Table of Contents

[Purpose 2](#_Toc170065232)

[Project Overview 2](#_Toc170065233)

[2 Scope 2](#_Toc170065234)

[In-Scope 2](#_Toc170065235)

[Out-of-Scope 2](#_Toc170065236)

[3 Testing Strategy 3](#_Toc170065237)

[3.1 Test Objectives 3](#_Toc170065238)

[3.2 Test Assumptions 3](#_Toc170065239)

[3.3 Data Approach 3](#_Toc170065240)

[3.4 Testing types 3](#_Toc170065241)

[3.4.1 Functional Testing 4](#_Toc170065242)

[3.4.2 non-Functional Testing 4](#_Toc170065243)

[Performance Testing: 4](#_Toc170065244)

[Security Testing: 4](#_Toc170065245)

[Usability Testing: 5](#_Toc170065246)

[3.4.3 White-box Testing 5](#_Toc170065247)

[3.5 testing levels 6](#_Toc170065248)

[3.5.1 Unit Testing 6](#_Toc170065249)

[3.5.2 User Acceptance Testing 6](#_Toc170065250)

[Overall User Experience: 6](#_Toc170065251)

[Security and Privacy: 7](#_Toc170065252)

[3.5.3 Regression Testing 7](#_Toc170065253)

[4 Execution Strategy 8](#_Toc170065254)

[Entry Criteria 8](#_Toc170065255)

[Exit criteria 9](#_Toc170065256)

[5 Validation and Defect Management 10](#_Toc170065257)

[6 Environment Requirements 11](#_Toc170065258)

[Test Environments 11](#_Toc170065259)

[7 Dependencies 11](#_Toc170065260)

1. **Introduction**

## Purpose

To ensure the core functionalities of asana are thoroughly tested and meet the required standards.

## Project Overview

This project provides users with the ability to send, receive, organize, and collaborate with each other. This project focuses on testing Asana functionalities and features.

# 2 Scope

## In-Scope

**Functional Testing**:

1. Login process
2. Base page application main buttons
3. Creating a new project
4. Sending a message
5. Testing drag and drop feature

## Out-of-Scope

Non-functional aspects, like performance testing, security, usability and ease of use.

White box testing, like code paths and data base management etc.

# Testing Strategy

## 3.1 Test Objectives

* Verify Asana's main operations work correctly and efficiently.
* Ensure the various buttons and functionalities works correctly.

## 3.2 Test Assumptions

* Asana servers and infrastructure are operational and accessible during testing.
* Test environments accurately represent production environments.
* Users have stable internet connections for testing.

## 3.3 Data Approach

* Utilize a mix of synthetic and anonymized real data to simulate various project management scenarios.
* Test with different browsers and project names and message subjects and contents.

## 3.4 Testing types

List the types of testing to be performed.

|  |  |  |
| --- | --- | --- |
| **Test Type** | **Description** | **Responsible Parties** |
| Functional Testing | Validate core Asana operations and functionalities | QA Team |

## 3.4.1 Functional Testing

**Login process:**

* Logging in with a signed-up email and a right password
* Logging in with an unsigned email
* Logging in with a signed-up email and a wrong password

**Creating a new project:**

* Testing the blank project page

**Message pop-up:**

* Test sending a message and pressing on the link to show the message right after sending.

**Drag and drop feature:**

* testing if when adding two charts, and dragging the left chart to the right, if it actually sticks there, and the right chart goes to the left and sticks there. It means, if the drag and drop feature works well.

**Participants:**

|  |  |  |
| --- | --- | --- |
| **Tester’s Name** | **Department/ Area** | **Role** |
| Bahaa | QA | QA Tester |
| Shibel | QA | QA Tester |

## 3.5 testing levels

## 

## 3.5.1 Unit Testing

* Testing "Create" button in the base page application
* Testing some pop-up buttons after pressing "Create"
  + Project
  + Message
* Testing "view message" feature
  + Seeing the message that just been sent to check if the title appears and is identical to what the user typed.
* Testing the drag and drop feature in the Dashboard page, adding two charts, they align one near the other, testing if dragging the left chart to the right will work as intended.

**Participants:**

|  |  |  |
| --- | --- | --- |
| **Tester’s Name** | **Department/ Area** | **Role** |
| Bahaa | Development | Software Engineer |
| Shibel | Development | Software Engineer |

# 4 Execution Strategy

## Entry Criteria

* The entry criteria for initiating test execution for Gmail are as follows:

|  |  |
| --- | --- |
| **Entry Criteria** | **Notes** |
| Test environment(s) is available | Ensure that the test environment, including Asana's staging environment, is set up and accessible for testing. |
| Test data is available | Verify that the necessary test data, including sample projects, attachments, and user configurations, is prepared and accessible for testing scenarios. |
| Code has been merged successfully | Confirm that all code changes related to the Asana application have been successfully merged into the testing environment. |
| Development has completed unit testing | Ensure that all individual components of the Asana application have undergone unit testing and are ready for integration testing. |
| Test scripts are completed, reviewed and approved by the Project Team | Validate that all test cases and scripts related to Asana testing have been developed, reviewed, and approved by the project team. |

## Exit criteria

* *The exit criteria are the desirable conditions that need to be met to proceed with the implementation.*
* *Exit criteria are flexible benchmarks. If they are not met, the test team will assess the risk, identify mitigation actions and provide a recommendation.*

|  |  |
| --- | --- |
| **Exit Criteria** | **Notes** |
| 100% Test Scripts executed | All test scripts have been executed without any skipped or unresolved scripts. |
| 90% pass rate of Test Scripts | Ensure that at least 90% of the executed test scripts have passed successfully, allowing for minor exceptions. |
| No open Critical and High severity defects | Verify that all critical and high severity defects have been addressed and closed. |
| All remaining defects are either cancelled or documented as Change Requests for a future release | Ensure that any remaining defects are either resolved or documented for future consideration. |
| All expected and actual results are captured and documented with the test script | Confirm that all test results, including expected and actual outcomes, are documented alongside the test scripts. |
| All test metrics collected based on reports from daily and Weekly Status reports | Gather all relevant test metrics from daily and weekly status reports for analysis and reporting. |
| All defects logged in Defect Tracker/Spreadsheet | Ensure that all identified defects are logged and tracked in the designated defect tracking system or spreadsheet. |
| Test environment cleanup completed and a new back up of the environment | Perform a thorough cleanup of the test environment and create a new backup to ensure readiness for future testing phases. |

## 5 Validation and Defect Management

**Validation of Test Cases / Test Scenarios:**

* Test cases/test scenarios should be validated by executing them according to the defined test procedures.
* Test results should be compared against expected outcomes to ensure that the application behaves as expected.

**Defect Management:**

* Defects found during testing will be tracked using a Defect Tracker or Spreadsheet.
* Testers are responsible for opening defects, retesting to verify fixes, and closing defects once they are resolved.
* Defects will be categorized based on severity and impact as follows:

|  |  |
| --- | --- |
| **Severity** | **Impact** |
| 1 (Critical) | * Functionality is blocked and no testing can proceed * Application/program/feature is unusable in the current state |
| 2 (High) | * Functionality is not usable and there is no workaround, but testing can proceed |
| 3 (Medium) | * Functionality issues but there is workaround for achieving the desired functionality |
| 4 (Low) | * Unclear error message or cosmetic error which has minimum impact on product use. |

# Environment Requirements

## Test Environments

* **Staging Environment:** A replica of the production environment where testing will be conducted.
* **Access to Asana Test Accounts:** Testers should have access to test accounts with various configurations to simulate different user scenarios.
* **Compatible Browsers and Devices:** Ensure compatibility with major web browsers (Chrome, Firefox, Safari, Edge) and devices (desktop, tablet, mobile).
* **Internet Connectivity:** Stable internet connection to access Asana services and perform testing activities.

# Dependencies

* **Test-Item Availability:** Availability of test items such as test environments, test data, and test scripts is crucial for testing to proceed smoothly.
* **Testing-Resource Availability:** Availability of testing resources including testers, tools, and equipment is necessary to execute tests effectively.
* **Deadlines:** Adherence to project deadlines and milestones is essential for coordinating testing activities within the project timeline.