Roq

CredersiVend Test Plan

Team RoVR

Credersi

24/03/23

Version

2.0

1. Introduction

1.1. Project Background

Credersi-Vend is a retail organisation that sells vending machines to a number of clients across multiple sectors.

Credersi-Vend have developed a new backend and User Interface for their vending machine maintenance web application.

As part of the project, RoVR is responsible for developing and executing a comprehensive test plan to improve confidence in their web application. RoVR will test the application to meet the functional requirements of Credersi-Vend.

The objective of this test plan is to identify and mitigate any defects within the web application. The testing will contain elements of component testing, integration testing, system testing and user acceptance testing to provide a high level of test coverage across the system.

1.2. Purpose

The purpose of this document is to describe the scope and high-level approach for the work to be undertaken, along with supporting material on factors that will affect the testing.

1.3. Test Objectives

The primary objective of our testing is to ensure that all features present in the Credersi-Vend web application meet the requirements of all stakeholders. As Credersi-Vend uses a database for storing customer information, ensuring this information is stored correctly and is accessible to users is of critical importance.

We will also be testing the user interface of the web application in order to ensure a suitable level of usability is achieved.

Our secondary objective is to critically analyse all defects and ensure a suitable logging of each defect against a graded chart to highlight the importance and potential impact on the functionality of the web application. This will ensure that all uncovered defects are dealt with appropriately.

2. Scope

2.1. Test Scope – Inclusions

2.1.1. Systems Under Test

The Credersi-Vend system is composed of two different programmes that each have their own various elements in need of testing:

- The console application is used to manage customer details
- The web application is used to provide details on the vending machines and their location within the customer site

Item	Purpose
Credersi-Vend Customers	Create and manage customer details in a local database. Displays the backend console application used to add and remove customers from the Database while also providing information relating to customers in the system.
Credersi-Vend Admin	The system responsible for mapping the maintenance routes displayed in the web application. The system also contains the Svelte frontend programme which users interact with to see the maintenance routes.

2.1.2. Features Under Test

Feature	Description
Customer Registration in Credersi-Vend Customers	Creating new customers and inputting their details Automated User Category Assignment based upon the number of machines they have
Vending Machine Database	Displaying all the data relating to each machine and it's position within the customer sites Displaying the information around which route a user would take to move from machine to machine
Create Vending Machine in Credersi-Vend Admin	The ability to create route information for each machine The location information for a particular machine Customer details for each vending
User Interface for Credersi-Vend	The frontend web application used by a User to create company details and new vending machines for Credersi-Vend

2.2. Test Scope – Exclusions

Within the 'VendDatabase' there is a class called 'CustomersContext' and the developers have explicitly stated that this is outside the scope of testing. The class sets up the connection to a database and performs operations such as saving customer information located in the specified local data folder.

The initial login field used to enter the web application is outside the scope of our testing. We have been given the access login and password for the explicit purpose of testing the application. They can't be modified or updated and will remain a constant throughout all our tests.

3. Approach

We will be adopting an agile testing plan taking place over a 2 week sprint. Our aim is to test the multiple functionalities of the web application in stages to ensure maximum test coverage.

We will use a number of test tools to help with the various different aspects of testing the application:

- Postman will be used for testing the API connections between the User Interface and the database
- Cypress/Selenium will be used to automate the testing of the User Interface
- Github will be used to collaboratively work on the project and facilitate elements of defect management
- Trello will be used for project management and resource allocation

We will be using a combination of manual and automation test strategies to test different parts of the application. We will use Selenium/Cypress alongside a test framework such as Jest or Cucumber to automate a number of tests as this will also provide us with test scripts we can use to run future regression testing.

Manual testing will be employed to test aspects of the CredersiVend Customers Console Application as well as query elements in the Neo4j vending machine database.

We will create User Stories to outline the product requirements and detail each element of the application to test.

4. Acceptance Criteria

4.1. Entry Criteria

The web application programme has been developed and all necessary documents are up to date and have been passed over to the test team.

4.2. Exit Criteria

All test cases have been executed with accurate recording of any defects. All defects have been graded and prioritised before a detailed report is created and passed back to the development team.

5. Tasks and Deliverables

5.1. Test Milestones

Task	Milestone	Planning Date
Test Planning	Create the Test Plan documentation	20/3/2023
Test Case Design	Complete the design of the test cases both manual and automation.	23/3/2023-24/3/2023
Setting up the Test Environment	Initialise the Test Environment	20/3/2023
Design of the user stories	Creating multiple user stories for the application	21/3/2023
Creating test cases for each defined user story	Create at least 3 test cases for each user story	22/3/2023
Create test data for the test cases	Create test data entries for the test cases	22/3/2023
Component testing of the application	Test the different components of the application.	23/03/2023-27/03/2023
Integration testing using Postman API	Create a series of automated tests that expect to pass ensuring the environment is usable.	23/3/2023-28/3/2023
Acceptance testing using Cypress.	Create acceptance criterias based on the product requirements	24/3/2023-26/3/2023
Defect report write up	Write up a comprehensive list of any uncovered defects	28/03/2023
Presentation prep	Planning our presentation	29/03/2023-30/03/2023

5.2. Test Deliverables

Deliverable	Description
Test Case Design	The test case design will outline all our test cases along with unique identifiers that will allow us to track our progress during each phase of testing
Defect Reporting	Trello will be used for keeping a log of any defects found during testing
Regression Testing	We will use regression testing to ensure the programme is still functional after modifying information in the database
Test Levels	Implementing tests at the various test levels to ensure maximum test coverage
Product Requirements	Create User Stories to influence our test strategy
Traceability	Recording results from all executed tests and accurately logging any defects

6. Roles and Responsibilities

Name	Responsibilities
Nathan	Automated System and Acceptance Testing using Cypress Designing Test cases & Manual Test Execution
Tom	Automated Test design for component testing Designing Test cases & Manual Test Execution
Tommy	System Integration Testing between Postman and Neo4j Designing Test cases & Manual Test Execution
Will	API Testing using Postman Load Testing using Postman Designing Test cases & Manual Test Execution

7. Test Environment Needs

The test environment will be used by the testing team to develop and execute tests on the software. The environment will consist of development workstations running the required software tools, such as IDEs and compilers. The environment will also include version control systems and code repositories.

The support and controls needed for the development environment include:

- Access control and security management
- Regular backups of the code repositories

- The environment will consist of multiple servers and systems, including database servers, application servers, and web servers.

The Acceptance Testing environment will be used to perform user acceptance testing of the software by end-users. The environment will consist of multiple systems, including database servers, application servers, and web servers.

In addition to the computer system requirements, the test environment will also require rooms and other facilities for testing and training purposes. These facilities will include meeting rooms, training rooms, and computer labs. Some peripheral devices, such as printers and scanners, will be shared among the different environments. The allocation and sharing of these devices will be managed through a centralised system. External interfaces to other systems or outside the organisation will be supported through the use of APIs and other integration technologies.

8. Staffing and Training Needs

Our team will need additional training in Cypress and Cypher in order to allow them to better design test cases. This will also allow us to more efficiently write and execute the test phases at each stage of testing.

9. Test and Defect Management

9.1. Test Management

Trello will be used as a platform for managing the different phases of our test plan. Each tester will be able to upload any blockers or discovered defects and coordinate with other members of the test team. This will allow us to resolve any potential issues in an efficient and timely manner.

9.2. Defect Management

Defects will be managed in Trello with each defect logged and graded with a priority number (1-4). The defects will be logged along with any supporting evidence such as test suite results or screenshots of manual test executions. This will provide us with key information when we come to develop our test report.

10. Assumptions

Description	Impact
Access to all required dependencies	This will allow us to accurately create our test environments

11. Constraints

Description	Impact
Working from home may result in poor quality internet	This could cause connectivity issues when working collaboratively using Teams

12. Risks

Description	Impact	Likelihood	Risk Factor	Owner
Dependencies may be unavailable	We cannot run test environment	3		

12.1. Document Sign-off

This document has been reviewed, and approved for issue at the indicated issue status by the following:

CredersiAdmin Project Manager or Authorised Representative

Name:

Position:

Signature:

Date:

Roq Test Project Manager or Authorised Representative

Name:

Position:

Signature:

Date: