***Test Plan SOEN Solutions***

**Prepared for sprint 2: 2021-02-24.**

**Prepared by Jake Lamothe 40552279**

**Table of Contents**

[**SECTION I: QA TEAM**](#_e0opb7mxrgq6)2

[1. Internal Testers](#_9m3fskos7x3r) 3

[2. External Test Resources](#_akq0vlw55saq) 3

[**SECTION II: TESTING PROCEDURES**](#_c18r4z6km08l)3

[1. General Strategy](#_43ypf3ioz0uu) 3

[a. Basic Responsibilities of Test Team](#_mmw7my9r45z3) 3

[ii. Maintain the Daily Build](#_dwr0pllf7lcg) 3

[iii. Levels of Communication.](#_xwl9n0qazuu5) 4

[2. Daily Activities](#_vxdda1h8j4y2) 4

[3. Daily Reports](#_njx6y13bs8l4) 4

[4. Weekly Activities](#_4gikjdpsrrk) 4

[**SECTION III: HOW TESTING REQUIREMENTS ARE GENERATED**](#_t51bu9oga210)5

[**SECTION IV: BUG TRACKING SOFTWARE**](#_y58uw0z1trrs)5

[**SECTION V: BUG CLASSIFICATIONS**](#_8mg7llsgk87v)5

[1. “Hard crash” bugs](#_4bv7efrzl7d0) 5

[2. “Functionality breaking” bugs.](#_j3wy73535dr4) 6

[3. “Non functional” bugs.](#_vaunhp18w9wg) 6

[**SECTION VI: BUG TRACKING**](#_sbcj98leyfgi)6

[1. Who classifies the bug?](#_kthg1dik0lki) 6

[2. Who assigns the bug?](#_rkv9l0s63nh0) 6

[3. What happens when the bug is fixed?](#_4c6kq0rry5j5) 6

[4. What happens when the fix is verified?](#_3gb7nq4kyvwg) 6

[**SECTION VII: SCHEDULING AND LOADING**](#_pkjitw62r1x3)7

[1. Rotation Plan.](#_fw5pmbn80x0t) 7

[2. Loading Plan.](#_aa2ubznpbwl4) 7

# SECTION I: QA TEAM

## 1. Internal Testers

For our project, we have the people who are responsible for writing the functional code also responsible for making the tests as well as performing their own unit tests before merging code.

## 2. External Test Resources

We will be using third party frameworks to aid in testing (primarily jest and axiom) as Node.js and React (our backend and frontend frameworks respectively) do not have sufficient native testing.

# SECTION II: TESTING PROCEDURES

## 1. General Strategy

Our General strategy towards testing is to begin with many thorough unit tests, then as the system progressively builds, we will start to focus on larger integration testing of modules. As of sprint 2 we are still only at the level of doing unit tests for our different backend and frontend components. As we require more complex tests people will be assigned to just generating test suites that focus on integration and system testing. In this document the term “tester” will be used to refer to people who code but are currently working on testing and refactoring (this will change as the complexity of the existing project increases and we assign dedicated testers).

### a. Basic Responsibilities of Test Team

The basic responsibilities of the test team are to generate test cases, find bugs, report and patch the bugs. As of now the development team is also the test team so if the bugs are small, they will be patched out or reported if they require more than 2-man hours to fix.

#### ii. Maintain the Daily Build

In order to maintain the daily build, testers will run the test suites and report any bugs whenever a branch has been merged to master (although the development team is still responsible for ensuring they do not break any of the previous code unless it was planned for refactoring).

#### iii. Levels of Communication.

Currently due to the very small modules and complexity of the product, bugs are reported to in a channel in our discord and are dealt with by the people with the most expertise (usually the person/s who wrote the code responsible).

## 2. Daily Activities

At the current level of complexity of our program there are no daily activities for testers. Rather testing is triggered by two things.   
1) the finding of a bug that is estimated to take more than 2-man hours to fix

2)after the features have been locked down then all the test suites are run as well as having as many members of the team as possible use the project to ensure current functionality and find bugs.

## 3. Daily Reports

Our “reports” are generated and maintained by GitHub. Assuming all the test passes, and no bugs have been discovered then no reports are made, however nontrivial bugs are posted to the GitHub as an issue and assigned a complexity and type to allow the team to see how important a bug is and how much time is estimated to be needed to finish it. These will also be added to google sheets.

## 4. Weekly Activities

**a.** Weekly tests

As of sprints 1 and 2 we have not had the need for weekly tests however as of sprint 3 we will be conducting a review of the master branch and its tests. This includes verifying closed bugs, ensuring non-functional requirements are maintained (if they have been included at that point) as well as creating new tests if code is lacking in tests (although the unit tests should be done by those responsible for coding). If any bugs need to be reopened or bugs have been rediscovered, then they will become issues in the GitHub and assigned in the Sunday meeting.

# SECTION III: HOW TESTING REQUIREMENTS ARE GENERATED

Testing requirements will be generated predominantly through three methods. The first being that a coder needs to have a specific code verified due to it being crucial to a function or non-functional requirement. Another is generated through the weekly meetings that are held by the testing team if they find certain bugs that are not being caught by the test cases then the requirements will need to be changed to include these bugs. Lastly the team has already planned that most nontrivial methods will need to have unit tests and integration tests when the interaction between modules/methods may produce unexpected results. Testing requirements may also be generated if during the Sunday meeting a pattern is noticed (e.g., a particular segment of code keeps producing bugs after many patches).

# SECTION IV: BUG TRACKING SOFTWARE

In order to track larger bugs, we will be using a mixture of discord and GitHub and google sheets. Discord will be used to notify relevant members of the bugs, the type and the scale. git hub will be used to create a prioritised issue to assign to someone who can then fix it, upload a patch and close the issue. Lastly sheets will be used to create easy to fill tables to create a report that is easy to gather data from.

# SECTION V: BUG CLASSIFICATIONS

## 1. “Hard crash” bugs

Hard crash bugs are any bugs that can cause the product to “crash”. This can include causing the database to stop responding or causing issues with the server that the website is hosted on.

## 2. “Functionality breaking” bugs.

These bugs break the functionality of the program. An example would be a bug that does not let the user perform otherwise valid actions like logging in or adding a new business location.

## 3. “non-functional” bugs.

These are bugs that although do not break the functionality of the program do cause issues that can range from annoying to severely important. An example of both could be unnecessarily long loading times or a bug that allows users to perform actions that they do not have the permission for and breaking the security of the database.

# SECTION VI: BUG TRACKING

## 1. Who classifies the bug?

Testers when they find the bug or coders if they notice bizarre behaviour unrelated to their section during execution. These will be placed in the “type” column in sheets.

## 2. Who assigns the bug?

Bugs are self assigned. When coders/testers have the available time, they will self assign the tasks (or will be requested to self assign by a member in the discord).

## 3. What happens when the bug is fixed?

When the bug is fixed it will be merged with the main branch after local testing has been done to verify that the bug has been successfully patched but also does not introduce any new bugs (due to time constraints we measure this by doing a quick run through as well as running the test suites).

## 4. What happens when the fix is verified?

The issue is closed, and the branch is merged to master.

# SECTION VII: SCHEDULING AND LOADING

## 1. Rotation Plan.

We use a similar methodology to the method plan where at least one member from the coding team will be present for testing while having some members of the documentation team assist with testing to ensure both familiarity and fresh perspectives.

## 2. Loading Plan.

For sprints 1 and 2 we currently have tester and coders as the same group however, as the complexity increases, we will be adding more members from the documentation team to assist in testing which will add fresh eyes to the tests as well as freeing up some members of our coding team to continue working on new features. This is subject to change as if a large amount of man hours is needed to fix a bug or a set of bugs then more members will be assigned to the testing team from the coding team and new features will be temporarily halted to prevent new bugs from piling up.