Funciones clave:**

- **cargar\_datos()**: Consulta SQL para visualizar registros.
- **insertar\_registro()**: Añadir nuevos usuarios desde GUI.
- **actualizar\_registro()**: Editar datos existentes.
- **eliminar\_registro()**: Eliminar entradas específicas.
- **sincronizar\_comboBox()**: Actualizar ComboBox dinámicamente.

**Otros Frames Funcionales**



0983613267



[www.istg.edu.ec](http://www.istg.edu.ec)



ISTGtv



Frame	Función principal
<code>PacientesFrame.py</code>	Registro, búsqueda y edición de datos de pacientes
<code>DoctoresFrame.py</code>	Administración del personal médico
<code>RecetasFrame.py</code>	Generación y visualización de recetas médicas
<code>ConfiguracionFrame.py</code>	Ajustes del sistema y del entorno de trabajo

## Base de Datos

El sistema utiliza una base de datos MySQL, conectada mediante la librería `pymysql`, para gestionar la información de pacientes, doctores, usuarios y recetas. La conexión se establece desde los módulos backend como `AdminFrame.py`, permitiendo operaciones CRUD en tiempo real.

### Estructura de Tablas (Ejemplo)

Tabla	Campos clave	Propósito funcional
<code>usuarios</code>	<code>id_usuario</code> , nombre, rol, contraseña	Gestión de usuarios del sistema
<code>pacientes</code>	<code>id_paciente</code> , nombre, edad, historial_clinico	Registro de pacientes del centro médico
<code>doctores</code>	<code>id_doctor</code> , nombre, especialidad, contacto	Información del personal médico
<code>recetas</code>	<code>id_receta</code> , fecha, <code>id_paciente</code> , diagnóstico	Almacenamiento y generación de recetas médicas

### Conexión a la base de datos

Librería usada: `pymysql` Método común:

- Establecer conexión con los parámetros: host, usuario, contraseña, nombre de base de datos.
- Ejecutar comandos SQL desde funciones dentro de cada módulo (ej. insertar, actualizar, eliminar).



0983613267



www.istg.edu.ec



ISTGtv



CAMPUS COLEGIO GUAYAQUIL - Gómez Rendón 1403 y Av. Machala  
CAMPUS CENTRO MÚLTIPLE DE INSTITUTOS - Av. Quito y Padre Solano





## Python

```
import pymysql

def conectar():
    return pymysql.connect(
        host='localhost',
        user='root',
        password='tu_clave',
        database='proyecto_medico',
        cursorclass=pymysql.cursors.DictCursor
    )
```

## 7. Descripción de Módulos Funcionales

Cada módulo funcional está encapsulado como un **Frame** de la interfaz gráfica, siguiendo un patrón modular que facilita su integración con el sistema principal (**main.py**) mediante la estructura **switchFrame**.

### AdminFrame.py — Módulo Administrativo

#### Propósito funcional:

- Gestionar usuarios del sistema y su información (ej. nombre, rol, credenciales).
- Interacción directa con la base de datos para ejecutar operaciones CRUD desde la GUI.

#### Componentes clave del módulo:

- **Treeview**: Para mostrar datos tabulados en tiempo real.
- **Entry, Label, ComboBox**: Para captura y visualización de datos.
- **Button**: Para ejecutar acciones como insertar, actualizar o eliminar registros.
- **Frame** contenedor: Organiza visualmente los elementos.



0983613267



www.istg.edu.ec



ISTGtv



```

Al iniciar AdminFrame:
    Conectar con la BD
    Cargar todos los usuarios en Treeview
    Inicializar ComboBox con roles disponibles

Botón 'Insertar':
    Validar campos
    Ejecutar INSERT en la BD
    Actualizar Treeview

Botón 'Editar':
    Obtener usuario seleccionado
    Aplicar cambios
    Ejecutar UPDATE

Botón 'Eliminar':
    Confirmar acción
    Ejecutar DELETE
    Actualizar Treeview
    
```

## Base de Datos y Operaciones CRUD

### Modelo relacional

La base de datos implementa varias tablas interconectadas para estructurar la información de pacientes, doctores, usuarios del sistema y recetas médicas. Estas tablas se relacionan por claves primarias (**id**) y, en algunos casos, por claves foráneas (FK).

Tabla	Campos principales	Descripción
<b>usuarios</b>	id_usuario, nombre, rol, contraseña	Datos de acceso y control administrativo
<b>pacientes</b>	id_paciente, nombre, edad, historial_clínico	Registro clínico de pacientes
<b>doctores</b>	id_doctor, nombre, especialidad, contacto	Información sobre médicos del centro
<b>recetas</b>	id_receta, fecha, id_paciente, diagnóstico	Documentación de recetas médicas emitidas



0983613267



[www.istg.edu.ec](http://www.istg.edu.ec)



ISTGtv



CAMPUS COLEGIO GUAYAQUIL - Gómez Rendón 1403 y Av. Machala  
CAMPUS CENTRO MÚLTIPLE DE INSTITUTOS - Av. Quito y Padre Solano

