fsd-demo-Test-Plan

December 30, 2024

0.1 Test Plan for full-stack demo - Flask and SQLite application(fsd-demo):

---

1 Test Plan

1.1 1. Introduction

1.1.1 1.1 Purpose

The purpose of this test plan is to outline the testing strategy and approach for the Flask and SQLite application. The goal is to ensure that the application meets its requirements and functions correctly.

1.1.2 1.2 Scope

This test plan covers the testing of all functionalities of the application, including creating, reading, updating, and deleting user information.

1.1.3 1.3 Objectives

- Verify that the application meets the specified requirements.
- Ensure that all functionalities work as expected.
- Identify and fix any defects in the application.

1.2 2. Test Items

The items to be tested include: - User creation - User retrieval - User update - User deletion

1.3 3. Features to be Tested

- **Create User**: Adding a new user to the database.
- **Read Users**: Displaying a list of all users.
- **Update User**: Modifying the details of an existing user.
- **Delete User**: Removing a user from the database.

1.4 4. Features Not to be Tested

- Performance under high load (as this is a simple application).
- Security features beyond basic input validation.

1.5 5. Test Approach

1.5.1 5.1 Types of Testing

- **Unit Testing**: Testing individual functions in `models.py`.
- **Integration Testing**: Testing the interaction between the Flask application and the SQLite database.
- **Functional Testing**: Testing the application against the functional requirements.
- **User Interface Testing**: Ensuring the UI is intuitive and functions correctly.

1.5.2 5.2 Test Environment

- **Hardware**: Standard server hardware.
- **Software**: Python, Flask, SQLite, web browser.

1.6 6. Test Cases

1.6.1 6.1 Create User

- **Test Case ID**: TC001
- **Description**: Verify that a new user can be added to the database.
- **Preconditions**: The application is running.
- **Steps**:
  1. Navigate to the home page.
  2. Fill out the form with user details (name, email, age).
  3. Submit the form.
- **Expected Result**: The new user is added to the database and displayed on the home page.

1.6.2 6.2 Read Users

- **Test Case ID**: TC002
- **Description**: Verify that the list of users is displayed correctly.
- **Preconditions**: The application is running.
- **Steps**:
  1. Navigate to the home page.
- **Expected Result**: The list of users is displayed with correct details.

1.6.3 6.3 Update User

- **Test Case ID**: TC003
- **Description**: Verify that an existing user's details can be updated.
- **Preconditions**: The application is running, and at least one user exists.
- **Steps**:
  1. Navigate to the home page.
  2. Fill out the update form with new user details.
  3. Submit the form.
- **Expected Result**: The user's details are updated in the database and displayed on the home page.

1.6.4 6.4 Delete User

- **Test Case ID**: TC004
- **Description**: Verify that a user can be deleted from the database.
- **Preconditions**: The application is running, and at least one user exists.
- **Steps**:
  1. Navigate to the home page.
  2. Click the delete button next to a user.
- **Expected Result**: The user is deleted from the database and no longer displayed on the home page.

1.7 7. Test Schedule

- **Unit Testing**: Week 1
- **Integration Testing**: Week 2
- **Functional Testing**: Week 3
- **User Interface Testing**: Week 4

1.8 8. Test Deliverables

- Test cases
- Test scripts
- Test results
- Defect reports

1.9 9. Resources

- **Testers**: 2
- **Developers**: 1
- **Tools**: Python, Flask, SQLite, web browser

1.10 10. Risks and Contingencies

- **Risk**: Limited testing resources.
- **Contingency**: Prioritize critical test cases and automate where possible.

1.11 11. Approval

- **Prepared by**: [Your Name]
- **Approved by**: [Approver's Name]

This test plan outlines the strategy and approach for testing your application.

[ ]: