Zachariah Collins
Alexander Cordero
Daniel Strawn

CSCI 3308
Main Project
Software Development Tools and Methods

Fridge Forager
Project Writeup

- **1. Introduction** :

  The goal of this project was to develop a simple mobile application for the Android platform, the purpose of which is to return possible meal recipes when given one or multiple ingredients as search queries.

  GitHub Repository:
  `https://github.com/nzimmermann/Fridge-Forager`

  Project Tracker Page:
  `https://trello.com/b/MLj3y3rn/software-methods-and-tools-project`

  Android .apk file:
  `https://github.com/nzimmermann/Fridge-Forager/tree/master/Searchview/app/build/outputs/apk`

- **2. Methodologies Employed** :

  The following is a brief list of the methodologies used used throughout the development of the app:

  - Peer Code Review
  - Test Driven Development (a lot of it)
  - Remote Development and Collaboration via project tracker and cvs ...
  - Agile Development model
  - Pair Programming

  We assigned our most experienced java programmer the task of coding, a designer with the task of designing, and a project manager to keep check on everything and direct workflow.

- **3. Software Tools Employed** :

  The following is a brief list of the Software tools and suites used throughout the development of the app.

  - Android Studio
  - Food2Fork API
    1. Provided an effective means with which to get recipes when queried with ingredients
    2. Trivializes the neccessity to use an offline or locally created database
    3. Provided experience developing with a third party API
    4. `http://food2fork.com/about/api`
  - JavaDoc via Android Studio
    1. Changes documented in git repo
  - LaTeX
    1. Used to write this document
  - Git Content Tracking System hosted via GitHub.com
  - Trello for project management
    1. Cross Platform
    2. Android App available
    3. Emailed participants with project updates, changes, and to-dos
  - Lint Static Analysis

- **4. Test Driven Development** :

  The development of our app was handled in concise test cases such that progress was made one small step at a time; never too much too quickly.

  1. Test Case 0: Basic Test, Compilation (recurring)
  2. Test Case 1: Food2Fork API Usage
  3. Test Case 2: (basic functionality) Assuming previous tests pass
  4. Test Case 3: (basic functionality part deux) Assuming previous tests pass
  5. Test Case 4: (minor functionality), Assuming previous tests pass

Case	Test	Test Description	If Fail
0	a	Is the code syntatically correct?	Static Analysis, Peer Code Review
0	b	Does the code correctly call Android functions and classes without error?	Revert to previous test, repeat
1	a	Does the code correctly call the Food2Fork API?	Revert to test 0.a, consult Food2Fork API documentation
1	b	Does the code correctly pass user-inputted queries to the API?	Revert to test 0.a
1	c	Does the code correctly parse the JSON returned by the API?	Revert to test 0.a
2	a	Does the app install in the emulator without error?	Static Analysis, Peer Code Review
2	b	Does the app start in the emulator without error?	Revert to test 0.a
2	c	Are the app's UI elements correctly layed out in the emulator?	Revert to test 0.a
2	d	Does the app correctly accept a query in the text field without fail?	Revert to test 0.a
2	e	Does the app correctly return a list of recipes pertaining to the query entered without fail?	Revert to tests 0.a, 1.c
2	f	When a recipe is selected, does the app correctly launch the Emulator's default web browser to the webpage containing the recipe without fail?	Revert to test 1.a
3	a	Repeat all tests from Case 2 on a live Android phone rather than an emulator	Revert to test 2.a
3	b	Repeat all tests from Case 2 on a live Android tablet rahter than an emulator	Revert to test 2.a
4	a	Does the app use the correct display icon?	Revert to test 0.a
4	b	Does the app rotate correctly if the device's orientation changes without fail?	Revert to test 0.a
4	c	Does the app correctly call third-party web browsers other than the system default without fail?	Revert to test 0.a