Android Text Editor – Project Documentation

# 1. Project Overview

The Android Text Editor project was developed to provide a lightweight, mobile-based solution for creating, editing, and testing Kotlin code directly on Android devices. The application replicates essential features of desktop text editors and extends them with developer-focused tools, including syntax highlighting and compiler integration. It is aligned with the requirements specified in the assignment brief.

# 2. Technologies Used

• Programming Language: Kotlin   
• UI Framework: Jetpack Compose with Material3   
• Platform: Android (API 28+, Android 9 or above)   
• Backend Compilation Support: Python Flask server   
• External Tools: Java JDK, Kotlin Compiler (CLI), ADB (Android Debug Bridge)

# 3. Functionalities

1. File Management   
 • New, Open, Save, and Auto-Save.   
 • Support for multiple file extensions (.txt, .kt, .java).   
  
2. Editing Tools   
 • Copy, Paste, Cut, Undo, Redo.   
 • Real-time word and character count.   
  
3. Find & Replace   
 • Find, Replace, Replace All.   
 • Case-sensitive and whole-word matching.   
  
4. Syntax Highlighting   
 • Native Kotlin syntax highlighting.   
 • Configurable highlighting via JSON files for other languages (Python, Java, etc.).   
  
5. Compiler Integration (USB only)   
 • Compile Kotlin files directly from the app.   
 • ADB reverse tunneling used to connect the app to the Python Flask server.   
 • Real-time error/output reporting.   
  
6. Error Handling   
 • Robust error messages for compiler errors and server issues.   
 • Visual status for compile success/failure.

# 4. Explanation of Work Completed

The project was executed in iterative phases. First, core editing features were implemented (file operations, text manipulation, undo/redo). Next, syntax highlighting was added for Kotlin and extended via JSON for other languages. The final phase integrated compiler functionality through ADB and a Flask server. This combination enables the app to compile Kotlin code in real time. Additional attention was given to error handling, ensuring robustness and a smooth user experience.

# 5. Team Contribution

• 23020458-DYAKannangara: File management and editing features (New/Open/Save, undo/redo).   
• Member 2: Syntax highlighting implementation (Kotlin + JSON configurable languages).   
• Member 3: Compiler integration with Flask server and ADB, error handling.   
• All Members: Testing, debugging, documentation, and presentation video preparation.

# 6. Android Text Editor – Compilation Server Setup (USB Only)

This section provides the step-by-step guide for setting up the compilation server on a Windows PC. The server allows the Android Text Editor app to compile Kotlin code in real time via USB using ADB reverse.

# Step 1 – Install Java JDK

Download and install the Java JDK (e.g., Temurin or Oracle JDK). During installation, tick the option to add Java to the system PATH if available. Verify installation by opening Command Prompt and running:  
 java -version

# Step 2 – Install Kotlin Compiler

Download the Kotlin compiler from the official website and extract it to a stable location (for example, C:\Kotlin). Next, add the bin folder inside this directory to the system PATH using the Environment Variables settings in Windows. This allows the kotlinc command to be executed from any terminal. After updating the PATH, open Command Prompt and verify the installation with:  
 kotlinc -version

# Step 3 – Install Python & Flask

Install Python 3.10 or above from python.org. During installation, tick the box 'Add python.exe to PATH'. After installation, verify with:  
 python --version  
Install Flask using:  
 pip install flask

# Step 4 – Install ADB (Platform Tools)

Download Android Platform Tools from the official Android developer site and extract them (e.g., C:\Android\platform-tools). Add this folder to the system PATH so `adb.exe` can be used from any terminal. Verify with:  
 adb version  
On the Android device, enable Developer Options and turn on 'USB debugging'. Connect the device and run:  
 adb devices  
The device should appear in the list.

# Step 5 – Run the Flask Server

Place the provided `compile\_server.py` file in the `server/` directory of the project. Start the server by running:  
 python compile\_server.py  
Allow Python through the firewall if prompted.

# Step 6 – Reverse Port with ADB

Run the following command to map the device port 5000 to the PC server:  
 adb reverse tcp:5000 tcp:5000  
This step must be repeated if the USB cable is reconnected or the device/PC is restarted.

# Step 7 – Test Compilation

Open the Android Text Editor app, create a Kotlin file, and press 'Compile'. The app will send the code to the server via USB, the server will compile it, and the output or errors will be displayed inside the app.

# 7. JSON Syntax Configuration

The app supports configurable syntax highlighting through JSON files stored in configs/.   
  
Example (configs/python.json):   
{  
 "language": "python",  
 "keywords": ["def","class","import","if","else","for","while","return"],  
 "comments": { "line": "#", "block": null },  
 "strings": [""", "'"]  
}

# 8. Project Structure

TextEditorApp/   
├─ app/ (Android Studio project source code)   
├─ apks/ (Installable APK builds)   
├─ configs/ (JSON syntax files)   
├─ server/ (Flask server and compile\_server.py)   
├─ docs/ ( README, assignment documentation)

# 9. Demonstration Video

A full demonstration video is available at the following link:

👉 [Insert Google Drive/YouTube video link here]