TEST PLAN

- Test plan id

Fast Food Fantasy Test Plan

- Purpose of the test plan

This test plan will help facilitate the testing process of the project to ensure rigidity in the code, algorithms, and UI.

- Scope of the test plan
    - What will be tested

Testing will include the UI functionality

- restaurants displayed in a scrollable list as buttons to open their respective menu
- Menu buttons for +\- items
- Window changes

Planning trips algorithm for shortest route

- custom trip
- 10 initial from saddleback
- 12 restaurants shortest route
- Dominos shortest route

Maintenance controls

- Log-in functionality only one correct code
- Restaurant menu +/- items
- Change price functionality
- +\- restaurants

- Overall test strategy

The overall test strategy revolves around the entire team following the activity diagram to run through each element of the program. An element or part of the program will be tested directly after it is completed to ensure the main branch runs as intended. It will first be tested by the creator, after the test is passed then the element will be tested by the whole team. A final test will be done at the completion of the stories to find remaining bugs.

- What features will be tested from a user's perspective

Users perspective includes
viewing the menus of each restaurant in the program
Show trip options:

- the custom trip plan
- 10 initial restaurants trip
- 12 restaurants trip

- dominos trip

Display distances of restaurant trips

- maintenance (change menu and +\- restaurants)
- The system will import the given file and add the information to a container

- What features will not be tested from a user's perspective and what the system does

Features not tested from users perspective

- SQL database
- GUI structure
- UML diagrams
- Test plan

- Entry criteria
    - Conditions needed to start the testing

To start testing, a running element of the program must be completed. The element should not produce errors. A git pull should be done before testing to ensure compatibility.

- Exit criteria
    - When to stop testing

Testing will be complete when each team member has gone through the activity diagram and program, checking off each element as correct and meeting requirements. Team will decide when the exit criteria is met.

- Suspension criteria
    - Conditions when to temporary stop testing

Testing will be suspended when:

- an error occurs during testing
- Unexpected output occurs
- An element is incomplete when approaching the sprint deadline
- A fault is found in the algorithm/plan

- Roles and responsibilities
    - Product owner, scrum master, team

Product owner:

- ensures team members are testing according to the plan
- Tests elements produced by themselves
- Tests the entire program once stories are complete

Scrum master:

- Logs the completed tests

- Tests elements produced by themselves
- Tests the entire program once stories are complete

Team:

- Tests elements produced by themselves
- Tests the entire program once stories are complete
- Has final say on when the feature or program is complete

- Approval process

Test is approved by the entire team to ensure nothing is missed. The team will take a majority vote once an element is completed and tested whether to move on or if changes are needed. The developer working on the element will have the final say if a tie occurs because they will have the best idea of how it will work.

- White box or black box testing or both?
- We will be using black box testing which is a method of testing software that examines the functionality of an application without reviewing the source code, which also means the test cases will be built around specifications and requirements

- Schedule
    - When to testing will commence

Testing will commence as soon as an element of the program is completed. Each element is tested individually as they are completed. After all elements are completed, the team will test the program as a whole.

Sprint 1:

- 3 use cases
- 3 state diagrams
- 1 class diagram
- Activity diagram(s)

Sprint 2:

- SQL database
- UI of restaurant list
- Importing file and fill list
- Implementation of custom trip, 10 initial trip, dominos trip, and 12 restaurants trip
- Implementation of purchasing items and showing total spending

Sprint 3:

- admin log in
- Maintenance of restaurants
    - Change price
    - Change menu items

    - Add/remove restaurants
- Testing / big O analysis

- Necessary training needed for testing

There is no necessary training for testing, if someone does not know how to test a feature, the rest of the team will discuss how to test it.
SQLite training will be necessary

- Environment description
    - Hardware and software needed for testing

The testing environment will be solely on windows operating systems and QT

- For software, we will be using Qt Test as a framework for unit testing as well as testing graphical user interface. For hardware, we all use Windows OS and make sure it runs the testings

- Configuration management approach
    - GITHUB utilization
    - Branch structure
    - What happens when a test fails

GitHub is used to hold user stories, diagrams, and the code.
Each team member will have their own branch to code and test in, this will be merged with main branch for final testing.
Each team member has their own branch to edit and test in. (If excess branches are needed, they are labeled and removed once satisfied)
The main GitHub branch will be the running branch that working code is merged into.
Pulls will also come from the main branch.

- Test deliverables
- Class Diagram
- Activity Diagram
- 3 Use Cases
- 3 States Diagrams
- Qt website provides testing process/actions
- Qt creator for testing environment
- Big-Oh analysis of at least five methods
- Test plan
- Agile stories
- Scrum log

- Documents that support the test plan

- Qt website for QT testing resources (unit testing)
- SQL website
- GITHUB

- Glossary of terms
- Element: a part or segment of the program that can be tested
- GITHUB: version control management and system that allows collaboration
- QT - widget toolkit for creating graphical user interfaces and applications.