Simple Login Screen

1. What is XML?

- XML (Extensible Markup Language) is a structured text format used to store and transport data.
- It relies on tags (similar to HTML) but is designed to describe data rather than display it.
- XML is both human-readable and machine-readable, making it ideal for defining structured information.

2. Role in Android Development

XML role in Android development is primarily in defining user interfaces and resources:

- UI Layouts:
  - Android apps use XML files to define the structure of screens (e.g., buttons, text fields, images).
  - Example: A simple login screen is created using XML before adding functionality with Java/Kotlin.
- Separation of Concerns:
  - XML handles the presentation layer (how the app looks).
  - Java/Kotlin handles the logic layer (how the app behaves).
  - This separation makes apps easier to design, maintain, and scale.
- Project Structure & Resources:
  - XML is used in resource files (colors, strings, dimensions, styles).
  - It ensures consistency across the app and allows easy localization (different languages, screen sizes).
- Integration with Android Studio:
  - Developers visually design layouts in Android Studio, which generates XML under the hood.
  - XML files are stored in the res/layout directory of the project.

Hands-on: Create Simple Login Screen using XML

1. XML Layout (UI Definition) - This is stored in res/layout/activity_main.xml:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:padding="24dp"
   android:gravity="center">

   <EditText
      android:id="@+id/usernameInput"
```

```
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Username"
    android:inputType="textPersonName"/>

  <EditText
    android:id="@+id/passwordInput"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Password"
    android:inputType="textPassword"/>

  <Button
    android:id="@+id/loginButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Login"/>
</LinearLayout>
```

This defines the look and structure of the login screen: two input fields and a button.

2. Java/Kotlin Code (Logic Layer)

This goes in MainActivity.java or MainActivity.kt:

Java Example

```
package com.example.loginapp;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    EditText username = findViewById(R.id.usernameInput);
    EditText password = findViewById(R.id.passwordInput);
    Button loginButton = findViewById(R.id.loginButton);

    loginButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        String user = username.getText().toString();
        String pass = password.getText().toString();
```

```
            if(user.equals("admin") && pass.equals("1234")) {
                Toast.makeText(MainActivity.this, "Login Successful!",
Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(MainActivity.this, "Invalid Credentials",
Toast.LENGTH_SHORT).show();
            }
        }
    });
  }
}
```

3. How They Work Together

- XML defines the UI elements (username field, password field, button).
- Java/Kotlin retrieves those elements using findViewById() and adds behavior (what happens when the button is clicked).
- Together, they create a functional login screen: XML handles the design, Java/Kotlin handles the logic.

In summary: XML is the blueprint for the app's interface, while Java/Kotlin brings it to life with functionality.