Imam Abdulrahman Bin Faisal University
College of Computer Science & Information Technology
Department of Computer Science

CS 411 – Software Engineering
Term 1 – 2024/2025

Software Quality Assurance Plan

Task tracker

Instructor Dr. Muawiya Abdul majid

Submission date:7 December 2024

TABLE OF CONTENTS

TABLE OF TABLES

1. INTRODUCTION

This Software Test Plan (STP) aims to give an approach to confirm and validate the security, performance, and functionality of the Task Tracker System. The scope, goals, and methodology of testing are outlined in this document to make sure the system satisfies the requirements and is ready for deployment.

1.1 Objectives

This paper is a key resource for identifying Task Tracker System components and features that need to be thoroughly tested. It describes the various types of testing to be carried out, allocates tasks to team members, and specifies the resources required for successful testing. It also specifies a timeline, budget, and effort estimate for finishing the testing phase. The paper also discusses potential risks related with the testing process.

1.2 Testing Strategy

The testing strategy is comprehensive, aiming to ensure the reliability, functionality, and usability of the Task Tracker System. The strategy employs a variety of testing types, including component, integration, interface, security, performance, and regression testing. Each type targets specific aspects of the system, such as user interfaces, data management, and security protocols. Furthermore, acceptance and beta testing are included to evaluate the system in real-world environments and refine it based on user feedback.

1.3 Scope

To ensure that our task tracker system works well without any errors or faults, we will test it continuously. This test plan should test the systems functionalities using component testing, integration testing, job stream testing, interface testing, security testing, performance testing, acceptance testing and beta testing. All the details will be mentioned in the next sections.

1.4 Reference Material

[1] IEEE Standard for Software Project. [Online] 1998. IEEE Std 1058-1998

[2] Somerville, Ian, “Software Engineering”, Addison Wesley (10th edition), 2018, Retrieved October 22, 2023 ISBN-10: 9332582696, ISBN-13: 978-9332582699.

1.5 Definitions and Acronyms

List of definitions:

Definition	Meaning
Software Test Plan (STP)	A document outlining the approach to confirm and validate the security, performance, and functionality of a system.
Software Requirements Specification (SRS)	A detailed description of a software system to be developed, covering functional and non-functional requirements.
Software Design Specification (SDS)	A document describing the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements.
Unit Testing	Testing individual components or modules of a system in isolation to verify they function as expected.
Integration Testing	Testing combined parts of a system to ensure they work together as intended.
Beta Testing	Testing performed by end users in a real-world environment to identify bugs or usability issues before final deployment.
Performance Testing	Testing to ensure the system performs well under specific conditions, such as heavy load or limited resources.

Table 9: List of definitions

List of acronyms:

Acronyms	Meaning
STP	Software Test Plan
SRS	Software Requirements Specification
SDS	Software Design Specification
IDE	Integrated Development Environment
JUnit	A unit testing framework for Java
JIRA	A task management tool used for bug tracking and project management
SQL	Structured Query Language
UI	User Interface
MySQL	A relational database management system

Table 10: List of acronyms

2. TEST ITEMS

In this section, we want to ensure that all parts of the Task Tracker System are tested to confirm they work correctly, handle errors effectively, and provide good user experience. The goal is to make the system reliable and user-friendly before it is finalized. We will test every component in the system such as the database, interfaces, and all associated data.

2.1 Program Modules

The modules that will be tested are user interfaces and data management.

User Interfaces: in this part we will test the navigation of the interfaces using usability and use-case testing. in terms of architecture, design, and usability.
In the usability test, the users will interact with the software through the interfaces. Then, we will determine if our design was user-friendly or not.
We will perform the use-case test on different use-case scenarios, then the test will ensure the consistency and flow of user interfaces, by validating the ease of navigation and user interactions with the interfaces. In use-case test, different use-case scenarios will be tested to validate user interaction with the interfaces and the ease of navigation.
Data Management: we will perform many MySQL queries to validate database operations.

2.2 Job Control Procedures

We will perform comprehensive testing.
Unit testing, to test the functionality of objects or methods. Component testing, to test component interfaces. System testing, to test the system as a whole by testing the component interactions.
In other words, every component in the system whether it’s a method or an object class will be tested.

2.3 User Procedures

All essential user procedures will be carried out on our system to comprehensively ensure the correctness and completeness of its user documentation.

2.4 Operator Procedures

Several operator procedures will be implemented to guarantee that the Task Tracker System works smoothly, several operator procedures will be applied such as disaster recovery testing, performance testing, and help desk testing. The Disaster Recovery Test ensures that the program can recover and restore its data in the event of failure. Furthermore, the performance test will provide an accurate evaluation of the software's nonfunctional attributes like responsiveness.

3. FEATURES TO BE TESTED

This section lists the key features of the application that will be tested to ensure they meet the requirements and function as expected. The focus is on verifying core functionality, usability, and reliability.

Feature	Design Specification
User Registration	A test to validate the registration process with proper input validation (username, password, confirm password) and ensure successful user creation in the database.
Password Strength Check	A test to ensure that the password meets strength requirements (minimum length, inclusion of uppercase, lowercase, digits, and special characters).
Duplicate Username Check	A test to ensure unique usernames are enforced and proper error messages are displayed for duplicates.
User Login	A test to validate the authentication process for registered users with correct credentials and to ensure incorrect logins are handled properly.
Guest Access	A test to ensure guest users can access task management features without requiring registration and that tasks created in this mode are not persisted in the database.
Task Creation	A test to verify that tasks can be added with valid descriptions and due dates, and the data is saved correctly for registered users.
Default Due Date	A test to confirm that tasks without specified due dates are assigned the current date as a default value.
Task Deletion	A test to verify that tasks can be deleted and are removed from both the UI and the database for registered users.
Task Completion Status	A test to ensure marking tasks as completed updates their status in the database and reflects immediately in the UI.
Task Sorting	A test to ensure tasks are sorted by their due dates in ascending order and displayed correctly in the user interface.
UI Navigation	A test to verify seamless navigation between the main menu, login, registration, and task management screens.
Database Integrity	A test to confirm that tasks are correctly linked to the logged-in user, maintaining referential integrity between users and tasks.
Input Validation	A test to validate inputs for all forms (e.g., empty fields, invalid dates) and ensure error messages are displayed for invalid data.
Error Handling	A test to ensure meaningful error messages are shown for database errors, invalid inputs, and failed actions.

Table 2: Features to be Tested with Description

4. FEATURES NOT TO BE TESTED

This section highlights features that are not included in the scope of testing. These include unimplemented features, advanced optimizations, and areas not critical to the current project objectives.

Feature Not to Be Tested	Reasoning
UI Aesthetics	Visual elements like colors, fonts, and button styles are not tested, as they do not affect functionality.
Third-Party Library Functionality	The behavior of Java Swing components (e.g., JPanel, JButton) and JDBC drivers is assumed to be correct as they are external libraries.
Database Engine (MySQL)	The core functionality of MySQL (e.g., query execution, indexing) is not tested, as it is outside the project's scope.
Cross-Platform Testing	The application is tested on windows and macOS. It may theoretically run on Linux or other platforms; it has not been explicitly tested or targeted in this project.
High-Load Stress Testing	Performance under extremely high loads (e.g., thousands of concurrent users) is beyond the current project scope.
Mobile Device Testing	The application is not designed for mobile platforms, so its functionality on mobile devices will not be tested.
Localization and Internationalization	The application is designed for English only and will not be tested for multiple languages or regions.
Future Features	Features like task sharing, notifications, or reminders are not implemented and will not be tested.
Multi-User Task Collaboration	Collaborative features where tasks are shared among users are not part of the current implementation and are excluded.

Table 3: Features not to be Tested with Reasoning

5. APPROACH

5.1 Component Testing

5.1.1 Common Functionalities

5.1.1.1 Login Interface

Test ID	Login_01
Prerequisite	The user must have valid credentials stored in the database.
Test Procedure	- Type valid username and password. - Type empty username and valid password. - Type valid username and empty password. - Type valid username and incorrect password. - Type incorrect username and valid password.
Expected Result	- Valid credentials: Successfully logs in and displays the Task Manager interface. - Invalid or incomplete credentials: Error message “Invalid username or password.” is displayed.
Actual Result	Matches expected results.
Verified (Yes/No)	Yes

Table 4: Log in Test Case

5.1.1.2 Registration Interface

Test ID	Register_01
Prerequisite	Database is accessible, and username is not already in use.
Test Procedure	- Populate username, password, and confirm password with valid fields. Input a password which doesn't fulfill the strength requirements. - Give unmatching password and confirm password. - Leave any/ all the above field empty.
Expected Result	- In case of Valid Input: able to create successful account, thus being capable to sign in - In case of error in inputs shows various appropriate messages related to cases. Some examples: "Cannot be Empty, Password did not match." among other errors.
Actual Result	Matches expected results.
Verified (Yes/No)	Yes

Table 5: Registration test case

5.1.1.3 Guest Interface

Test ID	Guest_01
Prerequisite	The application is accessible, and no login is required.
Test Procedure	- Navigate to the guest interface. - Perform actions available to guest users - Ensure no login or registration prompt appears. - Exit the application and verify that no information persists.
Expected Result	- The guest user can perform permitted actions without logging in. - Temporary data is cleared after the session ends. - No user information is stored in the database.
Actual Result	Matches expected results.
Verified (Yes/No)	Yes

Table 6: Guest test case

5.1.1.4 Task Manager Interface

Test ID	TaskManager_01
Prerequisite	The user is logged in or is a guest.
Test Procedure	- Add a task with a valid date format. - Add a task with an invalid date format. - Leave the task description empty. - Use "Sort by Date" button.
Expected Result	- Valid task input: Task is added to the list. - Invalid date: Error message "Invalid date format. Use yyyy-MM-dd." - Empty description: Error message "Task description cannot be empty." - Sort functionality: Tasks are displayed in ascending order by due date.
Actual Result	Matches expected results.
Verified (Yes/No)	Yes

Table 7: Task Manager Interface test case

5.2 Integration Testing

Description: Integration testing is meant to ensure that all parts of the task tracker system (login, registration, creation of tasks, sorting, and deletion) work together error-free and without conflicts.

Technique: Test the interaction of the login/registration system with the database for the proper saving and validation of user credentials.

Test the connection of the created task with database storage to ensure the tasks are correctly saved for registered users.

Ensure that sorting works as expected with stored tasks and displays correctly.

Verify the deletion of tasks checks out and is removed from the database properly.

Completion Criteria: everything works together without bugs integrated; data integrity and function of system maintained.

5.3 Conversion Testing

Conversion testing ensures that the tasks added by the registered users store in the database without failure in case of data loss or misshaping of data during all over processing and so on. The create tasks in guest mode must be excluded because they may not be designed to be stored in the database, whereas switching between guest and registered user functionality should be without hassle.

Technique: The task-saving functionality is tested by toggling from guest to registered mode; only the tasks created after registration should be saved in the database.

Completion Criteria: The tasks created by a registered user are saved appropriately in the database whereas the tasks created in guest mode are not saved.

5.4 Job Stream Testing

Job stream testing checks that the tasks are created, viewed, edited, deleted, or sorted appropriately and in the right order.

Technique: The web application should provide the user with the facility to create tasks, mark them as completed, sort them by date, and delete tasks using proper flow and interaction between the components.

Completion Criteria The ability to successfully perform each kind of task action-create, view, sort, delete-and any dependency between actions is handled seamlessly.

5.5 Interface Testing

Interface testing shall ensure the application interface for login, registration, and task management is user-friendly and intuitive. It also ensures error messages for invalid or missing inputs-e.g., empty username, passwords not matching-are displayed correctly.

Technique: The testing of the application should be done with valid and invalid inputs with the navigation of the different elements of the interface to test for expected behavior.

Completion Criteria: The interfaces are completely approved by the project supervisor.

Special Considerations: All error messages, such as "This username is already taken", must follow a pattern. Also, every action must include instructions.

5.6 Security Testing

Security testing will ensure that the saving of tasks in the database is done by an authorized user. The Guest user should have limited functionality to ensure security of the data. Passwords should be encrypted before saving.

Technique: Test the Login and Registration with different kinds of input to ensure that only the valid user can access the registered functionalities. Test for SQL injection or other security vulnerability.

Completion Criteria: There is no unauthorized access and data is kept secure.

5.7 Recovery Testing

Recovery testing provides assurance that in case of any sudden failure, the tasks are not lost to registered users. Tasks will not be recovered for Guest users because they are not saved.

Technique: The system will be tested based on simulated failures and check that user tasks are recovered from the database.

Completion Criteria: Tasks for registered users are recoverable post system restart or failure of system.

5.8 Performance Testing

Performance testing aims to ensure the system can support multiple users logging in and registering and executing task management actions without degradation of performance.

Technique: Simulate several users executing creates, sort, delete actions, among others to determine the response time, how well the database handles the queries, etc.

Completion Criteria: The system maintains responsiveness with different and varied loads according to predefined measures for performance.

5.9 Regression Testing

Regression testing ensures recent changes - say, adding sort functionality- does not negatively affect already existing features such as create and delete tasks.

Technique: Testing of the code after every update for the smooth working of all the functionalities, whether it be login, registration, or task management. Completion Criteria: No previous working functionality breaks after the integration of a new feature.

5.10 Acceptance Testing

Acceptance testing ensures the system will meet the functional and non-functional requirements of the task tracker.

Strategy: The system is subjected to user testing in order to analyze its functionality regarding creating, sorting, and deleting tasks. This is followed by taking feedback and making necessary improvements.

Completion Criteria: The system has met all the requirements of the user and is ready to go live.

5.11 Beta Testing

Beta testing consists of a group of users using the system in the intended environment to discover possible bugs and other usability issues.

Technique: Allow beta testers to perform actions like registering, creating tasks, and sorting them. Gather feedback and address any identified issues.

Completion Criteria: The system operates error-free, and all feedback is addressed before the final release.

6. PASS / FAIL CRITERIA

6.1 Suspension Criteria

Suspension criteria are some factors that temporarily caused the testing process to be paused or suspended, which serve as a crucial factor in helping us to determine when to suspend the testing process, some of them are:

- Lack of resources.
- Significant changes from the stakeholders.
- Loss of communication with the stakeholders.
- Functionality failure.
- Loss of essential infrastructure or tools.
- Legal or compliance problems.

6.2 Resumption Criteria

Resumption criteria are the factors that determine when a suspended process can be restarted or resumed, which means that all the suspended processes have been solved successfully, these factors helps us to decide when to start the process again some of them are:

- Availability of resources.
- Strong and continuous communication is available.
- Fixed functionality and bugs successfully.
- ensure that vulnerabilities have been identified and managed successfully.

6.3 Approval Criteria

For our project, the approval criteria for the testing process are specific conditions that need to be satisfied before decide to approve and accept the results, the problems and any existing failures during the test process must be solved and fixed successfully, to ensure that project functionalities align with the defined goals, quality standards, and stakeholders requirements, by following the approval criteria, the development team can ensure quality, and accuracy during all stages of the development process.

7. TESTING PROCESS

Here we will demonstrate test deliverables, testing tasks, responsibilities, resources and schedule.

7.1 Test Deliverables

The software test plan is going to be the deliverable from testing plan, after we finish testing steps, then the project will be ready to be delivered.

7.2 Testing Tasks

Software requirements (SRS), and design specification (SDS). After performing them, software and hardware testing must be applied.

7.3 Responsibilities

All of our team members were responsible for managing, designing, preparing, executing, witnessing, checking, and resolving test activities.

7.4 Resources

Resource category	description
Team members	Each team member was responsible for a certain area.
Hardware	Ensuring a good quality Wi-Fi connection.
software	NetBeans (Java interpreter) MySQL (for database connection)

Table 8: testing resources

7.5 Schedule

Task	Date
Software Requirements Specification (SRS)	Nov 10, 2024
Software Design Specification (SDS)	Nov 25, 2024
Develop test cases	Nov 27, 2024
Execute testing using various testing procedures	Nov 30, 2024
Troubleshooting errors	Nov 30, 2024
Modifying the system	Dec 1, 2024
Develop the Software Testing Plan (STP)	Dec 2, 2024

Table 9: testing activity schedule

8. ENVIRONMENTAL REQUIREMENTS

All the necessary characteristics of the testing environment are covered in detail in the following section.

8.1 Hardware

The following hardware is needed in the testing environment:

1-Developer Workstations: Windows 10/11 compatible PCs with a minimum of 8 GB RAM and i5 or above processors.

2-Mobile Devices: Android and iOS devices for cross-platform testing.

3-Network Equipment: Reliable internet connectivity for database synchronization.

8.2 Software

1- Operating Systems: Windows 10/11 for development.

2- Development Tools:

*DE for NetBeans Application Development.

*MySQL Workbench for database administration.

3- Testing Software: Software to test the functionalities, such as JUnit for unit testing.

8.3 Security

The following security considerations must be met by the test environment:

1- User authentication with strong password protection on developer systems.

2- Database encryption to securely store and transit the data.

3- Access to test environments is restricted to only the personnel concerned.

8.3 Tools

The following is needed: mysqworkbench: Managing the relational database.

1- JUnit: unit testing for functionalities such as login and task management.

2- Source Control: Git used to track changes in code, collaborate.

3- Task management tool: Jira is used to track testing progress and bugs.

8.4 Publications

The following documents support the testing activities:

1- Software Design Specification (SDS).

2- User manuals for MySQL Workbench and NetBeans IDE.

3- JUnit testing guidelines and MySQL database validation.

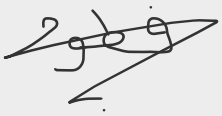
8.5 Risks and Assumptions

1. Risks:
   Delayed database integration due to team familiarity issues.
   Limited availability of team members during exams.
2. Assumptions:
   All team members will have access to the required hardware and software.
   Any changes in feature requirements will be communicated promptly.
3. Contingency Plan:
   Allocate additional time for database testing and validation.
   Schedule meetings during non-exam days for collaborative progress.

9. CHANGE MANAGEMENT PROCEDURES

Change management will be done through a well-structured process to handle modifications in the project effectively. Any team member can raise a proposal for changes in functionalities by identifying any need for improvement or adjustment. The proposed change will then be reviewed in group meetings, where the team members will discuss the feasibility, potential impact, and alignment with project goals. This review process will be overseen by the team leader to ensure that careful evaluation is made. Lastly, any change must be approved by the majority of the group members and obtain formal endorsement from the instructor to ensure that all parties are on the same page.

10. PLAN APPROVALS

No.	Student Name	Signature	Date
1 (Leader)	Ritaj Alhamli		*5/12/2024*
2	Fatimah Alageel		
3	Fatimah Alwarsh		
4	Anwar Aldahan		
6	Shahad Alshehab		
7	Maram Alnabrees		
8	Fatima Alawami