



FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS
CARRERA DE COMPUTACIÓN

MATERIA:

DISPOSITIVOS MÓVILES

DOCENTE:

ING. GIOVANNY MONCAYO

INTEGRANTES:

- Andino Jhon
- Borja Diego
- Cajamarca Anthony
- Cruz Kevin
- Jami Mateo
- Quiguango Wulfer
- Valle Armando

TEMA:

Modelo de Navegación – App Giova's SexShop

FECHA DE ENTREGA:

15 de diciembre del 2025



UNIVERSIDAD CENTRAL DEL ECUADOR

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

CARRERA DE COMPUTACIÓN

Documentación del Modelo de Navegación - App "Giova's Sex Shop"

1. Tipo de Navegación

El proyecto utiliza una arquitectura de Single-Activity, que es el estándar moderno para el desarrollo de aplicaciones en Android. Toda la navegación ocurre dentro de una única actividad (MainActivity.kt).

- **Librería Principal:** Se utiliza Jetpack Navigation para Compose (androidx.navigation:navigation-compose), la solución oficial y recomendada por Google para gestionar flujos de navegación en aplicaciones de Jetpack Compose.
- **Componente Central:** La navegación se orquesta a través de un NavHost, que actúa como un contenedor que intercambia las diferentes pantallas (Composables).
- **Controlador de Navegación:** El estado y las acciones de navegación (como ir a una pantalla o volver atrás) se gestionan mediante un NavController, que se instancia en AppNavigation() con la función rememberNavController().

2. Lógica y Flujo de Navegación

La lógica completa está centralizada en el Composable AppNavigation() dentro de MainActivity.kt. A continuación se detalla el flujo y las rutas definidas:

Ruta Inicial: La aplicación siempre se inicia en la ruta login.

Rutas (Destinos de Navegación):

a. Pantalla de Login

- **Ruta:** login?showSuccess={showSuccess}
- **Descripción:** Pantalla de inicio de sesión para autenticar al usuario. Es la pantalla de arranque de la aplicación.
- **Argumentos:**
 - showSuccess (Booleano, Opcional): Si llega con el valor true, muestra un Snackbar con el mensaje "¡Usuario creado con éxito!".
- **Navega Hacia:**
 - home/{userName}: Tras una autenticación exitosa, pasando el nombre del usuario como argumento.
 - register: Si el usuario presiona el botón para registrarse.



UNIVERSIDAD CENTRAL DEL ECUADOR

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

CARRERA DE COMPUTACIÓN

b. Pantalla de Registro

- Ruta: register
- Descripción: Formulario para que nuevos usuarios creen una cuenta (nombre, apellido y contraseña).
- Navega Hacia:
 - login?showSuccess=true: Tras un registro exitoso. Se utiliza popUpTo("login") para limpiar la pila de navegación, evitando que el usuario pueda volver a la pantalla de registro presionando "atrás".

c. Pantalla de Inicio (Catálogo)

- Ruta: home/{userName}
- Descripción: Pantalla principal que muestra el catálogo de productos y da la bienvenida al usuario.
- Argumentos:
 - userName (String): El nombre del usuario que ha iniciado sesión, usado para el saludo "Hola, [userName]".
- Navega Hacia:
 - login: Al cerrar sesión (previa confirmación con un diálogo). Limpia toda la pila de navegación para que el usuario no pueda volver a la HomeScreen sin iniciar sesión de nuevo.
 - product: Para añadir un nuevo producto (la ruta no lleva ID).
 - product?id={productId}: Para editar un producto existente, pasando su ID como argumento.

d. Pantalla de Producto (Crear/Editar)

- Ruta: product?id={id}
- Descripción: Formulario para agregar un nuevo producto o modificar los datos de uno existente.
- Argumentos:
 - id (Entero, Opcional): Si se provee un ID, la pantalla se abre en modo "Edición", precargando los datos del producto. Si el ID es nulo, se abre en modo "Creación".
- Navega Hacia:



UNIVERSIDAD CENTRAL DEL ECUADOR

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

CARRERA DE COMPUTACIÓN

- Vuelve a la pantalla anterior (home): Al guardar los cambios, se utiliza `navController.popBackStack()` para regresar a la pantalla de inicio, que mostrará los datos actualizados.

3. Lógica Adicional (Diálogos de Confirmación)

La aplicación utiliza diálogos modales (`AlertDialog`) para confirmar acciones destructivas. Es importante destacar que estos diálogos no son destinos de navegación dentro del `NavHost`.

- Lógica: Su visibilidad se controla mediante variables de estado booleanas (`showDialog`, `showLogoutDialog`) dentro del `Composable` correspondiente (`HomeScreen` y `MainActivity`).
- Flujo: Al presionar un botón de acción (ej. "Eliminar" o "Cerrar Sesión"), se cambia el estado del booleano a `true`, lo que provoca que el `AlertDialog` se renderice. Las acciones del diálogo (Confirmar/Cancelar) cambian el estado de vuelta a `false` y, si es necesario, ejecutan la acción de navegación correspondiente.