

|  |  |
| --- | --- |
| ROQ Main Contact Details: | <<Client>> Main Contact Details |
| * <<Name>> | * <<Name>> |
| * <<Title>> | * <<Title>> |
| * <<Telephone No>> | * <<Telephone No>> |
| * <<Email Address>> | * <<Email Address>> |

<<Client>>

<<Publishing Date>>

<<Document title>>

Version <<Version No>>

Contents

[1. Introduction 3](#_Toc256774621)

[1.1. Project Background 3](#_Toc256774622)

[1.2. Purpose 3](#_Toc256774623)

[1.3. Test Objectives 3](#_Toc256774624)

[2. Scope 3](#_Toc256774625)

[2.1. Test Scope – Inclusions 3](#_Toc256774626)

[2.1.1. Systems Under Test 3](#_Toc256774627)

[2.1.2. Features Under Test 3](#_Toc256774628)

[2.2. Test Scope – Exclusions 3](#_Toc256774629)

[3. Approach 3](#_Toc256774630)

[4. Acceptance Criteria 3](#_Toc256774631)

[4.1. Entry Criteria 3](#_Toc256774632)

[4.2. Exit Criteria 3](#_Toc256774633)

[4.3. Suspension Criteria 3](#_Toc256774634)

[5. Tasks and Deliverables 3](#_Toc256774635)

[5.1. Test Project Plan 3](#_Toc256774636)

[5.2. Test Milestones 3](#_Toc256774637)

[5.3. Test Deliverables 3](#_Toc256774638)

[6. Roles and Responsibilities 3](#_Toc256774639)

[7. Test Environment Needs 3](#_Toc256774640)

[8. Test Data Needs 3](#_Toc256774641)

[9. Staffing and Training Needs 3](#_Toc256774642)

[10. Test and Defect Management 3](#_Toc256774643)

[10.1. Test Management 3](#_Toc256774644)

[10.2. Defect Management 3](#_Toc256774645)

[11. Assumptions 3](#_Toc256774646)

[12. Constraints 3](#_Toc256774647)

[13. Risks 3](#_Toc256774648)

[14. Templates 3](#_Toc256774649)

[15. Document Control 3](#_Toc256774650)

[15.1. Document Review 3](#_Toc256774651)

[15.2. Change History 3](#_Toc256774652)

[15.3. Distribution 3](#_Toc256774653)

[15.4. Referenced Documents 3](#_Toc256774654)

[15.5. Document Sign-off 3](#_Toc256774655)

1. Introduction
   1. Project Background

This project is concerned on the establishment of a vending machine business with 3 labors, located in (where ever it is) and this business starting with (how many machines?) with total capital of £300,000. A vending machine is a machine for business, which dispenses certain products to the customers after they have inserted currency inside the machine. In today’s market, automated machines are in high demand for they make numerous activities not only easier but also more efficient. These machines often require limited human intervention to do its job, making it more convenient. These vending machines typically have sanitary supplies to beverages and has applied a solid sales strategy in the business world.

Starting its operation using conventional mechanical controls, Credersi-vend have employed electronic components to add more functional features to their vending machines which in turn has increased the accuracy of their system operation. Due to this added functionality, the demand for their vending machines have also increased, resulting in the need for a new piece of software to follow where the vending machines are located allowing for engineers to be able to locate a set route plan for their jobs.

This Software is otherwise known as Credersi-Vend Admin. This simple piece of software allows for an engineer to be able to log in and follow a pathway to each vending machine in a set area, thus allowing the engineer to be able to get to each job more efficiently.

Before this software has been published, Credersi-vend have located a third-party client to run different types of tests on the backend, frontend and the routes part of their software. This is to ensure that the software is at a satisfactory level without any major defects that could result in delaying the launch.

* 1. Purpose

This documentation has been created to describe the scope and high-level approach that will be undertaken, along with supporting material on factors that could potentially affect the testing stages. This document will also outline any known defects that had been found during the testing stages, and any potential concerns that could arise in the future depending on the outcome of the tests.

* Alpha release
  1. Test Objectives
* To produce user stories that will give clear non-ambiguous outlines for creating tests
* Produce documentation of tests that is easily understood
* Produce scripts and code that allows of testing of the Credersi-vend webapp
* Using a vast coverage of different testing types to produce documentation
* Produce defect reports through GitHub push system

1. Scope
   1. Test Scope – Inclusions
      1. Systems Under Test

Define the system under test. Summarise its constituent testable systems, sub-systems or component parts. Include versions where possible.

| Item | Purpose | Version |
| --- | --- | --- |
| CredersiVendAdmin | An admin tool that is used to locate vending machines in a specific area and the pathway throughout the area to each machine. | Alpha |

* + 1. Features Under Test

Summarise all the major features and combinations of features to be tested, include non-functional aspects and documentation references, e.g. Business Requirements. Include versions where possible.

| Feature | Sub-Feature | Reference Document and Version |
| --- | --- | --- |
| Log-in | Allow for login or Incorrect credential throws denial. |  |
| Add machine | Name machine and add to breadcrumb trail  Creates unique identifiers for these machines to allow for querying |  |
| Add Customer | Add customer name and sites. Create machines and locations at customer sites.  Creates unique identifiers for these customers to allow for querying |  |
| Retrieve machine data and location | Provide information for specific machine with unique identifiers and fields of data |  |
| Create path for technician to follow | Addition to a bread crumb trail from each machine and location. |  |
| Create new origin machine (Bread crumb trail) | Starting of new pathway from new base in which more machines can be created and pathway extended |  |

* 1. Test Scope – Exclusions

Identify all items and/or software features excluded from the scope of testing, explain why. Note that this information is often more important than specifying what is in scope.

(FILL IN)

1. Approach

If the plan covers multiple types or phases of testing then this section can be split into an Overall Approach and a separate specific approach for each of the testing types. Where the different types of testing will be handled in a very different way it is better to have multiple test plan documents instead.

Specify the major activities, techniques and tools that are to be used to test the items and features in scope.

The approach should be sufficiently detailed to enable identification of the major tasks and estimation of time to do each one. When developing the approach consider:

The test policy/strategy (if applicable)

Significant constraints on testing, e.g. test resource and/or environment availability, time, etc and the impact on the approach

Risk based testing

How much testing is required: too much is a waste of time and money, as is too little and the wrong testing is the worse for little or no gain

The type of industry

Contractual, legal, regulatory or specific customer requirements

Team experience

Resource availability

Documentation availability

Etc.

Define the test specification document requirements, how many specifications will be required, map these to the items/features identified as being in scope.

State the expected number of test cycles.

Indicate how test coverage and completion will be determined.

Define how the tests will be developed and test results captured. Describe how repeatability is assured, the extent of regression needs and how these are supported.

* Strategy: Backend – J-unit

Frontend – Selenium

Routes – Postman, Neo4j

* Constraints – short time period limits testing extensivity, Approach will be more rushed than usual projects and could limit how many bugs can be found and processed.
* Risk based - Communication, Time Management
* How much testing is required? - The amount of testing required for this application is just enough to get it passable enough to reach alpha phase release
* The type of industry - Service industry (Maintenace)
* Contract length – 9 days
* Team experience – Team experience made up of fresh testers and past developers
* Resource availability – Program in alpha phase
* Document availability – Little to no documentation very small README file.

The test approach will go as follows the application will be divided into three subsection which will be Backend, Frontend and Routes these shall be tested use the frameworks J-unit, Selenium followed by Postman and Neo4j respectively. The constraints of this testing exercise will be the very limited provided to finish the work in and could lead into risks if time not managed as well as should be. The amount of testing required for this release will be to make sure this product is to a standard to be released to the alpha audience. The type of industry this will go to is the service industry the provision for a pathway to make the technicians journey between machines as short as possible. The contract length of this testing process is 9 days therefore pointing out the aforementioned shortage of time. The experience of this team more so fresh starters to the role with varied levels of technical abilities. The availability of resources and document materials are what have been provided by the company these being the pre-alpha software and very minimal written documentation leading to the summation that this approach will require a level of explorative testing.

1. Acceptance Criteria
   1. Entry Criteria

The entry criteria for the project will be to ensure that we have a fully composed test strategy and test plan with clearly outlined test cases. A clear outline for the two weeks needs to be developed and each sprint needs to be planned in detail for how we are going to approach each section of testing. Alongside this the GitHub will need to be set up with branches for each member of the team to allow collaborative work.

* For Postman testing the entry criteria will be to have the API functioning and working in Postman to allow thorough testing.
* For Selenium Frontend testing, the entry criteria will have a documented section for postman completed and then a Selenium Framework up and running and ensuring the local host website is functional.
* For the J-Unit Backend testing the entry criteria will be to have a documented set of Frontend tests completed and then have a working database to allow the testing for linking the database and frontend.
* The Neo4J routes testing will have an entry criterion of completing and documenting the other 3 forms of testing and to have a basic understanding of how the database communicates with the frontend, this has been researched during the development of the test strategy and through testing the other sections will solidify any further understanding needed.
  1. Exit Criteria

For the entire project there are two primary exit criteria, one is that testing will stop when each sprint for all 4 types of testing has been completed and the results documented, the other is time based and testing/the project will be exited on 15/09/2022 at 16:00pm. At the end of this, and for the exit criteria to be met, all four forms of testing need to be complete, and a presentation and showcase developed to be shown on the 16/09/2022. In regard to individual sprints for each form of testing the exit criteria will be primarily time based as each form of testing has been assigned a two-day sprint, however a second exit criteria is to be assigned to each sprint. If in a one of the scrums the team decides that the testing for the current section of the project is at a strong enough and an acceptable amount of documentation has been provided, discussions will be held to determine if it is worth exiting the current sprint and using the remaining time to focus on another section of the program to ensure acceptable levels of testing have been completed all around.

* 1. Suspension Criteria

For this project the suspension criteria will be primarily time based due to strict time constraints. Each sprint will last for two days, once this time is up for each section of testing the suspension criteria will be met and testing will stop for the moment in time. Time has been allocated at the end of all sprints to allow testing to be finalised if it is pertinent that it is finished. Another suspension criteria are if that the testing being done is found to be of no use, it will be suspended, and the manpower will be implemented into a different area in hopes of providing more exhaustive testing for the project.

1. Tasks and Deliverables
   1. Test Project Plan

Define the project plan which governs the test phases and activities, summary Gantt chart style, showing testing tasks, timescales, dependencies and milestones, and resource assignments. Show how this fits with the overall project plan.

* 1. Test Milestones

| Task | Milestone | Planning Date |
| --- | --- | --- |
| Planning Completed | Have the test plan and user stories completed for each section of the webapp. | 05/09/2022 |
| Test Environment Created | Set up and create the test environment to have a functioning website on the localhost server | 06/09/2022 |
| Test Sprint 1 – Frontend | Have functioning Selenium tests executed and outcomes analysed | 08/09/2022 |
| Test Sprint 2 – Backend | Have functioning Junit tests executed with outcomes analysed | 12/09/2022 |
| Test Sprint 3 – Routes | Have functioning Neo4j tests executed, and outcomes analysed | 14/09/2022 |
| Presentation and Documentation Sprint | Have the presentation ready to show stakeholders with all required documentation | 15/09/2022 |
| Showcase | Present all work to stakeholders | 16/09/2022 |

* 1. Test Deliverables

| Deliverable | Description | Task |
| --- | --- | --- |
| The physical item created to aid the project. E.g., Test Specification. | This document will record the testable requirements for the system. | Carry out test analysis. |
| Approach & Planning | Our approach will be to have regular stand-up meetings in the morning and afternoon. The use of SCRUM sprints will also be followed to ensure the success of the team.  The contract length has been set to be nine days, meaning planning is crucial to the success of the team. | Create a Trello board and start to input tasks along with automated timeframe so we are aware when certain tasks are expected to be completed by. |
| Product Requirements | Due to no feature documentation, we will create user stories to help define how the software will be used. This would give a clear idea of tests that we may run. Explorative testing | We may possibly take an ‘Exploratory testing’ approach which would allow for us to be able to “think outside of the box” and attempt to come up with use cases that might not necessarily be covered in a test case |
| Test-Case Designs | Design test cases before the tests are written up. This will be documented on Trello. The use of the ROQ Test Script Template may also be used, this would allow for having multiple methods of documenting the test-cases | Develop test cases which would be centralised around the product requirements. |
| Testing the Levels | Ensure that the software adheres to the requirements that had been set. To access the quality, we will be assessing three major components that would typically be integrated into a system. | We will be testing the frontend, backend and the routes of the software. |
| Defect Reporting | When reporting defects, we will establish an appropriate method for posting the defects to a shared environment, this would allow for other members of the team to pick-up any issues found and peer review and test to see if we can reconstruct the same defect. | Setup appropriate defect reporting tools. In this case, we will be creating a repository in GitHub and creating different branches which would result in allowing each member to work on the project simultaneous |
| Regression Testing | Regression testing will help to ensure that any changes of the code within the software simply won’t have an inpact on the existing functionality of the software. | A GitHub respoitory created to hold the most up to date code.  Create separate branches using Bash commands, this will allow each group member to make changes without affecting the Main. |
| Route Documentation | Due to the complexity of the CrederiRoute component, it would be important to update the Javadoc so that it is clearer for us to understand how the route works. | Update the Javadoc so that it is easier to understand how the route works. |

1. Roles and Responsibilities

Define the roles which need to be undertaken to fulfil the plan, and list the responsibilities of that role. Also record who will undertake each role. Remember that each role can potentially be carried out by more than one person, and a single person can potentially undertake multiple roles.

| Role | Responsibilities | Person(s) |
| --- | --- | --- |
| Planning and organisation of work | * Update Trello * Write user stories * Write up test documentation | Connor,  James,  Matt |
| Frontend Testing | * Execute user acceptance tests * Log actual results from these tests | Connor,  James,  Matt |
| Backend Testing | * Execute intergration tests * Log actual results from the tests | Connor,  James,  Matt |
| Routes | * Execute component tests * Log actual results | Connor,  James,  Matt |

Please note that a single member of staff may be able to undertake more than one role, subject to time constraints.

1. Test Environment Needs

Where testing requires several environments with different attributes then it is advisable to split these out into separate sub headings.

Define the test environment and the support and controls needed. Identify any aspects which will need construction and build (e.g. harnesses or simulators).

Include non-computer system requirements e.g. rooms and other facilities.

Identify what exists and what needs to be procured or commissioned.

Make clear how peripheral devices might be allocated, shared and used.

Make clear how external interfaces to other systems or outside the organisation are supported.

Summarise any key administrative requirements and functions (e.g. backup/restore, batch execution and schedule management, printer management).

The environments for testing are as follows: Frontend testing requires a selenium-based testing process; Backend will require the environment eclipse; Routes will require testing in the environments Neo4j and Postman. The frontend, backend and routes will all need building and deployment prior to the testing phase.

There aren’t any non-computer requirements as this is a small testing operation therefore room requirements and other facilities are unneeded.

1. Test Data Needs

Describe the data strategy, how it supports the testing defined in this plan and how data will be built or generated.

Make clear how data is aged if this is required in the test approach.

Make clear how production data is sourced, and any timing or data protection issues associated with production data use.

Data from this testing process will be produced through the selected tools these being eclipse, selenium, neo4j and postman. The data will then be analysed for relevance and adapted for use in the presentation to stakeholders.

1. Staffing and Training Needs

Identify the staff required to deliver this plan, indicate whether these resources are available and assigned, if not how they will be obtained, identify any specialist skills required.

Consider and identify any training needs to prepare the test team for the test activity, for example testing training, training in the system under test, and training in the use of a test tool. Ensure that any training activities are included in the test project plan.

1. Test and Defect Management
   1. Test Management

Describe how the work will be test managed. Most importantly this section needs to define what the reporting channels will be for the planned work, and specifically what will be reported. Explain if and how and test tools will be used to manage or coordinate the work.

The tests tools used in each area will be specific to the needs of the tests, therefore Backend will be J-unit tests tested through eclipse, Frontend will be selenium tests through visual studio code and routes Neo4j as well as postman the findings will then be documented as mentioned above.

The J-unit tests will be used to test the backend via intergration testing this will allow for tests to be written to see if the back end is connecting the frontend to the database routes as expected and changing what it receives into a way that it can be used by the frontend.

Selenium tests will act as if a user working on the user acceptance testing this will automated and allow for when produced the scripts to carry out human like functions for test data.

Neo4j will be used for manual testing physically being able to see how test scripts affect the branching of the database itself will be beneficial in analysis and documentation of this testing exercise.

Postman will be used for API testing ensuring the API is working as expected and can be tested for possibilities where it may not act as expected.

* 1. Defect Management

Explain in broad terms how defects will be managed. Typically this involves stating the defect management tool to be used (if any) and a reference to the defect coordination document. Defect classifications and SLAs for defect resolution should be detailed here.

Raised defects will be reported through GitHub with the filling out of a template produced by the team prior, this defect will then be serialised and included into the defect report in the closing test activities. The defect flagged will be given a severity and an urgency level as well as a unique identifier with the date found and the labelled with the person who discovered the defect.

1. Assumptions

Record any assumptions used during the preparation of this plan. Assumptions are typically positive things which enable project work, but cannot currently be demonstrated or proved to be true. Also explicitly state what the impact to the plan would be if the assumption proved to be false or incorrect.

All assumptions need to be agreed by an appropriate authority, usually the client project manager (although some items are better confirmed by subject matter experts); record who agreed the validity of the assumption and when. The plan should not be signed off without all the assumptions first being explicitly agreed to.

| Description | Impact | Agreed By | Agreed Date |
| --- | --- | --- | --- |
| The set-up of backend would be easily done | Loss of 1 day of proper test set up | Connor,  James,  Matt |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Constraints

Constraints are things which do or definitely will restrict the way that work is carried out. Typically, they are unequivocal and factual in nature. As well as stating a constraint it is essential to explain the impact this constraint will have on the testing in terms of efficiency, scope or risk.

The constraints need to be signed off by a suitable authority, typically the client project manager (though sometimes a subject matter expert is better placed to agree the constraints); the purpose of this is to ensure the client is aware of any constraints and so can potentially choose to do something about them.

| Description | Impact | Agreed By | Agreed Date |
| --- | --- | --- | --- |
| Loss of third of workforce | 1/5 | Connor,  James,  Matt | 06/09/2022 |
| Issues with updates in software (eclipse) | 2/5 | Connor,  James,  Matt | 06/09/2022 |
|  |  |  |  |
|  |  |  |  |

1. Risks

A risk is something which might happen in the future and if it did would negatively impact the project. The risk is given numeric rating for impact (1-4) and likelihood (1-3). These numbers are multiplied together to generate the overall Risk Factor; the higher the risk factor the more effort should be invested to ameliorate the risk. Each risk needs an owner who is charged with monitoring the risk and taking proportionate steps to see that the risk does not occur. Alternatively, a low risk can simply be accepted by the project.

During the planning phase it is often possible to express risks as Assumptions or Constraints.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Impact | Likelihood | Risk Factor | Owner |
| Inadequate funding or delays | 1 | 1 | Not Likely |  |
| Inaccurate test activity estimation | 3 | 3 | Likely |  |
| Substantial re-work from late changes | 3 | 2 | Neutral |  |
| Unable to communicate freely with the developer | 4 | 3 | Very Likely |  |

1. Templates

The planning template supplied by ROQ will be used in preparation for the planning documentation. The test script template will also be used in conjunction with the planning template, this would offer a suitable way for us to document any test scripts that have been run.

1. Document Control
   1. Document Review

Record who has participated in both the internal and external reviews of the document. Where a person reviews a document multiple times it is only necessary to record the last date of review.

| Name | Role Title | Date |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. Change History

Record the different versions of the document which get distributed. Each new version of the document should receive a minor increment (e.g. from 0.01 to 0.02) unless the document is a major revision (e.g. the document has been signed off)(e.g. from 0.03 to 1.00). The level of information recorded in the Description of Changes field depends on the amount of sign-off received. Where external sign-off has been achieved the changes should be recorded in a high level of detail.

Not every version needs approval. Generally approval indicates passing internal review and therefore being made available to the client, or passing external review and being signed off by the client.

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description of Changes | Approval |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* 1. Distribution

List the people who the document will ultimately be circulated to. The roles are:

Review: The named person will review the document and feedback

Approval: The named person will approve the document on behalf of their organisation

Information: The names person may be interested in the document but is not reviewing or approving the document

|  |  |  |
| --- | --- | --- |
| Name | Organisation | Document Role |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. Referenced Documents

List all the documents referenced in the production of this document. Each item needs a number so it can be uniquely identified. The document title and version should be specified. Finally, it is important to record who or where the document is available from so a reader of this document can get copies of all the references documents.

|  |  |  |
| --- | --- | --- |
| Ref | Document and Version | Available from |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. Document Sign-off

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

<<Client>> Project Manager or Authorised Representative

|  |  |
| --- | --- |
| Name: |  |
| Position: |  |
| Signature: |  |
| Date: |  |

ROQ Test Project Manager or Authorised Representative

|  |  |
| --- | --- |
| Name: |  |
| Position: |  |
| Signature: |  |
| Date: |  |