Software Requirements Specification

Restaurant Finder

Prepared by Ethan Kolkmeier, Filimon Gebrekidan, James Durflinger, Carl Trautwein
Team: The Developers
October 02 2019

Table of contents

1. Introduction

1.1 Purpose

The purpose of this project is to develop an android application called Restaurant Finder. Restaurant Finder is an application that will assist users in making dining decisions for themselves or a group by filtering out options. The application will also allow users to experience new restaurants and food by making suggestions and random selections.

1.2 Scope

Restaurant Finder provides the user with relevant information about each restaurant selected, such as distance, crowding, and price. The app provides this information in a long list format as well as a map view. In addition Restaurant Finder helps people decide based on their preferences, moods, or by random. Each user will have their own account as well as favorites to better tailor what restaurants get recommended by the app. Restaurant Finder will work in Denton, TX, but should scale to any location supported by the Google Maps API.

1.3 Definitions, Acronyms, and Abbreviations

- **Mbps**: Megabits per second
- **GB**: Gigabyte
- **RF**: Restaurant Finder
- **FB**: Firebase
- **ms**: millisecond
- **UI**: User Interface
- **API**: Application Programming Interface

1.4 Overview

An overview of the Software Requirements Specification Document

- Section 2 provides a description of the application
- Section 3 provides the requirements for the application
- Section 4 includes the appendix

2. Description

2.1 Product Perspective

One often joked about trail that many people face is deciding where to eat. Many people are fortunate enough to live in areas with many dining options, and, for some, the sheer amount of options can be overwhelming. Existing software like google maps, yelp, and many others try to help others help make these decisions, but often rely on truthful opinions of other users, or resort back to one huge list of locations (often making the user more overwhelmed). Restaurant finder aims to help these customers decide where to spend their time and money based on user ratings as well as the individual preferences of the user. Whether it be at a five-star restaurant or a fast-food joint RF's goal is to help people make quick decisions on food that they will be satisfied with.

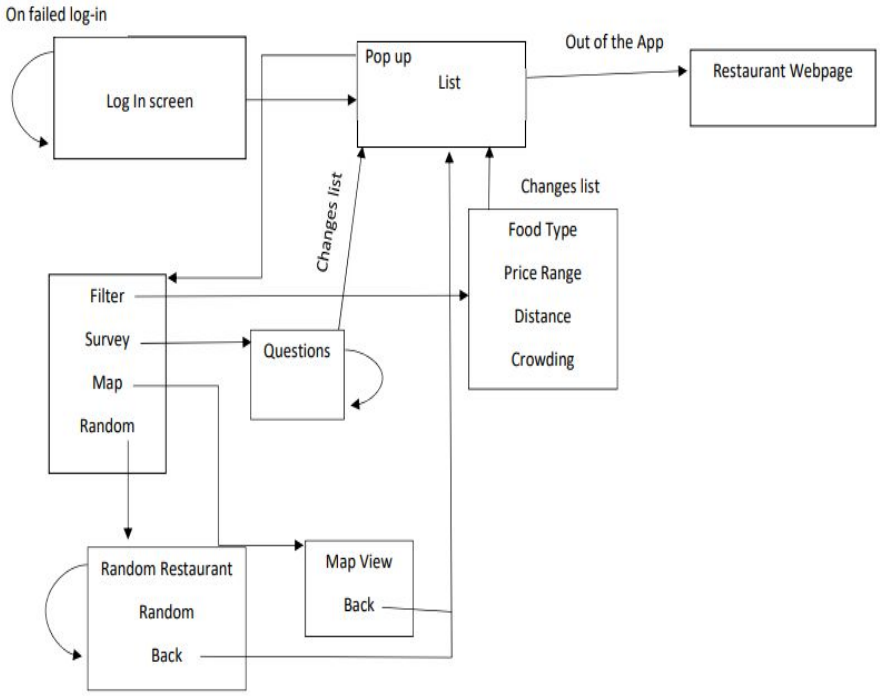

Figure 1:user interface

2.2 Product functions

Restaurant Finder(RF) will compile a list of all restaurants within 50 miles of the user. Using this data RF will be able to show the user some suggestions. The app will also offer users a short quiz and make restaurant suggestions based on the results. If all else fails RF will offer the user a "Random" option that picks a random restaurant from the list. Each user will have a unique username and password which they will use to sign in. Each user will be able to assign their own restaurant preferences, which the app will store in a database and suggest more often.

2.3 User Characteristics

Primary users of RF will be people who eats outside of their home often. A typical user may be a young student who eats at various fast-food restaurants. Other expected users will be people who eat at the same dining establishments and need help discovering new places. However, anyone who wants to eat out for any occasion could be assisted by RF.

2.4 Constraints

Options are limited to Google APIs and tools for Android Studio. Also, price and time will be limiting the scope of this project. The application will only be making use of free software interfaces and tools and will need to be completed by the end of this semester.

2.5 Assumptions, and Dependencies

The application must be functional in English speaking and developed regions of the world. The app must also be able to perform with 5 Mbps and above (which should be sufficient for performing a search query).

3. Requirements

3.1 External Interface

3.1.1 User Interface

The application's UI will include a login screen, which will have text boxes for users to input data, such as user name, email address and age. Once a user is logged in, the app will have a search screen for sorting through restaurants based on user specifications. The app will also have a screen for displaying found restaurants in list form, each object in the list should have the functionality to redirect to the restaurant's website if one is available. In addition, there will be a map of the surrounding area which displays both the user and the closest 10 restaurants. The app will also have a survey mode where it asks the user multiple choice questions (ie “How does Mexican sound?”) and then display a list of possible choices. A random button will be available that displays one random restaurant, this button can be pressed as many times as the user wants.

3.1.2 Software Interface

The application will connect with other Google services for map data and drawing a route on the screen. Login information, user preferences, and favorited restaurants can be saved in Google’s Firebase(FB): an online database service.

3.1.3 Communication Interface

The application will require the internet to browse for restaurants, restaurant info, and for connecting to external software interfaces. For this application, it will need to connect with FB. Also, with email information, the application can send back login details in case the user forgot their password.

3.2 Functional Requirements

3.2.1 Log-in

3.2.1.1 Login Success 1.1

When users input correct credentials allow them to login to the ‘Main’ screen.

3.2.1.2 Login Failed 1.2

Tell user “Login Failed” and allow them to try again.

3.2.2 Main Page

3.2.2.1 List 2.1

Go to the 'List' mode.

3.2.2.2 Survey 2.2

Go to 'Survey' mode.

3.2.2.3 Random 2.3

Display one random restaurant.

3.2.3 List

3.2.3.1 Filter 3.1

User can select to view subsets of the big list based on things like distance, price, and food type.

3.2.3.2 Go To Restaurant 3.2

If a user selects a restaurant they are directed to a larger information panel with restaurant information including website, address, etc.

3.2.3.3 List 3.3

Display restaurants in a list view.

3.2.3.4 Map 3.4

Display restaurants in a map view.

3.2.3.5 Favorite 3.5

Add a restaurant to user's list of favorites.

3.2.4 Survey

3.2.4.1 Give Survey 4.1

Ask user a series of questions to determine what to recommend.

3.2.4.2 Go Back 4.2

Go back to 'Main Page' mode.

3.2.4.3 Display Results 4.3

Display some recommended restaurants based on the results of the survey.

3.3 Performance Requirements

The sign in process will take no more than 50 ms because of the data being stored locally. Upon sign in, the phone should get all restaurants within 50 miles which should take less than 1 second with an internet connection. If the user changes the search distance the app should query again up to a maximum of 200 miles and should take less than 10 seconds to return results.

3.4 Design Constraints

Because developing ios application requires an annual developers fee, the app will only be available on android devices [2]. RF is also constrained by the scope of the Google Maps API. We will be using Android Studio which primarily uses Java to create mobile apps. FB has a “Spark Plan”, which is free but is limited to 1GB of storage. Finally, because RF is a mobile app it will have memory and size constraints depending on the device using it.

4. Appendices

[1] Google Developers. (2019). *Overview | Places API | Google Developers*. [online] Available at: https://developers.google.com/places/web-service/intro [Accessed 23 Sep. 2019].

[2] T. Mackenzie, "App store fees, percentages, and payouts: What developers need to know," TechRepublic, 2019. [Online]. Available: https://www.techrepublic.com/blog/software-engineer/app-store-fees-percentages-and-payouts-what-developers-need-to-know/. [Accessed: 23- Sep- 2019].