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, algorith

- 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 e
branch runs as intended. 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, the custom trip plan,

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

Features not tested from users perspective

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

- 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

- 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

- Approval process

Test is approved by the entire team to ensure nothing is missed.

- 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.

- 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.

- 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 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
- Compare the uml diagrams to the tested element for proper implementation
- Qt website provides testing process/actions
- Qt creator for testing environment

- Documents that support the test plan
- Qt website for QT testing resources (unit testing)

- Glossary of terms
- Element: a part or segment of the program that can be tested

1. During development, the person working on a story will test inputs or functionality of the code. Once the story and tests are complete from the developer, it will be pushed to a testing branch for the team to individually test before meeting to discuss problems or if no problems are found, it will be merged with the main branch.
2. If there is a disagreement on how a story/element of the program should be, it will be taken to majority vote and if a tie occurs, the primary developer of the story/element will have the advantage in the vote.
3. To begin testing, first, ensure the method you are testing is complete (take steps necessary to meet the requirements of the story). Then, analyze how testing will be done (different number of inputs, different orders of actions, etc.). Finally, input/test the function (making changes if necessary).
4. Two stages of testing: unit testing and testing the project as a whole. Begin with testing of individual methods/stories and once successful, merge with the testing branch to check for successful integration.
5. All group members will contribute testing to ensure the feature works as intended.
6. After testing is complete the product owner will take a look at the implemented feature and decide whether it meets the requirements.
7. Hardware/Software: 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.
8. 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.
9. Scheduling: Testing will commence after each story completed and each function added.
10. To finish testing, each team member must follow the uml diagrams to ensure the project aligns with the requirements