

https://www.nimbal.io/

**2 |** P a g e

# **INDEX**

Sr No.	Content Description	Page No.
1.	INTRODUCTION	3
2.	PRE-REQUISITE	3
3.	CREATING TEST CASES	3
4.	CODES AND TEST CASES	3 -8
	<ul> <li>Run 1: Running Happy Path Scenario IDE Approach</li> <li>Run 2: Running Negative Path Scenario IDE Approach</li> <li>Run 3: Running the whole feature</li> <li>Tags Used+++</li> </ul>	
5.	RUNNING TESTS USING TAGS	8-13
6.	REPORTS	13-17
	Reviewing Reports	
7.	REFERENCES	17

## **INTRODUCTION**

Testing is the method of evaluating and verifying that a software program product or utility does what it is meant to do. The advantages of testing is finding and getting rid of bugs, decreasing development fees, and enhancing performance.

Involvement of testers in requirement critiques and consumer tale refinement- Involving testers at some stage in the requirement segment guarantees identity of a number of the requirement defects even earlier than their implementation. It considerably reduces the solving cost. Also, the tester profits considerable task perception at this stage, it turns to facilitate him in the execution segment of the task.

## PRE-REQUISITE

- 1. Knowledge of <u>XPath</u>
- 2. Basics of <u>Testing concepts</u>
- 3. Access to Nimbal Web IDE i.e., its URL, such as <u>nimbal-webide.getskills.co.nz</u>
- 4. Your browser should have the following programs installed
  - 1. <u>Selector Hub</u> : it is a helper browser extension, which gives us prebuilt XPath.
  - 2. <u>ChroPath</u>: it is an alternative extension for XPath.

## **CREATING TEST CASES**

Let us create a new feature page and test the following cases using some scenarios/cases

Follow the steps below to create a new file

- 1. Click on the file's D icon.
- 2. Navigate to src > java > feature.
- 3. A dialog box will appear and will ask for the name of the new file enter any name with login. feature extension

For example- login. feature for login related feature test scenarios as shown in the below screenshot



#### Figure 1: Path to the file

## **CODES AND TEST CASES**

### Setting up the project configuration

We need to add some properties in the following files in Web IDE

- 1. config.dev.properties : Located at /home/project/src/test/resources/env/config.dev.properties Add the following values as shown below in Figure 4
  - **a.** app.gmail = gmail.com

instead of gmail, you can replace it with any constant value of your concern. For example, app.website = <u>www.website.com</u>

b. web.browser=chromeheadless

Specified which browser to the user for running the test. other values possible are firefox

-			
Go Deb	oug Terr	ninal Help	
web.f	eature	config.dev.properties ×	locators.json
	1 ##	*************************	******************
	2		
	3 ар	p.gmail=https://gmail.com	
	4	· · · · · · · · · · · · · · · · · · ·	
	6 #0	prions - chrome , chromeld	cal, chromeheadle
	7 we	b.browser=chromeheadless	
	8 10	cal.driver.chrome=	
	9	b.grid=false	
	10 ur	l.grid=	
	11		



2. locators.json : Located at /home/project/src/test/resources/locators.json .We will use this file to add XPath key pair values, while key will be user understandable keyword and value will be an XPath, which is used to locate an HTML element within a webpage. For example, XPath for

inputting email id in gmail.com page will be **//input[@id='identifierId']** as shown below in Figure 2.

ſ	Go Sig to continu	ogle yn in ue to Gmail		٦		
		Emoji	Win+Period			
Fo	rgot email?	Undo	Ctrl+Z			_
		Redo	Ctrl+Shift+Z			
No	ot your computer? Use	Cut	Ctrl+X			_
Le	arn more	Сору	Ctrl+C			
		Paste	Ctrl+V			
Cr	eate account	Paste as plain text Select all	Ctrl+Shift+V Ctrl+A			/4e
		Spell check		•		//input[
		Writing Direction		•	XPath Training	s
		SelectorsHub		•	SelectorsHub Certificate	Path
English (Ur	nited Kingdom) 🔻	Inspect			Copy Rel cssSelector	5
				_	Conv IC noth	att

Figure 3: illustrates how to choose the XPath through Selectors Hub

Therefore, the locators.json file will look something like the below after making changes

	File Edit Selection View Go	Debug Terminal	Help			
<b>F</b>	EXPLORER: PROJECT	¢ 🗐 …	windows.feature	web.feature	locators.json ×	TestRunner.java
	<ul> <li>Coounty cooperate</li> <li>Instrumer.java</li> <li>Instrumer.jav</li></ul>	M M M	1 { 2   "1, 3   4   } 5 }	ogin": { "email_inp":	"//input[@id='	identifierId']"

Figure 4: locator.json file after adding XPath for Gmail input location from Gmail.com

 TestRunner.java: Located at /home/project/src/test/java/TestRunner.java it is a file used to indicate which tests need to be run using tags. A tag can be placed on a scenario/test or a feature. It usually starts "@" keyword.

For example,

 Figure 6 depicts the functioning of the TestRunner.java file and the tag provided in the tags section written at line number 14 as tags={"@gmailLoginNegativePath"}

Later we will replace this tag to run the tests we want to execute.

2. If tags are empty then the TestRunner.java file will run all the feature files present in the project as shown below

tags={""}



Figure 5: TestRunner.java file

### Ways to run the test.feature file

1. Right-click on the TestRunner.java file and click Run. The results will be shown in the terminal.

) >	EXPLORER      PROJECT      Ø performance.feature      ø security.feature		TestRunner.java × 1 import 2 import 3 import 4 import	<pre>config.dev.properties web.feature cucumber.api.CucumberOptions; cucumber.api.junit.Cucumber; org.junit.AfterClass; org.iunit.runner.RunWith:</pre>	Go to Definition Go to Implementations Go to References	Ctrl+F11 Ctrl+F12 Shift+F12	<b>ا</b>
	<ul> <li>SignUp.feature</li> <li>web_nimbal.feature</li> <li>web.feature</li> </ul>		5 6 @ <mark>R</mark> unWi1 7 @Cucuml	th(Cucumber.class) berOptions()	Go to Symbol Go to Type Definition Peek	Ctri+Snitt+O	•
-	<ul> <li>Ø windows.feature</li> <li>&gt; b steps</li> </ul>			<pre>glue = {"co.nz.getskills.auto.appsteps"}, drvRun = false.</pre>	Change All Occurrences Format Document	Ctrl+F2 Alt+Shift+F	
i	∉ TestRunner.java			<pre>plugin = { "json:target/json-cucumber-reports/cuke</pre>	Format Selection	Ctrl+K, Ctrl+F	
Ň	<ul> <li>✓ ■ resources</li> <li>✓ ■ config</li> <li>↓↓ config.properties</li> </ul>		12 13 14 15	"junit:target/junit-cucumber-reports/cukej "html:target/junit-cucumber-reports/cukeju tags = {"@signHappy1"}	Refactor Rename Symbol Source Action	Ctrl+Shift+R F2	
	✓ ■ env ## config.dev.properties	M M	16 ) 17		Redo	Ctrl+Shift+Z	
	> 🖿 getskills		Run Tes	t   Debug Test	Undo	Ctrl+Z	
	E data.json E locators.json		Terminal 0 × Debu	Ig Console	History Toggle Blame Annotations	Alt+B	D
	↔ log4j2.xml		root@ip-10-121-8	is-as:/home/project# cd 85-83:/home# git clone https://aatisha8@bitbucket.c bise-auto'	Cut	Ctrl+X	
	> allsafe-reports		Password for 'ht remote: Enumerat	tips://attisha8@bitbucket.org': ting objects: 319, done.	Paste	CultC	
	Generated-report     E attachments		remote: Counting	objects: 100% (319/319), done.	Run		

Figure 6:Running a test case

2. In the terminal, enter **mvn install** command and it will show the results in the terminal.



Figure 7:TestRunner.java

### Run 1: Running Happy Path Scenario

7 | Page

Following is the code is written for the Negative path as shown in code snippet 1. Happy path test is a well-defined test case using known input, which executes without exception and produces an expected output.

#### Feature: Login in the app

@gmailLoginHappyPath Scenario: Login in the Gmail- Happy path Given I open gmail.com And I fill input email\_inp with abcd123 And I fill input password\_inp with 123@abcd And I click element login\_btn And I can see the text "Compose"

Code Snippet 1: Happy Path for Gmail Login





#### **Explanation of the Code**

- 1. Title of the Feature, always start with Feature Keyword
- 2. The keyword of the whole case (No need to write the complete code for testing just enter the keyword)
- 3. Description of the code
- 4. The user opens the URL gmail.com
- 5. Then users enter the data as abcd123 in the email field (fill input is a keyword for giving input)
- 6. Then users enter the data as 123@abcd in the password field (fill input is a keyword for giving input)
- 7. Then by clicking the login\_btn (Button named Login)
- 8. User login successful can view the compose button

## Run 2: Running Negative Path Scenario

Following is the code is written for the Negative path as shown in code snippet 2. Negative testing ensures that your application can gracefully handle invalid input or unexpected user behavior.

@gmailLoginFeature
Feature: Login in the app
Agmail aginNegativeDath
wgmaileoginnegaciverath
Scenario: Login in the Gmail- Negative path
Given I open gmail.com
And I fill input email_inp with abcd123
And I fill input password_inp with 123@abcd
And I click element login_btn
And I check the message "Wrong password. Try again or click 'Forgot password' to
reset it."



	File Edit Selection View Go Run Terminal	Help	
۵ı	EXPLORER ····	login.feature ×	æ
Q	> OFEN EDITORS > PROJECT > 12 destures > 12 destures	src > test > java > features > demo > ♥ login.feature 1 @gmailLoginFeature 2 Feature: Login in the app 3	
¢ ₽ ₽	a drind feature     a pi.feature     a pi.feature     d b.feature     d b.feature     f filesystem-ttp.feature     f filesystem-ttp.feature     g gwt.feature     g gwt.feature     w gwt.feature     w jOS Web feature	<ul> <li>@gmailLoginNegativePath</li> <li>Scenario: Login in the gmail- Negative path</li> <li>Given I open gmail.com</li> <li>And I fill input email_inp with abcd123</li> <li>And I fill input password_inp with 123@abcd</li> <li>And I click element login_btn</li> <li>And I check the message "Wrong password. Try again or click 'Forgot password' to reset it."</li> </ul>	
	Ø iOS.feature           Ø login.feature         U		
	performance.feature     security.feature     web_nimbal.feature		
	♥ web.feature M ♥ windows.feature > Ҽ steps ●		

Figure 9: Reference image in Web IDE for Negative path

**Explanation of the Code** 

1. The keyword of the whole case (No need to write the complete code for testing just enter the keyword)

- 2. Title
- 3. The keyword of case 2(No need to write the complete code for testing just enter the keyword)
- 4. Description of the code
- 5. The user opens the URL gmail.com
- 6. Then users enter the data as abcd123 in the email field (fill input is a keyword for giving input)
- 7. Then users enter the data as 123@abcd in the password field (fill input is a keyword for giving input)

8. Then by clicking the login\_btn (Button named Login) User login successful, can view the message "Wrong password. Try again or click 'Forgot password' to reset it."

### Run 3: Running the whole feature

Following is the complete code for our feature. Which can be executed by the adding tag **@gmailLoginFeature** in the **TestRunner.java** file.

@gmailLoginFeature

Feature: Login in the app

@gmailLoginHappyPath Scenario: Login in the Gmail- Happy path Given I open gmail.com And I fill input email\_inp with abcd123 And I fill the input password with 123@abcd And I click element login\_btn And I can see the text "Compose"

```
@gmailLoginNegativePath
Scenario: Login in the Gmail- Negative path
Given I open gmail.com
And I fill input email_inp with abcd123
And I fill input password_inp with 123@abcd
And I click element login_btn
And I check the message "Wrong password. Try again or click 'Forgot password' to
reset it."
```

Code Snippet 3: complete feature file code

```
11 | Page
```



Figure 10: Reference image in Web IDE for complete code

## **RUNNING TEST USING TAGS**

There are two components to understand how to run tests (features/scenarios) by tags

#### 1. Tags:

Various tags are used in feature files. We can identify a tag as a group of features or a group of scenarios or a group of both features and tags. Depending on a tester.

For example, in login.feature file above files we have used the following tags

- @gmailLoginFeature To execute complete feature
- @gmailLoginHappyPath To execute the happy path scenario
- @gmailLoginNegativePath To execute the negative path scenario

#### 2. TestRunner.java file:

It is used for running the test with the help of tags. Every feature and scenario has a tag associated with it. We will insert this tag in the TestRunner.java file and run the program to get the desired outcome of the test using reports.

Following is the way to utilize the TestRunner.java file to run the tests.

```
12 | Page
```

1. Click on the file's icon. Navigate to src > java > TestRunner.java

```
TestRunner.java ×
          import cucumber.api.CucumberOptions;
         import cucumber.api.junit.Cucumber;
         import org.junit.AfterClass;
         import org.junit.runner.RunWith;
         @RunWith(Cucumber.class)
         @CucumberOptions(
                  features = {"src/"},
                  glue = {"co.nz.getskills.auto.appsteps"},
                  dryRun = false,
                  plugin = { "json:target/json-cucumber-reports/cukejson.json",
                          "junit:target/junit-cucumber-reports/cukejunit.xml",
                          "html:target/junit-cucumber-reports/cukejunit.html"},
                  tags = {"@AddYourTagHere"}
          )
         public class TestRunner
          {
             @AfterClass
             public static void tearDown()
             }
          }
```

#### Figure 11: Test case Runner

2. In any pre-created tag can be used for the test by taking any Tag from an existing code then writing it in the place of **"@AddYourTagHere"** 

For example tags = {"@gmailLoginFeature"}

Here the tag @gmailLoginFeature is used.



Figure 13: Generated Report -JSON

## **REPORTS**

The report is generated each time when the user runs a tag using the maven command or by running right click on the TestRunner.java file.

1. The report generated can be seen under the targets folder that is located at /home/project/target.

```
14 | Page
```

### 2. The json reports are found at /home/project/target/json-cucumber-reports.

<b>N</b>	EXPLORER	TestRunner.java ×	
م م		1 import o 2 import o 3 import o 4 import o	cucumber.api.CucumberOptions; cucumber.api.junit.Cucumber; org.junit.AfterClass; org.junit.runner.RunWith;
× **	<ul> <li>&gt; b getskills</li> <li>&gt; b platformJar</li> <li>idata.json</li> <li>M</li> <li>iocators.json</li> <li>M</li> <li>olg4j2.xml</li> <li>&gt; b failsafe-reports</li> <li>&gt; b generated-report</li> <li>&gt; b generated-test-sources</li> <li>&gt; ison-cucumber-reports</li> <li>&gt; b junit-cucumber-reports</li> <li>&gt; maven-archiver</li> <li>&gt; maven-status</li> </ul>	5 6 @RunWitt 7 @Cucumbo 8 9 10 11 12 13 14 15 16 ) 17 Run Test 18 public o	<pre>h(Cucumber.class) erOptions( features = {"src/"}, glue = {"co.nz.getskills.auto.appsteps"}, dryRun = false, plugin = { "json:target/json-cucumber-reports/cukejson.json",                             "junit:target/junit-cucumber-reports/cukejunit.xml",                                "html:target/junit-cucumber-reports/cukejunit.html"}, tags = {"@gmailLogin"} </pre>
\$03 203	<ul> <li>TreeReportUpload</li> <li>getskills-automation-1.9.4.jar</li> <li>TreeReportUpload</li> <li>.gitignore</li> <li>auto-platform-1.9.3.pom</li> <li>core.20211124.135143.390.0001.dmp</li> <li>core.20211124.142024.11530.0001.dmp</li> <li>core.20211124.142026.21100.0001.dmp</li> <li>JAVA DEPENDENCIES</li> <li>if getskills-automation</li> </ul>	Terminal 0 × [INFO] - source J [INFO] - generate [INFO] - expand b [INFO] - expand b [INFO] - expand d [INFO] - expand d [INFO] - expand d [INFO] - cuecum [INFO] - cuecum	JSON report directory : /home/project/target/json-cucumber-reports ad HTML report directory : /home/project/target/generated-report pefore/after hooks : true step hooks : true doc strings : false hber Report: /home/project/target/generated-report/index.html -install-plugin:2.5.2:install (default-install) @ getskills-automation

Figure 14: Generated Report -JSON

3. The generated HTML report is found at /home/project/target/generated-report.

4. If any test fails the auto-generated report then render's clarity where it is failing, one can see the captured screenshots at **/home/project/target/generated-report/attachments** 

Google Sign in to continue to Gmail Email or phone Forgot email? Not your computer? Use Guest mode to sign in privately. Learn more
Not your computer? Use Guest mode to sign in privately.         Learn more         Create account         Next         English (United States) +         Help       Privacy

Figure 15: Screenshots in auto-generated report

It shows the page that we are getting on login.

### **Reviewing Report**

1. Right-click on target/generated-report as shown below



Figure16: Auto-generated report

2. A file will be downloaded as shown below

- →	nis PC > Downloads			
	Name	Date modified	Туре	Size
🖈 Quick access				
📃 Desktop 🛛 🖈	✓ Yesterday (5)			
🖹 Documents 🔹 🖈	meetingAttendanceReport(test runner) (1)	11/24/2021 11:48 PM	Microsoft Excel C	2 KB
Deventeerde	🔯 meetingAttendanceReport(test runner)	11/24/2021 11:48 PM	Microsoft Excel C	2 KB
	🔯 93.0.4577.63_chrome_installer	11/24/2021 5:05 PM	Application	76,117 KB
Pictures 🖈	generated-report.tar	11/24/2021 5:01 PM	TAR File	2,730 KB
OneDrive - DENS G	🔯 ChromeSetup	11/24/2021 3:29 PM	Application	1,310 KB
10731	ANTUAN - 200A0A			

Figure17: Download the auto-generated report

				represent to, the
generated-repo		Open with	٩M	TAR File 2,730 KB
👸 ChromeSetup	-P	Compare using MobaDiff	PM	Application 1,310 KB
<ul> <li>Earlier this wee</li> </ul>		Edit with MobaTextEditor	-	
Documentation		Move to OneDrive	PM	Microsoft Excel W 25 KB
🔊 Documentation		7-Zip		Open archive
<ul> <li>Last week (22)</li> </ul>		CRC SHA	Τ	Open archive >
G GrammarlyAdd	2	Edit with Notepad++		Extract files
🛓 jdk-17_window	Ê	Share	-	Extract Here
🛃 jdk-17_window		Give access to	L	Extract to "generated-report\"
🖁 haxm-windows		Restore previous versions		Test archive
app-debug.apk		Send to		Add to archive
Appium-Inspec		Cut		Compress and email
Appium-Server		Conv		Add to "generated-report_2./z"
android-studio-		сору		Compress to "generated-report_2./z" and email
idk-11.0.3 wind		Create shortcut		Add to "generated-report_2.zip"
		Delete		Compress to "generated-report_2.zip" and email
Grammariyinsta		Rename	PIVI	Application 15,811 KB
Manisha Arora		Properties	PM	IVIICTOSOTT Eage P 200 KB
Manisha Arora		· · · · · · · · · · · · · · · · · · ·	_PM	A Microsoft Word D 116 KB

3. Extract this file in your folder as shown below

Figure18: Extract to auto-generated report

4. After the extraction, a folder name **generated-report** will appear in the same directory as shown below,

← → × ↑ 🕹 > This PC > Downloads										
🖈 Quick access	Name	Date modified	Туре	Size						
Desktop	y ⊂ Today (1) generated-report	11/25/2021 12:59 PM	File folder							
Documents X										

Figure 19: Downloaded file

5. Inside this folder open the index file in the browser to see the report.

17	Ρ	а	g	е
----	---	---	---	---

I     →     I     generated-report       File     Home     Share     View							
$\leftarrow$ $\rightarrow$ $\checkmark$ $\uparrow$ $\square$ > This PC > Downloads > generated-report							
	Name	^	Date modified		Туре	Size	
A Quick access	attachment	5	11/24/2021 5:00 P	PM I	File folder		
Desktop 📌	css		11/24/2021 1:15 P	PM	File folder		
🗄 Documents 🛛 🖈	font		11/24/2021 1:15 P	PM	File folder		
🖊 Downloads 🛛 🖈	js		11/24/2021 1:15 P	PM	File folder		
📰 Pictures 🛛 🖈	pages		11/24/2021 1:15 P	PM I	File folder		
👩 OneDrive - DENS Gr	💽 index	Open	00 P	PM I	Microsoft Edge H	8 KB	
🐅 OneDrive - DENS Gro		Compare using wobabin					
Microsoft Teams Ch		🍃 Edit with MobaTextEditor					
Notebooks		📥 Move to OneDrive					
		7-Zip	>				
💻 This PC		CRC SHA	>				
3D Objects		Edit with Notepad++					

Figure 20: Screenshots in auto-generated report

6. Here is example of the report shown below

C A B file///C/Users/manisha/Downloads/generated-report/index.html						
Scenario Summa	ary All Features All Ta			021/11/24 12:10:23		
		All Scenarios		_		
		Scenario Result Chart	Scenario Summary			
		passed failed skipped	1 Scenario(s): 1 ✔ 0 ★ 0 ↔ Duration: 1m 08s 736ms			
Passed Scenarios						
		Show 25 + entries	Search:			
	Feature	<sup>†↓</sup> Scenario	<sup>↑↓</sup> Duration <sup>↑↓</sup>			
	Login in the app	Login in the Gmail - Happy path	1m 08s 736ms			

Figure 21: Scenario

## REFERENCES

www.javatpoint.com www.w3schools.com www.geeksforgeeks.com