Test Plan Document

QA Automation Engineer: Nermin Maslo

1 Introduction

After analyzing the product and answering these questions (Who will use this website? What is it used for? How will it work?), a test plan was written.

This test plan document contains guidelines for testing the Contact List web application. Guidelines include:

- Business Background
- Test Objectives
- Defining test strategy
- Defining test techniques
- Backlog Grooming
- Defining the type of testing
- Environment
- Bug Classification
- Schedule
- Release Criteria

2 Business Background

A web application that allows users to register, log in, log out, create Contacts, list Contacts, update Contacts, and delete Contacts.

3 Test Objectives - Scope

The primary objective is to ensure that all acceptance criteria are met.

The objective is to test features:

- Sign Up
- Log in
- Log out
- Create Contact
- Edit Contact
- Delete Contact
- View Contacts

After testing the features separately, I will do Integration testing to verify that the features work in a group as expected.

4 Test Strategy

- **Test Levels:**
  - **Manual Testing:** Unit Testing, Integration Testing, System Testing, API Testing
  - **Automation:** Smoke Tests (UI + API)
- **Test Data:** Realistic user inputs (emails, passwords, contact details)
- **Test Execution Tools:** Manual testing + (automation with Playwright)

Testing Approach:

- Analysis of features
  - The description and acceptance criteria were read
  - Based on the description and acceptance criteria I decided to do functional testing, where I will test each module separately and finally test all modules in groups.
- Creation of backlog items
  - Backlog items are created by an analysis of the feature. Backlog items include Features, PBIs, tasks, and test cases. For every phase of testing (analysis, design, execution, etc…) individual tasks should be created.
  - Backlog items should be assigned and logically linked with each other.
- Test case creation (TCC)
  - The creation of a test case must include all necessary steps to ensure that requirements from the description and acceptance criteria are met.
- Test case development (TCD)
  - Write a test case
- Test case execution (TCE)
  - In test case execution I will compare current and expected results

Bug management

- Any bugs found should be created as work items and linked with corresponding test cases.
- Steps to reproduce should be extracted from the Test Run and updated/extended if needed in order to provide detailed information to the Development team so they can recreate it and be able to fix it in a short manner.

- In addition to the steps to reproduce a QA Engineer should provide support documentation such as screenshots, SQL queries, Logs, etc. for easier triage of the Bug.
- Define the severity and priority(by PO) of the bug.
- The QA Engineer who created the Bug work item is responsible for tracking and monitoring the Bug Life Cycle in terms of informing all relevant stakeholders if needed, and retesting the item on all relevant environments until the item is fully certified and closed.

Testing techniques

- User story
- Use case
- Boundary values
- Equivalence partitioning
- State transition

Execution

- Run test cases using compatible software (Google Chrome) for the type of test that is being run.

Pass/Fail Criteria

- Test Case Pass: The expected result matches actual behavior.
- Test Case Fail: Any unexpected error or incorrect system behavior.

5 Backlog grooming (Guidelines)

Backlog grooming is the process of creating work items depending on the type of work that needs to be performed.

When creating work items following naming convention should be applied:

Example:

TCC – Contact List Web App – Verify Register Functionality

If PBI is related to Feature naming it should follows:

When creating a Task in PBIs that have Feature as parent naming of PBI and Task should be the same.

Example:

Feature: Contact List Web App - Register functionality

PBI: TCC - Contact List Web App - Register functionality

Task: TCC – Contact List Web App - Register - Verify functionality

Guidelines

Feature (Product Owner)

When QA gets assigned to the Feature several fields need to be updated:

- Add a Tag to the Feature that will point out who will be assigned to it
  - <Test by Name>
- Testing Guidelines need to be filled out according to Acceptance Criteria and should contain Use Cases and/or Test Scenarios in the form of short descriptions of that feature.
- The QA creates several Product Backlog Items (PBI) depending on the nature of the assigned Feature and has them as a child of the Feature.
  - TCC
  - TCM

    - TCE
    - TCD
    - TCR
- Feature must have Analysis and Design PBI and Task and deliverable of this PBI and Task is features acceptance criteria.

PBI

- When creating PBI following fields need to be updated:
- Assignment to a team member
- Define iteration
- Write a description
- Define effort
- The QA creates Task(s) depending on the nature of the PBI and has them as children of the PBI
- PBI may not have linked Feature as a parent work item

Task

When creating a Task following fields need to be updated:

- Assignment to a team member
- Define iteration
- Write description(optional)
- Define estimate
- The QA creates a Test Case depending on the nature of the Task and has him as a child of Task
- Tasks may not have linked child work items

Deliverables:

Before:

- Test plan document
- Test case document
- Test Design specification

During:

- Test Scripts
- Test Data

After:

- Test Results/reports

6 Test Types Identified

Test types used in the test plan will be:

- Functional Testing
- API Testing

7 Environment

The most used environment for testing is the QA Environment. Besides this environment there could be more environment:

- Dev Environment
- Staging Environment
- Production Environment

8 Bug Classification Mechanism

Severity of Bugs:

Critical: The bug impacts the most crucial functionality of the Application and the QA team cannot continue with the validation of the application under test without fixing it.

Example: login does not work

High: A bug impacts a functional module; the QA team cannot test that module but continue with the validation of other modules.

Example: Cannot access Forgot your password module.

Medium: The bug has issues with a single screen or related to a single function, but the system is still functioning. The bug here does not block any functionality.

Low: It does not impact the functionality. It may be a cosmetic bug, UI inconsistency for a field, or a suggestion to improve the end-user experience from the UI side.

Example: UX/UI design bugs.

9 Test Automation Strategy

Automation Scope

The automation will cover API and UI testing to ensure core functionalities work correctly and efficiently. The focus areas include:

API Tests: Validate request-response structures, data integrity, and system behavior.

UI Tests: Validate front-end interactions, page navigation, and form validations.

Automation Tools & Frameworks

- Test Automation Framework: Playwright
- Programming Language: TypeScript / JavaScript
- Test Runner: Playwright Test Runner
- API Testing Tool: Playwright APIRequest

Automated Test Coverage

API Tests

- User Registration (Sign Up)

- User Authentication (Login, Logout)

- Contact Management (Create, Update, Delete Contact)

UI Tests

- Sign Up Flow (Form Validations, Error Handling)

- Login Flow

- Contact CRUD Operations

- Logout Functionality

Automation Execution & Reporting

Test Execution: Automated tests will run on:

- Local Environment (for development and debugging)

Reporting & Logs:

- Playwright HTML Reports

Pass/Fail Criteria

- API Tests Pass when responses return expected status codes and data matches the expected schema.
- UI Tests Pass when elements render correctly, actions trigger the expected results, and there are no UI crashes.

9 Release Criteria

Release criteria for this test plan would be:

- All critical and high-severity bugs must be resolved.
- 95% of other bugs found must be resolved.
- Only 5% of minor bugs can persist.