IN PARTNERSHIP WITH PLYMOUTH UNIVERSITY

Name: Ilangasekara Ilangasekara (Group Leader)
Student Reference Number: Group 42

Module Code: PUSL2020	Module Name: Software Development Tools and Practices
Coursework Title: C1	
Deadline Date:11/05/2023	Member of staff responsible for coursework: Dr. Rasika Ranaweera | Ms. Pavithra Subhashini
Programme: Bsc (Hons) Software Engineering / Bsc (Hons) Computer Science	

Please note that University Academic Regulations are available under Rules and Regulations on the University website www.plymouth.ac.uk/studenthandbook.

Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.

Ilangasekara Ilangasekara – 10820284
Cassimdeen Mohamed – 10820255
Mohamad Ahamad – 10819531
Nekath Jayathilaka – 10820283
Nuwani Kariyawasam – 10818833
Rashani Arachchi - 10819471

We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.

Signed on behalf of the group: 

Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.

I *have used/not used translation software.

If used, please state name of software……

Overall mark _____% Assessors Initials _____ Date_____

*Please delete as appropriate Sci/ps/d:/students/cwkfrontcover/2013/14

UNIVERSITY OF PLYMOUTH

PUSL2020 Software Development Tools

GROUP 42

Name	Index Numbers	Degree Program
Ilangasekara Ilangasekara	10820284	Computer Science
Nekath Jayathilaka	10820283	Computer Science
Mohamad Ahamad	10819531	Software Engineering
Nuwani Kariyawasam	10818833	Software Engineering
Cassimdeen Mohamed	10820255	Computer Science
Rashani Arachchi	10819471	GTF Member's Testing

Contents

Testing Strategies for the Garbage Collection Web Application

Introduction:

The purpose of this report is to outline the testing strategies for the Garbage Collection web application. The web application comprises three main parts: the admin, Green Captain, and GTF members. Each user role has specific functionalities, and it is essential to ensure that the application functions correctly, meets user requirements, and provides a seamless experience. This report will describe the test cases, mock objects, unit and integration testing procedures, functional test plans, and a critical analysis of the test strategy.

1. Test Cases:

Test cases are designed to validate the various functionalities of the Garbage Collection web application. Let's take a look at some examples of test cases for different features:

1.1 User Registration:

- Test Case 1: Verify that a new user can successfully register with valid information.
- Test Case 2: Verify that an error message is displayed if the user tries to register without providing a username.
- Test Case 3: Verify that an error message is displayed if the user tries to register with an existing email address.

Test Steps:

1. Navigate to the registration page.

2. Enter valid registration details, including name, email, and password.

3. Click on the "Sign Up" button.

4. Verify that the user is redirected to the login page and a success message is displayed.

5. Attempt to log in using the registered credentials.

6. Verify that the login is successful.

1.2 User Login:

- Test Case 1: Verify that a registered user can log in successfully with the correct username and password.
- Test Case 2: Verify that an error message is displayed if an incorrect password is entered during login.
- Test Case 3: Verify that an error message is displayed if an unregistered username is entered during login.

Test Steps:

1. Navigate to the login page.

2. Enter valid login credentials.

3. Click on the "Login" button.

4. Verify that the user is redirected to the appropriate dashboard or landing page.

5. Repeat the test with various combinations of incorrect usernames or passwords.

6. Verify that appropriate error messages are displayed for invalid login attempts.

1.3 Incident Reporting:

- Test Case 1: Verify that a GTF member can report an incident by providing all the required information, including images.
- Test Case 2: Verify that an error message is displayed if the GTF member tries to report an incident without providing a location.
- Test Case 3: Verify that a GTF member can update the details of their own incidents successfully.

Test Steps:

1. Log in as a GTF member.

2. Navigate to the incident reporting page.

3. Enter incident details, including location, description, and upload relevant images.

4. Click on the "Submit" button.

5. Verify that the incident is successfully recorded in the database.

6. Try updating or deleting the incident details.

7. Verify that the changes are reflected correctly in the system.

1.4 Incident Approval:

- Verify that Green Captains can view reported incidents as a list and on a map.
- Verify that Green Captains can select and view individual incident details.
- Verify that Green Captains can approve or reject incidents and provide appropriate feedback.
- Verify that Green Captains can set importance flags for incidents accurately.

1.5 Admin Functionality:

- Verify that the admin can create accounts for Green Captains and staff successfully.
- Verify that the admin can post articles for public awareness about garbage collection.
- Verify that the admin can add garbage spots accurately.

2. Mock Objects:

2.1 Database Mock:

A mock database object can be used to simulate database interactions during unit testing. It allows for controlled data retrieval and modification to test various scenarios without affecting the actual production database.

3. Unit and Integration Testing:

Unit testing involves testing individual components or units of code in isolation to verify their functionality. Integration testing focuses on testing the interactions between different components to ensure they work together as expected. The following steps can be followed to run unit and integration tests:

3.1 Unit Testing:

- Utilize a unit testing framework such as NUnit or MSTest to write unit tests for each component or class in the application.
- Mock any dependencies or external services using mock objects.
- Write test cases that cover different scenarios and edge cases.
- Run the unit tests using the testing framework, analyze the results, and fix any identified issues.

3.2 Integration Testing:

- Identify key integration points in the application, such as the interaction between the user interface, database, and external APIs.
- Design integration tests that cover these interaction points and verify the correct flow of data and functionality.
- Run the integration tests and analyze the results, ensuring that all components integrate seamlessly.

4. Functional Test Plans:

Functional testing focuses on testing the application's features and ensuring they meet the specified requirements. The following functional test plans can be implemented:

4.1 User Registration:

- Verify that users can register successfully and their information is stored correctly in the database.
- Verify that appropriate error messages are displayed for missing or invalid information during registration.
- Test the registration process with different scenarios, such as registering with an existing email or username, to ensure proper validation.
- Verify that upon successful registration, users can log in with their credentials.

4.2 User Login:

- Verify that registered users can log in successfully with their correct credentials.
- Test the login process with various scenarios, such as entering incorrect passwords or non-existent usernames, to ensure appropriate error handling.
- Verify that users are redirected to the correct pages upon successful login.

4.3 Incident Reporting:

- Test the incident reporting functionality by submitting incidents with different combinations of data, including images.
- Verify that incidents are accurately recorded in the database and associated with the correct location.
- Test updating and deleting incidents to ensure that changes are reflected correctly.
- Verify that the appropriate validations are in place to handle missing or invalid data.

4.4 Incident Approval:

- Verify that Green Captains can view the reported incidents as a list and on a map.
- Test the selection of individual incidents to ensure that all details are displayed correctly.
- Verify that Green Captains can approve or reject incidents and provide feedback as necessary.
- Test the flagging of incidents based on importance and verify that the correct flags are set.

Steps:

1. Log in as a Green Captain.
2. Navigate to the incident approval page.
3. Verify that the reported incidents are displayed correctly.
4. Select an incident to view the details.
5. Approve the incident and verify that the status is updated in the database.
6. Reject the incident and verify that the status is updated accordingly.

4.5 Admin Functionality:

- Verify that the admin can create accounts for Green Captains and staff successfully.
- Test the creation of accounts with different combinations of data to ensure accurate storage in the database.
- Verify that the admin can post articles for public awareness about garbage collection and that the articles are displayed correctly.
- Test the addition of garbage spots and verify that they are accurately marked on the map.

Steps:

1. Log in as an admin.
2. Navigate to the account creation page.
3. Enter the required details for a new Green Captain or staff member.
4. Click on the "Create Account" button.
5. Verify that the account is created successfully and the new user can log in with the provided credentials.

6. Navigate to the article posting page.
7. Enter the article details, including title, content, and relevant information.
8. Click on the "Post Article" button.
9. Verify that the article is posted successfully and displayed on the public awareness section.

5. Critical Analysis of Test Strategy:

The chosen test strategy focuses on validating the core functionalities of the Garbage Collection web application. Unit testing ensures that individual components work correctly, while integration testing ensures the seamless interaction between components. Functional testing ensures that the application meets user requirements.

The test strategy prioritizes the key user roles: GTF members, Green Captains, and the admin. It covers user registration, login, incident reporting, approval processes, and admin functionalities. By designing test cases that cover various scenarios and edge cases, the test strategy aims to ensure robustness and reliability of the application.

Mock objects are utilized to isolate dependencies and external services during testing, allowing for controlled and focused validation of components. This approach enhances testability and minimizes dependencies on external systems.

To run the tests, a unit testing framework such as NUnit or MSTest can be utilized. Integration tests can be executed by simulating interactions between components and analyzing the results.

The test strategy, while comprehensive, may benefit from additional considerations such as performance testing, security testing, and usability testing. These aspects could be addressed in future iterations to ensure a well-rounded testing approach.

Overall, the chosen test strategy aligns with the functional requirements and aims to provide a robust and reliable Garbage Collection web application.

6. Conclusion:

This comprehensive report has outlined the testing strategies for the Garbage Collection web application, covering test cases, mock objects, unit and integration testing procedures, functional test plans, and a critical analysis of the test strategy. By following these strategies, the application can be thoroughly tested, ensuring that it meets user requirements, functions correctly, and provides a seamless experience for all users.

Individual Contribution and workload Matrix

Name	Index Numbers	Contribution
Ilangasekara Ilangasekara	10820284	Creating Admin's Part
Nekath Jayathilaka	10820283	Creating Green Captain's Part
Mohamad Ahamad	10819531	Admin's Part Testing
Nuwani Kariyawasam	10818833	Creating GTF Member's Part
Cassimdeen Mohamed	10820255	Green captain's Part Testing
Rashani Arachchi	10819471	GTF Member's Testing