Experiment 9: Design test cases and generate test scripts in Selenium

Learning Objective: Students will be able to create unit test cases.

Tools: Selenium record and playback

Theory:

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation.

Designing test cases and generating test scripts in Selenium involves several steps to ensure comprehensive testing of web applications. Here is a brief theory on how to approach this process:

- **Understanding Requirements**: Begin by thoroughly understanding the requirements of the web application under test. This includes functional requirements, non-functional requirements, and any other specifications provided.
- **Identifying Test Scenarios**: Based on the requirements, identify different test scenarios that need to be covered. Test scenarios should cover various aspects of the application such as login functionality, form submissions, navigation, data validation, etc.
- **Designing Test Cases**: Once the test scenarios are identified, design detailed test cases for each scenario. Test cases should be clear, concise, and cover all possible scenarios including positive, negative, and boundary cases.
- **Prioritizing Test Cases**: Prioritize test cases based on criticality and risk. This helps in allocating testing efforts effectively, ensuring that high-risk areas are thoroughly tested.
- **Creating Test Scripts in Selenium**: After designing test cases, create Selenium test scripts to automate the execution of these test cases. Selenium provides various APIs and libraries for different programming languages such as Java, Python, C#, etc.
- **Handling Test Data**: Ensure proper handling of test data within test scripts. This may involve utilizing data-driven testing techniques to execute the same test with multiple sets of data.
- **Handling Synchronization**: Handle synchronization issues such as waits and delays to ensure that test scripts interact with web elements at the right time.
- **Logging and Reporting**: Implement logging mechanisms within test scripts to capture relevant information during test execution. Additionally, generate comprehensive test reports to provide insights into test results.
- **Executing Test Scripts**: Execute the test scripts across different browsers and platforms to ensure cross-browser compatibility and validate the behaviour of the application under different environments.
- **Analyzing Results and Debugging**: Analyze test results to identify any failures or defects. Debug test scripts to address any issues encountered during test execution.

By following these steps, you can effectively design test cases and generate test scripts in Selenium to ensure the quality and reliability of web applications.

By integrating these practices into **LingoLeap's** (language learning platform) development, we ensure the quality, reliability, and performance of the language learning platform through effective test case design and Selenium test script generation.

Sr no	Input	Output
1	-2	Beyond the range
2	0	Beyond the range
3	1	Square of 1 is 1
4	100	Square of 100 is 10000
5	101	Beyond the range
6	4	Square of 4 is 16
7	62	Square of 62 is 3844

Test Cases

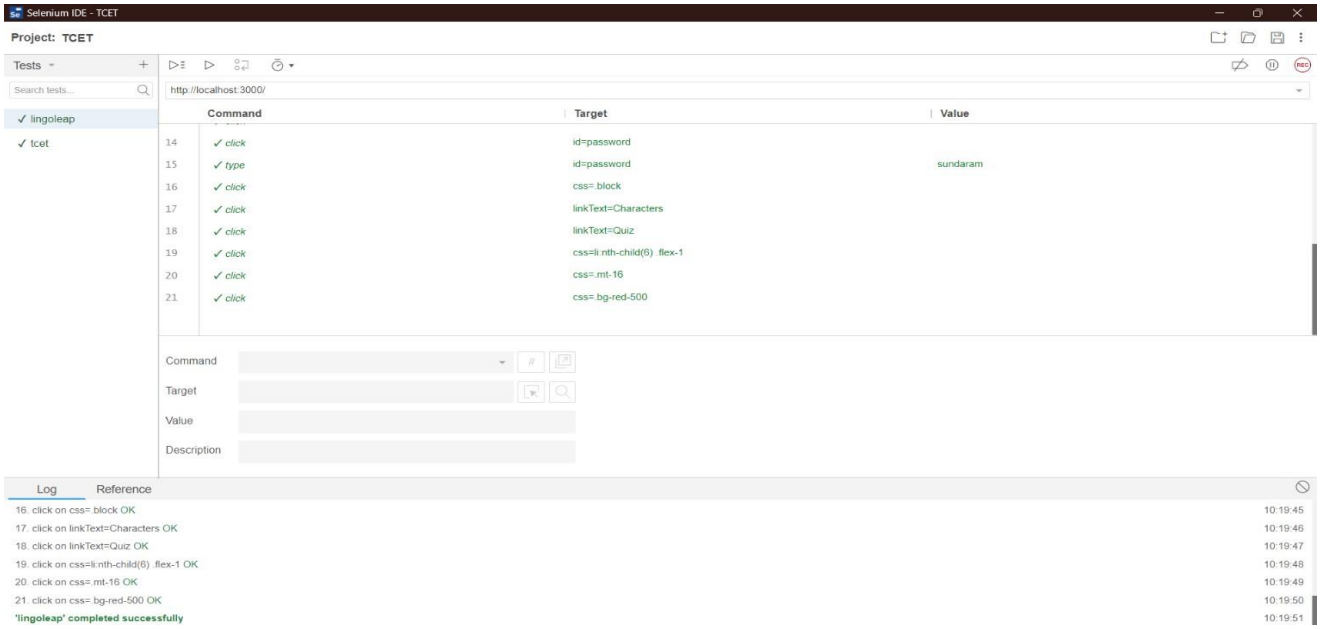
Test case 1: {I1, O1}
Test case 2 : {I2 ,O2}
Test case 3 : {I3, O3}
Test case 4 : {I4, O4}
Test case 5 : {I5, O5}
Test case 6 : {I6, O6}
Test case 7 : {I7, O7}

Black-box testing

Knowing the specified function that a product has been designed to perform, test to see if that function is fully operational and error free. Includes tests that are conducted at the software interface. Not concerned with internal logical structure of the software. It uncovers:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external data base access
- Behavior or performance errors
- Initialization and termination errors

Result and Discussion:



Learning Outcomes: Students should have the ability to

LO1: Students will be able to understand Software Testing Concepts and the various Software standards.
LO2: to test a software with the help of Junit
LO3: create test cases
LO4: To understand different tools for testing

Outcomes: Upon completion of the course students will be able to write test cases for the project.

Conclusion:

We have implemented white box testing and understood the use of white box testing in real life projects for test cases.

For Faculty Use

Correction Parameters	Formative Assessment [40%]	Timely completion of Practical [40%]	Attendance / Learning Attitude [20%]	
Marks Obtained				