****

**Software Requirements Specification**

**for**

**<**Jiak Lah!**>**

**Version 1.0 approved**

**Prepared by < S**ze Wei Yan, Noel Wee, Pramod Sanjay, Rizwan Nusrath Fathima**>**

**<**Foodies**><**2 February 2023**>**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

# **Introduction**

## **Purpose**

The purpose of this document is to provide a detailed description of the proposed website, *Jiak Lah!*. This document includes information on the purpose, features and interfaces of the website as well as any design considerations.

## **Document Conventions**

The following conventions were followed in this document write-up:

Font size: 11

Font type: Arial

Font style: Bold is used for headers

## **Intended Audience and Reading Suggestions**

This SRS document is intended for the following group of people:

| Group | Purpose of SRS document |
| --- | --- |
| Clients | Formal documentation which serves as an assurance that the organization understands the problems to be solved and the software behaviour necessary to address these problems.  Clients may refer to the *Overall Description,* [*User Interface*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.2jxsxqh)and *Other Nonfunctional Requirements.* |
| Developers | Formal documentation which serves as an input to the design specification built on the agreement of requirements with the client.  Developers may refer to the [*Overall Description*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.g2b8ewxv1sgk)*,* [*Design and Implementation Constraints*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.slnf40ge2838)*, System Features* and *Other Nonfunctional Requirements* |
| Testers | Formal documentation which serves as a guideline to develop structured testing strategies, which will be applied to the requirements for verification in the testing stage.  Testers may refer to the [*Overall Description*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.g2b8ewxv1sgk)*,* [*Design and Implementation Constraints*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.slnf40ge2838)*,* [*User Interface*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.2jxsxqh)*, System Features, Other Nonfunctional Requirements*, Initial Dialog Map *(References)* and *Appendix C: User Manual.* |

## 

## Table 1: Purpose of SRS document for different stakeholders

## **Product Scope**

Singapore’s ever-changing and complex socio-economic climate has culminated in a melting pot of cultures and flavours. Our Little Red Dot prides itself on its food and rightly so, seeing how food plays an integral role in today’s modern society, with culinary institutions such as hawker centres, street food vendors and restaurants being commonplace. Unsurprisingly, the wide array of food choices available has also given rise to food recommendation applications and websites, each endeavoring to provide its consumers information on the best deals and gastronomic delights. Said applications and websites also allow users to check up on other information pertaining to these food establishments such as location, type of cuisines available and prices amongst others, all while enabling users to leave reviews as much as they please.

While these platforms are more than apt in providing details of their listings, they are not perfect. Based on our personal observations, these platforms, more often than not, do not elaborate on the accessibility of the eatery for disabled people nor do they state any implementations in place to aid disabled people dine in the eateries they list. Ergo, we have identified this issue and we are of the opinion that this is a missed opportunity. We should strive to be more inclusive by making these disabled people’s lives easier, allowing them to enjoy their favourite meals without any inhibitions.

With Jiak Lah!, users may no longer find the process of identifying suitable disability-friendly eateries a hassle. In addition, Jiak Lah! also strives to provide more detailed information such as any existing infrastructure in place to help the disabled, or if eateries employ staff that are trained to help the disabled. All these are readily available and easily accessible with just a click of a button, bringing much convenience to our consumers.

**References**

| Use Case Model | Use Case Diagram, Use Case Description |
| --- | --- |
| Source Code | - |
| Sequence Diagrams | Login, home, search, view review, leave review, profile |
| Dialog Map | - |
| PHP File | - |
| User Manual Video | - |
| Class Diagrams (jpeg + vpp) | Class Diagram (Per Use Case) |
| Work Breakdown Structure | - |
| UI Mockup | - |
| Test Cases | - |
| Meeting Minutes | - |
| Data Dictionary | - |
| System Architecture | - |

Table 2: References

# **Overall Description**

## **Product Perspective**

*Jiak Lah!* is a website which relies on the Google Places API to retrieve the location and the eateries’ quintessential details. *restaurantFinder* also stores and retrieves user-related information in the 4 databases created using PHP, which is elaborated in [*Communication Interface*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.dfl3vh4e0lts). A Model-View-Control design pattern is implemented in the system architecture and a layered architecture design in the backend system. The logic flows from top-down in the hierarchy and the façade layer enables this design to be more scalable as changing one layer would not have a significant impact on another. The classes in the system also each have a single responsibility, which follows the Single Responsibility Principle, adding high cohesion and low coupling.

## **Product Functions**

***Diagram***

Jiak Lah! consists of the following functions that users are able to perform:

1. User Register
   1. Register new user to database
2. User Login
   1. Authenticate user
3. Search
   1. Allows users to search for restaurants through string input
4. Filtered Search
   1. Allow users to search for restaurants through filters such as price and type of cuisine available
   2. Allow users to set their profile preferences
5. Listings Page
   1. Display detailed information of restaurants such as telephone number, address, website link, location on map as well as ratings and reviews
6. Review
   1. Allow registered users to post reviews and rate eateries
7. Report
   1. Allow registered users to report any problems occurred on the website

## **User Classes and Characteristics**

The expected users of the website are everyone including those with disabilities to search for their preferred eatery.

1. Guest Users:
   1. Able to perform basic searches for eateries by name, or filter them using our in-built filters.
2. Registered Users
   1. Able to leave reviews and ratings on restaurant page

## **Operating Environment**

*<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>*

## **Design and Implementation Constraints**

A stable internet connection is required as users need to access and utilise our JiakLah! website.

**User Documentation**

Refer to [*Appendix C: User Manua*](https://docs.google.com/document/d/1n9n0xzxzsPINOo4UTC4dSdIOjs8L3M--/edit#heading=h.fa7qtodf2vyp)*l* for the user guide to navigate the website.

## **Assumptions and Dependencies**

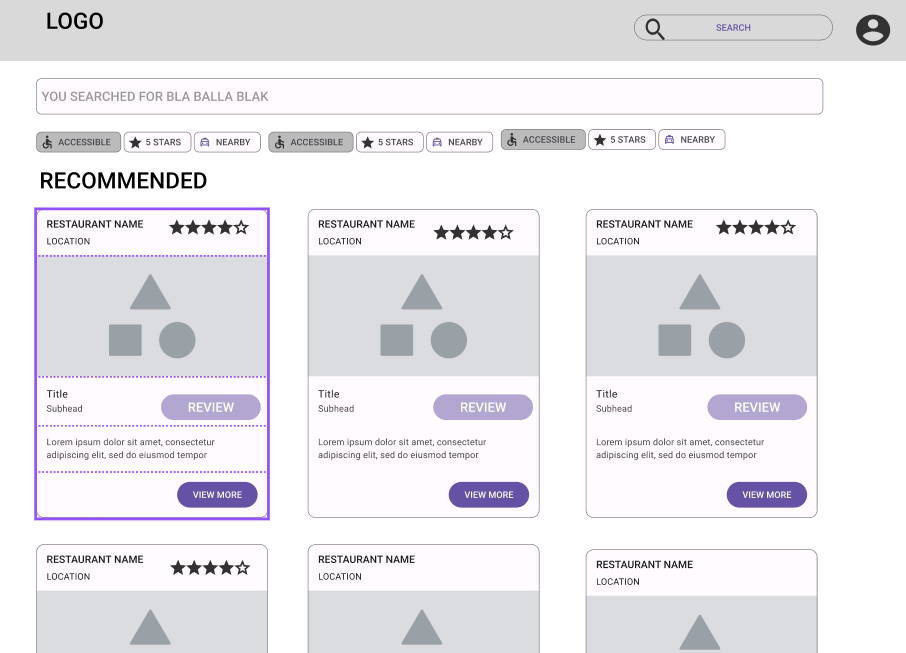
* Users have a stable internet connection on their devices

# **External Interface Requirements**

## **User Interfaces**

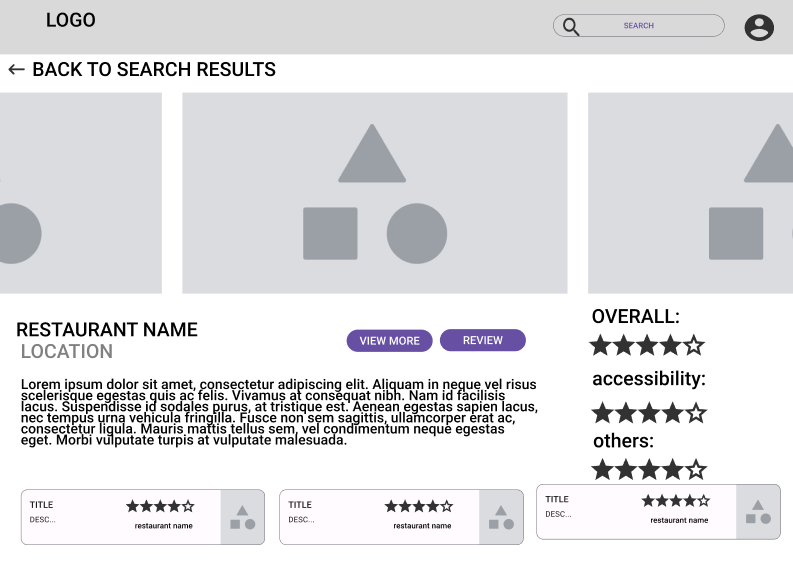
## Login Page:

Users have the option to use the Website as a Guest, login as an existing user or to sign up and create a new account.



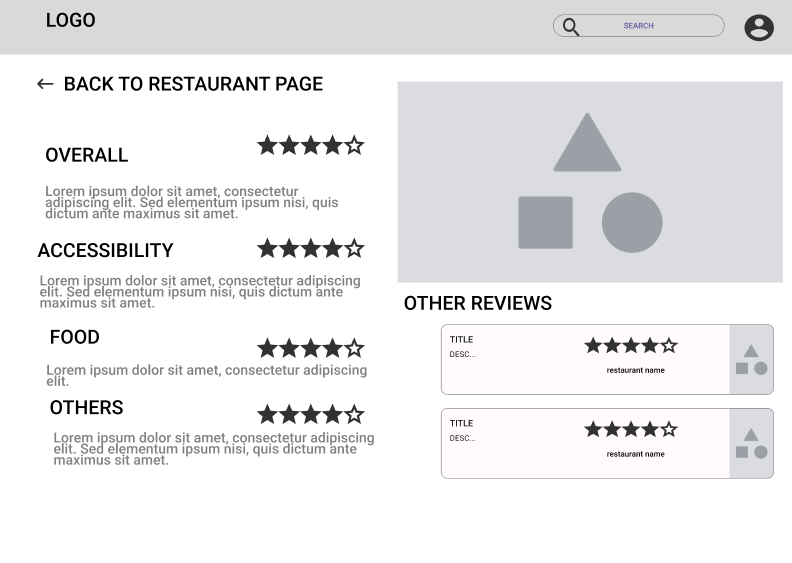
Home Page:

This page provides the user with 2 options: Filtered Search or Search. Users can conduct Basic Searches by inputting the eatery’s name (String Input), or narrow their searches by indicating their choices on the filterbar. Clicking on the view more button takes the user to the Restaurant page.

**

Listing Page:

This page provides users with a brief description of the restaurant, type of cuisine served, prices, ratings and reviews etc. Clicking on the “See All Reviews” link located below the eatery’s rating takes the user to the View Reviews Page. A “Leave Review” button located at the bottom of the page takes the user to the Leave Review page.



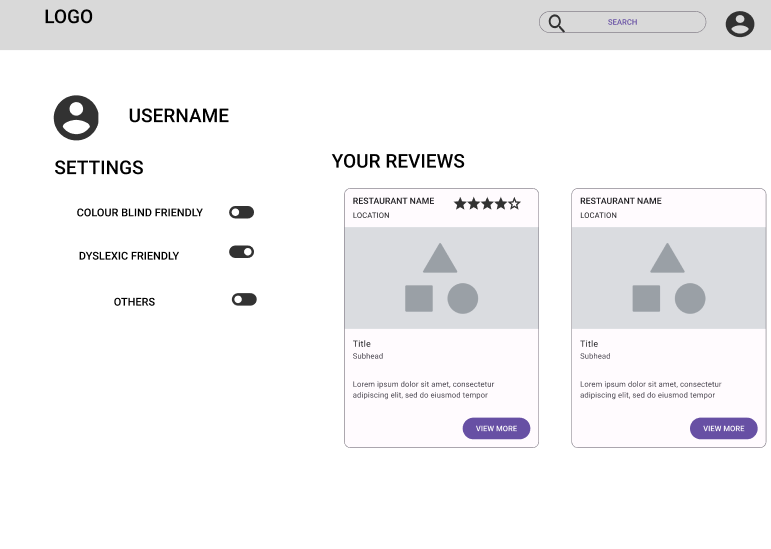
View Reviews Page:

This page allows the users to read other users’ reviews, but they are not able to rate nor write a review.

## 

Leave Reviews Page:

This page allows the users to submit their ratings and reviews for the eatery.



Profile Page:

This page allows the user to set their profile preferences and shows the user’s past reviews.

## Hardware **Interfaces**

* IOS Phone/ Android Phone
* Desktop/ Laptop

## Software Interfaces

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

## **Communications Interfaces**

Data Transfer: Google Places API

# **System Feature**s

## **System Feature 1**

## Login

| Description: | Authenticate the user |
| --- | --- |
| Postconditions: | 1. User is able to login. 2. User is unable to login. |
| Priority: | High |
| Frequency of Use: | High |
| Flow of Events: | 1. User inputs ‘USERNAME’ and ‘PASSWORD’. 2. User clicks the login button. 3. System interacts with the user database and authenticates the user. 4. User is logged in. |
| Alternative Flows: | AF1: User clicks on “Register Account” button   1. User is led to the Account Creation Page.   AF2: User clicks on “Login as guest” button   1. User is led to the Home Page.   AF1: If the password is wrong   1. User is shown ‘INVALID LOGIN’.   AF2: If the USERNAME is wrong   1. User is shown ‘INVALID LOGIN’. |
| Assumptions: | All Servers are up and running with no connection problems. |
| Functional Requirements: | |
| REQ-1: Users must be able to choose to create a new account, login as a registered user or as a guest  Registered users must be able to login.   * 1. Users must be able to sign out at any point in time.   2. System must be able to identify and prompt the user to re-enter their credentials if they entered it wrongly   REQ-2: Information Databases   1. Database management systems must be able to retrieve the information needed when the Website demands it.    1. The Database must be able to store the login credentials where passwords are encrypted when stored. 2. Database management systems must be able to modify databases when the Website has a new user registered to the database. | |

## Register

| Description: | Register user into user database. |
| --- | --- |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Flow of Events: | 1. User selects the ‘Register Account’ button. 2. User is prompted to enter ‘USERNAME’, ‘PASSWORD’ and ‘EMAIL’. 3. System checks whether the user is already registered. 4. System creates a new user. 5. User is prompted to login. |
| Alternative Flows: | AF: If user already exist   1. System informs the user ‘USER ALREADY EXISTS’ 2. User is prompted to enter another email address. |
| Functional Requirements: | |
| REQ-1: Users must be able to choose to login or use a guest account.  1.1 Guest users must be able to register.  1.2.1 Username constraints  1.2.1.1 Username must be unique  1.2.2 System must be able to store newly created accounts.  REQ-2: Information Databases  2.1 Database management systems must be able to retrieve the information needed when the website demands it.  2.1.1 Database must be able to store the login credentials.  2.2 Database management systems must be able to modify databases when the website has a new user to add to the database .  2.2.1 Database must be updated when a new user registered. | |

## Review and Rate

| Description: | Post reviews and rating for restaurant |
| --- | --- |
| Priority: | Medium |
| Frequency of Use: | Medium |
| Flow of Events: | 1. User inputs personal comments. 2. User inputs rating. 3. User selects ‘SUBMIT’. 4. System gets the restaurant name from the google map database 5. System stores reviews and rating to the review database with user name and restaurant as the key. 6. User is prompted that the review and rating has been successfully posted. 7. Use case ends. |
| Functional Requirements: | |
| REQ-1: Registered users have access to memory reliant features of the website.  1.1 Registered users must be able to leave a star rating on the restaurant in the system.  1.2 Registered users must be able to leave reviews on the pages of the restaurants displayed.  REQ-2: Website must display the ratings and reviews that other users made for the restaurant.  2.1 All reviews must be displayed with the registered user’s name and star rating of the restaurant.  REQ-3: Information Databases  3.1 Database management systems must be able to retrieve the information needed when the Website demands it.  3.1.1 Database must be able to store the reviews, ratings and comments made by users.  3.2 Database management systems must be able to modify databases when the Website has a new element to add to the database.  3.2.1 Database must be updated when a review or rating is left on the restaurant. | |

## Search

| Description: | Basic search for a specific restaurant |
| --- | --- |
| Priority: | High |
| Frequency of Use: | High |
| Flow of Events | 1. User types in the name of the restaurantto search for. 2. System interacts with the google API to retrieve a list of restuarant based on the input. 3. System displays a list of restaurants to the user. 4. User selects the ‘View More’ button. 5. System retrieves restaurant information from database and lead them to the restaurant’s page 6. System display the restaurant’s page. 7. User can read the description of the restaurants and the reviews by the other users. |
| Alternative Flows: | AF1: If no matching restaurant is found.   1. System shows a “No Restaurant is FOUND” message. 2. Back to searching   AF2: If the user is logged in.   1. User can post a review for the restaurant |
| Functional Requirements: | |
| REQ-1: Information on the restaurant must be retrieved from google API.   1. 1.1 Website must display the list of restaurant by ascending alphabetical order.    1. Website must allow users to scroll through the list manually.    2. Website must allow users to select a restaurant of their choice to view more information. 2. Restaurant must display the list of restaurants by ascending alphabetical order with respect to the filters the user has Websitelied.    1. Website must allow users to select a restaurant of their choice to view more information. 3. Website must display more information of whichever restaurant the user selected.    1. Website must allow users to scroll through the restaurant’s displayed information.    2. Website must display the address of the restaurant.    3. Website must display the user’s selected restaurant’s location on Google Maps.       1. Website must allow the users to drag and navigate around the map manually. 4. Website must display the ratings, review and comments that other users made for the restaurant.    1. All reviews must be displayed with the registered user’s name and star rating of the restaurant.   REQ-2: Guest must be able to access the basic non-memory reliant features of the application  2.1 Users must be able to perform searches for restaurant.  2.1.1 Users must be able to search by name.  2.1.2 Users must be able to search by filtering their criteria.  2.1.2.1 Users must be able to filter by location. (e.g. west, northeast, central)  2.1.2.2 Users must be able to filter by wheelchair accessibility  2.1.2.3 Users must be able to filter by rating. (0-5 stars)  REQ-3: Information Databases  3.1 Database management systems must be able to retrieve the information needed when the website demands it.  3.1.1 Database must be able to store the login credentials.  3.1.2 Database must be able to store the reviews, ratings and comments made by users.  3.1.3 Database must be able to store the reports made by the users. | |

# **Other Nonfunctional Requirements**

## **Performance Requirements**

1. The website should be able to produce result and display all required data within 3s of searching.
2. The website must be maintainable.
   1. The website must run smoothly and not crash frequently
   2. Modularisation of codes allows ease of rfeusing methods or functions in the program
3. The website must be functional 24 hours a day, 365 days a year.
4. The database must be scalable for high demand use

## **Safety Requirements**

1. The user’s credentials are safely stored in the user’s database to ensure security and privacy.
2. The user’s login credentials are hashed upon entry to ensure security and privacy.

## **Security Requirements**

1. If user enters his/her password wrong multiple times, his/her account will be locked.

## **Software Quality Attributes**

1. The website must be able to work on all browsers (google chrome, mobafire, microsoft edge).
2. The website must be able to work on various screen sizes.
3. The website must ensure all data received from the user is in the correct format to ensure data consistency in the database.
4. The website’s interface must be user friendly.
   1. The website prompts and guides the user when an invalid input is entered.
   2. The website must be mobile compatible, so users can access the website via their smartphones.
   3. The website must has an effective navigation, allow users to go where they want to be in a quick efficient manner.

## **Business Rules**

1. The registered user must be able to review and rate restaurants
2. The guest user must not be able to review and rate restaurants

# **Other Requirements**

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

**Appendix A: Glossary**

Data Dictionary

| Term | Definition |
| --- | --- |
| User | The user of the app. There are two type of user:   1. User that had an registered account on our app 2. Guest user that use the app without signing in |
| Username | A name user used to log in to their registered account |
| User password | A password user used to log in to their registered account |
| Restaurant Name | Name of the restaurant |
| Restaurant Address | Address of the restaurant |
| Restaurant description | A short history description of the restaurant and information about their price and phone number. |
| Restaurant accessibility | State whether the restaurant is accessible by wheelchair. |
| Rating | A rating system for a user to rate for the quality of the restaurant and the overall service and management of their experiences with the restaurant. |
| Review | A review system for a user to view or rate for the quality of the restaurant and the overall service, food and ambience. |

**Appendix B: Analysis Models**

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

**Appendix C:** User Manual

| User Manual | |
| --- | --- |
| Screens | User’s Action |
|  | 1)  User enter “Default” page at FoodFinder in web browser  User can select ‘Sign up’ button, ‘Proceed as guest’ button, ‘Login’ button.  Click ‘Sign Up’ button at ‘Default’ page to navigate to ‘Sign up’ page if the user does not have an account and wants to create an account.  Click ‘Guest User’ button to navigate to ‘Home’ page if user wants to access the app.  2)  User enters Username and Password,  Click ‘Login’ button at ‘Default’ page to navigate to ‘Home’ page if the user has an account.  Click ‘Sign Up’ button at ‘Default’ page to navigate to ‘Sign up’ page if the user does not have an account and wants to create an account.  Click ‘Guest User’ button to navigate to ‘Home’ page if user wants to access the app. |
|  | 3)  User enters ‘Home’ page.  User enter name of restaurant at the ‘Search bar’ to search.  User can select options on the ‘Filter bar’ to do a filtered search.  User can click on desired Restaurant to navigate to ‘Restaurant’ page.  User can click on the ‘Profile’ button on the top right to navigate to ‘Profile’ page. |
|  | 4)  User enter ‘Profile’ page.  User can click on their past reviews.  User can change their profile settings according to their disability needs. |
|  | 5)  User enters ‘Restaurant’ page.  User can select ‘View more’ to navigate to ‘View Review’ page.  User can select ‘Review’ to navigate to ‘Leave Review’ page. |
|  | 6) User enters ‘View Review’ page.  User can click on past reviews and view the contents. |
|  | 7)  User enters ‘Leave Review’ page.  User can ‘Submit’ a review and rate the restaurant. |

Source: http://www.frontiernet.net/~kwiegers/process\_assets/srs\_template.doc