**## Project**

This Testing framework is designed to test the NATIVE app of Seek.com (i.e. Android App). The logic would be more or less the same for any Android native app, i.e. the sign in functionality.

**## Requirement**

The design of this Testing Framework is in keeping with the requirements of the Home Assignment,( as per the email received from the Careers Team of ‘Avant Mutual’ on 7th May 2020, for the role of ‘Test Automation Engineer’.

**## Home Assignment(as quoted in the email)**

You are tasked to implement a testing framework and a test case for any NATIVE mobile app of your choice (iOS/Android) available on the market that has a sign-in/sign-up functionality. You can be as creative as you like and choose any app that you are familiar with.

This test-case should locate elements of the sign-in screen and set some valid values to proceed with signing in. The test case is considered as passed successfully when the next screen after sign-in appears. We expect you to use the Appium framework, however you are able to utilise other frameworks if you are able to explain why that framework was the most appropriate.  The solution should be submitted as github, gitlab or bitbucket repository.

**## Downloads**

1. Download Java.

Source: <https://www.java.com/en/download/>

1. Android Studio and SDK tools are installed.

Source: <https://developer.android.com/studio>

1. Download and install Appium

Source: <http://appium.io/downloads.html>

1. Nodejs download

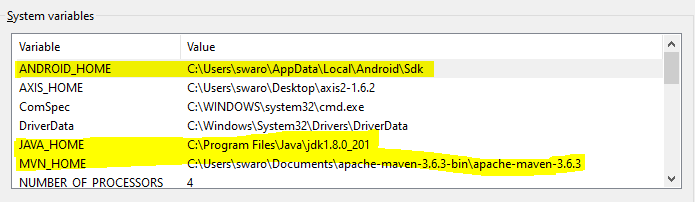
Source: <https://nodejs.org/en/download/>

1. Maven download

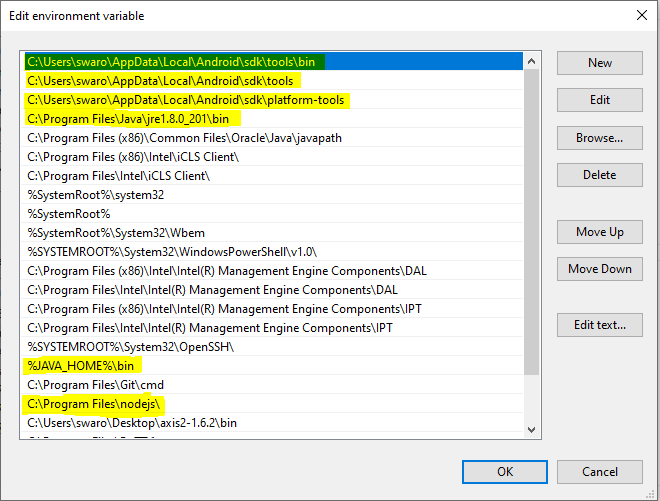
Source: <https://maven.apache.org/download.cgi>

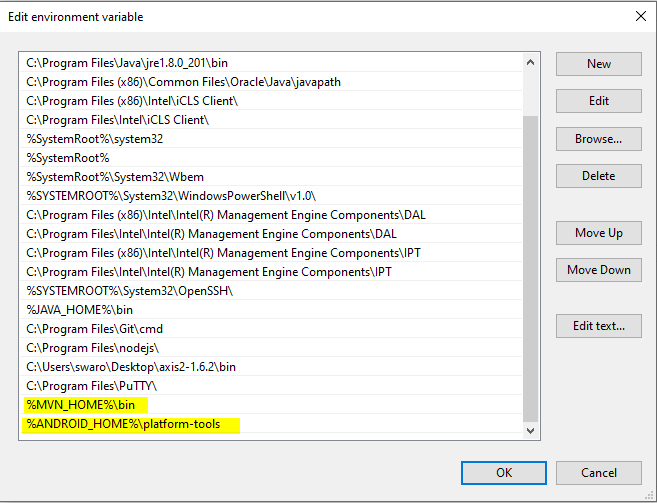
**## Pre-requisites**

The necessary software, as given in the download-section are downloaded and installed accordingly. The environment variables for the ANDROID\_HOME, JAVA\_HOME, MVN\_HOME are set as shown below:



The Path environment variables are set as shown below:





Note:

The configuration shown above, is considering the fact, that we’ve a Windows 10, 64-bit system in place to run the framework.

Note:

The Framework was tested on a real Android Device. So we need to have an android device in place. The Framework was tested on Android 9.

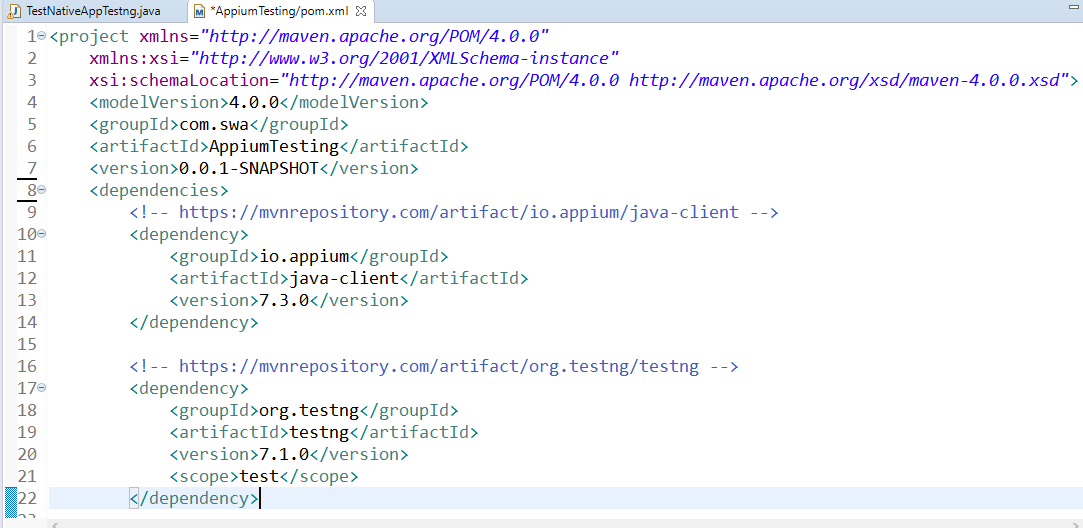
The developer options in the System settings of the Android device needs to be enabled. Also please do not forget to **download the Seek app from the Google playstore** on your Real Android device.

Note:

The Framework was developed and built on Eclipse IDE.

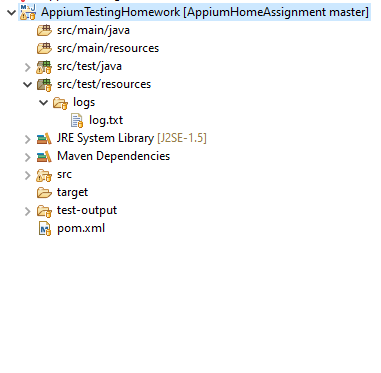
**## Framework**

The framework, which we’ve designed, is a TestNG framework, implemented as a Maven Project. And the libraries could be configured by quoting the dependencies in the POM.xml.

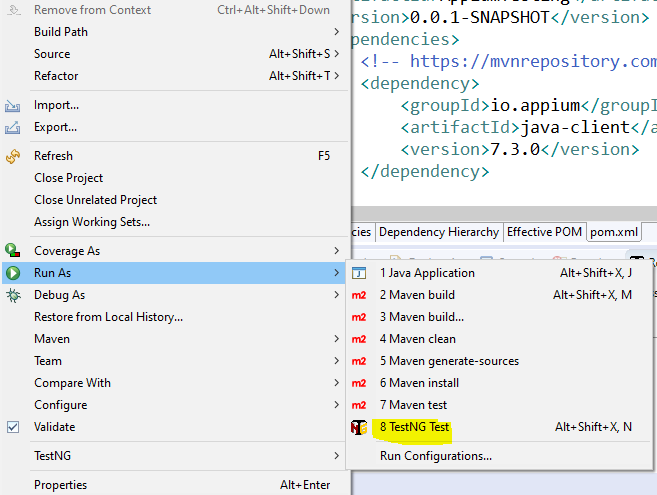


**## Running the Framework**

To run the Framework, right click on the ‘AppiumTestingHomework’, as shown in the below pic,

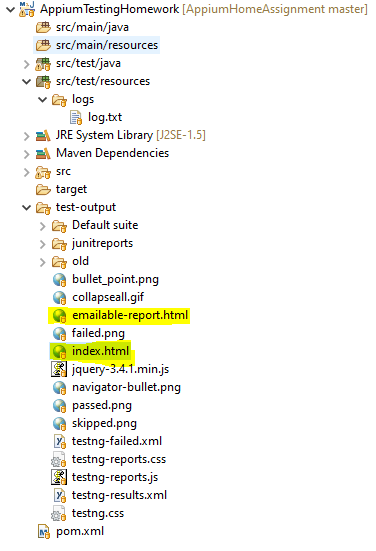


Click on the ‘Run As’ and TestNG Test as shown in the below screenshot.

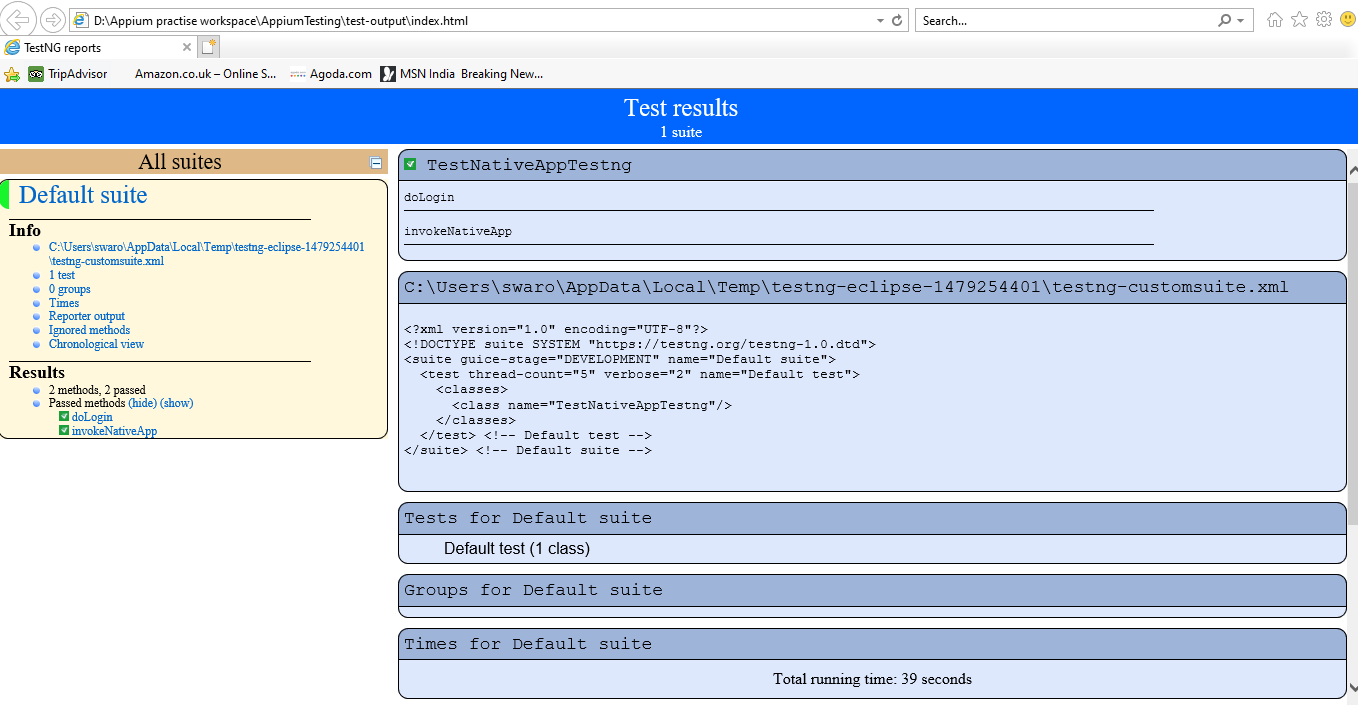


**## Reports Generated**

The TestNG framework, that we’ve used has automatic reporting enabled, so whenever we run the Framework, the reports generated could be examined by right clicking on the ‘AppiumTestingHomework’ framework, clicking on refresh and clicking on the test-output as shown in the screenshot below.

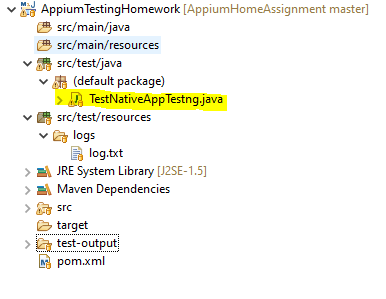


As shown in the above pic, we need to open the index.html/emailable-report, wherein we can see the detailed report of each test case run. Please refer the below pic for the same.



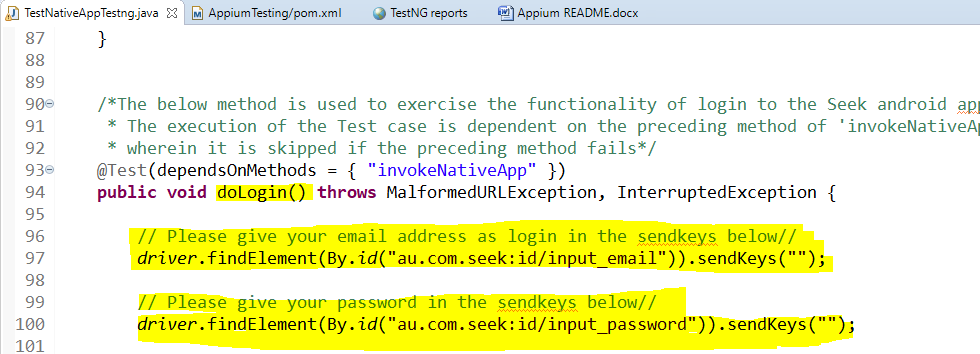
**## The Test Cases location**

The location of the Test Case in the Framework, is as shown in the below screenshot. The ‘TestNativeAppTestng.java’ could be opened to see the work done.



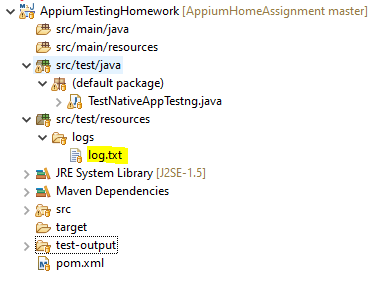
**## The login credentials for Seek app.**

Please give the login credentials in the **sendkeys** of the **doLogin** method as shown below. This is needed in order to pass the Test case and login into the Seek app, in the real device.

****

**## The logs generated**

The logs could be examined after every Framework run. The location of the logs, could be seen below.



**## Further improvements/enhancements to the Framework.**

While, every care and effort is taken to ensure, that the Framework is robust and runs smoothly, yet there is always a room for improvement in any Testing Framework.

Additional reporting features could be incorporated in the Framework, such as ReportNG.

The total run time of 39 seconds of the Framework, could be further reduced to 20 seconds by the usage of efficient locators, such as Accessibility-id.

Actions class could be used in place of sendkeys, to ensure that the inputs given are fool-proof.

A separate config file could be used to place the login credentials.

And any more suggestions to improve on the Framework, are most welcome.

**## Thanks giving note**

Here, I’d like to sincerely thank you, for taking time to go thru my hard work done.