# Test Design Document

for

# <To-do it Web Application>

**Version 1.2 approved**

**Prepared by <Nguyễn Tuấn Anh>**

**<Group 12, 2324II INT2208E 23, VNU-UET>**

**14-05-2024**

# Table of Contents



# Revision History

| Name | Date | Reason for Changes | Version |
|---|---|---|---|
| Nguyễn Tuấn Anh | 14-05-2024 | Initial Document Creation | 1.0 |
| Bùi Đức Anh | 16-05-2024 | Add More Test Cases | 1.1 |
| Nguyễn Tuấn Anh | 20-05-2024 | Performance Test Cases | 1.2 |

# 1. Introduction

## 1.1. Purpose

Testing is crucial for ensuring the To-do it web application functions correctly and meets user requirements. It helps identify defects, ensures reliability, and improves the overall user experience. The goal is to verify that all features work as expected and that the system is robust and secure.

## 1.2. Intended Audience and Reading Suggestions

This document is intended for developers, project managers, users, testers, and documentation writers. It provides a detailed overview of the testing strategy and approach for the To-do it web application.

## 1.3. Product Scope

- In Scope: The scope of testing for the To-do it web application includes a comprehensive evaluation of its features and functionalities to ensure optimal performance, reliability, and security. This encompasses task management functionalities such as creation, editing, deletion, and viewing of tasks, as well as user authentication processes including registration, login, logout, and password management. Both client-side and server-side components are evaluated for responsiveness, usability, performance, and security. Integration and system testing are conducted on the complete, integrated system to ensure seamless interaction between all components. Additionally, unit testing is applied during each phase of the incremental development Scrum methodology to facilitate early detection and resolution of issues. This holistic approach ensures that the To-do it web application is robust, user-friendly, and meets all specified requirements.

- Out of Scope: The following items are not included in the scope of testing for this project:
    - Third-Party Integrations: Testing of ads, external content sharing features, and any third-party services not directly related to core functionalities of the To-do it web application.
    - Mobile-Specific Features: Any features or functionalities that are specific to mobile platforms, including but not limited to mobile app interfaces, mobile-specific performance optimizations, and mobile notifications.
    - Future Enhancements and Features: Any planned features or enhancements that have not yet been implemented in the current version of the application.

# 2. Test Strategy and Approach

## 2.1. Test Types

- Unit Tests: Individual components like task creation and user authentication.
- Integration Tests: Interactions between front-end and back-end components.
- System Tests: End-to-end functionality including task management.
- Acceptance Tests: User-centric tests to verify that features meet requirements.
- Performance Tests: Load, stress, and scalability testing.

## 2.2. Test Techniques

- Black-box testing: is a method where the internal structure or workings of the application are not known to the tester. The tester interacts with the system's interface, providing inputs and verifying outputs without considering how the software achieves these outputs. This technique is essential for validating the system's functionality against its specifications.

- Exploratory testing: is an approach where testers actively explore the application to identify defects and issues that may not be covered by structured test cases. It emphasizes the tester's creativity, intuition, and experience to uncover unexpected behavior.

### 2.3. Test Environments

- Hardware: Standard development and testing machines.
- Software: Latest versions of web browser Chrome, operating system Windows, and development frameworks.
- Network: Standard office LAN and Wi-Fi configurations.

## 3. Test Cases

This section provides the detailed explanation for each test case accompanied by inputs, outcomes, environmental and procedural requirements along with the dependencies among test cases. For each test case, there are 7 fields: Test Case ID, Test Description, Inputs, Expected Output, Pass/Fail Criteria, Priority, Dependencies.

### 3.1. Functionality Testing

| Test Case ID | Test Description | Inputs | Expected Output | Pass/Fail Criteria | Priority | Dependencies |
|---|---|---|---|---|---|---|
| TC-001 | Verify that a valid email address and password is able to log in | Valid email address, password | Successful login, main page loads | Login page redirects and shows main page | High | None |

| | | | | | | |
|---|---|---|---|---|---|---|
| TC-002 | Verify that a wrong email address(i.e not in the database) and password is not able to log in | Invalid username, password | Unsuccessful login | Stays at the login page and an invalid email/password alert pops up | High | TC-001 |
| TC-003 | Verify that a valid email address and an incorrect password is not able to log in | Valid username, invalid password | Unsuccessful login | Stays at the login page and an invalid email/password alert pops up | High | TC-001 |
| TC-004 | Verify that an incorrect email address and an incorrect password is not able to log in | Invalid username, invalid password | Unsuccessful login | Stays at the login page and an invalid email/password alert pops up | Medium | TC-001 |
| TC-005 | Verify that a valid email address and a valid password(i.e longer than 6 characters) was used for registration | Valid email address, password | Successful registration, user data is sent to the database | Sign up page redirects and shows main page | High | TC-001 |
| TC-006 | Verify that an invalid email address (i.e blank or no @)is not able to sign up | Invalid email address, password | Unsuccessful registration | Stays at the signup page | Medium | TC-005 |
| TC-007 | Verify that a valid email address and an invalid password(i.e not longer than 6 chars) | Valid email address, in valid password | Unsuccessful registration | Stays at the signup page | Medium | TC-005 |

| | | | | | | |
|---|---|---|---|---|---|---|
| TC-008 | Verify that an invalid email address and an invalid password(i.e not longer than 6 chars) | Invalid email address, in valid password | Unsuccessful registration | Stays at the signup page | Medium | TC-005 |
| TC-009 | Verify that a valid email address and an invalid password(i.e not longer than 6 chars) | Valid email address, in valid password | Unsuccessful registration | Stays at the signup page | Medium | TC-005 |
| TC-010 | Check for correct re-enteration of the password | Valid email address, valid password, invalid re-enter password | Unsuccessful registration | Stays at the signup page, “Passwords do not match” alert | Medium | TC-005 |
| TC-011 | Check for pre-existing email address in registration | Pre-existing email address(i.e already in the database), dummy password | Unsuccessful registration | Stays at the signup page, “Email is already in use” alert | Medium | TC-005 |
| TC-012 | Check for default creation of task | Task name, Due date, Time of completion, Project | Successful task creation, task is sent to the database | Task is created and in selected project, appears in the “All Tasks”, “Upcoming”, and if in the time range of the day will appear in “Today” | High | None |
| TC-013 | Task creation with past time of completion (i.e in the past) | Task name, Due date, time of completion in | Disallow the ability to set in the past | Task can only be created with valid time of completion | Medium | TC-012 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | the past, Project | | | | |
| TC-014 | Task creation with past due date(i.e in the past) | Task name, past due date, Project | Disallow the ability to set time in the past | Task can only be created with valid due date | Medium | TC-011 |
| TC-015 | Task creation with same date,time, name | Task name, due date, completion time, project * n times | Disallow the ability to create tasks with the same name and due date as well as completion time and project | Remove the ability to create multiple tasks with same name, due date and completion time and project | Low | TC-011 |
| TC-016 | Task removal | Interaction with chosen task | Successful task removal from the application and the database | Task is removed from the application | Medium | None |
| TC-017 | Default task edit | Interaction with chosen task, due date, choose time, project | Successful edit of the task in the application and in the database | Task is sucessfully edited | Medium | None |
| TC-018 | Task edit with past due date | Interaction with chosen task, past due date, choose time, project | Disallow the ability to set time in the past | Task can only be edited with valid due date | Medium | None |
| TC-019 | Task edit with past time of completion | Interaction with chosen task, due date, invalid time, project | Disallow the ability to set time in the past | Task can only be edited with valid due date | Medium | None |
| TC-020 | View today’s tasks | Interaction with “Today” | Tasks for today is | Today’s tasks appear in “Today” | Low | None |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | button in the main page | viewed successfully | | | |
| TC-021 | View upcoming tasks | Interaction with “Upcoming” button in the main page | Upcoming tasks are viewed successfully | Upcoming tasks appear in “Upcoming” | Low | None |
| TC-022 | View all tasks | Interaction with “All Tasks” button in the main page | Tasks are viewed successfully | All tasks appear in “All Tasks” | Low | None |
| TC-023 | Create project | Project name | Project appears in the “Lists” tab, is sent to the database | Project appears in the “Lists” tab | Medium | None |
| TC-024 | Create project with the same name as an pre-existing project | Pre-existing project name | Disallow the ability to set same project name as pre-exisitng project | “Project already exists” alert | Medium | None |
| TC-025 | Edit project | Project name | Edited project appears in the “Lists” tab, updated in the database | Edited project appears in the “Lists” tab | Medium | None |
| TC-026 | Edit project with the same name as an pre-existing project | Pre-existing project name | Disallow the ability to set same project name as pre-exisitng project | “Project with the same name already exists!” alert | Medium | None |
| TC-027 | Remove project | Interaction with delete button | Remove the project from the application and the database | Project no longer exist | Medium | None |

| | | | | | | |
|---|---|---|---|---|---|---|
| TC-028 | Logout | Interaction with sign out button | The user is logged out | The user is logged out | High | None |
| TC-029 | Login through URL | from login page change the append /app to the url | Users who are not signed in cannot access their account | Users who are not signed in cannot access their account | Medium | None |
| TC-030 | Past due date tasks | Login with email address and password | Tasks that are past deadline can only be alerted only once after login | Tasks that are past deadline can only be alerted only once after login | Low | None |
| TC-031 | Completable past due date tasks | A deadline task | A past due task can be marked as completed | A past due task can be marked as completed | Low | None |
| TC-032 | Real Time Task Completion | A task | Update realtime task | A task that is not past due can be updated as completed in real time | Medium | None |
| TC-033 | Sorted Tasks | List of tasks, varied due dates and time of completion | Sorted tasks by due date and time of completion | Tasks are sorted by time | Low | None |
| TC-034 | Number of Tasks | List of tasks | Updated number of tasks in projects | When adding tasks into a project, the counter(counts the number of tasks in the project) | Low | None |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | is updated | | |
| TC-035 | Task in No Project | Task | Task can be in no project | Allow the ability to create tasks without choosing a project | Medium | |
| TC-036 | No going back to login or signup after in home page | Sign Out | User in homepage will not be able to go back to login or signup page without logging out | User in homepage will not be able to go back to login or signup page without logging out | Low | None |
| TC-037 | Priority Tasks | List of tasks and some completed tasks | Completed tasks are in low priority | Tasks that are completed are sorted down the list as in low priority | Medium | None |
| TC-038 | Unlimited amount of tasks | List of tasks | Amount of task upto 100 | Tasks are unlimited to 100 tasks / user | High | None |
| TC-039 | Edit of past due tasks | Task that is past due | Past due tasks are allowed to be edited to become available again | Past due tasks are allowed to be edited to become available again | Medium | None |

### 3.2. Performance and Efficiency Testing

| Test Case ID | Test Description | Inputs | Expected Output | Pass/Fail Criteria | Priority | Dependencies |
|---|---|---|---|---|---|---|
| TC-040 | Measure the overall performance of the To-do it web application using Lighthouse | URL of the To-do it web application | Various performance scores | Average performance score above 90 is considered a pass | High | None |
| TC-041 | Measure the First Contentful Paint (FCP) of the To-do it web application using Lighthouse | URL of the To-do it web application | FCP score in seconds | FCP less than 2 seconds | High | None |
| TC-042 | Measure the Speed Index of the To-do it web application using Lighthouse | URL of the To-do it web application | Speed Index score in seconds | Speed Index less than 4 seconds | High | None |
| TC-043 | Measure the Time to Interactive (TTI) of the To-do it web application using Lighthouse | URL of the To-do it web application | TTI score in seconds | TTI less than 5 seconds | High | None |
| TC-044 | Measure the Total Blocking Time (TBT) of the To-do it web application | URL of the To-do it web application | TBT score in milliseconds | TBT less than 300 milliseconds | High | None |

| | | | | | | |
|---|---|---|---|---|---|---|
| | using Lighthouse | | | | | |
| TC-045 | Measure the Largest Contentful Paint (LCP) of the To-do it web application using Lighthouse | URL of the To-do it web application | LCP score in seconds | LCP less than 2.5 seconds | High | None |
| TC-046 | Measure the performance of the To-do it web application under load using Lighthouse | URL of the To-do it web application with simulated load conditions. | Performance score and key metrics under load. | Performance score above 80 under load conditions | Medium | None |
| TC-047 | Measure the accessibility score of the To-do it web application using Lighthouse | URL of the To-do it web application | Accessibility score | Accessibility score above 90 | Medium | None |
| TC-048 | Measure the best practices score of the To-do it web application using Lighthouse | URL of the To-do it web application | Best practices score | Best practices score above 90 | Medium | None |
| TC-049 | Measure the SEO score of the To-do it web application using Lighthouse | URL of the To-do it web application | SEO score | SEO score above 90 | Medium | None |

# 4. Test Data

## 4.1. Data Sources

To ensure a comprehensive testing process, test data for the To-do it web application will be sourced from both generated data and real-world samples. This combination will help simulate various user scenarios and interactions, ensuring the application is robust and performs well under different conditions.

### 4.1.1. Generated Test Data

- Purpose: Generated test data is used to create a wide range of test scenarios, including edge cases and high-volume situations that might not be easily obtainable from real-world data. This helps in testing the application's behavior under different conditions and ensures that it can handle various input combinations.
- Types of Generated Test Data
  - User accounts
    - Examples: “user1@example.com”, “user2@example.com”, with passwords following complexity rules.
  - Tasks
    - Tasks with varying attributes such as titles, due dates, statuses and projects.
    - Examples: Tasks with titles like “Learning”, “Playing”, or even “AAAAAAAAAAAAA…”.
  - Task interactions
    - Simulated interactions such as task creation, editing, deletion, and status updates.

### 4.1.2. Real-World Data Samples

- Purpose: Real-world data samples help ensure that the test scenarios are realistic and reflective of actual user behavior. This helps in identifying issues that might only arise under real-world conditions.
- Types of Real-World Data
  - User accounts

        - Data from actual users, anonymized to protect privacy.
        - Examples: Email addresses and login credentials following typical user patterns.
    - Tasks
        - Data from actual tasks created by users, including a variety of task types and attributes.
        - Examples: Real task titles, due dates, and completion statuses from user submissions.
- Data Collection Method: Collecting data from beta testers about their typical tasks and usage patterns.

4.2. Data Preparation

- Data Cleaning: Ensure that all generated data is validated and corrected to remove any inconsistencies or errors. This includes checking for duplicate entries, incorrect formats, and other anomalies that could affect test outcomes.
- Data Formatting: Format the data to ensure compatibility with the application's input fields and database requirements. This involves converting data into the necessary formats (e.g., date formats, string lengths, numeric ranges) that the application expects.
- Boundary Value Analysis: Create test data that includes boundary values to test the application's handling of input limits. This involves generating data at the edge of acceptable input ranges (e.g., minimum and maximum values) to ensure robust validation and error handling.

# 5. Test Execution

5.1. Test Schedule

To be finalized based on the development timeline. The detailed schedule will be determined once the development phases are more clearly defined, ensuring alignment with project milestones and delivery deadlines.

## 5.2. Test Tools

- Performance Testing: Google Lighthouse.
- Other Testing: Manual testing methods.

## 5.3. Defect Tracking

### 5.3.1. Tool Usage

Jira will be used for defect tracking. Each defect will be logged with a unique identifier for easy reference and tracking.

### 5.3.2. Defect Logging

When a defect is identified, it will be logged into Jira with detailed information including:

- Title: A brief summary of the defect.
- Description: Detailed information about the defect, including steps to reproduce, expected results, and actual results.
- Priority: The urgency for fixing the defect (High, Medium, Low).

### 5.3.3. Defect Resolution

- Developers will work on resolving assigned defects.
- Once a defect is resolved, it will be moved to the 'Ready for Retest' status in Jira.

### 5.3.4. Defect Retesting

- Testers will retest the resolved defects to ensure they are fixed.
- If the defect is not resolved, it will be reopened and sent back to the developer.
- If the defect is resolved, it will be marked as 'Closed'.

# 6. Test Reporting

## 6.1. Test Metrics

### 6.1.1. Test Case Pass/Fail Rate

| | |
|---|---|
| TC-001 | Pass |
| TC-002 | Pass |
| TC-003 | Pass |
| TC-004 | Pass |
| TC-005 | Pass |
| TC-006 | Pass |
| TC-007 | Pass |
| TC-008 | Pass |
| TC-009 | Pass |
| TC-010 | Pass |
| TC-011 | Pass |
| TC-012 | Pass |
| TC-013 | Fail |
| TC-014 | Fail |
| TC-015 | Fail |
| TC-016 | Pass |
| TC-017 | Pass |
| TC-018 | Fail |
| TC-019 | Fail |
| TC-020 | Pass |
| TC-021 | Pass |
| TC-022 | Pass |
| TC-023 | Pass |

| | |
|---|---|
| TC-024 | Pass |
| TC-025 | Pass |
| TC-026 | Pass |
| TC-027 | Pass |
| TC-028 | Pass |
| TC-029 | Pass |
| TC-030 | Pass |
| TC-031 | Pass |
| TC-032 | Pass |
| TC-033 | Pass |
| TC-034 | Pass |
| TC-035 | Fail |
| TC-036 | Fail |
| TC-037 | Fail |
| TC-038 | Pass |
| TC-039 | Pass |
| TC-040 | Fail |
| TC-041 | Pass |
| TC-042 | Pass |
| TC-043 | Pass |
| TC-044 | Pass |
| TC-045 | Fail |
| TC-046 | Fail |
| TC-047 | Pass |
| TC-048 | Pass |

| | |
|---|---|
| TC-049 | Pass |

- Total test cases
  - Functional Testing: 39
  - Performance Testing: 10
  - Total Test Cases: 49
- Test Case Results
  - Functional Testing
    - Test Cases Passed: 31
    - Test Cases Failed: 8
    - Pass Rate: 79.49%
    - Fail Rate: 20.51%
  - Performance Testing
    - Test Cases Passed: 7
    - Test Cases Failed: 3
    - Pass Rate: 70%
    - Fail Rate: 30%
- Overall Test Results
  - Total Test Cases Passed: 38
  - Total Test Cases Failed: 11
  - Overall Pass Rate: 77.55%
  - Overall Fail Rate: 22.45%

### 6.1.2. Defect Density

Languages

| language | files | code | comment | blank | total |
|---|---|---|---|---|---|
| JavaScript | 30 | 1,381 | 34 | 197 | 1,612 |
| CSS | 4 | 1,112 | 68 | 178 | 1,358 |

- Lines of Code (LOC): nearly 3000
- Total defects found: 6
- Defect density: 2 defects per KLOC

#### 6.1.3. Test Execution Time

- Total test execution time: about 14 hours
- Average execution time per test case: 0.29 hours per test case

### 6.2. Test Summary Report

The majority of the application's functionalities are working as expected, with an 79.49% pass rate for the executed test cases. The application showed good performance in standard environments, but further testing is recommended under high-load conditions. Identified defects should be prioritized and resolved to enhance the application's stability and user experience.

In conclusion, the To-do it web application has undergone comprehensive testing, and the results indicate that it meets most of the specified requirements. Addressing the identified defects will further improve the application's quality and reliability. The testing process has provided valuable insights into the application's performance, and the recommendations outlined in this report will help guide future development and testing efforts.