In this lab, you will use **Android Studio** to build a **multi-screen app**. The app topic is open for you to choose, allowing you to create an application that interests you, but with specific requirements regarding the type and number of inputs. This lab will help you learn how to handle **multiple activities (screens)**, manage different input types, and ensure smooth navigation and functionality between the screens.

To ensure originality and creativity, you must incorporate a **minimum number of different input types** and design components, which will make it difficult to replicate from existing apps online.

Instructions:

1. **Set up Android Studio:**
    - Install **Android Studio** from Android Studio Official Site.
    - Follow instructions to set it up on your machine.
2. **Install an External Android Emulator:**
    - Choose and install any external Android emulator of your choice (e.g., **Genymotion**, **Bluestacks**, **Nox Player**, etc.). This will be used to run your Android app in addition to the default emulator that comes with Android Studio.
3. **App Requirements:**
    - **App Topic:** The app topic is of your choice but must serve a functional purpose (e.g., a quiz, educational tool, game, productivity tool, etc.).
    - **Multi-Screen Functionality:** Your app must have **at least two screens** (activities).
        - **Screen 1:** Introductory or main screen (e.g., welcome page or main menu).
        - **Screen 2:** Functional screen where user interaction occurs (e.g., a quiz page, form, game screen, etc.).
    - **Required Components:**
        - Include **at least five different input types** from the following:
            - **EditText (text fields)** for user input.
            - **Buttons** to trigger actions or navigate between screens.

    - **CheckBoxes** or **RadioButtons** for selection options.
    - **Spinners (Dropdown menus)** for selecting predefined options.
    - **Sliders (SeekBar)** to control numeric values.
    - **ImageView or VideoView** for media display.
    - **Canvas** or **DrawingView** for drawing or visual interactions.
    - **Switches** or **Toggles** for on/off functionality.
  - Use **Intents** to navigate between the screens.
  - Incorporate at least one **sensor** or **external data interaction** (e.g., camera, location services, accelerometer, or text-to-speech).
4. **App Design:**
  - Ensure the app has a **professional yet personalized design**. Customize colors, fonts, and images to suit the theme of your app. Make the user interface clean, intuitive, and appealing.
5. **App Behavior:**
  - The app must have meaningful functionality based on the topic you choose. Examples include:
    - A **quiz** where users input answers and receive feedback.
    - A **to-do list** where users can input tasks, mark them as complete, or delete them.
    - A **mini-game** with user interaction and score tracking.
  - Use appropriate logic in the **MainActivity.java** and other activity files to control inputs and ensure smooth navigation between activities (screens).
6. **Run and Test the App:**
  - Test the app using an Android Studio **emulator** and a **physical Android device** (if available) or on the external emulator that you downloaded before.
  - Ensure all functionality works properly and that there are no crashes when navigating between screens.

7. **Submit:**
    - Submit your **Android Studio project files** in a compressed ZIP format.
    - Submit a **professional report** that includes:
        - A copy of your **app code**.
        - **Screenshots** of both screens running on the emulator or physical device.
        - A reflection on the process, highlighting your experience and any challenges you faced.
    - Submit a Video Reflection in which you demo the testing processes on both the android emulator and external testing (either android phone or android emulator)

Tasks to Complete:

1. **Design and Create the App (40%):**
    - **Design a multi-screen app based on a topic of your choice.**
    - **Incorporate at least five different input types and one sensor or external data interaction.**
    - **Ensure smooth navigation between the screens using Intents.**
2. **Run and Test the App (Video Demo) (35%):**
    - **Test the app on both an Android emulator and a physical Android device (if available).**
    - **Ensure the app functions correctly across all screens and input types.**
3. **Prepare and Submit a Report (25%):**
    - **Include screenshots of your app code and the running app on both screens.**
    - **Write a reflection on the assignment, discussing challenges and what you learned from working with Android Studio.**

Submission Guidelines:

Submit the following items through the designated portal:

1. **Android Studio Project Files**:
    - Compress your **entire Android Studio project folder** into a **ZIP file** and upload it. Ensure that all necessary files (source code, layout, resources) are included.
    - The ZIP file should be named appropriately, using the following format: **LastName_FirstName_AndroidProject.zip**.
2. **Report (Word or PDF document)**:
    - Submit a **professional and well-structured report** in either Word or PDF format, including the following:
        - A copy of your **app code** (with syntax highlighting, if possible).
        - **Screenshots** of the app running in both the Android Studio emulator and the external emulator.
        - A reflection on the assignment, detailing any challenges you encountered and your overall experience with Android Studio and the external emulator.
    - Ensure the report includes:
        - A **cover page** with the title, your name, course code, and submission date.
        - **Page numbers** and a **table of contents** for clear navigation.
        - A **professional layout** with proper headings, fonts, and spacing for readability.
3. A Video Demo of your testing Process be named appropriately, using the following format: **LastName_FirstName_Lab3Demo**. **NOTE: NO VIDEO MEANS ZERO on the Lab**

Submit the ZIP file of your project folder, the report and the video as separate, clearly labeled files before the submission deadline.