

# Documento Técnico: Modelo de Navegación

## Aplicación Móvil Android

---

**Electronicazytron**



**Equipo de Desarrollo - Grupo 4**

Byron Condolo    Pamela Fernandez    Marielena Gonzalez  
Angelo Lascano    Ruth Rosero    Joan Santamaria  
                    Dennis Trujillo

**Asignatura: Dispositivos Móviles**

**Facultad de Ingeniería en Computación**

# Índice

1.	Documento de Especificación Técnica . . . . .	1
1.1.	Identificación del Documento . . . . .	1
1.2.	Propósito y Alcance . . . . .	1
1.3.	Contexto del Proyecto . . . . .	1
2.	Arquitectura de Navegación . . . . .	2
2.1.	Tipo de Navegación Implementado . . . . .	2
2.2.	Justificación Arquitectónica . . . . .	2
3.	Componentes de Navegación . . . . .	3
3.1.	Estructura de NavHost . . . . .	3
3.2.	Tabla de Rutas y Destinos . . . . .	3
4.	Lógica de Navegación por Pantalla . . . . .	4
4.1.	HomeScreen - Punto de Entrada . . . . .	4
4.2.	LoginScreen - Control de Acceso . . . . .	5
4.3.	RegistrarScreen - Creación de Usuarios . . . . .	6
4.4.	ProductScreen - Núcleo de la Aplicación . . . . .	7
4.5.	Pantallas de Productos (Insert/Update) . . . . .	8
5.	Diagramas de Flujo . . . . .	10
5.1.	Diagrama de Navegación de Pantallas . . . . .	10
5.2.	Tabla de Flujo Principal . . . . .	11
5.3.	Flujos Secundarios y Casos Especiales . . . . .	11
6.	Relación con la Implementación . . . . .	12
6.1.	Implementación Práctica . . . . .	12
7.	Patrones y Buenas Prácticas Implementadas . . . . .	12
7.1.	Patrón Single-Activity . . . . .	12
7.2.	State-Hoisting en Navegación . . . . .	12
7.3.	Gestión del Back Stack . . . . .	13
7.4.	Casos de Prueba Críticos . . . . .	13
8.	Conclusiones . . . . .	13
8.1.	Conclusiones . . . . .	13
9.	Anexos . . . . .	14
9.1.	Glosario de Términos . . . . .	14

# 1 Documento de Especificación Técnica

## 1.1 Identificación del Documento

<b>Título del Documento:</b>	Modelo de Navegación - Aplicación Android Electronicazytron
<b>Código:</b>	NAV-MOD-001-2025
<b>Proyecto:</b>	Sistema de Gestión de Inventarios Electronicazytron
<b>Alcance:</b>	Definición de arquitectura y flujos de navegación
<b>Stakeholders:</b>	Equipo de Desarrollo, Profesor, Cliente Académico
<b>Estado:</b>	Aprobado para implementación

## 1.2 Propósito y Alcance

El presente documento tiene como propósito principal documentar de manera exhaustiva el modelo de navegación implementado en la aplicación móvil Android **Electronicazytron**. Este documento establece:

- La arquitectura de navegación adoptada
- Los patrones y tipos de navegación implementados
- La lógica de transición entre pantallas
- La estructura de rutas y parámetros
- Los flujos de usuario autorizados
- Las validaciones y controles de seguridad en la navegación

## 1.3 Contexto del Proyecto

La aplicación **Electronicazytron** es un sistema de gestión de inventarios desarrollado como proyecto académico para la asignatura **Dispositivos Móviles**. El sistema permite:

- Registro y autenticación de usuarios
- Gestión completa de productos (CRUD)
- Navegación fluida e intuitiva

- Arquitectura moderna basada en Single-Activity

#### Tecnologías Principales:

- Jetpack Compose (UI declarativa)
- Navigation Component
- Arquitectura MVVM
- State Management con ViewModel
- Persistencia en memoria (Listas estructuradas)

## 2 Arquitectura de Navegación

### 2.1 Tipo de Navegación Implementado

La aplicación implementa un **modelo de navegación declarativa jerárquica**, alineado con las mejores prácticas de Android y Google Material Design. Las características principales incluyen:

Característica	Descripción Técnica
Paradigma	Navegación Declarativa (Declarative Navigation)
Patrón Arquitectónico	Single-Activity con múltiples Destinos (Composables)
Componente Principal	NavController con NavHost
Gestión de Estado	ViewModel con StateFlow/State
Persistencia de Datos	Memoria RAM (Listas en ViewModel)
Control de Flujos	Navegación Tipada con rutas definidas

### 2.2 Justificación Arquitectónica

La selección de navegación declarativa se fundamenta en:

1. **Separación de responsabilidades:** La lógica de navegación está desacoplada de la UI
2. **Predictibilidad:** Flujos claramente definidos y testables
3. **Seguridad de tipos:** Rutas tipadas que previenen errores en tiempo de ejecución
4. **Gestión del ciclo de vida:** Integración automática con el ciclo de vida de Compose
5. **Back Stack nativo:** Manejo automático del historial de navegación

## 3 Componentes de Navegación

### 3.1 Estructura de NavHost

El componente NavHost actúa como contenedor principal que define el grafo de navegación:

```
NavHost( navController = navController, startDestination = "home") {  
    composable("home") { HomeScreen(navController) } composable("login")  
    { LoginScreen(navController, viewModel) } composable(insertUser") {  
        RegistrarScreen(navController, viewModel) } composable( "productos",  
        arguments = listOf(navArgument(userId") { type = NavType.StringType }) ) {  
        backStackEntry ->val userId = backStackEntry.arguments?.getString(userId")  
        ProductScreen(navController, viewModel, userId) } // ... demás rutas }
```

### 3.2 Tabla de Rutas y Destinos

Código Ruta	Nombre Pantalla	Tipo de Destino	Descripción Funcional
home	HomeScreen	Destino Raíz	Pantalla de bienvenida y punto de entrada principal
login	LoginScreen	Destino Autenticación	Formulario de autenticación de usuarios
insertUser	RegistrarScreen	Destino Registro	Formulario de creación de nuevos usuarios
productos	ProductScreen	Destino Principal	Dashboard de gestión de productos
insertProduct	InsertProductScreen	Destino Modal	Formulario de creación de productos
updateProduct	UpdateProduct	Destino Dinámico	Formulario de edición de productos

## 4 Lógica de Navegación por Pantalla

### 4.1 HomeScreen - Punto de Entrada

**Función:** Actúa como landing page y controlador de acceso inicial.

Elemento UI	Lógica de Navegación
Botón Iniciar Sesión	navController.navigate("login")
Botón Registrar	navController.navigate("insertUser")
Condición Inicial	startDestination del NavHost
Restricciones	Acceso público sin autenticación



Figura 1: Pantalla Home de la aplicación ElectronicaZytron

## 4.2 LoginScreen - Control de Acceso

**Función:** Validación de credenciales y control de flujo post-autenticación.

- **Validación:** Credenciales verificadas contra lista en memoria

- **Navegación Exitosa:**

```
navController.navigate("productos") { popUpTo("home") { inclusive = true } }
```

- **Back Stack Management:** Limpieza de historial para evitar retornos no deseados

- **Mensajes de Error:** Validación en tiempo real con feedback visual



Figura 2: Pantalla inicio de sesión de la aplicación Electronicazytron

### 4.3 RegistrarScreen - Creación de Usuarios

**Flujo:**

1. Captura de datos del formulario (nombre, apellido, contraseña)
2. Validación de campos obligatorios
3. Creación de objeto User y almacenamiento en ViewModel
4. Navegación automática a LoginScreen:

```
navController.navigate("login") { popUpTo("home") { inclusive = false } }
```

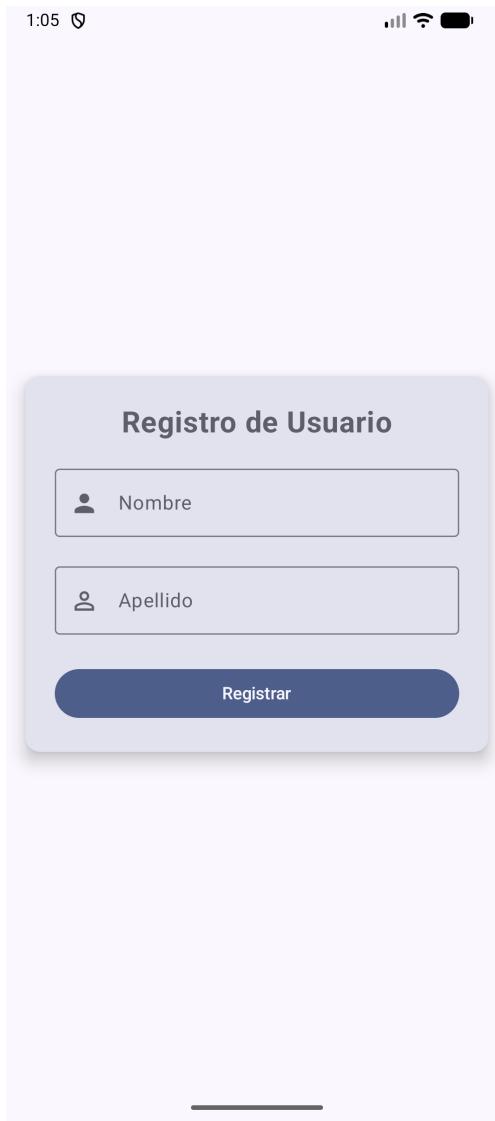


Figura 3: Pantalla registro de usuario de la aplicación Electrónica Zytron

## 4.4 ProductScreen - Núcleo de la Aplicación

**Función:** Centro de operaciones y gestión principal.

Acción	Implementación de Navegación
Nuevo Producto	navController.navigate(insertProduct)
Editar Producto	navController.navigate(updateProduct/\$productId)
Eliminar Producto	Operación local (sin navegación)
Cerrar Sesión	navController.navigate("login") con limpieza de stack



Figura 4: Pantalla de productos de la aplicación Electrónica Zytron

## 4.5 Pantallas de Productos (Insert/Update)

### Características comunes:

- Formularios con validación en tiempo real
- Patrón de parámetros en ruta para UpdateProductScreen
- Navegación de retorno automática tras operación exitosa
- Manejo de estado con ViewModel compartido

### Navegación con parámetros:

```
// Navegar a edición navController.navigate(updateProduct/${product.id}")  
// En NavHost composable( updateProduct/{productId},  
arguments = listOf(navArgument("productId") { type =  
NavType.StringType }) ) { backStackEntry ->val productId =  
backStackEntry.arguments?.getString("productId") // Cargar datos del producto  
}
```

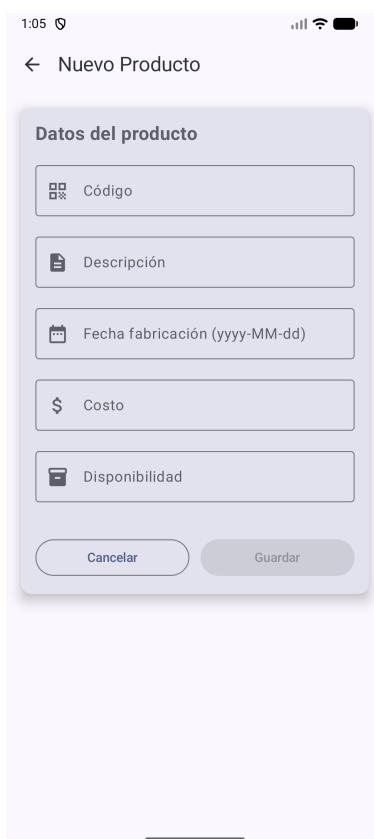


Figura 5: Pantalla de ingreso de productos de la aplicación Electrónica Zytron



Figura 6: Pantalla de actualización de productos de la aplicación Electrónica Zytron

## 5 Diagramas de Flujo

### 5.1 Diagrama de Navegación de Pantallas

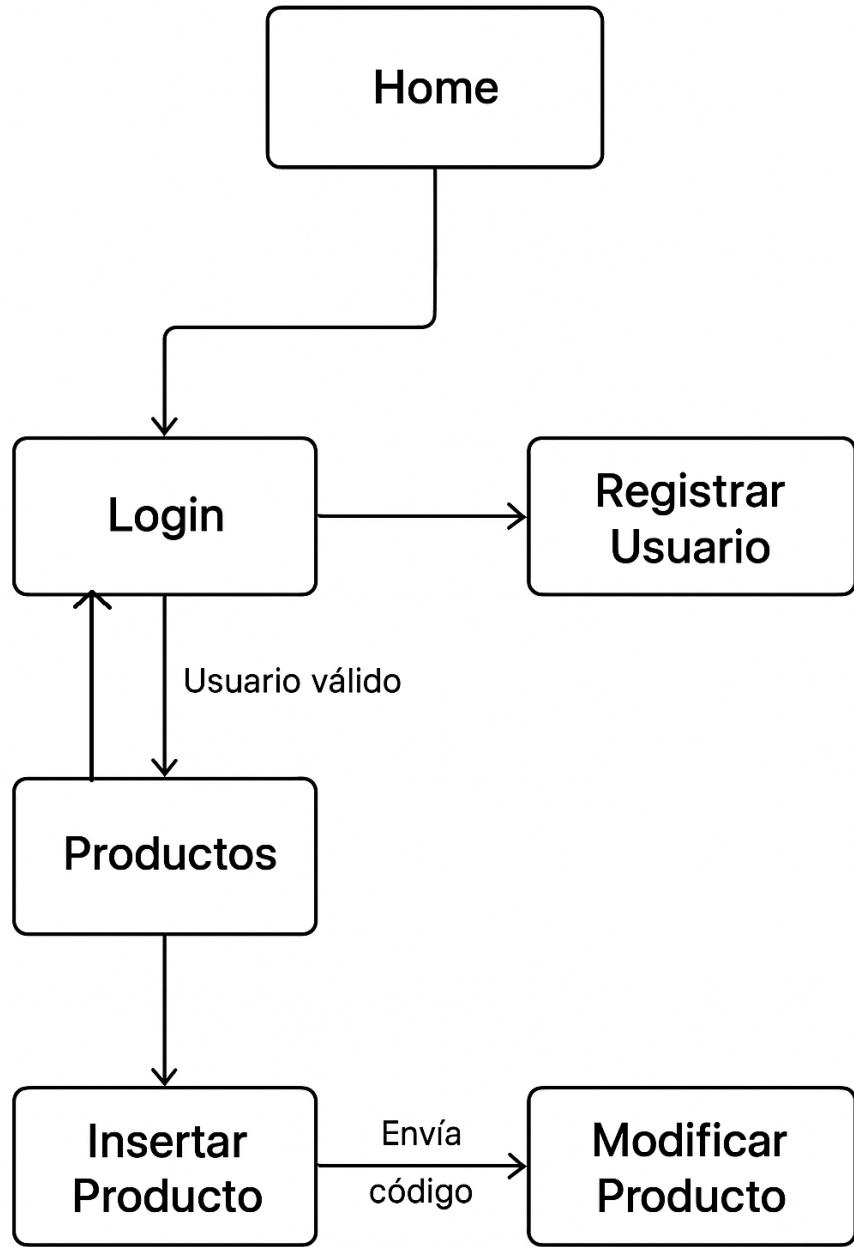


Figura 7: Diagrama de flujo de navegación completo de la aplicación Electronicazytron

## 5.2 Tabla de Flujo Principal

Paso	Pantalla Origen	Pantalla Destino
1	HomeScreen	LoginScreen
2	HomeScreen	RegistrarScreen
3	LoginScreen	ProductScreen
4	RegistrarScreen	LoginScreen
5	ProductScreen	InsertProductScreen
6	ProductScreen	UpdateProductScreen
7	InsertProductScreen	ProductScreen
8	UpdateProductScreen	ProductScreen

Cuadro 2: Tabla resumen del flujo principal de navegación

## 5.3 Flujos Secundarios y Casos Especiales

### 1. Flujo de Autenticación:

- Home → Login → [Validación] → Productos (Dashboard)
- Home → Registrar → Login (con datos precargados)

### 2. Flujo de Gestión de Productos:

- Productos → InsertProduct → [Guardar] → Productos
- Productos → UpdateProduct/{id} → [Actualizar] → Productos
- Productos → [Eliminar] → Productos (refresh)

### 3. Flujo de Cierre de Sesión:

- Productos → [Logout] → Login (clear back stack)

## 6 Relación con la Implementación

Cada pantalla representada en el diagrama corresponde a una ruta definida en el NavHost, implementada mediante composable. La navegación es controlada por NavController, permitiendo:

- Transiciones explícitas entre pantallas
- Validaciones en tiempo de ejecución
- Envío seguro de parámetros entre pantallas
- Gestión automática del historial de navegación
- Validación de estados de autenticación

### 6.1 Implementación Práctica

La figura 7 muestra visualmente cómo cada pantalla está conectada mediante rutas específicas. Esta representación gráfica complementa la tabla 2, proporcionando una visión holística del sistema de navegación.

## 7 Patrones y Buenas Prácticas Implementadas

### 7.1 Patrón Single-Activity

- **Ventaja:** Mejor gestión del ciclo de vida
- **Implementación:** MainActivity como contenedor único
- **Beneficio:** Transiciones más rápidas entre pantallas

### 7.2 State-Hoisting en Navegación

- **Patrón:** Elevación del estado de navegación al ViewModel
- **Implementación:** NavController inyectado en ViewModel
- **Ventaja:** Lógica de navegación testeable y separada de UI

### 7.3 Gestión del Back Stack

Escenario	Estrategia de Back Stack
Login exitoso	popUpTo("home") con inclusive = true
Registro exitoso	popUpTo("home") con inclusive = false
Cierre de sesión	popUpTo(0) (limpieza completa)
Navegación modal	Mantenimiento del contexto anterior

Cuadro 3: Estrategias de gestión del historial de navegación

### 7.4 Casos de Prueba Críticos

- Navegación con parámetros válidos/inválidos
- Comportamiento del botón "Back" del sistema
- Recuperación ante rotación de pantalla
- Flujos de autenticación/desautenticación

## 8 Conclusiones

### 8.1 Conclusiones

El modelo de navegación implementado en **Electronicazytron** demuestra:

1. **Eficiencia:** Flujos optimizados con mínimo número de pantallas intermedias
2. **Claridad:** Estructura jerárquica fácil de entender y mantener
3. **Robustez:** Manejo adecuado de casos excepcionales y errores
4. **Escalabilidad:** Diseño que permite adición de nuevas características

## 9 Anexos

### 9.1 Glosario de Términos

Término	Definición
NavController	Componente que gestiona la navegación entre destinos
NavHost	Contenedor que muestra el destino actual del grafo de navegación
Composable	Función de UI en Jetpack Compose
Back Stack	Pila que almacena el historial de navegación
ViewModel	Componente que almacena y gestiona datos relacionados con la UI

Cuadro 4: Glosario de términos técnicos