Tecnologías
MediCare Desk

Cada dosis, una muestra de cuidado

Presentado por:

Yamid Alfonso Gonzalez Torres

Jenny Catherine Herrera Garzon

Edwin Andres Marin Vanegas

Diego Steven Pinzon Yossa

Profesor:

Oscar Eduardo Alvarez Rodriguez

Universidad Nacional de Colombia

Facultad de Ingeniería

Ingeniería de software

2025

TECNOLOGÍAS SELECCIONADAS Y JUSTIFICACIÓN TÉCNICA

I. Lenguaje de Programación y Entorno de Desarrollo

Para el desarrollo de la aplicación **MediCareDesk** se seleccionó el lenguaje de programación Python junto con el módulo Tkinter como herramienta para la construcción de la interfaz gráfica. Esta decisión responde a criterios de eficiencia, compatibilidad con los requerimientos del curso y experiencia previa del equipo.

TABLA I.

COMPARATIVO DE TECNOLOGÍAS DE INTERFAZ GRÁFICA

Tecnología	Ventajas	Desventajas
Python + Tkinter	Ligero, fácil de aprender, multiplataforma, ideal para desarrollo local	Interfaz básica, menos personalizable que otras alternativas
Java + JavaFX	Interfaz moderna, control gráfico avanzado, tipado fuerte	Mayor complejidad y requerimientos de configuración
C# + WinForms/WPF	Buen rendimiento en Windows, herramientas visuales robustas	No multiplataforma, requiere conocimiento específico de C#

II. Justificación de la Selección

Python fue elegido por su sintaxis clara, amplia documentación y familiaridad del equipo. Tkinter, al ser parte del paquete estándar de Python, permite el desarrollo de interfaces gráficas sin

instalar dependencias externas, lo cual es compatible con los requerimientos del curso y el objetivo de mantener la aplicación completamente funcional en entornos locales. Dado que el foco principal del proyecto es la lógica de negocio, la interfaz visual simple de Tkinter es suficiente para esta primera versión funcional.

III. Base de Datos Relacional

Se seleccionó **SQLite** como sistema de gestión de bases de datos relacional debido a su facilidad de uso, portabilidad, y compatibilidad directa con Python a través de bibliotecas como `sqlite3`. Esta elección se alinea con los requerimientos de un entorno de ejecución local, sin necesidad de configurar un servidor externo.

TABLA II.

COMPARATIVO DE TECNOLOGÍAS EVALUADAS

Tecnología	Ventajas	Desventajas
SQLite	Ligera, no necesita instalación, ideal para almacenamiento embebido	Limitada en concurrencia y escalabilidad
MySQL	Robusta, ampliamente utilizada, buen soporte para relaciones complejas	Requiere configuración de servidor
PostgreSQL	Alta potencia y flexibilidad, soporte para tipos avanzados de datos	Mayor complejidad de gestión en proyectos pequeños

IV. Justificación de la Selección

La elección de **SQLite** responde a la necesidad de implementar una solución funcional que pueda ejecutarse de forma **local**, sin requerir instalación de servidores o configuraciones adicionales. Al ser un sistema **liviano y embebido**, permite simplificar el desarrollo y despliegue del proyecto en

diferentes dispositivos durante el curso. Aunque MySQL y PostgreSQL ofrecen ventajas en términos de escalabilidad y soporte de múltiples conexiones concurrentes, para un sistema monousuario con una interfaz de escritorio desarrollada en Python y Tkinter, SQLite proporciona una alternativa adecuada, funcional y de bajo mantenimiento. Además, permite conservar la **estructura relacional** necesaria para modelar adecuadamente la lógica de pacientes, tratamientos, medicamentos y tomas.

V. Bibliotecas y Herramientas Complementarias

Durante el desarrollo del sistema se emplearán herramientas que complementan el lenguaje base, garantizando el cumplimiento de los requerimientos funcionales y no funcionales del proyecto

Herramienta/ Librería	**Propósito**
Tkinter	Interfaz gráfica de escritorio
sqlite3	Conexión entre Python y la base de datos SQLite
datetime	Manejo de fechas y horas para programación de tomas
schedule / threading	Ejecución de recordatorios y procesos en segundo plano
reportlab / fpdf	Generación de reportes en formato PDF
pytest / unittest	Pruebas unitarias de componentes del sistema
pylint/flake8	Validación de estilo y estructura del código fuente

VI. Justificación

Las herramientas seleccionadas son de **código abierto**, **ligeras** y **compatibles con Python**, lo que garantiza una integración fluida en entornos locales sin depender de infraestructura externa. La biblioteca **sqlite3** permite trabajar con una base de datos relacional de manera embebida, facilitando el desarrollo y despliegue del sistema con mínimo esfuerzo de configuración. Además,

herramientas como **Tkinter** y **schedule** permiten mantener una arquitectura modular y eficiente, adecuada para los objetivos académicos del proyecto y las capacidades del equipo.