Test Plan and Cases (TPC)

Fooder

QWERTY

Jake Motta - Co-Lead Programmer, Graphics Designer

Alex Le - Project Manager, Co-Lead Programmer

Brandy Kao - Co-Lead Programmer, Tester

Jason Springer - Co-Lead Programmer, Quality Assurance

3/8/2017

Version History

Date	Author	Version	Changes made	Rationale

3/08/17	JS	1.0	Initial app prototype created	· Created without any dependencies
3/14/17	JS	1.0	Menu icons updated, swiping capabilities added	·

Table of Contents

Test Plan and Cases (TPC) i
Version History. ii
Table of Contents.. iii
Table of Tables.. iv
Table of Figures.. v
1. Introduction. 6
2. Test Strategy and Preparation. 7
2.1 Hardware preparation. 7
2.2 Software preparation. 7
2.3 Other pre-test preparations 7
2.4 Requirements Traceability. 7
3. Test Identification. 8
3.1 Test Identifier 8
3.2 Test Identifier 10
4. Resources and schedule 11
4.1 Resources 11
4.2 Staffing and Training Needs 11
4.3 Schedule 11

Table of Tables

Table 1: Requirements Traceability Matrix 8
Table 2: TC-01-01 Check report completeness 9
Table 3: TC-01-02 Check report correctness 9
Table 4: Testing Schedule 11

Table of Figures

Figure 1: <Figure Title>. 21

1. Introduction

Initial barebones testing of Fooder conducted through the use of similar style apps, such as Tinder, Grinder, and Yelp. The focus of this is to see the functionality, style, color scheme, and audience of similar existing successful apps, and understand why these characteristics work together in the app.

As Fooder develops, we will begin to implement test comparisons between style and response time to that of Tinder and Yelp, ensuring that we are of comparable (if not better) rate of app use.

2. Test Strategy and Preparation

Fooder test strategy follows an environment similar to that of “value-based test prioritization”, where we’ll be testing based off several priority steps:

1. Determine Risk Exposure for Each Test Case (RE)
2. Calculate Risk Reduction Leverage for Each Test Case (RRL)
3. Prioritize the Test Cases According to their RRL Value

2.1 Hardware preparation

Hardware preparation for Fooder includes:

- 2 x Laptops
    - Laptop 1: MacBook Pro Retina 13-inch
        - 2.8GHz, Intel Core i5, 8GB RAM
        - OS X 10.12.3 (Sierra)
    - Laptop 2: Lenovo
        - Windows 10 Home (10.0.14393 Build 14393)
        - Intel Core i7-6700HQ, 2.60GHz, 8GB RAM

- 2 x Cell Phones
    - Phone 1: Iphone 7 Plus
        - iOS 10.2.1
    - Phone 2: LG V20
        - Android 7.0

Laptops are used within our testing environment to research any web-based applications that we deem similar style to our ideal application for Fooder, and for virtual application building and testing. Both phones are used to run Tinder, Grinder, and Yelp applications, ensuring that any noticable in-app differences between Android and iOS are noted.

2.2 Software preparation

Software preparation needed included:

- Tinder
    - iOS: Version 7.1.0
    - Android: Version 6.8.3
- Grinder
    - iOS: Version 3.4.1
    - Android: Version 3.2.0
- Yelp
    - iOS: Version 11.10.1
    - Android: Version 9.6

Tinder and Grinder are used for similar app design and functionality reference. These two apps provide profile privacy and security provided through Facebook Login API. These apps also inform the user that no associations between the app and Facebook are made, so that the user has no fear of their apps being linked, besides login functionality.

Yelp was used for context on how the app's database functions, how complex the menus could be, and how in-depth users could define search criteria.

2.3 Other pre-test preparations

Apps described in 2.2 needed to be downloaded to all phones described in 2.1.

2.4 Requirements Traceability

Table 1: Requirements Traceability Matrix

Requirement ID	***Verification Type***	***Test Case ID (if applicable)***
AR-0 Application Reference	*Testing*	*TC-01: Application Reference*

3. Test Identification

3.1 Test Identifier

TC-01 Application Reference: Testing and observation of various applications described in 2.2

3.1.1 Test Level

TC-01 Application Reference:

- Unit Testing - validate how each unit of the software performs

3.1.2 Test Class

TC-01 Application Reference will employ timing tests and functional tests

3.1.3 Test Completion Criteria

The test for TC-01 Application Reference will be completed when:

- All apps defined in 2.2 have been installed
- The following app features have been tested and documented:
    - General app functionality
    - Shapes and color scheme
    - User-enrollment
    - User-friendliness
    - Response time
    - User in-and-out time

3.1.4 Test Cases

Table 2: TC-01-01 Application Reference completeness

Test Case Number	*TC-01-01 Application Reference completeness*
Test Item	-*General app functionality*: How does the app work? -*Shapes and color scheme*: What kinds of shapes does the app employ? Are things rounded? What colors do they use? -*User-enrollment:* How can user sign up for the app? -*User-friendliness*: Is the app hard to use? -*Response time*: Does the app lag? How quickly do menus appear and actions perform? -*User in-and-out time*: How long does it generally take for a normal user to open the app, use the app, and close the app?
Test Priority	*<<The relative importance of this test to satisfy the project requirement of the client organization; using the MSCW (MoSCoW) prioritization scheme: M (Must have), S (Should have), C (Could have), W (Want to have)>>* *This test*

Pre-conditions	*<< Describe the prerequisite conditions that must be established prior to performing this test case>>*
Post-conditions	*<< Describe the conditions that must be established after performing this test case>>*
Input Specifications	*<< Describe each input required to execute the test case>>*
Expected Output Specifications	*<<Identify all the expected results of the test case>>*
Pass/Fail Criteria	*<< Identify all the pass/fail criteria to be used for evaluating the results of this test case>>*
Assumptions and Constraints	*<< Identify any assumptions made or constraints imposed in this test case>>*
Dependencies	*<< List the identifiers of the test cases that depend on this test case i.e. the test cases that must be executed prior to the execution of this test case>>*
Traceability	*< < Provide mapping to requirement(s) from SSRD >>*

3.2 Test Identifier

<< Repeat the same structure for the next test identifier.
For example: TC-02 Submit job requests >>

3.2.1 Test Level

3.2.2 Test Class

3.2.3 Test Completion Criteria

3.2.4 Test Cases

4. Resources and schedule

<<In this section, specify the people, time, budget, resources and schedule allocated for testing, etc. >>

4.1 Resources

<< Identify all resources need for testing, such as test data set, software, budget, and etc >>

4.2 Staffing and Training Needs

<< Identify the stakeholders responsible for managing, designing, preparing, executing, witnessing, inspecting and resolving test items. In addition, provide the groups responsible for providing items to be tested. Specify test-staffing needs by skill level. Identify training options for providing necessary skills. >>

4.3 Schedule

Table 4: Testing Schedule

Date	**Test Identifier**	**Responsible person**	**Resources**	**Training needs**
01/02/09	*TC-01-01 to TC-01-04*	*John Smith*	*Report test data sets, JUnit*	*N/A*