

Android App Development using Jetpack Compose

Module 1: Jetpack Compose Basics and Layouts

- Introduction to Jetpack Compose
 - What is Jetpack Compose?
 - Key differences from XML layouts
 - How Compose simplifies UI development
- Basic Composables
 - Text, Button, Image, Column, Row, Box
 - Modifiers for styling and layout
- Layouts in Jetpack Compose
 - ConstraintLayout, LazyColumn, and LazyRow
 - Building responsive UIs with Modifiers
 - Handling multiple screen sizes and density
 - Adjusting for different screen sizes and densities with Modifier, BoxWithConstraints
- **Lab 1:**
 - Build a simple app using basic composables and MVVM for managing state, such as a counter app with ViewModel.
 - Implement responsive layouts for different screen sizes.

Module 2: State Management, Theming, and Navigation

- State Management in Jetpack Compose
 - State in Compose
 - remember, mutableStateOf, and state hoisting
- Theming in Jetpack Compose
 - Material Design theming in Compose
 - Custom themes: typography, color palettes, and dark mode
- Navigation in Jetpack Compose

- Setting up the navigation component
- Navigating between screens
- Passing data between composables
- MVVM Architecture
 - Introduction to Model-View-ViewModel (MVVM) in Android
 - Using ViewModel in Jetpack Compose
 - State handling and LiveData
- **Lab 2:**
 - Build a multi-screen app using Jetpack Compose navigation, incorporating MVVM and dynamic themes.

Module 3: Handling User Input

- Handling User Input
 - TextField, Buttons, Switches, and Sliders
 - Form validation and user input management
- **Lab 3:**
 - Implement user input forms with validation and basic animations to enhance user experience.

Module 4: Animations

- Animations in Jetpack Compose
 - Simple and complex animations
 - Using AnimatedVisibility, animateFloatAsState, and transitions
 - Advanced Animations
 - Building custom animations using updateTransition, animateContentSize, etc.
 - Using keyframes and animation specs for fine-grained control
- **Lab 4:**
 - Create a sample app that utilizes various animations:
 - Implement a loading spinner using AnimatedVisibility.

- Use `animateFloatAsState` to create a smooth transition effect for a button that changes size on press.

Module 5: Networking

- Networking with Retrofit
 - Performing API requests with Retrofit
 - Displaying remote data in Compose
 - Handling asynchronous data with Coroutines and Flow
- **Lab 5:**
 - Build an app that retrieves data from an API, displays it using Compose UI, and integrates animations

Module 6: Data Persistence and Advanced State Handling

- Data Persistence with Room
 - Setting up Room database in Jetpack Compose
 - Performing CRUD operations
 - Integrating Room with ViewModel and LiveData
- Advanced State Handling
 - Combining State and ViewModel
 - Working with Coroutines and Flow
 - Managing complex states across screens
- **Lab 6:**
 - Build an app that stores user data in a Room database and Flow to handle complex state

Module 7: Localization in Android

- Introduction to Localization
 - Overview of localization and its importance.
 - Android's localization mechanism: Resources and the `res/values` directory.

- Defining language-specific resources (strings, layouts, etc.).
- Implementing Localization in Android
 - Creating string resources for different languages.
 - Using values-<locale> folders for translations.
 - Loading localized content based on device settings.
- **Lab 7:**
 - Create a basic app with string resources in two languages (e.g., English and Spanish).
 - Change device language and observe the app's behavior.

Module 8: Unit and UI Testing

- Unit Testing in Jetpack Compose
 - Introduction to Unit Testing in Android
 - Writing tests for ViewModel and business logic
 - Testing with JUnit and Mockito
- UI Testing in Jetpack Compose
 - Testing UI components, navigation, and interactions
- **Lab 8:**
 - Implement unit tests for ViewModel logic and UI tests for your Compose-based app, testing user interactions and navigation.