Workbook

Software Quality Engineering (SE-309)

Name

Roll No.

Batch

Year

Department

Workbook

Software Quality Engineering (SE-309)

Prepared by
Engr. Sana Fatima
Lecturer- SE

Approved by
Chairperson

Department of Software Engineering

Table of Contents

NED University of Engineering & Technology

Department of Software Engineering

Course Code and Title: SE-309 Software Quality engineering

Laboratory Session No: _____ Date: _____

Psychomotor Domain Assessment Rubric – Level P3 (for Laboratory Work)

Rubric Table

Criterion	1 - Not Able	2 - Inadequate	3 - Adequate	4 - Effective	5 - Outstanding
Problem Identification in Test Planning	Fails to identify testing scope or required data.	Partially identifies testing scope but lacks clarity.	Identifies major test areas and basic data.	Clearly identifies testing requirements with most data.	Completely identifies scope, inputs/outputs, and dependencies.
Test Strategy and Design Skills	No defined strategy; test cases are missing or irrelevant.	Some strategy present; test cases are incomplete or vague.	Applies basic strategy; test cases cover standard flows.	Well-structured strategy with meaningful test coverage.	Excellent use of strategy (e.g., BVA, ECP) with complete, traceable test cases.
Test Case Implementation and Execution	No test cases implemented or test tool not used.	Basic execution attempted; errors or missing cases.	Functional implementation with partial results.	Most test cases executed correctly using appropriate tools.	Flawless execution and effective automation (if applicable).
Output Validation and Reporting	Fails to match outputs to expectations or report results.	Basic validation done; report lacks structure.	Expected outputs matched; simple reporting used.	Good result validation and clear documentation.	Precise validation, comprehensive report/logs with screenshots or data.
Depth of subject Knowledge	Does not demonstrate understanding of testing principles.	Limited understanding; misapplies concepts.	Basic understanding and some correct application.	Good grasp and application of core testing principles.	Advanced application of principles (e.g., white/black box, automation).

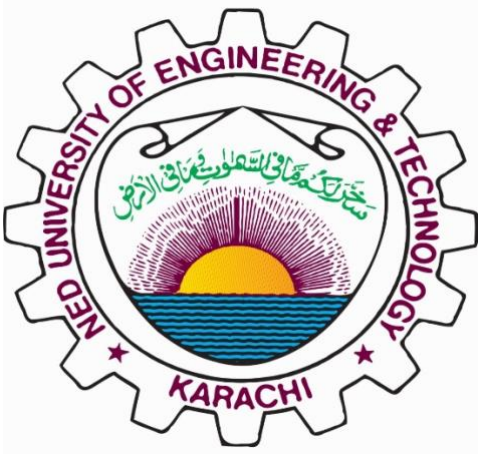
Weighted CLO (Cognitive) Score: _____

Remarks: _____

Instructor's Signature with Date: _____

NED University of Engineering and Technology

Department of Software Engineering



SOFTWARE QUALITY ENGINEERING (SE-309)

LAB MANUAL

Group Members

M Ali Arsalan	(SE-22032)
Syed Ibad Ali	(SE-22034)
Samiullah	(SE-22046)
Vaiz Hasan	(SE-22047)
Faez Ansar Ahmed	(SE-22050)

Submitted To: Ms. Sana Fatima

Lab 01: Introduction to Software Testing

EXERCISE:

1. Name the stages involved in the software testing life cycle, and differentiate test planning and test analysis.

Software Testing Life Cycle consists of six (generic) phases:

- Test Planning.
- Test Analysis.
- Test Design.
- Construction and verification.
- Test Cycles.
- Final Testing Implementation.

Difference Between Test Planning and Test Analysis

Test Planning

Test planning involves defining the scope, resources, approach, and schedule for testing. Its primary objective is to create a high-level roadmap for testing, specifying what needs to be tested and how. The deliverable is the Software Test Plan (STP).

Test Analysis

Test analysis focuses on identifying specific test conditions and deciding on testing methodologies. It involves reviewing documentation and results in the creation of the Functional Validation Matrix, test scenarios, and prioritized test cases.

2. Differentiate between white box and black box testing techniques. Which technique do you prefer where you are asked to perform testing at component level and why? Explain with suitable examples.

Difference Between White Box and Black Box Testing Techniques

Aspect	White Box Testing	Black Box Testing
Definition	Testing based on the internal structure, logic, and flow of the application code.	Testing based on the functionality and user interface (UI) of the application without any knowledge of the internal code.
Focus	Primarily focuses on code quality, logic, and execution paths.	Primarily focuses on validating the application's input-output behavior, user interface.
Testing Levels	Primarily used for unit testing and integration testing.	Commonly used for system testing, user acceptance testing, and UI testing.
Techniques Used	- Path testing, Statement and branch coverage, Condition Testing	- Equivalence partitioning, Boundary value analysis, Decision table testing
Example	Testing the logic of a sorting algorithm to ensure all conditions and branches are handled correctly.	Testing the UI of a login page: verify the layout, input validation, error messages, and overall functionality without inspecting the code.

Preferred Technique for Component-Level Testing

White Box Testing is the preferred technique for **component-level testing**.

Reasons:

1. **Component-Level Verification**:
    - At the component level, testing focuses on validating the logic, flow, and behavior of individual units or modules. White box testing enables a thorough examination of these aspects.
2. **Early Bug Detection**:
    - It identifies issues at the earliest stage of development, such as syntax errors, logical flaws, or unhandled exceptions.

Example

Scenario: Calculator

Testing a function in a calculator application that calculates the factorial of a number (factorial(n)).

- **White Box Testing**:
    - Verify the function handles valid inputs (e.g., factorial(4) returns 24).
    - Test edge cases like factorial(0) (returns 1) and factorial(1) (returns 1).
    - Check for negative inputs and ensure the function throws appropriate exceptions or errors.
    - Analyze and test loops (e.g., correct number of iterations).
- **Black Box Testing**:
    - Input: n = 4, Expected Output: 24.
    - Input: n = 0, Expected Output: 1.
    - Input: n = -3, Expected Output: "Invalid input".

3. Explain software testing levels.

Software testing levels represent the different stages in the software development lifecycle (SDLC) where testing is performed. These levels help ensure that each component of the software works correctly both individually and collectively. Below are the main testing levels:

1. Unit Testing

- **Purpose**: Focuses on verifying the smallest testable parts of the application (units/modules), such as functions, classes, or procedures.
- **Performed By**: Typically, developers who write the code.
- **Goal**: Ensure that individual units function as intended.
- **Example**: Testing a function that calculates the total price of items in a shopping cart.
- **Tools**: JUnit, NUnit, TestNG.

2. Integration Testing

- **Purpose**: Verifies that different modules or units interact correctly with each other.
- **Approaches**:

  Top-Down: Testing starts with high-level modules and progresses to lower-level modules.

  Bottom-Up: Testing begins with lower-level modules and moves upward.

  Big-Bang: All modules are integrated and tested at once.

- **Goal**: Detect interface issues between modules.
- **Example**: Testing data flow between a user interface and a backend API.
- **Tools**: Postman, SoapUI, JUnit.

3. System Testing

- **Purpose**: Tests the software as a complete and integrated system.
- **Performed By**: Independent testing teams, not the developers.
- **Focus**: Functional and non-functional requirements.
- **Goal**: Validate that the system meets the specified requirements and behaves as expected under realistic conditions.
- **Example**: Testing the entire e-commerce platform for order placement, payment processing, and order tracking.
- **Tools**: Selenium, QTP, JMeter.

4. User Acceptance Testing (UAT)

- **Purpose**: Ensures that the software meets end-user requirements and is ready for deployment.
- **Performed By**: End-users or client representatives.
- **Types**:

  Alpha Testing: Performed in a controlled environment with a limited audience (internal users).

  Beta Testing: Performed in a real-world environment with a broader audience (external users).

- **Goal**: Validate that the software is user-friendly, functional, and meets business needs.
- **Example**: Validating whether the navigation of a website is intuitive for users.

These levels of testing ensure comprehensive validation and verification of the software throughout its development lifecycle. Each level targets specific objectives, contributing to the overall quality and reliability of the software.

4. Write down some advantages of software testing

Following are some advantages of software testing:

1. **Detects and Fixes Bugs Early**: By identifying bugs and defects early in the development process, testing reduces the cost and time needed for fixes later.

2. **Improves User Satisfaction**: Reliable and bug-free software leads to a better user experience, resulting in higher customer satisfaction.

3. **Reduces Development Costs**: Early detection and correction of issues reduce the cost of fixing problems during later stages of development.

4. **Facilitates Compliance**: Testing helps ensure that the software adheres to industry standards, legal requirements, and compliance guidelines.

5. **Builds Developer Confidence**: A well-tested application boosts the confidence of developers, making deployment and maintenance smoother.

6. **Supports Continuous Improvement**: Regular testing during the software lifecycle supports iterative development and continuous improvement.

5. Write a test set for copying a file XYZ from folder A to folder B. (at least 5 test cases)

TEST SET:

1. To verify that the file XYZ exists in folder A.
2. To Verify the system permissions for copying files from folder A to folder B.
3. To Verify the paste functionality when a file with the same name (XYZ) already exists in folder B.
4. To Verify the paste functionality when there is not enough space in folder B to accommodate file XYZ.
5. To Verify the copy-paste functionality when file XYZ is open or locked in another location.

TEST CASES:

Project Name	Microsoft Windows 10	**Test Designed By**	Ali Arsalan
Test Case ID	MS_01	**Test Designed Date**	04-Jan-25
Test Priority (Low/Medium/High)	Medium	**Test Executed By**	Faez Ansar
Module Name	Copy-Paste File Module	**Test Executed Date**	21-Jan-25
Test Title	Verify the Copy/Paste functionality of windows		
Description	Test the copy/paste functionality of windows		
Pre-conditions	Folder A must contain file XYZ. Folder B must be accessible for copying files from Folder A		
Dependencies	Read access to the source file and write access to the destination folder for the copy operation, proper file system access, required user permissions, file system compatibility.		

S.No	Test Case	Test Steps	Test Data	Expected Output	Actual Output	Status	Comments
MS_Windows10_1	To verify that the file XYZ exists in folder A	Navigate to folder A, search for file XYZ	File XYZ in folder A	File XYZ is present in folder A	File XYZ is present in folder A	Pass	–
MS_Windows10_2	To verify system permissions for copying files from folder A to folder B	Attempt to copy file XYZ from folder A to folder B	File XYZ from folder A	Successful copy with proper permissions	"File XYZ was successfully copied from folder A to folder B without any permission denial."	Pass	–
MS_Windows10_3	To verify the paste functionality when a file with the same name (XYZ) already exists in folder B	Copy file XYZ to folder B where a file named XYZ already exists	File XYZ from folder A, Folder B with XYZ file	The destination already have file name "XYZ" Replace the file Skip file Compare info of both files	The replace or skip file message is successfully delivered	Pass	–
MS_Windows10_4	To verify the paste functionality when there is not enough space in folder B to accommodate file XYZ	Copy file XYZ to folder B with insufficient space	File XYZ from folder A, Folder B with insufficient space	System displays an error message about insufficient space	System displays an error message about insufficient space	Pass	–
MS_Windows10_5	To verify the copy-paste functionality when file XYZ is open or locked in another location	Attempt to copy file XYZ while it is open/locked	File XYZ from folder A, File XYZ open in another program	System displays an error or warning that the file is open/locked in another location	The file is being successfully pasted	Fail	The file is being pasted succeffuly when open/locked in othe location

Post Condition	Verify the file exists in folder B if copied successfully and no system errors occur during the copy-paste process.

Test Designed By	Ali Arsalan
Test Designed Date	04-Jan-25
Test Executed By	Faez Ansar
Test Executed Date	21-Jan-25

Lab 02: Manual Testing: Test Cases Creation

EXERCISE:

Write detailed Test Cases for Facebook Signup and Login pages by using template.

Facebook Signup Page:

Project Name:	Facebook
Module Name:	Signup
Reference Document:	-
Created by:	Samiullah
Date of Creation:	17-Jan-25
Date of Review:	27-Jan-25

Test Case ID	Test Scenario	Test Case	Pre-condition	Test Steps	Test Data	Expected Output	Actual Output	Pass/Fail	Comments
TC_001	Valid account creation	User enters valid credentials	The system should not have any existing account with the entered email	1) Navigate to the Sign-Up UI	First Name = Ali	A new account should be created	New account is created	Pass	-
				2) Enter all required details	Surname = Arslan				
				3) Press the Sign-Up button	Date of Birth = 25-03-2004				
					Gender = Male				
					Email = arsalanali873@gmail.com				
					Password = aliNED2022				
TC_002	Duplicate email registration	User tries to sign up with an already registered email	The email address is already registered in the system	1) Navigate to the Sign-Up UI	Email = arsalanali873@gmail.com	A warning message: "Account with this email exists" should appear	Account with this email exists	Pass	-
TC_003	Invalid email format	User enters an invalid email format	No email format validation error has occurred yet	1) Navigate to the Sign-Up UI	Email = arsalan823ali.com	A warning message: "Invalid email format" should appear	Account created	Fail	Email validation mechanism is not correctly implemented
TC_004	Missing required fields	User submits the form with one or more required fields empty	All required fields must be visible on the Sign-Up form	1) Navigate to the Sign-Up UI	First Name = Ali	A warning message: "Missing required fields" should appear	Warning appeared	Pass	-
TC_005	Weak password detection	User enters a weak password (e.g., insufficient complexity)	Password field should accept input of any length but require complexity rules	1) Navigate to the Sign-Up UI	First Name = Ali	The system should warn the user about a weak password	Account created	Fail	-
TC_006	Age validation	User tries to sign up but is below the minimum age requirement	The system should have a minimum age requirement of 18 years	1) Navigate to the Sign-Up UI	First Name = Ali	A warning message: "User must be at least 18 years old" should appear	Account created	Fail	Age validation logic is incorrect (accepting underage users)
TC_007	Mobile number sign-up	User signs up using a mobile number instead of an email address	The user chooses to enter a mobile number instead of an email	1) Navigate to the Sign-Up UI	First Name = Ali	The system should send an SMS verification code to the mobile number	User is allowed to proceed	Pass	-
TC_008	Successful sign-up with confirmation	User successfully signs up, and a confirmation email is sent	The system has a valid email server for sending confirmations	1) Navigate to the Sign-Up UI	First Name = Ali	The system should display a message: "Account created successfully," and send an email confirmation	"Account created successfully" displayed, email received	Pass	-

TC_009	Mismatched password and confirm password	User enters mismatched password and confirm password fields	Both password fields are visible on the form	1) Navigate to the Sign-Up UI	Password = aliNED2022, Confirm Password = aliNED2023	A warning message: "Passwords do not match" should appear	Warning appeared	Pass	-
TC_010	Duplicate mobile number	User attempts to sign up with an already used mobile number	The mobile number entered is already registered	1) Navigate to the Sign-Up UI	Mobile Number = 03002182764	A warning message: "This mobile number is already in use" should appear	Warning appeared	Pass	-
TC_011	Invalid phone number format	User attempts to sign up with an invalid phone number	Phone number field should allow numeric input only	1) Navigate to the Sign-Up UI	Mobile Number = 12345	A warning message: "Invalid phone number" should appear	Account created	Fail	Phone number validation is not implemented correctly
TC_012	Special characters in name	User enters special characters in the name field	The system allows the user to type in the name field	1) Navigate to the Sign-Up UI	First Name = @li!	A warning message: "Invalid characters in name field" should appear	Account created	Fail	Name field validation is weak
TC_013	Agree to terms and conditions	User signs up without agreeing to terms and conditions	The terms and conditions checkbox should be visible	1) Navigate to the Sign-Up UI	Terms and Conditions = Not Checked	A warning message: "You must agree to terms and conditions" should appear	Warning appeared	Pass	-
TC_014	Email length validation	User enters a very long email address	The system does not have a check for excessively long email addresses	1) Navigate to the Sign-Up UI	Email = samiullah21january123456789anderylongemail@gmail.com	A warning message: "Email is too long" should appear	Account created	Fail	Email length validation is missing
TC_015	Username already taken	User enters an already used username	The username is already taken	1) Navigate to the Sign-Up UI	Username = ali123	A warning message: "Username already taken" should appear	Warning appeared	Pass	-

Facebook Login Page:

Project Name:	Facebook
Module Name:	Login
Reference Document:	N/A
Created by:	Vaiz Hasan
Date of Creation:	15-Jan-25
Reviewed by:	All team members
Date of Review:	20-Jan-25

Test Case ID	Test Scenario	Test Case	Pre-condition	Test Steps	Test Data	Expected Result	Post Condition	Actual Result	Status (Pass/Fail)
TC_LOGIN_001	Valid login with correct credentials	Enter valid email/phone and password	User has a valid Facebook account.	1. Open the Facebook login page. 2. Enter valid email/phone. 3. Enter correct password. 4. Click "Login" button.	<Valid email/phone> and <Valid password>	User successfully login and is redirected to the home page.	User is logged in.	User successfully login and is redirected to the home page.	Pass
TC_LOGIN_002	Invalid email/phone number	Enter an invalid email/phone number and valid password	User has a valid Facebook account.	1. Open the Facebook login page. 2. Enter invalid email/phone. 3. Enter valid password. 4. Click "Login" button.	<Invalid email/phone> and <Valid password>	Error message: "The email or phone number you've entered doesn't match any account."	User is not logged in.	Error message: "The email or phone number you've entered doesn't match any account."	Pass
TC_LOGIN_003	Invalid password	Enter valid email/phone and invalid password	User has a valid Facebook account.	1. Open the Facebook login page. 2. Enter valid email/phone. 3. Enter invalid password. 4. Click "Login" button.	<Valid email/phone> and <Invalid password>	Error message: "The password you've entered is incorrect."	User is not logged in.	Error message: "The password you've entered is incorrect."	Pass
TC_LOGIN_004	Empty email/phone field	Enter email/phone as empty and valid password	User is on the Facebook login page.	1. Open the Facebook login page. 2. Leave the email/phone field blank. 3. Enter valid password. 4. Click "Login" button.	<Blank email/phone>, <valid password>	Error message: "Please enter an email or phone number."	User is not logged in.	Error message: "Please enter an email or phone number."	Pass
TC_LOGIN_005	Empty password field	Enter valid email/phone and leave the password field empty	User is on the Facebook login page.	1. Open the Facebook login page. 2. Enter valid email/phone. 3. Leave the password field blank. 4. Click "Login" button.	<Valid email/phone>, <Blank password>	Error message: "Please enter your password."	User is not logged in.	Error message: "Please enter your password."	Pass
TC_LOGIN_006	Both fields empty	Enter both email/phone and password as empty	User is on the Facebook login page.	1. Open the Facebook login page. 2. Leave both email/phone and password fields blank. 3. Click "Login" button.	<Blank email/phone> and <blank password>	Error message: "Please enter an email or phone number and your password."	User is not logged in.	Error message: "Please enter an email or phone number and your password."	Pass
TC_LOGIN_007	Login with uppercase email/phone	Enter email/phone in uppercase and valid password	User has a valid Facebook account.	1. Open the Facebook login page. 2. Enter email/phone in uppercase (e.g., USER@DOMAIN.COM). 3. Enter valid password. 4. Click "Login" button.	<Uppercase email/phone>, <valid password>	User should be logged in successfully. Facebook should treat email as case-insensitive.	User is logged in.	Error message stating the "Account is not found" because the email with uppercase was not treated case-insensitively.	Fail
TC_LOGIN_008	Login with special characters	Enter email/phone with special characters and valid password	User has a valid Facebook account.	1. Open the Facebook login page. 2. Enter email/phone with special characters (e.g., user@domain.com!) 3. Enter valid password. 4. Click "Login" button.	<Special characters in email/phone>, <Valid password>	User should be logged in successfully.	User is logged in.	User should be logged in successfully.	Pass
TC_LOGIN_009	Password visibility toggle	Enter password and use the visibility toggle	User is on the Facebook login page.	1. Open the Facebook login page. 2. Enter text in the password field. 3. Click the "eye" icon to toggle password visibility.	<Valid email/phone>, <Any password>	Password is displayed when the "eye" icon is clicked, and hidden again when clicked.	Password visibility toggles correctly.	Password is displayed when the "eye" icon is clicked, and hidden again when clicked.	Pass

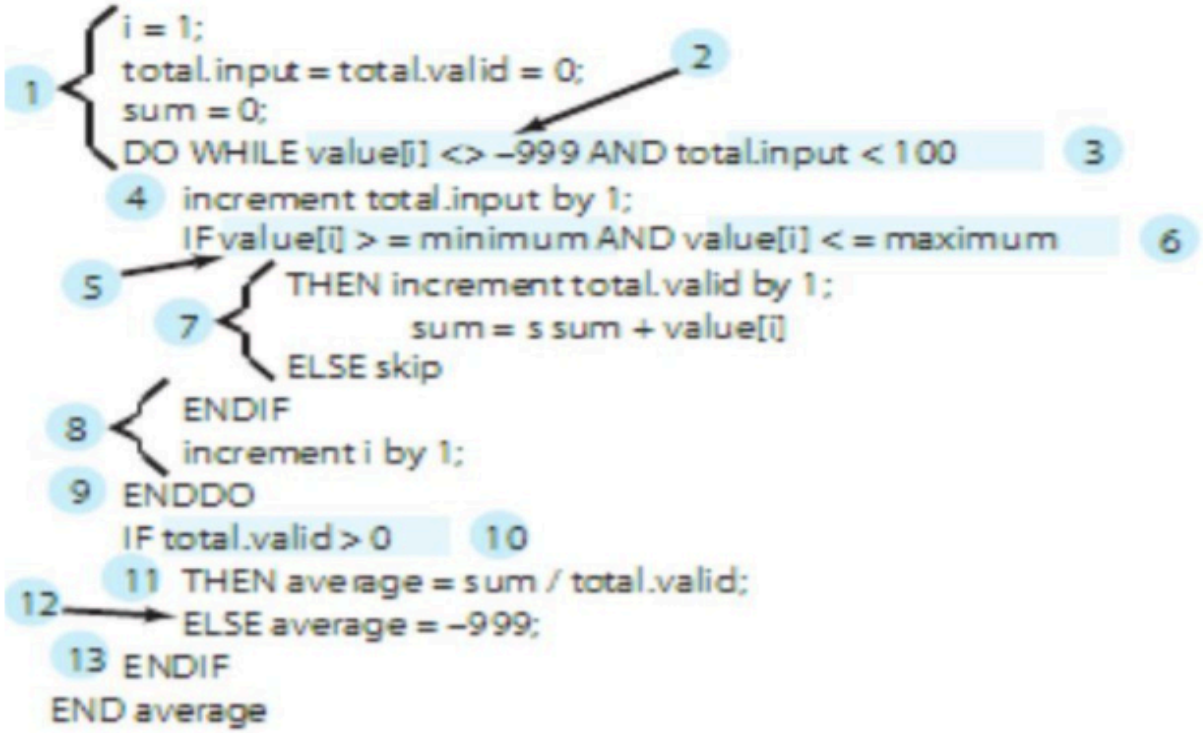
TC_LOGIN_010	Forgotten password link	Enter email/phone for password recovery	User is on the Facebook login page.	1. Open the Facebook login page. 2. Click "Forgotten password?". 3. Enter valid email/phone. 4. Click "Search".	<Valid email/phone>	User is redirected to the password recovery page.	User is on the password recovery page.	User is redirected to the password recovery page.	**Pass**
TC_LOGIN_011	Multiple failed login attempts	Enter incorrect credentials multiple times	User has a valid Facebook account.	1. Open the Facebook login page. 2. Enter incorrect credentials multiple times. 3. Check if system blocks attempts or shows CAPTCHA.	<Invalid credentials multiple times>	System displays CAPTCHA or blocks further login attempts after multiple failed tries.	User is temporarily blocked or CAPTCHA shown.	System displays CAPTCHA or blocks further login attempts after multiple failed tries.	**Pass**
TC_LOGIN_012	Login with incorrect email format	Enter incorrectly formatted email and valid password	User is on the Facebook login page.	1. Open the Facebook login page. 2. Enter an incorrectly formatted email (e.g., user@domain) 3. Enter valid password. 4. Click "Login" button.	<Invalid email format>, <Valid password>	Error message: "Please enter a valid email address."	User is not logged in.	Error message: "Please enter a valid email address."	**Pass**
TC_LOGIN_013	Login page responsiveness	Open login page across different devices to test responsiveness	User is accessing Facebook on different devices	1. Open the Facebook login page on mobile, tablet, and desktop. 2. Ensure the page layout adjusts properly across screen sizes.	N/A	The login page should be responsive and adapt to different screen sizes.	User sees a responsive design on all devices.	The login page should be responsive and adapt to different screen sizes.	**Pass**

TC_LOGIN_014	Login with a non-registered email	Enter a non-registered email/phone and random password	User is on the Facebook login page.	1. Open the Facebook login page. 2. Enter a non-registered email/phone. 3. Enter a random password. 4. Click "Login" button.	<Non-registered email/phone> and <Random password>	Error message: "The email or phone number you've entered doesn't match any account."	User is not logged in.	Error message: "The email or phone number you've entered doesn't match any account."	**Pass**
TC_LOGIN_015	Disabled Login Button for Empty Fields	Verify Login button is disabled when email/phone or password is empty	User is on the Facebook login page	1. Open the Facebook login page. 2. Leave the email/phone or password field empty. 3. Check if the Login button is disabled	<Empty email/phone> and <Empty password>	The Login button should be disabled and not clickable.	User cannot click the Login button.	Login button is enabled, user can click it.	**Fail**

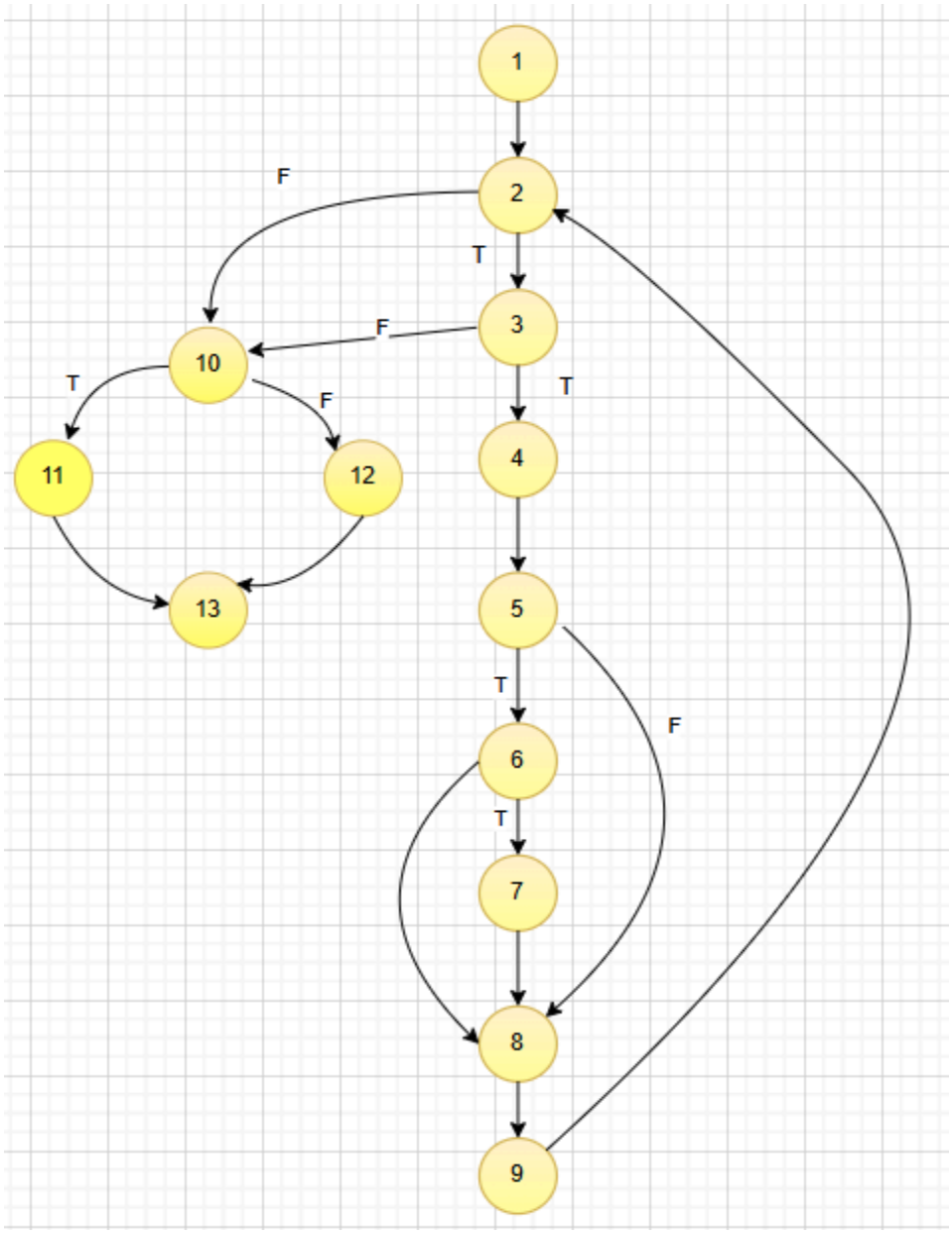
Lab 03: white box Testing

EXERCISE

Q1) Consider the code given below. Calculate cyclomatic complexity and no. of linearly independent paths and Design Testcases for each path. Attach all printout here



CONTROL FLOW GRAPH:



CODE SCREENSHOTS:

1) **When all inputs are within range**

```
#include <iostream>
#include <vector>

using namespace std;
double calculateAverage(const vector<int>& values, int minimum,
    int maximum) {
    int i = 0;
    int totalInput = 0;
    int totalValid = 0;
    int sum = 0;

    while (values[i] != -999 && totalInput < 100) {
        totalInput++;
        if (values[i] >= minimum && values[i] <= maximum) {
            totalValid++;
            sum += values[i];
        }
        i++;
    }
```

```
        i++;
    }
    if (totalValid > 0) {
        return static_cast<double>(sum) / totalValid;
    } else {
        return -999.0;
    }
}
int main() {

    vector<int> values = {10,20,30,40,-999};
    int minimum = 0;
    int maximum = 100;

    double result = calculateAverage(values, minimum, maximum);
    cout << "Average: " << result << endl;

    return 0;
}
```

2) When one input value is within range and other greater than maximum

```
using namespace std;
double calculateAverage(const vector<int>& values, int minimum, int maximum)
    int i = 0;
    int totalInput = 0;
    int totalValid = 0;
    int sum = 0;

    while (values[i] != -999 && totalInput < 100) {
        totalInput++;
        if (values[i] >= minimum && values[i] <= maximum) {
            totalValid++;
            sum += values[i];
        }
        i++;
    }
    if (totalValid > 0) {
        return static_cast<double>(sum) / totalValid;
    } else {
        return -999.0;
    }
}
int main() {

    vector<int> values = {10,120,-999};
    int minimum = 0;
    int maximum = 100;

    double result = calculateAverage(values, minimum, maximum);
    cout << "Average: " << result << endl;

    return 0;
}
```

3) When one input value is within range and other less than minimum

```
using namespace std;
double calculateAverage(const vector<int>& values, int minimum, int maximum) {
    int i = 0;
    int totalInput = 0;
    int totalValid = 0;
    int sum = 0;

    while (values[i] != -999 && totalInput < 100) {
        totalInput++;
        if (values[i] >= minimum && values[i] <= maximum) {
            totalValid++;
            sum += values[i];
        }
        i++;
    }
    if (totalValid > 0) {
        return static_cast<double>(sum) / totalValid;
    } else {
        return -999.0;
    }
}
int main() {

    vector<int> values = {-10,20,-999};
    int minimum = 0;
    int maximum = 100;

    double result = calculateAverage(values, minimum, maximum);
    cout << "Average: " << result << endl;

    return 0;
}
```

4) When one input value is greater than Maximum and other is smaller than minimum

```
using namespace std;
double calculateAverage(const vector<int>& values, int minimum, int maximum) {
    int i = 0;
    int totalInput = 0;
    int totalValid = 0;
    int sum = 0;

    while (values[i] != -999 && totalInput < 100) {
        totalInput++;
        if (values[i] >= minimum && values[i] <= maximum) {
            totalValid++;
            sum += values[i];
        }
        i++;
    }
    if (totalValid > 0) {
        return static_cast<double>(sum) / totalValid;
    } else {
        return -999.0;
    }
}
int main() {

    vector<int> values = {-10,200,-999};
    int minimum = 0;
    int maximum = 100;

    double result = calculateAverage(values, minimum, maximum);
    cout << "Average: " << result << endl;

    return 0;
}
```

5) When no input is given

```
using namespace std;
double calculateAverage(const vector<int>& values, int minimum, int maximum) {
    int i = 0;
    int totalInput = 0;
    int totalValid = 0;
    int sum = 0;

    while (values[i] != -999 && totalInput < 100) {
        totalInput++;
        if (values[i] >= minimum && values[i] <= maximum) {
            totalValid++;
            sum += values[i];
        }
        i++;
    }
    if (totalValid > 0) {
        return static_cast<double>(sum) / totalValid;
    } else {
        return -999.0;
    }
}
int main() {

    vector<int> values = {};
    int minimum = 0;
    int maximum = 100;

    double result = calculateAverage(values, minimum, maximum);
    cout << "Average: " << result << endl;

    return 0;
}
```

6) When all inputs are invalid

```
using namespace std;
double calculateAverage(const vector<int>& values, int minimum, int maximum) {
    int i = 0;
    int totalInput = 0;
    int totalValid = 0;
    int sum = 0;

    while (values[i] != -999 && totalInput < 100) {
        totalInput++;
        if (values[i] >= minimum && values[i] <= maximum) {
            totalValid++;
            sum += values[i];
        }
        i++;
    }
    if (totalValid > 0) {
        return static_cast<double>(sum) / totalValid;
    } else {
        return -999.0;
    }
}
int main() {

    vector<int> values = {120,140,-120,-999};
    int minimum = 0;
    int maximum = 100;

    double result = calculateAverage(values, minimum, maximum);
    cout << "Average: " << result << endl;

    return 0;
}
```

LINEAR INDEPENDENT PATHS:

V=E-N+2

V=17-13+2

V=6

TEST SUMMARY REPORT

Project Name	**Average Calculator**
Test Suite ID	**Calculate SE-22032**
Test Title	**Test the average value on giving mutiple inputs**
Test Priority	**Medium**
Module Name	**DO_AVERAGE**
Designed By	**M.Ali Arslan**
Designed Date	**01/02/25**
Executed By	**M.Ali Arslan**
Executed Date	**02/02/25**
Description of Tes	**To verify the program's output (average) by giving different inputs**
PRE-REQUISITES	
Pre-Condition	**Not required**
Dependencies	**No depedencies in this test**

S.No	Test Case description	Test Steps	Test Data	Expected Result	Actual Output	Status	Comments
Avg_Calculate_01	To verify the program functionality when all values are given within min. and max. range	1->2->3->4->5->6->7->8->9->2->..	{10,20,30,40,-999}	Average:25	Average:25	Pass	
Avg_Calculate_02	To verify the program's functionality when one value is within range and other greater than max	1->2->3->4->5->6->8->9->2->..	{10,120,-999};	Average:10	Average:10	Pass	
Avg_Calculate_03	To verify the program's functionality when one value is within range and other less than the min.	1->2->3->4->5->8->9->2->..	{-10,20,-999};	Average:20	Average:20	Pass	
Avg_Calculate_04	To verify the program's functionality when one value is greater than max and other less than min.	1->2->10->11->13	{-10,200,-999};	Average:-999	Average:-999	Pass	
Avg_Calculate_05	To verify the program's functionality when no input is given	1->2->10->12->13	{ }	Average:-999	Average:-999	Pass	
Avg_Calculate_06	To verify the program's functionality when no valid inputs are given	1->2->3->10->12->13	{120,140,-120,-999};	Average:-999	Average:-999	Pass	

Q3) Write a program (Any Language) to find odd or even number. Design and execute test cases using branch coverage method using template

CODE:

```
void checkOddEven(int number) {
    if (number % 2 == 0) {
        cout << number << " Number is Even." << endl;
    } else {
        cout << number << " Number is Odd." << endl;
    }
}

int main() {
    int num;
    cout << "Enter a number: ";
    cin >> num;
    checkOddEven(num);
    return 0;
}
```

OUTPUTS:

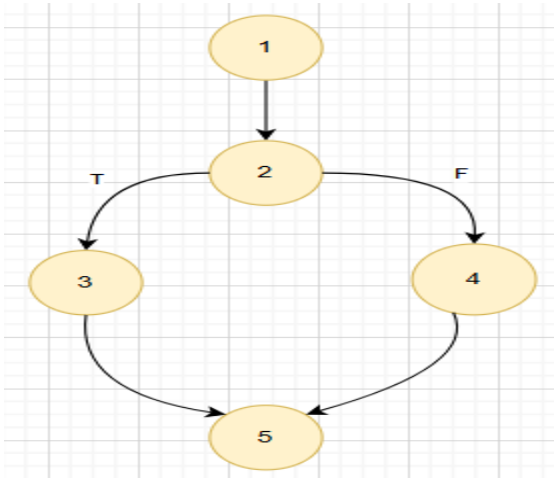
Output

```
Enter a number: 22
22 Number is Even.
```

Output

```
Enter a number: 15
15 Number is Odd.
```

CONTROL FLOW GRAPH:



TEST SUMMARY REPORT:

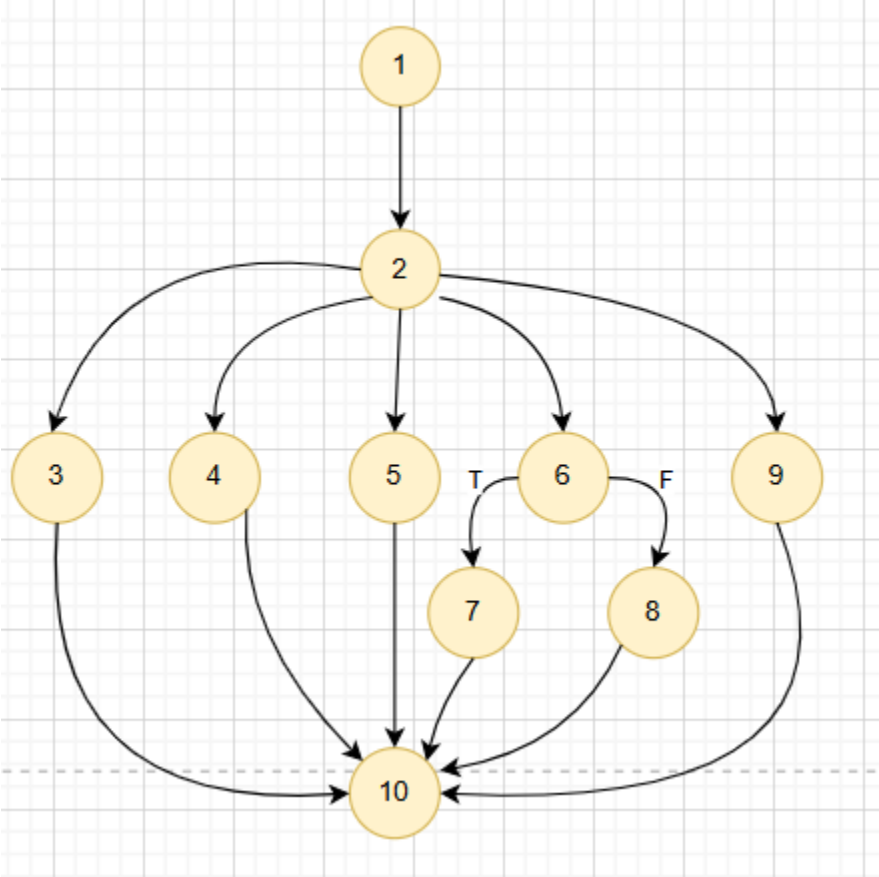
Project Name	Even_Odd_Finder
Test Suite ID	Find_SE-22032
Test Title	Test if the given input is even or odd
Test Priority	Medium
Module Name	Conclude_EvenOdd
Designed By	M.Ali Arslan
Designed Date	01/01/2025
Executed By	M.Ali Arslan
Executed Date	01/01/2025
Description of Test	Verify that the correct decision is made about whether the provided input is even or odd
PRE-REQUISITES	
Pre-Condition	Not required
Dependencies	No dependencies in this test

S.No	Test Case description	Test Steps	Test Data	Expected Output	Actual Output	Status	Comments
Even_Odd check 01	Verify branch coverage by giving even number as input	1->2->3->5	num=22	22 Number is even	Same as expected	Pass	
Even_Odd check 02	Verify branch coverage by giving odd number as input	1->2->4->5	num=15	15 Number is Odd.	Same as expected	Pass	

Q4) Write a program (Any Language) to perform addition, subtraction, multiplication and division. Observe the output and Test it using any one of white box technique

TECHNIQUE: All Path Coverage

CONTROL FLOW GRAPH:



CYCLOMATIC COMPLEXITY:

V(G)= E-N+2

V(G)=14-10+2 = 6

CODE SCREENSHOT:

```
#include <iostream>

using namespace std;

void add(int a, int b) {

    cout << "Result: " << a + b << endl;

}

void subtract(int a, int b) {
```

```
    cout << "Result: " << a - b << endl;
}
void multiply(int a, int b) {
    cout << "Result: " << a * b << endl;
}
void divide(int a, int b) {
    if (b != 0) {
        cout << "Result: " << (double)a / b << endl;
    } else {
        cout << "Error: Cannot divide by zero!" << endl;
    }
}
int main() {
    int choice, num1, num2;
    cout << "Select an operation: \n";
    cout << "1. Addition\n";
    cout << "2. Subtraction\n";
    cout << "3. Multiplication\n";
    cout << "4. Division\n";
    cout << "Enter your choice (1-4): ";
    cin >> choice;
    cout << "Enter the first number: ";
    cin >> num1 ;
    cout<<"Enter second number: ";
    cin>>num2;
    switch (choice) {
        case 1:
```

```
            add(num1, num2);
            break;
        case 2:
            subtract(num1, num2);
            break;
        case 3:
            multiply(num1, num2);
            break;
        case 4:
            divide(num1, num2);
            break;
        default:
            cout << "Invalid choice! Please enter a number between 1 and
4." << endl;
    }
    return 0;
}
```

OUTPUT SCREENSHOTS:

1)

```
Select an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice (1-4): 1
Enter the first number: 23
Enter second number: 66
Result: 89
```

2)

```
Select an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice (1-4): 2
Enter the first number: 89
Enter second number: 23
Result: 66
```

3)

```
Select an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice (1-4): 3
Enter the first number: 23
Enter second number: 66
Result: 1518
```

4)

```
Select an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice (1-4): 4
Enter the first number: 23
Enter second number: 66
Result: 0.348485
```

5)

```
Select an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice (1-4): 4
Enter the first number: 23
Enter second number: 0
ERROR!
Error: Cannot divide by zero!
```

6)

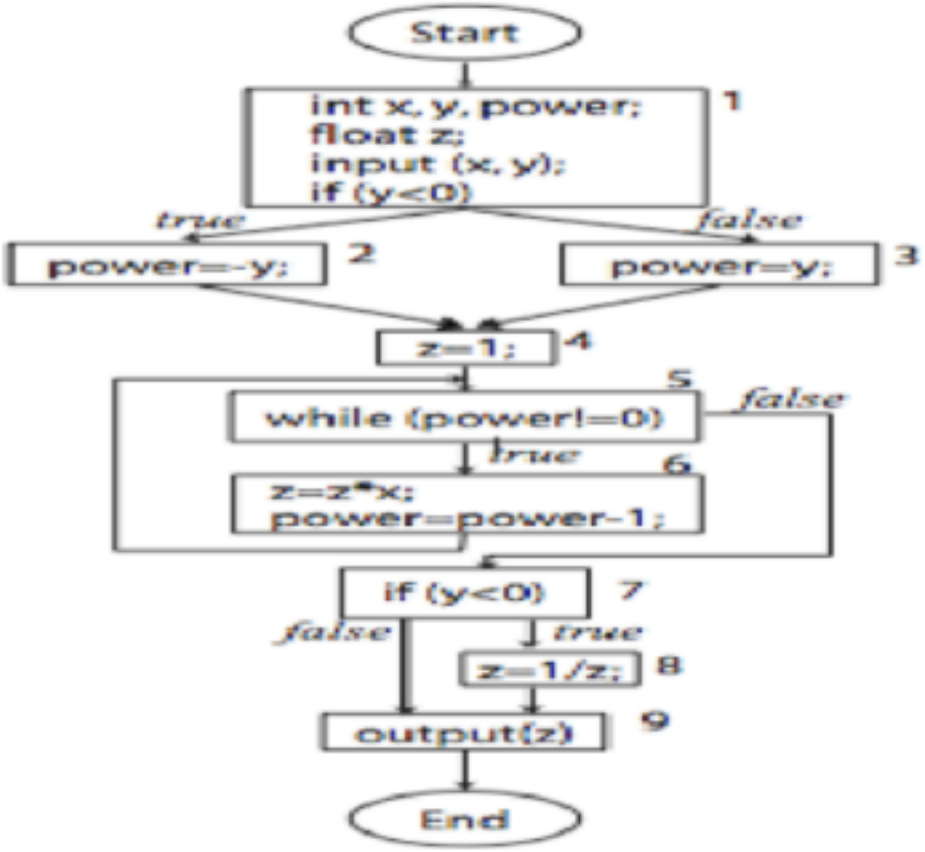
```
Select an operation:
1. Addition
2. Subtraction
3. Multiplication
4. Division
Enter your choice (1-4): 5
Enter the first number: 12
Enter second number: 23
Invalid choice! Please enter a number between 1 and 4.
```

TEST SUMMARY REPORT:

Project Name	Simple Calculator
Test Suite ID	Calculate_SE-22032
Test Title	Test the basic operations of calculator
Test Priority	Medium
Module Name	Do_Calculation
Designed By	M.Ali Arslan
Designed Date	01/01/2025
Executed By	M.Ali Arslan
Executed Date	02/02/2025
Description of Test	Verify that the all calculator operations are producing correct result.
PRE-REQUISITES	
Pre-Condition	Not required
Dependencies	No dependencies in this test

S.No	Test Case description	Test Steps	Test Data	Expected Output	Actual Output	Status	Comments
Calc_Operation_01	To verify the additional functionality of two numbers	**1->2->3->10 Enter values for choice, num1 and num2**	Choice=1, num1=23 ,num2=66	Result=66	Result=66	Pass	
Calc_Operation_02	Two verify the subtraction functionality of two numbers	**1->2->4->10 Enter values for choice, num1 and num2**	Choice=2, num1=89, num2=23	Result=66	Result=66	Pass	
Calc_Operation_03	To verify the mutiplication fucntionality of two numbers	**1->2->5->10 Enter values for choice, num1 and num2**	Choice=3, num1=23, num2=66	Result=1518	Result=1518	Pass	
Calc_Operation_04	To verify the division fucntionality of two numbers	**1->2->6->7->10 Enter values for choice, num1 and num2**	Choice=4, num1=23, num2=66	Result=0.348	Result=0.348	Pass	
Calc_Operation_05	To verify the division by zero functio--nality	**1->2->6->8->10 Enter values for choice, num1 and num2**	Choice=4, num1=23, num2=0	Error msg should be displayed telling users that divsion by 0 is error	**Error:Can't divide by 0**	Pass	
Calc_Operation_06	To verify the invalid choice function--ality	**1->2->9->10 Enter values for choice, num1 and num2**	Choice=5 , num1=12, num2=23	Error message should be displayed telling users to enter correct choice	**Invalid choice! Plz enter a number b/w 1 and 4**	Pass	

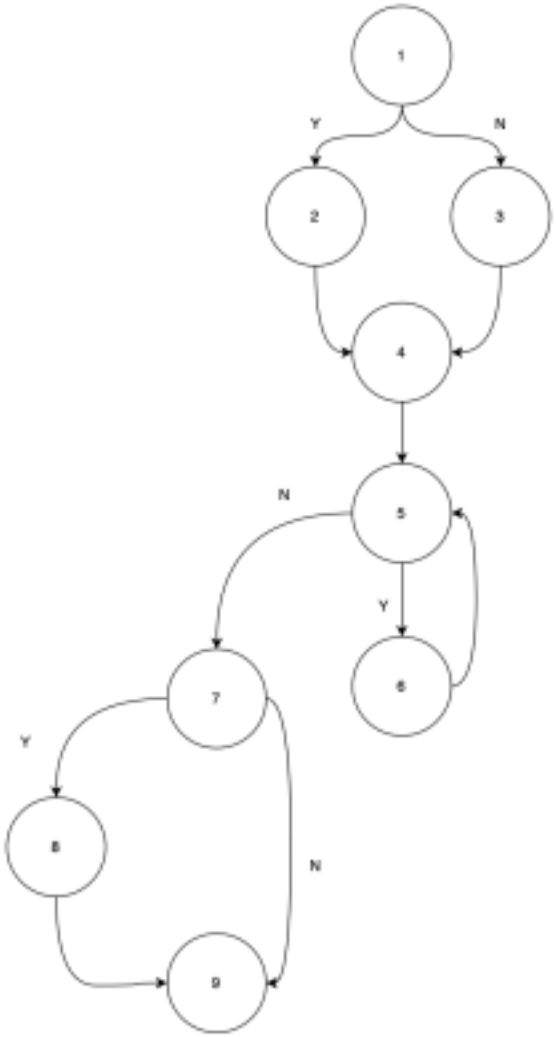
2. Consider the flow chart given below and design test cases by using statement coverage method, AND Independent path method also execute test cases using template (Attach All printout here).



CODE:

```dart
import 'dart:io';

void main() {
  int x, y, power;
  double z;

  // Input x and y
  stdout.write("Enter the base (x): ");
  x = int.parse(stdin.readLineSync()!);

  stdout.write("Enter the exponent (y): ");
  y = int.parse(stdin.readLineSync()!);

  // Determine power based on the value of y
  power = y < 0 ? -y : y;

  z = 1;

  // Calculate x^y
  while (power != 0) {
    z *= x;
    power--;
  }

  // Handle negative exponents
  if (y < 0) {
    z = 1 / z;
  }

  // Output the result
  print("Result: $z");
}
```

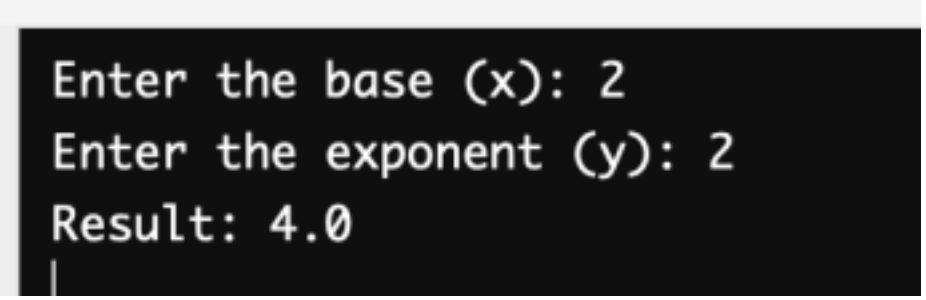
CFG:



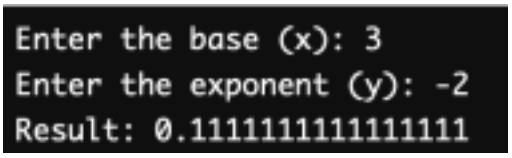
StatementCoverage:

OUTPUT SCREENSHOT:

1:



2:



TEST SUMMARY REPORT:

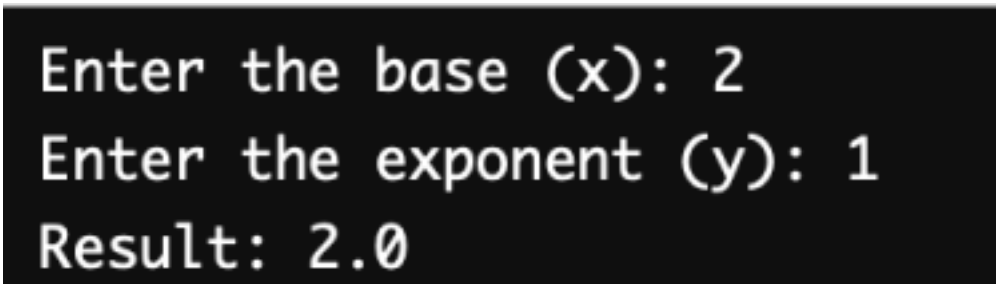
S-NO	TEST CASE	TEST STEPS	TEST DATA	Expected Output	Actual Output	Status
TC-01	Positive exponention	1-Enter x value	x=2,y=2	z=4	z=4	pass
		2-Enter y value				
TC-02	Negative exponention	1-Enter x value	x=3,y= -2	z=0.111	z=0.111	pass
		2-Enter y value				

LINEAR INDEPENDENT PATH:

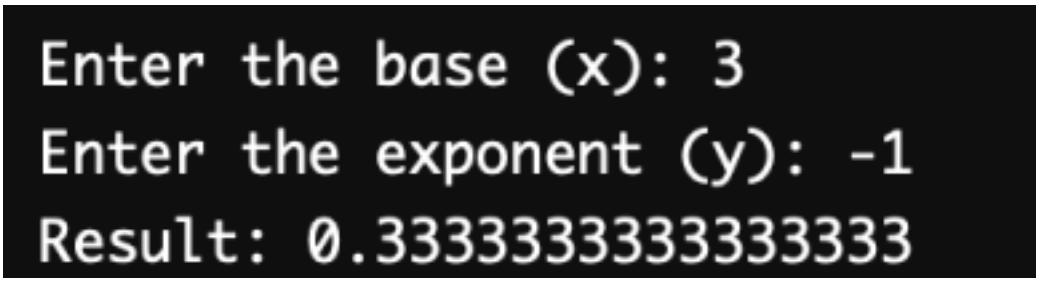
V(G)= E-N+2= 11-9+2 = 4

OUTPUT SCREENSHOT:

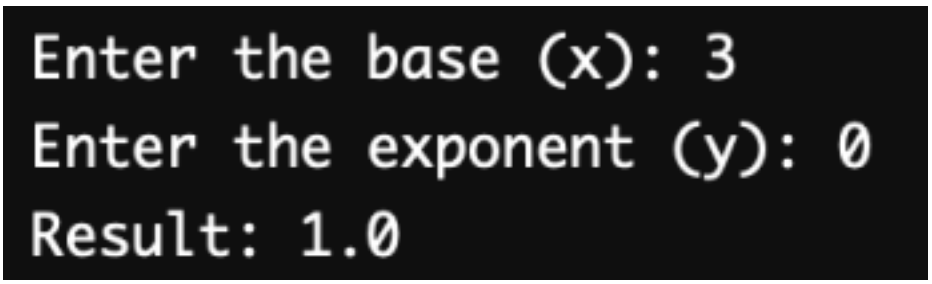
1



2



3



TEST SUMMARY REPORT:

S-NO	PATH	TEST DATA	Expected Output	Actual Output	Status
TC-01	1-3-4-5-6-5-7-9	x=2,y=1	z=2	z=2	pass
TC-02	1-2-4-5-6-5-7-8-9	x=3,y=-1	z=0.333	z=0.333	pass
TC-03	1-3-4-5-7-9	x=3,y=0	z=1	z=1	pass
TC-04	1-3-4-5-6-5-7-8-9	-	-	-	

Lab 04a: Equivalence Class Partitioning(ECP)

Q1) EXERCISE:

Consider Triangle problem : Design and develop a program in a language of your choice to solve the triangle problem defined as follows . Accept three integers which are supposed to be the three sides of triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all.

Derive test cases for your program based on ECP approach, execute the test cases and discuss the results

Pre-condition : $1 \leq a \leq 10$, $1 \leq b \leq 10$ and $1 \leq c \leq 10$ and $a < b + c$, $b < a + c$ and $c < a + b$

Brief Description : Check whether given value for a Equilateral, Isosceles , Scalene triangle or can't form a triangle

CODE SCREENSHOT:

```
function determineTriangleType(a, b, c) {

    if (a < 1 || a > 10 || b < 1 || b > 10 || c < 1 || c > 10) {
        return "Invalid input: sides should be between 1 and 10";
    }
    if (a >= b + c || b >= a + c || c >= a + b) {
        return "Not a Triangle";
    }
    if (a === b && b === c) {
        return "Equilateral Triangle";
    } else if (a === b || b === c || a === c) {
        return "Isosceles Triangle";
    } else {
        return "Scalene Triangle";
    }
}
const a = parseInt(prompt("Enter side a: "));
const b = parseInt(prompt("Enter side b: "));
const c = parseInt(prompt("Enter side c: "));
console.log(determineTriangleType(a, b, c));
```

CODE OUTPUTS:

Execution of TC_001

```
Output

Enter side a: 5
Enter side b: 5
Enter side c: 5
Equilateral Triangle
```

Execution of TC_002

```
Output

Enter side a: 3
Enter side b: 3
Enter side c: 5
Isosceles Triangle
```

Execution of TC_003

```
Output

Enter side a: 3
Enter side b: 5
Enter side c: 3
Isosceles Triangle
```

Execution of TC_004

```
Output

Enter side a: 4
Enter side b: 5
Enter side c: 5
Isosceles Triangle
```

Execution of TC_005

```
Output

Enter side a: 5
Enter side b: 6
Enter side c: 7
Scalene Triangle
```

Execution of TC_006

```
Output

Enter side a: 6
Enter side b: 2
Enter side c: 2
Not a Triangle
```

Execution of TC_007

```
Output

Enter side a: 2
Enter side b: 8
Enter side c: 5
Not a Triangle
```

Execution of TC_008

```
Output

Enter side a: 2
Enter side b: 2
Enter side c: 6
Not a Triangle
```

Execution of TC_009

Output

```
Enter side a: 0
Enter side b: 5
Enter side c: 6
Invalid input: sides should be between 1 and 10
```

Execution of TC_010

Output

```
Enter side a: 12
Enter side b: 5
Enter side c: 6
Invalid input: sides should be between 1 and 10
```

Execution of TC_011

Output

```
Enter side a: 5
Enter side b: 0
Enter side c: 6
Invalid input: sides should be between 1 and 10
```

Execution of TC_012

```
Enter side a: 5
Enter side b: 12
Enter side c: 6
Invalid input: sides should be between 1 and 10
```

Execution of TC_013

Output

```
Enter side a: 8
Enter side b: 6
Enter side c: 0
Invalid input: sides should be between 1 and 10
```

Execution of TC_014

```
Output

Enter side a: 6
Enter side b: 4
Enter side c: 12
Invalid input: sides should be between 1 and 10
```

TEST SUMMARY REPORT

Project Name	**Triangle Problem Decision (Equivalence Partitioning)**
Test ID	Solve_Triangle_EP_TESE32
Test Title	Use Equivalence partitioning approach for triangle problem
Test Priority	Low
Module Name	Identify Traingle
Test Data	Enter the 3 Integer Value (a, b and c)
Designed By	M.Ali Arslan
Designed Date	18/02/2025
Executed By	M.Ali Arslan
Executed Date	19/02/2025
Description of Test	Check whether the given values form an equilateral, isosceles, scalene triangle or no triangle using equivalence Partitioning approach.
PREREQUISITES	This test has no pre-requisites
Pre-Conditions	Range in which testing is to be done: $1 \leq a \leq 10$, $1 \leq b \leq 10$ and $1 \leq c \leq 10$ and $a < b + c$, $b < a + c$ and $c < a + b$
Dependencies	No dependencies in this test.

CLASS PARTITIONS	TEST CASE ID
$1 \le a \le 10$, $1 \le b \le 10$, $1 \le c \le 10$.	TC_001 to TC_008
a<1	TC_009
a>10	TC_010
b<0	TC_011
b>10	TC_012
c<0	TC_013
c>10	TC_014

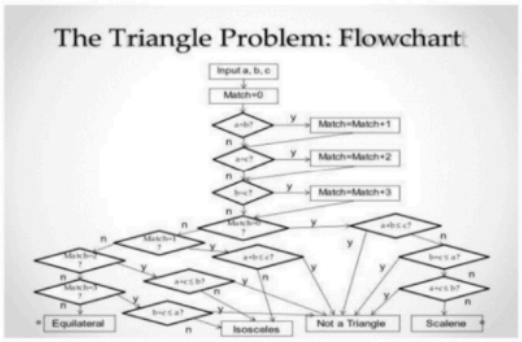
Test Case ID	Test Steps	Test Data	Expected Result	Actual Result	Status
TC_001	1. Run the program	a=5,b=5, c=5	Equilateral Triangle	Equilateral Triangle	Pass
	2. Enter values for a,b & c				
TC_002	1. Run the program	a=3,b=3, c=5	Isosceles Triangle	Isosceles Triangle	Pass
	2. Enter values for a,b & c				
TC_003	1. Run the program	a=3,b=5,c=3	Isosceles Triangle	Isosceles Triangle	Pass
	2. Enter values for a,b & c				
TC_004	1. Run the program	a=4,b=5,c=5	Isosceles Triangle	Isosceles Triangle	Pass
	2. Enter values for a,b & c				
TC_005	1. Run the program	a=5, b=6 , c=7	Scalene Triangle	Scalene Triangle	Pass
	2. Enter values for a,b and c				
TC_006	1. Run the program	a=6, b=2, c=2	Not a Triangle	Not a Triangle	Pass
	2. Enter values of a,b and c				
TC_007	1. Run the program	a=2, b=8 , c=5	Not a Triangle	Not a Triangle	Pass
	2. Enter values of a,b and c				
TC_008	1. Run the program	a=2, b=2, c=6	Not a triangle	Not a triangle	Pass
	2. Enter values of a,b and c				

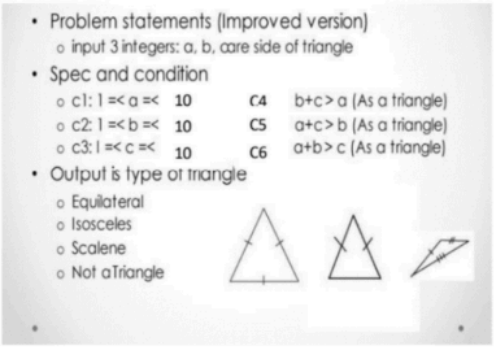
TC_009	1. Run the program	a=0, b=5, c=6	Invalid Input	Invalid Input	Pass
	2. Enter values of a,b and c				
TC_010	1. Run the program	a=12, b=,5 c=6	Invalid Input	Invalid Input	Pass
	2. Enter values of a,b and c				
TC_011	1. Run the program	a=5, b-0, c=6	Invalid input	Invalid Input	Pass
	2. Enter values of a,b and c				
TC_012	1. Run the program	a-5,b=12,c=6	Invalid input	Invalid input	Pass
	2. Enter values of a,b and c				
TC_013	1. Run the program	a=8, b=6 , c=0	Invalid input	Invalid input	Pass
	2. Enter values of a,b and c				
TC_014	1. Run the program	a=6, b=4 , c=12	Invalid input	Invalid input	Pass
	2. Enter values of a,b and c				

Lab 04b: Write Test Cases Using Boundary Value Analysis

EXERCISE SOLUTIONS:

Recall the triangle problem and write test cases using BVA method using template:





Use test summary report template //attach all screenshots here.

Triangle Problem Code in Python

main.py

```python
def classify_triangle(a, b, c):
    if not (1 <= a <= 10 and 1 <= b <= 10 and 1 <= c <= 10):
        return "Invalid input"

    if b + c <= a or a + c <= b or a + b <= c:
        return "not a triangle"

    if a == b == c:
        return "equilateral"
    elif a == b or b == c or a == c:
        return "isosceles"
    else:
        return "scalene"

# Taking input
a, b, c = map(int, input("Enter three side lengths: ").split())
print(classify_triangle(a, b, c))
```

Project Name	Triangle Problem BVA
Test ID	TP_BVA_G_08
Test Title	Using BVA for triangle problem
Test Priority	Low
Module Name	Categorize Triangle
Test Data	Enter 3 integers
Designed By	Faez Ansar
Designed Date	Feb-19,2025
Executed By	Faez Ansar
Executed Date	Feb-19,2025
Description of Test	Check if given values are valid ,and categorize them as scalene,isoceles or equilateral using BVA method
PRE-Requisites	-
Pre-Conditions	1 <= a <= 10 and 1 <= b <= 10 and 1 <= c <= 10 and b + c > a and a + c > b and a + b > c
Dependencies	-

Test Case ID	Test Data(a,b,c)	Expected Output	Actual Output	Status
TC01	(3, 3, 3)	equilateral	equilateral	Pass
TC02	(5, 5, 3)	isosceles	isosceles	Pass
TC03	(6, 7, 8)	scalene	scalene	Pass
TC04	(2, 2, 4)	not a triangle	not a triangle	Pass
TC05	(1, 1, 2)	not a triangle	not a triangle	Pass
TC06	(10, 10, 10)	equilateral	equilateral	Pass
TC07	(7, 7, 5)	isosceles	isosceles	Pass
TC08	(4, 6, 9)	scalene	scalene	Pass
TC09	(2, 3, 10)	not a triangle	not a triangle	Pass
TC10	(8, 9, 1)	not a triangle	not a triangle	Pass
TC11	(3, 4, 5)	scalene	scalene	Pass
TC12	(10, 10, 5)	isosceles	isosceles	Pass

Test Cases Execution:

TC01
Input: (3, 3, 3)
Result: equilateral

TC02
Input: (5, 5, 3)
Result: isosceles

TC03
Input: (6, 7, 8)
Result: scalene

TC04
Input: (2, 2, 4)
Result: not a triangle

TC05
Input: (1, 1, 2)
Result: not a triangle

TC06
Input: (10, 10, 10)
Result: equilateral

TC07
Input: (7, 7, 5)
Result: isosceles

TC08
Input: (4, 6, 9)
Result: scalene

TC09
Input: (2, 3, 10)
Result: not a triangle

TC10
Input: (8, 9, 1)
Result: not a triangle

TC11
Input: (3, 4, 5)
Result: scalene

TC12
Input: (10, 10, 5)
Result: isosceles

Lab 05: Black Box Testing: Decision Table

1.Consider Triangle problem : Design and develop a program in a language of your choice to solve the triangle problem defined as follows : Accept three integers which are supposed to be the three sides of triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all. Derive test cases for your program based on decision-table approach, execute the test cases and discuss the results

ii. Make initial decision table consisting of columns (using the formula 2^conditions)

iii. Reduced the table to remove redundant/impossible scenarios.

iv. write test cases for reduced decision table v. execute code for triangle problem and execute test cases (iii)

i) Code

```
def triangle_type(a, b, c):
    if a <= 0 or b <= 0 or c <= 0:
        return "Invalid input: Sides must be positive integers"

    if a + b <= c or a + c <= b or b + c <= a:
        return "Not a Triangle"
    elif a == b == c:
        return "Equilateral Triangle"
    elif a == b or a == c or b == c:
        return "Isosceles Triangle"
    else:
        return "Scalene Triangle"

try:
    a = int(input("Enter side a: "))
    b = int(input("Enter side b: "))
    c = int(input("Enter side c: "))

    print("Triangle Type:", triangle_type(a, b, c))
except ValueError:
    print("Invalid input: Please enter integer values")
```

ii) (2^6=64)

Conditions:

a<b+ c , b<a + c, c<a+ b, a = b, a = c, b = c represented by C1,C2,C3,C4,C5,C6

Actions:

Not a Triangle, Scalene, Isosceles, Equilateral, Impossible represented by A1,A2,A3,A4,A5

Initial Table

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26	R27	R28	R29	R30	R31	R32
C1	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
C2	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
C3	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
C4	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
C5	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
C6	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F
A1									YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES
A2								YES																								
A3				YES		YES	YES																									
A4	YES																															
A5		YES	YES		YES																											

R33	R34	R35	R36	R37	R38	R39	R40	R41	R42	R43	R44	R45	R46	R47	R48	R49	R50	R51	R52	R53	R54	R55	R56	R57	R58	R59	R60	R61	R62	R63	R64
F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F
YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES

iii) Reduced the table to remove redundant/impossible scenarios.

	R1	R2	R3	R4	R5	R6	R7	R8	R9-R16	R17-32	R33-64
C1	T	T	T	T	T	T	T	T	T	T	F
C2	T	T	T	T	T	T	T	T	T	F	-
C3	T	T	T	T	T	T	T	T	F	-	-
C4	T	T	T	T	F	F	F	F	-	-	-
C5	T	T	F	F	T	T	F	F	-	-	-
C6	T	F	T	F	T	F	T	F	-	-	-
Not a triangle									YES	YES	YES
Scalene								YES			
Isosceles				YES		YES	YES				
Equilateral	YES										
Impossible		YES	YES		YES						

iv) write test cases for reduced decision table

s.no	Test Case	Test Steps	Test Data (a, b, c)	Expected Output	Actual Output	Pass/Fail	Comments
1	Verify Not a Triangle	Input (1,2,3)	(1,2,3)	Not a Triangle	Not a Triangle	Pass	-
2	Verify Not a Triangle	Input (12,4,7)	(12,4,7)	Not a Triangle	Not a Triangle	Pass	
3	Verify Not a Triangle	Input(5,6,15)	(5,6,15)	Not a Triangle	Not a Triangle	Pass	
4	Verify Equilateral Triangle	Input (3,3,3)	(3,3,3)	Equilateral Triangle	Equilateral Triangle	Pass	-
5	Verify Isosceles Triangle	Input (3,3,5)	(3,3,5)	Isosceles Triangle	Isosceles Triangle	Pass	-
6	Verify Scalene Triangle	Input (5,7,10)	(5,7,10)	Scalene Triangle	Scalene Triangle	Pass	-
7	Verify Invalid Input	Input (-3,4,5)	(-3,4,5)	Impossible	Impossible	Pass	Negative value

v)

```
Enter side a: 1
Enter side b: 2
Enter side c: 3
Triangle Type: Not a Triangle
```

```
Enter side a: 12
Enter side b: 4
Enter side c: 7
Triangle Type: Not a Triangle
```

```
Enter side a: 5
Enter side b: 6
Enter side c: 15
Triangle Type: Not a Triangle
```

```
Enter side a: 3
Enter side b: 3
Enter side c: 3
Triangle Type: Equilateral Triangle
```

```
Enter side a: 3
Enter side b: 3
Enter side c: 5
Triangle Type: Isosceles Triangle
```

```
Enter side a: 5
Enter side b: 7
Enter side c: 10
Triangle Type: Scalene Triangle
```

```
Enter side a: -3
Enter side b: 4
Enter side c: 5
Triangle Type: Invalid input: Sides must be positive integers
```

Testing Using Python unittest

triangle.py > ...

```python
def triangle_type(a, b, c):
    if a <= 0 or b <= 0 or c <= 0:
        return "Invalid input: Sides must be positive integers"

    if a + b <= c or a + c <= b or b + c <= a:
        return "Not a Triangle"
    elif a == b == c:
        return "Equilateral Triangle"
    elif a == b or a == c or b == c:
        return "Isosceles Triangle"
    else:
        return "Scalene Triangle"
```

```python
import unittest
from triangle import triangle_type

class TestTriangleType(unittest.TestCase):
    def test_equilateral_triangle(self):
        self.assertEqual(triangle_type(3, 3, 3), "Equilateral Triangle")

    def test_isosceles_triangle(self):
        self.assertEqual(triangle_type(3, 3, 5), "Isosceles Triangle")

    def test_scalene_triangle(self):
        self.assertEqual(triangle_type(5, 7, 10), "Scalene Triangle")

    def test_not_a_triangle(self):
        self.assertEqual(triangle_type(1, 2, 3), "Not a Triangle")
        self.assertEqual(triangle_type(12, 4, 7), "Not a Triangle")
        self.assertEqual(triangle_type(5, 6, 15), "Not a Triangle")

    def test_invalid_input(self):
        self.assertEqual(triangle_type(-3, 4, 5), "Invalid input: Sides must be positive integers")

if __name__ == "__main__":
    unittest.main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Warning: PowerShell detected that you might be using a screen reader and has disabled PS
ReadLine for compatibility purposes. If you want to re-enable it, run 'Import-Module PSR
eadLine'.

PS C:\Users\786COMPUTERS\Desktop\Testing with Python> python -u "c:\Users\786COMPUTERS\D
esktop\Testing with Python\test_triangle.py"
.....
----------------------------------------------------------------------
Ran 5 tests in 0.002s

OK
PS C:\Users\786COMPUTERS\Desktop\Testing with Python>
```

Q2. Consider a library management system(LMS) which issues book(s) to a registered user by checking the outstanding fee i.e. if a registered user have no pending fee then requested book may be issued(if and only if the potential user has under borrow limit)

i. Make initial decision table consisting of columns (using the formula 2^conditions)
ii. Reduced the table to remove redundant/impossible scenarios.
iii. write test cases for reduced decision table
iv. execute code for LMS problem and execute test cases (iii)

Design and develop a program in a language of your choice to develop above mentioned problem and attached all printouts /screenshots/summary report here

SOLUTION:

**** Conditions:***
Condition 1: Is user registered?
Condition2: User has no pending fees?
Condition3: Under borrow limit?

**** Actions:***
Action1: Yes, book can be issued
Action2: No, book can't be issued

i) Initial Descision Table:
2^3 = 8 columns

Condition	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
C1: Is User registered?	T	T	T	T	F	F	F	F
C2: User has no outstanding fees?	T	T	F	F	T	T	F	F
C3: Under borrow limit?	T	F	T	F	T	F	T	F
Action1: Yes, book can be issued	☑							
Action2: No book can't be isssued		☑	☑	☑	☑	☑	☑	☑

ii) Reduced Table

Condition	Rule 1	Rule 2	Rule 3 - Rule 4	Rule 5 - Rule 8
C1: Is User registered?	T	T	T	F
C2: User has no outstanding fees?	T	T	F	-
C3: Under borrow limit?	T	F	-	-
Action1: Yes, book can be issued	✓			
Action2: No book can't be isssued		✓	✓	✓

iii) write test cases for reduced decision table

TEST SUMMARY REPORT:

Project Name	**Library Management System**
Test ID	LMS_TESE32
Test Title	Use decision table approach to test book issuance functionality
Test Priority	Medium
Module Name	Issue_book
Test Data	Enter user status(registered or unregistered), enter pending fees and enter number of books already borrowed by user
Designed By	M.Ali Arslan
Designed Date	18/02/2025
Executed By	M.Ali Arslan
Executed Date	19/02/2025
Description of Test	Check whether book can be issued to a library user
PREREQUISITES	
Pre-Conditions	No pre-conditions for this test
Dependencies	No dependencies in this test.

Test Case ID	Conditions	Test Data	Expected Outcome	Actual Outcome	Status
TC_01	1. User should be registered	isRegistered=yes	Book can be issued.	Book can be issued.	Pass
	2. No outstanding fees	hasPendingFees=0			
	3. Should be under borrow limit	borrowedBooks=3			
TC_02	1. User should be registered	isRegistered=yes	Book cannot be issued.	Book cannot be issued.	Pass
	2. No outstanding fees	hasPendingFees=0			
	3. Should be under borrow limit	borrowedBooks=6			
TC_03	1. User should be registered	isRegistered=yes	Book cannot be issued.	Book cannot be issued.	Pass
	2. No outstanding fees	hasPendingFees=50			
	3. Should be under borrow limit	borrowedBooks=3			
TC_04	1. User should be registered	isRegistered=no	Book cannot be issued.	Book cannot be issued.	Pass
	2. No outstanding fees	hasPendingFees=0			
	3. Should be under borrow limit	borrowedBooks=2			

iv) execute code for LMS problem and execute test cases (iii)

CODE:

```dart
import 'dart:io';
void main() {
  stdout.write("Is the user registered? (yes/no): ");
  String? isRegistered = stdin.readLineSync()?.toLowerCase();

  stdout.write("Enter the pending fees?: ");
 int? hasPendingFees = int.parse(stdin.readLineSync()!);

  stdout.write("How many books has the user borrowed? ");
  int borrowedBooks = int.tryParse(stdin.readLineSync() ?? '0') ?? 0;

  if (isRegistered == 'yes' && hasPendingFees ==0  && borrowedBooks < 5) {
    print("Book can be issued.");
  } else {
    print("Book cannot be issued.");
  }
}
```

Execution of TC_01:

```
Is the user registered? (yes/no): yes
Enter the pending fees?: 0
How many books has the user borrowed? 3
Book can be issued.
```

Execution of TC_02:

```
Is the user registered? (yes/no): yes
Enter the pending fees?: 0
How many books has the user borrowed? 6
Book cannot be issued.
```

Execution of TC_03:

```
Is the user registered? (yes/no): yes
Enter the pending fees?: 50
How many books has the user borrowed? 3
Book cannot be issued.
```

Execution of TC_04:

```
Is the user registered? (yes/no): no
Enter the pending fees?: 0
How many books has the user borrowed? 2
Book cannot be issued.
```

OEL Rubrics

Rubric	Testing Approach	Criteria	2 Marks (Excellent)	1 Mark (Satisfactory)	0 Marks (Poor/Not Attempted)
Rubric 1: Test Case Quality (Black Box)	Black Box	Functional test cases based on requirements	Test cases are comprehensive, clearly aligned with functional requirements, covering both typical and edge cases.	Test cases address core functionality, but may lack edge case coverage or completeness.	Test cases are vague, incomplete, or not aligned with requirements.
Rubric 2: Input-Output Validation (Black Box)	Black Box	Input variation and output verification	Demonstrates strong input coverage (valid, invalid, boundary) with clear expected outputs for each scenario.	Uses some variation in input types, but expected outputs are generic or partially defined.	Inputs lack variation or outputs are missing/inaccurate.
Rubric 3: Code Path Testing (White Box)	White Box	Statement/ branch/path coverage	Effectively applies techniques like statement, decision, or path coverage with evidence (e.g., coverage report or annotated code).	Applies basic white box strategy (e.g., statement coverage) with partial evidence.	No white box strategy applied or lacks demonstration.
Rubric 4: Logical Condition Testing (White Box)	White Box	Testing of loops, conditions, and logic paths	Test cases explicitly target logic structures (e.g., if-else, loops) with multiple test paths shown.	Limited attention to logic conditions; test paths are few or repetitive.	No focus on logic testing or paths.
Rubric 5: Documentation & Justification	Both	Clarity of approach and rationale	Provides clear documentation of testing approach, explains why selected methods were used, and shows understanding of both white and black box techniques.	Some documentation provided with limited rationale or unclear distinctions between methods.	No documentation or rationale provided.

Lab 06 (a)

Exercise: Select your previous project and Design Test cases for at least 5 components using any of white box strategy also answer the following questions

i. Draw a comparison of selected techniques with all other ones.
ii. Why you choose this technique?
iii. On what level we can apply white box and when we should stop software testing

Attached all printouts and screenshots of application [code+ application o/p on test data] here. Also attached summary report and calculate code coverage of each test case.

i. Draw a comparison of selected techniques with all other white-box techniques

Technique	Application in Project	Pros	Cons
Statement Coverage	Executes every line in `load_pdf_file()` and `text_split()`	Verifies code is reachable; easy to measure	May miss missing decision logic or exception paths
Branch Coverage	Tests both true/false outcomes in ifs and exceptions	Catches Boolean decision errors	Doesn't cover all paths through combined conditions
Path Coverage	All possible paths through loops and branches	Exhaustive	Impractical for functions with many branches
Linear Independent Paths	Minimal set of independent paths based on cyclomatic complexity	Balances thoroughness and test count	Requires manual identification of paths

ii. **Why choose Basis Path (Linear Independent Paths)?**

- Cyclomatic complexity for our functions is low (≤3), so enumerating each independent path fully covers decision logic, loop execution, and exceptions without an explosion of test cases.

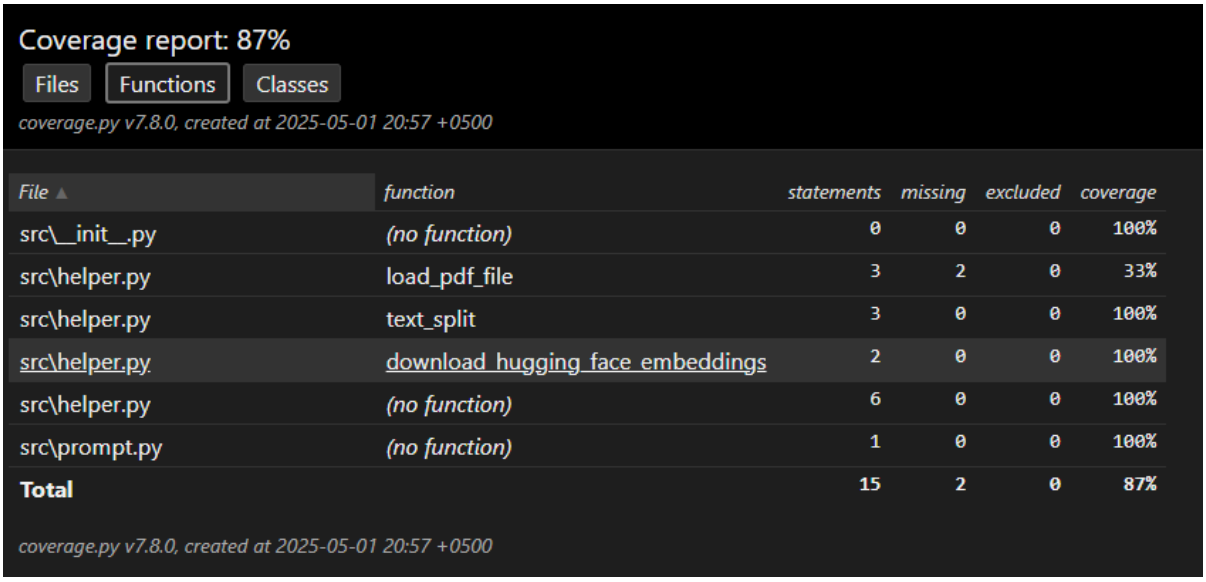
iii. **On what level to apply white-box and when to stop testing?**

- Apply at **unit/module level** for each function.
- Stop when all independent paths (per function) are covered—i.e., 100% of the calculated cyclomatic paths.

Test cases codes:

```
import os import
pytest
from langchain.schema import Document
from src.helper import load_pdf_file, text_split

@pytest.fixture(autouse=True) def
patch_loaders(monkeypatch):
    class DummyPDFLoader:
        def _init_(self, path, **kwargs): self.path = path def
        load(self): return [Document(page_content="dummy",
metadata={"source":self.path})]

    class DummyDirLoader:
        def _init_(self, dirpath): self.dirpath = dirpath def
        load(self):
            docs = []
            for fname in os.listdir(self.dirpath): if
                fname.endswith(".pdf"):
                    docs.extend(DummyPDFLoader(os.path.join(self.dirpath,
fname)).load())
            return docs monkeypatch.setattr("src.helper.PyPDFLoader",
    DummyPDFLoader) monkeypatch.setattr("src.helper.DirectoryLoader",
    DummyDirLoader)

# Test Case 1: load_pdf_file() – Empty Directory def
test_load_pdf_empty(tmp_path):
    empty = tmp_path / "empty"; empty.mkdir() assert
    load_pdf_file(str(empty)) == []

# Test Case 2: load_pdf_file() – Mixed Contents def
test_load_pdf_mixed(tmp_path):
    mixed = tmp_path / "mixed"; mixed.mkdir()
    (mixed / "a.pdf").write_text("ignored")
    (mixed / "b.txt").write_text("ignored") docs
    = load_pdf_file(str(mixed))
    assert len(docs) == 1
    assert docs[0].metadata["source"].endswith("a.pdf")

# Test Case 3: text_split() – Overlap Logic def
test_text_split_overlap():
    text = " ".join(f"w{i}" for i in range(600)) docs =
    [Document(page_content=text)]
    chunks = text_split(docs)
    assert len(chunks) == 2
    overlap_end   = chunks[0].page_content.split()[-20:]
    overlap_start = chunks[1].page_content.split()[:20]
    assert overlap_end == overlap_start
```

Code coverage report:



Test summary report:

S.no	Test Case	Test Steps	Test Data	Expected Output	Actual Output	Pass/Fail	Comments
Tc_001	Test load_pdf_file() empty dir	1. Create empty dir2. Call load_pdf_file()	Empty directory	Returns empty list []	[]	Pass	—
Tc_002	Test load_pdf_file() mixed	1. Create dir with .pdf and .txt2. Call load_pdf_file()	1 PDF + 1 non-PDF	Returns only 1 PDF document	1 document	Pass	Filters non-PDFs
Tc_003	Test text_split() overlap	1. Create a 600-word Document2. Call text_split()	600 words	2 chunks with 20-word overlap	2 chunks with overlap	Pass	Validates correct overlap logic

Lab 06 (b)

Exercise: Select your previous project and Design Test cases for at least 5 features using any of black box strategy [you can choose different techniques to test different features] also answer the following questions

i. **Draw a comparison of selected techniques with all other ones.**
ii. **ii. Why you choose this technique?**
iii. **iii. On what level we can apply black box and why?**

Attached all printouts and screenshots of application [UI+ application o/p on test data] here. Also attached summary report.

Features Selected (5):

- CLI PDF upload & index trigger
- Chat UI load (GET /)
- Chat query handling (POST /get)
- Similarity threshold + k parameter combinations
- Error message when .env is missing

i. **Draw a comparison of selected techniques with all other black-box techniques**

Technique	**Feature Context**	**Pros**	**Cons**
Equivalence Partitioning	CLI PDF upload paths (valid vs missing vs empty)	Covers representative input partitions with few tests	May miss boundary errors
Boundary Value Analysis	User query length limits (0, max, max+1)	Targets edge conditions where failures often occur	Doesn't cover non-length-related errors
Decision Table Testing	Similarity threshold + `k` combinations	Systematically covers all meaningful input-output pairs	Test count grows as combinations increase

ii. **Why choose these techniques?**

- **EP** validates directory parameters quickly.
- **BVA** ensures handling of empty, max-length, and overflow inputs.
- **Decision Table** confirms correct behavior across threshold and k settings.

iii. **On what level to apply black-box and why?**

- Apply at **system/integration level**, testing API and CLI surfaces without internal implementation details.

Test cases codes:

Test case 1:

```
import pytest
from click.testing import CliRunner
from app import cli

@pytest.fixture(autouse=True)
def patch_build(monkeypatch):
        monkeypatch.setenv("PINECONE_API_KEY",
    "fake") monkeypatch.setenv("GOOGLE_API_KEY",
    "fake") # stub out actual index building
        monkeypatch.setattr("store_index.build_index", lambda path: None)

# Test Case 1: CLI PDF Upload (EP) def
test_cli_pdf_valid():
        runner = CliRunner()
        result = runner.invoke(cli, ["--path", "any"])
    assert result.exit_code == 0
        assert "Index built" in result.output

    def test_cli_missing_option():
    runner = CliRunner()
        result = runner.invoke(cli,
    []) assert result.exit_code != 0
        assert "Missing option" in result.output
```

Test case 2:

```
import pytest
from app import app

@pytest.fixture
def client():
            with
        app.test_client() as c:
        yield c

# Test Case 2: Chat UI Load (Smoke) def
test_ui_index_page(client):
        rv = client.get("/")
        assert rv.status_code == 200
        # your container is now <div id="messageFormeight">
```

Test case 3:

```
# tests/test_query_bva.py
import pytest
from app import app

# Test Case 3: Chat Query Handling (BVA)
@pytest.fixture
def client():
            with
        app.test_client() as c:
        yield c

# Empty input (lower boundary) def
test_query_empty(client):
        rv = client.post("/get", data={"msg": ""})
    assert rv.status_code == 400

# Maximum allowed length (upper boundary) def
test_query_max_length(client):
        msg = "x" * 1000    # assume 1000 is max
        rv = client.post("/get", data={"msg": msg})
```

Code coverage report:

Coverage report: 87%

Files | Functions | Classes

coverage.py v7.8.0, created at 2025-05-01 20:57 +0500

File ▲	function	statements	missing	excluded	coverage
src__init__.py	*(no function)*	0	0	0	100%
src\helper.py	load_pdf_file	3	2	0	33%
src\helper.py	text_split	3	0	0	100%
src\helper.py	download_hugging_face_embeddings	2	0	0	100%
src\helper.py	*(no function)*	6	0	0	100%
src\prompt.py	*(no function)*	1	0	0	100%
Total		15	2	0	87%

coverage.py v7.8.0, created at 2025-05-01 20:57 +0500

Test summary report:

S.no	Test Case	Test Steps	Test Data	Expected Output	Actual Output	Pass/Fail	Comments
Tc_004	CLI PDF Upload (EP)	1. Create temp folder with .pdf2. Run cli --path path	Vali d PDF path	Exit code 0, output: “Index built”	“Index built”	Pass	Validate s expected CLI flow
Tc_005	CLI PDF Missing Path	1. Run cli --path no_such_dir	Non-existent path	Exit code non-zero + error	Error message	Pass	Catches missing input directory
Tc_006	Chat UI Load (Smoke)	1. GET /	—	Page loads with messageFormeight div	HTTP 200 with div present	Pass	Confirms UI template loading
Tc_007	Chat Query Empty (BVA)	1. POST /get with empty msg	Empty string	400 Bad Request	400 status	Pass	Empty queries correctly blocked
Tc_008	Chat Query Max Length	1. POST /get with 1000-char msg	1000 chars	200 OK, returns bot response	200 OK	Pass	Max input handled gracefully

Lab 7: Unit Testing in Python

Exercise:

1. **Validate Basic Calculator application through unittest module. Use atleast 6 assertions methods in your testfile.**

Calculator Class

calculator.py > Calculator

```python
class Calculator:
    def add(self, a, b):
        return a + b

    def subtract(self, a, b):
        return a - b

    def multiply(self, a, b):
        return a * b

    def divide(self, a, b):
        if b == 0:
            raise ValueError("Cannot divide by zero")
        return a / b
```

Writing Unit Tests For calculator.py

test_calculator.py > ...

```python
import unittest
from calculator import Calculator

class TestCalculator(unittest.TestCase):

    def setUp(self):
        self.calc = Calculator()

    def test_add(self):
        self.assertEqual(self.calc.add(10, 5), 15)

    def test_subtract(self):
        self.assertNotEqual(self.calc.subtract(10, 5), 0)

    def test_multiply(self):
        self.assertEqual(self.calc.multiply(3, 7), 21)

    def test_divide(self):
        self.assertAlmostEqual(self.calc.divide(10, 4), 2.5)

    def test_divide_by_zero(self):
        with self.assertRaises(ValueError):
            self.calc.divide(5, 0)

    def test_type(self):
        self.assertIsInstance(self.calc.add(3, 2), int)

if __name__ == "__main__":
    unittest.main()
```

OUTPUT:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Technezo\Documents\Uni test in python> python test_calculator.py
......
----------------------------------------------------------------------

OK
```

2. **Validate any 2 module of your previous python application through unitest**

   1. **LOGIN MODULE:**

auth.py > signup

```python
def login(username, password):
    if username == "admin" and password == "pass":
        return True
    return False

def signup(username, password):
    if len(username) < 4 or len(password) < 6:
        return "Invalid input"
    return "Signup successful"
```

WRITING Unit Tests for Login Module

test_auth.py > TestAuth > test_signup_short_username

```python
import unittest
from auth import login, signup

class TestAuth(unittest.TestCase):

    def test_login_success(self):
        self.assertTrue(login("admin", "pass"))

    def test_login_fail(self):
        self.assertFalse(login("user", "wrong"))

    def test_signup_success(self):
        self.assertEqual(signup("aliarslan", "aliNED2022"), "Signup successful")

    def test_signup_short_username(self):
        self.assertEqual(signup("ali", "aliNED20222026"), "Invalid input")

    def test_signup_short_password(self):
        self.assertEqual(signup("validuser", "123"), "Invalid input")

if __name__ == "__main__":
    unittest.main()
```

OUTPUT

```
PS C:\Users\Technezo\Documents\Uni test in python> python test_auth.py
.....
----------------------------------------------------------------------
Ran 5 tests in 0.002s

OK
```

2. **GPA Calculation module**

gpa.py > get_grade

```python
def average(grades):
    if not grades:
        return 0
    return sum(grades) / len(grades)

def get_grade(gpa):
    if gpa >= 3.5:
        return "A"
    elif gpa >= 3.0:
        return "B"
    elif gpa >= 2.0:
        return "C"
    else:
        return "F"
```

test_gpa.py > ...

```python
import unittest
from gpa import average, get_grade

class TestGPA(unittest.TestCase):

    def test_average(self):
        self.assertAlmostEqual(average([4.0, 3.0, 2.0]), 3.0)

    def test_average_empty(self):
        self.assertEqual(average([]), 0)

    def test_get_grade_A(self):
        self.assertEqual(get_grade(3.7), "A")

    def test_get_grade_B(self):
        self.assertEqual(get_grade(3.2), "B")

    def test_get_grade_F(self):
        self.assertEqual(get_grade(1.5), "F")

if __name__ == "__main__":
    unittest.main()
```

OUTPUT:

```
PS C:\Users\Technezo\Documents\Uni test in python> python test_gpa.py
.....
----------------------------------------------------------------------
Ran 5 tests in 0.001s

OK
```

Lab 08 : Introduction to Katalon Studio + Installation

Exercises:

1. **State the difference between Testcase & Test Suit**

Aspect	Test Case	Test Suite
Definition	A set of steps, input, and expected output to test a specific feature.	A collection of test cases grouped together to run as a batch.
Purpose	Verifies a specific function of the application.	Allows execution of multiple test cases together.
Execution	Run individually.	Run sequentially or in parallel.
Example	Testing login with valid credentials.	A test suite that includes login, dashboard, and logout tests.

2. **Why you preferred Katalon over any other Automation tool available in market ? state in your words also justify your answer with suitable example**

   My Preference: Katalon Studio

   - Beginner-friendly GUI: Ideal for non-programmers as well.
   - All-in-one Tool: Combines Web, API, Mobile, and Desktop automation.
   - Built on Selenium & Appium: So you get the power of those tools with a simpler interface.
   - Record & Playback: Reduces test script creation time.
   - Data-Driven Testing: Easily integrates with Excel, CSV, DB, etc.

   Example:
   Using Selenium, a tester has to write Java code for login automation, wait handling, and assertions.
   In Katalon, the same scenario can be built using a recording tool, and validations can be done with a few clicks—no coding required.

3. **State the different views of Katalon GUI.**

Katalon provides a user-friendly GUI with the following key views:

1. Test Explorer: Displays folders like Test Cases, Test Suites, Object Repository, etc.
2. Manual View: Allows you to create test cases using a step-by-step GUI without code.
3. Script View: Lets you write or edit Groovy/Java test scripts directly.
4. Object Repository View: Stores UI elements captured or defined for reuse.
5. Console Log: Shows logs of the test execution.
6. Execution View: Provides real-time progress and result during test runs.
7. Integration View: Manage integrations like Jenkins, Git, JIRA, etc.

4. **How many types of files you can attach to Katalon ? which feature of katalon allows you to attach file in katalon test suit.**

Types of Files We Can Attach:

- .xlsx, .csv — for **Data-Driven Testing**
- .apk, .ipa — for **Mobile app testing**
- .jar — external libraries
- .xml, .json — for **Web Services/API testing**
- .properties, .txt, .sql — for configuration and test inputs

Feature That Allows File Attachment:

- **Test Data (Data Files)**: Attach Excel or CSV for **Data-Driven Tests**
- **Desired Capabilities**: Attach .apk or .ipa for mobile test setup
- **Library Management**: Attach .jar files under Include > Scripts > Libraries
- **Test Suite Collection**: You can bind data files or configurations while defining suite execution

5. **What is the purpose of Object Spy and Object Repository ? Explain with suitable example**

Object Spy:

- A tool in Katalon that captures UI elements (e.g., buttons, fields) from your application under test.
- It saves these elements into the Object Repository for reuse.

Example:
You open your web app and launch Object Spy, click on the “Login” button. Katalon will capture its XPath or CSS selector and save it as Login_Button in the Object Repository.

Object Repository:

- A central storage for all UI test objects (buttons, fields, links, etc.)
- Helps you reuse elements across multiple test cases and maintain them easily.

Example:
If your test uses the same “Submit” button in 5 test cases, you only define it once in the Object Repository and reference it everywhere.

Lab 09: Create Test Case [Manual & Script View]

EXERCISE:

1. Write Testcases for login/logout features (Atleast 6 combinations) of "Cura Health Care " website . Use username= John Doe, Password=ThisIsNotAPassword ;

Also Print "Invalid credentials "for incorrect attempt on console.

/*attach Testreport and screenshot Here*

/*Hint: use WebSpy to capture object and save it to object repository*/

SOLUTION:

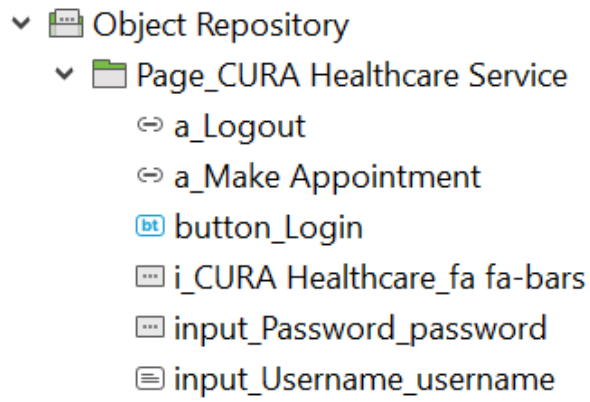
TestCases for Login:

Item	Object	Input	Output
1 - Method Call Statement		performLogin("John Doe", "ThisIsNotAPassword", "success", "positive_login.png")	
2 - Method Call Statement		performLogin("WrongUser", "WrongPassword", "failure", "wrong_user_wrong_pass.png")	
3 - Method Call Statement		performLogin("John Doe", "WrongPassword", "failure", "correct_user_wrong_pass.png")	
4 - Method Call Statement		performLogin("WrongUser", "ThisIsNotAPassword", "failure", "wrong_user_correct_pass.png")	
5 - Method Call Statement		performLogin("", "ThisIsNotAPassword", "failure", "empty_user_correct_pass.png")	
6 - Method Call Statement		performLogin("John Doe", "", "failure", "correct_user_empty_pass.png")	
7 - Method Call Statement		performLogin("", "", "failure", "empty_user_empty_pass.png")	
performLogin()			

Test Suite

No.	ID	Description	Flakiness (%)	Latest Run	Avg.Duration	Run
1	Test Cases/Login		--	--	--	

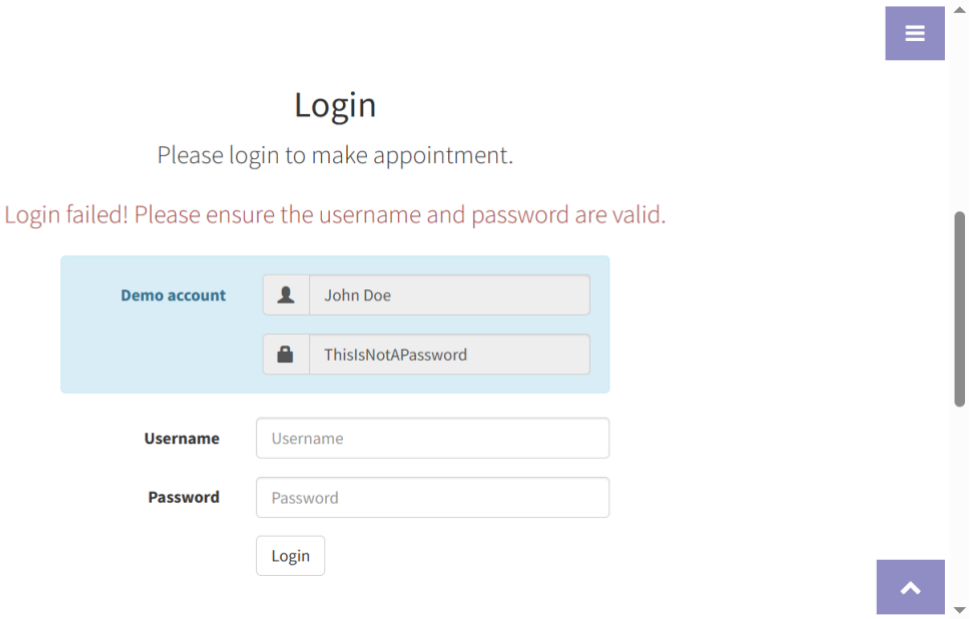
Object Repository:



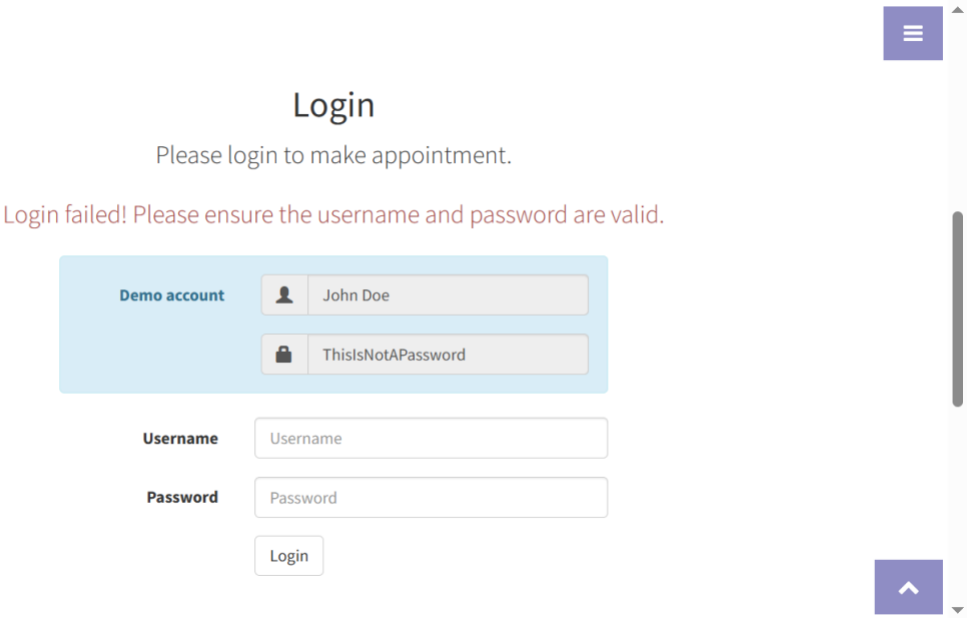
Screens Shots of all 7 Test Cases:

1. Correct Username & Correct Password

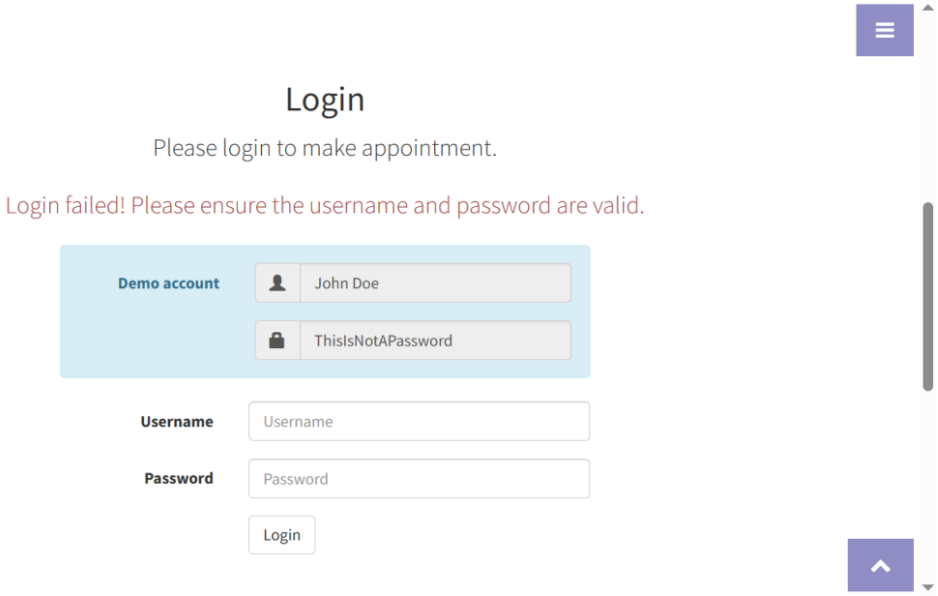
2) Wrong Username & Wrong Password



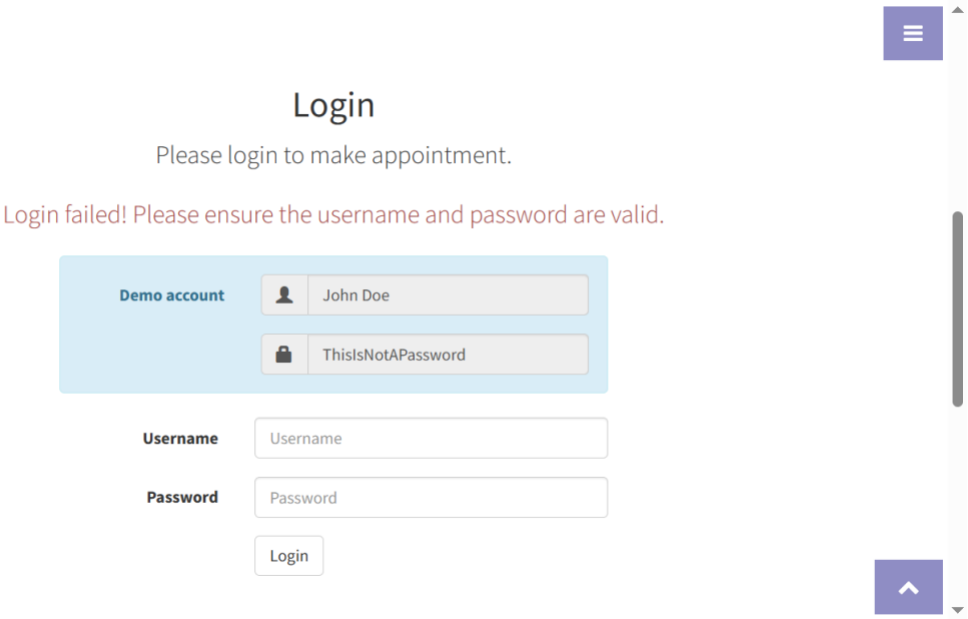
3. Correct Username & Wrong Password



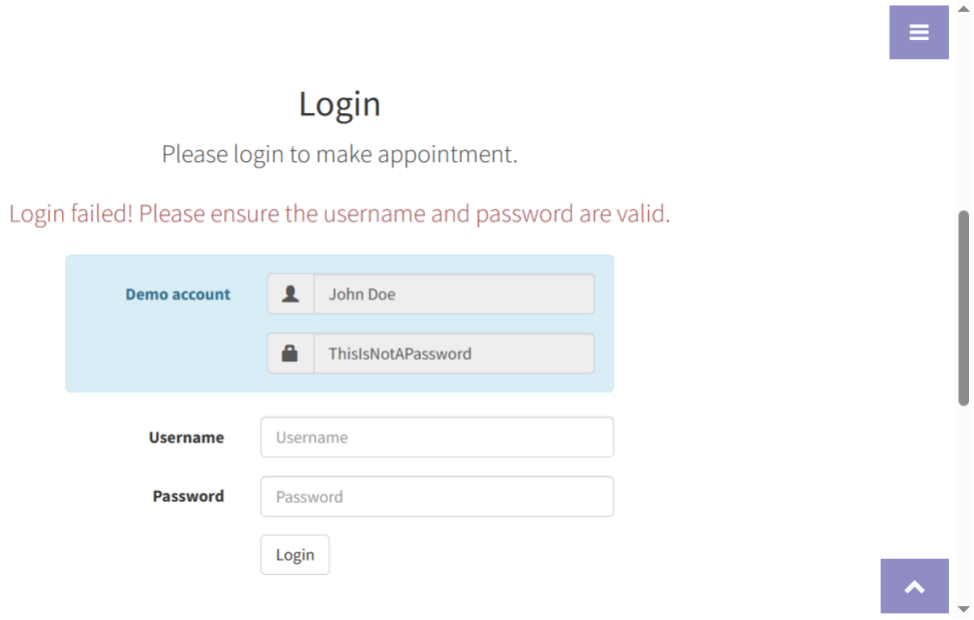
4. Wrong Username & Correct Password



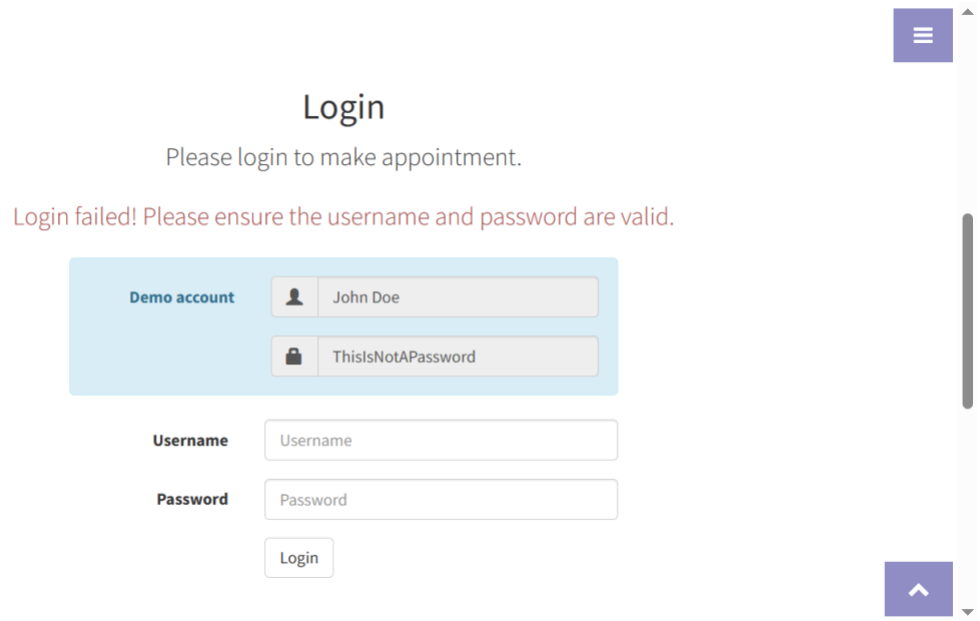
5. Empty Username & Correct Password



6. **Correct Username & Empty Password**



7. **Empty Username & Empty Password**



TEST SUMMARY REPORT

Tc login1

Execution Environment

Host name	Technezo - DESKTOP-2SQFQEV
Local OS	Windows 10 64bit
Katalon version	10.1.1.0
Browser	Chrome 135.0.7049.115
Device name	

Summary

ID	Test Suites/Tc login1		
Description			
Total	1		
Passed	1	Failed	0
Error	0	Incomplete	0
Skipped	0		
Start	04-05-2025 15:22:07	End	04-05-2025 15:23:41
Elapsed	1m - 34.279s		

#	ID	Description	Status
1	Test Cases/Login		PASSED

NOTE:
See full test summary report at https://github.com/MuhammadAliArsalan/SQE-lab9-Test-summary

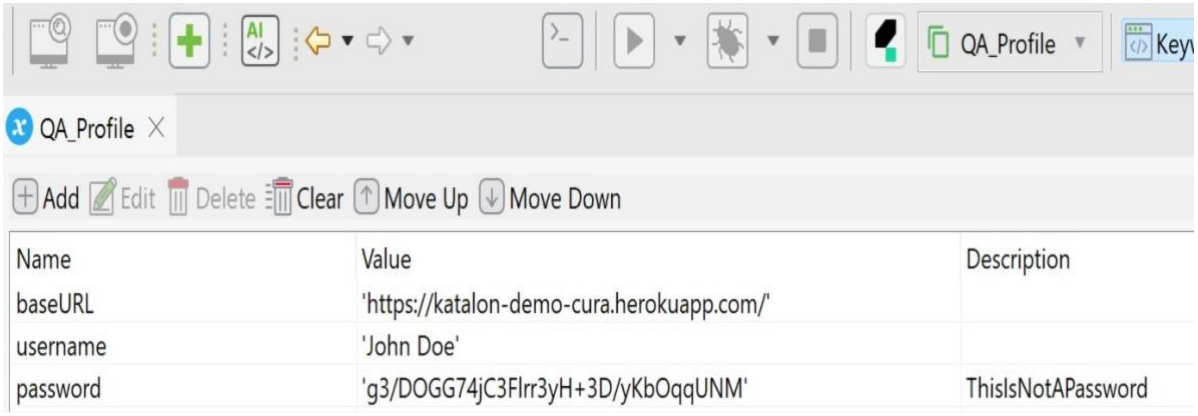
Lab 10

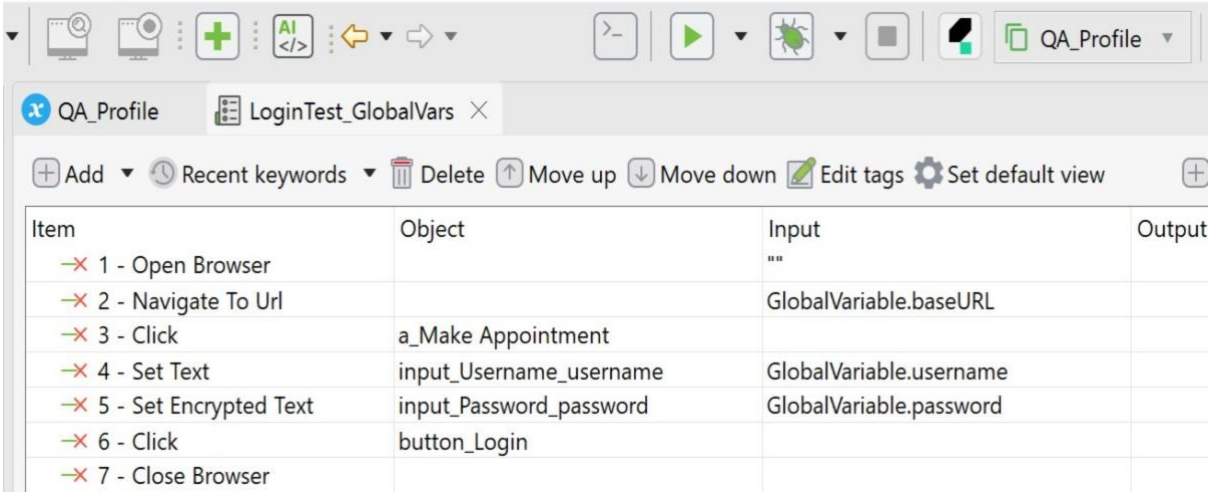
Variables (Local & Global) and Profiles

Exercise:

1. Set up execution profile for QA environments and run tasks(below) under QA profile.

2. Define global variables for base URL, login credentials,use it in a testcase and execute.

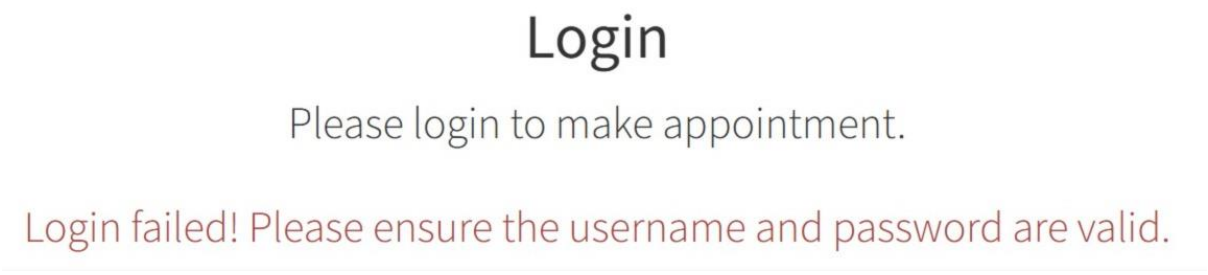


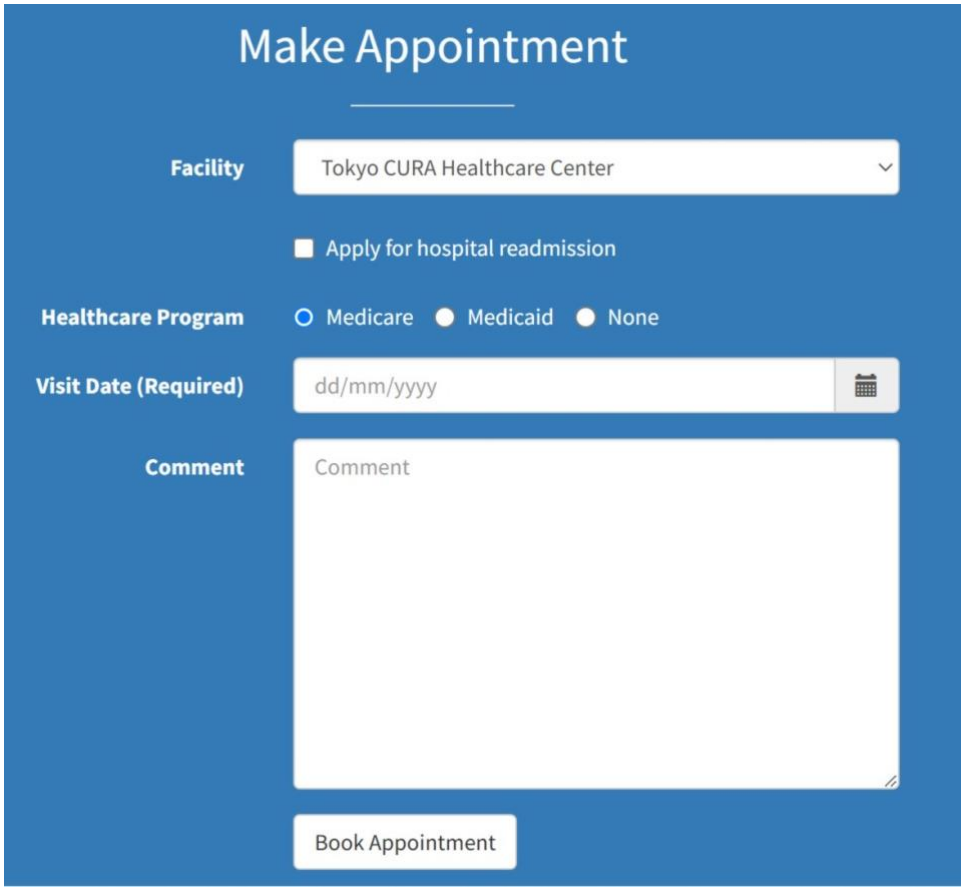


3. **Run the same test case(under different profiles and compare responses**

Name	Value	Description
baseURL	'https://katalon-demo-cura.herokuapp.com/'	
username	'Jane Doe'	
password	'RyzK/uu8Q1+7/zuOsKmZ+g=='	WrongPassword

COMPARING RESPONSES ACROSS PROFILES: (Q3 and Q2 respectively)

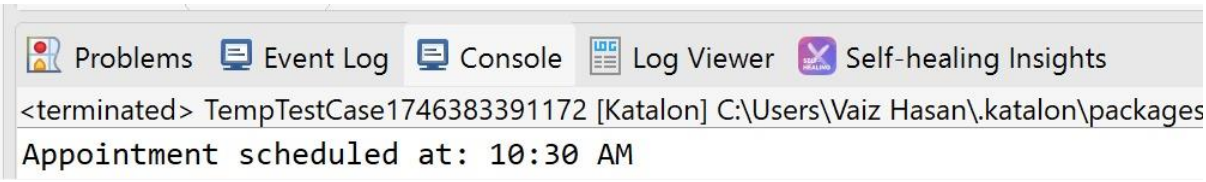




4. **Create local variables for appointment time slots during test case execution.**

No.	Name	Type	Default value
1	appointmentTime	String	"10:30 AM"

```
WebUI.openBrowser('')
WebUI.navigateToUrl(GlobalVariable.baseURL)
WebUI.click(findTestObject('Page_CURA Healthcare Service/a_Make Appointment'))
WebUI.setText(findTestObject('Page_CURA Healthcare Service/input_Username_username'), GlobalVariable.username)
WebUI.setEncryptedText(findTestObject('Page_CURA Healthcare Service/input_Password_password'), GlobalVariable.pas
WebUI.click(findTestObject('Page_CURA Healthcare Service/button_Login'))

println "Appointment scheduled at: ${appointmentTime}"

WebUI.closeBrowser()
```



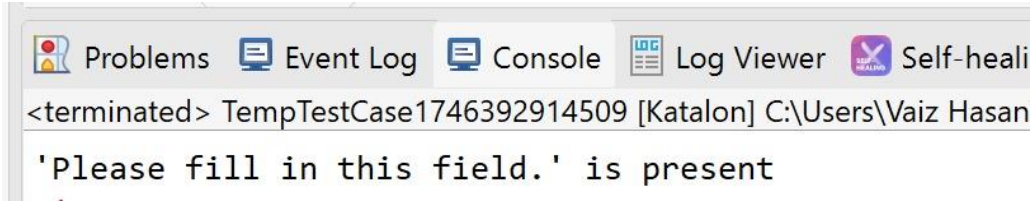
5. **Simulate failed login attempts with incorrect credentials — catch authentication exceptions.**

```
try {
    WebUI.click(findTestObject('Page_CURA Healthcare Service/button_Login'))
    WebUI.verifyTextPresent('Login failed', false)
} catch (Exception e) {
    WebUI.comment("Authentication failed as expected: " + e.message)
} finally {
    WebUI.closeBrowser()
}
```

```
05-05-2025 12:56:21 AM verifyTextPresent("Login failed", false)

Elapsed time: 0.537s

Text 'Login failed' is present on page
```

6. **Attempt booking with invalid data (e.g., empty fields, past date) — validate input exceptions.**

```
try {
    WebUI.setText(findTestObject("Page_CURA Healthcare Service/input_Visit Date (Required)_visit_date"), '')

    WebUI.click(findTestObject('Page_CURA Healthcare Service/button_Book Appointment'))

    WebUI.verifyTextPresent('Please fill in this field.', false)
    WebUI.comment("Validation error caught for empty date field")
} catch (Exception e) {
    println("Input validation exception caught for empty field: ${e.message}")

}
```




```
try {
    WebUI.setText(findTestObject('Page_CURA Healthcare Service/input_Visit Date (Required)_visit_date'),
         '2020-01-01')
    WebUI.click(findTestObject('Page_CURA Healthcare Service/button_Book Appointment'))

    WebUI.verifyElementPresent(findTestObject('Page_CURA Healthcare Service/Error_PastDate'), 3)
    WebUI.comment("Validation error caught for past date field")
} catch (Exception e) {
    println "Input validation exception caught for past date: ${e.message}"
} finally {
    WebUI.closeBrowser()
}
```

Note: The Katalon demo web does not validate past dates, so no error is shown when booking with a past appointment date.

7. **Disconnect network mid-session — test timeout and connection exceptions.**

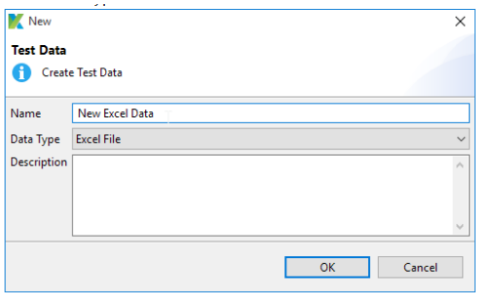
```
try {
    WebUI.click(findTestObject('Page_CURA Healthcare Service/a_Make Appointment'))
    WebUI.setText(findTestObject('Page_CURA Healthcare Service/input_Username_username'),
        GlobalVariable.username)
    WebUI.setEncryptedText(findTestObject('Page_CURA Healthcare Service/input_Password_password'),
        GlobalVariable.password)
    WebUI.delay(10)
    WebUI.click(findTestObject('Page_CURA Healthcare Service/button_Login'))
    WebUI.verifyTextPresent("This site can’t be reached", false, FailureHandling.STOP_ON_FAILURE)
    WebUI.comment("Action succeeded – network might still be connected.")
} catch (Exception e) {
    WebUI.comment("Network disconnection or timeout handled: " + e.message)
}
```

```
05-05-2025 02:56:43 AM verifyTextPresent("This site can’t be reached", false,
STOP_ON_FAILURE)

Elapsed time: 0.575s

Text 'This site can’t be reached' is present on page
```

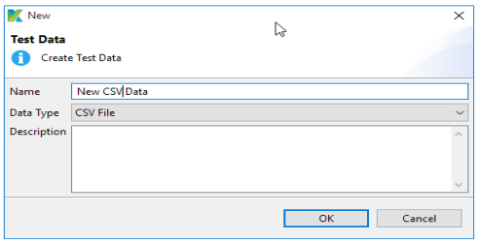
Lab 11: Data Driven Testing

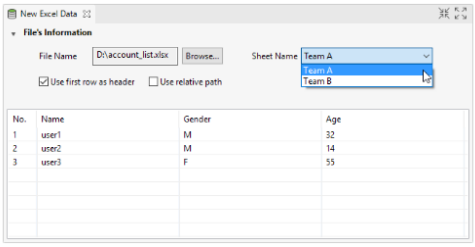
Create an Excel Test Data

1. Select **File > New > Test Data** from the main menu. The **New Test Data** dialog is displayed. Enter the name for your test data and select **Data Type** as **Excel File**. Click **OK**.



2. Browse to the Excel file that you want to import into Katalon Studio. Data from the selected Excel file is populated into the **preview section** below.

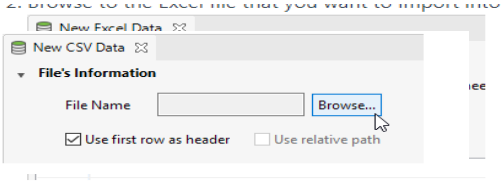




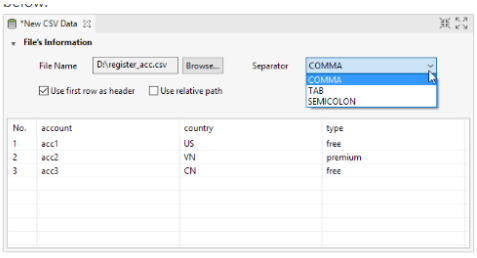
3. Save the Test Data when you finish. The data set defined here can be utilized in other configurations. For example, you can use it to input data for keywords in Manual View or data for Test Execution when setting up test suites.

Create a CSV Test Data

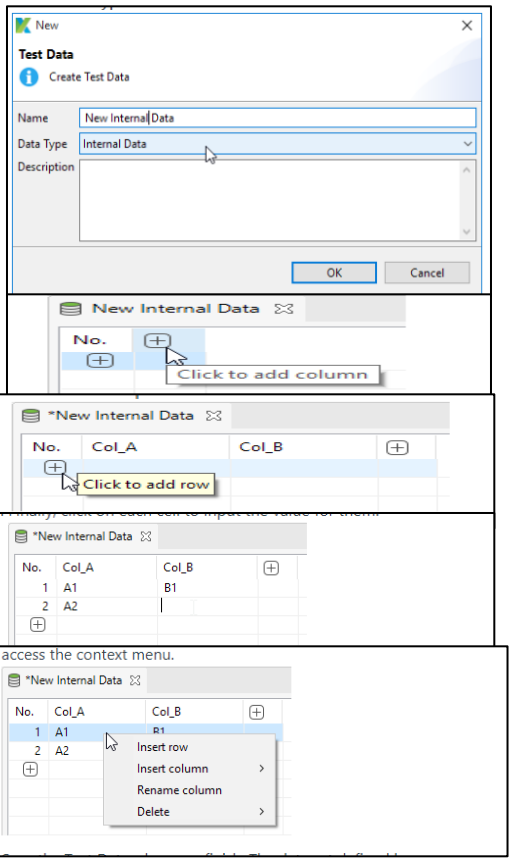
1. Select **File > New > Test Data** from the main menu. The **New Test Data** dialog is displayed. Enter the name for your test data and select **Data Type** as **CSV File**. Click **OK**.



2. Browse to the CSV file that you want to import into Katalon Studio.Data from the selected CSV file is populated to the **preview section** below.



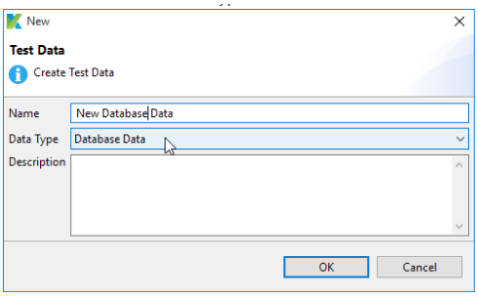
3 .Save the Test Data when you finish. The data set defined here can be utilized in other configurations. For example, you can use it to input data for keywords in Manual View or data for Test Execution when setting up test suites.



Create an Internal Test Data

With **Internal Data**, you can freely define the data in tabular format. It's up to you to decide how many columns, rows, and what value for each cell.

1. Select **File > New > Test Data** from the main menu. The **New Test Data** dialog is displayed. Enter the name for your test data and select **Data Type** as **Internal Data**. Click **OK**.
2. In the Editor View, select the option to add a new column.
3. Select the option to add a new row.
4. Finally, click on each cell to input the value for them.
5. You can edit or delete the columns (or rows) by right-clicking to access the context menu.
6. Save the Test Data when you finish. The data set defined here can be utilized in other configurations. For example, you can use it to input data for keywords in Manual View or data for Test Execution when setting up test suites.



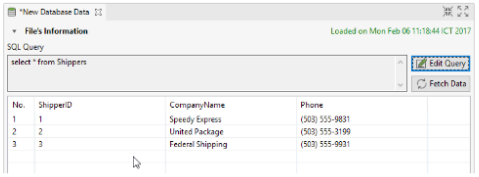
Create a Database Data

With **Internal Data**, you can freely define the data in tabular format. It's up to you to decide how many columns, rows, and what value for each cell.

1. Select **File > New > Test Data** from the main menu. The **New Test Data** dialog is displayed. Enter the name for your test data and select **Database Data** as **Data Type**. Click **OK**.

2. Click **Edit Query** to open the **Database Connection and Query settings** dialog.

3. Enter the connection details as well as the data query then click **OK**.



Starting from **Katalon Studio version 7.0.0 and later**, you can query data from additional database sources with the **JDBC Driver** field in the dialog. You can enter the ClassDriverName of the database that has a library support connection (JDBC).

4. Queried data is fetched and loaded respectively into the **preview section** .

5. Save the Test Data when you finish. The data set defined here can be utilized in other configurations. For example, you can use it to input data for keywords in Manual View or data for Test Execution when setting up test suites.

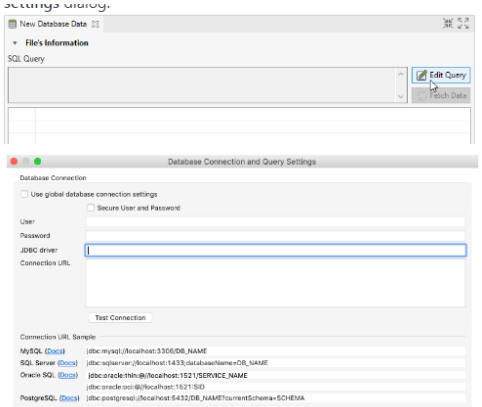
How to use Data from Excel File in Katalon:

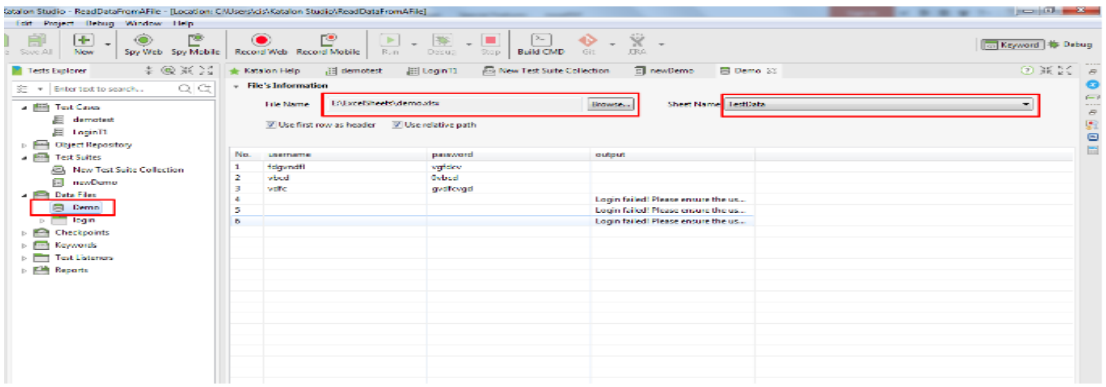
Step 1: create a new data file(excel) example “demo.xlsx”.

Step 2: And enter some data into that file.

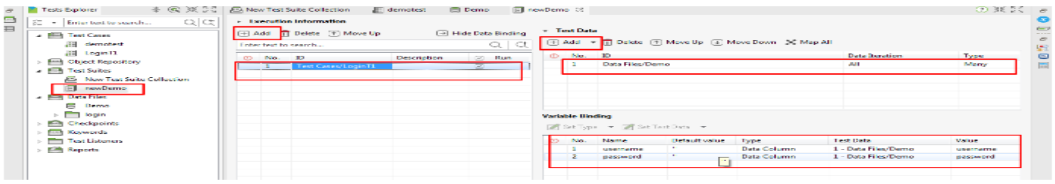
Step 3: Create a new data file in katalon studio.

Step 4: And browse your external file and select the sheet.





Step 5: Create test case and **define some variable in the test case**.
Step 6: Now create a TestSuite and configure a test case. And bind the variables with the column name.



Step 7. Set up a project. You can choose to create your new project either from scratch or from one of project samples (available with katalon studio).

Go to File > New > Project, then select your preference. Insert the project's name, the location to store its data and description (optional).

Step 8. Create your first test caseTo create a test case, right-click Test Cases > New > Test Case To simplify and speed up the process of writing test cases, especially in high volume, you can use Katalon Studio standard format. For example:

WebUI.openBrowser('https://katalon-demo-cura.herokuapp.com/')

WebUI.click(findTestObject('Page_CURA Healthcare Service/a_MakeAppointment'))

WebUI.setText(findTestObject('Page_CURAHealthcareService/input_Username_username'), 'John Doe')

WebUI.setEncryptedText(findTestObject('Page_CURA Healthcare Service/input_Password_password'), 'g3/DOGG74jC3Flrr3yH+3D/yKbOqqUNM')

WebUI.click(findTestObject('Page_CURA Healthcare Service/button_Login'))

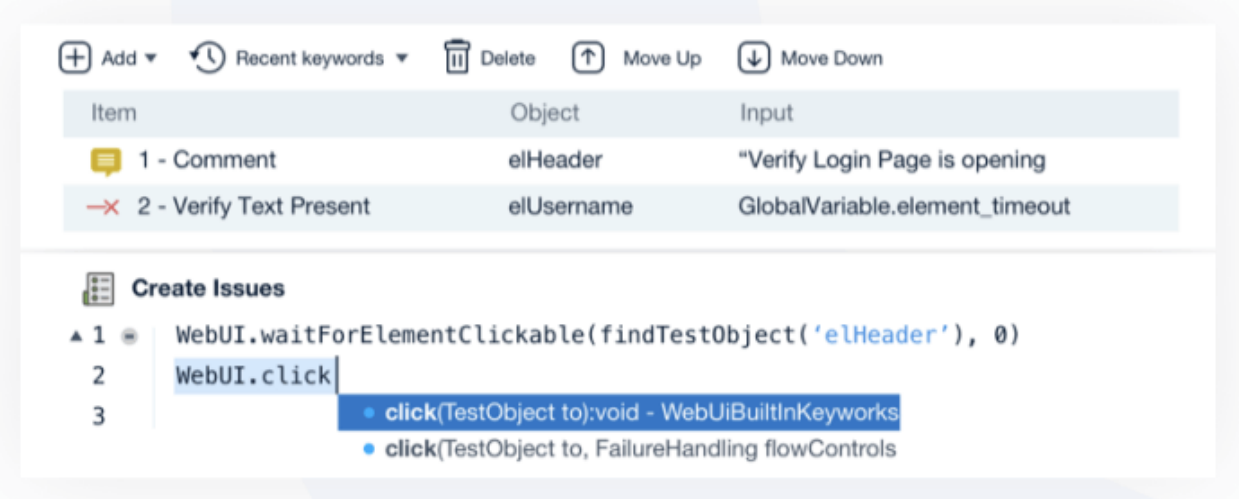
Step9. Click Run to see how it works.

Next, we will go through the "ChromeDriver," "click" and "findTestObject" methods, and how to use the "Page_CURA Healthcare Service/a_Make Appointment" argument to begin testing with Katalon Studio.

- Browser drivers: Browser drivers, i.e., ChromeDriver are abstracted away from the code and set up at the execution time so that the code is enabled for cross-browser testing.

- Click and setText: This is an example of Katalon Studio's **open-source built-in keywords** library to wrap and enhance the limited set of Selenium keywords. You can also create your own set of custom keywords or import the shared custom keywords made by other users.
- Page_CURA Healthcare Service/ a_Make Appointment: are the test objects designed to follow the Page Object Model pattern. Katalon Studio supports your test objects with the "auto-healing" feature to make them sustainable, despite the continuously changing nature of the AUT. The script and objects above can be quickly generated by the Recording and Spying

Katalon Studio provides a dual-script interface for both manual and automated tests:



In practice, a team of testers and developers can apply these steps to their complete workflow:

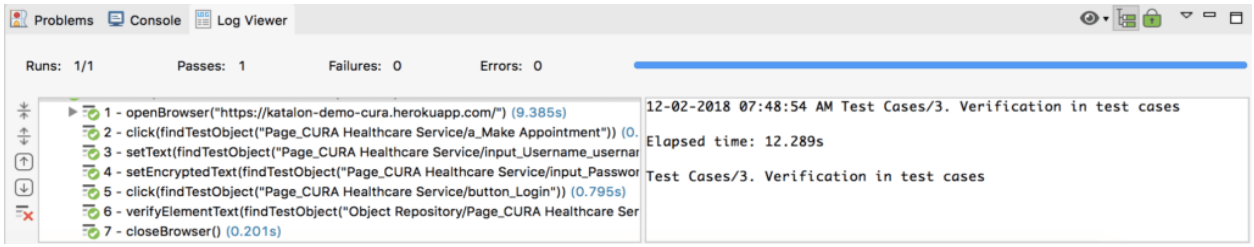
- Firstly, the automation experts will prepare all the scripts needed to build test cases such as the custom keywords, test listeners, calling test cases, etc.
- Then, these prepared scripts will be transferred to the manual QAs to utilize without having to handle scripting.
- As a result, the manual QAs can gradually learn how to script and become automation scripters.

Verification in the test case: To get the result of the login process, you need to add the verification script to the test case. As the fundamental of scripting in Katalon Studio for the first test case has been explained, we can move on to another test case: "Make Appointment" header verification. You can leverage Katalon Studio's rich set of verification keywords to work on that requirement. Note that this kind of script can be done in both Manual and Script mode.

Item	Object	Input
1 - Open Browser		"https://katalon-demo-cura.herokuap
2 - Click	a_Make Appointment	
3 - Set Text	input_Username_username	"John Doe"
4 - Set Encrypted Text	input_Password_password	"g3/DOGG74jC3Flrr3yH+3D/yKbOqq
5 - Click	button_Login	
6 - Verify Element Text	h2_Make Appointment	"Make Appointment"
7 - Close Browser		

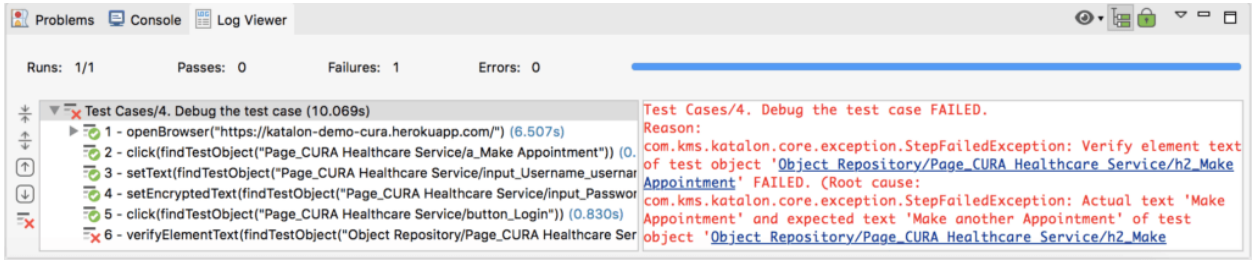
WebUI.verifyElementText(findTestObject('Object Repository/Page_CURA Healthcare Service/h2_Make Appointment'), 'Make Appointment', FailureHandling.STOP_ON_FAILURE)

You can then check the execution results in the Log Viewer



Debugging the test case: Try changing the verification text to “Make another Appointment.” to make the test case fail. Here are a few options of how you can check the reasons for failure in Katalon Studio:

- Option 1: Investigate error logs



- Option 2: Debug mode

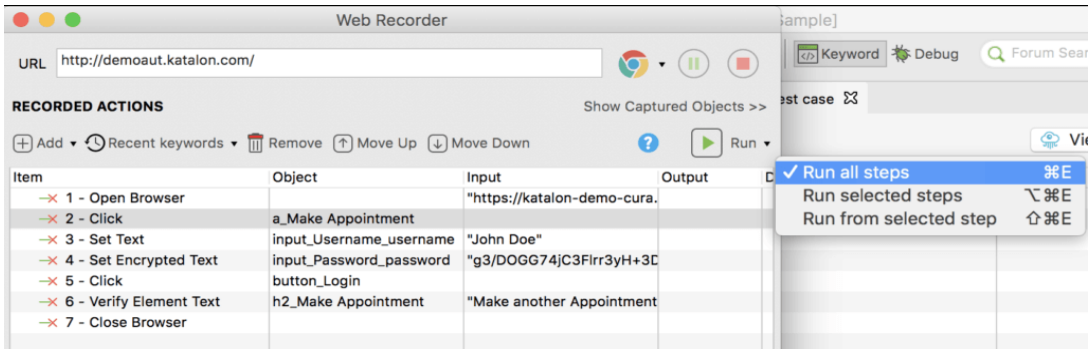
For complex cases, Katalon Studio offers a debug mechanism that works similarly to the code debug mechanism in the advanced developer IDE.

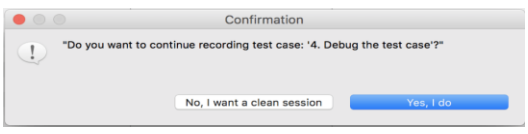
- Option 3: Manual debug

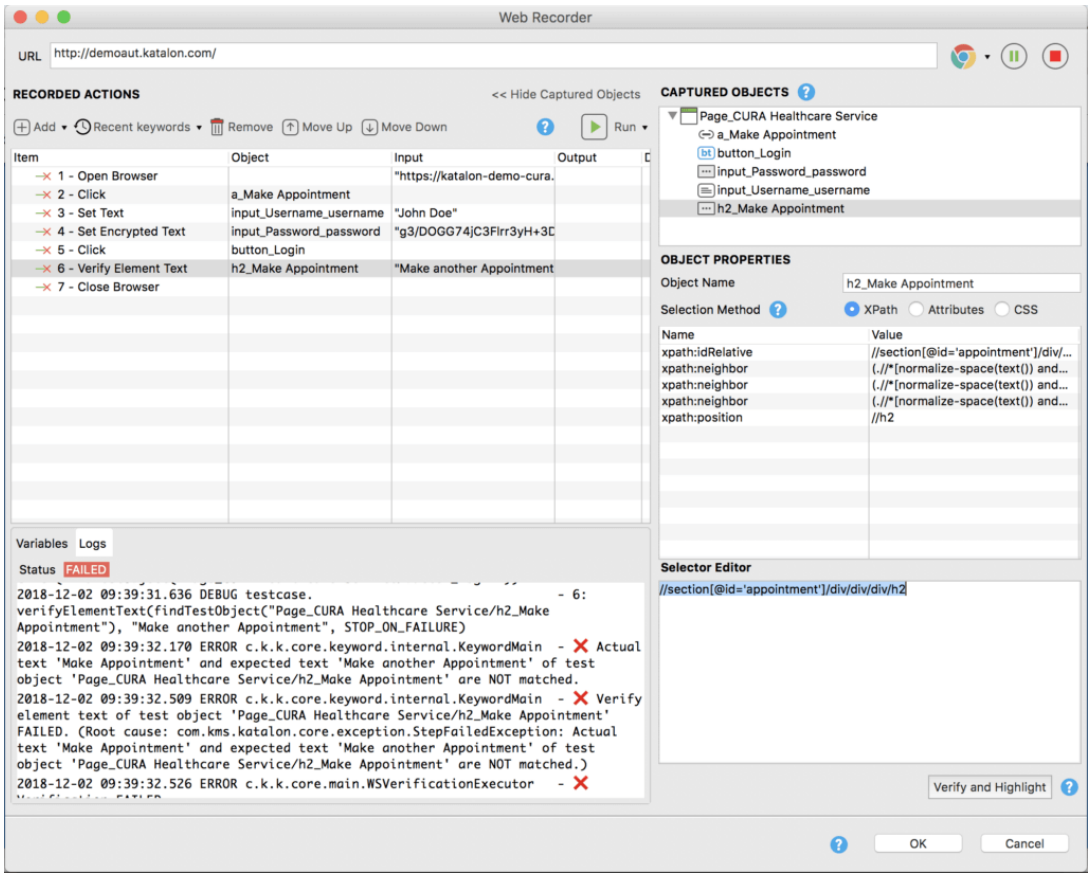
It is possible to use the ‘Recording’ feature of Katalon Studio for debugging. The tool supports recording the failed test case, running the error test step for you to analyze and fix the issue directly in the recording mode.

When your test case failed, click “Yes, I do” in the window that shows up to continue recording the failed test cases.

Choose Run all steps to run the error test step. After that, you can investigate and fix the issue directly in the recording mode.

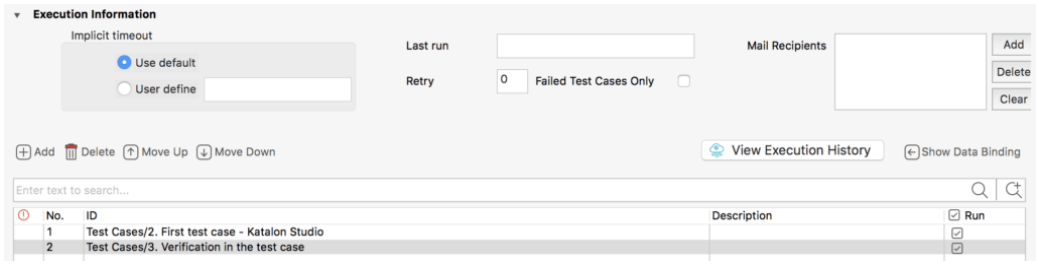






Plan the test case in the test suite: Right-click Test Suites > New > Test Suite

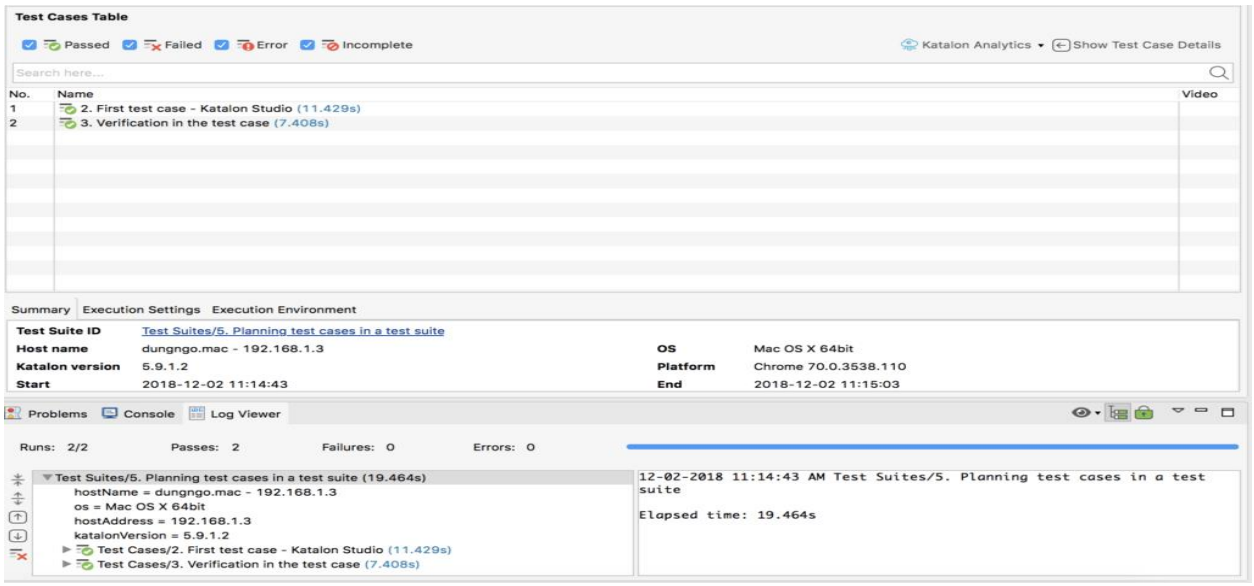
To create test cases with multiple configurations (e.g., retrying on failure, email sending or data-driven binding), use the test suite capability. The added configurations of a test suite execution can be managed by expanding the Execution Information section. See the example below:



Execute the test suite and view the result: Select Test Suite > Run

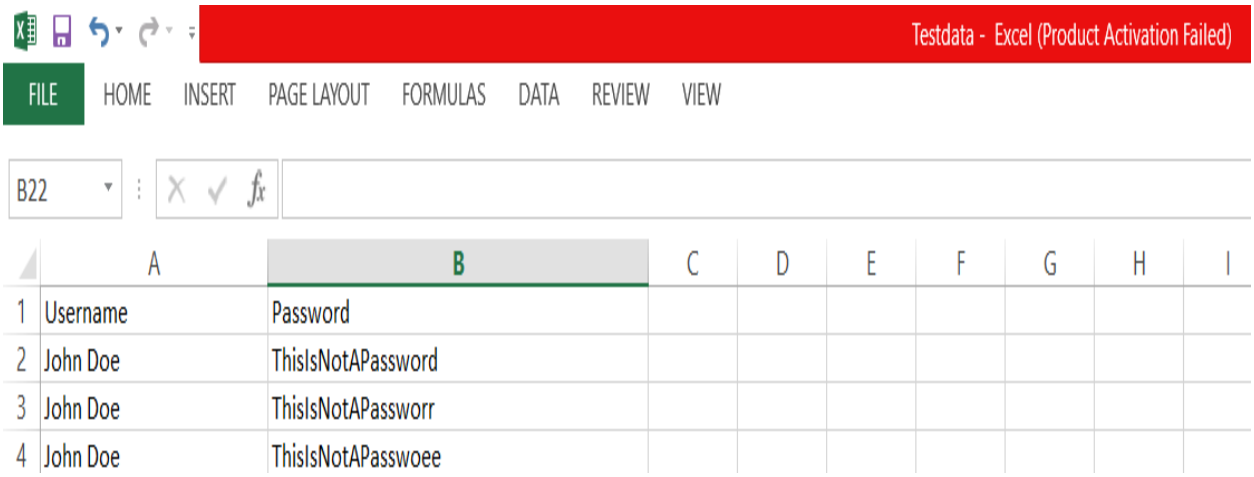
The final step is to execute a test case in the designed test suite:

- After being successfully planned, the test suites or test suite collections can be executed not only directly in Katalon Studio, but also by the command line.
- Combining the standard JUnit format of the execution results and pre-built Docker images, these features will give you the flexibility to integrate Katalon Studio into a CI/CD pipeline with tools such as Jenkins or CircleCI.

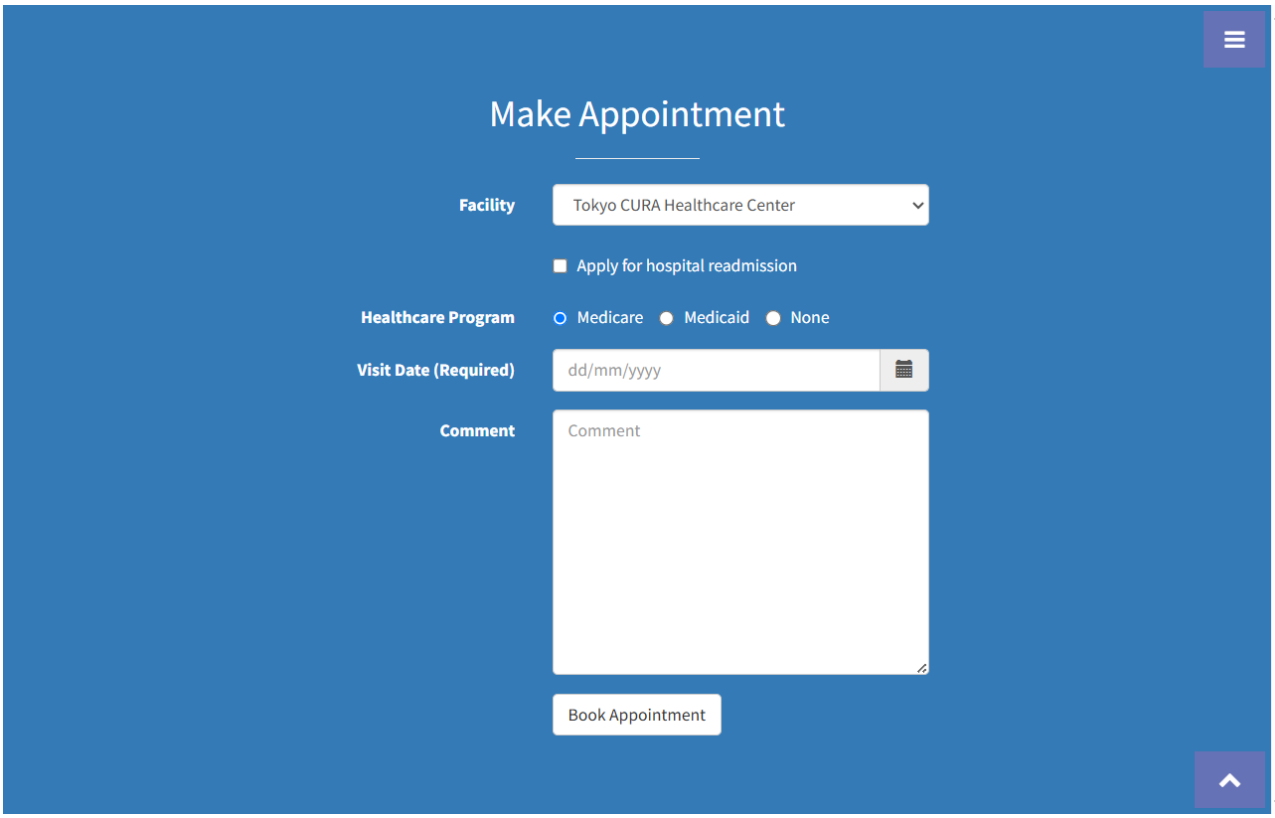


Exercise:

1.Attach an excel sheet ”Testdata”(having Username and password) , Create and execute testcases for cure health ;generate and attach Testreport/*Attach screenshot/printout here*/ Testdata must have two invalid and one valid password, Use any loop through scripting language.



	Suite/Test/Step Name	Browser	Descriptio	Tag	Start time	End time	Duration	Status
2	LoginSuite	Chrome 135.0.7049.115			########	########	40.742s	PASSED
3								
4	Test Cases/LoginDataDriven	Chrome 135.0.7049.115			########	########	38.610s	PASSED
5	loginData = findTestData("Data Files/newone")	Chrome 135.0.7049.115			########	########	1.456s	PASSED
6	for ([i = 1, i <= loginData.getRowNumbers(), (i++)])	Chrome 135.0.7049.115			########	########	0.238s	PASSED
7	url = "https://katalon-demo-cura.herokuapp.com/"	Chrome 135.0.7049.115			########	########	0.004s	PASSED
8	openBrowser("")	Chrome 135.0.7049.115			########	########	5.016s	PASSED
9	navigateToUrl(url)	Chrome 135.0.7049.115			########	########	3.692s	PASSED
10	click(findTestObject("Object Repository/Page_CURA Healthcare Service/a_Make Appointment"))	Chrome 135.0.7049.115			########	########	1.573s	PASSED
11	setText(findTestObject("Object Repository/Page_CURA Healthcare Service/input_Username_username"), username)	Chrome 135.0.7049.115			########	########	1.023s	PASSED
12	setText(findTestObject("Object Repository/Page_CURA Healthcare Service/input_Password_password"), password)	Chrome 135.0.7049.115			########	########	0.980s	PASSED
13	click(findTestObject("Object Repository/Page_CURA Healthcare Service/button_Login"))	Chrome 135.0.7049.115			########	########	1.312s	PASSED
14	if (expectedResult == "success")	Chrome 135.0.7049.115			########	########	0.556s	PASSED
15	closeBrowser()	Chrome 135.0.7049.115			########	########	0.363s	PASSED
16	url = "https://katalon-demo-cura.herokuapp.com/"	Chrome 135.0.7049.115			########	########	0.001s	PASSED
17	openBrowser("")	Chrome 135.0.7049.115			########	########	1.789s	PASSED
18	navigateToUrl(url)	Chrome 135.0.7049.115			########	########	3.311s	PASSED
19	click(findTestObject("Object Repository/Page_CURA Healthcare Service/a_Make Appointment"))	Chrome 135.0.7049.115			########	########	1.747s	PASSED
20	setText(findTestObject("Object Repository/Page_CURA Healthcare Service/input_Username_username"), username)	Chrome 135.0.7049.115			########	########	0.944s	PASSED
21	setText(findTestObject("Object Repository/Page_CURA Healthcare Service/input_Password_password"), password)	Chrome 135.0.7049.115			########	########	0.932s	PASSED
22	click(findTestObject("Object Repository/Page_CURA Healthcare Service/button_Login"))	Chrome 135.0.7049.115			########	########	0.460s	PASSED
23	if (expectedResult == "success")	Chrome 135.0.7049.115			########	########	0.002s	PASSED
24	else if (expectedResult == "failure")	Chrome 135.0.7049.115			########	########	1.238s	PASSED
25	closeBrowser()	Chrome 135.0.7049.115			########	########	0.248s	PASSED
26	url = "https://katalon-demo-cura.herokuapp.com/"	Chrome 135.0.7049.115			########	########	0.001s	PASSED
27	openBrowser("")	Chrome 135.0.7049.115			########	########	1.913s	PASSED
28	navigateToUrl(url)	Chrome 135.0.7049.115			########	########	3.610s	PASSED
29	click(findTestObject("Object Repository/Page_CURA Healthcare Service/a_Make Appointment"))	Chrome 135.0.7049.115			########	########	1.091s	PASSED
30	setText(findTestObject("Object Repository/Page_CURA Healthcare Service/input_Username_username"), username)	Chrome 135.0.7049.115			########	########	1.016s	PASSED
31	setText(findTestObject("Object Repository/Page_CURA Healthcare Service/input_Password_password"), password)	Chrome 135.0.7049.115			########	########	0.873s	PASSED
32	click(findTestObject("Object Repository/Page_CURA Healthcare Service/button_Login"))	Chrome 135.0.7049.115			########	########	0.553s	PASSED
33	if (expectedResult == "success")	Chrome 135.0.7049.115			########	########	0.001s	PASSED
34	else if (expectedResult == "failure")	Chrome 135.0.7049.115			########	########	1.179s	PASSED
35	closeBrowser()	Chrome 135.0.7049.115			########	########	0.211s	PASSED



Login

Please login to make appointment.

Login failed! Please ensure the username and password are valid.

Demo account John Doe

ThisIsNotAPassword

Username Username

Password Password

Login

Login

Please login to make appointment.

Login failed! Please ensure the username and password are valid.

Demo account John Doe

ThisIsNotAPassword

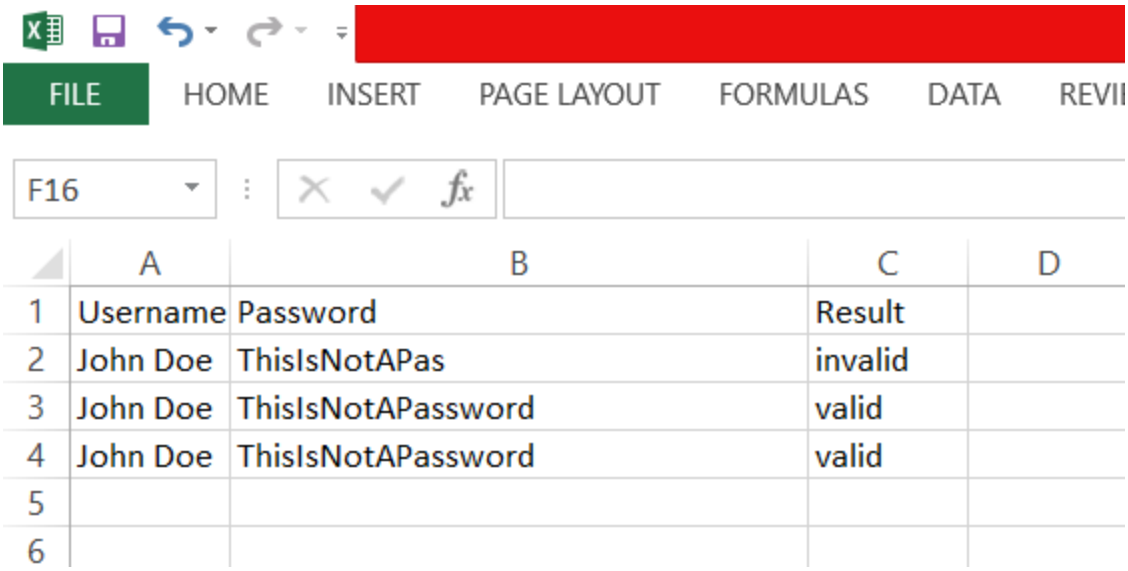
Username Username

Password Password

Login

2. use excelsheet(1 a) and write “valid” [if the username & password is correct] or” invalid” [if the username & password is incorrect] from katalon.

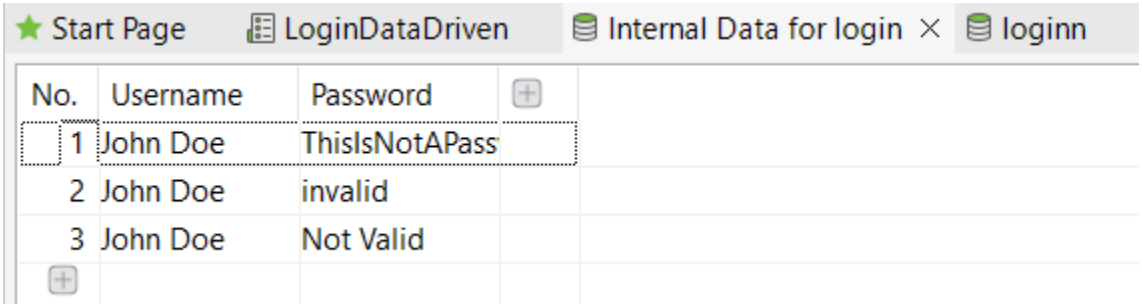
```
import static com.kms.katalon.core.testobject.ObjectRepository.findTestObject
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUI
import com.kms.katalon.core.testdata.TestDataFactory as TestDataFactory
import java.nio.file.Files
import java.nio.file.Paths
import org.apache.poi.ss.usermodel.*
import org.apache.poi.xssf.usermodel.XSSFWorkbook

def writeResultToExcel(String filePath, String sheetName, int rowIndex, int columnIndex, String value) {
    FileInputStream fis = new FileInputStream(filePath)
    XSSFWorkbook workbook = new XSSFWorkbook(fis)
    Sheet sheet = workbook.getSheet(sheetName)

    Row row = sheet.getRow(rowIndex)
    if (row == null) row = sheet.createRow(rowIndex)

    Cell cell = row.getCell(columnIndex)
    if (cell == null) cell = row.createCell(columnIndex)

    cell.setCellValue(value)

    fis.close()

    FileOutputStream fos = new FileOutputStream(filePath)
    workbook.write(fos)
    workbook.close()
    fos.close()
}

def ensureScreenshotDirectoryExists(String path) {
    def dir = Paths.get(path)
    if (!Files.exists(dir)) {
        Files.createDirectories(dir)
    }
}


def loginData = TestDataFactory.findTestData('Data Files/loginn')

String excelFilePath = 'C:/Path/To/Your/loginn.xlsx'
String sheetName = 'Sheet1'
int resultColumnIndex = 2

def performLogin(String username, String password, String expectedResult, String screenshotFileName) {
    String projectDir = System.getProperty("user.dir")
    String screenshotDir = projectDir + '/Screenshots/'
    ensureScreenshotDirectoryExists(screenshotDir)

    WebUI.openBrowser('')
    WebUI.navigateToUrl('https://katalon-demo-cura.herokuapp.com/')
    WebUI.click(findTestObject('Object Repository/Page_CURA Healthcare Service/a_Make Appointment'))

    WebUI.setText(findTestObject('Object Repository/Page_CURA Healthcare Service/input_Username_username'), username)
    WebUI.setText(findTestObject('Object Repository/Page_CURA Healthcare Service/input_Password_password'), password)
    WebUI.click(findTestObject('Object Repository/Page_CURA Healthcare Service/button_Login'))

    boolean result = false

    if (expectedResult == 'success') {
        result = WebUI.verifyTextPresent('Make Appointment', false, FailureHandling.OPTIONAL)
    } else {
        result = WebUI.verifyTextPresent('Login failed! Please ensure the username and password are valid.', false, FailureHandling.OPTIONAL)
    }

    WebUI.takeScreenshot(screenshotDir + screenshotFileName)
    WebUI.closeBrowser()
    return result
}


for (int i = 1; i <= loginData.getRowNumbers(); i++) {
    String username = loginData.getValue('Username', i)?.trim()
    String password = loginData.getValue('Password', i)?.trim()

    println("Row $i - Username: $username, Password: $password")

    String expectedResult = ('John Doe'.equals(username) && 'ThisIsNotAPassword'.equals(password)) ? 'success' : 'failure'
    String fileName = "login_${i}_${username.replaceAll('\\s+', '_')}_${password.replaceAll('\\s+', '_')}.png"

    boolean isCorrect = performLogin(username, password, expectedResult, fileName)

    String resultText = isCorrect ? 'valid' : 'invalid'
    writeResultToExcel(excelFilePath, sheetName, i, resultColumnIndex, resultText)

    println("Result: ${resultText} written to Excel at row ${i + 1}")
}
```



	A	B	C	D
1	Username	Password	Result	
2	John Doe	ThisIsNotAPas	invalid	
3	John Doe	ThisIsNotAPassword	valid	
4	John Doe	ThisIsNotAPassword	valid	
5				
6				

3..Create internal file having data values for user names and password ; Execute TCs to read test data values from internal file.

Internal File



No.	Username	Password
1	John Doe	ThisIsNotAPass
2	John Doe	invalid
3	John Doe	Not Valid

Script:

```
import static com.kms.katalon.core.testobject.ObjectRepository.findTestObject
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUI
import com.kms.katalon.core.testdata.TestDataFactory as TestDataFactory
import java.nio.file.Files
import java.nio.file.Paths

def ensureScreenshotDirectoryExists(String path) {
    def dir = Paths.get(path)
    if (!Files.exists(dir)) {
        Files.createDirectories(dir)
    }
}

def loginData = TestDataFactory.findTestData('Data Files/Internal Data for login')

def performLogin(String username, String password, String expectedResult, String
screenshotFileName) {
    String projectDir = System.getProperty("user.dir")
    String screenshotDir = projectDir + '/Screenshots/'
    ensureScreenshotDirectoryExists(screenshotDir)
```

```
    String url = 'https://katalon-demo-cura.herokuapp.com/'

    WebUI.openBrowser('')
    WebUI.navigateToUrl(url)
    WebUI.click(findTestObject('Object Repository/Page_CURA Healthcare Service/a_Make
Appointment'))

    WebUI.setText(findTestObject('Object Repository/Page_CURA Healthcare
Service/input_Username_username'), username)
    WebUI.setText(findTestObject('Object Repository/Page_CURA Healthcare
Service/input_Password_password'), password)
    WebUI.click(findTestObject('Object Repository/Page_CURA Healthcare
Service/button_Login'))

    if (expectedResult == 'success') {
        if (WebUI.verifyTextPresent('Make Appointment', false)) {
            println("✅ Login successful [Username: '${username}', Password:
'${password}']")
            WebUI.takeScreenshot(screenshotDir + screenshotFileName)
        } else {
            println("❌ Login failed - Expected success but failed [Username:
'${username}', Password: '${password}']")
            WebUI.takeScreenshot(screenshotDir + 'failed_' + screenshotFileName)
        }
    } else if (expectedResult == 'failure') {
        if (WebUI.verifyTextPresent('Login failed! Please ensure the username and
password are valid.', false)) {
            println("✅ Correctly handled invalid credentials [Username:
'${username}', Password: '${password}']")
            WebUI.takeScreenshot(screenshotDir + screenshotFileName)
        } else {
            println("❌ Error - Expected failure but not detected [Username:
'${username}', Password: '${password}']")
            WebUI.takeScreenshot(screenshotDir + 'failed_' + screenshotFileName)
        }
    }
    WebUI.closeBrowser()
}

for (int i = 1; i <= loginData.getRowNumbers(); i++) {
    String username = loginData.getValue('Username', i)?.trim()
    String password = loginData.getValue('Password', i)?.trim()

    println("Testing Row $i: Username='${username}', Password='${password}'")
    String expectedResult = ('John Doe'.equals(username) &&
'ThisIsNotAPassword'.equals(password)) ? 'success' : 'failure'

    // Generate safe screenshot filename
    String safeUsername = (username ?: 'emptyUser').replaceAll('\\s+', '_')
    String safePassword = (password ?: 'emptyPass').replaceAll('\\s+', '_')
    String fileName = "login_${i}${safeUsername}${safePassword}.png"

    performLogin(username, password, expectedResult, fileName)
```

```
}
```

Lab 13

Exercise:

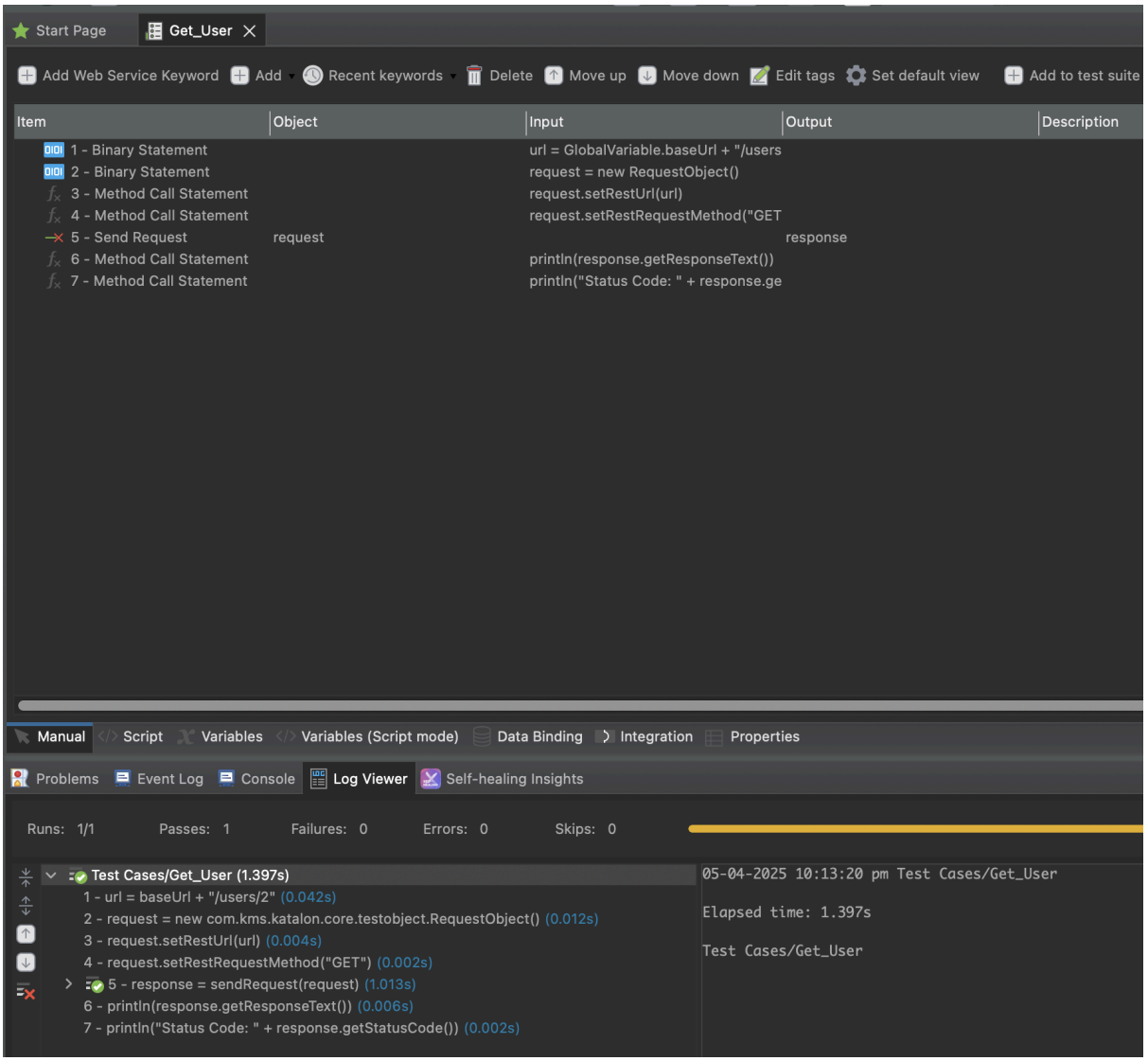
1.Use **https://reqres.in/api** and create Testcase to execute Get, Put,Patch,Post and delete requests also use variable to define url. /*attach all printout here*/

GET

Script:

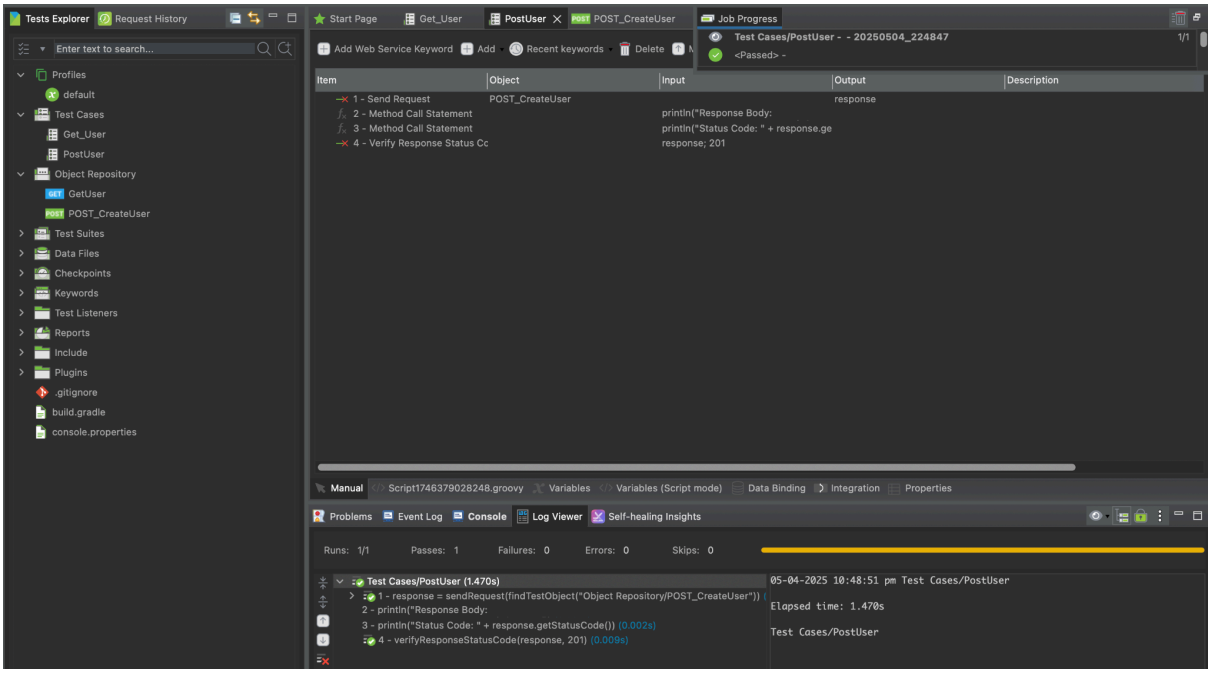
```
import com.kms.katalon.core.testobject.RequestObject

// Dynamically create the URL
String url = GlobalVariable.baseUrl + "/users/2"

// Send GET request
RequestObject request = new RequestObject()
request.setRestUrl(url)
request.setRestRequestMethod("GET")

def response = WS.sendRequest(request)
println(response.getResponseText())
println("Status Code: " + response.getStatusCode())
```

TestCase:

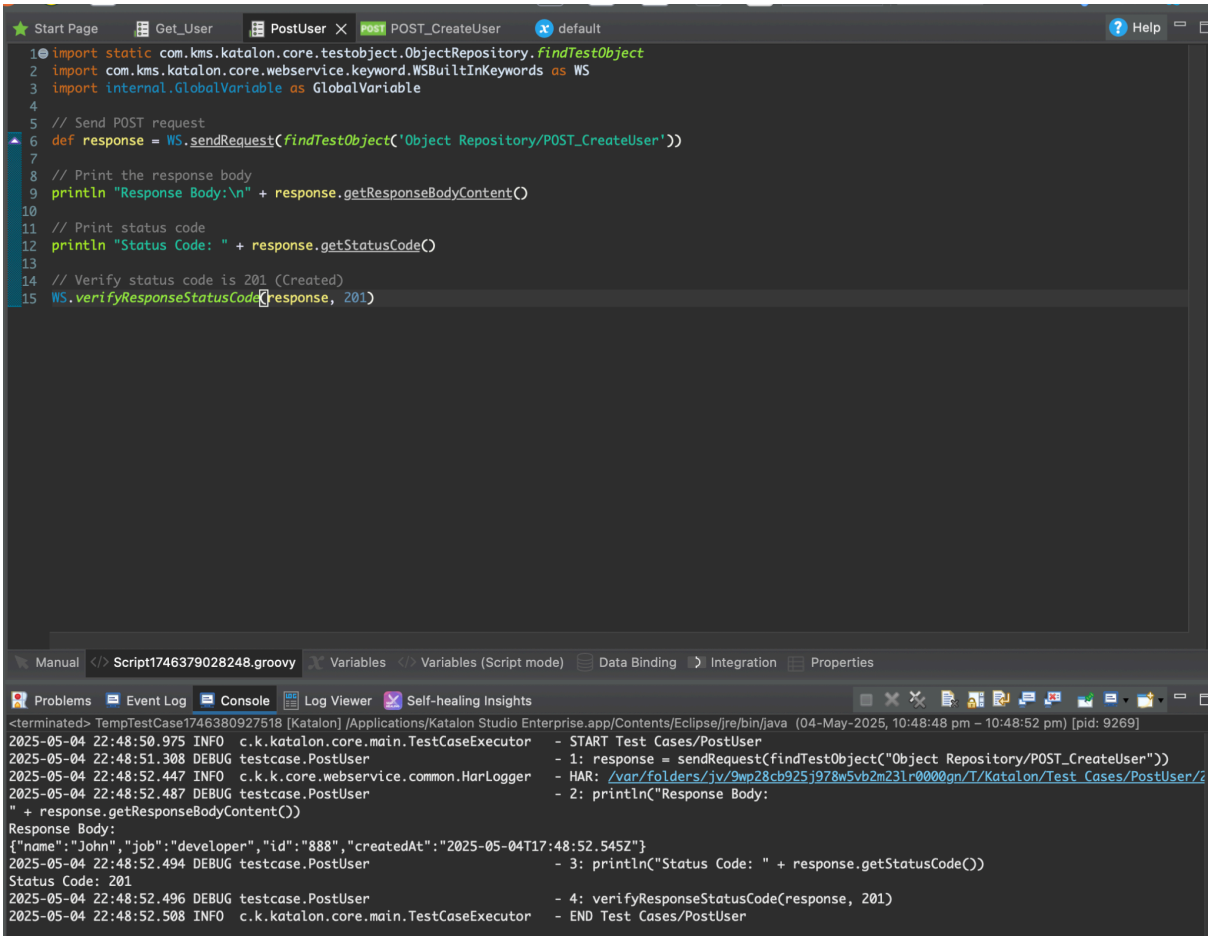


Console:

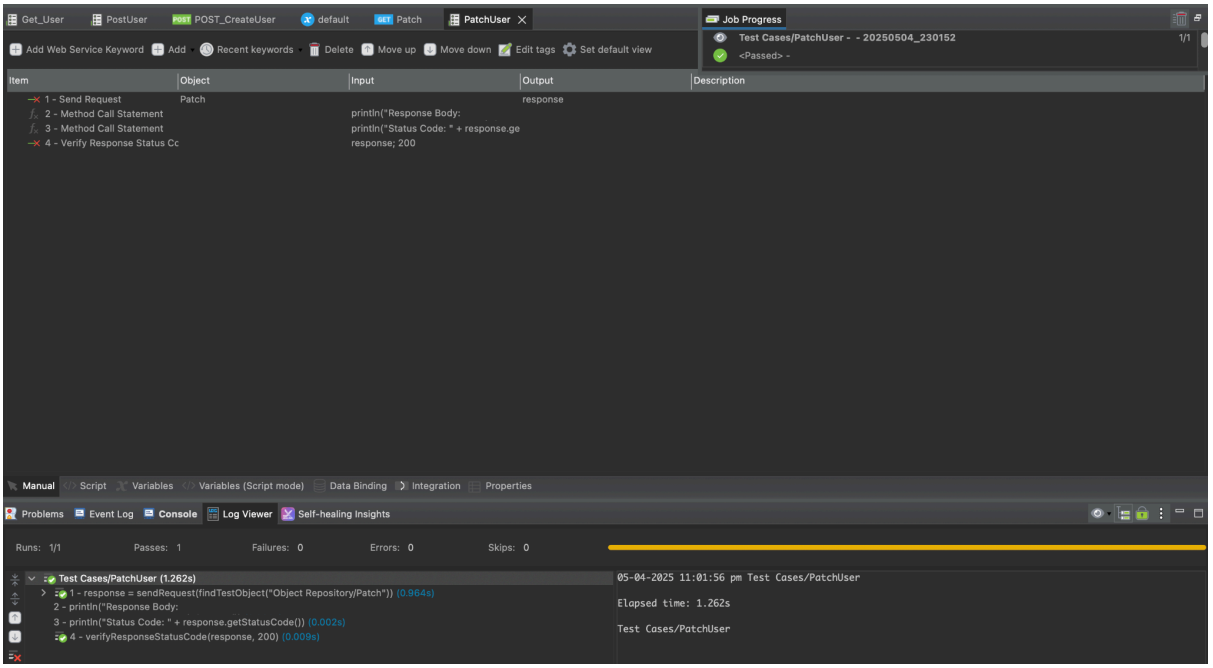
Post User:

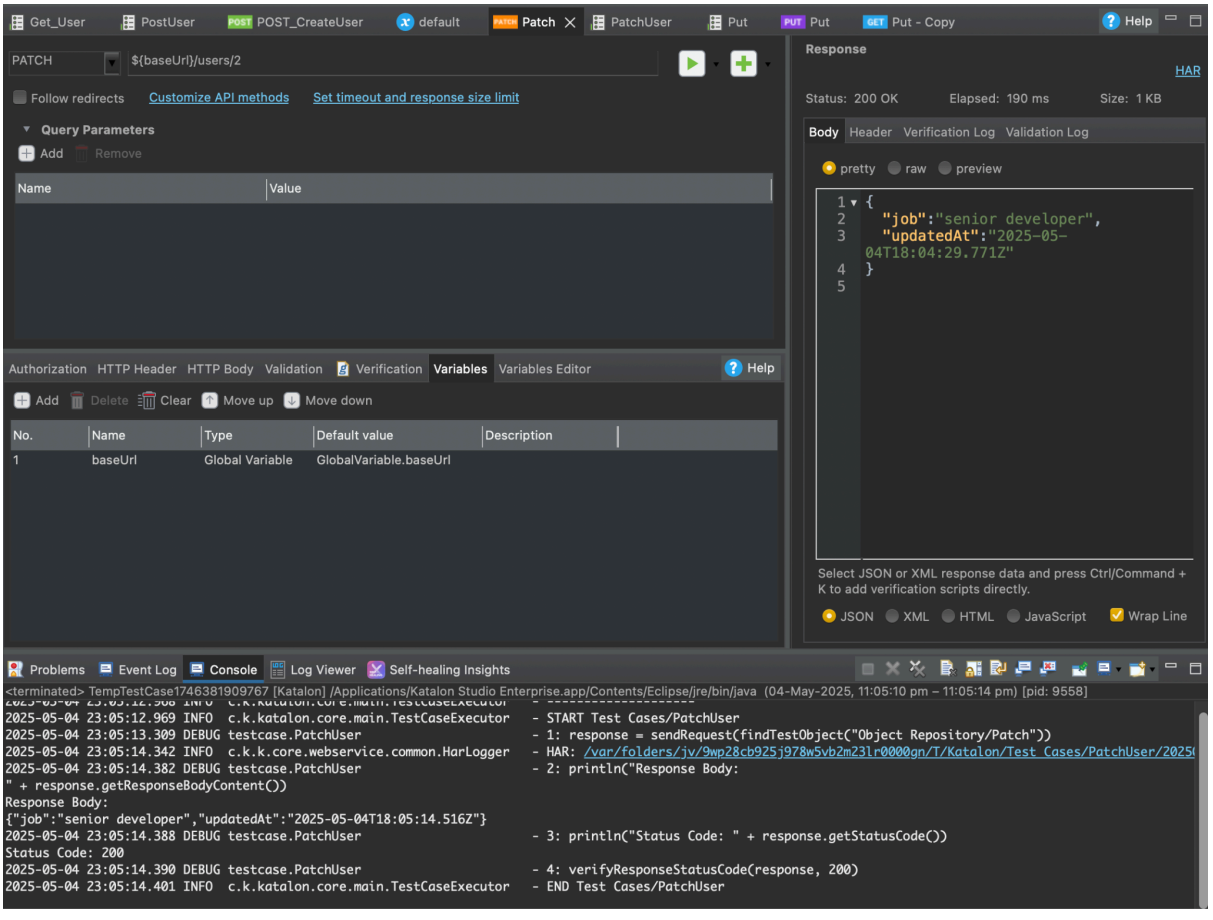


Script & Console:

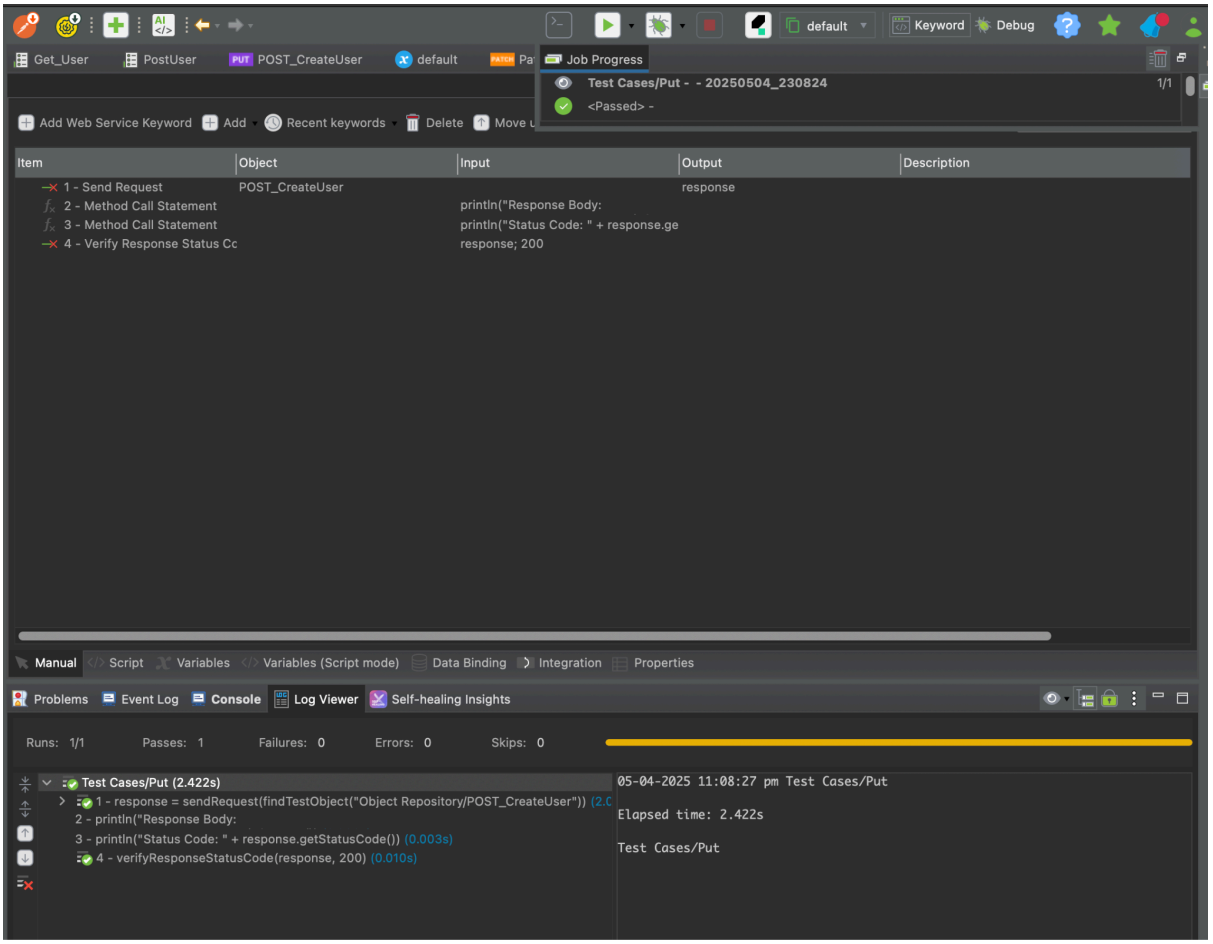


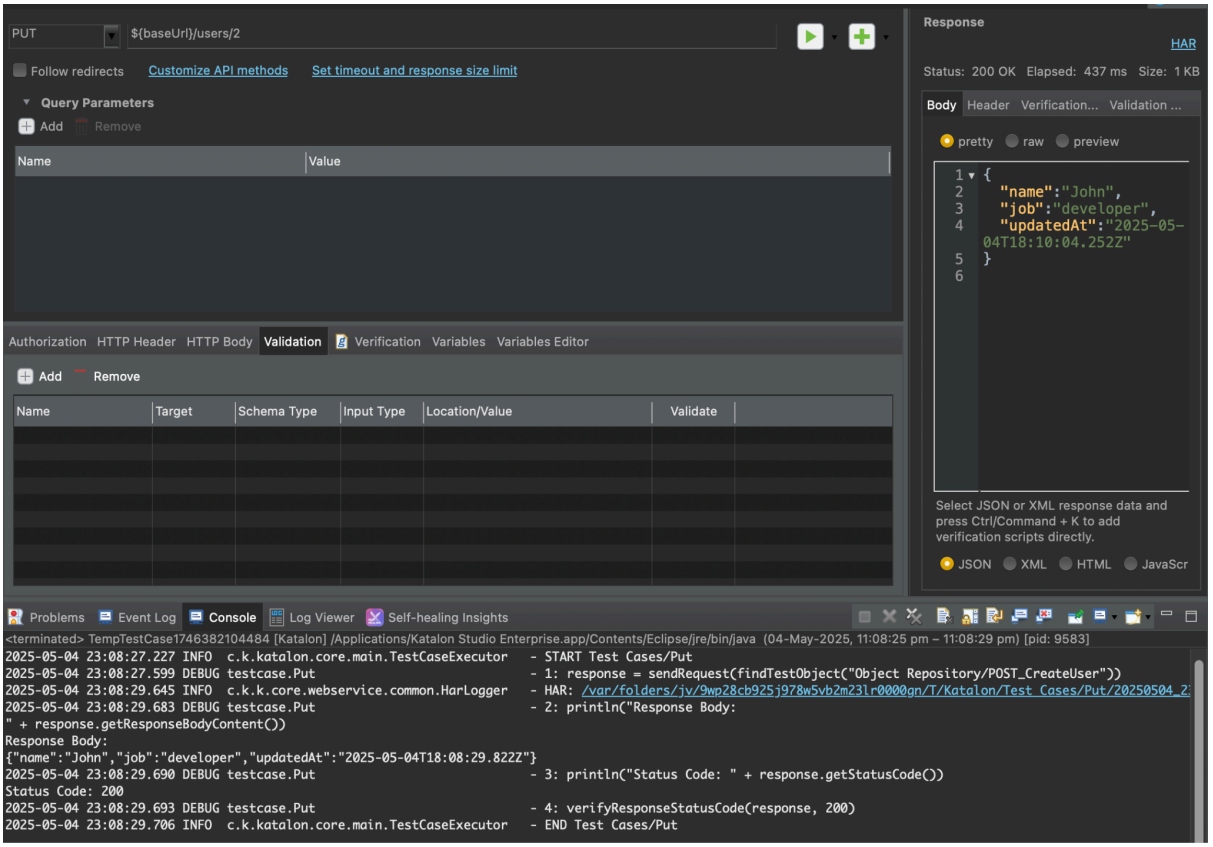
Patch:


Get_User
PostUser
POST_CreateUser
default
Patch
PatchUser
Job Progress
Test Cases/PatchUser - - 20250504_230152
<Passed> -
Add Web Service Keyword
Add
Recent keywords
Delete
Move up
Move down
Edit tags
Set default view
Item
Object
Input
Output
Description
1 - Send Request
Patch
response
2 - Method Call Statement
println("Response Body:
3 - Method Call Statement
println("Status Code: " + response.ge
4 - Verify Response Status Cc
response; 200
Manual
Script
Variables
Variables (Script mode)
Data Binding
Integration
Properties
Problems
Event Log
Console
Log Viewer
Self-healing Insights
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/PatchUser (1.262s)
1 - response = sendRequest(findTestObject("Object Repository/Patch")) (0.964s)
2 - println("Response Body:
3 - println("Status Code: " + response.getStatusCode()) (0.002s)
4 - verifyResponseStatusCode(response, 200) (0.009s)
05-04-2025 11:01:56 pm Test Cases/PatchUser
Elapsed time: 1.262s
Test Cases/PatchUser

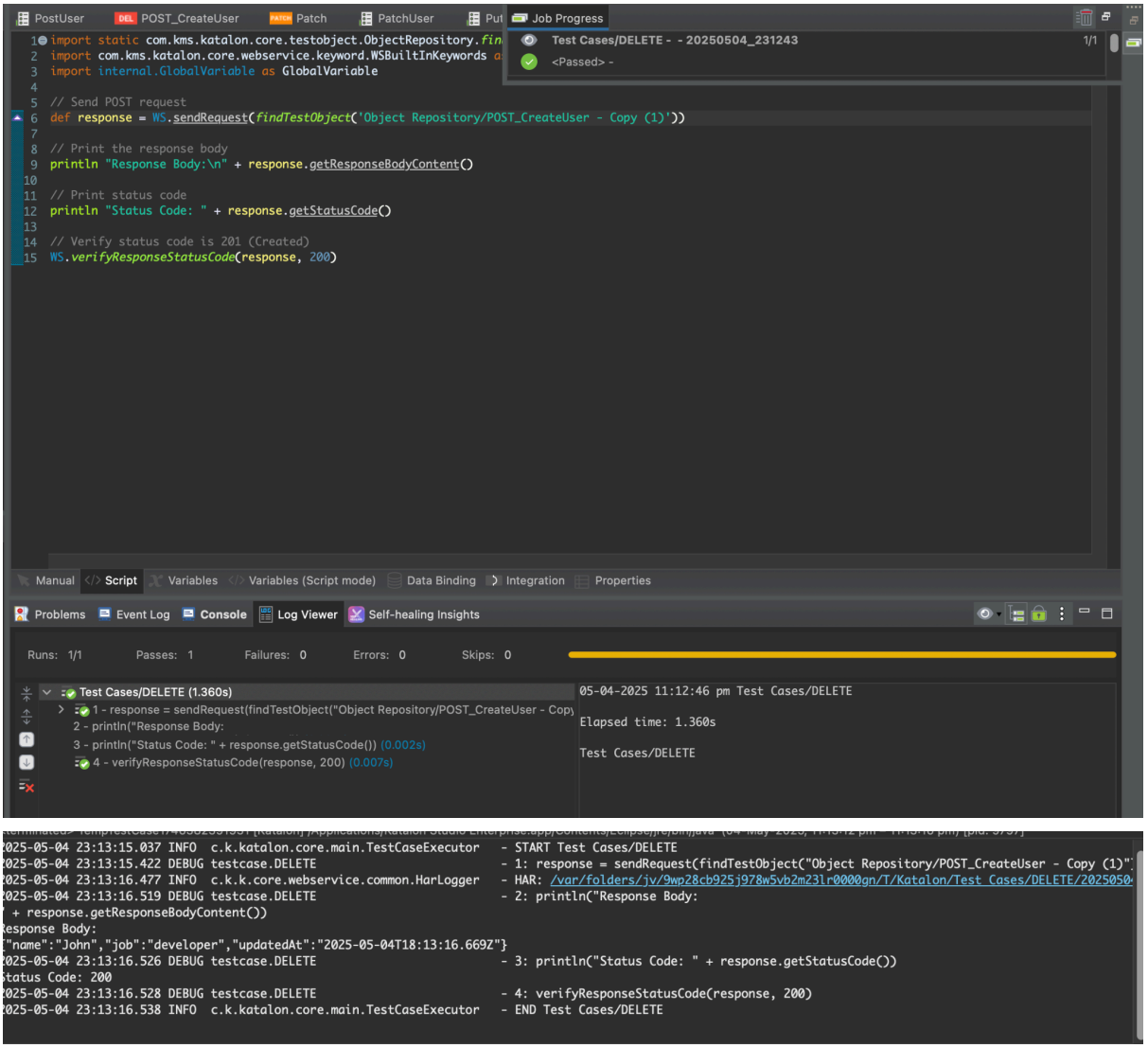

Get_User
PostUser
POST_CreateUser
default
Patch
PatchUser
Put
Put
Put - Copy
Help
PATCH
${baseUrl}/users/2
Follow redirects
Customize API methods
Set timeout and response size limit
Query Parameters
Add
Remove
Name
Value
Authorization
HTTP Header
HTTP Body
Validation
Verification
Variables
Variables Editor
Help
Add
Delete
Clear
Move up
Move down
No.
Name
Type
Default value
Description
1
baseUrl
Global Variable
GlobalVariable.baseUrl
Response
HAR
Status: 200 OK
Elapsed: 190 ms
Size: 1 KB
Body
Header
Verification Log
Validation Log
pretty
raw
preview
{
"job":"senior developer",
"updatedAt":"2025-05-04T18:04:29.771Z"
}
Select JSON or XML response data and press Ctrl/Command + K to add verification scripts directly.
JSON
XML
HTML
JavaScript
Wrap Line
Problems
Event Log
Console
Log Viewer
Self-healing Insights
<terminated> TempTestCase1746381909767 [Katalon] /Applications/Katalon Studio Enterprise.app/Contents/Eclipse/jre/bin/java (04-May-2025, 11:05:10 pm – 11:05:14 pm) [pid: 9558]
2025-05-04 23:05:12.969 INFO c.k.katalon.core.main.TestCaseExecutor - START Test Cases/PatchUser
2025-05-04 23:05:13.309 DEBUG testcase.PatchUser - 1: response = sendRequest(findTestObject("Object Repository/Patch"))
2025-05-04 23:05:14.342 INFO c.k.k.core.webservice.common.HarLogger - HAR: /var/folders/jv/9wp28cb925j978w5vb2m23lr0000gn/T/Katalon/Test Cases/PatchUser/2025
2025-05-04 23:05:14.382 DEBUG testcase.PatchUser - 2: println("Response Body:
" + response.getResponseBodyContent())
Response Body:
{"job":"senior developer","updatedAt":"2025-05-04T18:05:14.516Z"}
2025-05-04 23:05:14.388 DEBUG testcase.PatchUser - 3: println("Status Code: " + response.getStatusCode())
Status Code: 200
2025-05-04 23:05:14.390 DEBUG testcase.PatchUser - 4: verifyResponseStatusCode(response, 200)
2025-05-04 23:05:14.401 INFO c.k.katalon.core.main.TestCaseExecutor - END Test Cases/PatchUser

Put:


Get_User
PostUser
POST_CreateUser
default
Job Progress
Test Cases/Put - - 20250504_230824
<Passed> -
1/1
Add Web Service Keyword
Add
Recent keywords
Delete
Item
Object
Input
Output
Description
1 - Send Request
POST_CreateUser
response
2 - Method Call Statement
println("Response Body:
3 - Method Call Statement
println("Status Code: " + response.ge
4 - Verify Response Status Cc
response; 200
Manual
Script
Variables
Variables (Script mode)
Data Binding
Integration
Properties
Problems
Event Log
Console
Log Viewer
Self-healing Insights
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/Put (2.422s)
1 - response = sendRequest(findTestObject("Object Repository/POST_CreateUser"))
2 - println("Response Body:
3 - println("Status Code: " + response.getStatusCode()) (0.003s)
4 - verifyResponseStatusCode(response, 200) (0.010s)
05-04-2025 11:08:27 pm Test Cases/Put
Elapsed time: 2.422s
Test Cases/Put


PUT
${baseUrl}/users/2
Follow redirects
Customize API methods
Set timeout and response size limit
Query Parameters
Add
Remove
Name
Value
Authorization
HTTP Header
HTTP Body
Validation
Verification
Variables
Variables Editor
Add
Remove
Name
Target
Schema Type
Input Type
Location/Value
Validate
Response
HAR
Status: 200 OK
Elapsed: 437 ms
Size: 1 KB
Body
Header
Verification...
Validation ...
pretty
raw
preview
{
"name":"John",
"job":"developer",
"updatedAt":"2025-05-04T18:10:04.252Z"
}
Select JSON or XML response data and press Ctrl/Command + K to add verification scripts directly.
JSON
XML
HTML
JavaScr
Problems
Event Log
Console
Log Viewer
Self-healing Insights
<terminated> TempTestCase1746382104484 [Katalon] /Applications/Katalon Studio Enterprise.app/Contents/Eclipse/jre/bin/java (04-May-2025, 11:08:25 pm – 11:08:29 pm) [pid: 9583]
2025-05-04 23:08:27.227 INFO c.k.katalon.core.main.TestCaseExecutor - START Test Cases/Put
2025-05-04 23:08:27.599 DEBUG testcase.Put - 1: response = sendRequest(findTestObject("Object Repository/POST_CreateUser"))
2025-05-04 23:08:29.645 INFO c.k.k.core.webservice.common.HarLogger - HAR: /var/folders/jv/9wp28cb925j978w5vb2m23lr0000gn/T/Katalon/Test Cases/Put/20250504_2
2025-05-04 23:08:29.683 DEBUG testcase.Put - 2: println("Response Body:
" + response.getResponseBodyContent())
Response Body:
{"name":"John","job":"developer","updatedAt":"2025-05-04T18:08:29.822Z"}
2025-05-04 23:08:29.690 DEBUG testcase.Put - 3: println("Status Code: " + response.getStatusCode())
Status Code: 200
2025-05-04 23:08:29.693 DEBUG testcase.Put - 4: verifyResponseStatusCode(response, 200)
2025-05-04 23:08:29.706 INFO c.k.katalon.core.main.TestCaseExecutor - END Test Cases/Put

DELETE:



2.state all status codes along with query types.

HTTP Method	Operation	Status Code	Meaning
GET	Retrieve user	200	OK
POST	Create user	201	Created
PUT	Update full user	200	OK
PATCH	Update partial user	200	OK
DELETE	Delete user	204	No Content

DEPARTMENT OF SOFTWARE ENGINEERING
BACHELORS IN SOFTWARE ENGINEERING
Course Code: SE-309
Course Title: Software Quality Engineering
CEA
TE Batch 2022, Spring Semester 2025
Software Quality Engineering

CLO Covered:

CLO 2: (P3-PLO3) Design and Development of Solution.

Complex problem-solving attributes (CPA) covered (as per PEC - OBA manual – 2019)

CPA-1 Range of Resources: Involve the use of diverse resources (and for this purpose resources includes people, money, equipment, materials, information and technologies).
CPA-2 Level of interaction: Require resolution of significant problems arising from interactions between wide-ranging or conflicting technical, engineering or other issues.
CPA-3 Familiarity: Can extend beyond previous experiences by applying principles-based approaches.

Guidelines and Evaluation Criteria Overview

You are part of a QA team responsible for testing a software application. You can choose **any project of your interest** (e.g., e-commerce website, food delivery app, banking portal, learning management system, IoT device app, etc.). Your task is to **design test cases and validate complete application** using a testing tool of your choice such as **Katalon Studio, Selenium, Cypress, Appium, etc.** Your response should cover the following components:

Project Description

- Briefly describe the project you have chosen (max 200 words).
- Mention the platform (web, mobile, desktop) and core features of the application.

Tool Selection

- Which testing tool are you choosing and why?
- How does it suit the needs of your project (e.g., language support, browser/device compatibility)?

Test Scenarios

- Write **at least 5 test scenarios (**high-level description of **what to test** in a particular feature of an application) that you plan to automate.
- Each should include a clear objective (e.g., “Verify successful login with valid credentials”).

Execution Plan

- How will you run your tests ad why (manually, scheduled runs, CI/CD)?
- Which environments (browsers, OS, devices) will you test on?

Challenges & Solutions

- Identify at least two challenges in your test automation project.
- Suggest practical solutions or mitigation plans.

Optional Bonus

- Propose how you would generate test reports and integrate with tools like Jenkins, GitHub, TestRail, or Jira.

Submission Guidelines: Please submit your responses along with all evidence i.e. a detailed test summary report, Excel Files,relevant code snippets, and user interface (UI) screenshots to support your testing approach and validate your implementation."

- Platform: Katalon Studio or any other Testing Tool.
- Report Format: The report must strictly follow the Standards[attach summary report, manually created Testcases, snippets and /or files containing data values if any]

Evaluation Breakdown

- Total Marks: 10

DEPARTMENT OF SOFTWARE ENGINEERING
BACHELORS IN SOFTWARE ENGINEERING
Course Code: SE-309
Course Title: Software Quality Engineering
Complex Engineering Problem
TE Batch 2022, Spring Semester 2025

Group Member, Information

Student no.	**Student Roll#**	**Student Name**
S-01	SE-22032	M.Ali Arslan
S-02	SE-22034	Syed ibad
S-03	SE-22046	Samiullah
S-04	SE-22047	Vaiz
S-05	SE-22050	Faez Ansar

Rubric	**CPA**	**Criteria**	**2 Marks (Excellent)**	**1 Mark (Satisfactory)**	**0 Marks (Poor/Not Attempted)**
Resource Utilization & Tool Justification	CPA-1	Tool selection and resource integration	Clear, well-justified tool choice aligned with platform, language, and project needs. Demonstrates effective use of diverse resources (tools, devices, data, collaboration).	Tool selection is somewhat justified; partial match with platform or needs. Limited explanation or partial use of available resources.	No clear justification of tool choice or poor tool-platform alignment. Minimal or no demonstration of resource usage.
Test Coverage & Scenario Design	CPA-2	Test scenarios' relevance and interaction depth	At least 5 well-structured, high-level scenarios that test key workflows, include interdependent features, and handle valid and edge cases.	3–4 scenarios provided. Some test real-world flows but with limited interaction or coverage.	Fewer than 3 scenarios. Scenarios are too basic or generic with minimal relevance.
Problem Solving in Testing Context	CPA-2	Handling of testing challenges	Identifies at least 2 realistic testing challenges and proposes sound, practical solutions or mitigation strategies.	Identifies only 1 challenge or provides general/partial solution.	No relevant challenge identified or no solutions provided.
Principle-Based Approach & Customization	CPA-3	Application of testing strategies	Applies testing principles (e.g., data-driven, boundary, negative testing) and demonstrates tool customization or advanced use.	Applies some principles but shows limited understanding or no customization of tools.	No application of testing principles or tool features beyond basic usage.
Reporting & Integration with Modern Tools	CPA-1 / CPA-3	Test reporting and modern tool integration	Provides test reports and integrates with tools like Jenkins, GitHub, Jira, etc. Shows understanding of automated workflows or dashboards.	Reports provided but lack integration or limited use of external tools.	No test reports or tool integration shown.

LAB 14

Exercises:

1. Test at least 5 features of your application using Katalon Studio.

1. PROJECT DESCRIPTION

In today's digitally driven world, shopping for plants and garden-related services should be as serene and seamless as nurturing a plant. However, many plant e-commerce platforms offer a disjointed experience—either focusing solely on product listings or neglecting customer engagement features like scheduling consultations or personalized service. This leaves users overwhelmed, uncertain, and often dissatisfied with their online plant-buying journey.

A.R LandScape is an innovative, web-based plant e-commerce platform designed to simplify and personalize the process of buying plants and scheduling consultations with expert sellers. With a focus on user-friendly design and smooth functionality, **A.R LandScape** makes plant shopping accessible, educational, and convenient. Whether users are looking for indoor plants, outdoor garden solutions, or expert gardening advice, **A.R LandScape** unites it all in one clean, easy-to-use system.

1.1 Platform:

A.R LandScape is a **web-based application** optimized for use across modern browsers and devices. Its **responsive interface** ensures a seamless experience on both desktop and mobile without requiring downloads or installations.

1.2 Core Features:

1.2.1 Add to Cart

Users can easily browse through various plant categories and add selected products to their virtual cart. The cart dynamically updates with item count and prices, offering a user-friendly experience while navigating the catalog.

1.2.2 Checkout Process

A.R LandScape features a smooth and secure checkout experience. Users can review cart items, apply discount codes, choose delivery options, and complete payments via secure gateways. The platform ensures clear order confirmation and transaction security.

1.2.3 Schedule Appointment

A.R LandScape allows users to book **consultations or appointments** with expert sellers for plant care advice, personalized garden planning, or custom orders. The feature includes date/time selection, calendar syncing, and automatic reminders to ensure reliability.

1.2.4 Signup

A.R LandScape offers a secure and straightforward **signup** process. New users can register using their email address and set a password to create an account.

1.2.5 Login

Returning users can securely **log in** to access their profiles, order history, saved plants, and scheduled appointments. The system includes validation for incorrect credentials and supports session management for a smooth and secure experience.

2. TOOL SELECTION

For this project, **Katalon Studio** was selected as the primary tool for automated testing. Its ease of use, support for **web automation**, and built-in **Groovy scripting** enabled the team to automate key test scenarios like login, form validation, and cart interactions.

Key reasons for using Katalon:

- Supports multiple browsers (e.g., Chrome, Firefox)
- Includes reusable test steps and data-driven testing
- Generates detailed test reports with screenshots for debugging
- Allows scalable, repeatable test execution

In parallel, **Microsoft Excel** was used for **manual testing**, especially during early development stages. Excel helped in planning, recording, and reviewing manual test cases (e.g., checkout flow, UI validation), including expected vs. actual outcomes.

---

3. TEST SCENARIO

3.1 Add to Cart

- Verify users can add multiple plants to the cart and the correct item count and prices are displayed.
- Test how the cart handles duplicate items, quantity updates, and deletions.
- Validate that product thumbnails, names, and prices display correctly in the cart summary.

3.2 Checkout Process

- Confirm that users can proceed to checkout from the cart.
- Validate form fields (address, payment details), coupon codes, and shipping options.

3.3 Schedule Appointment

- Ensure appointment booking allows users to select a date, time, and preferred expert.
- Validate that scheduled appointments appear in the user’s dashboard or confirmation email.

- Test for edge cases like booking unavailable time slots or submitting forms with missing data.

3.4 Login & Signup

- Validate registration with valid and invalid inputs (e.g., password mismatch, email format).
- Ensure login works correctly with valid credentials and shows errors for incorrect ones.
- Test for duplicate accounts and secure session handling.

---

4. EXECUTION PLAN

The testing approach used **automated testing** to maximize coverage and efficiency:

Automated Testing (Katalon):

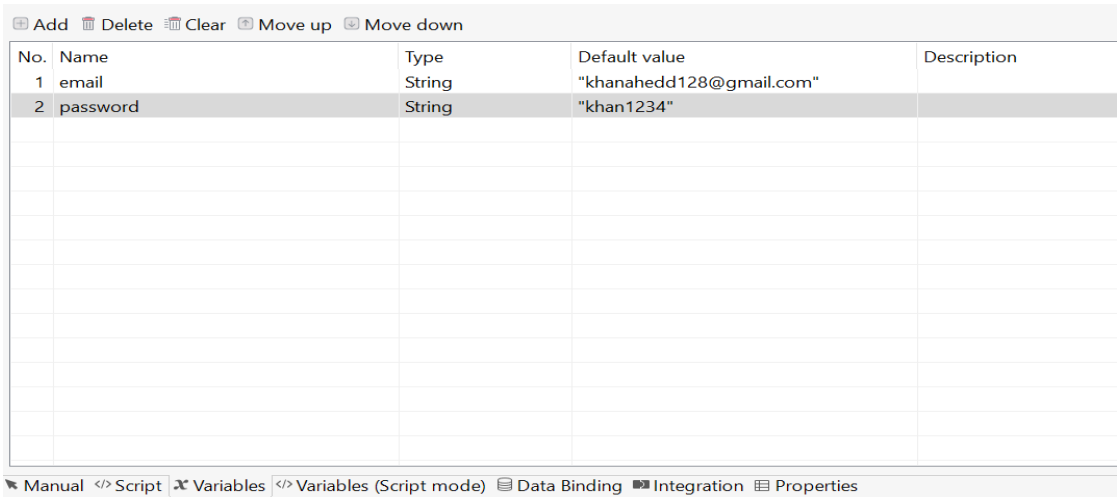
- Login/signup, cart functionality, and appointment booking were automated using test suites.
- **Data-driven testing** was used to simulate different user input combinations for login and checkout.
- Screenshots and detailed logs were generated to validate results and debug issues.

Tests were primarily executed on **Google Chrome (Windows 10)** to reflect typical user environments. Future testing may include cross-browser and mobile responsiveness checks.

TESTING OF ALL FEATURES

FEATURE 1: LOGIN

VARIABLES:



CODE SNIPPET

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

WebUI.openBrowser('')

WebUI.navigateToUrl('http://localhost:5173/')

WebUI.click(findTestObject('Object Repository/Page_TechWare/svg_TechWare_MuiSvgIcon-root MuiSvgIcon-fon_fb816f'))

WebUI.click(findTestObject('Object Repository/Page_TechWare/a_Log In'))

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Sign In_email'), email)

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Sign In_password'), password)

WebUI.selectOptionByValue(findTestObject
('Object Repository/Page_TechWare/select_Select Your RoleAdminSellerBuyer'), 'Seller',
true)

String baseDir = 'C:/Users/Technezo/Pictures/Screenshots/'
```

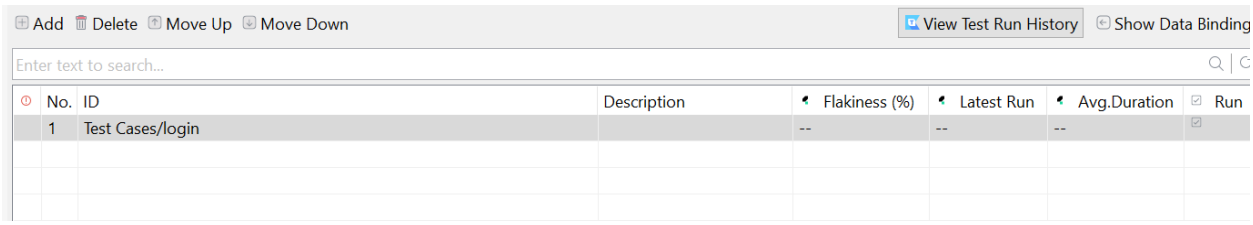
```
if ( email != 'khanahedd128@gmail.com' || password != 'khan1234' ) {

    String invalidShot = baseDir + 'invalid_' + System.currentTimeMillis() + '.png'
    WebUI.takeScreenshot(invalidShot)

    KeywordUtil.markFailedAndStop("✕ Invalid credentials supplied: ${email} / ${password}")
} else {

    String validShot = baseDir + 'valid_' + System.currentTimeMillis() + '.png'
    WebUI.takeScreenshot(validShot)

    WebUI.click(findTestObject('Page_TechWare/button_Sign In'))


    WebUI.waitForElementPresent(findTestObject('Page_TechWare/div_DashboardWelcome'), 10, FailureHandling.OPTIONAL)

    // take a post-login screenshot
    String postLoginShot = baseDir + 'postLogin_' + System.currentTimeMillis() + '.png'
    WebUI.takeScreenshot(postLoginShot)
}
WebUI.delay(3)
WebUI.closeBrowser()
```

MANUAL TAB

Item	Object	Input
1 - Open Browser		""
2 - Navigate To Url		"http://localhost:5173/"
3 - Click	svg_TechWare_MuiSvgIcon-root Mu	
4 - Click	a_Log In	
5 - Set Text	input_Sign In_email	email
6 - Set Text	input_Sign In_password	password
7 - Select Option By Value	select_Select Your RoleAdminSellerB	"Seller"; true
8 - Binary Statement		baseDir = "C:/Users/Technezo/Pictures/Screenshots/"
9 - If Statement		email != "khanahedd128@gmail.com" || password != "khan1234"
10 - Else Statement		
11 - Delay		3
12 - Close Browser		

Manual | Script | Variables | Variables (Script mode) | Data Binding | Integration | Properties

TEST SUITE:



TEST SUMMARY REPORT:

login

Execution Environment

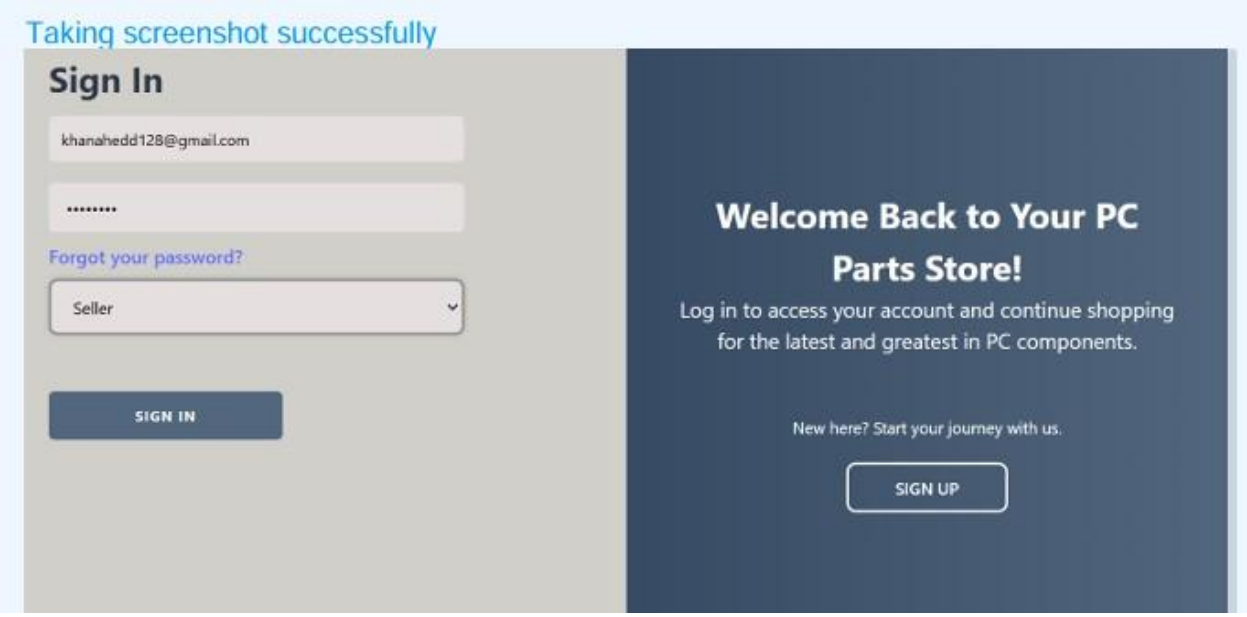
Host name	Technezo - DESKTOP-2SQFQEV
Local OS	Windows 10 64bit
Katalon version	10.1.1.0
Browser	Chrome 135.0.7049.115
Device name	

Summary

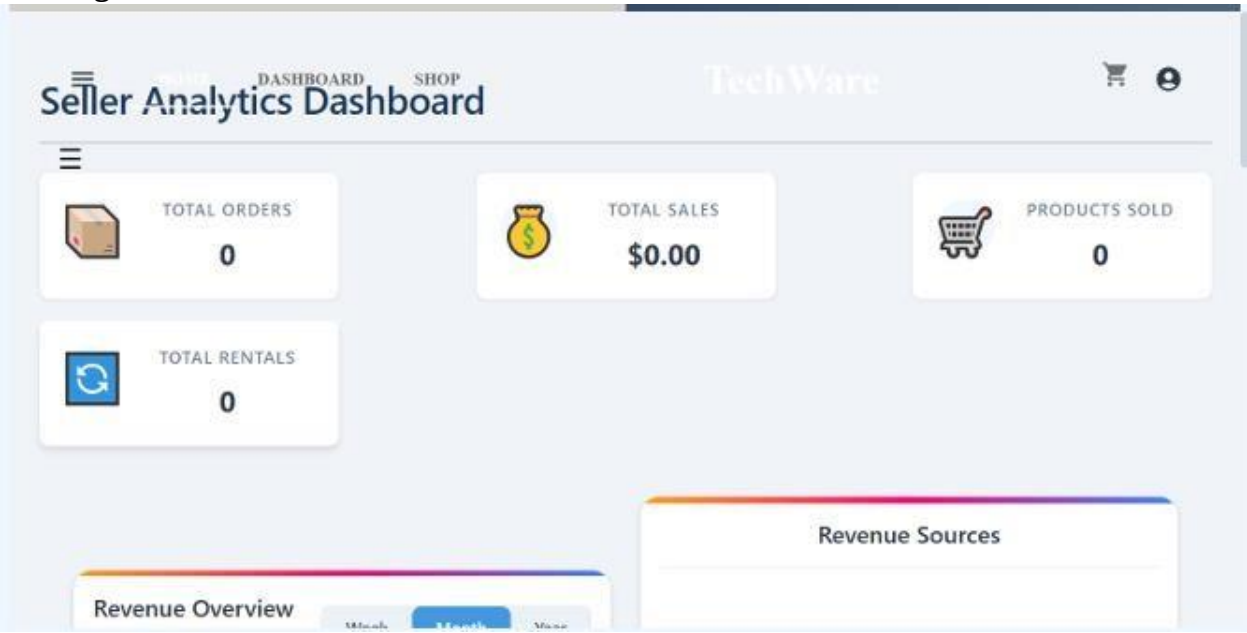
ID	Test Suites/login		
Description			
Total	1		
Passed	1	Failed	0
Error	0	Incomplete	0
Skipped	0		
Start	06-05-2025 16:31:13	End	06-05-2025 16:31:39
Elapsed	26.112s		

#	ID	Description	Status
1	Test Cases/login		PASSED

SCREENSHOTS OF SUCCESSFUL LOGIN



Post-login Screen



FEATURE 2: SIGN UP

VARIABLES:

Add Delete Clear Move up Move down

No.	Name	Type	Default value
1	username	String	"Saleem habib"
2	email	String	"saleemahmed125@gmail.com"
3	password	String	"saleem123"
4	confirmPwd	String	"saleem123"
5	Address	String	"Multan"
6	Phone	String	"03156719123"

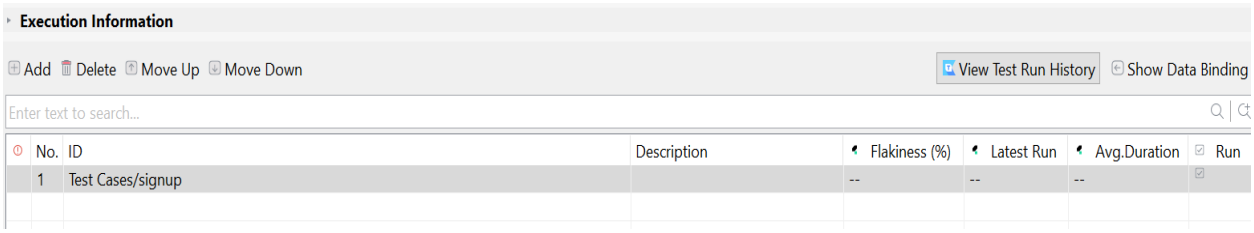
MANUAL:

Item	Object	Input
1 - Open Browser		""
2 - Navigate To Url		"http://localhost:5173/"
3 - Binary Statement		baseDir = "C:/Users/Technezo/Pictures/Screenshots/"
4 - Click	svg_TechWare_MuiSvgIcon-root	
5 - Click	a_Sign Up	
6 - Click	button_Sign Up	
7 - Set Text	input_Create Account_name	username
8 - Set Text	input_Create Account_email	email
9 - Set Text	input_Create Account_password	password
10 - Set Text	input_Create Account_confirmPa:	confirmPwd
11 - Set Text	input_Create Account_phoneNo	Phone
12 - Set Text	input_Create Account_address	Address
13 - Select Option By Value	select_Select Your RoleAdminSell	"Buyer"; true
14 - Delay		2
15 - Click	button_Sign Up_1	
16 - Take Screenshot		

SCRIPT:

```
WebUI.openBrowser('')

WebUI.navigateToUrl('http://localhost:5173/')

String baseDir = 'C:/Users/Technezo/Pictures/Screenshots/'

WebUI.click(findTestObject('Object Repository/Page_TechWare/svg_TechWare_MuiSvgIcon-root MuiSvgIcon-fon_fb816f'))

WebUI.click(findTestObject('Object Repository/Page_TechWare/a_Sign Up'))

WebUI.click(findTestObject('Object Repository/Page_TechWare/button_Sign Up'))

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Create Account_name'), username)

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Create Account_email'), email)

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Create Account_password'), password)

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Create Account_confirmPassword'), confirmPwd)

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Create Account_phoneNo'), Phone)

WebUI.setText(findTestObject('Object Repository/Page_TechWare/input_Create Account_address'), Address)

WebUI.selectOptionByValue(findTestObject('Object Repository/Page_TechWare/select_Select Your RoleAdminSellerBuyer'), 'Buyer',
    true)


WebUI.delay(2)
WebUI.click(findTestObject('Object Repository/Page_TechWare/button_Sign Up_1'))

WebUI.takeScreenshot(baseDir)

WebUI.delay(5)

WebUI.closeBrowser()
```

TEST SUITE:



TEST SUMMARY REPORT:

SigningUp

Execution Environment

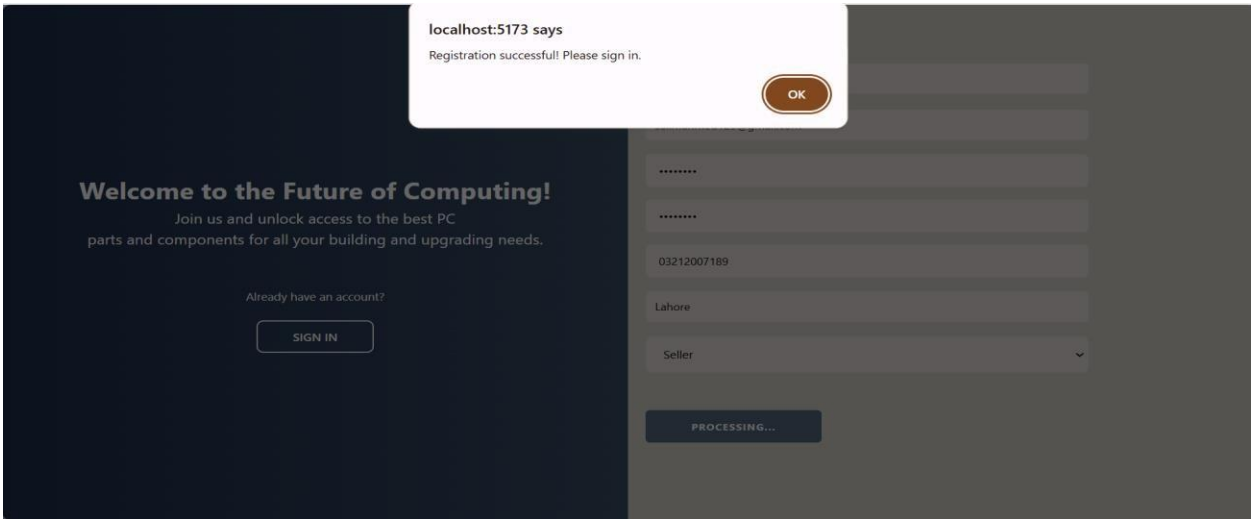
Host name	Technezo - DESKTOP-2SQFQEV
Local OS	Windows 10 64bit
Katalon version	10.1.1.0
Browser	Chrome 135.0.7049.115
Device name	

Summary

ID	Test Suites/SigningUp		
Description			
Total	1		
Passed	1	Failed	0
Error	0	Incomplete	0
Skipped	0		
Start	06-05-2025 15:38:28	End	06-05-2025 15:38:54
Elapsed	25.959s		

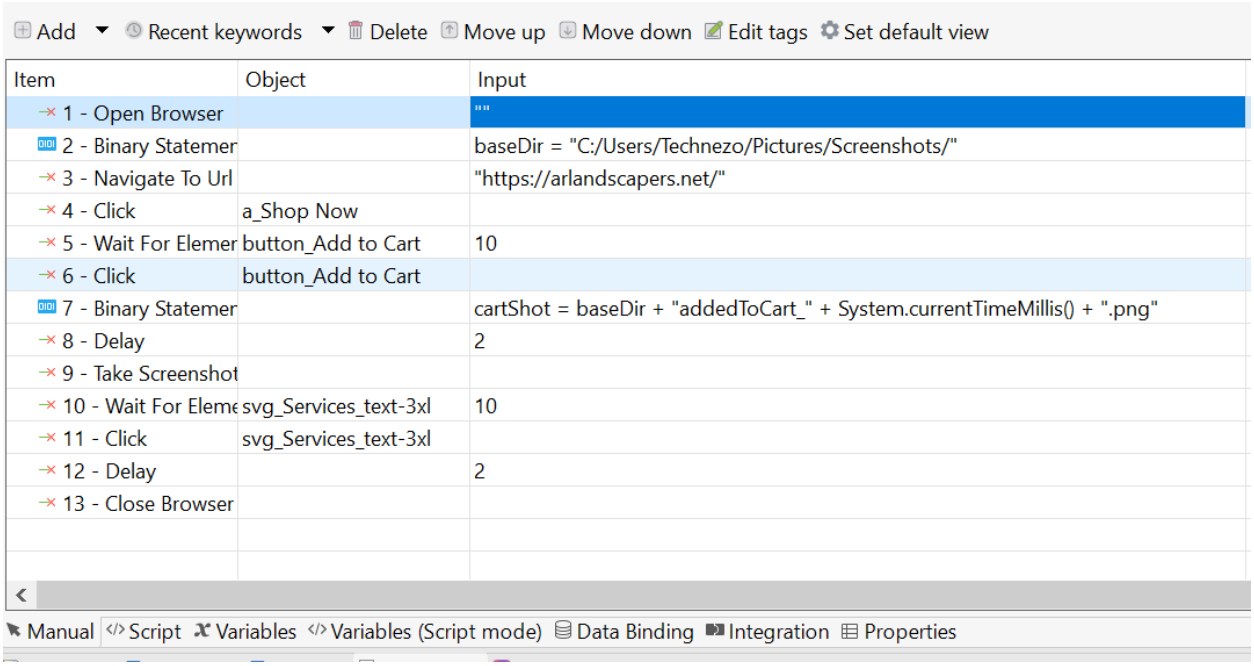
#	ID	Description	Status
1	Test Cases/signup		PASSED

SCREENSHOT OF SUCCESSFUL SIGNUP:



FEATURE 3: ADD TO CART

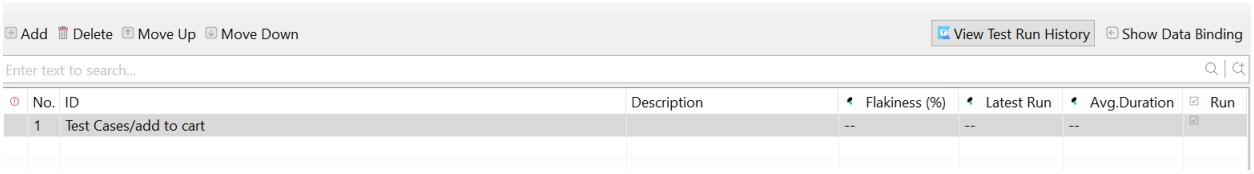
MANUAL:



SCRIPT:

```groovy
import static com.kms.katalon.core.testobject.ObjectRepository.findTestObject
import com.kms.katalon.core.webui.keyword.WebUiBuiltInKeywords as WebUI

// --- Setup ---
WebUI.openBrowser('')
String baseDir = 'C:/Users/Technezo/Pictures/Screenshots/'
WebUI.navigateToUrl('https://arlandscapers.net/')

WebUI.click(findTestObject('Page_A.R. Landscape - Transform Your Outdoors/a_Shop Now'))
WebUI.waitForElementClickable(
    findTestObject('Page_Shop Landscaping Essentials  A.R. Landscape/button_Add to Cart'),
    10
)
WebUI.click(findTestObject('Page_Shop Landscaping Essentials  A.R. Landscape/button_Add to Cart'))

String cartShot = baseDir + 'addedToCart_' + System.currentTimeMillis() + '.png'
WebUI.delay(2)
WebUI.takeScreenshot(cartShot)


WebUI.waitForElementClickable(
    findTestObject('Page_Shop Landscaping Essentials  A.R. Landscape/svg_Services_text-3xl'),
    10
)
WebUI.click(findTestObject('Page_Shop Landscaping Essentials  A.R. Landscape/svg_Services_text-3xl'))

WebUI.delay(2)
WebUI.closeBrowser()
```

TEST SUTE:



TEST SUMMARY REPORT:

add To cart

Execution Environment

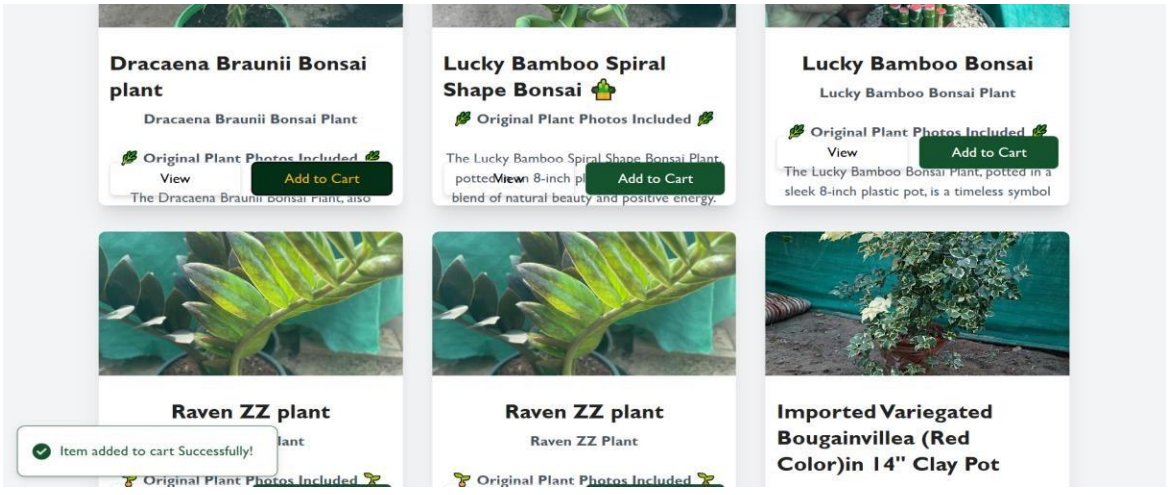
Host name	Technezo - DESKTOP-2SQFQEV
Local OS	Windows 10 64bit
Katalon version	10.1.1.0
Browser	Chrome 135.0.7049.115
Device name	

Summary

ID	Test Suites/add To cart		
Description			
Total	1		
Passed	1	Failed	0
Error	0	Incomplete	0
Skipped	0		
Start	06-05-2025 17:19:11	End	06-05-2025 17:19:35
Elapsed	24.660s		

#	ID	Description	Status
1	Test Cases/add to cart		PASSED

SCREENSHOT:



FEATURE 4: CHECKOUT:

MANUAL:

Add | Recent keywords | Delete | Move up | Move down | Edit tags | Set default view

Item	Object	Input	Output	Description
1 - Open Browser		""		
2 - Navigate To Url		"https://arlandscapers.ne		
3 - Click	a_Shop Now			
4 - Click	button_Pots			
5 - Delay		1		
6 - Click	button_Plants			
7 - Delay		1		
8 - Click	button_Add to Cart			
9 - Delay		1		
10 - Click	svg_Services_text-3xl			
11 - Click	button_Proceed to Checl			
12 - Set Text	input_Full name_usernam	"Muhammad Ali"		
13 - Set Text	input_Email_email	"arsalanali873@gmail.co		
14 - Set Text	input_Phone Number_ph	"03212007013"		
15 - Set Text	input_Address_address	"Karachi"		
16 - Set Text	input_Special Instruction	"bring to my home"		

Manual | Script | Variables | Variables (Script mode) | Data Binding | Integration | Properties

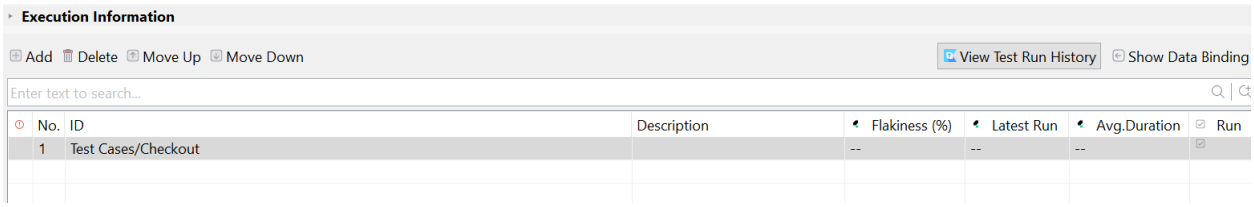
IF 17 - If Statement		WebUI.verifyElementPres
18 - Click	li_Sindh	
19 - Wait For Eleme	button_Place Your Order	10
20 - Enhanced Click	button_Place Your Order	
21 - Delay		3
22 - Close Browser		

Manual | Script | Variables | Variables (Script mode) | Data Binding

SCRIPT:

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

WebUI.openBrowser('')

WebUI.navigateToUrl('https://arlandscapers.net/')

// Navigate to Shop
WebUI.click(findTestObject('Object Repository/Page_A.R. Landscape - Transform Your Outdoors/a_Shop Now'))

// Browse Products
WebUI.click(findTestObject('Object Repository/Page_Shop Landscaping Essentials  A.R. Landscape/button_Pots'))
WebUI.delay(1)
WebUI.click(findTestObject('Object Repository/Page_Shop Landscaping Essentials  A.R. Landscape/button_Plants'))
WebUI.delay(1)

// Add Item to Cart
WebUI.click(findTestObject('Object Repository/Page_Shop Landscaping Essentials  A.R. Landscape/button_Add to Cart'))
WebUI.delay(1)

// Go to Cart
WebUI.click(findTestObject('Object Repository/Page_Shop Landscaping Essentials  A.R. Landscape/svg_Services_text-3xl'))

// Proceed to Checkout
WebUI.click(findTestObject('Object Repository/Page_Shopping Cart  A.R. Landscape/button_Proceed to Checkout'))

// Fill Checkout Form
WebUI.setText(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/input_Full name_username'), 'Muhammad Ali')
WebUI.setText(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/input_Email_email'), 'arsalanali873@gmail.com')
WebUI.setText(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/input_Phone Number_phoneNumber'), '03212007013')
WebUI.setText(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/input_Address_address'), 'Karachi')
WebUI.setText(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/input_Special Instruction (Optional)_instruction'),
'bring to my home')

// Close overlay if present
if (WebUI.verifyElementPresent(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/div_A. R Landscape_MuiBackdrop-root'),
    5, FailureHandling.OPTIONAL))
{
    WebUI.click(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/div_A. R Landscape_MuiBackdrop-root'))
    WebUI.waitForElementNotPresent(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/div_A. R Landscape_MuiBackdrop-root'
}

// Select Province
WebUI.click(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/li_Sindh'))

// Wait and submit order
WebUI.waitForElementClickable(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/button_Place Your Order'), 10)
WebUI.enhancedClick(findTestObject('Object Repository/Page_Checkout  A.R. Landscape/button_Place Your Order'))

// Close browser
WebUI.delay(3)
WebUI.closeBrowser()
```

TEST SUITE:



TEST SUMMARY REPORT:

Checkout

Execution Environment

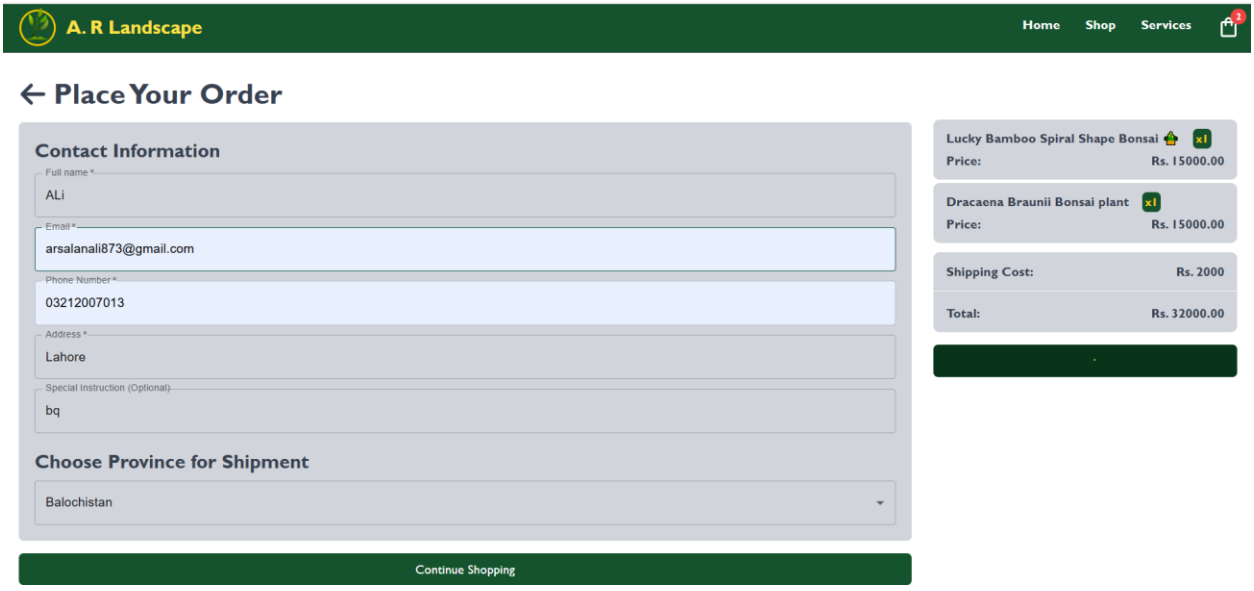
Host name	Technezo - DESKTOP-2SQFQEV
Local OS	Windows 10 64bit
Katalon version	10.1.1.0
Browser	Chrome 135.0.7049.115
Device name	

Summary

ID	Test Suites/Checkout		
Description			
Total	1		
Passed	1	Failed	0
Error	0	Incomplete	0
Skipped	0		
Start	06-05-2025 14:52:07	End	06-05-2025 14:52:49
Elapsed	41.437s		

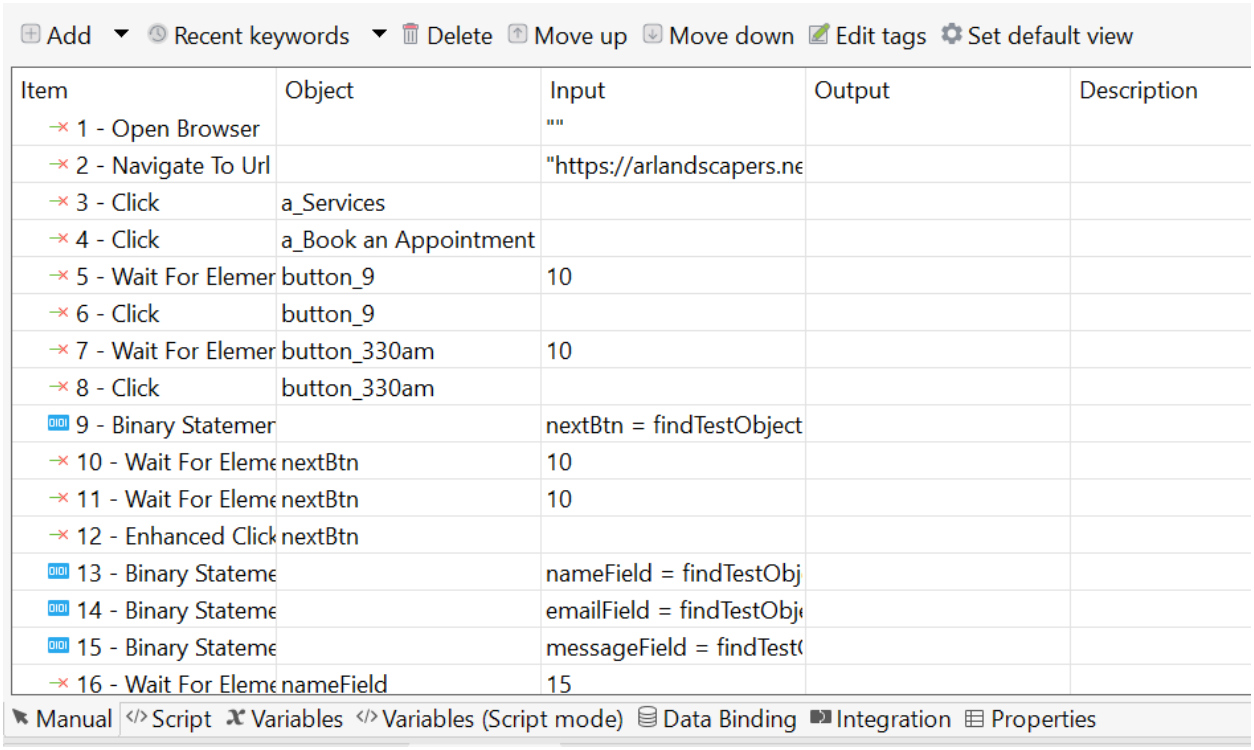
#	ID	Description	Status
1	Test Cases/Checkout		PASSED

SCREENSHOT:



FEATURE 5: BOOK APPOINTMENT:

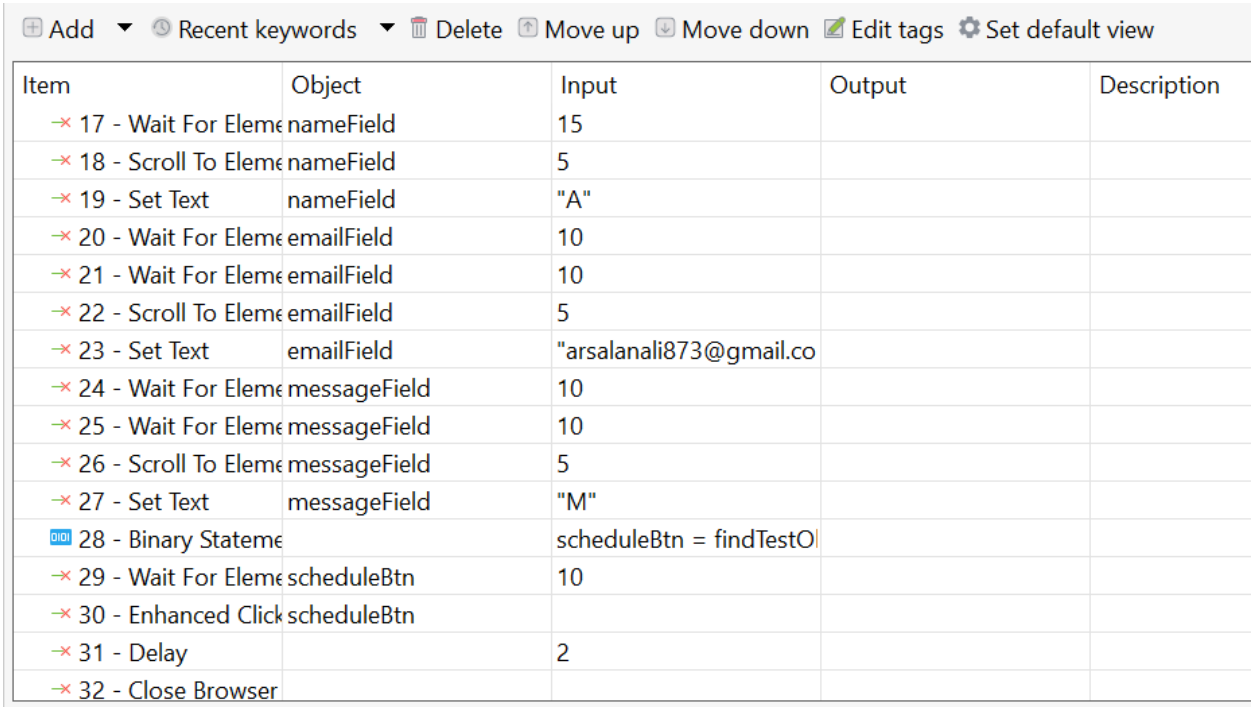
MANUAL:



Add | Recent keywords | Delete | Move up | Move down | Edit tags | Set default view

Item	Object	Input	Output	Description
1 - Open Browser		""		
2 - Navigate To Url		"https://arlandscapers.ne		
3 - Click	a_Services			
4 - Click	a_Book an Appointment			
5 - Wait For Elemer	button_9	10		
6 - Click	button_9			
7 - Wait For Elemer	button_330am	10		
8 - Click	button_330am			
9 - Binary Statemer		nextBtn = findTestObject		
10 - Wait For Eleme	nextBtn	10		
11 - Wait For Eleme	nextBtn	10		
12 - Enhanced Click	nextBtn			
13 - Binary Stateme		nameField = findTestObj		
14 - Binary Stateme		emailField = findTestObj		
15 - Binary Stateme		messageField = findTest(		
16 - Wait For Eleme	nameField	15		

Manual | Script | Variables | Variables (Script mode) | Data Binding | Integration | Properties



Add | Recent keywords | Delete | Move up | Move down | Edit tags | Set default view

Item	Object	Input	Output	Description
17 - Wait For Eleme	nameField	15		
18 - Scroll To Eleme	nameField	5		
19 - Set Text	nameField	"A"		
20 - Wait For Eleme	emailField	10		
21 - Wait For Eleme	emailField	10		
22 - Scroll To Eleme	emailField	5		
23 - Set Text	emailField	"arsalanali873@gmail.co		
24 - Wait For Eleme	messageField	10		
25 - Wait For Eleme	messageField	10		
26 - Scroll To Eleme	messageField	5		
27 - Set Text	messageField	"M"		
28 - Binary Stateme		scheduleBtn = findTestO		
29 - Wait For Eleme	scheduleBtn	10		
30 - Enhanced Click	scheduleBtn			
31 - Delay		2		
32 - Close Browser				

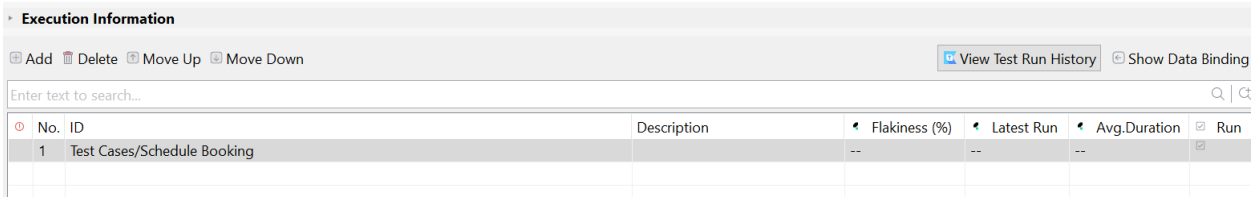
SCRIPT:

```
import static com.kms.katalon.core.checkpoint.CheckpointFactory.findCheckpoint

WebUI.openBrowser('')
WebUI.navigateToUrl('https://arlandscapers.net/')

// Go to Services → Book Appointment
WebUI.click(findTestObject('Page_A.R. Landscape - Transform Your Outdoors/a_Services'))
WebUI.click(findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/a_Book an Appointment'))

// Choose date and time
WebUI.waitForElementClickable(findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/button_9'), 10)
WebUI.click(findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/button_9'))

WebUI.waitForElementClickable(findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/button_330am'), 10)
WebUI.click(findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/button_330am'))

// Wait for the Next button to appear and be clickable
def nextBtn = findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/button_Next')
WebUI.waitForElementVisible(nextBtn, 10)
WebUI.waitForElementClickable(nextBtn, 10)
WebUI.enhancedClick(nextBtn)      // more robust than WebUI.click()
```

```
// Fill out the form  – but only after the form fields are ready
def nameField    = findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/input__full_name')
def emailField   = findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/input__email')
def messageField = findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/textarea_M')

// Wait for the name field to appear
WebUI.waitForElementVisible(nameField, 15)
WebUI.waitForElementClickable(nameField, 15)
WebUI.scrollToElement(nameField, 5)
WebUI.setText(nameField, 'A')

// Wait for the email field
WebUI.waitForElementVisible(emailField, 10)
WebUI.waitForElementClickable(emailField, 10)
WebUI.scrollToElement(emailField, 5)
WebUI.setText(emailField, 'arsalanali873@gmail.com')

// **This is your problematic step** – add waits before typing
WebUI.waitForElementVisible(messageField, 10)
WebUI.waitForElementClickable(messageField, 10)
WebUI.scrollToElement(messageField, 5)
```

```
WebUI.setText(messageField, 'M')

// Finally click “Schedule Event”
def scheduleBtn = findTestObject('Page_A.R. Landscape Services - Transform Yo_7daabb/span_Schedule Event')
WebUI.waitForElementClickable(scheduleBtn, 10)
WebUI.enhancedClick(scheduleBtn)

// Tear down
WebUI.delay(2)
WebUI.closeBrowser()
```

TEST SUITE:



TEST SUMMARY REPORT:

booking

Execution Environment

Host name	Technezo - DESKTOP-2SQFQEV
Local OS	Windows 10 64bit
Katalon version	10.1.1.0
Browser	Chrome 135.0.7049.115
Device name	

Summary

ID	Test Suites/booking		
Description			
Total	1		
Passed	1	Failed	0
Error	0	Incomplete	0
Skipped	0		
Start	06-05-2025 15:25:00	End	06-05-2025 15:27:37
Elapsed	2m - 36.164s		

#	ID	Description	Status
1	Test Cases/Schedule Booking		PASSED

SCREENSHOT: