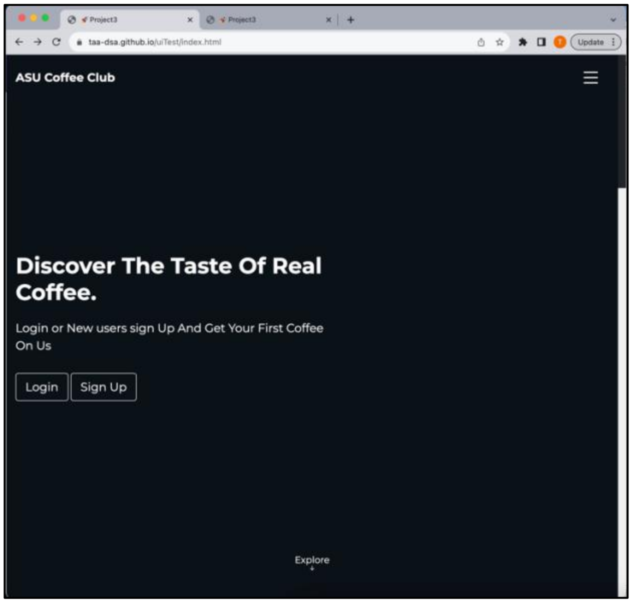
Project 3: Graphical User Interface Testing Project

Task 1: Develop two version of an application with a GUI

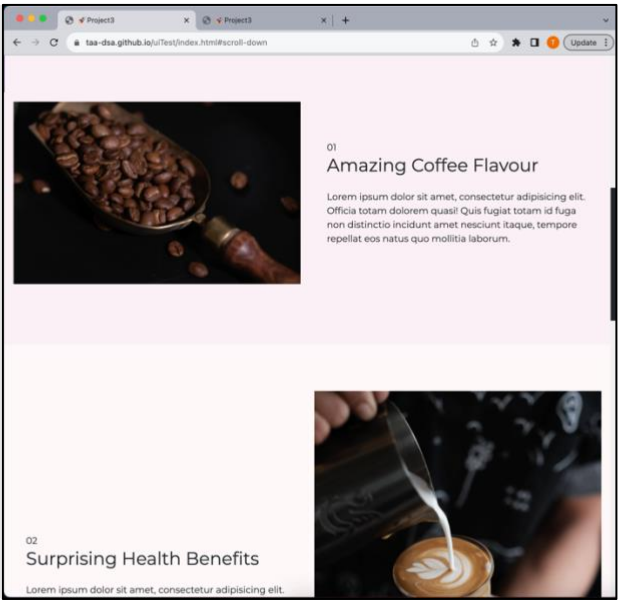
I have used an open-source GUI template and developed my own by customizing it to align with the requirements of the assignment specifications. The UI comprises of three pages: the index (home page), login page, and sign-up page. The technologies I used for the developing the GUI includes HTML, CSS, JavaScript and Bootstrap, and deployed it using GitHub pages. The live link for both version of the GUI are shared below.

Version 1: https://taa-dsa.github.io/uiTest/index.html



Discover The Taste Of Real Coffee.

Login or New users sign Up And Get Your First Coffee On Us

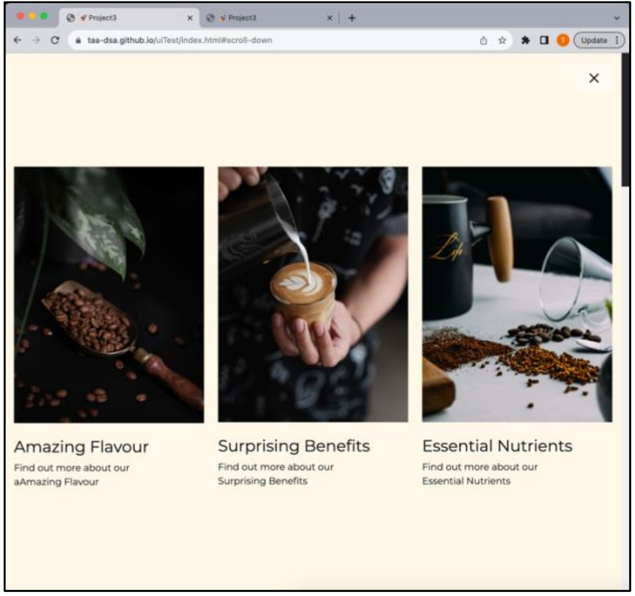


Amazing Coffee Flavour

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Officia totam dolorem quasi! Quis fugiat totam id fuga non distinctio incidunt amet nesciunt itaque, tempore repellat eos natus quo mollitia laborum.

Surprising Health Benefits

Lorem ipsum dolor sit amet, consectetur adipisicing elit.



Amazing Flavour

Find out more about our aAmazing Flavour

Surprising Benefits

Find out more about our Surprising Benefits

Essential Nutrients

Find out more about our Essential Nutrients

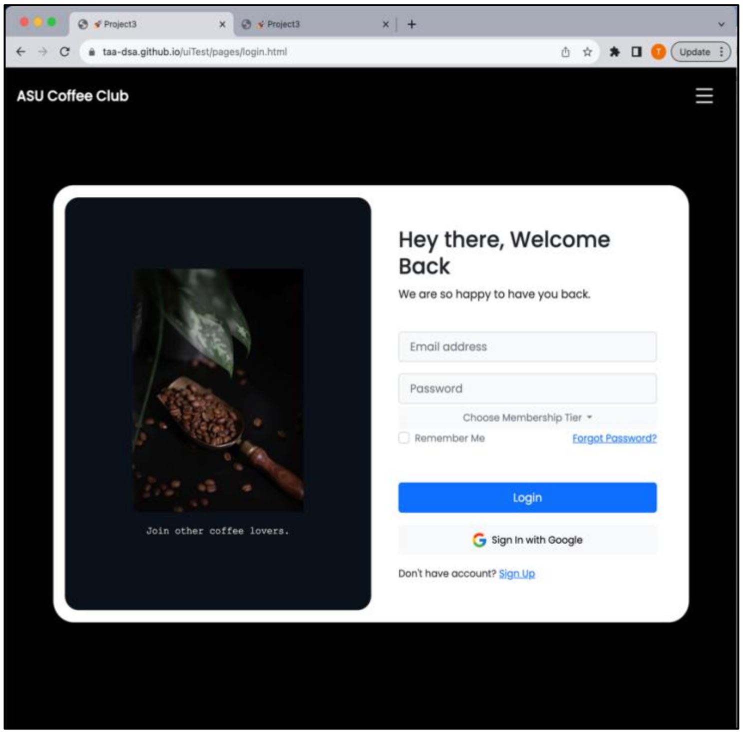
Page 1: index. Html (Home page)

GUI elements:

- Images
- Buttons
- Hamburger menu
- Scroll-down (label explore)

Flow:

- Login button navigates to login.html page
- Signup button navigates to signup.html page



Hey there, Welcome Back

We are so happy to have you back.

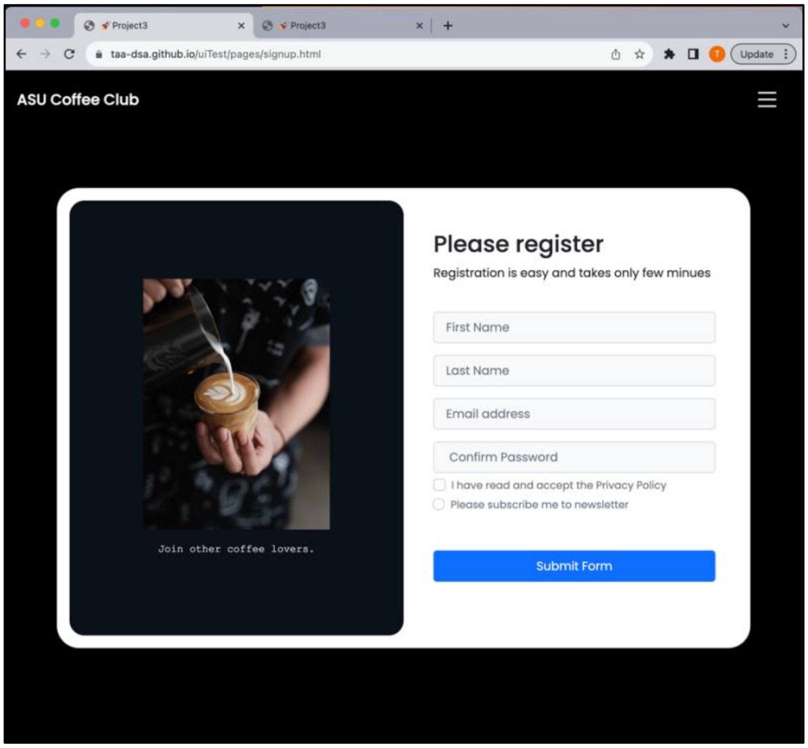
Join other coffee lovers.

Don't have account? Sign Up

Page 2: Login.html

GUI elements:

- Text boxes
- Drop-down list menu
- Checkbox
- Buttons



Please register

Registration is easy and takes only few minues

First Name

Last Name

Email address

Confirm Password

□ I have read and accept the Privacy Policy

○ Please subscribe me to newsletter

Submit Form

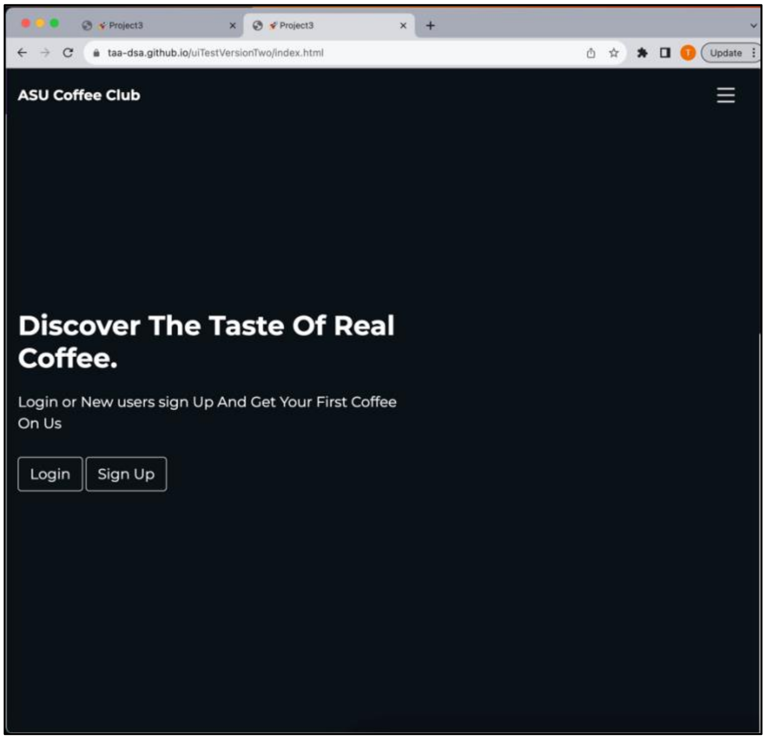
Join other coffee lovers.

Page 3: signup.html

GUI elements:

- Textboxes
- Checkbox
- Radio button
- Button

Version 2: https://taa-dsa.github.io/uiTestVersionTwo/index.html

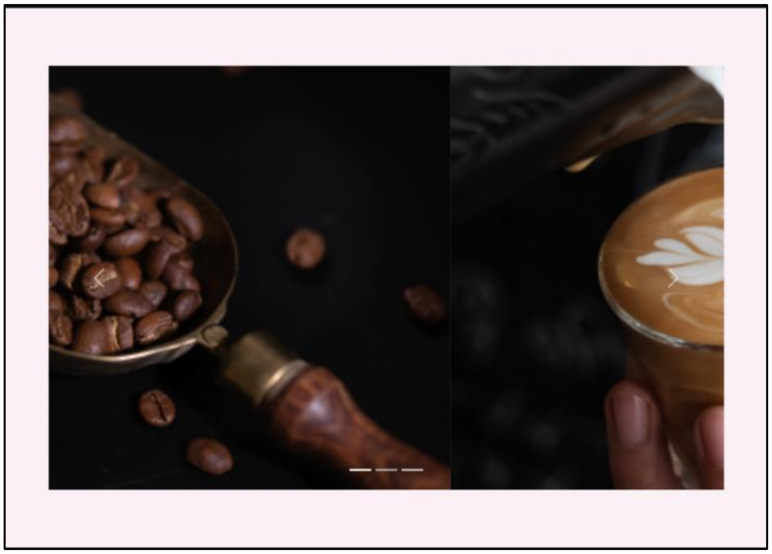


ASU Coffee Club

Discover The Taste Of Real Coffee.

Login or New users sign Up And Get Your First Coffee On Us

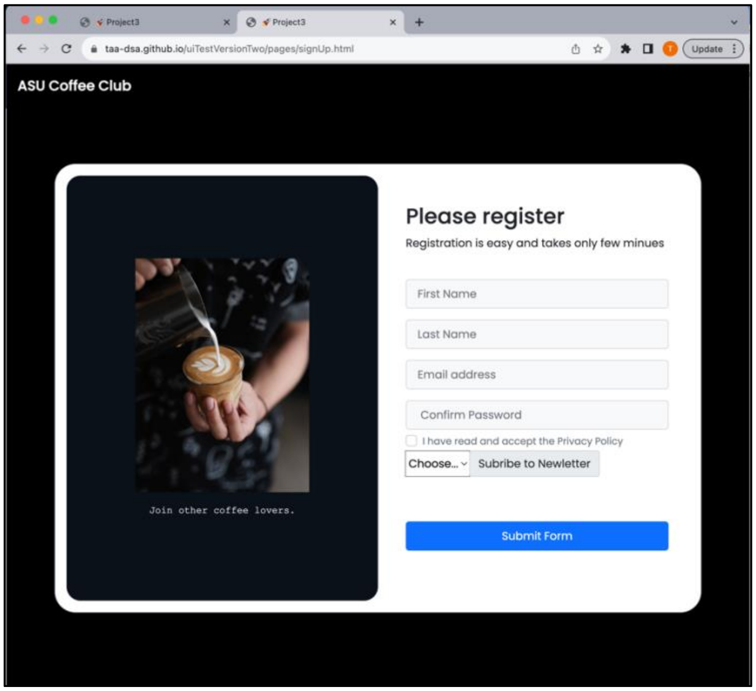
Login Sign Up



Page 1: index.html

Changes:

- Removed scroll-down
- Added slider / carousel
- Flow change: login button now navigates to signup page
- Flow change: signup button now navigates to login page



ASU Coffee Club

Please register

Registration is easy and takes only few minues

First Name

Last Name

Email address

Confirm Password

□ I have read and accept the Privacy Policy

Choose... Subribe to Newletter

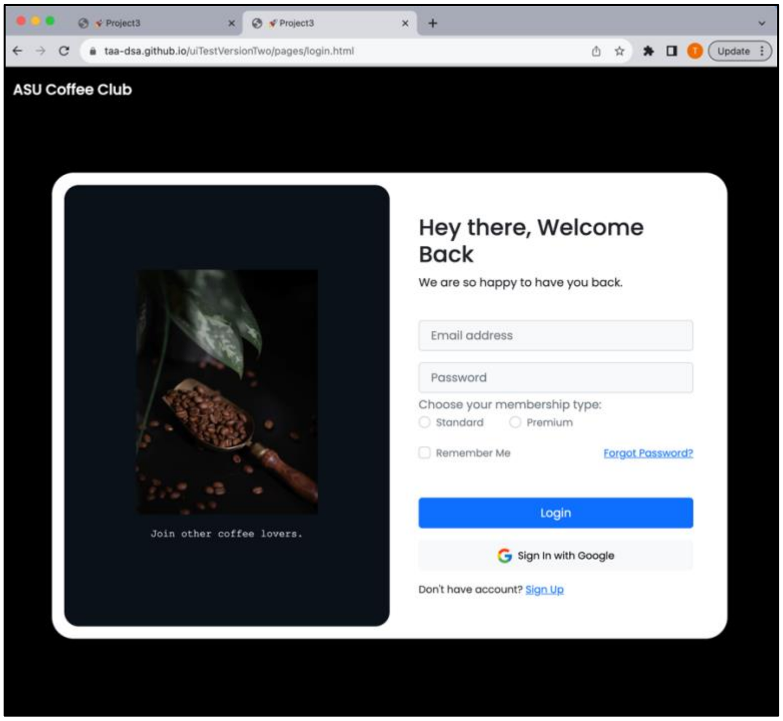
Submit Form

Join other coffee lovers.

Page 2: signup.html

Changes:

- Replaced radio button with drop-down list menu



ASU Coffee Club

Hey there, Welcome Back

We are so happy to have you back.

Email address

Password

Choose your membership type:

○ Standard ○ Premium

□ Remember Me Forgot Password?

Login

Sign In with Google

Don't have account? Sign Up

Join other coffee lovers.

Page 3: login.html

Changes:

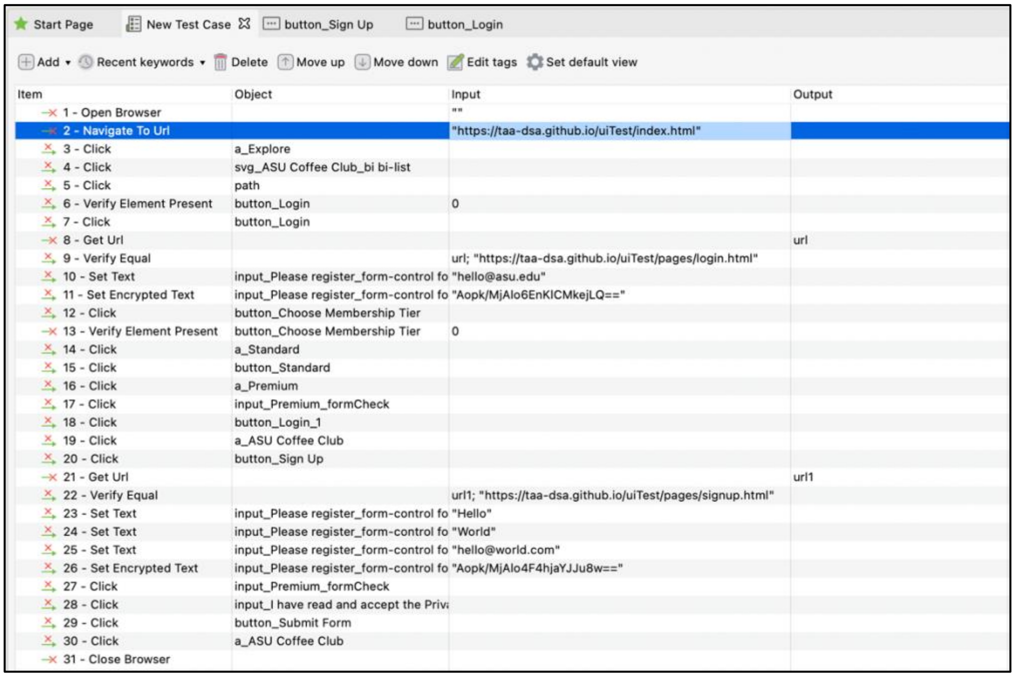
- Replaced drop-down with radio button

Task 2: Research and identify a tool for testing GUI of an application.

After conducting a thorough research, I came across Katalon studio an all-in-one test automation tool that will be discussed in details later in this report.

Task 3:

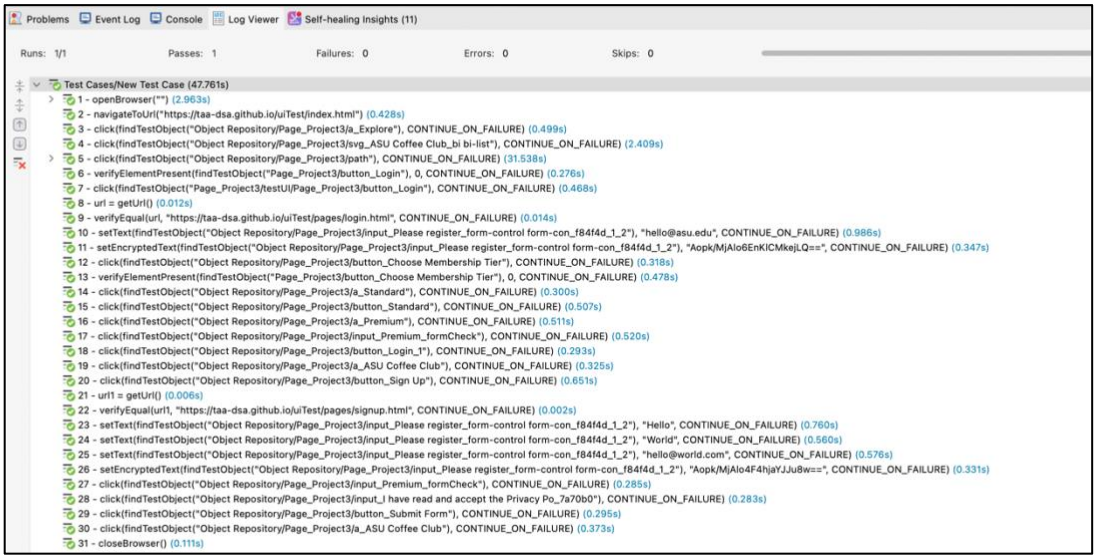
Test Cases:



Item	Object	Input	Output
1 - Open Browser		""	
2 - Navigate To Url		"https://taa-dsa.github.io/uiTest/index.html"	
3 - Click	a_Explore		
4 - Click	svg_ASU Coffee Club_bi bi-list		
5 - Click	path		
6 - Verify Element Present	button_Login	0	
7 - Click	button_Login		
8 - Get Url			url
9 - Verify Equal		url; "https://taa-dsa.github.io/uiTest/pages/login.html"	
10 - Set Text	input_Please register_form-control fo	"hello@asu.edu"	
11 - Set Encrypted Text	input_Please register_form-control fo	"Aopk/MjAlo6EnKlCMkejLQ=="	
12 - Click	button_Choose Membership Tier		
13 - Verify Element Present	button_Choose Membership Tier	0	
14 - Click	a_Standard		
15 - Click	button_Standard		
16 - Click	a_Premium		
17 - Click	input_Premium_formCheck		
18 - Click	button_Login_1		
19 - Click	a_ASU Coffee Club		
20 - Click	button_Sign Up		
21 - Get Url			url1
22 - Verify Equal		url1; "https://taa-dsa.github.io/uiTest/pages/signup.html"	
23 - Set Text	input_Please register_form-control fo	"Hello"	
24 - Set Text	input_Please register_form-control fo	"World"	
25 - Set Text	input_Please register_form-control fo	"hello@world.com"	
26 - Set Encrypted Text	input_Please register_form-control fo	"Aopk/MjAlo4F4hjaYJJu8w=="	
27 - Click	input_Premium_formCheck		
28 - Click	input_I have read and accept the Priv		
29 - Click	button_Submit Form		
30 - Click	a_ASU Coffee Club		
31 - Close Browser			

- ‘Click’ checks if target element is clickable.
- ‘Verify element present’ checks existence of the targeted UI element
- ‘Verify equals’ checks the correctness of the link flow.
- ‘Set text’ checks text box functionality
- “Set encrypted Text’ checks the password field

Result: Shows all test cases pass for version 1 of the GUI.



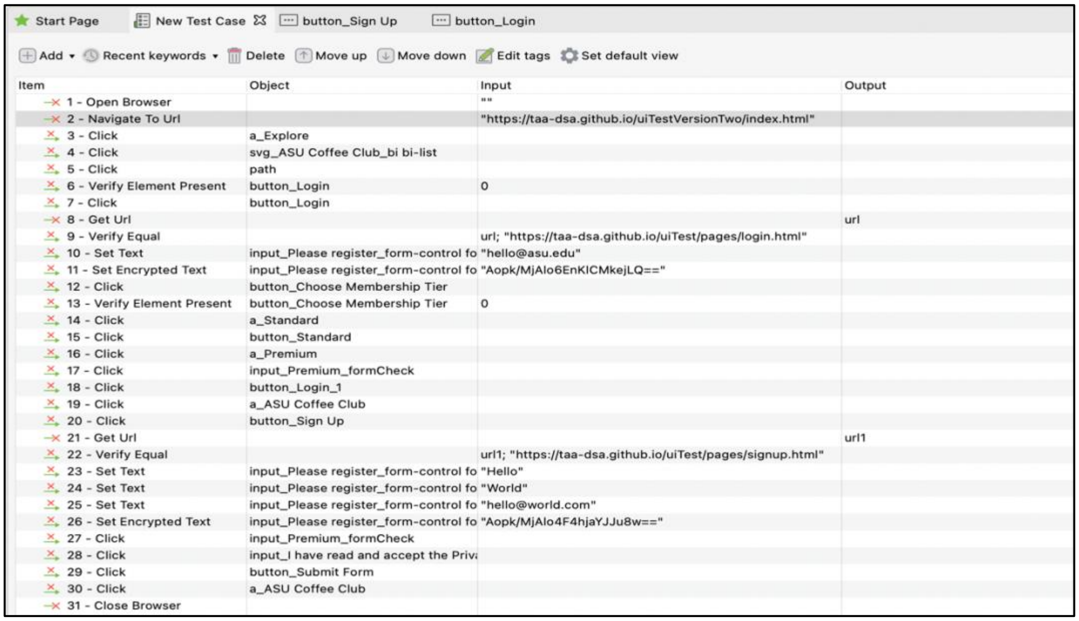
```
Runs:  1/1          Passes:  1          Failures:  0          Errors:  0          Skips:  0

Test Cases/New Test Case (47.761s)
  1 - openBrowser("") (2.963s)
  2 - navigateToUrl("https://taa-dsa.github.io/uiTest/index.html") (0.428s)
  3 - click(findTestObject("Object Repository/Page_Project3/a_Explore"), CONTINUE_ON_FAILURE) (0.499s)
  4 - click(findTestObject("Object Repository/Page_Project3/svg_ASU Coffee Club_bi bi-list"), CONTINUE_ON_FAILURE) (2.409s)
  5 - click(findTestObject("Object Repository/Page_Project3/path"), CONTINUE_ON_FAILURE) (31.538s)
  6 - verifyElementPresent(findTestObject("Page_Project3/button_Login"), 0, CONTINUE_ON_FAILURE) (0.276s)
  7 - click(findTestObject("Page_Project3/testUI/Page_Project3/button_Login"), CONTINUE_ON_FAILURE) (0.468s)
  8 - url = getUrl() (0.012s)
  9 - verifyEqual(url, "https://taa-dsa.github.io/uiTest/pages/login.html", CONTINUE_ON_FAILURE) (0.014s)
  10 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "hello@asu.edu", CONTINUE_ON_FAILURE) (0.986s)
  11 - setEncryptedText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "Aopk/MjAlo6EnKICMkejLQ==", CONTINUE_ON_FAILURE) (0.347s)
  12 - click(findTestObject("Object Repository/Page_Project3/button_Choose Membership Tier"), CONTINUE_ON_FAILURE) (0.318s)
  13 - verifyElementPresent(findTestObject("Page_Project3/button_Choose Membership Tier"), 0, CONTINUE_ON_FAILURE) (0.478s)
  14 - click(findTestObject("Object Repository/Page_Project3/a_Standard"), CONTINUE_ON_FAILURE) (0.300s)
  15 - click(findTestObject("Object Repository/Page_Project3/button_Standard"), CONTINUE_ON_FAILURE) (0.507s)
  16 - click(findTestObject("Object Repository/Page_Project3/a_Premium"), CONTINUE_ON_FAILURE) (0.511s)
  17 - click(findTestObject("Object Repository/Page_Project3/input_Premium_formCheck"), CONTINUE_ON_FAILURE) (0.520s)
  18 - click(findTestObject("Object Repository/Page_Project3/button_Login_1"), CONTINUE_ON_FAILURE) (0.293s)
  19 - click(findTestObject("Object Repository/Page_Project3/a_ASU Coffee Club"), CONTINUE_ON_FAILURE) (0.325s)
  20 - click(findTestObject("Object Repository/Page_Project3/button_Sign Up"), CONTINUE_ON_FAILURE) (0.651s)
  21 - url1 = getUrl() (0.006s)
  22 - verifyEqual(url1, "https://taa-dsa.github.io/uiTest/pages/signup.html", CONTINUE_ON_FAILURE) (0.002s)
  23 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "Hello", CONTINUE_ON_FAILURE) (0.760s)
  24 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "World", CONTINUE_ON_FAILURE) (0.560s)
  25 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "hello@world.com", CONTINUE_ON_FAILURE) (0.576s)
  26 - setEncryptedText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "Aopk/MjAlo4F4hjaYJJu8w==", CONTINUE_ON_FAILURE) (0.331s)
  27 - click(findTestObject("Object Repository/Page_Project3/input_Premium_formCheck"), CONTINUE_ON_FAILURE) (0.285s)
  28 - click(findTestObject("Object Repository/Page_Project3/input_I have read and accept the Privacy Po_7a70b0"), CONTINUE_ON_FAILURE) (0.283s)
  29 - click(findTestObject("Object Repository/Page_Project3/button_Submit Form"), CONTINUE_ON_FAILURE) (0.295s)
  30 - click(findTestObject("Object Repository/Page_Project3/a_ASU Coffee Club"), CONTINUE_ON_FAILURE) (0.373s)
  31 - closeBrowser() (0.111s)
```

Task 4:

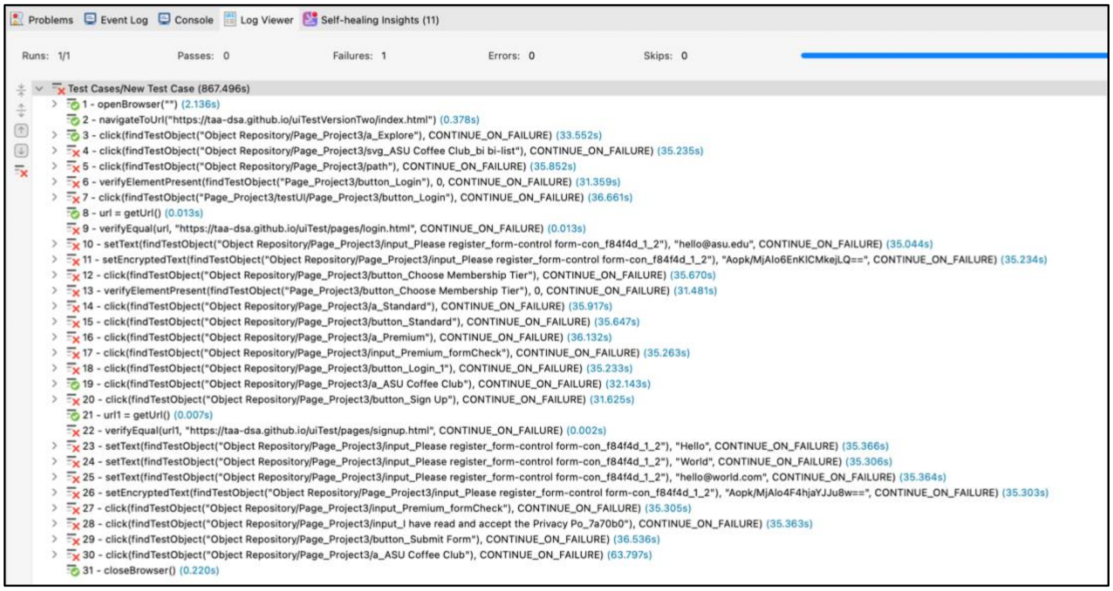
Executing the same test cases on second version of GUI resulted in 24 fail test cases. The screen shot of the test cases and result are shown below.

Test Cases:



Item	Object	Input	Output
1 - Open Browser		""	
2 - Navigate To Url		"https://taa-dsa.github.io/uiTestVersionTwo/index.html"	
3 - Click	a_Explore		
4 - Click	svg_ASU Coffee Club_bi bi-list		
5 - Click	path		
6 - Verify Element Present	button_Login	0	
7 - Click	button_Login		
8 - Get Url			url
9 - Verify Equal		url; "https://taa-dsa.github.io/uiTest/pages/login.html"	
10 - Set Text	input_Please register_form-control fo	"hello@asu.edu"	
11 - Set Encrypted Text	input_Please register_form-control fo	"Aopk/MjAlo6EnKICMkejLQ=="	
12 - Click	button_Choose Membership Tier		
13 - Verify Element Present	button_Choose Membership Tier	0	
14 - Click	a_Standard		
15 - Click	button_Standard		
16 - Click	a_Premium		
17 - Click	input_Premium_formCheck		
18 - Click	button_Login_1		
19 - Click	a_ASU Coffee Club		
20 - Click	button_Sign Up		
21 - Get Url			url1
22 - Verify Equal		url1; "https://taa-dsa.github.io/uiTest/pages/signup.html"	
23 - Set Text	input_Please register_form-control fo	"Hello"	
24 - Set Text	input_Please register_form-control fo	"World"	
25 - Set Text	input_Please register_form-control fo	"hello@world.com"	
26 - Set Encrypted Text	input_Please register_form-control fo	"Aopk/MjAlo4F4hjaYJJu8w=="	
27 - Click	input_Premium_formCheck		
28 - Click	input_I have read and accept the Priv:		
29 - Click	button_Submit Form		
30 - Click	a_ASU Coffee Club		
31 - Close Browser			

Results: Majority of test cases that passed the version one of the GUI failed including the flow as the links were change.



```
Runs: 1/1          Passes: 0          Failures: 1          Errors: 0          Skips: 0

Test Cases/New Test Case (867.496s)
  1 - openBrowser("") (2.136s)
  2 - navigateToUrl("https://taa-dsa.github.io/uiTestVersionTwo/index.html") (0.378s)
  3 - click(findTestObject("Object Repository/Page_Project3/a_Explore"), CONTINUE_ON_FAILURE) (33.552s)
  4 - click(findTestObject("Object Repository/Page_Project3/svg_ASU Coffee Club_bi bi-list"), CONTINUE_ON_FAILURE) (35.235s)
  5 - click(findTestObject("Object Repository/Page_Project3/path"), CONTINUE_ON_FAILURE) (35.852s)
  6 - verifyElementPresent(findTestObject("Page_Project3/button_Login"), 0, CONTINUE_ON_FAILURE) (31.359s)
  7 - click(findTestObject("Page_Project3/testUI/Page_Project3/button_Login"), CONTINUE_ON_FAILURE) (36.661s)
  8 - url = getUrl() (0.013s)
  9 - verifyEqual(url, "https://taa-dsa.github.io/uiTest/pages/login.html", CONTINUE_ON_FAILURE) (0.013s)
  10 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "hello@asu.edu", CONTINUE_ON_FAILURE) (35.044s)
  11 - setEncryptedText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "Aopk/MjAlo6EnKICMkejLQ==", CONTINUE_ON_FAILURE) (35.234s)
  12 - click(findTestObject("Object Repository/Page_Project3/button_Choose Membership Tier"), CONTINUE_ON_FAILURE) (35.670s)
  13 - verifyElementPresent(findTestObject("Page_Project3/button_Choose Membership Tier"), 0, CONTINUE_ON_FAILURE) (31.481s)
  14 - click(findTestObject("Object Repository/Page_Project3/a_Standard"), CONTINUE_ON_FAILURE) (35.917s)
  15 - click(findTestObject("Object Repository/Page_Project3/button_Standard"), CONTINUE_ON_FAILURE) (35.647s)
  16 - click(findTestObject("Object Repository/Page_Project3/a_Premium"), CONTINUE_ON_FAILURE) (36.132s)
  17 - click(findTestObject("Object Repository/Page_Project3/input_Premium_formCheck"), CONTINUE_ON_FAILURE) (35.263s)
  18 - click(findTestObject("Object Repository/Page_Project3/button_Login_1"), CONTINUE_ON_FAILURE) (35.233s)
  19 - click(findTestObject("Object Repository/Page_Project3/a_ASU Coffee Club"), CONTINUE_ON_FAILURE) (32.143s)
  20 - click(findTestObject("Object Repository/Page_Project3/button_Sign Up"), CONTINUE_ON_FAILURE) (31.625s)
  21 - url1 = getUrl() (0.007s)
  22 - verifyEqual(url1, "https://taa-dsa.github.io/uiTest/pages/signup.html", CONTINUE_ON_FAILURE) (0.002s)
  23 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "Hello", CONTINUE_ON_FAILURE) (35.366s)
  24 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "World", CONTINUE_ON_FAILURE) (35.306s)
  25 - setText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "hello@world.com", CONTINUE_ON_FAILURE) (35.364s)
  26 - setEncryptedText(findTestObject("Object Repository/Page_Project3/input_Please register_form-control form-con_f84f4d_1_2"), "Aopk/MjAlo4F4hjaYJJu8w==", CONTINUE_ON_FAILURE) (35.303s)
  27 - click(findTestObject("Object Repository/Page_Project3/input_Premium_formCheck"), CONTINUE_ON_FAILURE) (35.305s)
  28 - click(findTestObject("Object Repository/Page_Project3/input_I have read and accept the Privacy Po_7a70b0"), CONTINUE_ON_FAILURE) (35.363s)
  29 - click(findTestObject("Object Repository/Page_Project3/button_Submit Form"), CONTINUE_ON_FAILURE) (36.536s)
  30 - click(findTestObject("Object Repository/Page_Project3/a_ASU Coffee Club"), CONTINUE_ON_FAILURE) (63.797s)
  31 - closeBrowser() (0.220s)
```

Task 5: Assessment of the tool:

- Set of features and functionalities provided: Katalon studio is a powerful test automation tool widely used and offers a comprehensive set of features and functionality to automate testing. Katalon is easy to install and offers both codeless & code-based testing, catering to testers with coding and non-coding background. The programming language it uses is called Groovy and provides end-to-end automation solutions, including test case design, execution, debugging and reporting. Testers can seamlessly automate web, mobile and API testing within a single integrated environment. Katalon studio also offers robust test management capabilities, making it a multi-purpose tool for automated testing.

- Type of coverage: Katalon studio offers wide range of coverage across various testing types. It supports functional testing, smoke testing, regression testing, performance testing, security testing, acceptance testing and even continuous integration and delivery (CI/CD) integration. With support for both web and mobile applications, it caters to a wide range of testing requirements ensuring comprehensive coverage of the software application. The coverage implemented in this assignment is Web GUI.

- Reuse of test cases test results produced: Katalon support test case reusability, test scripts can be modularized and reused across different test scenarios, saving time and effort. For example, a test case for the login page can be written and saved in test suites to re-use on a different GUI. However, for this assignment, I didn't create a test suite instead reused the test case developed for version 1 to test version 2 by simply changing the navigation URL.

- Ease of usage: it offers a user-friendly interface suitable for both beginners and professional. For this assignment, I utilized the katalon's 'Spy Web' and 'Record Web' function to develop test cases. ‘Spy Web’ captures web elements simply by hovering over the element and right-clicking on the capture element option. This feature saves the element in the object repository and can be used to write test cases to check whether they meet the test objectives. ‘Record Web’ captures elements in the similar process and also writes test cases. However, it does require some time to learn and fully understand and utilize all the feature of the tool to perform a thorough GUI testing. ‘Record Web’ does not create all test case for example to test the link I had manually write the test case following its documentation that can be very efficient and helpful when using the tool for the first time. Overall, it is fairly easy-to-use tool for developing and executing simple test cases on GUI.

- Type of GUI elements that can be tested: Katalon studio supports the testing of a wide array of graphical user interfaces (GUIs) including buttons, images and textboxes. It can be used to test web application, including modern application based on framework as well as mobile application on iOS and Android platforms.

References:

https://katalon.com/business-outcomes

https://katalon.com/katalon-studio?utm_term=_c_&utm_campaign=postman&utm_source=adwords&utm_medium=pmcs&hsa_acc=4835360476&hsa_cam=20448028530&hsa_grp=&hsa_ad=&hsa_src=x&hsa_tgt=&hsa_kw=&hsa_mt=&hsa_net=adwords&hsa_ver=3&gclid=Cj0KCQjw1OmoBhDXARIsAAAYGSEdCMM3wTk34bAiuPg5bdid4a-OllkLNci26Tw7ab5uIc6Mc7Wa5CsaAjFpEALw_wcB

https://katalon.com/resources-center/blog/automation-testing-types

https://docs.katalon.com/docs/supported-technologies/supported-technologies-for-katalon-studio

https://docs.katalon.com/docs/create-tests/keywords/custom-keywords/how-to-use-a-custom-keyword-from-other-custom-keywords-and-step-definition-classes-in-katalon-studio#ariaid-title1

https://docs.katalon.com/docs/get-started/sample-projects/webui/webui-create-and-run-web-ui-test-case-using-record-and-playback-in-katalon-studio

https://theqalead.com/test-management/katalon-studio-overview/#:~:text=Katalon%20Studio's%20test%20reports%20can,for%20data%2Ddriven%20result%20analysis.

https://www.youtube.com/watch?v=DvfezgoBRzY&t=1423s

https://www.youtube.com/watch?v=oF28ns9eVdc&t=380s