

## **Application:**

A piece of code written to perform a required task is called as Application.

## **Types of Applications:**

1. Standalone Application
2. Client Server Application
3. Web Application

### **Standalone Application**

- No Internet
- No database
- No Server
- Installations is mandatory
- Single User will be using the application at a time

### **Web Application**

Any application which is launched using Web browser and Web URL

- Internet
- Database
- Server
- No Installation
- Multiple Users can use the application at a time

### **Client Server Application**

Two Application => **Client App** installed in local machine and **Server App** installed in Server

- Internet
- Database
- Server
- Installation is mandatory (**Client app**)
- Multiple Users can use the application at a time

In Client Server Application => we focus on Client App

Client App is the app which we install in our local machine(Mobile)

## **Types of Client App**

1. Native App
2. Web App
3. Hybrid App

### **Native App:**

App which is developed for a particular platform (i.e.) android or iOS.

Native App's are Platform specific.

Source code will be different for Android and iOS

Eg: Messages app, Calculator, Calendar

### **Languages used for development**

=> Java (Android)

=> Objective C or Swift (iOS)

**Native view** – the view which we get when we open the native app is called as Native view

**Native Elements** – the elements present in Native View is called as Native elements

**Native Context** – the Collection of Native Elements is called as Native Context

### **How to Capture Native Elements?**

- Using Fire Finder Tool in FireFlink.

### **Web App:**

App which is launched using the mobile browser or web URL is called as Web App.

Not Platform specific

Eg: Flipkart.com, Amazon.com, etc...

Languages used for development of Web App

=> HTML/CSS

**Web view** – the view which we get when we open the web app is called as web view

**Web Elements** – the elements present in web View is called as web elements

**Web Context** – the Collection of web Elements is called as web Context

How to Capture Web Elements?

- Using Chrome browser
- Fire Finder Tool

### Hybrid App:

Hybrid app is combination of both Native and Web App

Hybrid app has both Native and Web View.

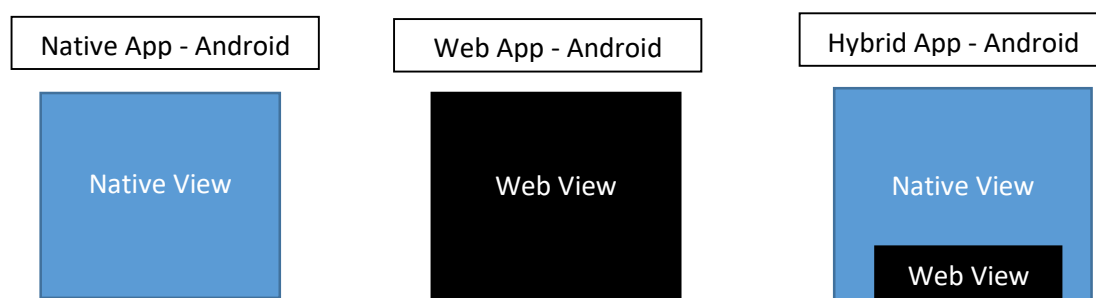
Eg: Flipkart app, Instagram app, Gmail app, etc...

Languages used for development

=> React JS

How to Capture Web Elements?

- Using Chrome browser (Web elements)
- Fire Finder Tool



## Why we develop scripts in Computer?

Because we can't write Automation scripts on Mobile so we write scripts in Computer and run them in Mobile.

In this case, the computer and the mobile needs to communicate so we have something called as ADB.

ADB – Android Debug Bridge, which is a set of commands which helps communicate between Computer and mobile.

In order to use ADB commands, Installation of Android studio is mandatory.

**UDID: Unique Device ID** which acts as an **ID number** of the mobile device which is connected to the Computer.

ADB command to find UDID in Command Prompt:

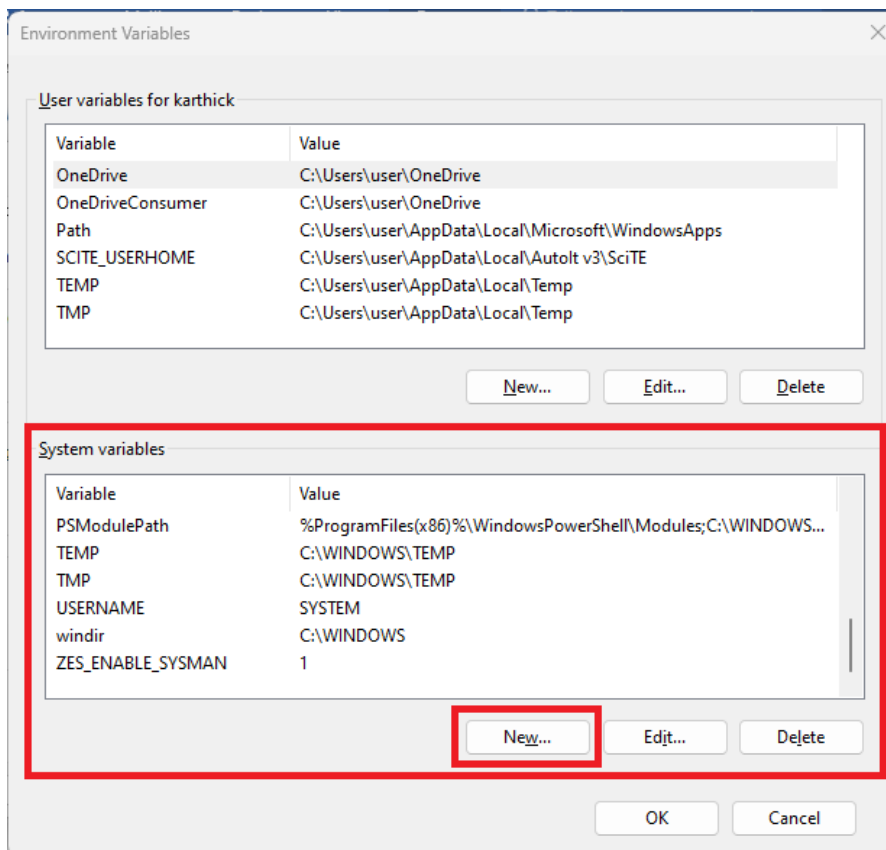
⇒ **adb devices**

## Required Software:

1. Android Studio – Library of ADB commands and for Emulators
2. Appium Server and Appium Inspector – For capturing the elements manually
3. Node JS – Supports Appium
4. Vysor – In order to reflect the Screen of mobile.
5. JDK 11

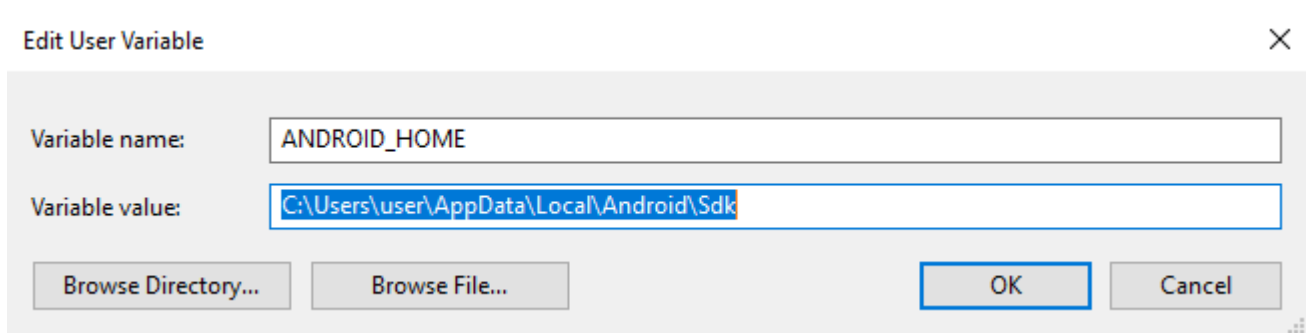
Once Installation is done, Path setting have to be done as mentioned below.

In Environment Variables, go to System Variables and add the below paths:



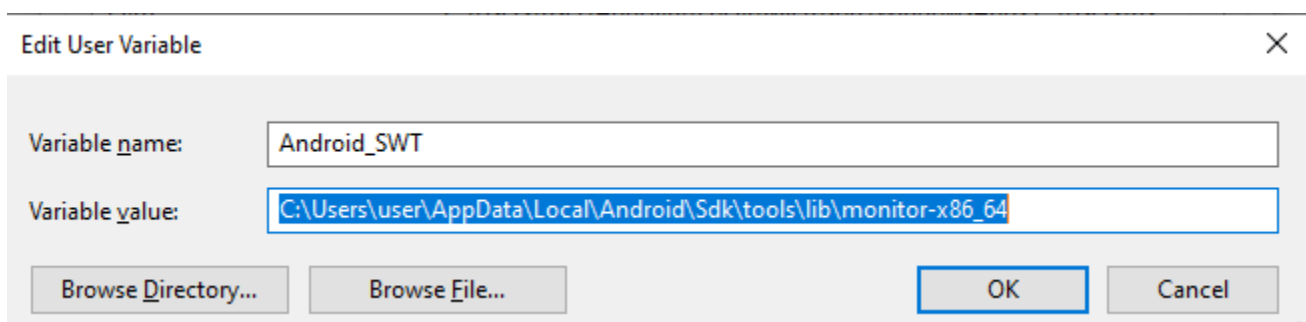
1)ANDROID\_HOME

C:\Users\User\AppData\Local\Android\Sdk



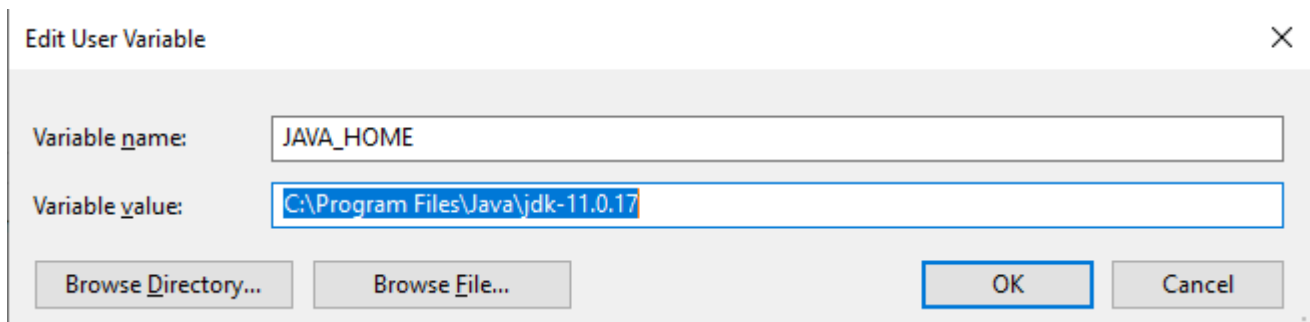
2)Android SWT

C:\Users\User\AppData\Local\ Android\Sdk\tools\lib\monitor-x86\_64

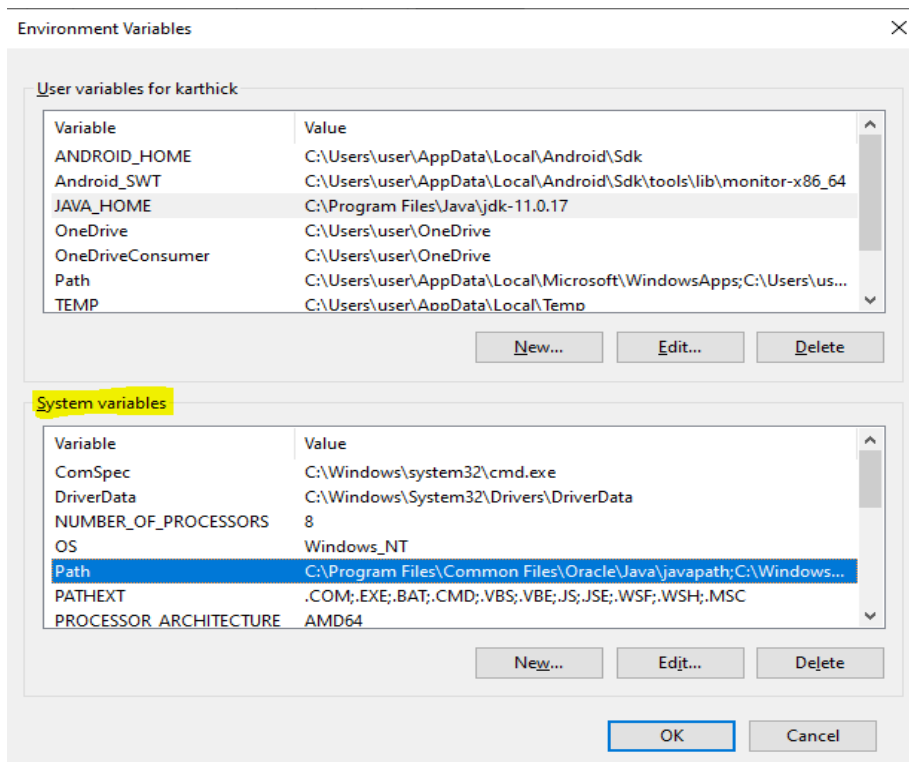


### 3)JAVA\_HOME

C:\Program Files\Java\jdk-11.0.16



In System Variables, double click on Path Variable and add the below mentioned path as per your system

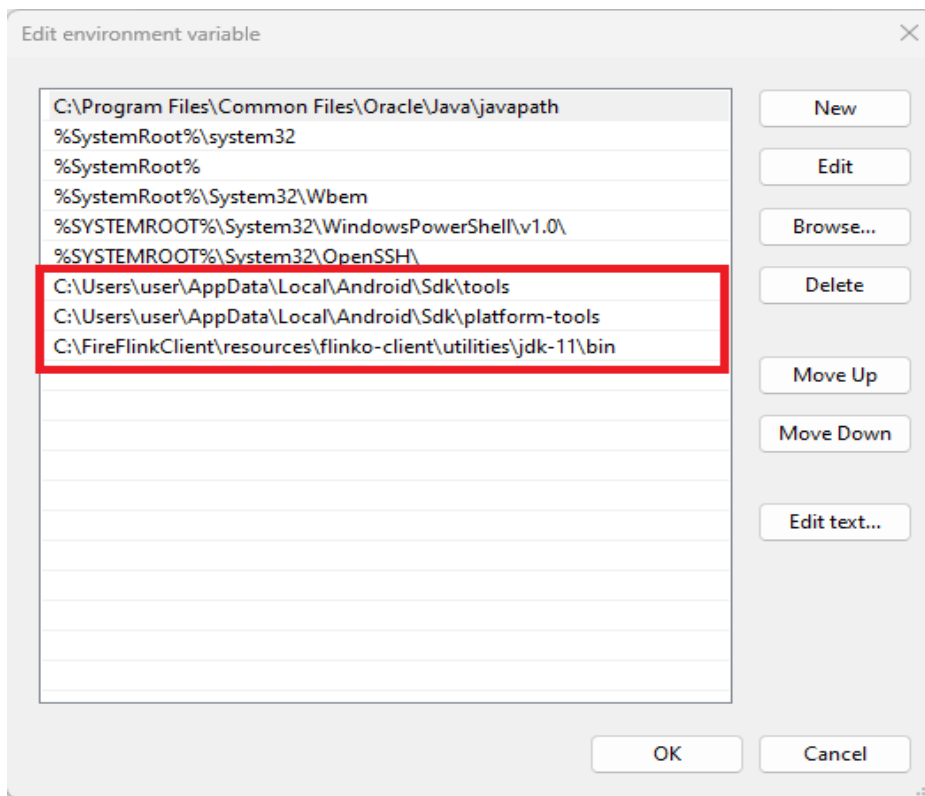


Paths:

C:\Users\user\AppData\Local\Android\Sdk\tools

C:\Users\user\AppData\Local\Android\Sdk\platform-tools

C:\FireFlinkClient\resources\flinko-client\utilities\jdk-11\bin



## How to create Virtual device(Emulator) using Android studio?

- Open Android Studio > Create a Project > Click on Emulator > Click on Device Manager link
- In Device Manager tab click on Create device button (Select Hardware popup appears)
- Choose a Mobile with Play store and click on next (Select Image popup appears)
- Download the preferred OS(Oreo) and click on next (Android Virtual Device popup appears)
- Click on Finish.
- In Device manager section the device will be added
- Click on play button of the device, the virtual device will be launched

## Steps to connect Real Mobile Device to Computer:

- Enable Developer option in mobile
- Enable USB debugging
- Connect the Mobile to computer using USB cable

## Two ways to install APP in Virtual Device/Connected Device mobile device:

1. Go to apk file location in your local system and open command prompt and enter the adb Command

ADB commands: **adb install apkFileName.apk**

2. Drag and drop the apk file on the virtual device

## App Package:

It is a package that is basically a file format that is used by Android Operating System. In simple terms it is information given to the server to launch the specified application in the device using the application's java package.

Eg: Amazon (in.amazon.mShop.android.shopping)

## App Activity:

An activity is the screen. In that the activity is very similar to a window in the Windows Operating system. An Android app contains activities, meaning one or more screens like Login activity(Screen), Home activity(screen).

Eg: Amazon (com.amazon.mShop.home.HomeActivity)

To find App Package and App Activity

1. Make sure the App is open in the Virtual device/Connected device and Open Command prompt and enter the adb command  
ADB command:  
a) **adb shell dumpsys window windows | findstr appName** (Note: In the place of appName – the application name should be provided)  
**or**  
b) **adb shell dumpsys window | find "mCurrentFocus"** (Note: Pass the same command as given)
2. Install **Packages Names** App and open app click on any app and it gives the App Package and App Activity of the particular app



## How to capture Elements manually using Chrome for Web Apps?

Open the web view in the Connected Mobile device and Open Chrome browser in the system and enter the URL as mentioned below:

**chrome://inspect**

Once the page is loaded, click on inspect link of the respective web App. The HTML tree structure of the respective Web App will be loaded, use locator strategies to find the required elements.

## How to capture Elements manually using Appium server and Appium Inspector for Native and Hybrid Apps?

Open Appium Server and click on startServer button.

Note: If the server is not starting, change the Port number to '4725' and Click on “startServer” button.

Once server is up & running, Open Appium Inspector and enter the following details

**Remote Server** : 127.0.0.1 (or) 0.0.0.0 (or) <http://localhost>

(Note: Remote server indicates local machine)

**Remote Port** : 4723 (if in Appium server, you have used 4725, use the same)

(Note: Remote port indicates port to be occupied in local machine)

**Remote Path** : /wd/hub

(Note: In Remote path, **wd** - indicates WebDriver and **hub** – indicates application)

Desired Capabilities are keys and values encoded in a JSON object, sent by Appium clients to the server when a new automation session is requested.

JSON is a language which helps us to transfer the data between applications, where the data is stored in key and value pair inside an object.

The desired capabilities have to be mapped to one Json Object in order to launch the automation session.

**Under Desired Capabilities add the following:**

- appPackage** : (value depends on App package of the application)-Mandatory
- appActivity** : (value depends on App Activity of the application) -Mandatory
- platformName** : (value depends on platform of the application) -Mandatory
- deviceName** : (value depends on device used for testing) -Mandatory
- noReset** : **true** (to save the initial settings or cache data) -Not Mandatory
- autoGrantPermission** : **true** (to provide allow permission automatically whenever asked) -Not Mandatory

Once all the desired capabilities are filled save the desired capabilities and click on Start Session button.

Application will be launched and using different locator strategy, we can identify the elements.

Locator strategies for Mobile Elements

- Id
- Name
- Classname
- Accessibility ID
- Xpath

Xpath Syntax: **//classname[@AttributeName='AttributeValue']**

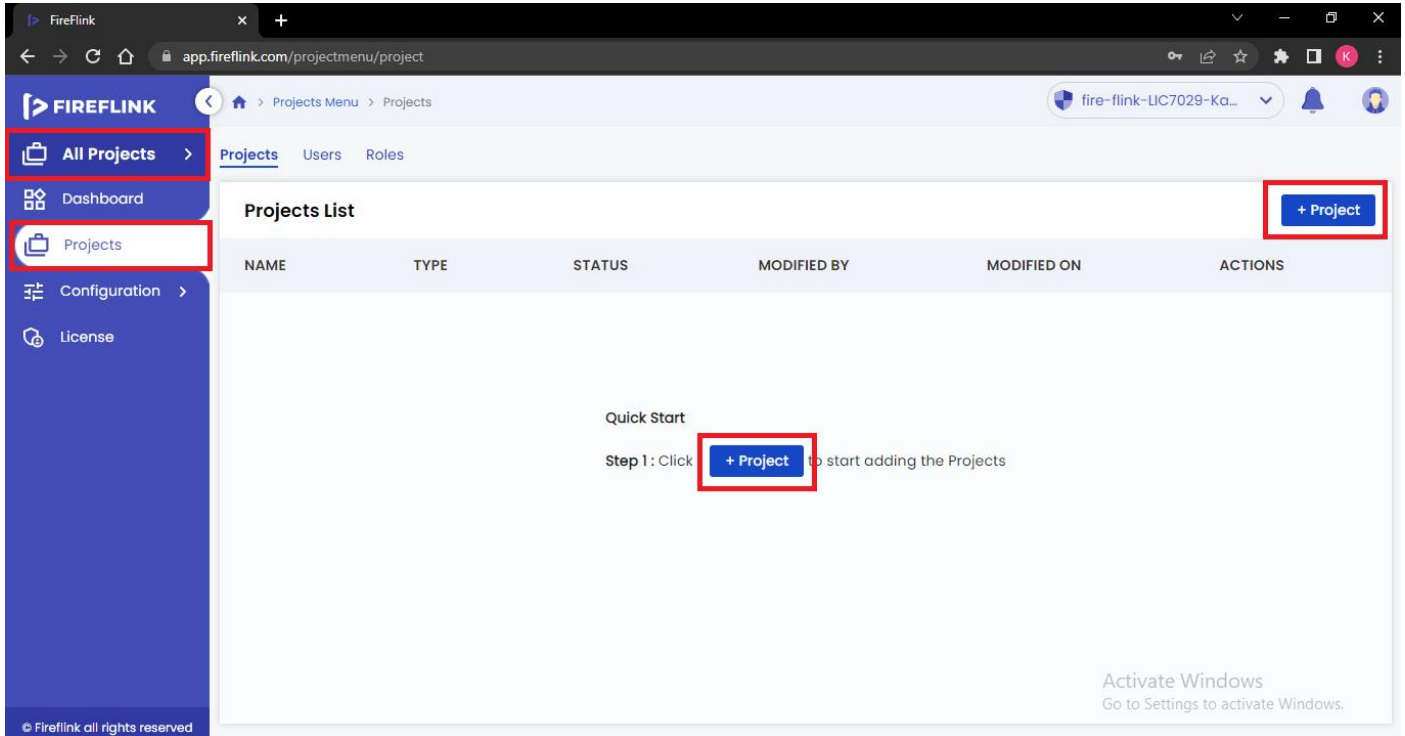
**Note:**

1. **tagname** & **classname** are same
2. **Text** is also considered as an **attribute** in Mobile App.

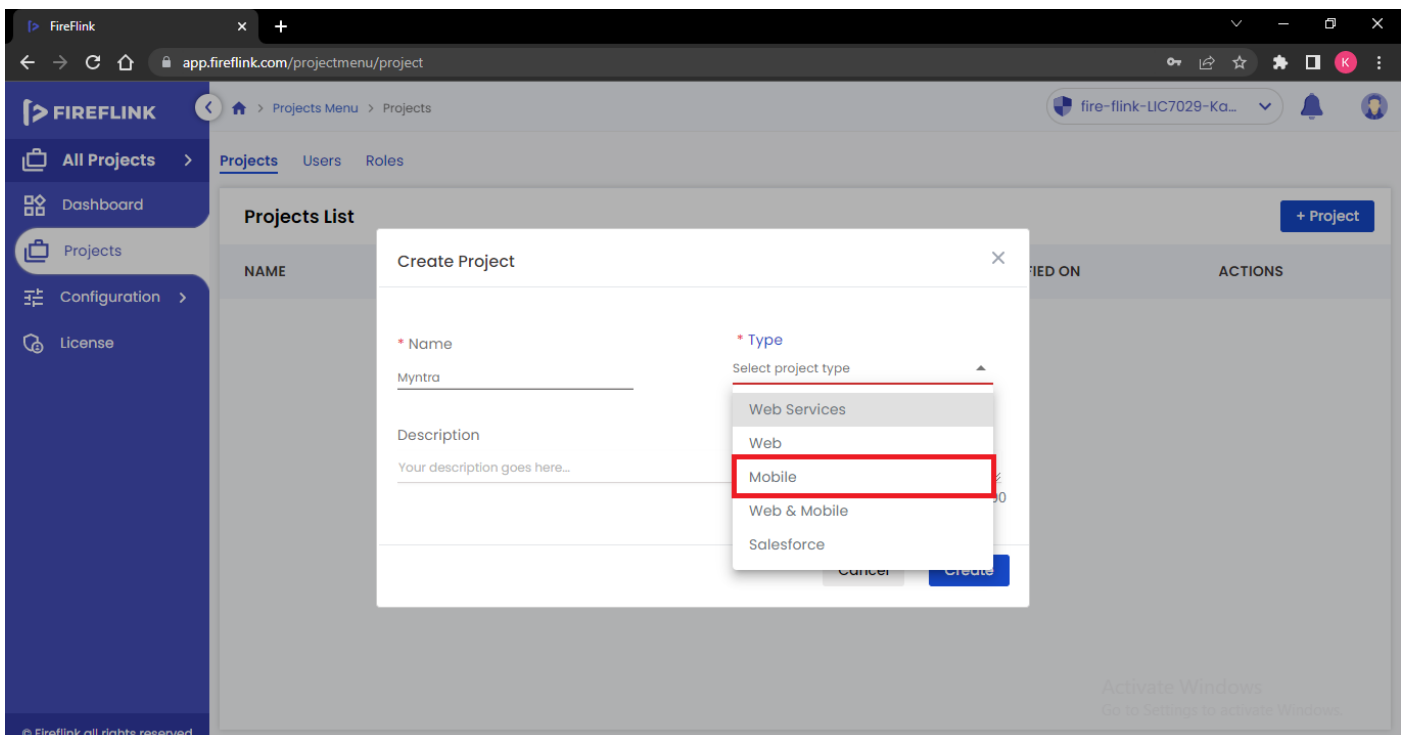
## FireFlink Mobile Automation

Steps to Create a Mobile Project in FireFlink:

Go to All Project level Project section. Click on +Project button.



Create Project popup will be displayed, enter the project name and select the type as Mobile



Select App type based on the type of application. For example, Myntra is Hybrid App.

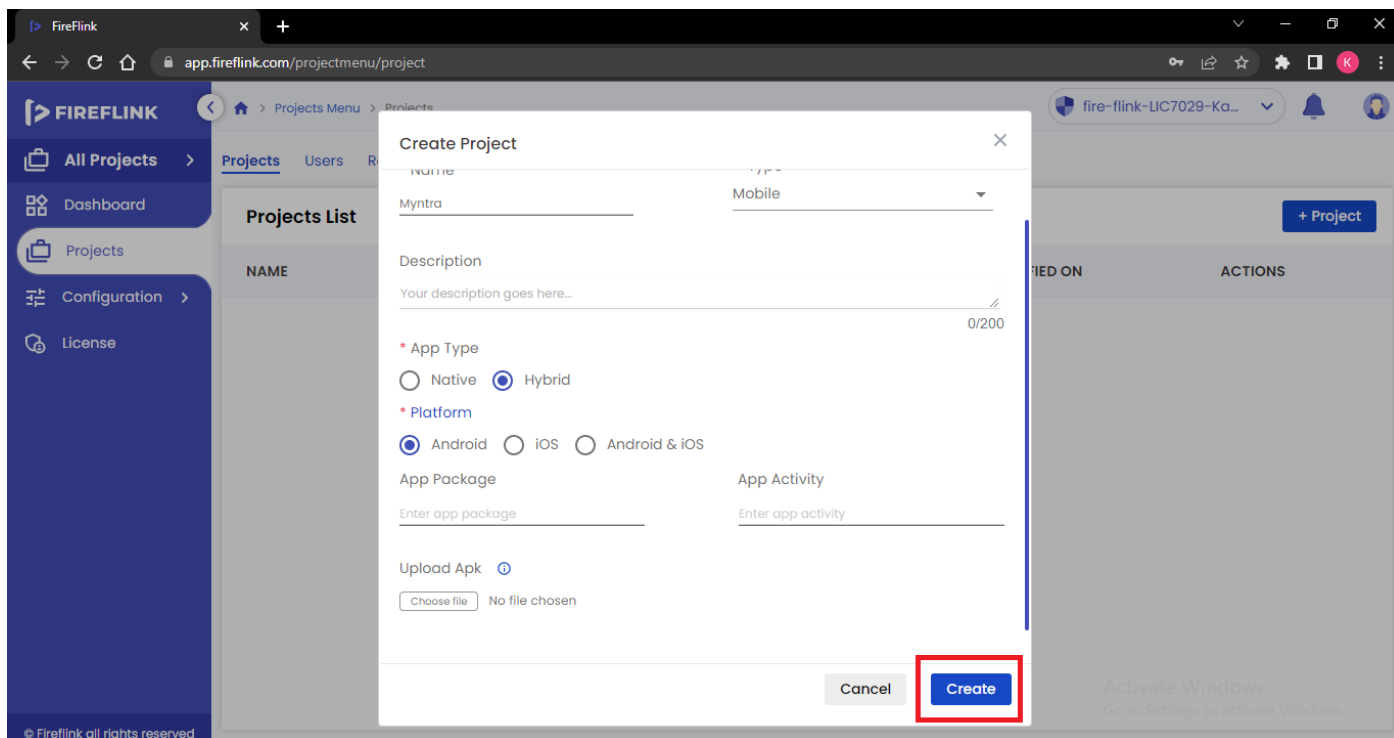
The screenshot shows the 'Create Project' dialog in the Firelink application. The dialog has a white background and a close button in the top right corner. It contains the following fields and options:

- Name:** A text input field containing 'Myntra'.
- Type:** A dropdown menu currently showing 'Mobile'.
- Description:** A text area with the placeholder 'Your description goes here...' and a character count of '0/200'.
- \* App Type:** A section with two radio button options: 'Native' and 'Hybrid'. The 'Hybrid' option is selected and is highlighted by a red rectangular box.
- \* Platform:** A section with three radio button options: 'Android', 'iOS', and 'Android & iOS'. None of these are selected.
- Buttons:** 'Cancel' and 'Create' buttons at the bottom right.

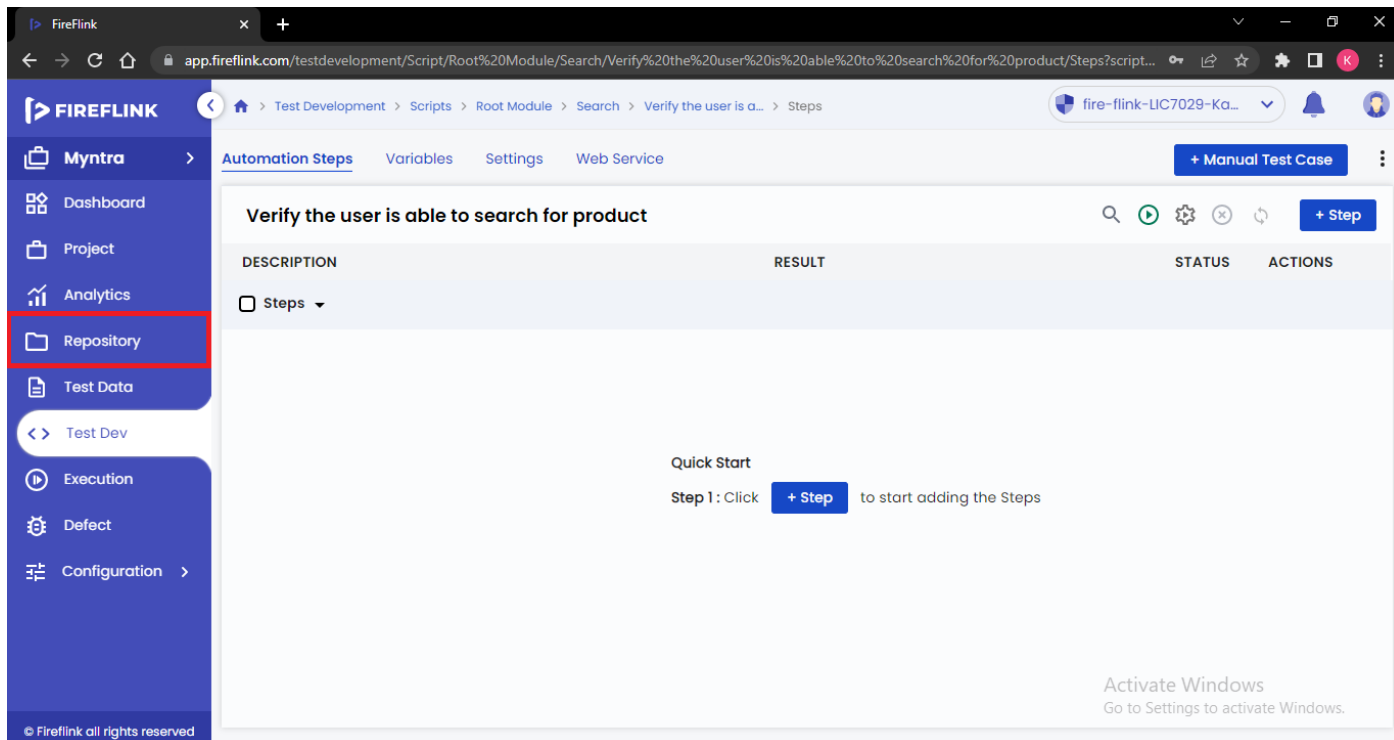
Select the Platform based on which platform, we need to automate the mobile app. For example, Android platform.

This screenshot shows the 'Create Project' dialog with the 'Platform' section highlighted by a red rectangular box. In addition to the 'App Type' section (which remains the same as in the previous image), the 'Platform' section now has the 'Android' radio button selected. The 'App Package' and 'App Activity' fields are now visible below the platform options, each with a text input field. The 'Upload Apk' section at the bottom includes a 'Choose file' button and the text 'No file chosen'.

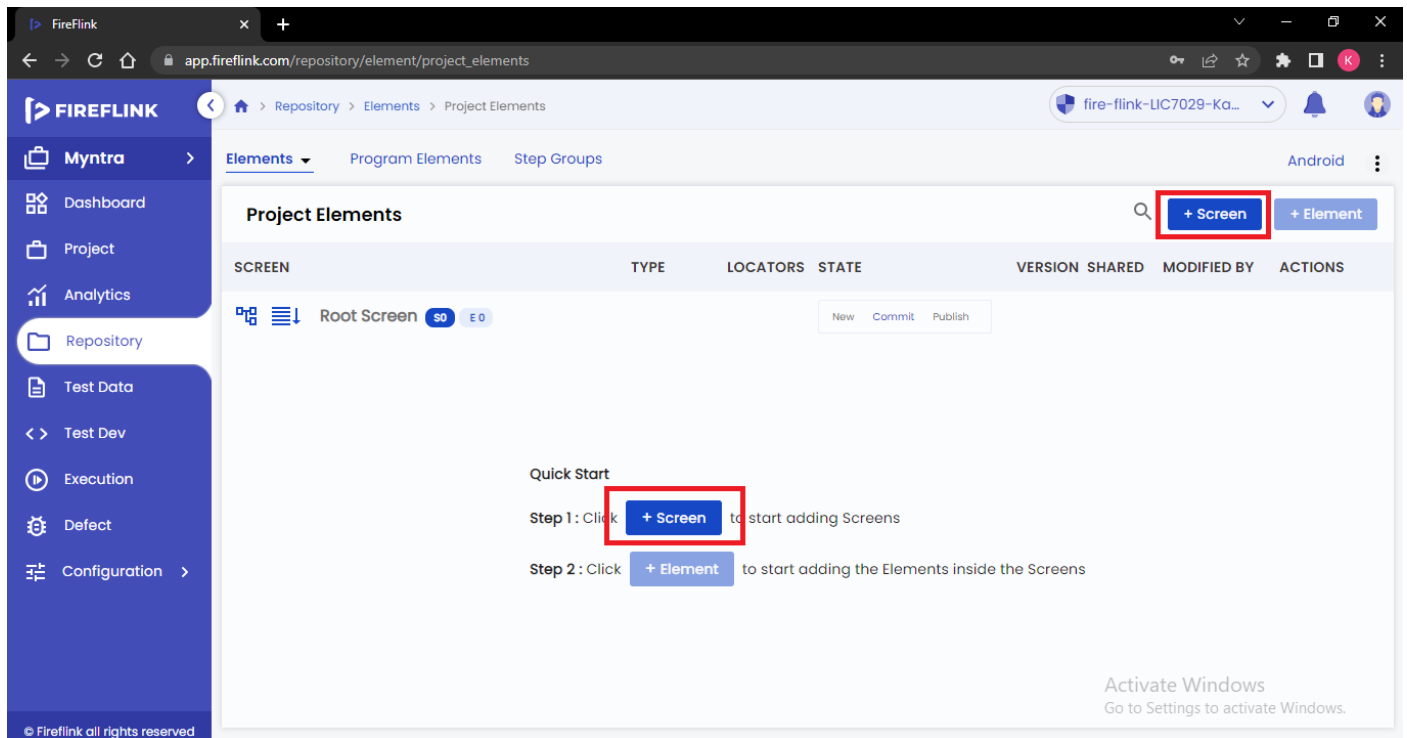
Now Click on Create button.



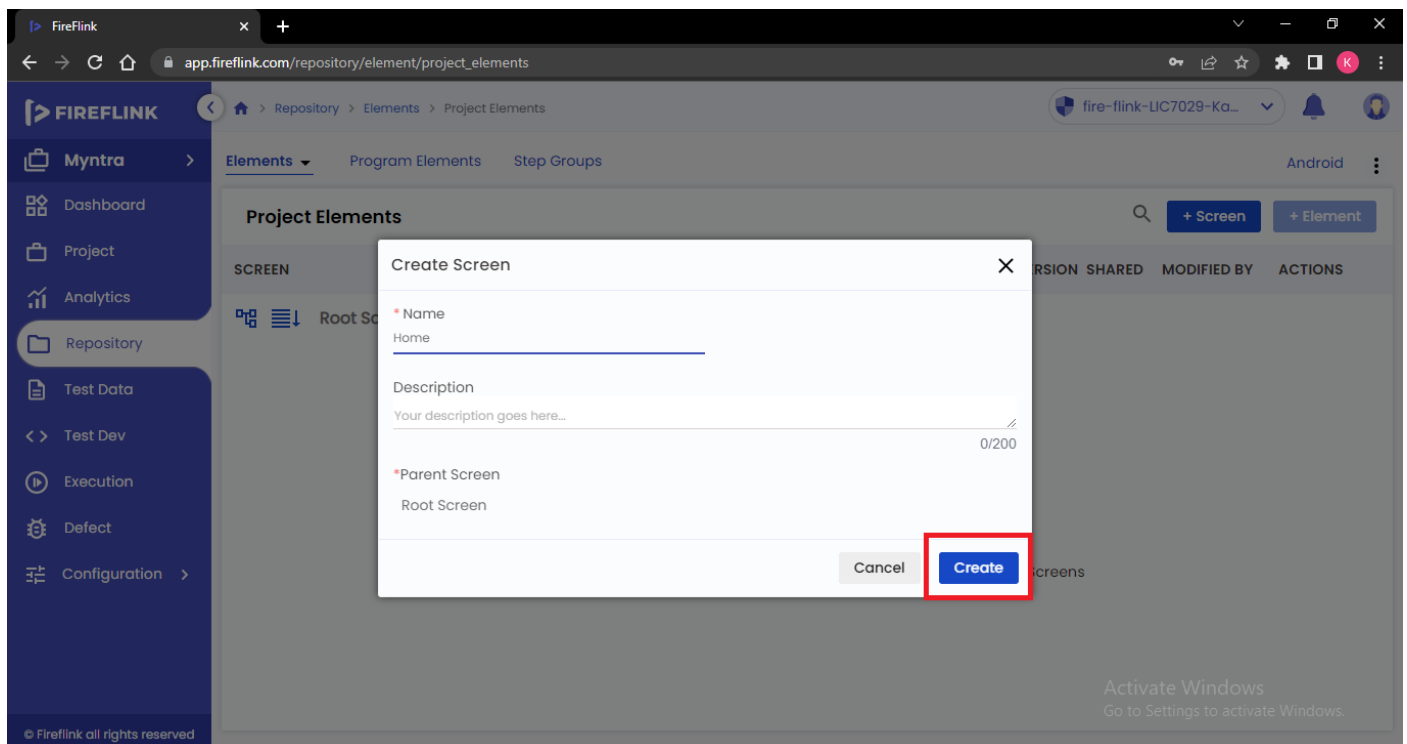
Open the Project. At Individual project level, now click on repository.



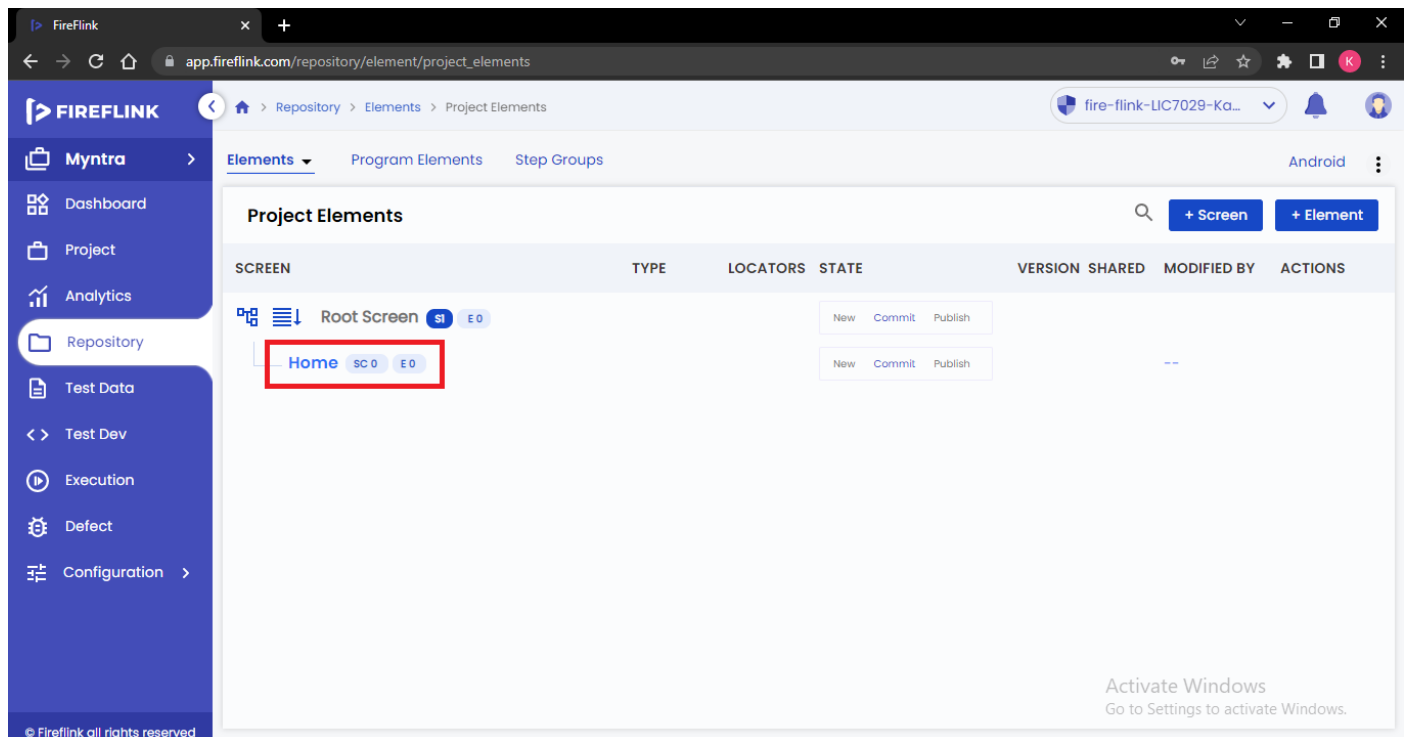
Project elements section of Repository will be displayed. Click on +Screen.



Create screen popup will be displayed. Enter the name and click on create button.



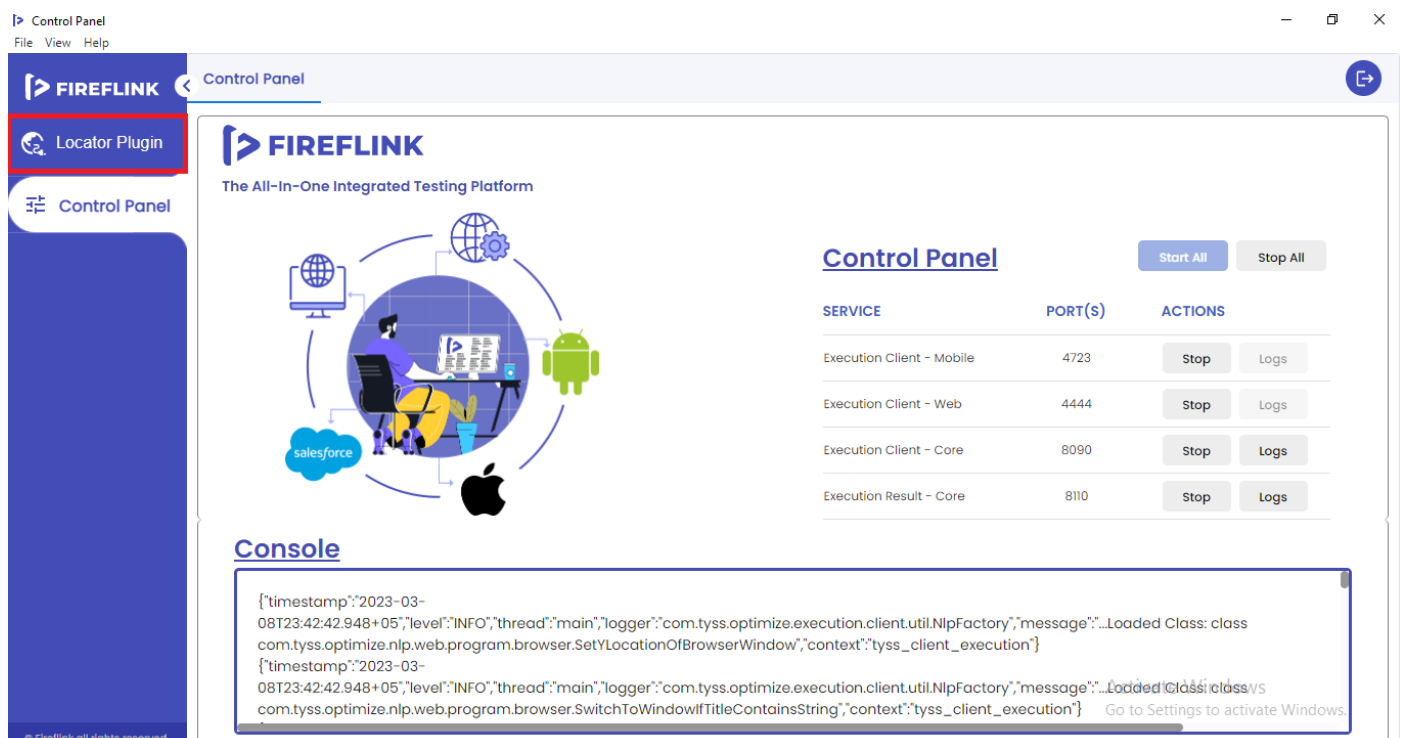
Screen will be created. Add all required screens as per the application, using the same steps.



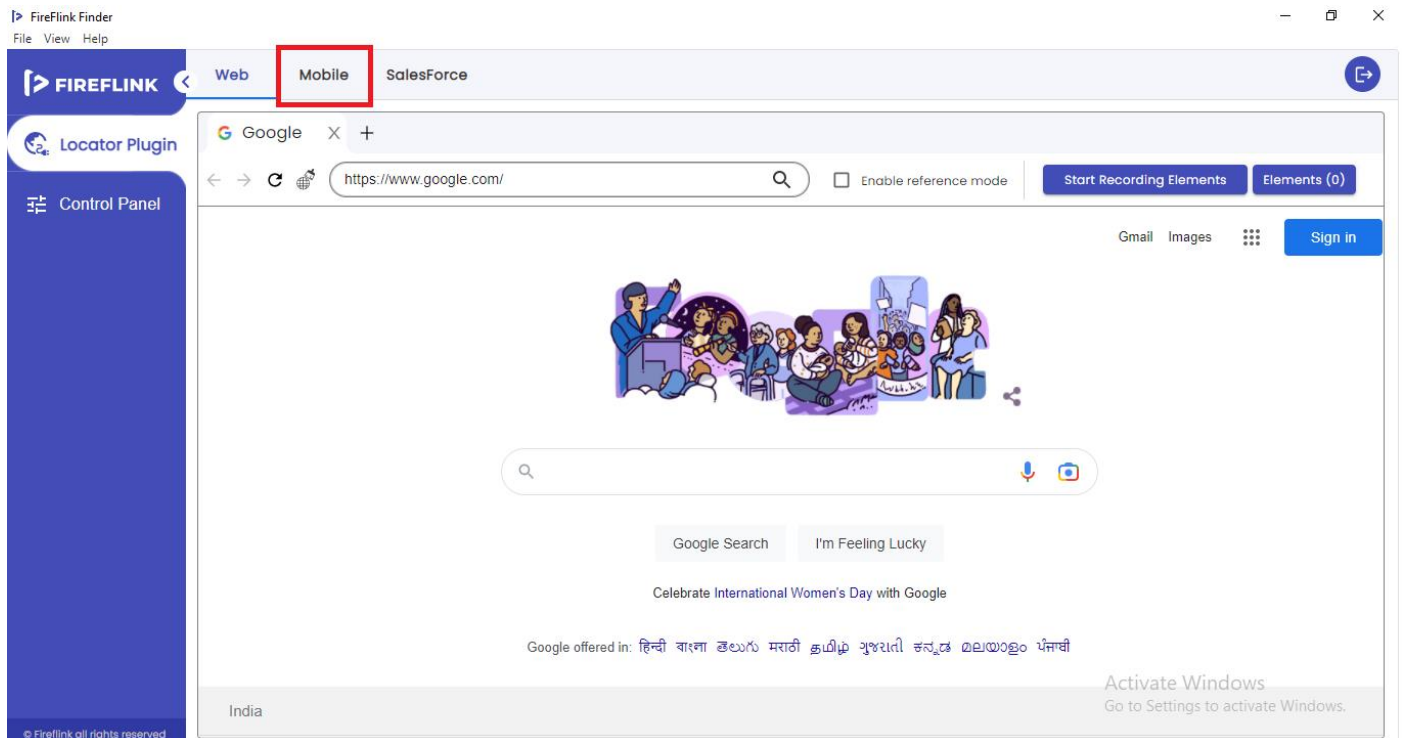
## Steps to capture Mobile app elements using Fire Finder tool:

Note: Connect the mobile device to the computer using USB cable, enable developer mode and enable USB debugging in the mobile device.

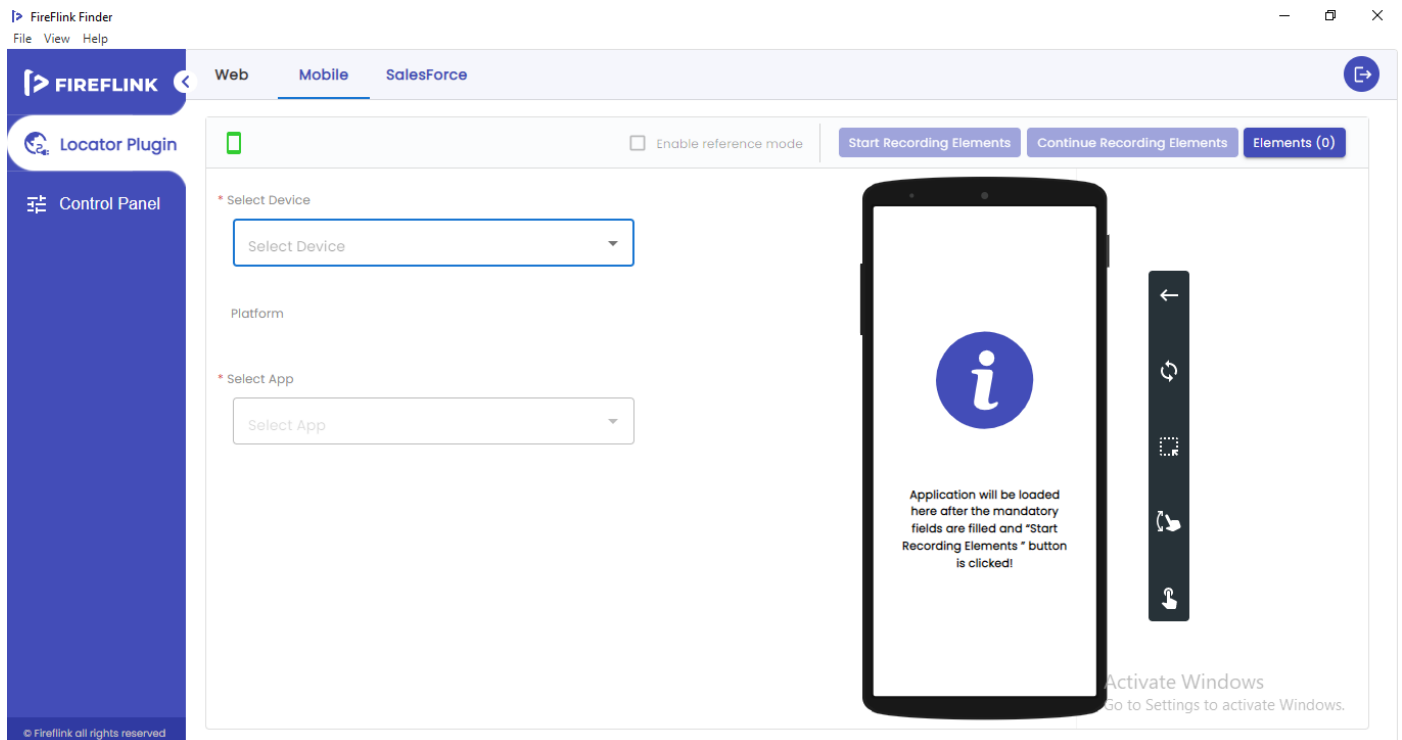
Open FireFlink Client and click on Locator plugin.



## Now Click on Mobile tab

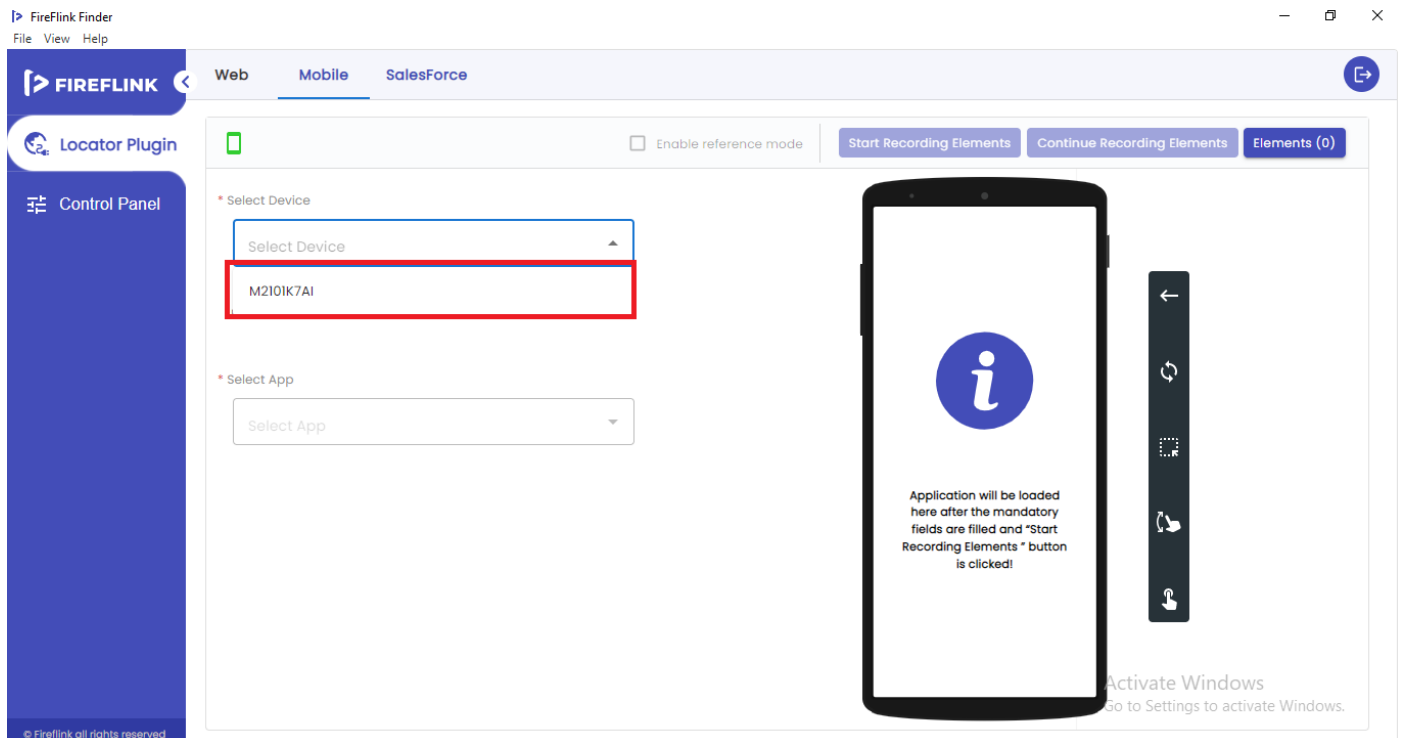


## Mobile Locator plugin will be displayed.

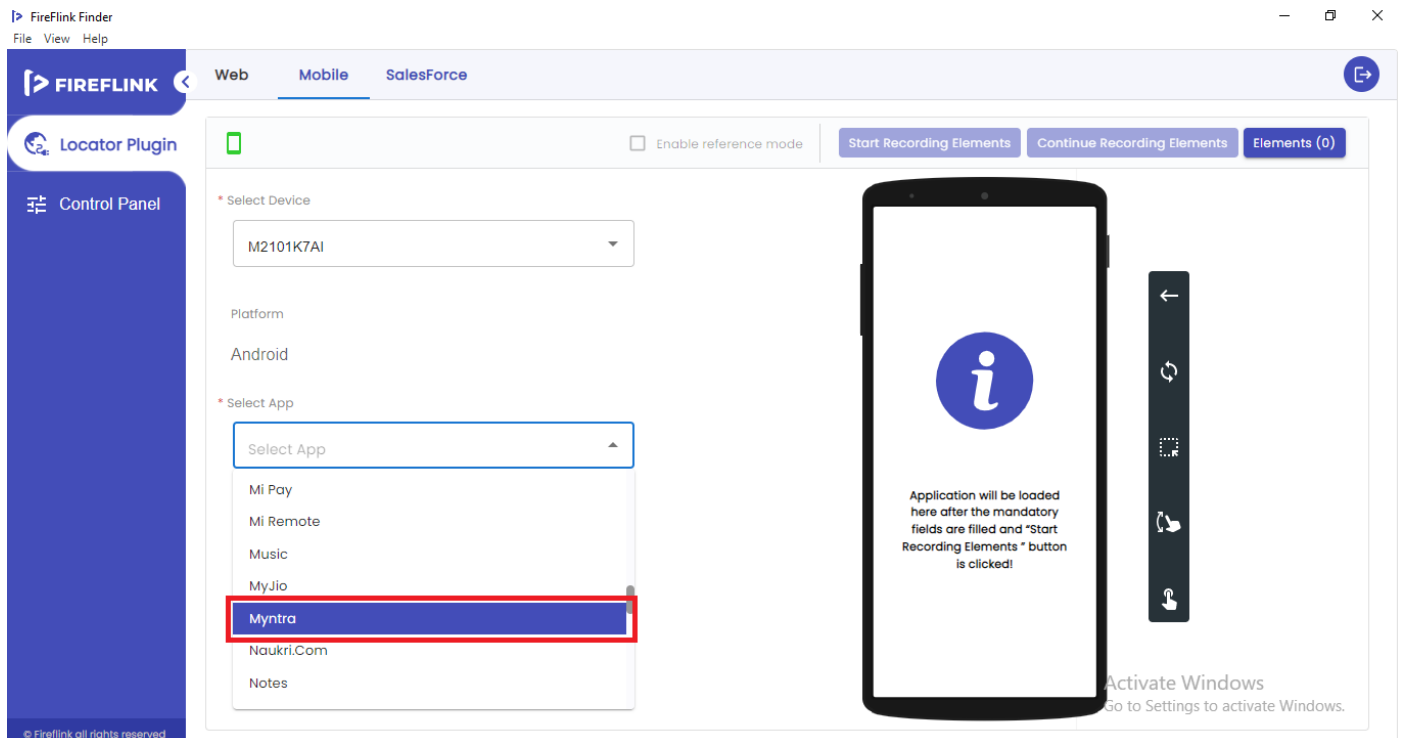




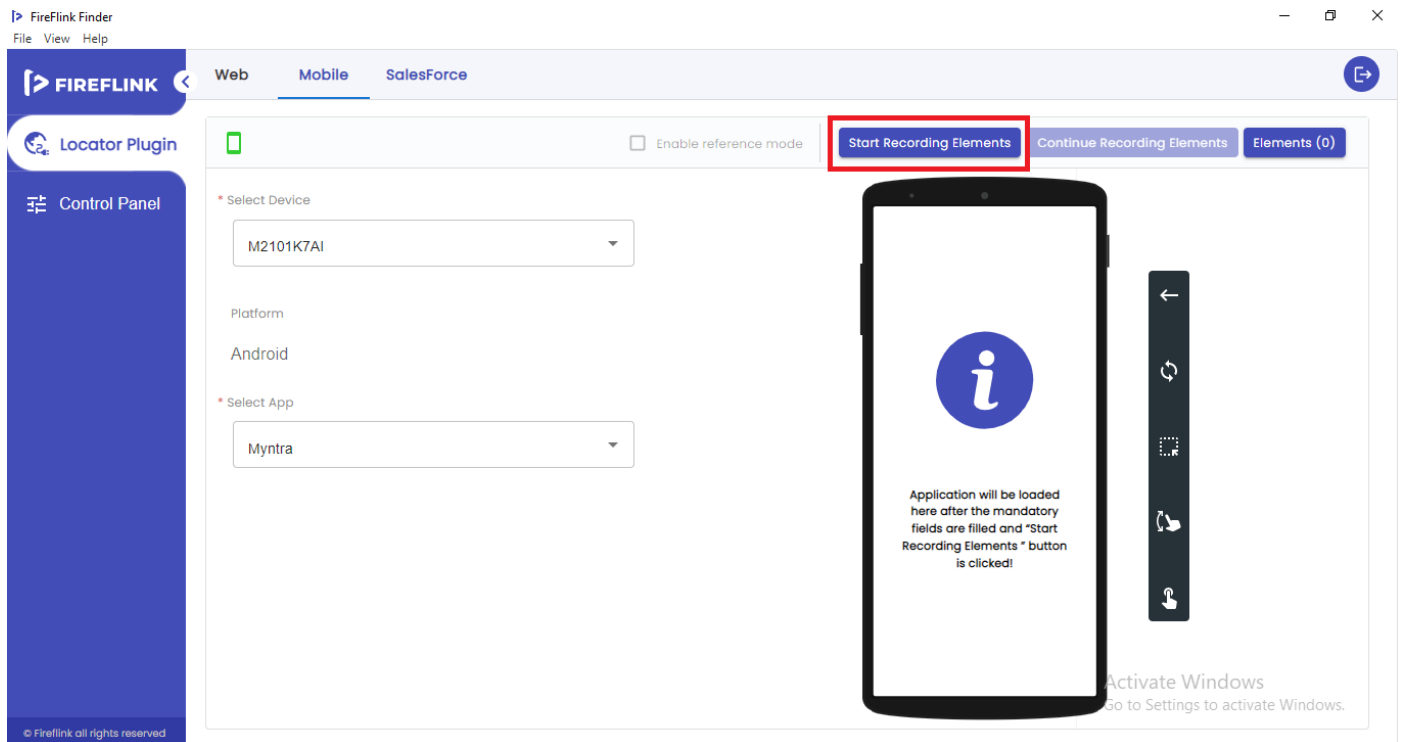
Select your connected device from Select Device dropdown.



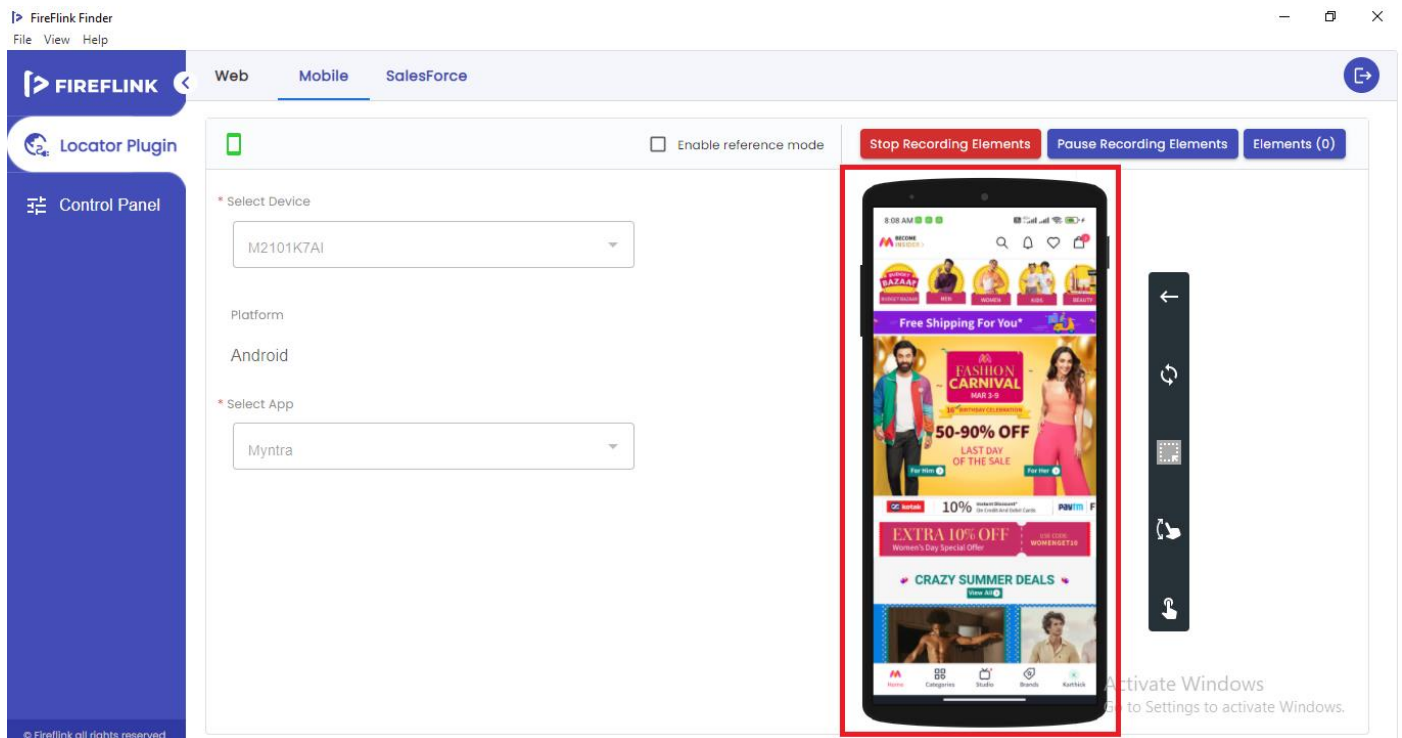
Select the App, from the Select App dropdown



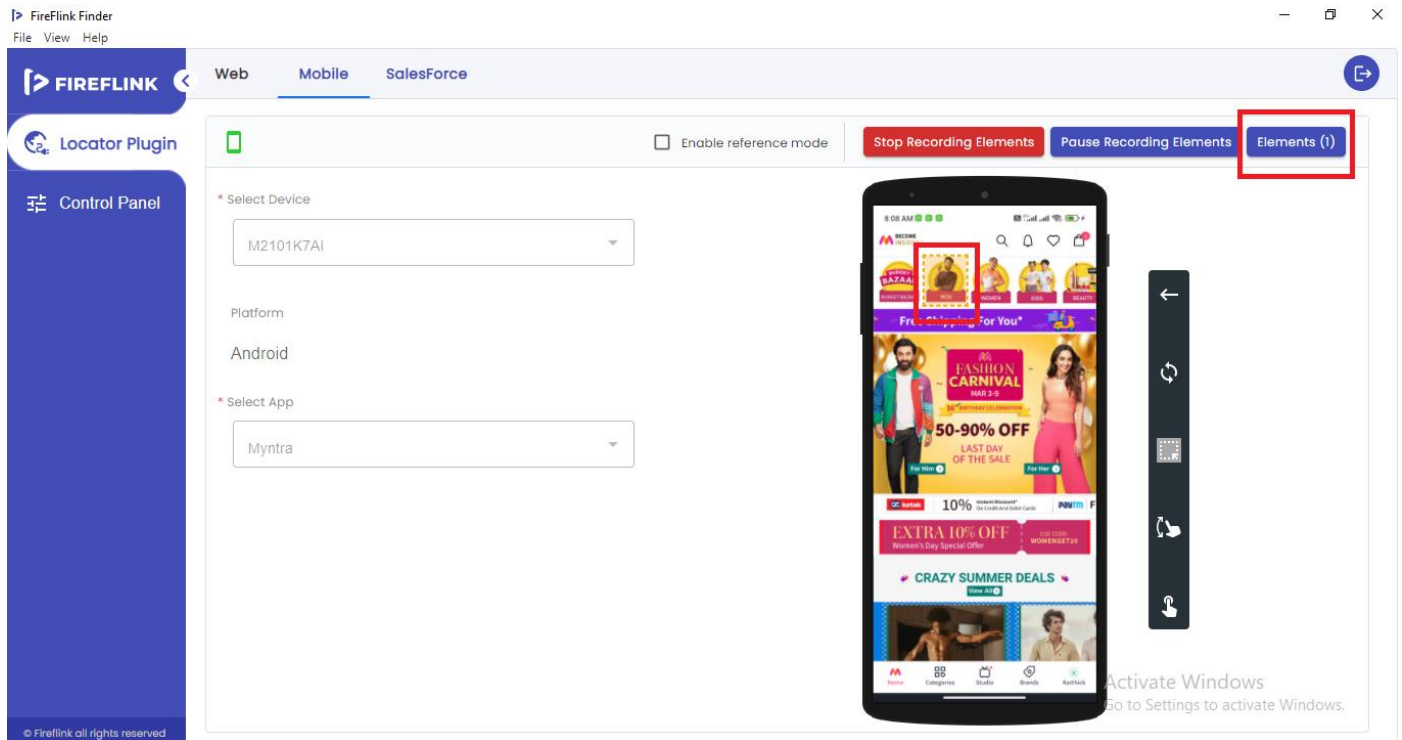
## Now Click on Start Recording Elements button



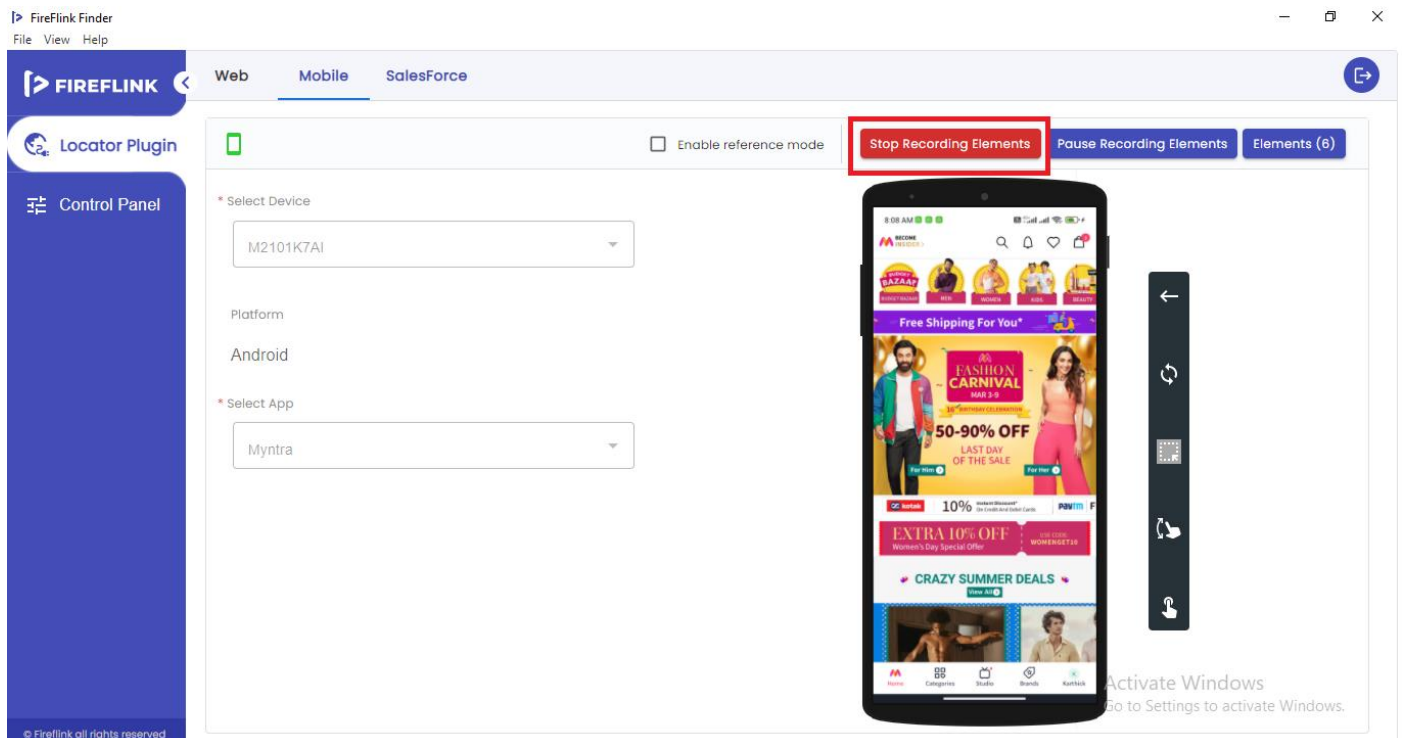
## Selected App will be launched and displayed in the mobile screen.



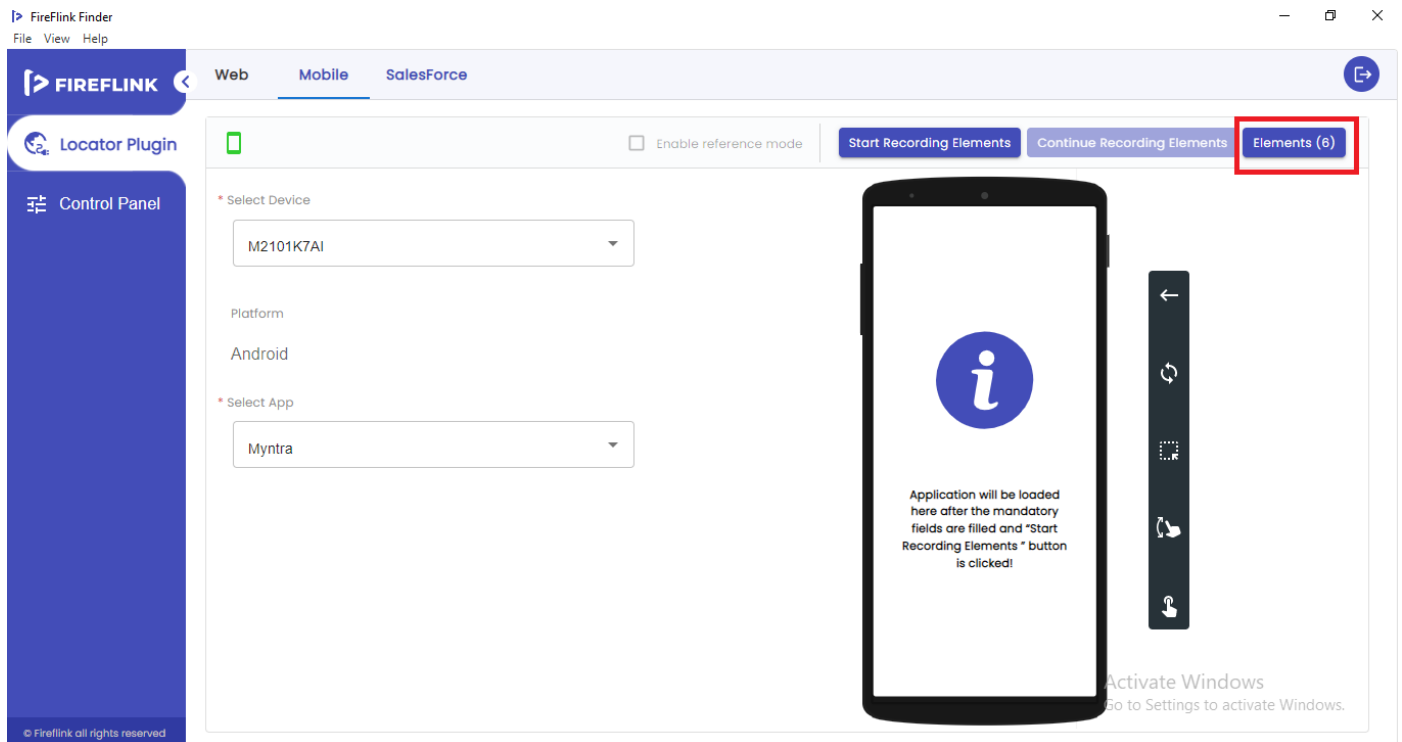
Mouse hover and tap on the elements which needs to be captured, respective elements get captured.



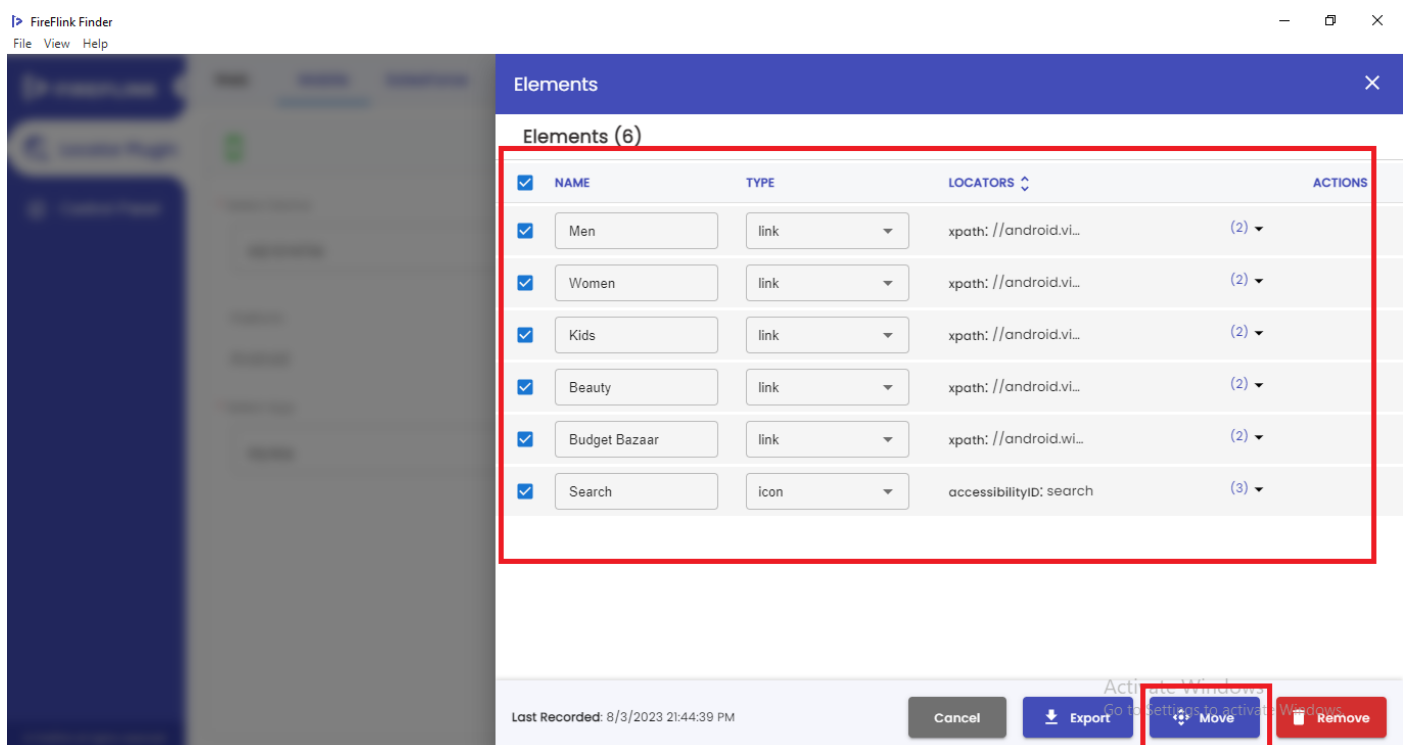
Capture all required elements, just by tapping on them. After capturing all required elements, click on Stop recording button.



## Click on Elements button



All captured elements will be displayed. Select all the elements and click on Move button.



## Select the License form the license Dropdown

FireFlink Finder  
File View Help

Elements

\* License

Select license

- fire-flink-LIC4845-a24d90f6-d09d-402a-98b0-88ba6d5be0
- fire-flink-LIC6003-36eef955-8b03-49b4-ac5f-e749cf249784**
- fire-flink-LIC7029-Karthick
- fire-flink-LIC7542
- fire-flink-LIC8497

\* Project

Select project

Assign Screen

LOCATORS	ACTIONS
xpath: //android.vi...	(2)
xpath: //android.vi...	(2)
xpath: //android.vi...	(2)
xpath: //android.wi...	(2)
accessibilityID: search	(3)

Last Recorded: 8/3/2023 21:44:39 PM

Cancel Back Go to Settings to activate Windows Windows Remove

## Select the Project from the Project Dropdown

FireFlink Finder  
File View Help

Elements

\* License

fire-flink-LIC6003-36eef955-8b03-49b4-ac5f...

\* Project

Select project

- MakeMyTrip
- JioMart\_WebAndMobile
- Amazon
- Myntra**

Elements (6)

SCREEN	NAME	TYPE	ACTIONS
Men	link	(2)	
Women	link	(2)	
Kids	link	(2)	
Beauty	link	(2)	
Budget Bazaar	link	(2)	
Search	icon	(3)	

Last Recorded: 8/3/2023 21:44:39 PM

Cancel Back Go to Settings to activate Windows Windows Remove

Now Click on Assign Screen Button.

The screenshot shows the FireFlink Finder application interface. The 'Elements' panel is open, displaying a list of 6 elements. The 'Assign Screen' button is highlighted with a red box. The interface includes a license field, a project dropdown set to 'Myntra', and a table of elements with columns for SCREEN, NAME, TYPE, LOCATORS, and ACTIONS.

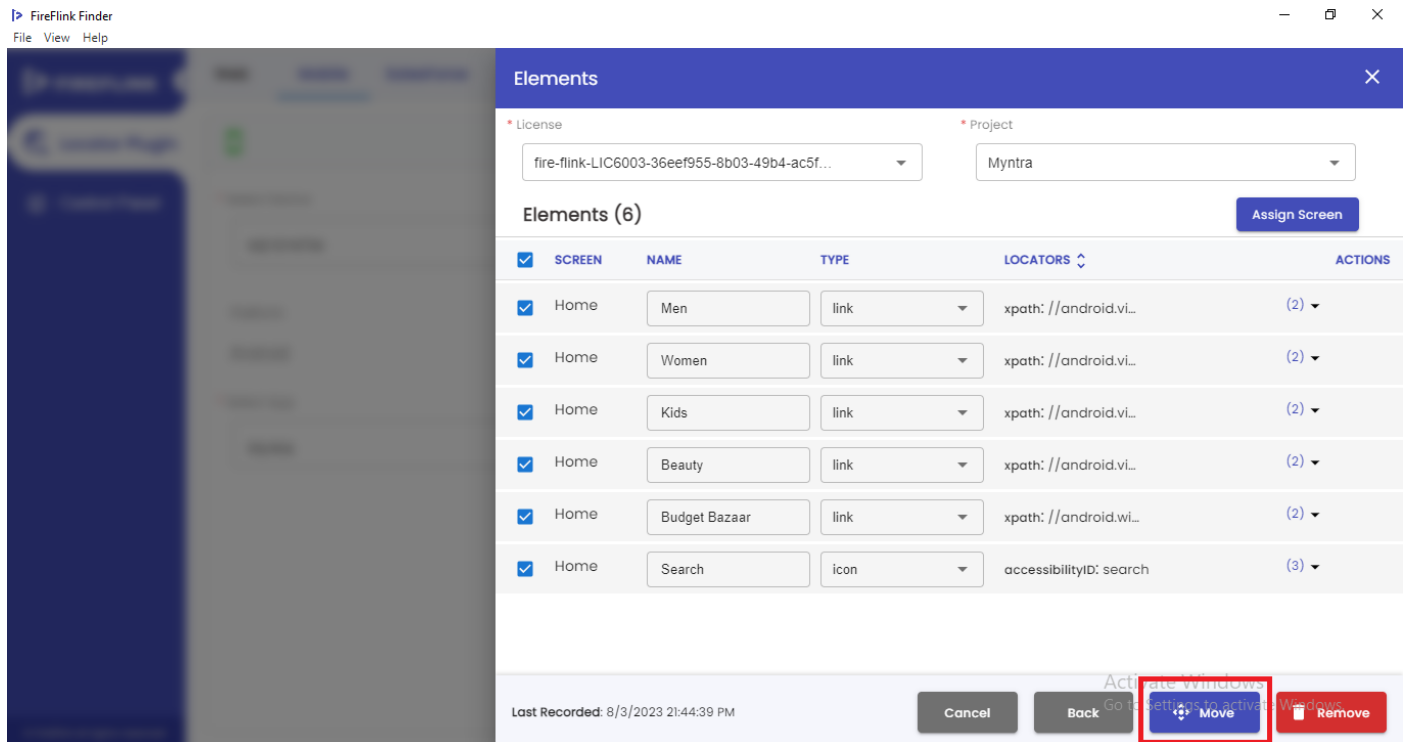
SCREEN	NAME	TYPE	LOCATORS	ACTIONS
<input checked="" type="checkbox"/>	Men	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Women	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Kids	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Beauty	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Budget Bazaar	link	xpath: //android.wi...	(2) ▾
<input checked="" type="checkbox"/>	Search	icon	accessibilityID: search	(3) ▾

Select the respective screen for all the captured elements and click on select button.

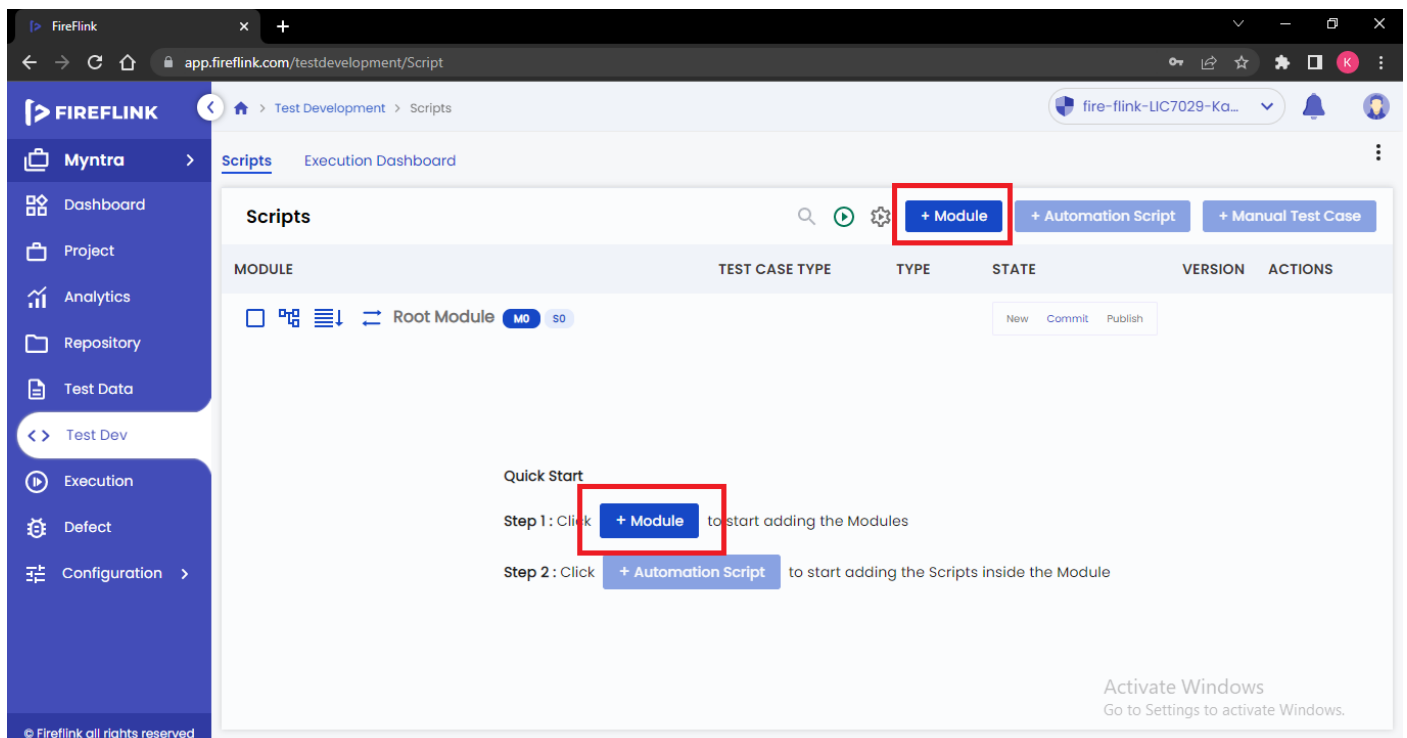
The screenshot shows the FireFlink Finder application interface with the 'Select Screen' dialog box open. The 'Home' option is selected in the dialog, and the 'Select' button is highlighted with a red box. The background shows the same 'Elements' panel as the previous screenshot.

SCREEN	NAME	TYPE	LOCATORS	ACTIONS
<input checked="" type="checkbox"/>	Men	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Women	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Kids	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Beauty	link	xpath: //android.vi...	(2) ▾
<input checked="" type="checkbox"/>	Budget Bazaar	link	xpath: //android.wi...	(2) ▾
<input checked="" type="checkbox"/>	Search	icon	accessibilityID: search	(3) ▾

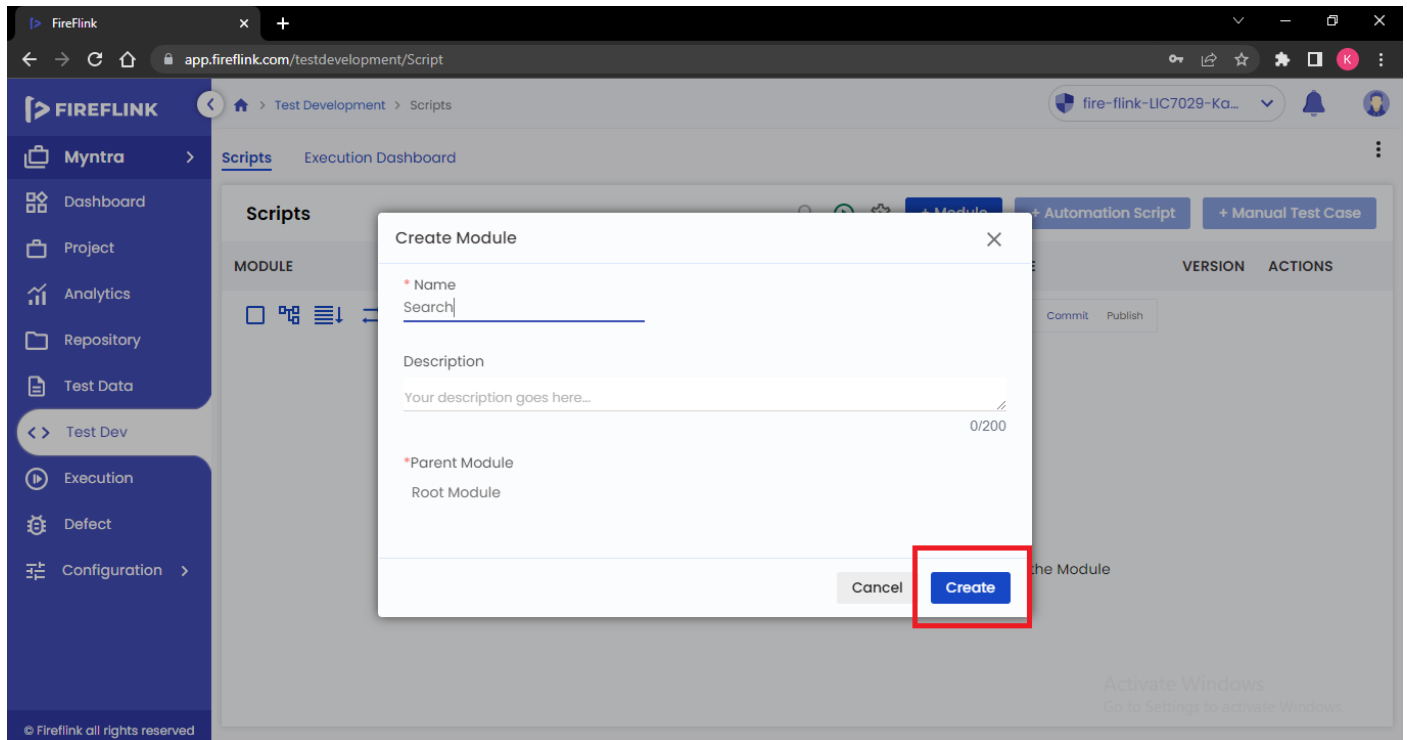
After assigning the screen, click on move button. All elements will be moved to Project elements page of Repository section of the selected project.



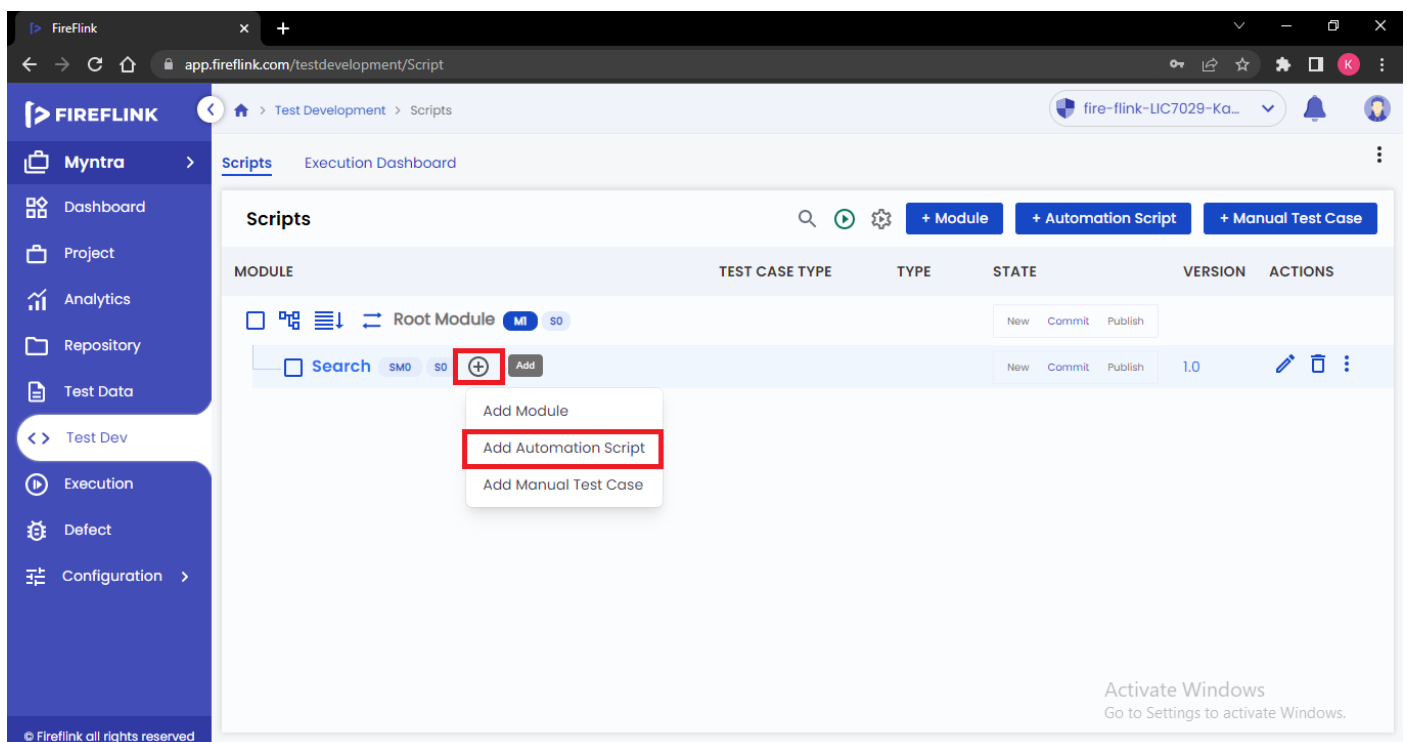
Open the Project. At Individual project level, in Test development section. Click on +Module button.



Create Module popup will be displayed. Enter the Module name and Click on Create button.

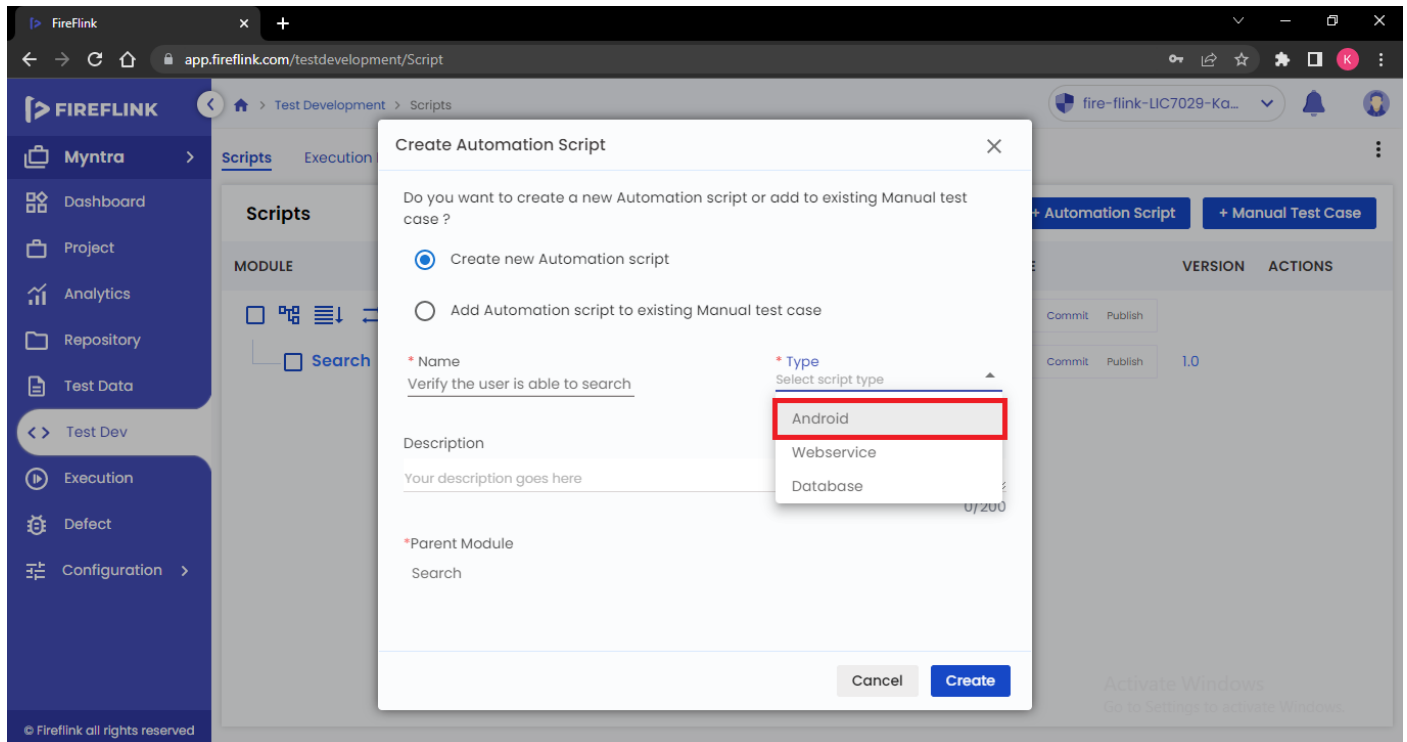


Module will be created. Now click on add button next to module name and click on Add Automation Script.

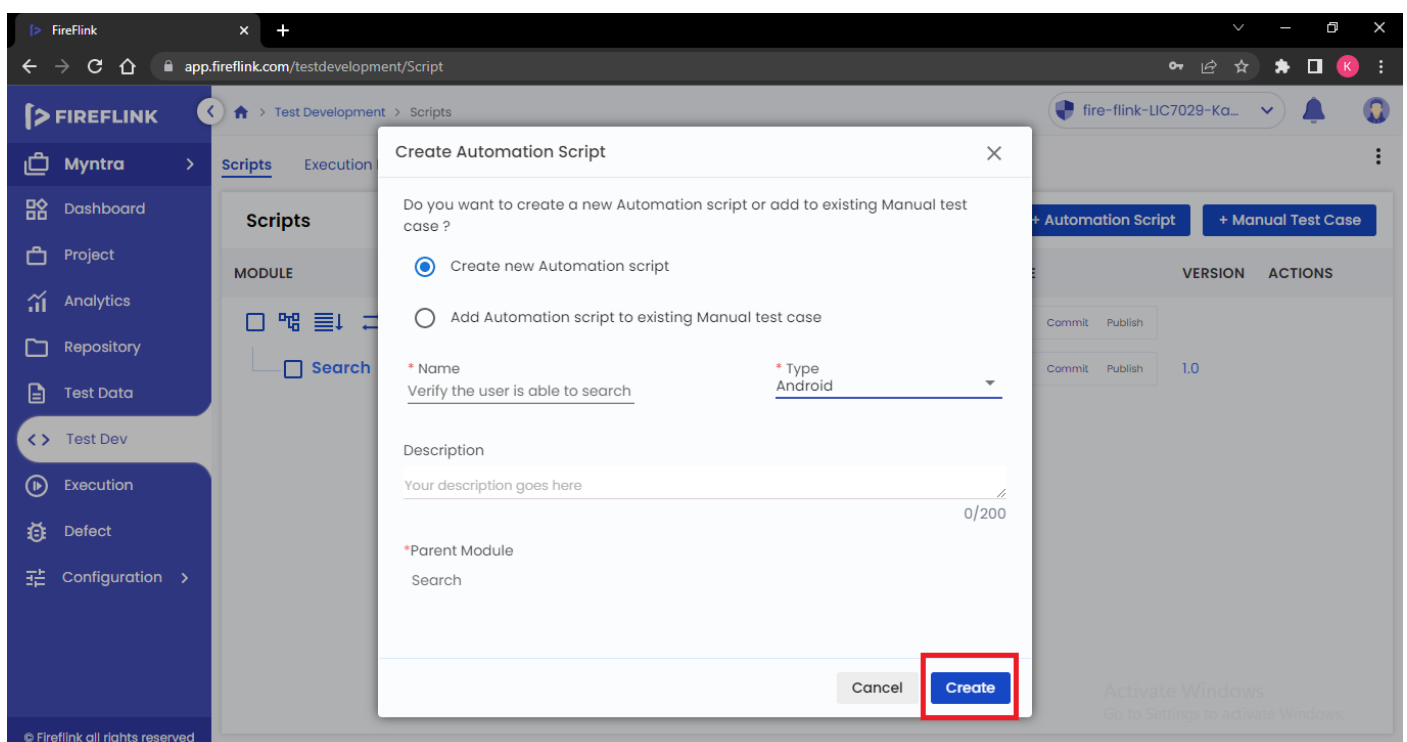




Create Automation script popup will appear. Enter the name of the script and select the type as Android.



Now click on Create button.



Automation script will be created and we will be navigated to script level by clicking on the automation script.

The screenshot shows the FireFlink web interface. On the left is a sidebar with navigation options: Dashboard, Project, Analytics, Repository, Test Data, Test Dev (highlighted with a red box), Execution, Defect, and Configuration. The main area is titled 'Scripts' and contains a table with columns: MODULE, TEST CASE TYPE, TYPE, STATE, VERSION, and ACTIONS. The table lists a 'Root Module' and a 'Shopping Bag' module. Under 'Shopping Bag', there is a script named 'Verify the user is able to add pr...' which is of type 'Automation' and platform 'Android'. This script is highlighted with a red box. At the top right of the main area, there are buttons for '+ Module', '+ Automation Script', and '+ Manual Test Case'. An 'Activate Windows' watermark is visible at the bottom right.

MODULE	TEST CASE TYPE	TYPE	STATE	VERSION	ACTIONS
Root Module			New Commit Publish		
Shopping Bag			New Commit Publish	1.0	
Verify the user is able to add pr...	Automation	Android	New Commit Publish	1.0	

Start adding steps/NLP's as per the manual test case.

The screenshot shows the 'Automation Steps' page in FireFlink. The breadcrumb trail is 'Test Development > Scripts > Shopping Bag > Verify the user is a... > Steps'. The page title is 'Verify the user is able to add product to shopping bag'. Below the title is a table with columns: DESCRIPTION, RESULT, STATUS, and ACTIONS. The table contains a list of 10 steps, which are highlighted with a red box. The steps are: 1. DesiredCapability create instance of DesiredCapabili..., 2. DesiredCapability set capability key: appPackage valu..., 3. DesiredCapability set capability key: appActivity value..., 4. DesiredCapability set capability key: platformName va..., 5. DesiredCapability set capability key: deviceName valu..., 6. DesiredCapability set capability key: noReset value: bo..., 7. DesiredCapability set capability key: autoGrantPermiss..., 8. App open, 9. Tap on search icon, and 10. Enter searchText into Search for brands & products tex... At the top right of the main area, there are buttons for '+ Manual Test Case' and '+ Step'. An 'Activate Windows' watermark is visible at the bottom right.

DESCRIPTION	RESULT	STATUS	ACTIONS
Steps			
1. DesiredCapability create instance of DesiredCapabili...			
2. DesiredCapability set capability key: appPackage valu...			
3. DesiredCapability set capability key: appActivity value...			
4. DesiredCapability set capability key: platformName va...			
5. DesiredCapability set capability key: deviceName valu...			
6. DesiredCapability set capability key: noReset value: bo...			
7. DesiredCapability set capability key: autoGrantPermiss...			
8. App open			
9. Tap on search icon			
10. Enter searchText into Search for brands & products tex...			

After adding all the required steps as per the manual test case, click on Run button for executing the script.

The screenshot shows the Firelink web application interface. The sidebar on the left contains navigation options: Dashboard, Project, Analytics, Repository, Test Data, Test Dev, Execution, Defect, and Configuration. The main area displays a test script titled "Verify the user is able to add product to shopping bag". The script is organized into a table with columns: DESCRIPTION, RESULT, STATUS, and ACTIONS. The table contains 10 steps for the test script. A red box highlights the "Run" button (a green circle with a play icon) in the top right corner of the table.

DESCRIPTION	RESULT	STATUS	ACTIONS
<input type="checkbox"/> Steps ▾			
1. DesiredCapability create instance of DesiredCapabiliti...			
2. DesiredCapability set capability key: appPackage valu...			
3. DesiredCapability set capability key: appActivity value...			
4. DesiredCapability set capability key: platformName va...			
5. DesiredCapability set capability key: deviceName valu...			
6. DesiredCapability set capability key: noReset value: bo...			
7. DesiredCapability set capability key: autoGrantPermiss...			
8. App open			
9. Tap on search icon			
10. Enter searchText into Search for brands & products tex...			

Activate Windows  
Go to Settings to activate Windows.