# Test plan for

# European Vacation Project

Milo Fisher, Aaron Geesink, Garrett Geesink, Noah Agudelo

*ChangeLog*

| Version | Change Date | By | Description |
|---|---|---|---|
| version number | Date of Change | Name of person who made changes | Description of the changes made |
| 0.5 | 10/6 | Aaron Geesink | - Added Shortest Distance Algorithm<br>-Created database querying<br>-calculates shortest distance for 11 starting cities<br>-started basic edit window |
| 1.0 | 10/19 | Aaron Geesink | -Can turn on and off cities in database<br>-added London Plan and Paris Plan<br>-added food selection and food receipt<br>-can add, remove, and edit foods |

# 1 Introduction

The Test Plan documents and tracks the necessary information required to effectively define the approach to be used in the testing of the product. The Test Plan is created during the Planning Phase of the project. Its intended audience is the project manager, project team, and testing team. Some portions of this document may on occasion be shared with the client/user and other stakeholder whose input/approval into the testing process is needed.

## 1.1 Scope

1. Test the ability to display all the traditional food items for any given city.
2. Test the ability to visit the initial 11 European cities starting at Paris.
3. Test the ability to plan the **shortest** trip starting at London
4. Test the ability to plan a custom trip
5. Test the ability to purchase multiple traditional food items
6. Test the ability to keep track of the number of traditional food items purchased at each city
7. Test the ability to display the total amount spent at each European city and a grand total for all cities visited

8. Test the ability to add new European cities and their corresponding food items by having your program read from the input file given to the class

9. Test the ability to change the prices of the traditional food items

10. Test the ability to add new traditional food items

11. Test the ability to delete traditional food items

12. Test the ability to visit the 13 European cities starting at Paris.

## 1.2 Quality Objective

- Ensure the Application Under Test conforms to functional and nonfunctional requirements
- Ensure the product meets the quality specifications defined by the client
- Bugs/issues are identified and fixed before go live

## 1.3 Roles and Responsibilities

- Product Owner - Envisions the complete product and presents the project during sprint reviews and assesses its process. Also determines what constitutes as a finished or unfinished project.
- Scrum Master - Works with the product owner to create tasks that the team members can implement in code. Also eliminates any distractions for any member in the project, and keeps track of the scrum log.
- Team Member - Coders that implement project requirements into a working product that meets the standards of the product owner and scrum master.

# 2 Test Methodology

## 2.1 Overview

We are using Agile to meet requirements set by Professor Lebowitz for this project. The use of Agile helps us accomplish a significant project in smaller sprints, which allow for a more effective way to produce a functioning project in smaller, usable increments, rather than one large implementation that happens a long way down the road.

## 2.2 Test Levels

Test functionality of all major aspects of the product as well as all listed requirements.  Make sure the user can plan a trip successfully by choosing food items and cities to visit, along with accounting for incorrect inputs and making sure the whole projects works smoothly and as intended.

## 2.3 Test Completeness

- Product working as intended
- User can successfully plan a trip
- All cities and food items are available
- Database features are all enabled
- Admin can log in successfully
- Invalid input does not cause fatal errors

# 3 Test Deliverables

Artifacts available during testing

- Code
- Test Plan
- Agile Stories
- Scrum Log
- Coding Standards
- Team Rules
- UML Diagrams
- Data Structures Used
- Big Oh Analysis

# 4 Resource & Environment Needs

## 4.1 Testing Tools

Qt Creator was the main tool used in testing purposes as it ran the project, displaying any errors to the console and display and visual issues or functionality issues in the run time viewing of the project.

## 4.2 Test Environment

The following **softwares** are required:

- Qt Creator 5.13.0 with a 32-bit C++ compiler kit
- Windows or Mac Operating System