AUTOMATED SOFTWARE TESTING
(502072)

Lab 1 – Test Definition
Prepared by Nguyen Thanh Quan (MSc)

1. **GOALS:** This lab helps students to
   - Understand the importance of test definition process.
   - Be familiar with defining test cases for certain features and/or the whole application.
   - Analyze test cases based on the software requirements/specifications.
   - Be familiar with writing the standard test report.

2. **OBJECTIVES**
   - Analyze the requirements/specifications of features and/or the whole application.
   - Define appropriate test strategies and test cases.
   - Understand the fundamental content of a standard test report.
   - Know how to define test pass/failed criteria.
   - Know the importance of test report and how to define a good test report.
   - Have experience about writing test report for new testers.

3. **CONTENT**

3.1 PREREQUISITES

To practise and complete this assignment, students must prepare to install text editor softwares or equivalences on their personal computers. The following are several common candidates to go:

- MS Office – Word
- Latex
- Overleaf
- Etc

3.2 INTRODUCTION TO TEST DEFINITION

Students must review theoretical lessons already taught in class and learning materials are also given in the elearning system. For instance,

- Bug/defect/error/fault/failures…
- The Automated Test Life-Cycle Methodology (ATLM)
- Test plan
- Test case
- Test report
- Etc

Students will focus on analyzing a feature of a web application, and then define test strategies, test cases, verification, validation, pass/failed definition… so that all the functions can be tested and the result is reported in the test report.

In general, a test report is an organized summary of testing objectives, activities, and results. It is created and used to help stakeholders (product manager, analysts, testing team, and developers) understand product quality and decide whether a product, feature, or a defect resolution is on track for release. Beyond product quality, a test report also provides insight into the quality of the testing and test automation activities. Organizations typically have four high-level questions about their test automation.

- What's wrong with the automation scripts?
- What's wrong with the backend?
- What's wrong with the lab?
- What's wrong with the executions?

Test reporting should help understand the achieved value of testing. For example, are the test team testing anything unnecessarily? Are defined tests stable? Was the test team able to uncover issues early in the process?

A good test reporting process gives insight and answers to all these important questions. We can not only improve quality of an app, but we can accelerate your releases.

Basic Test Report Summary

- The specification of the application to be tested.
- Goals of testing.
- Test Metrics and KPIs (Metrics and Key Performance Indicators (KPIs) quantify testing progress and effectiveness. They include pass rates, failure rates, defect density, test coverage percentage, and more).
- Test execution results (Detailed test execution results document which test cases passed and which failed. This information aids in defect identification and resolution).
- Defects and Issues (Defect reports highlight issues identified during testing, including descriptions, severity levels, and steps to reproduce. These reports guide developers in addressing defects).
- Test Coverage Analysis (Coverage reports show which parts of the software were tested and to what extent. They ensure that critical functionalities are adequately tested).
- Traceability Matrices (Traceability matrices link test cases to specific requirements or user stories. They demonstrate alignment with business objectives and regulatory compliance).
- Quality evaluation after testing.
- Advice, improvements proposed to increase quality of the software/application.
- Extra information such as time, cost, the number of issues found…

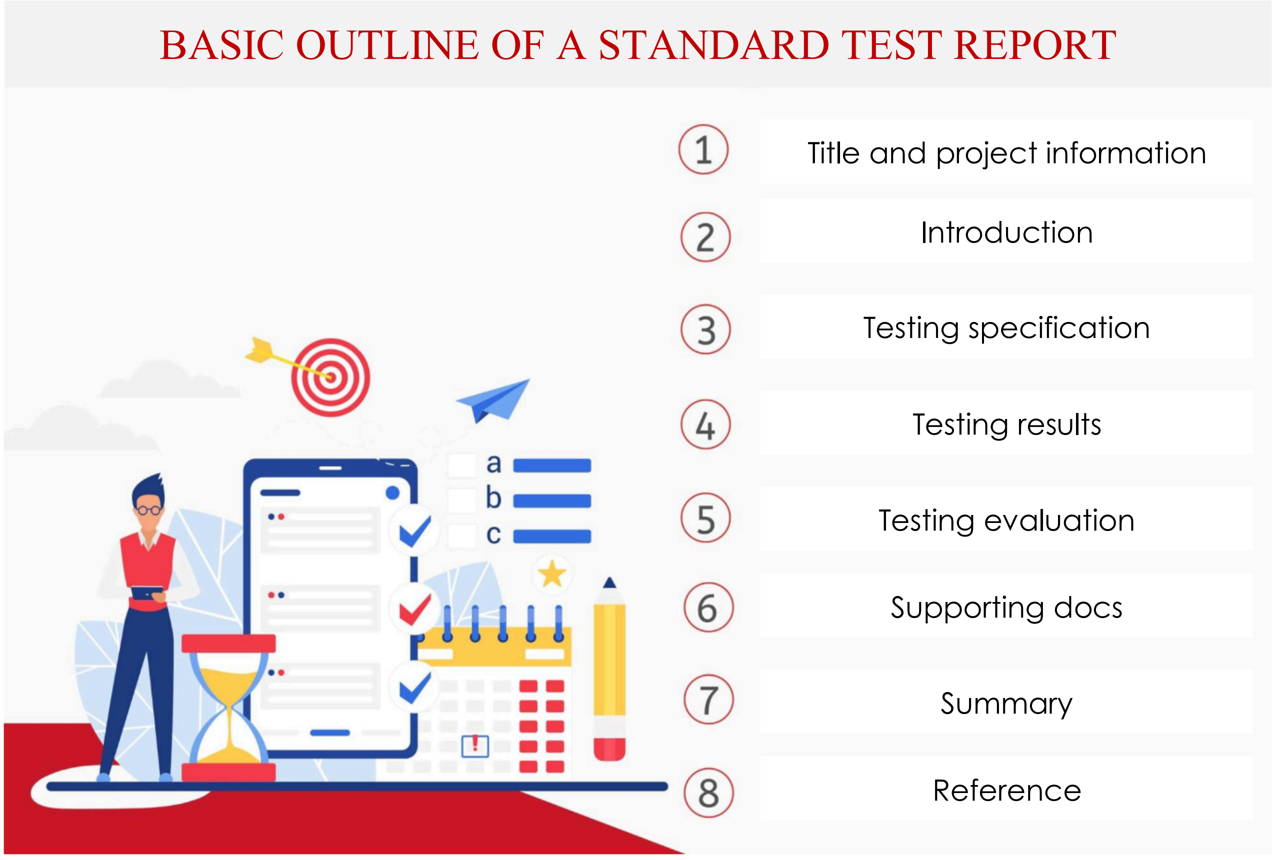
Components of Effective Test Reporting

- Real-time Insights (Real-time reporting provides immediate visibility into test execution progress, enabling quick action on critical issues).
- Clear Visualization (Clear and intuitive visualization, such as charts and graphs, makes complex data easily understandable for various stakeholders).
- Historical Trend Analysis (Comparing current results with historical data allows teams to identify patterns and make data-driven decisions for process improvement).
- Integration with Development Process (Test reporting should be integrated with the development process, such as Continuous Integration/Continuous Deployment (CI/CD) pipelines, for seamless updates).
- Collaboration and Communication (Reporting serves as a common ground for communication and collaboration among developers, testers, managers, and other stakeholders).

Writing a test report must follow the right standard because:

- Ensure the completeness and accuracy of information.
- Increase the reliability of the report.
- Ensure the understandability of information.
- Manange the testing process.
- Improve the testing process.

3.3 STANDARD TEST REPORT



BASIC OUTLINE OF A STANDARD TEST REPORT

1. Title and project information
2. Introduction
3. Testing specification
4. Testing results
5. Testing evaluation
6. Supporting docs
7. Summary
8. Reference

Figure 1 – Basic outline of a test report

- **Title and project information:** The title of the report must be given. In addition, the information of the project must be specified such as start/end date, staff, and the purpose of the report.
- **Introduction:** Introduce the content and purpose of the report.
- **Testing specification:** Scope, test cases, step-by-step of testing, results and problems found in the testing.
- **Testing results:** Statistics, testing coverage, defects found and solutions.
- **Testing evaluation:** Evaluate the testing results, propose improvements and recommendations for next releases.
- **Supporting docs:** (Optional) all related documments supporting to complete the testing report.
- **Summary:** Highlight key points in the testing results.
- **Reference:** Reference docs.

Experience sharing for freshers

- **Define Clear Reporting Objectives and Scope:** Establish specific goals/scope for reporting to ensure that the right information reaches the right stakeholders.
- **Choose Relevant Metrics and Documents:** Select metrics that align with project objectives and provide actionable insights.
- **Tailor Reports for Different Stakeholders:** Create reports tailored to different audiences, presenting information that is relevant and meaningful to each group. Remember to use technological terms in the report.
- **Ensure Consistency and Accuracy:** Maintain consistency in report formats and ensure data accuracy to build trust in the reporting process.
- **List out all defects found and solutions:** Provide detailed information about defects so that development team is able to take action as soon as possible.
- **Include Actionable Insights:** Reports should not only present data but also offer insights and recommendations for improvement.
- **Concentrate on results and evaluation:** Help the audience understand the importance of the report so as to focus on improving the product.
- **Review the report carefully before sending:** Ensure the completeness and accuracy of the report, and guarantee that the report meets the standard and expectations.

3.4 SAMPLE

[Start of the report]

Title: <Testing report of the library management application>

Project information:

- Name: <Library management software>

- Timeline: from dd/mm/yyyy to dd/mm/yyyy
- Staff: <Testing team A>
- Goals: <Evaluate testing results of the library management software>

Introduction: <Library management software is a…>

Testing specification:

- Testing scope: <Verify all functions of the software>
- Test cases (TC): <Team A defined and performed the following test cases:>
  - TC1: <Login>
  - TC2: <Borrow>
  - TC3: <Logout>
  - …
- Step by step:
  - Step 1: <Prepare data>
  - Step 2: <Perform TCs>
  - Step 3: <Collect information>
  - …
- Results and defects found: <Total>
  - Defect 1: <TBD>
  - Defect 2: <TBD>
  - …

Testing results:

- Results:
  - Total number of TCs: <TBD>
  - Total number of defects found: <TBD>
  - Testing coverage: <TBD>
- List of found defects:

ID	Name of defect	Description	Priority	Status
1	TBD	TBD	Blocker/High/ Medium/Low	Accepted
2	…	…	…	…

- Testing evaluation: <Still have a high number of defects, they need to be fixed as soon as possible prior to the upcoming release>

Improvement proposal:

- <Security>
- <Improve unittest>
- …

Supporting docs:

- <Optional>

Summary: <Briefly summarize the results of the testing>

Signature: <Team A>

Reference: <Reference docs>

[End of the report]

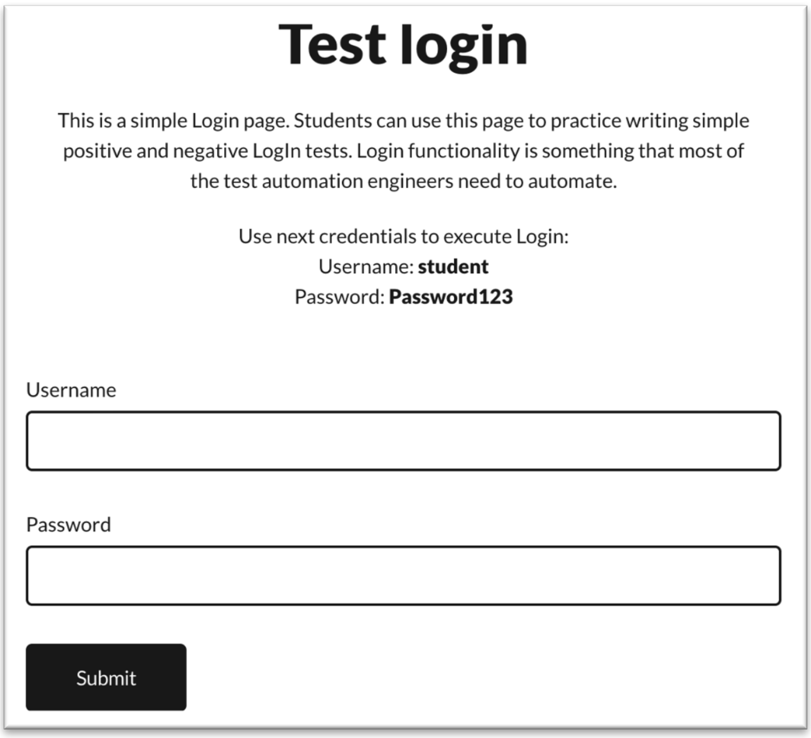
3.5 PRACTICE

Let write a test report for the login page with the format that you have learned.

This section explains test cases both negative and positive for the login screen. As we are testing a login screen we are limited to the application which is already published with limited set of information available to test. Therefore, our best approach is exploratory testing of the page.

Students should read through the standard structure of a test report provided earlier, and then write a test report for the login page with the following information:

- URL: https://practicetestautomation.com/practice-test-login/



Test login

This is a simple Login page. Students can use this page to practice writing simple positive and negative LogIn tests. Login functionality is something that most of the test automation engineers need to automate.

Use next credentials to execute Login:
Username: **student**
Password: **Password123**

Username

Password

Figure 2 – Sample of login page

Several "hints":

- Test requirements:

  As we don't have the official requirements for the login screen. We'll come up with our own requirement set for the login screen. For example, we can write the scenario based on these requirements.

  - Username should contain letter, number and period.
  - Username should not be left blank.
  - Username should not be more than 40 characters.
  - Username should not start with or contain any symbols.
  - Password should be atleast 6 characters.
  - Password should contain combination of letter, numbers and symbols.
  - Password should not contain spaces and period.
  - Password should not be more than 40 characters.

- User interface
  - Where is the cursor focus in text area when you load the page?
  - Does enter key works as a substitue for the sign in button action?
  - Does username and password text field appears in order?
  - Does the login page has register link for new users?
  - Does login screen behaves responsively to mobile or tablet screen?
  - Do the link on page remain active or are dead?

- Functionality
  - Does the login form works successfully?
  - Is logout link redirects back to the login screen? Is logout link functional?
  - Does forgot password link works? Does it redirect to valid password recovery page?
  - Do forward and backward buttons work properly while logged in? and also for logged out user?
  - Does form works similar in every popular browser?
  - How errors are handled and displayed?

- Security
  - Does textbox offers masking of characters in password field?
  - Does masked characters allow deciphering if copied?
  - Is it possible to copy and paste the password?
  - Is there any minimum password length?
  - Is the form giving away security information if the source is viewed?
  - Is the form vulnerable to SQL injection?
  - Does form allows accessing pages without logging in?
  - Is URL manipulation allows access to members only area of mail?
  - Is multiple accounts from same IP but different browser allowed at

the same time?

- Are cookies allowed? are they disabled or allowed to be edited?

- Positive test cases
  - Enter valid username and password.
  - Click on forgot password link and retrieve the password for the username.
  - Click on register link and fill out the form and register username and password.
  - Use enter button after typing correct username and password.
  - Use tab to navigate from username textbox to password textbox and then to login button.

- Negative test cases
  - Enter valid username and invalid password.
  - Enter valid password but invalid username.
  - Keep both field blank and hit enter or click login button.
  - Keep username blank and enter password.
  - Keep password blank and enter username.
  - Enter username and password wrong.

4. References:

[1] Littlewood, B. How Good Are Software Reliability Predictions? Software Reliability Achievement and Assessment. Oxford: Blackwell Scientific Publications, 1987.

[2] Daich, G., Price, G., Ragland, B., Dawood, M. "Software Test Technologies Report." STSC, Hill Air Force Base, Utah, August 1994.

[3] http://www.standishgroup.com/chaos.html.

5. Revision history

Revision	Date	Author(s)	Description
1.0	Dec 2023	Nguyen Thanh Quan (MSc)	Created