CONVERTER FOR SCIENTIFIC UNITS (Testing Using APPIUM)

Name : Y.Haswanth kumar
Reg No : 192111538
Dep.t : CSE
Subject : Software Testing for Android Applications
Code : CSA3732

PROJECT INCLUDES

- Objective
- Abstract
- Proposed System
- Flow chart
- Concept map
- No of Test Cases

- Apps and Tools Installation
- Appium Server
- Implementing Testing
- Test Case Outcomes
- Conclusion

OBJECTIVE

- To Test the different strategies (Test Cases) on Unit Converter Application.
- To find all possible Outcomes (Positive and Negative) of the Test.

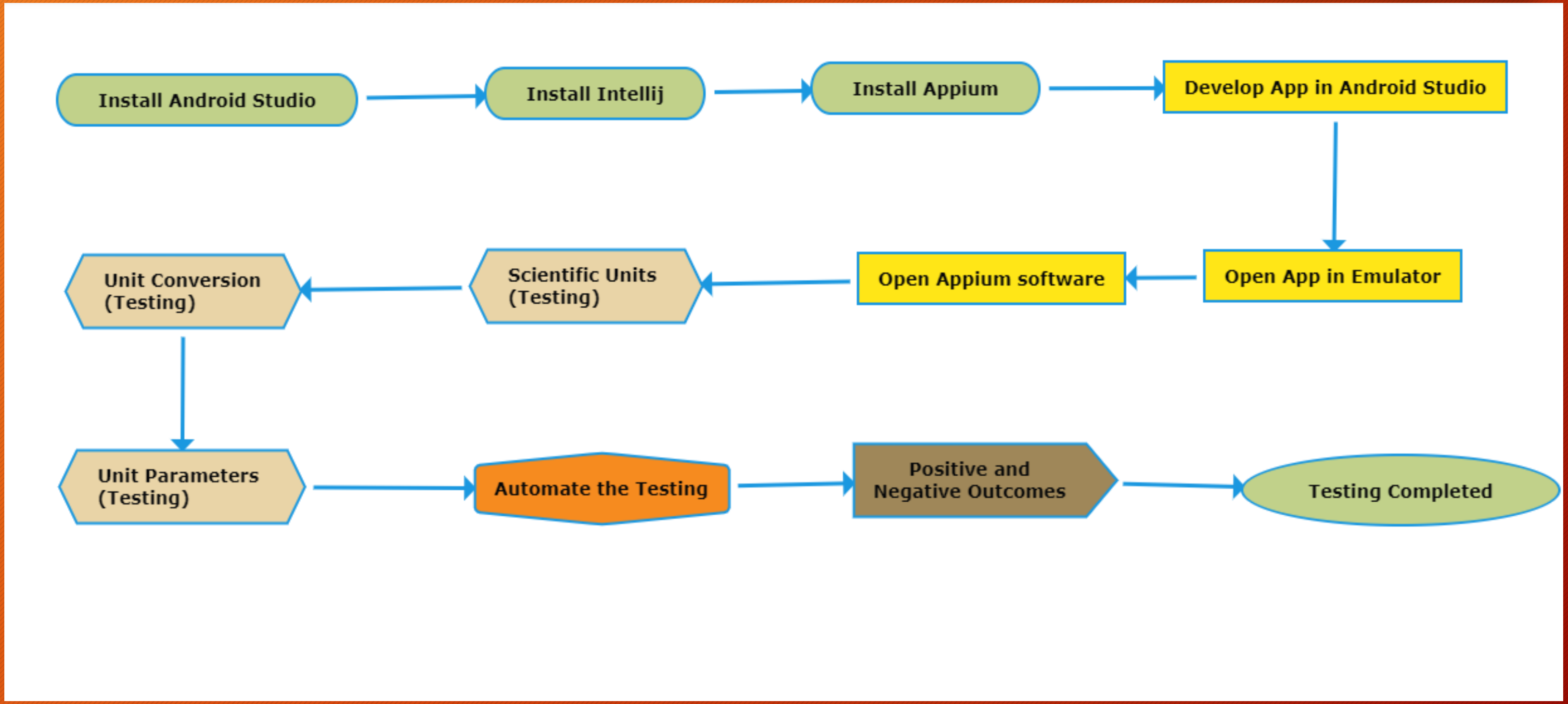
ABSTRACT

- Unit converter consists of many conversions like force, energy , time etc.
- User can give Input of Scientific Units.
- It consists of parameters for every Unit.
- For every unit you can change the parameters.
- Based on Parameters you can we can convert all the scientific units present in it.
- User Can get the Output in Desired Parameter.

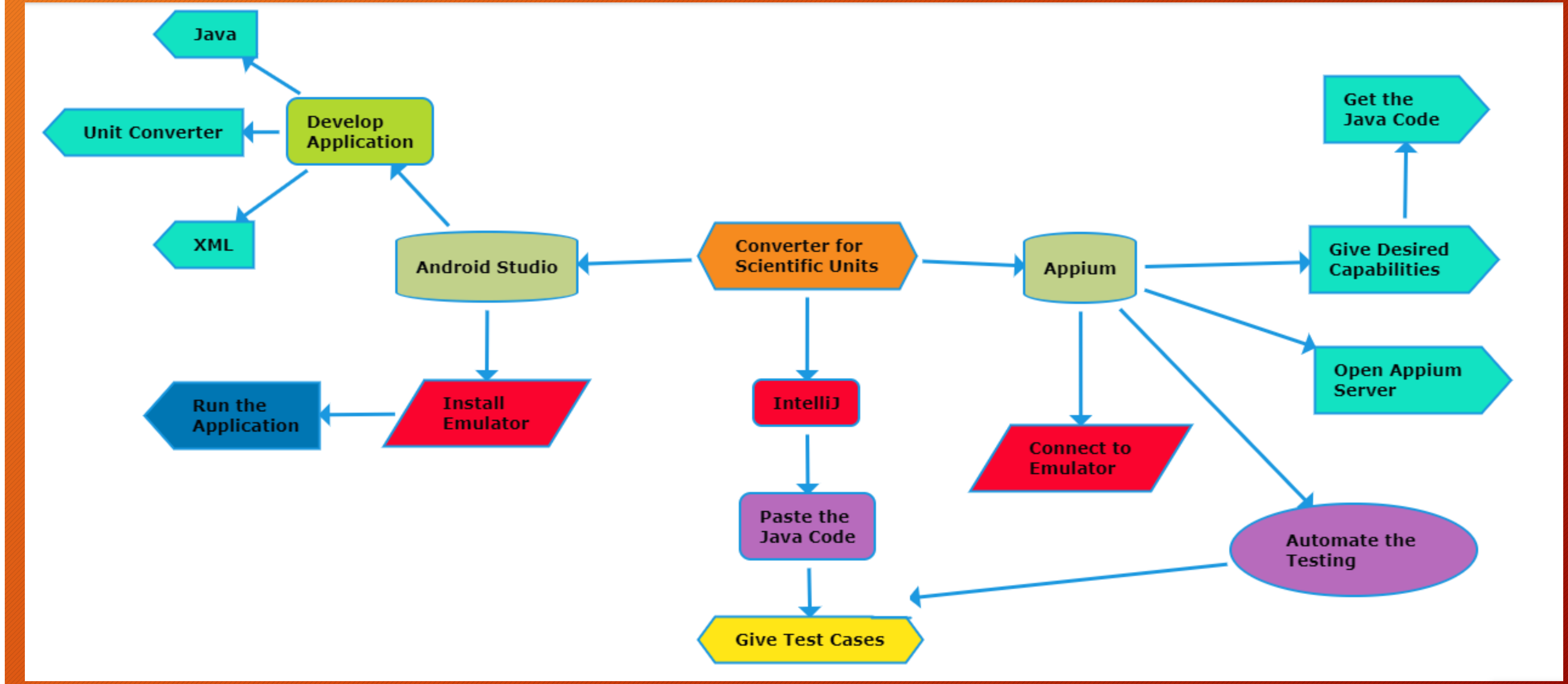
PROPOSED WORK

- Test cases like App Launch , Performance , Conversions and Parameters will be Tested.
- Testing is done using Appium Software.
- To test First Install Android Studio , IntelliJ and Appium Software.
- Develop Unit Converter Application in Android Studio.
- Run the Application using Android Studio Emulator.
- Then find the Positive and Negative outcomes of Testing.

DATA FLOW DIAGRAM


Install Android Studio
Install Intellij
Install Appium
Develop App in Android Studio
Open App in Emulator
Open Appium software
Scientific Units (Testing)
Unit Conversion (Testing)
Unit Parameters (Testing)
Automate the Testing
Positive and Negative Outcomes
Testing Completed

CONCEPT MAP


Java
Unit Converter
Develop Application
XML
Android Studio
Converter for Scientific Units
Appium
Get the Java Code
Give Desired Capabilities
Open Appium Server
Run the Application
Install Emulator
IntelliJ
Connect to Emulator
Paste the Java Code
Automate the Testing
Give Test Cases

NUMBER OF TEST CASES

- App Launch
- User Name
- Temp Converter
- Weight converter
- Time Converter

- String Input
- Performance
- Clear Input Button
- Font Color , Size
- Parameters of all converters

APPLICATIONS USED

APPIUM

INTELLIJ

ANDROID STUDIO

APPS AND TOOLS

- **Appium**

1. Open source Automation tool.
2. It is also a Automation Framework.
3. Automates different types of mobile applications testing.
4. It can test web , Native and Hybrid application.
5. It supports multiple programming languages.
   (Java , python , ruby and many more)

6.It works on Emulator as well as Real Devices.

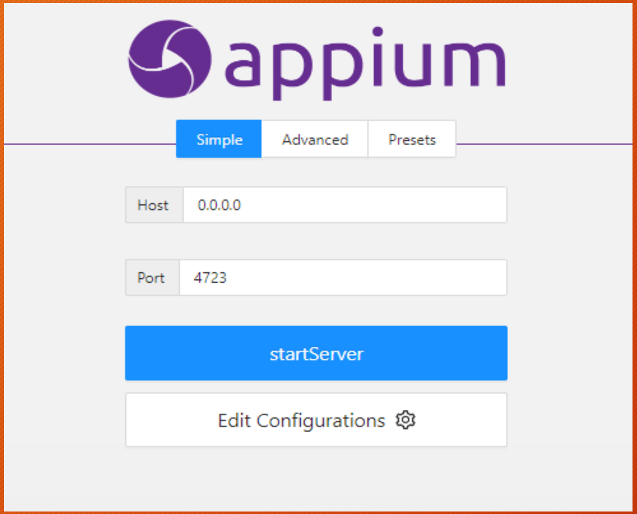
APPS AND TOOLS

- Android Studio

1. It is Integrated development Environment.
2. It has fast Emulator for App Testing.
3. Used to Develop applications for Android devices.
4. It is an open source software.

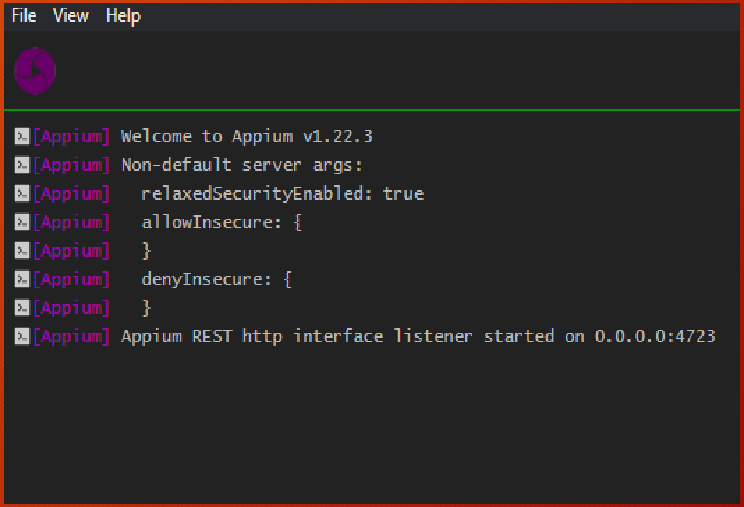
APPIUM SERVER

Before starting server



```
Host  0.0.0.0
Port  4723
```

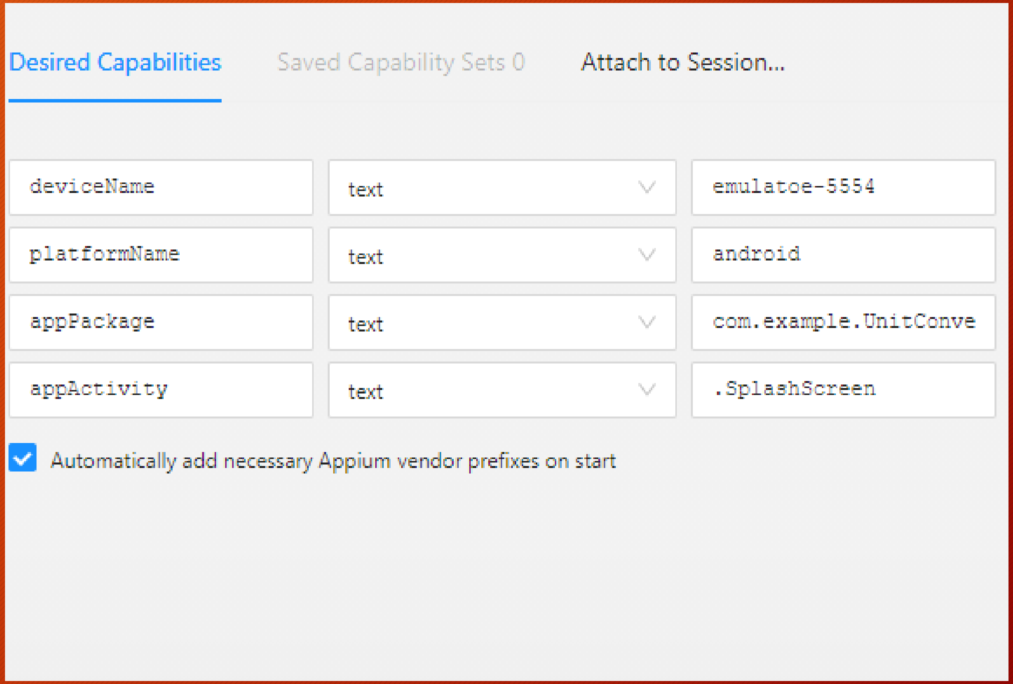
After starting Server



```
[Appium] Welcome to Appium v1.22.3
[Appium] Non-default server args:
[Appium]   relaxedSecurityEnabled: true
[Appium]   allowInsecure: {
[Appium]   }
[Appium]   denyInsecure: {
[Appium]   }
[Appium] Appium REST http interface listener started on 0.0.0.0:4723
```

DESIRED CAPABILITIES(APPIUM INSPECTOR)

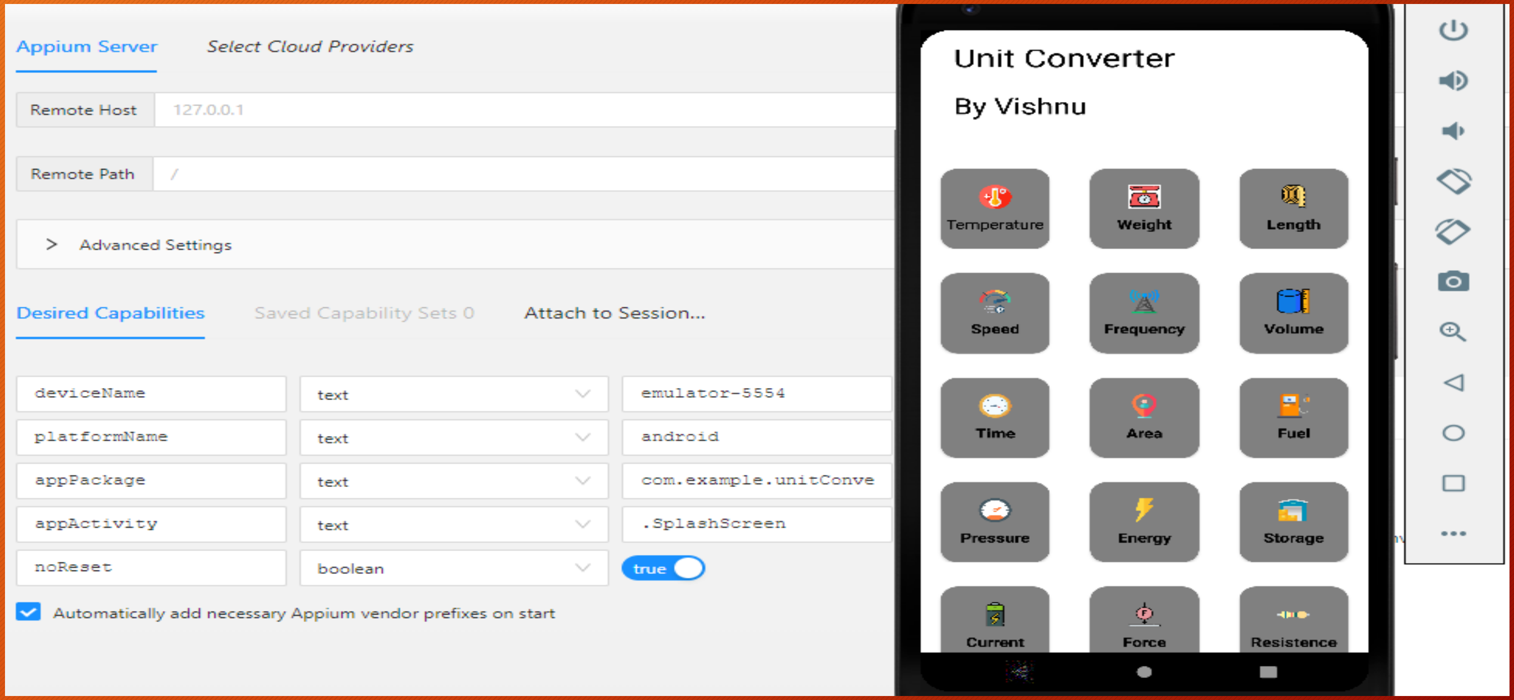
- Give the Desired Capabilities in Appium Inspector.
- It will start the session and gives you java code to start testing.
- Paste the code in IntelliJ IDE and Start the Testing.



deviceName	text	emulatoe-5554
platformName	text	android
appPackage	text	com.example.UnitConve
appActivity	text	.SplashScreen

[✓] Automatically add necessary Appium vendor prefixes on start

IMPLEMENTATION AND TESTING



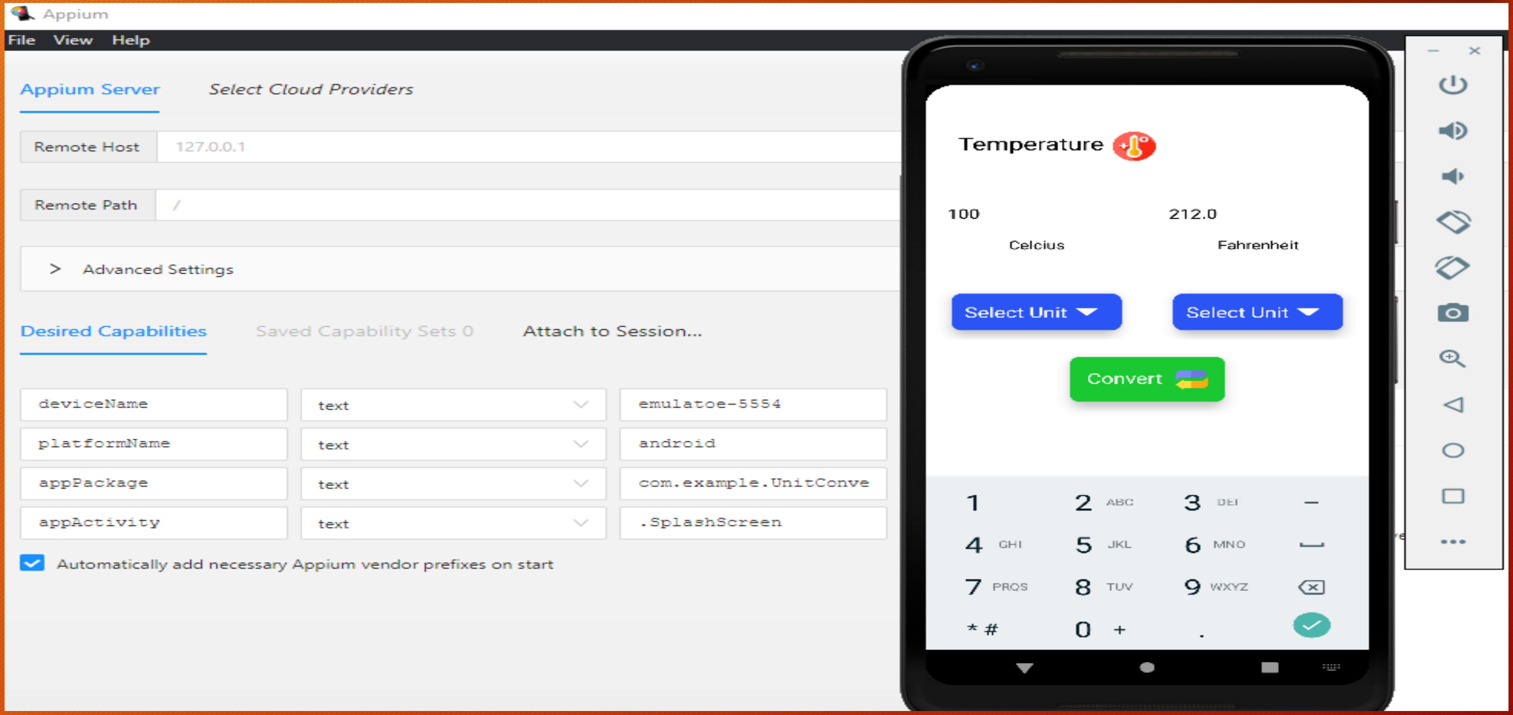
Remote Host: 127.0.0.1

Remote Path: /

deviceName	text	emulator-5554
platformName	text	android
appPackage	text	com.example.unitConve
appActivity	text	.SplashScreen
noReset	boolean	true

☑ Automatically add necessary Appium vendor prefixes on start

IMPLEMENTATION AND TESTING



Remote Host 127.0.0.1

Remote Path /

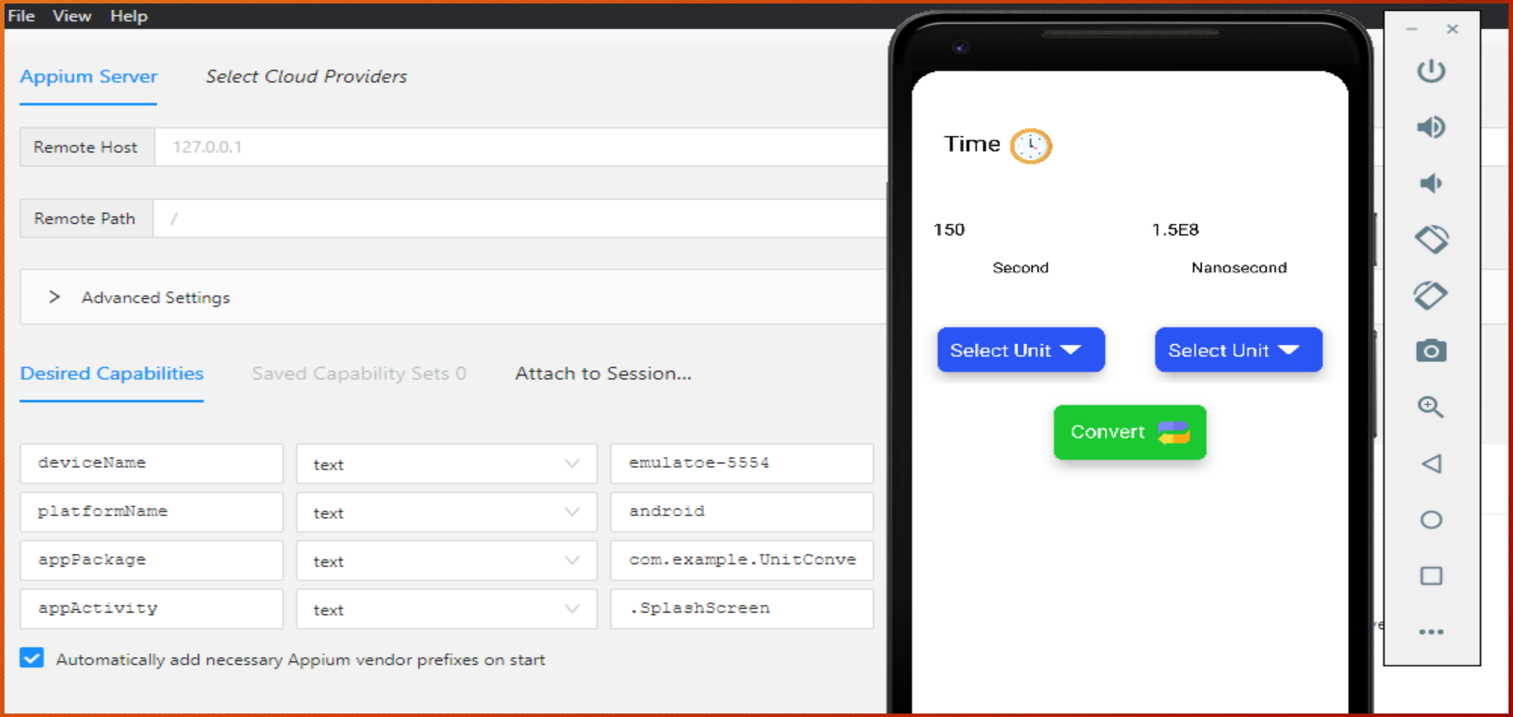
deviceName	text	emulatoe-5554
platformName	text	android
appPackage	text	com.example.UnitConve
appActivity	text	.SplashScreen

✓ Automatically add necessary Appium vendor prefixes on start

100 Celcius

212.0 Fahrenheit

IMPLEMENTATION AND TESTING



Remote Host 127.0.0.1

Remote Path /

deviceName	text	emulatoe-5554
platformName	text	android
appPackage	text	com.example.UnitConve
appActivity	text	.SplashScreen

[✓] Automatically add necessary Appium vendor prefixes on start

150 Second

1.5E8 Nanosecond

TEST CASE OUTCOMES

Test Cases	Outcome
1.Verify the application launch in Emulator	Positive
2.Check whether the User Name is visible	positive
3.Verify User can use Temperature Converter	positive
4. Verify User can use Weight Converter	positive
5. Verify User can use Time Converter	positive
6.Check Whether User can give String input	Negative
7.Verify the Performance of Application	positive
8.Verify User can clear the input	positive
9.Verify Use can change Font size and Colour	Negative
10.Check the parameters of all converters	positive

CONCLUSION

- All the Units , Conversions , Parameters and Possible Outcomes were Tested using the Appium Server.
- Both Positive and Negative Outcomes were found
- 80 Percentage of the Test cases were positive.
- 20 percentage were negative.
- By enabling String Input and Font size , Colour Changes we can Overcome Negative Outcomes.