

7.Android Emulator (Virtual Device) Setup

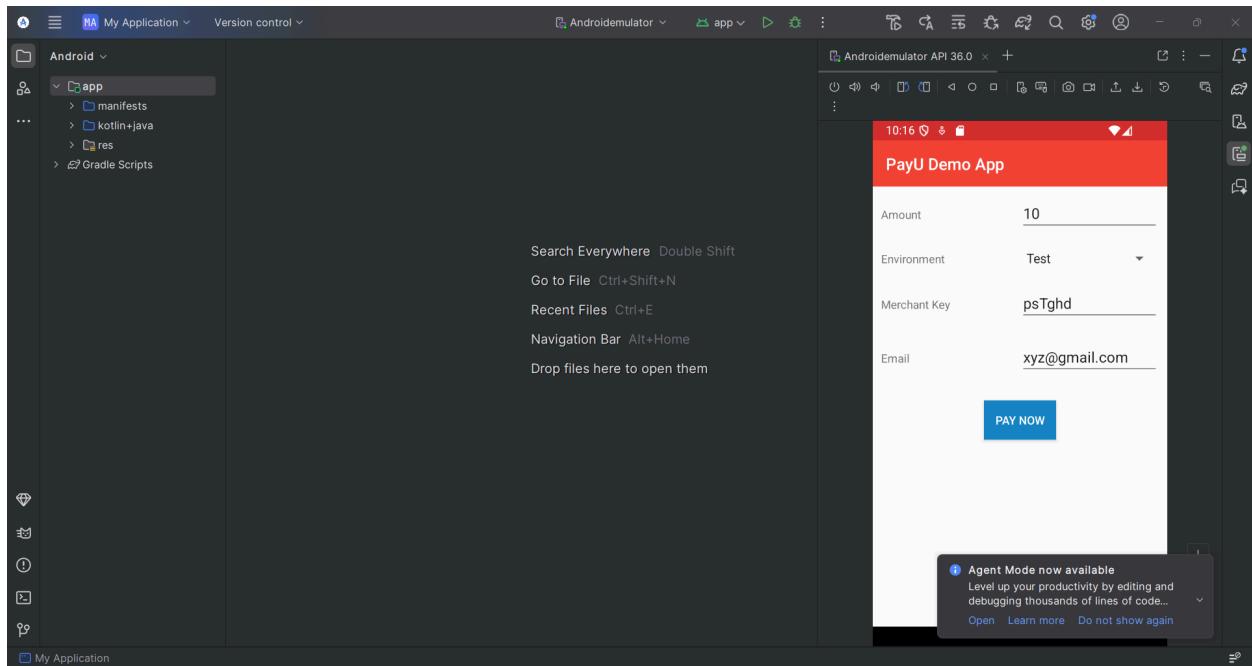
7.1 Creating Virtual Device

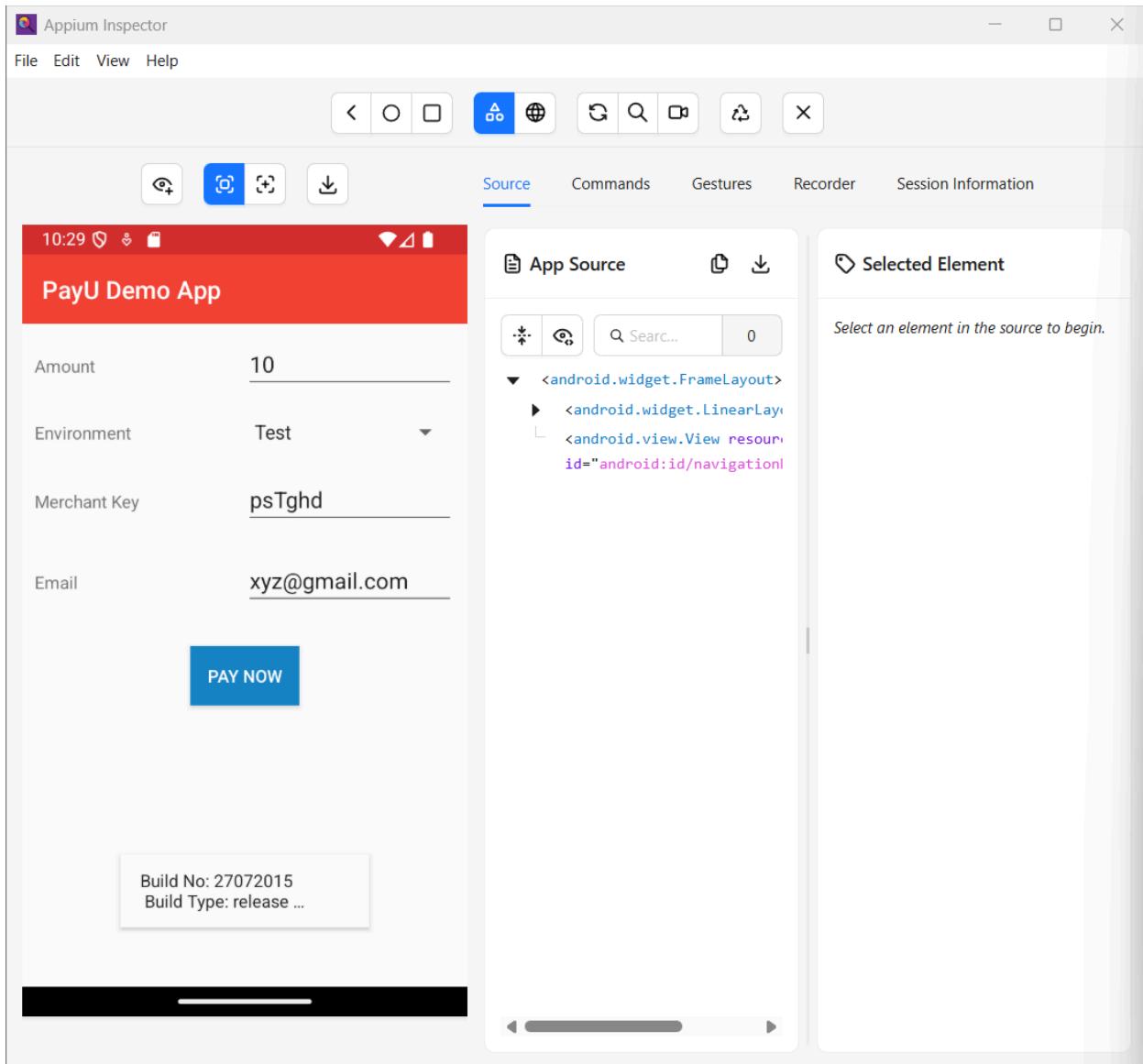
- Opened Android Studio → AVD Manager.
- Created a new Virtual Device with required Android version.

7.2 Verification

- Launched Emulator successfully.
- Verified device connection using:
adb devices

Purpose: Emulator is used for testing without a physical device.





8. Real Android Device Configuration

8.1 Device Preparation

- Enabled **Developer Options** on Android device.
- Enabled **USB Debugging**.

8.2 Connection

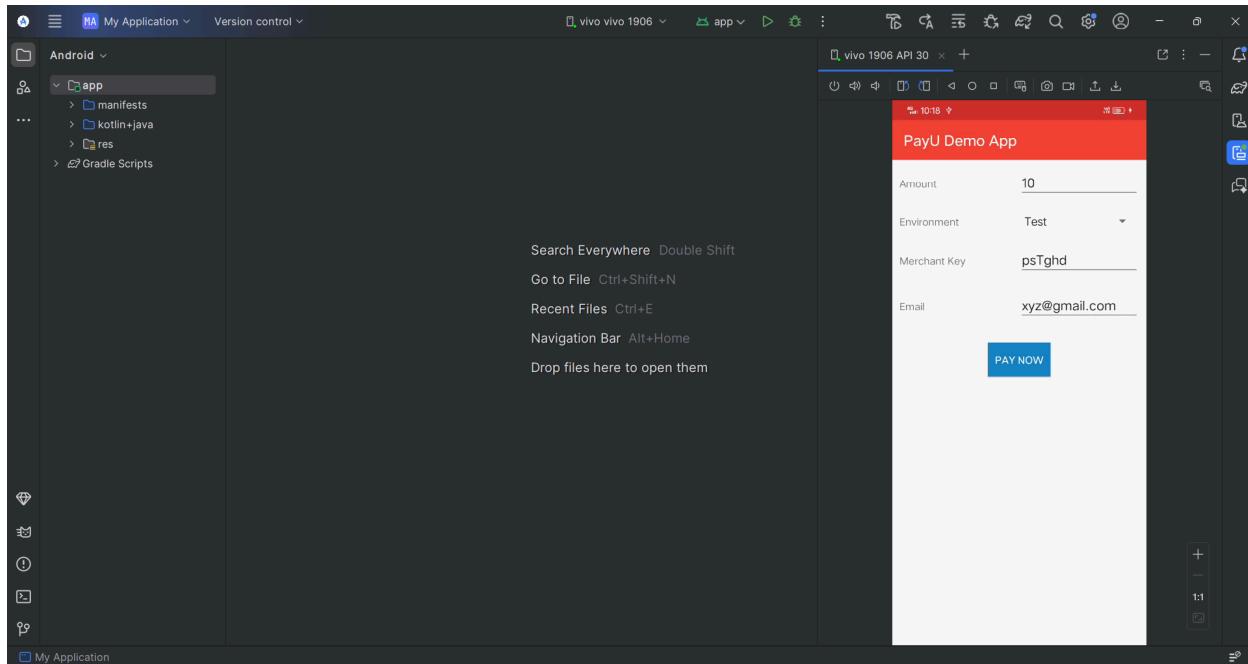
- Connected device to system via USB cable.

- Verified connection using:
adb devices

8.3 Authorization

- Allowed USB debugging permission on device.

Purpose: Real device testing ensures real-world behavior validation.



9. Installing APK on Real Device

- The same APK file was installed on the real Android device.
- Installation was done using:
 - ADB commands, or
 - Automation code written inside the Maven project
- The application was successfully installed and launched on the real device.

```
C:\Windows\System32\cmd.exe + 
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\pavit\apk files>adb.exe install SampleApp.apk
Performing Streamed Install
Success

C:\Users\pavit\apk files>
```

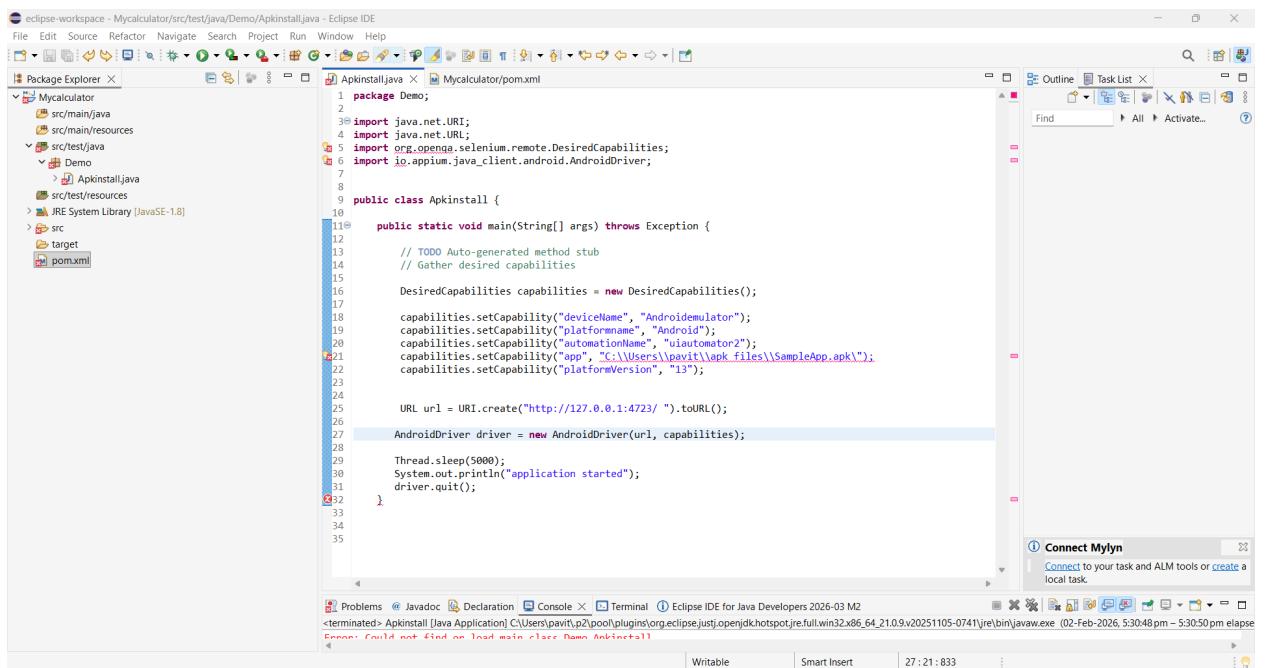
10. Maven Project Creation

10.1 Creating the Maven Project

- A new Maven project was created using the standard Maven archetype.
- Maven automatically generated the project structure with folders such as:
 - `src/main/java`
 - `src/test/java`
 - `pom.xml`

10.2 pom.xml Configuration

- Required dependencies (such as Appium Java Client, Selenium, and TestNG) were added to the `pom.xml` file.
- Maven was used to download and manage all dependencies automatically.



The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows the Maven project structure for "Mycalculator". It includes the `src/main/java`, `src/main/resources`, `src/test/java` (containing `Demo`), `src/test/resources`, and `pom.xml`.
- Editor:** Displays the `ApkInstall.java` file. The code is as follows:

```
1 package Demo;
2
3 import java.net.URI;
4 import java.net.URL;
5 import org.openqa.selenium.remote.DesiredCapabilities;
6 import io.appium.java_client.android.AndroidDriver;
7
8
9 public class Apkinstall {
10
11     public static void main(String[] args) throws Exception {
12
13         // TODO Auto-generated method stub
14         // Gather desired capabilities
15
16         DesiredCapabilities capabilities = new DesiredCapabilities();
17
18         capabilities.setCapability("deviceName", "Androidemulator");
19         capabilities.setCapability("platformName", "Android");
20         capabilities.setCapability("automationName", "uiAutomator2");
21         capabilities.setCapability("app", "C:\\Users\\pavit\\apk_files\\SampleApp.apk");
22         capabilities.setCapability("platformVersion", "13");
23
24
25         URL url = URI.create("http://127.0.0.1:4723/").toURL();
26
27         AndroidDriver driver = new AndroidDriver(url, capabilities);
28
29         Thread.sleep(5000);
30         System.out.println("application started");
31         driver.quit();
32     }
33
34
35 }
```

Bottom Status Bar: Shows "Writable", "Smart Insert", and the timestamp "27:21:833".