****

**UNIVERSIDAD AUTÓNOMA DE ASUNCIÓN**

**FACULTAD DE CIENCIAS Y TECNOLOGÍAS**

**TRABAJO GRUPAL DE INTRODUCCION A LA PROGRAMACION PARA DISPOSITIVOS MOVILES**

INFORME DEL PROYECTO FINAL: APLICACIÓN GESTOR DE TAREAS

**SANTIAGO FRANCO ORTIZ**

**DERLIS MATHIAS GAMARRA FRANCO**

**BRIAN RODRIGO ACOSTA BOGARIN**

**EDWARD ANDRES PERALBO DELGADO**

**PROF. DIEGO JESUS BONNIN ACOSTA**

**ASUNCIÓN, PARAGUAY**

**2025**

# Introducción

En el marco del curso de desarrollo de aplicaciones móviles, se ha llevado a cabo el diseño y la implementación de la aplicación Gestor de Tareas, una herramienta digital centrada en facilitar la organización personal y la administración eficiente del tiempo. Esta aplicación fue concebida no solo como un producto funcional para el usuario final, sino también como una oportunidad de integrar y aplicar de forma práctica los conocimientos adquiridos durante todo el proceso formativo, incluyendo principios fundamentales del desarrollo moderno en Android.

El desarrollo del proyecto implicó una planificación detallada que abarcó desde el diseño de la interfaz de usuario hasta la implementación de la lógica de negocio y la gestión de persistencia de datos. Se adoptó una arquitectura robusta basada en el patrón MVVM (Model-View-ViewModel), lo cual permitió una mejor separación de responsabilidades, mayor mantenibilidad del código y facilidad para futuras expansiones.

Durante el desarrollo, se utilizaron herramientas y bibliotecas actuales como Jetpack Compose para el diseño de interfaces declarativas, Room para la base de datos local, Navigation Compose para la gestión fluida de pantallas, y StateFlow junto con ViewModel para el manejo eficiente del estado de la aplicación. Esto permitió crear una experiencia de usuario moderna, visualmente atractiva y adaptable a distintos dispositivos.

Además, este proyecto incluyó la implementación de funcionalidades clave como validaciones en tiempo real, autenticación local con persistencia segura de datos, separación visual de tareas según su estado (pendientes o completadas), lógica automatizada para actualizar tareas vencidas y estrategias de retroalimentación inmediata mediante Snackbars, íconos e indicadores visuales que enriquecen la experiencia de uso.

# Objetivo de la Aplicación

El objetivo principal de la aplicación Gestor de Tareas es permitir que los usuarios puedan organizar y gestionar sus actividades diarias de forma eficiente, promoviendo así una mejor administración de su tiempo personal o profesional. La aplicación busca facilitar la productividad del usuario a través de una interfaz limpia, funcionalidades accesibles y validaciones que garanticen un flujo de uso fluido. Al mismo tiempo, el proyecto persigue un segundo objetivo académico: aplicar e integrar conocimientos clave del desarrollo móvil moderno, permitiendo afianzar buenas prácticas y patrones como MVVM, persistencia local con Room, diseño con Jetpack Compose y manejo de estados con ViewModel y StateFlow.

# Público Objetivo

La aplicación está diseñada para cualquier persona que desee mejorar su organización diaria mediante tecnología, especialmente estudiantes, profesionales y trabajadores que manejan múltiples actividades o proyectos. El diseño simple y amigable permite que tanto usuarios técnicos como no técnicos puedan utilizar la herramienta sin dificultades.

Además, por su enfoque académico, resulta útil también para otros desarrolladores como ejemplo práctico de aplicación bien estructurada con arquitectura moderna.

# Funcionalidades Principales

## Registro de usuario con validación de campos:

* 1. Permite crear cuentas locales con nombre de usuario y contraseña.
  2. Verificación de coincidencia de contraseñas y campos obligatorios.
  3. Diseño intuitivo con retroalimentación visual.

## Inicio de sesión con autenticación local:

* 1. Acceso con usuario y contraseña usando Room como base de datos.
  2. Redirección al menú principal al ingresar credenciales válidas.
  3. Posibilidad de mostrar/ocultar contraseña mediante ícono visual.

## Menú principal con navegación a funciones clave:

* 1. Accesos a tareas registradas, creación de nuevas tareas, y opciones de salida.
  2. Estética simple y directa con navegación clara.

## Creación de tareas con validaciones:

* 1. Formulario con título (obligatorio), descripción (opcional) y fecha límite.
  2. Validaciones en tiempo real con Snackbar.
  3. Conversión de la fecha al formato ISO para facilitar su almacenamiento y ordenamiento.

## Edición y eliminación de tareas:

* 1. Posibilidad de editar o eliminar tareas desde la misma pantalla.
  2. Confirmaciones con Snackbar para mayor interacción.

## Listado de tareas dividido:

* 1. Las tareas se dividen en dos secciones: **Pendientes** y **Completadas**.
  2. Visualmente diferenciadas con título tachado y checkbox marcado si están completadas.

## Checkbox para marcar tareas como realizadas:

* 1. Cada tarea tiene un checkbox que permite al usuario marcarla como hecha o desmarcarla si desea revertir el estado.
  2. El cambio de estado actualiza la sección en la que aparece la tarea.
  3. Mejora la experiencia visual y organiza mejor las actividades del usuario.

## Descripciones largas expandibles:

* 1. Las descripciones extensas aparecen recortadas por defecto (máximo 2 líneas).
  2. Un botón "Ver más" permite expandirlas y leer el contenido completo, con opción de contraer luego con "Ver menos".

## Actualización automática de tareas vencidas:

* 1. Tareas cuya fecha límite ya pasó y no están marcadas como completadas se actualizan automáticamente como completadas.
  2. Esta lógica se aplica al cargar las tareas desde el ViewModel, simulando un control más realista del tiempo del usuario.

## Filtrado visual y separación por estado de las tareas: (repetido)

* 1. Las tareas se organizan automáticamente en dos secciones dentro de la pantalla principal: "Pendientes" y "Completadas", facilitando la lectura y priorización.
  2. Esta separación se realiza desde el ViewModel y se refleja visualmente con títulos distintivos para cada grupo.
  3. Las tareas completadas se muestran con texto tachado y color desaturado, mientras que las pendientes mantienen un estilo normal, aportando claridad inmediata al usuario.

## Navegación estructurada con rutas dinámicas:

* 1. Se emplea Navigation Compose para definir una estructura clara de navegación entre pantallas como Login, Registro, Home, Crear Tarea, Editar Tarea y Lista de Tareas.
  2. Algunas rutas, como la edición de tareas, utilizan parámetros dinámicos (por ejemplo, editar\_tarea/{id}) para pasar datos específicos entre pantallas de forma segura.
  3. Esta implementación mejora la escalabilidad de la aplicación y permite mantener un control eficiente del flujo de navegación.

## Retroalimentación visual mejorada mediante Snackbars y validaciones en tiempo real:

* 1. Cada acción importante del usuario (guardar, editar, eliminar) genera una respuesta inmediata mediante Snackbars, mejorando la interacción y la confianza en la aplicación.
  2. Las validaciones de campos como título y fecha se realizan en tiempo real mientras el usuario escribe, mostrando mensajes de error específicos en caso de datos inválidos.
  3. Esta estrategia de feedback rápido promueve una experiencia de usuario fluida, intuitiva y libre de frustraciones.

# Arquitectura y Herramientas

La aplicación Gestor de Tareas adopta el patrón arquitectónico MVVM (Model–View–ViewModel), lo cual permite una separación clara de responsabilidades, mejor mantenibilidad y escalabilidad del proyecto. Esta estructura se complementa con tecnologías modernas del ecosistema Android para lograr una experiencia de usuario fluida y una lógica de programación bien organizada.

Estructura de la aplicación:

* Modelo (Model): incluye las entidades de datos, las interfaces DAO y la base de datos local utilizando Room, lo que garantiza persistencia eficiente y segura.
* Repositorio: sirve de puente entre la base de datos y los ViewModel, encapsulando la lógica de acceso a los datos y facilitando su reutilización.
* Vista (View): construida completamente con Jetpack Compose, permite el desarrollo de interfaces reactivas, limpias y adaptables, sin depender de XML.
* ViewModel: gestiona los estados de la aplicación mediante StateFlow, ofreciendo una respuesta reactiva a los cambios en los datos y permitiendo mantener el estado incluso ante recomposiciones de la interfaz.

Tecnologías clave:

* Kotlin fue el lenguaje elegido por su sintaxis expresiva, compatibilidad con Android moderno y soporte para programación funcional.
* Android Studio actuó como entorno de desarrollo, brindando herramientas para depuración, diseño visual, pruebas y compilación.
* Room Database se empleó como sistema de almacenamiento local, lo cual facilitó el uso de SQLite con una interfaz más intuitiva.
* Navigation Compose permitió manejar la navegación entre pantallas mediante rutas estáticas y dinámicas, haciendo que el flujo de usuario sea claro y mantenible.
* Material3 fue el sistema de diseño adoptado para asegurar coherencia visual y una experiencia moderna, tanto en dispositivos pequeños como en pantallas más grandes.

# Experiencia de Usuario

La aplicación Gestor de Tareas fue diseñada con un enfoque centrado en la facilidad de uso, la claridad visual y la eficiencia en la interacción. Desde el primer contacto, el usuario se encuentra con una interfaz limpia, organizada y adaptativa, compatible con distintos tamaños de pantalla y dispositivos.

Cada pantalla presenta una estructura clara con controles accesibles y acciones intuitivas, apoyadas por retroalimentación inmediata mediante Snackbars, íconos y colores informativos. Las validaciones en tiempo real evitan errores comunes al ingresar datos, mostrando mensajes claros junto a los campos con problemas. Por ejemplo, al crear una tarea, si el título es demasiado corto o la fecha no es válida, el sistema lo indica de forma visual y textual al instante, sin interrumpir el flujo de trabajo.

La navegación es fluida y coherente gracias a rutas bien definidas, lo que permite al usuario pasar entre pantallas como el menú principal, el listado de tareas o los formularios de edición, sin perder contexto. Además, la división automática de tareas en “Pendientes” y “Completadas” facilita la priorización y comprensión del estado de cada actividad. La interfaz también considera detalles como checkboxes para marcar tareas como realizadas, la expansión de descripciones largas solo cuando es necesario, y la actualización automática de tareas vencidas, lo que ofrece una experiencia rica y sin fricciones.

Todos estos elementos construyen una experiencia de usuario moderna, confiable y adaptada a las expectativas de una aplicación de productividad personal.

# Conclusión

El desarrollo de la aplicación Gestor de Tareas representó una experiencia formativa integral más allá de la mera construcción técnica. Fue necesario combinar planificación, diseño funcional y toma de decisiones precisas para lograr una herramienta que no solo funcione, sino que tenga sentido para quien la utiliza. Cada funcionalidad incorporada respondió a una necesidad real y fue pensada desde la lógica del usuario, buscando siempre la simplicidad sin perder profundidad.

Durante la construcción, se trabajó con tecnologías actuales y se siguieron buenas prácticas del ecosistema Android, pero el foco estuvo puesto en aplicar esos recursos de forma coherente, no solo en mencionarlos. La arquitectura por capas, la organización del código, la claridad visual de las pantallas y el manejo correcto de los estados no fueron solo requisitos, sino pilares sobre los cuales se consolidó una aplicación funcional y mantenible.

Todo el proceso implicó desafíos que exigieron atención al detalle: desde validar datos de forma eficiente hasta asegurar una navegación fluida, pasando por la correcta división entre tareas pendientes y completadas o la visualización adaptativa de descripciones extensas. Lejos de limitarse a cumplir, el proyecto ofreció oportunidades para reflexionar sobre la experiencia del usuario y tomar decisiones acordes.

Esta aplicación se sostiene sobre una base robusta, construida con criterio y claridad, y está preparada para evolucionar hacia futuras mejoras, integraciones o ampliaciones. Su valor radica tanto en lo que ofrece al usuario como en lo que aportó como ejercicio de aprendizaje estructurado y consciente.