

Contents

This guide provides step-by-step instructions for setting up and running the AssetPulse application. Depending on your needs, follow the appropriate section:

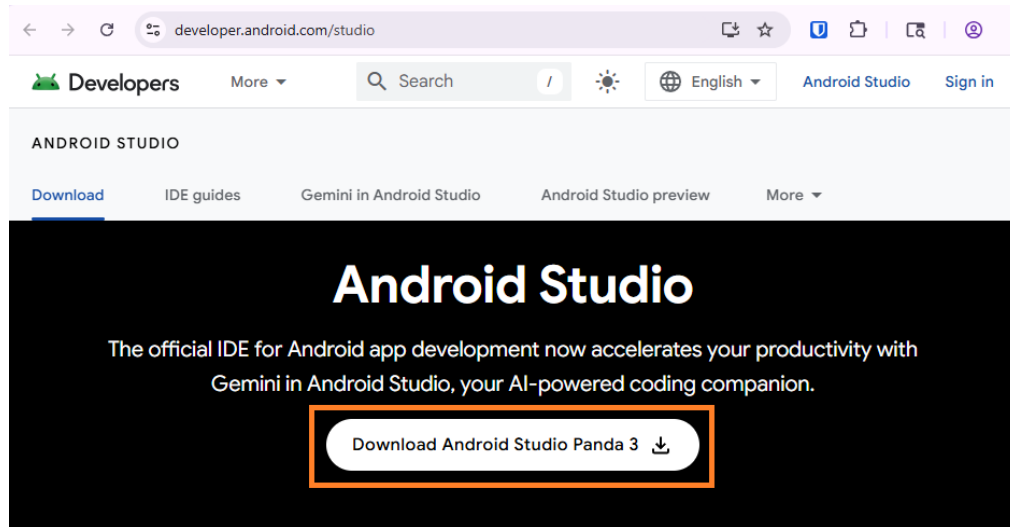
1. **Windows Installation:** Follow this section if you need to set up the full development environment to build, modify and run the project from source code using Android Studio.
2. **Installing the APK on a Physical Device:** Follow this section if you only need to install and use the app on an Android device without setting up the development environment.
3. **Environment Configuration Details:** Reference section that explains the key configuration files in the project. Useful for troubleshooting or understanding how the project is set up.

Windows Installation	2
Installing Android Studio	2
Configuring the Android SDK	6
Importing the Project	9
Setting Up an Android Emulator	12
Building and Running the Application	15
Installing the APK on a Physical Device	16
Environment Configuration Details	19

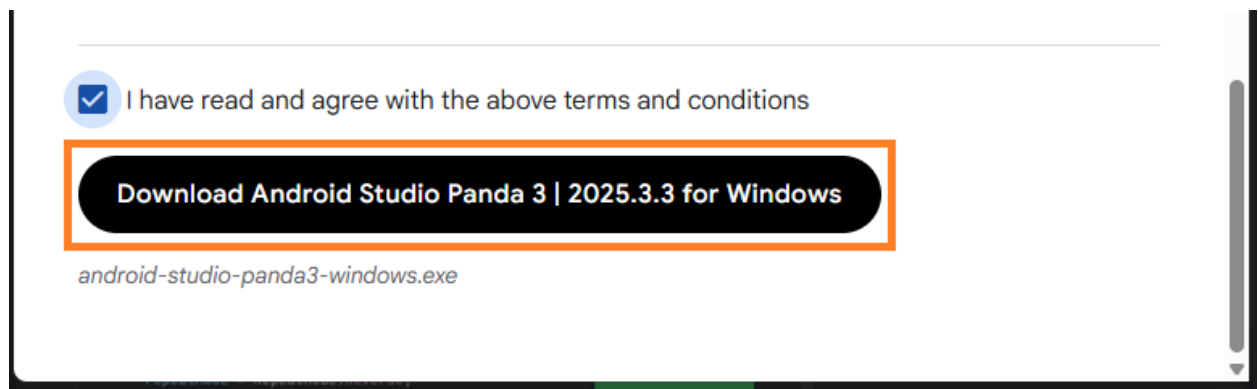
Windows Installation

Installing Android Studio

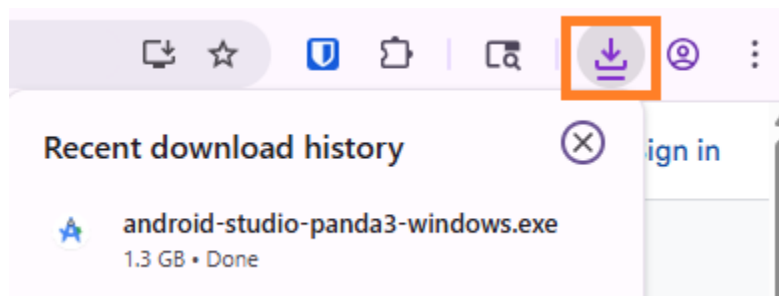
1. Open your web browser and navigate to <https://developer.android.com/studio>.
2. Click the “Download Android Studio” button.



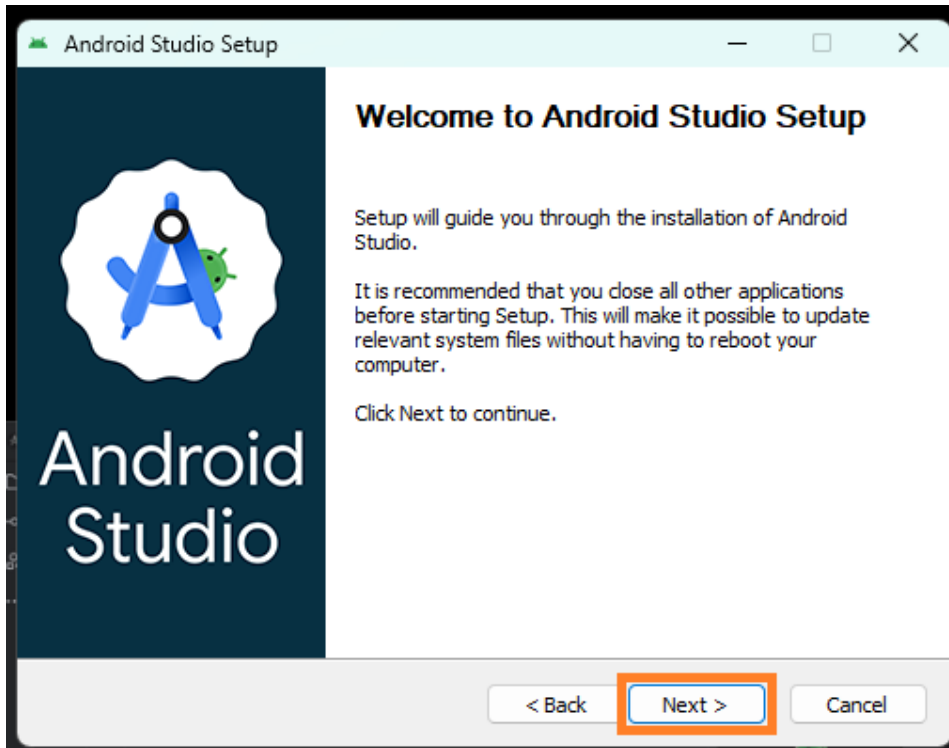
3. Review and accept the Terms and Conditions, then click the download button to begin downloading the .exe installer.



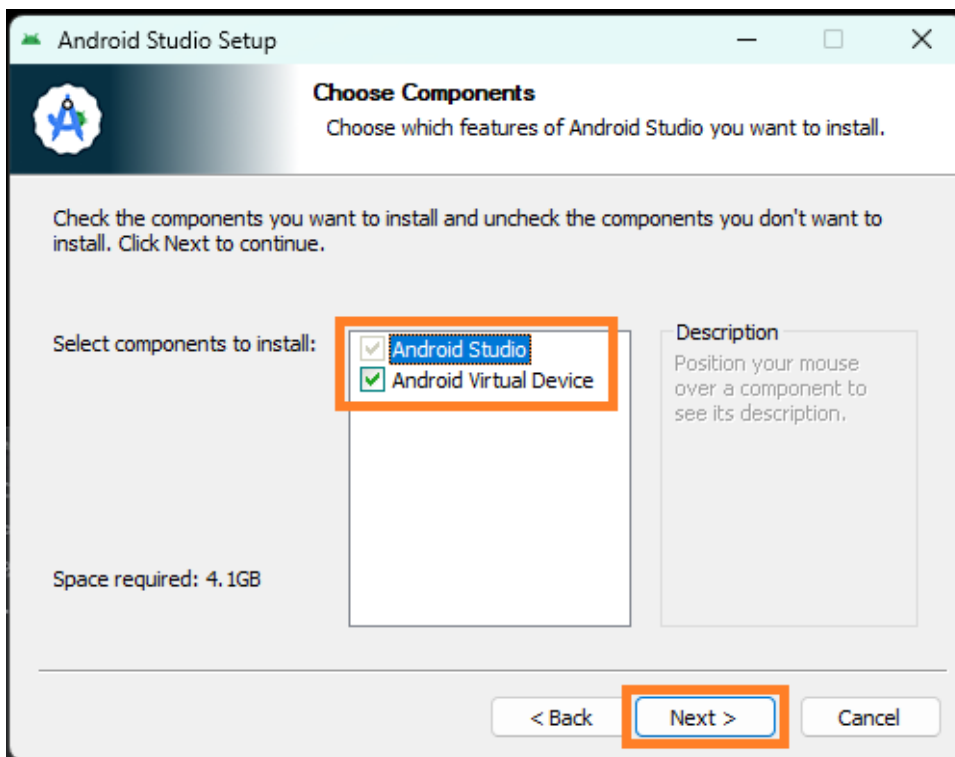
4. Once the download is complete, double-click the **.exe** file to launch the installer.



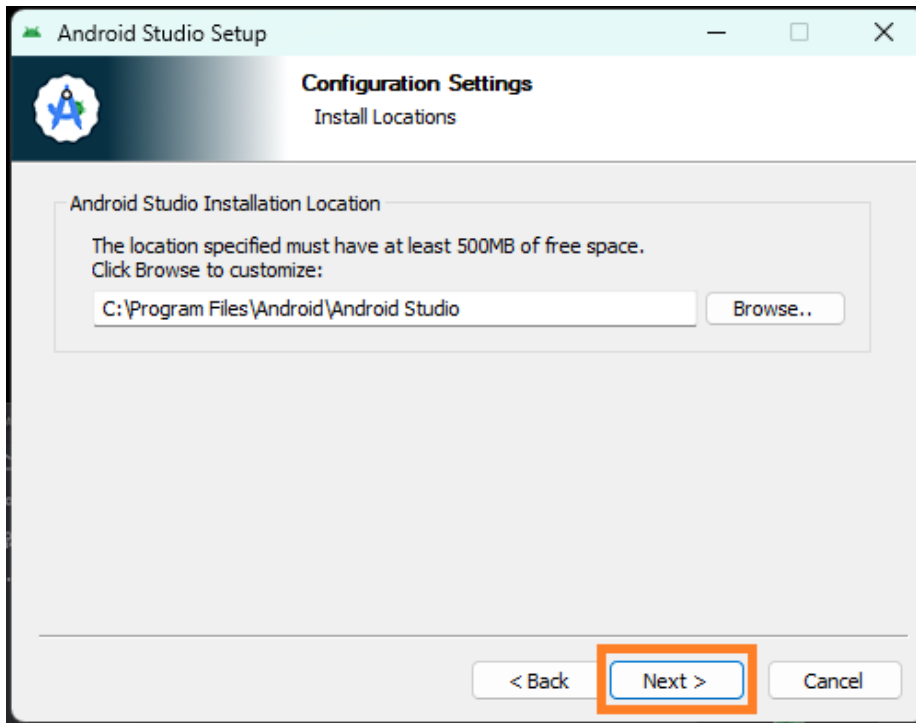
5. The Welcome screen will appear. Click **Next** to continue.



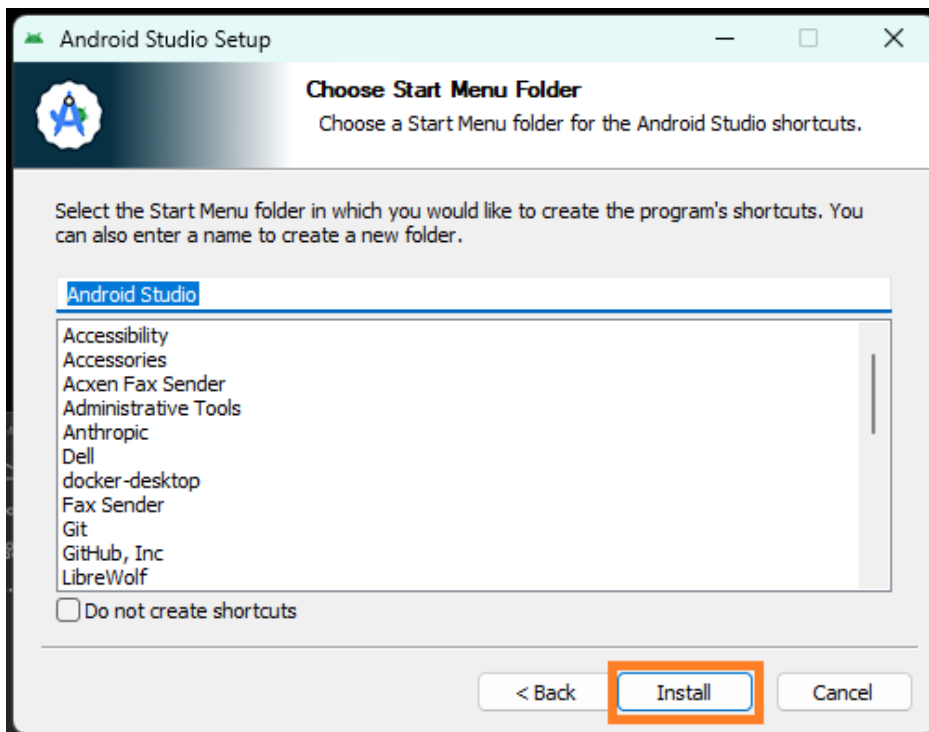
6. On the Choose Components screen, make sure both Android Studio and Android Virtual Device are checked. Click **Next**.



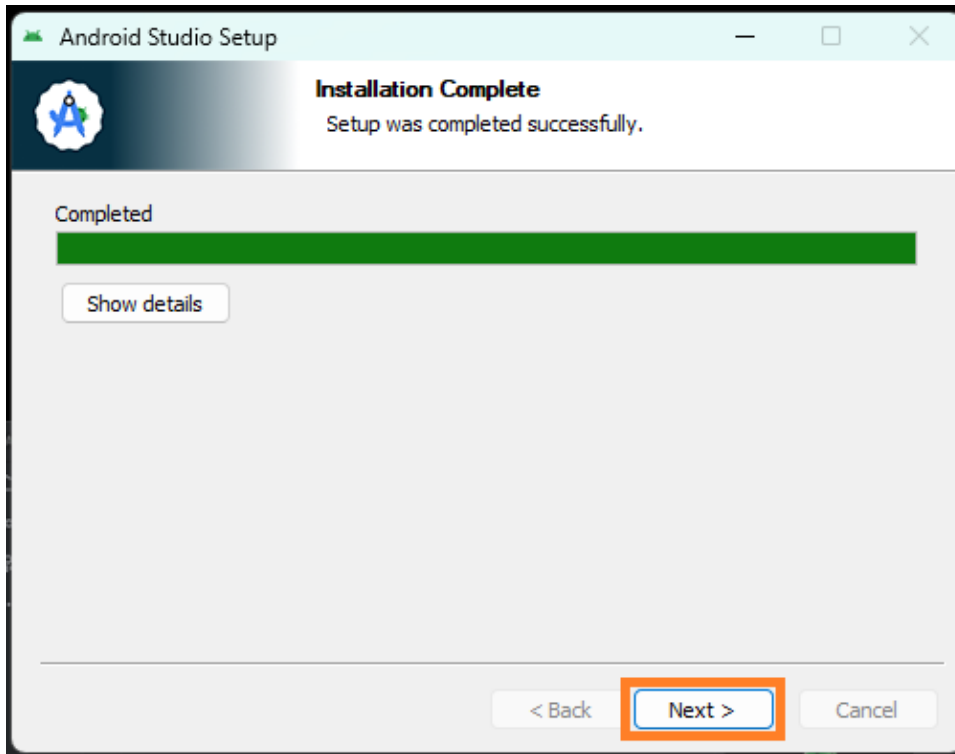
7. The installer will show the default installation path. You can leave it as the default or click **Browse** to choose a different location. Click **Next** to continue.



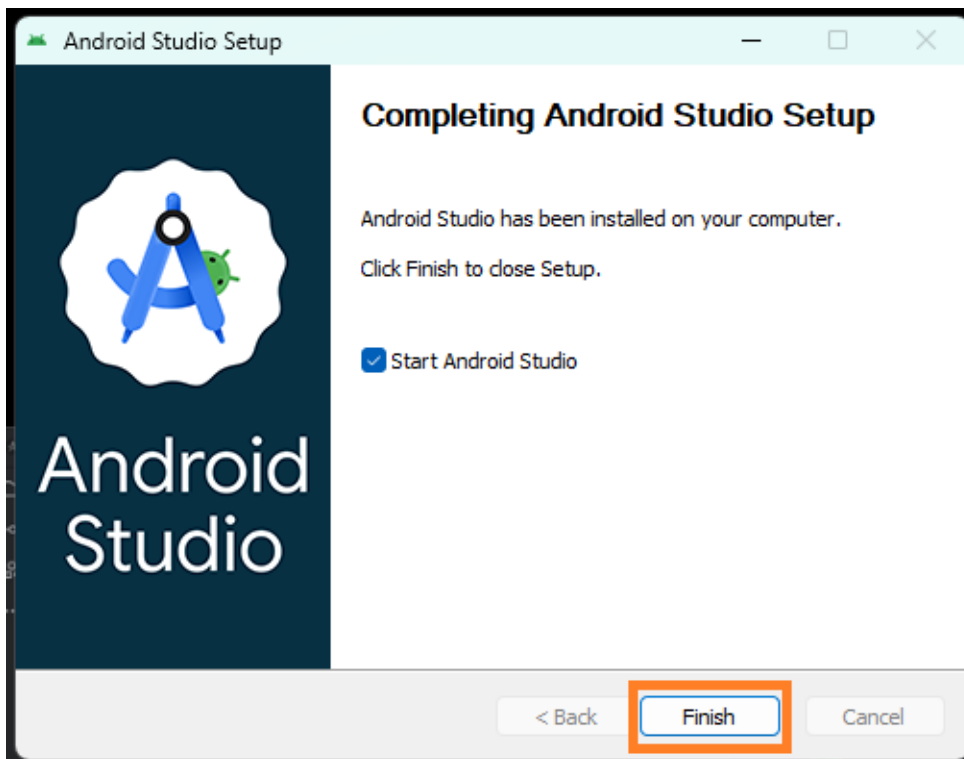
8. On the Choose Start Menu Folder screen, leave the default selection and click **Install**. The installation will begin.



9. Wait for the installation to be completed. Once the progress bar is full, click **Next**.

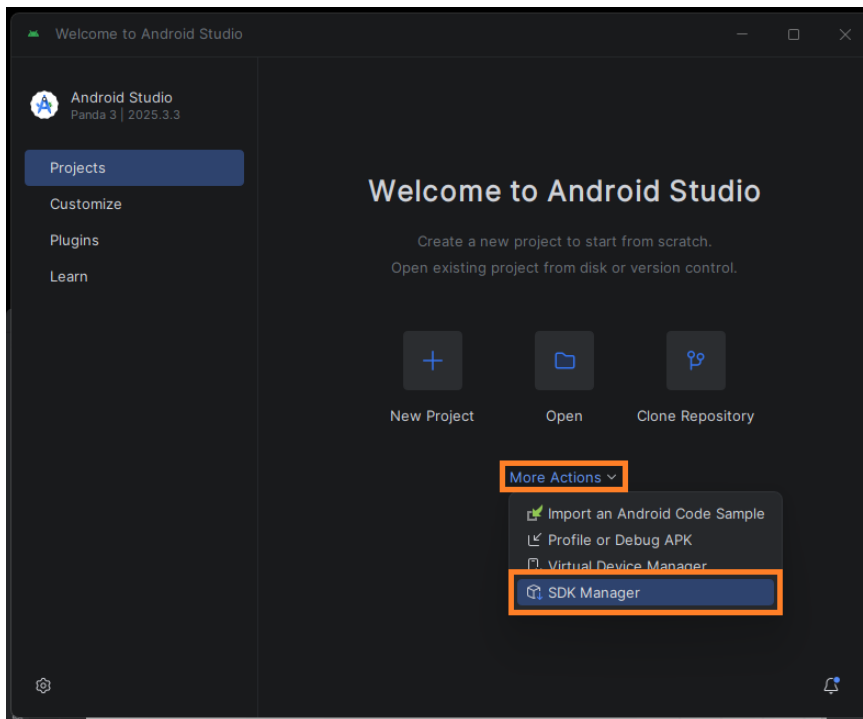


10. Make sure **Start Android Studio** is checked and click **Finish**. Android Studio will launch for the first time.

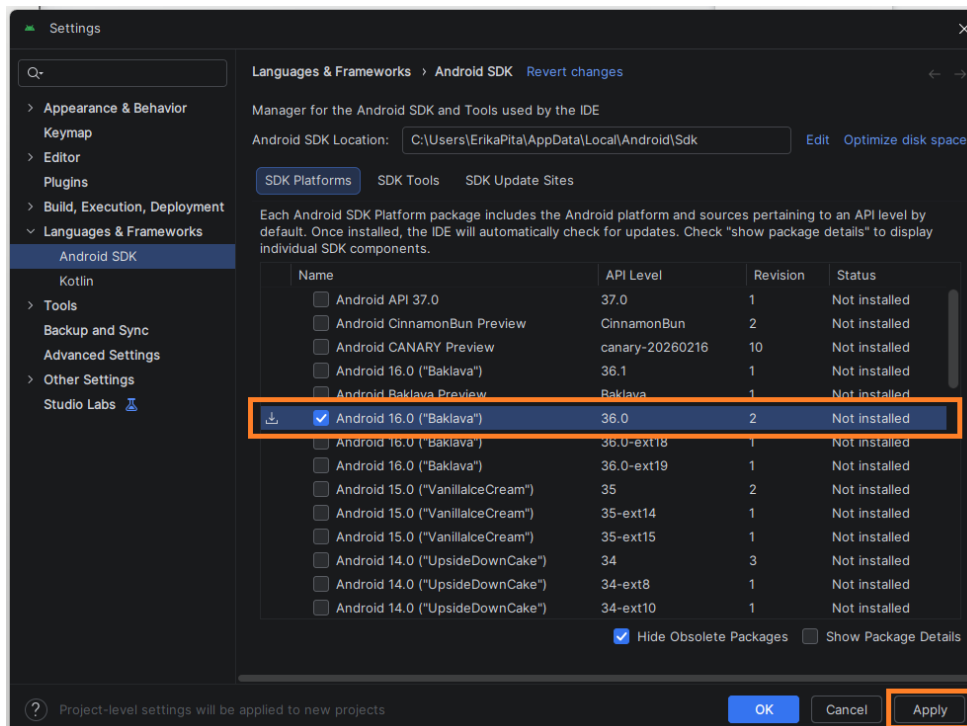


Configuring the Android SDK

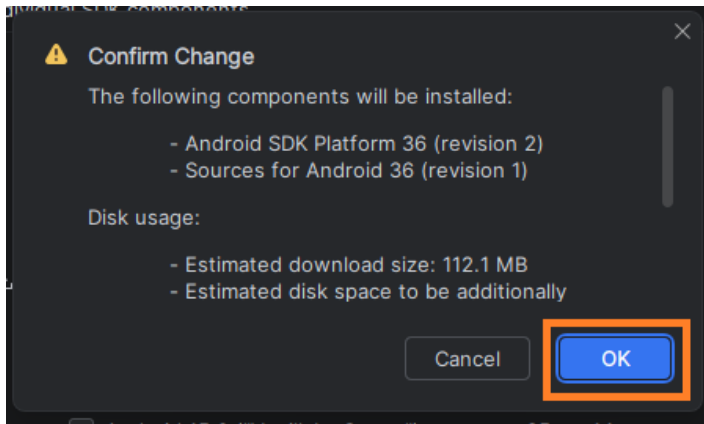
1. From the Welcome screen, click **More Actions** and select **SDK Manager**.



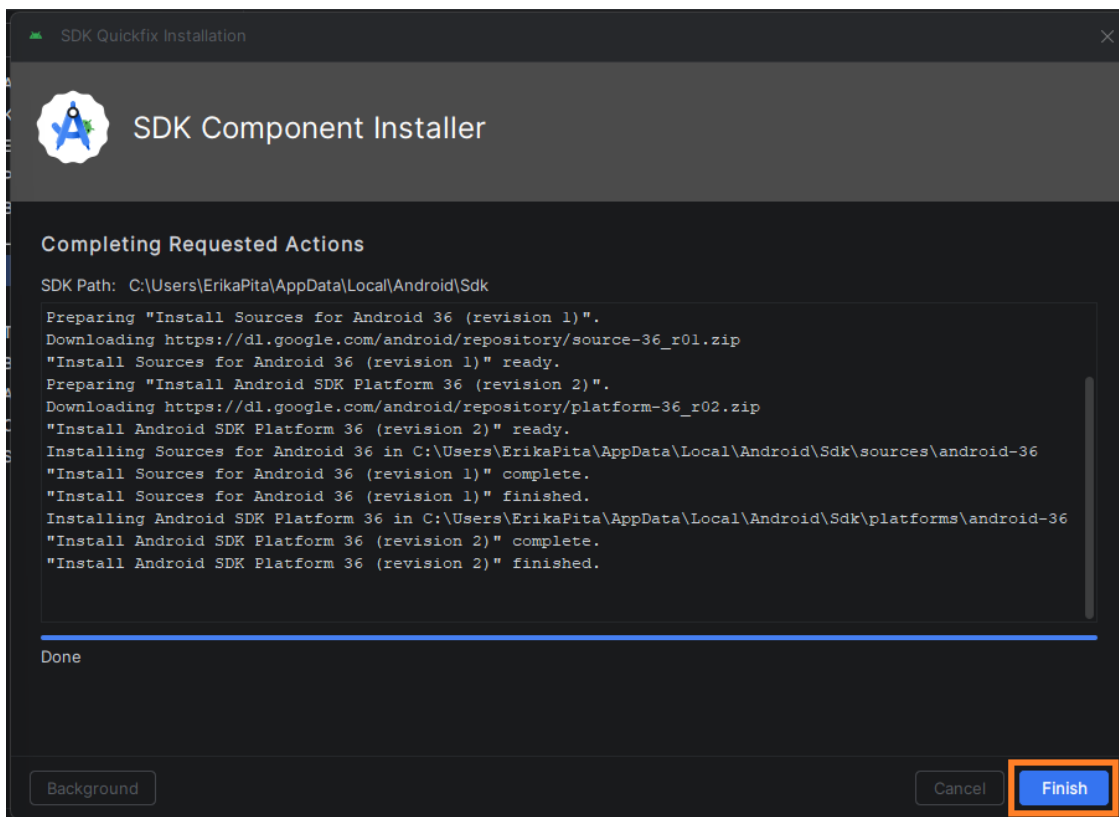
2. In the **SDK Platforms** tab, check the box for **Android 16.0 ("Baklava") – API Level 36.0** and click **Apply** to download and install it.



3. A confirmation dialog will appear showing the components to be installed. Click **OK** to proceed.



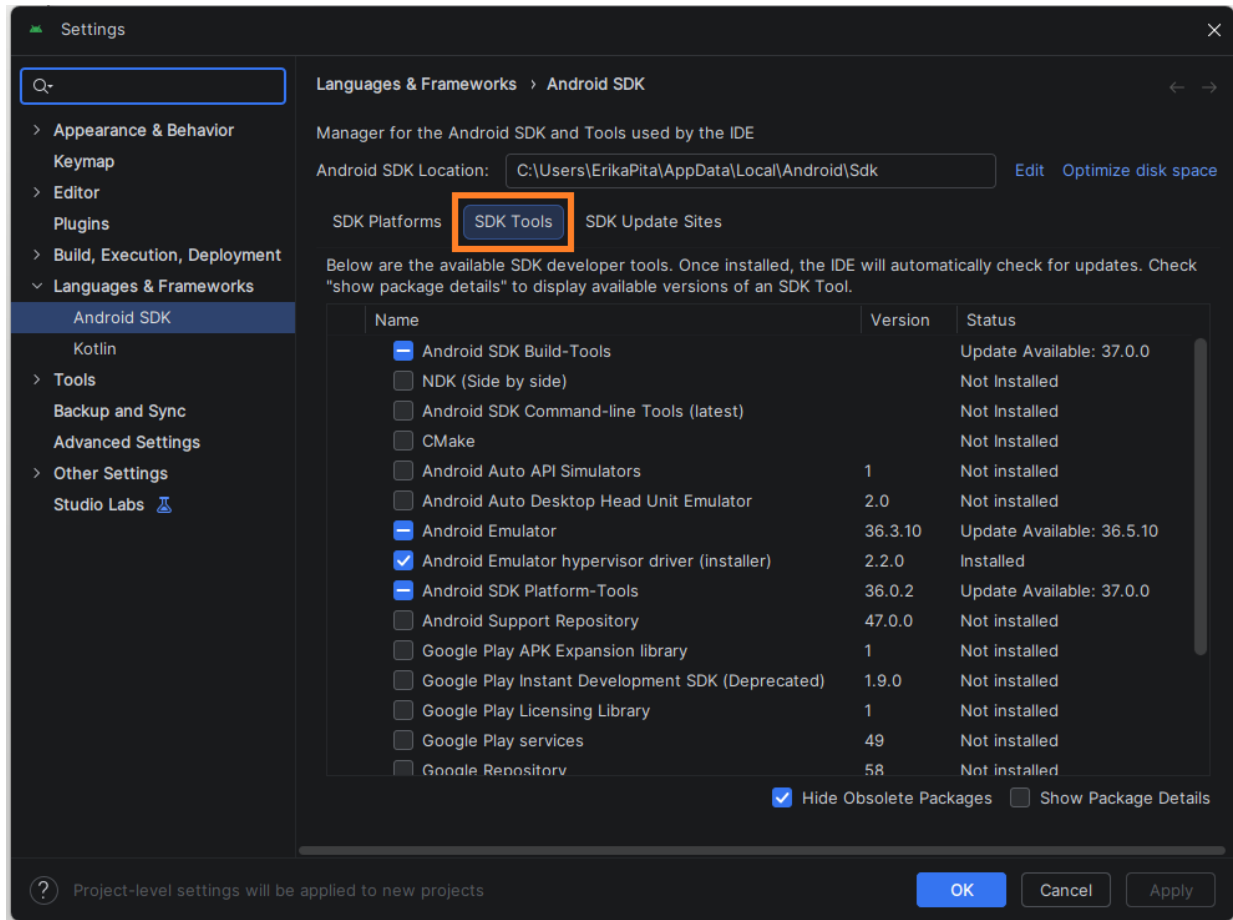
4. Wait for the download and installation to complete. Once it shows “Done”, click **Finish**.



5. Switch to the SDK Tools tab and verify that the following are installed:
 - a. Android SDK Build-Tools
 - b. Android Emulator
 - c. Android SDK Platform-Tools

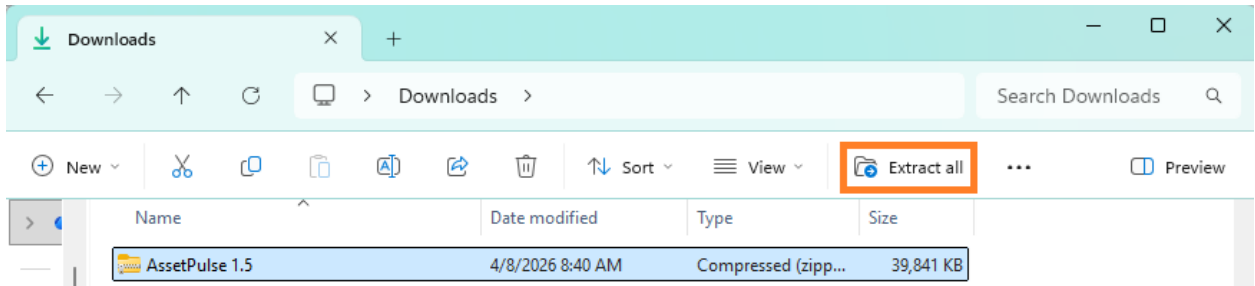
d. Android Emulator hypervisor driver

If any of these are not installed, check the corresponding boxes and click **Apply**.

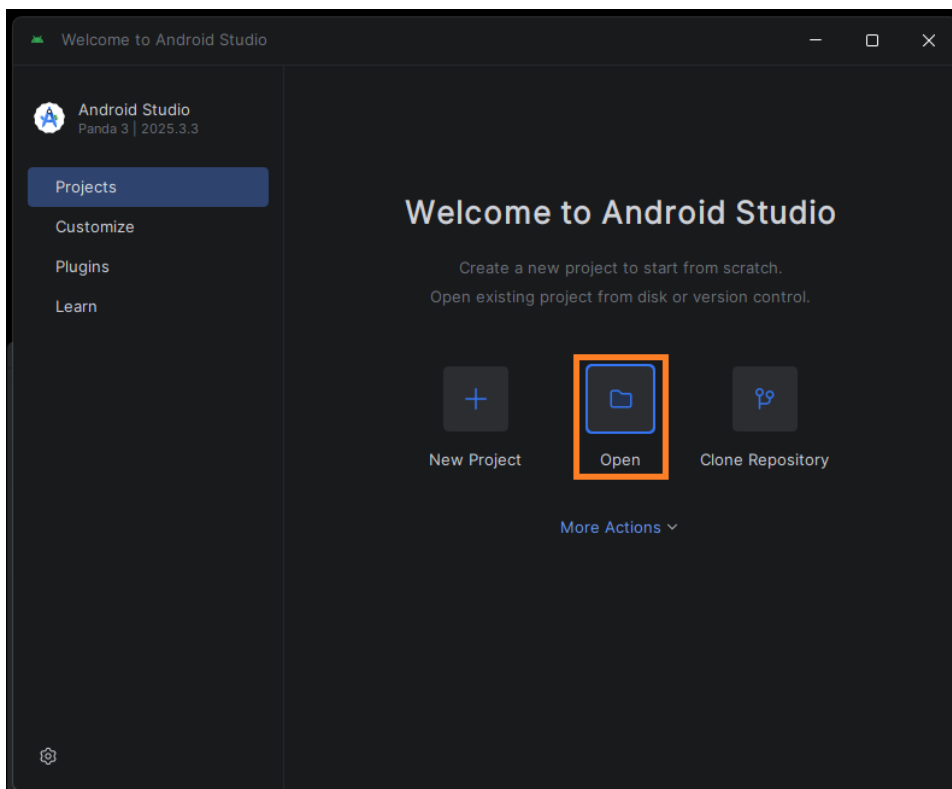


Importing the Project

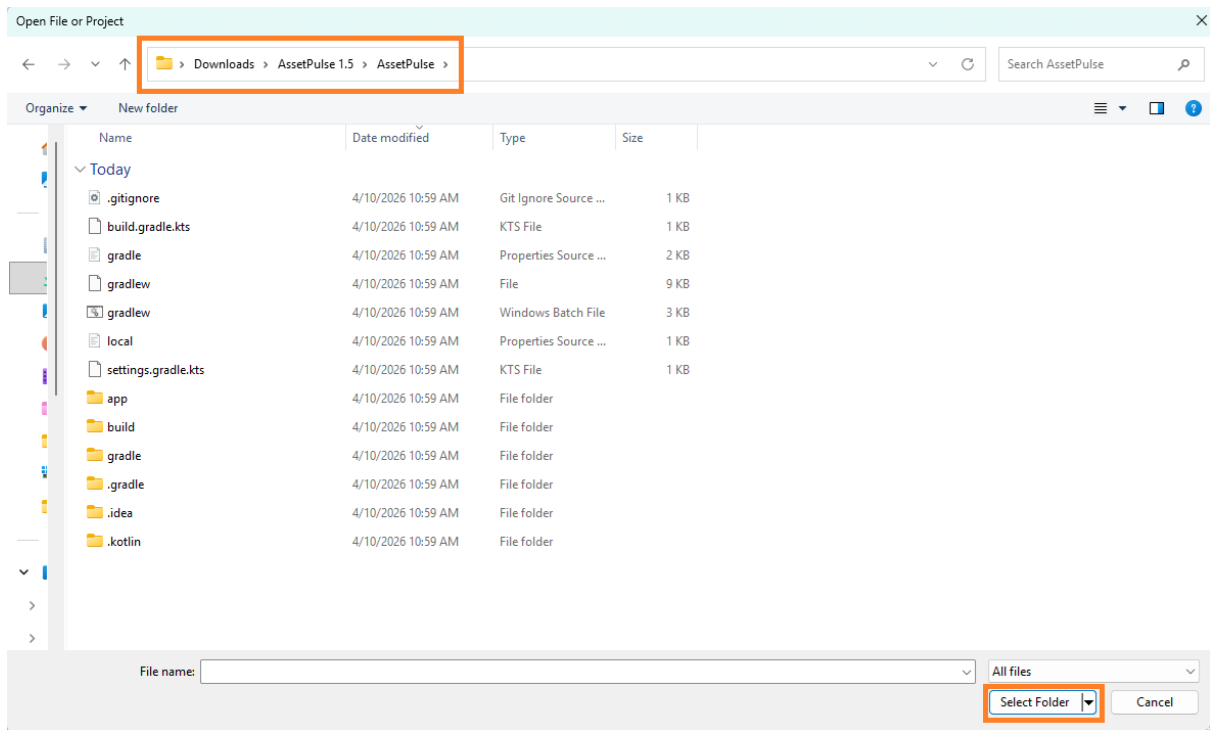
1. Locate the .zip file and extract it to a convenient location on your machine (e.g., your Downloads folder). Click the ZIP file and select **Extract All**.



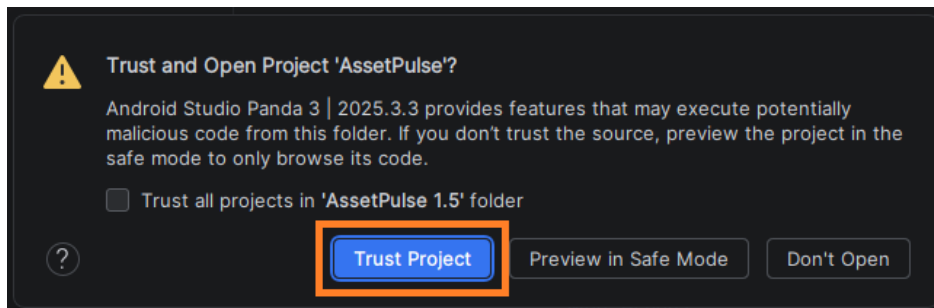
2. From the Android Studio Welcome screen, click **Open**.



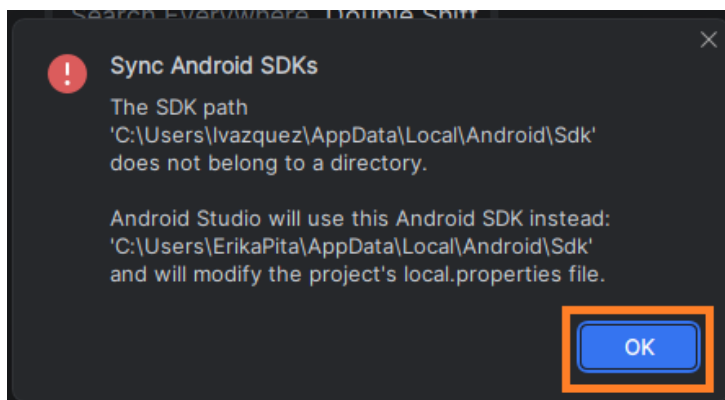
3. Navigate to the extracted folder and select the **AssetPulse** project folder (the one containing settings.gradle.kts). Click **Select Folder**.



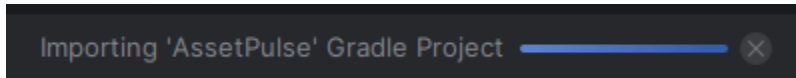
4. A trust dialog will appear. Click **Trust Project** to proceed.



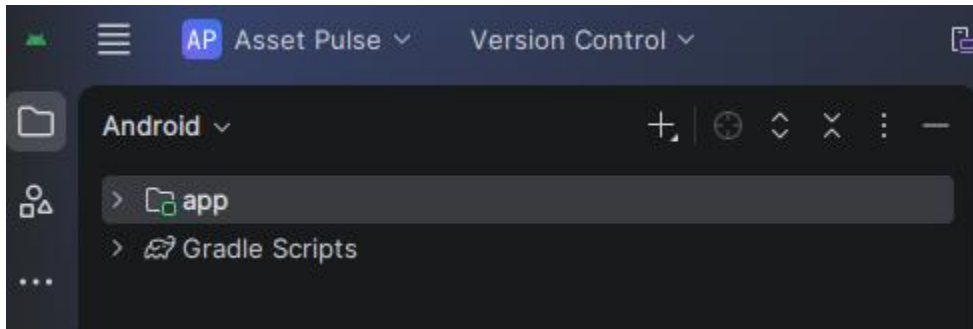
5. A dialog may appear indicating that the SDK path does not match. This happens because the project was developed on a different machine. Android Studio will automatically update the local.properties file with your local SDK path. Click **OK**.



6. Android Studio will begin syncing the Gradle project. A progress bar will appear at the bottom of the screen. This may take several minutes on the first run as Gradle downloads all required dependencies.

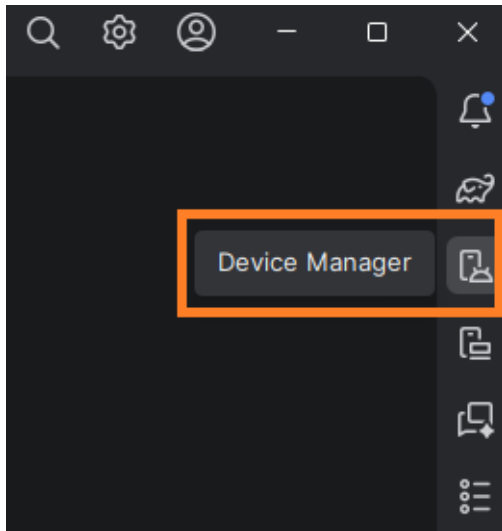


7. Once the sync completes, the project will open successfully. You should see the app module and Gradle Scripts in the project panel on the left.

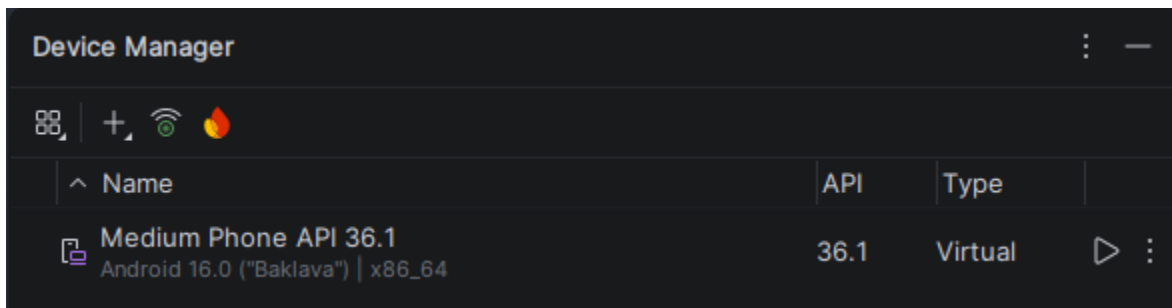


Setting Up an Android Emulator

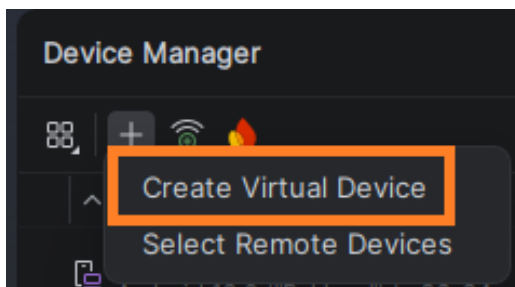
1. Click the **Device Manager** icon on the right sidebar.



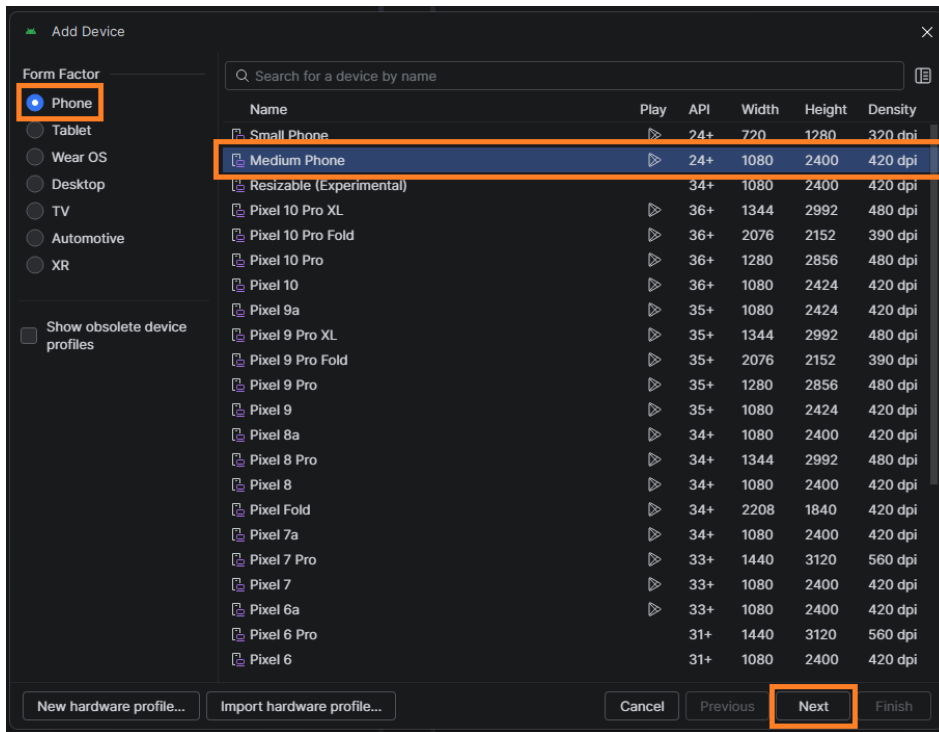
2. If you selected **Android Virtual Device** during the Android Studio installation, a default emulator will already be listed.



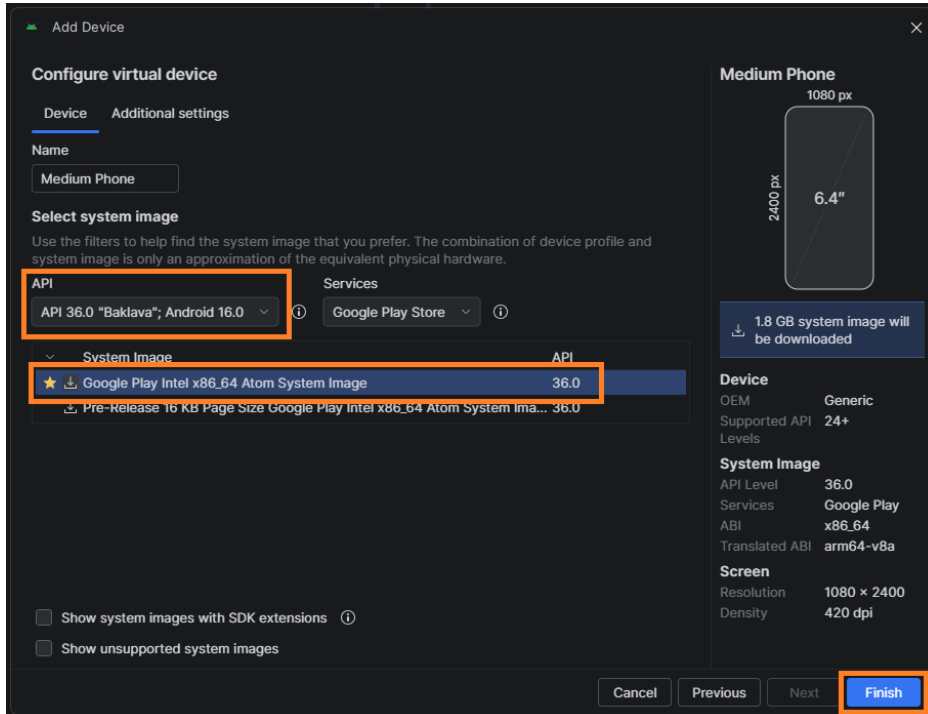
3. In the Device Manager, click the + button and select **Create Virtual Device**.



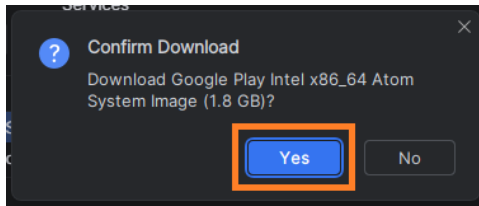
4. Select **Phone** as the form factor, then choose a device definition (e.g., **Medium Phone**). Click **Next**.



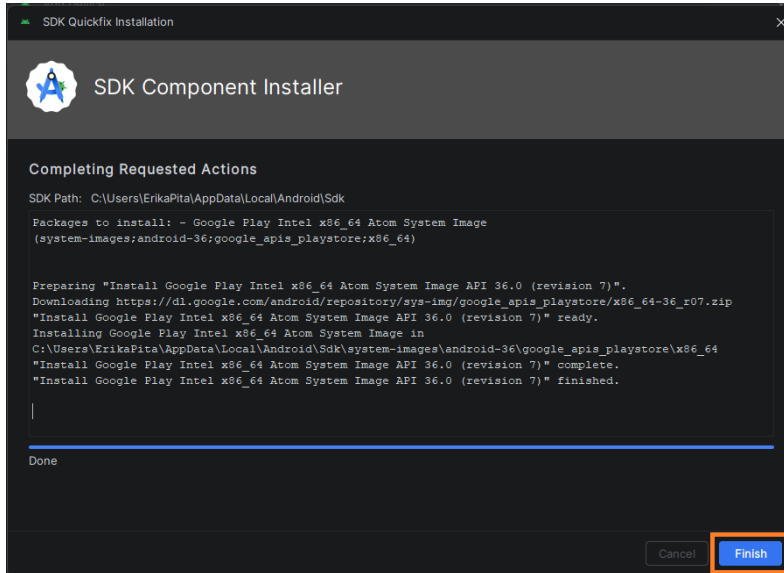
5. Select the Google Play Intel x86_64 Atom System Image for API 36.0 and click **Finish**. The system image will be downloaded automatically (approximately 1.8 GB).



6. A confirmation dialog will appear. Click **Yes** to begin the download.



- Wait for the system image to finish downloading. Once it shows “Done”, click **Finish**.

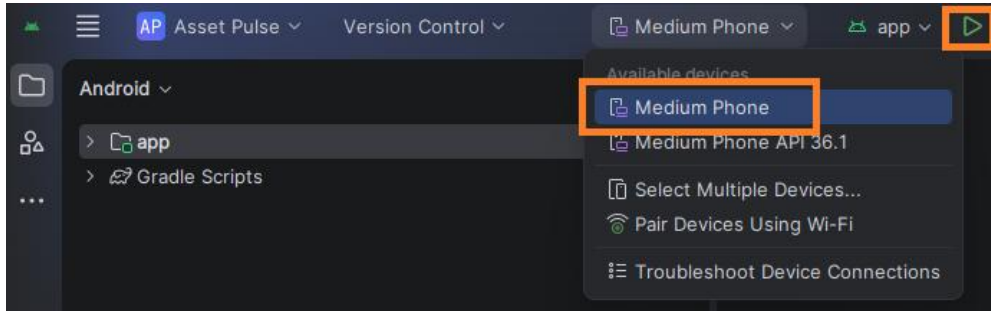


- The new virtual device will now appear in the Device Manager, ready to use.

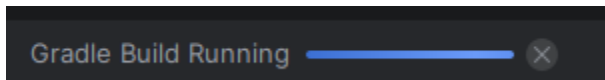


Building and Running the Application

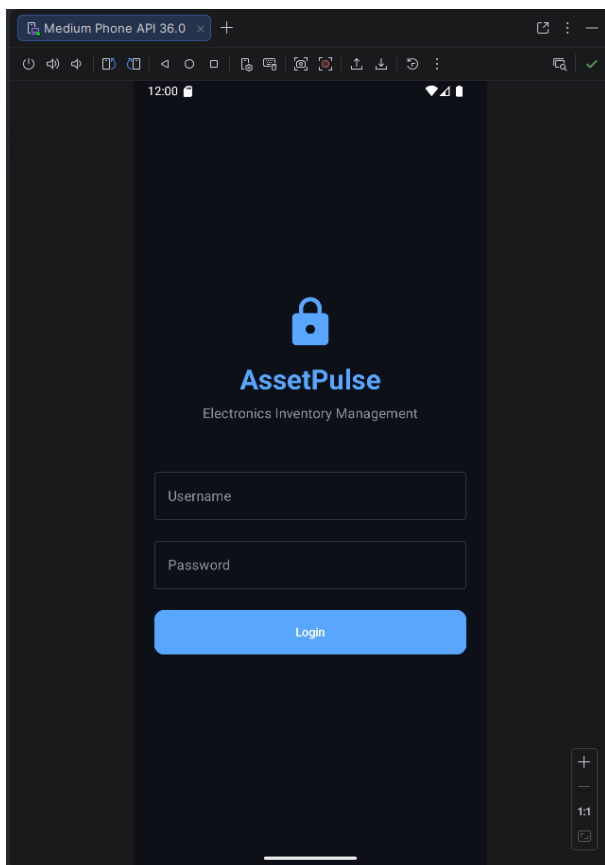
1. Make sure the desired emulator is selected in the device dropdown at the top of the toolbar. Click the **green Run** button to build and run the application.



2. The Gradle build process will start. A progress bar will appear at the bottom of the screen.

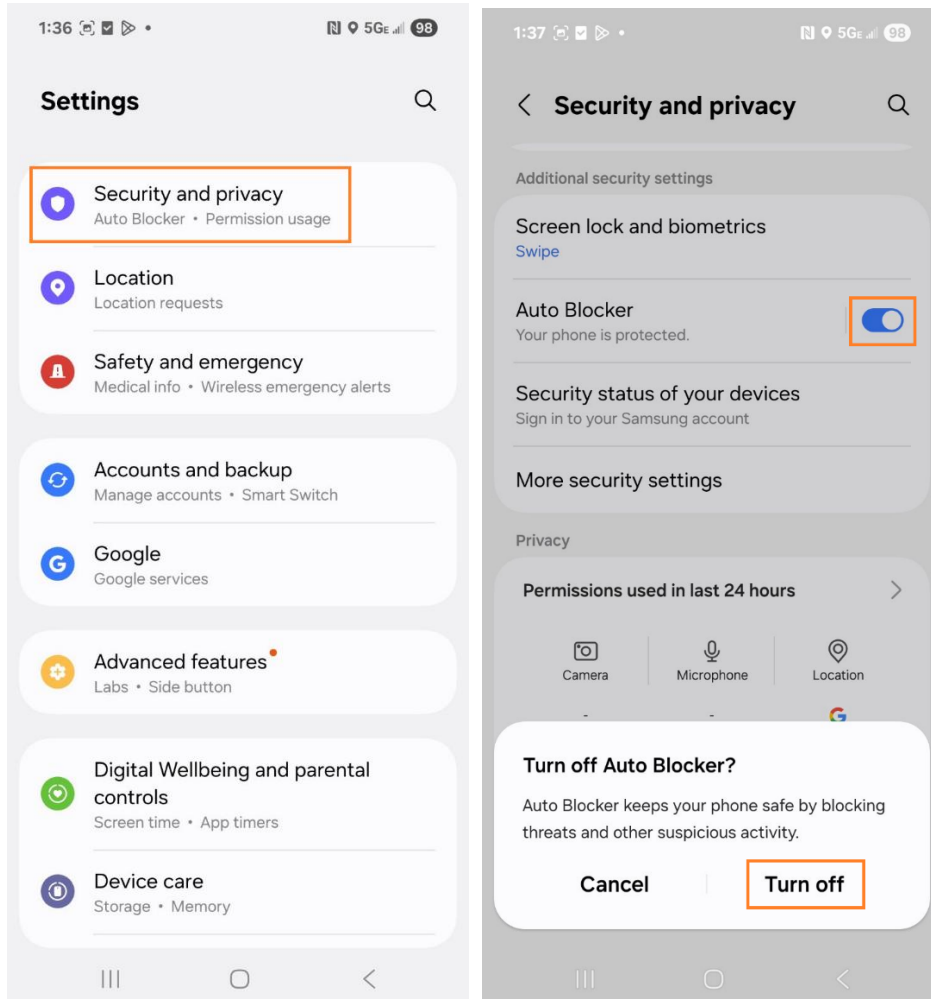


3. Once the build is completed and the app is installed, it will launch automatically on the emulator.

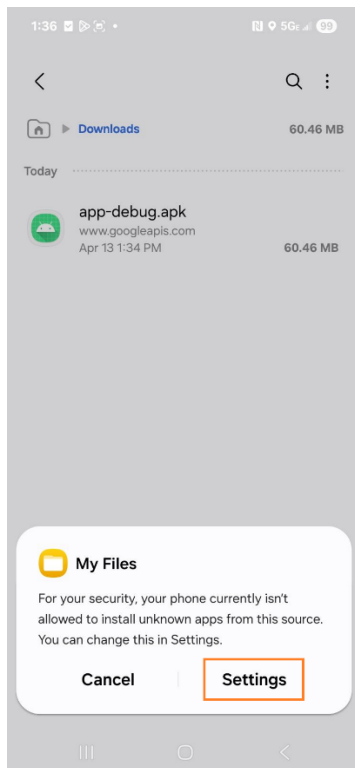


Installing the APK on a Physical Device

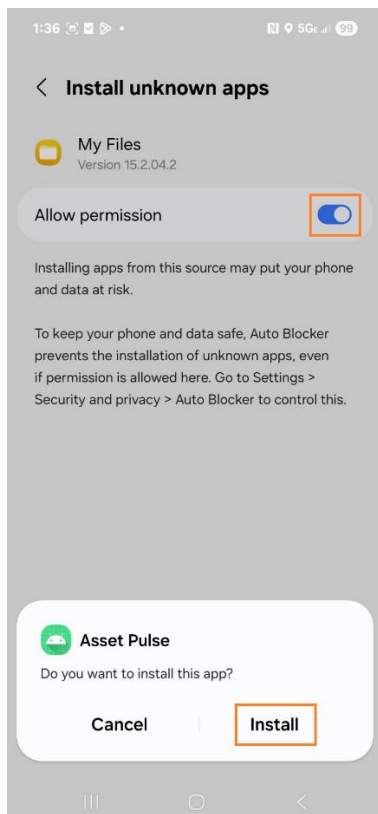
1. Go to **Settings > Security and privacy** and turn off **Auto Blocker**. This is required to allow installation from sources outside the Play Store.



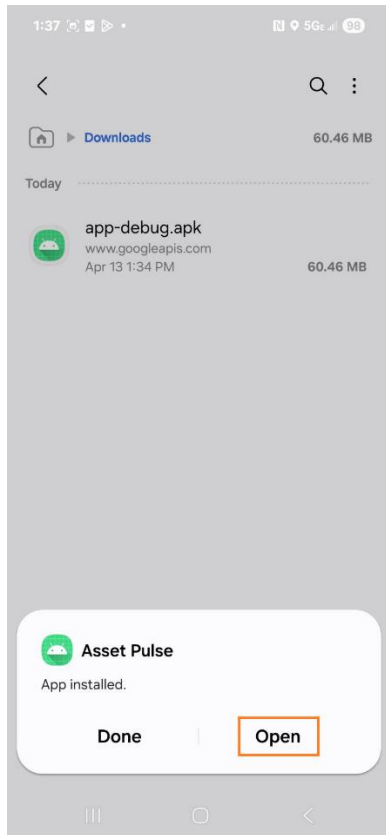
2. Transfer the APK file to the device (via Google Drive, email, USB, etc.).
3. Open the file manager and navigate to the Downloads folder. Tap the APK file. A dialog will appear indicating that your phone is not allowed to install unknown apps from this source. Tap **Settings**.



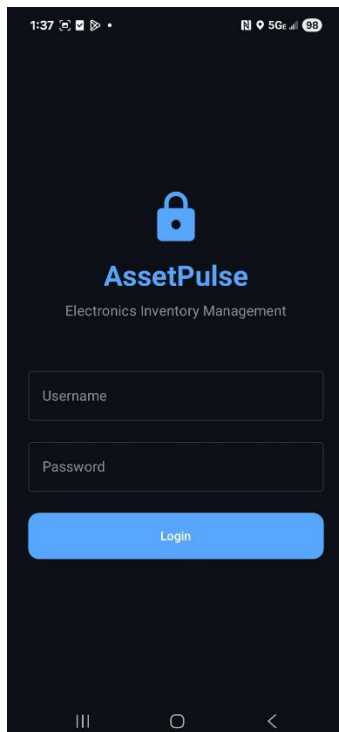
4. On the **Install unknown apps** screen, toggle **Allow permission** to ON. Then, when prompted, tap **Install**.



5. Once installed, tap **Open** to launch AssetPulse.



6. The AssetPulse login screen confirms that the APK was installed successfully.



Environment Configuration Details

The following files control the project's build environment and configuration. Understanding these files is important for troubleshooting and customization.

IMPORTANT: These configuration files are either auto-generated by Android Studio or already pre-configured in the project. No manual changes are required. This section is provided for reference so that developers understand what each file does and where to look if troubleshooting is needed.

local.properties

Stores the path to your local Android SDK installation. This file is auto-generated by Android Studio and is unique to each developer's machine. Example:

```
sdk.dir=C:\\Users\\YourName\\AppData\\Local\\Android\\Sdk
```

gradle.properties

Contains project-wide Gradle settings. Key configurations include: *org.gradle.jvmargs=-Xmx2048m* which allocates 2 GB of memory to the Gradle daemon, *android.useAndroidX=true* which enables AndroidX libraries and *android.nonTransitiveRClass=true* which reduces build times by scoping R classes.

build.gradle.kts (app module)

Defines the application's build configuration:

Setting	Value
Application ID	com.levis.assetpulse
Compile SDK	36
Minimum SDK	30 (Android 11)
Target SDK	36
Kotlin	2.3.10
Gradle	9.1.0 (via Gradle Wrapper)
Java Compatibility	Java 11
Compose Enabled	true

gradle/libs.versions.toml

The version catalog file centralizes all dependency versions in one place. If you need to update a library, change its version number here and Gradle will apply the update project-wide on the next sync. Key dependencies managed by this file include Jetpack Compose, Room (local database), Navigation Compose and KSP (annotation processing).