Software Requirements Specification

for

Mat3amk

Team G23

Faculty Of Engineering Ain Shams University, Cairo
Software Engineering

Contents

Revision History **1**

1 Introduction **2**

1.1 Executive Summary . . . 2

1.2 Document Overview . . . 2

1.3 Abbreviations and Terminologies . . . 3

1.4 References . . . 3

2 System Description **4**

2.1 Introduction . . . 4

3 System Modules, Environment **6**

4 System Models **8**

5 System Features **10**

5.1 Search up nearby Restaurants . . . 10

5.2 Getting Food Menus . . . 10

5.3 Adding Comments . . . 10

5.4 Chat Rooms . . . 11

5.5 Barcode Scanning . . . 11

5.6 Text Extraction . . . 11

5.7 Multi Language Support . . . 12

6 Other Nonfunctional Requirements **13**

6.1 Product requirements . . . 13

6.1.1 Portability requirements . . . 13

6.1.2 Reliability requirements . . . 14

6.1.3 Usability requirements . . . 14

6.1.4 Efficiency requirements . . . 14

6.2 Organizational requirements . . . 14

6.2.1 Delivery requirements 14
6.2.2 Implementation requirements 14
6.2.3 Standards requirements 14
6.3 External requirements 14
6.3.1 Ethical requirements 14
6.3.2 Interoperability requirements 15
6.3.3 Legislative requirements 15

7 System Interfaces **16**
7.1 User Interface 16

Appendices **19**

A REFERENCES **20**

B PROCESS EVALUATION **21**

Revision History

Revision	Date	Author(s)	Description
1.0	5.12.2018	Mohamed Rashad	Chapter 1 - Introduction
2.0	9.12.2018	Ahmed Ibrahim	Overview
3.0	16.12.2018	Mohamed Rashad	Chapter 7 - System Interfaces
4.0			

Chapter 1

Introduction

1.1 Executive Summary

Mobile applications can be one of the best ways to keep consumers engaged with a brand as they are on the move. With the increase in demand for smart phones and efficiency of wireless networks, the demand for mobile applications has increased incredibly. Android is one of the most popular open source platforms that offers the developer's full access to the framework API's so as to build innovative applications. The main aim of this project is to build an Android application that helps the users to find a Restaurant in a specified location and according to the specified tastes. The main features provided by the Your Restaurant application are as follows:

- Basic Search where the user can search for a particular restaurant based on any keyword and Advanced Search where the user can specify the category, rating and the distance range for the restaurants.
- Google Maps that shows the top 5 restaurants in the city of the current location and the routes to a particular restaurant.
- The users can write a review, see the reviews and invite a friend/colleague to meet at a particular restaurant.
- Google Calendar where the user can mark an event.

1.2 Document Overview

Software Requirements Specification document is designed to document and describe the agreement between the customer and the developer regarding the specification of the software product requested. Its primary purpose is to provide a clear and descriptive "statement of user requirements" that can be used as a reference in further development of the software system. This document is broken into a number of sections used to logically separate the software requirements into easily referenced parts. This SRS document aims to describe the Functionality, External Interfaces, Attributes and Design Constraints imposed on Implementation of the software system described throughout the rest of the document. Throughout the description of the software system, the languages and terminology used should unambiguous and consistent throughout the document. This SRS document describes the software requirements specifications for Your Restaurant which is an Android application that helps the users to find a Restaurant in a specified location and according to the specified tastes. This document includes a description of the software and its subsystems.

1.3 Abbreviations and Terminologies

UI	User interface (UI) design is the process of making interfaces in software or computerized devices with a focus on looks or style. Designers aim to create designs users will find easy to use and pleasurable. UI design typically refers to graphical user interfaces but also includes others, such as voice-controlled ones.
ADB	Short for Android Debug Bridge Software that bridges the gap between your Android device and a computer, allowing you to send high-level commands to your phone or tablet over a USB data cable.
API	Short for Application Programming Interface. APIs are functions that developers can call on to access specific features by calling upon programs, code,and services that others have written.
AOSP	Short for Android Open Source Project The base of Android as a whole,which is used by manufacturers and independent developers to create the firmware an Android device runs on. Used colloquially to refer to an unmodified version of Android in some cases
APK	Short for Android application package The extension used in Android app installation files (e.g., app.apk). Similar in nature to an EXE file on Windows.
CDMA	Short for code division multiple access A mobile voice and data communications standard used by cellular carriers such as Sprint and Verizon. A competing standard to GSM.
SDK	SDK stands for Software Development Kit, which is a programming package that enables developers to create apps for a particular software platform or framework.
JSON	JSON stands for JavaScript Object Notation. It is a lightweight data interchange format that is easy for humans to read and write and for machines to parse and generate.

1.4 References

- android.gadgethacks.com

Chapter 2

System Description

2.1 Introduction

Starting from a very high level of perspective, our application is a service-in-need provider. Imagine all of your favorite restaurants in one single place, always along with you in between your hands in almost every day-to-day life. With only picking up your smart phone and opening up our application ‘’your restaurant” you are good to go! Just pick the meal you would love to have and select the place you would love to visit and move right on.
The easiest it could be to fulfill your utter pleasure which no one could say else. Fig. 2.1

Figure 2.1

Our system in general, Our system is a self-explanatory system that helps users get a list of all possible restaurants available in various districts of Egypt.

Digging deeper into our system description, here are the low-level perspective of our application in details ...

User interactive, We have decided to make this application a mobile application as a reason for knowing the fact that almost everyone of us nowadays holds a mobile device any place any time along the road. So, we are providing multiple activities in which the user could pick data, insert and rate reviews info.

So, just by a simple click the user gets:

- restaurants customers ratings and reviews about the serving quality in a neat UI provided by our profi-

cient designers.

- the route to the nearest restaurant via a map by allocating the local position of the user as input. We do this by specifying a so-called implicit intent to retrieve the targeted data from the Google GPS application 'Maps – Navigate & Explore.

Confidentiality, We understand the power of collaboration and we trust and consider our human resources, so our system provides a way to express the standards which are in a great deal affect in various decisions, from this point we have come up with one of the concrete features of our application which is reviewing and rating.

Revision, our system is from users to users. We make sure that everything is safely considered in our app which helps share our thoughts and opinions to give them to all who are in such need.

Chapter 3

System Modules, Environment

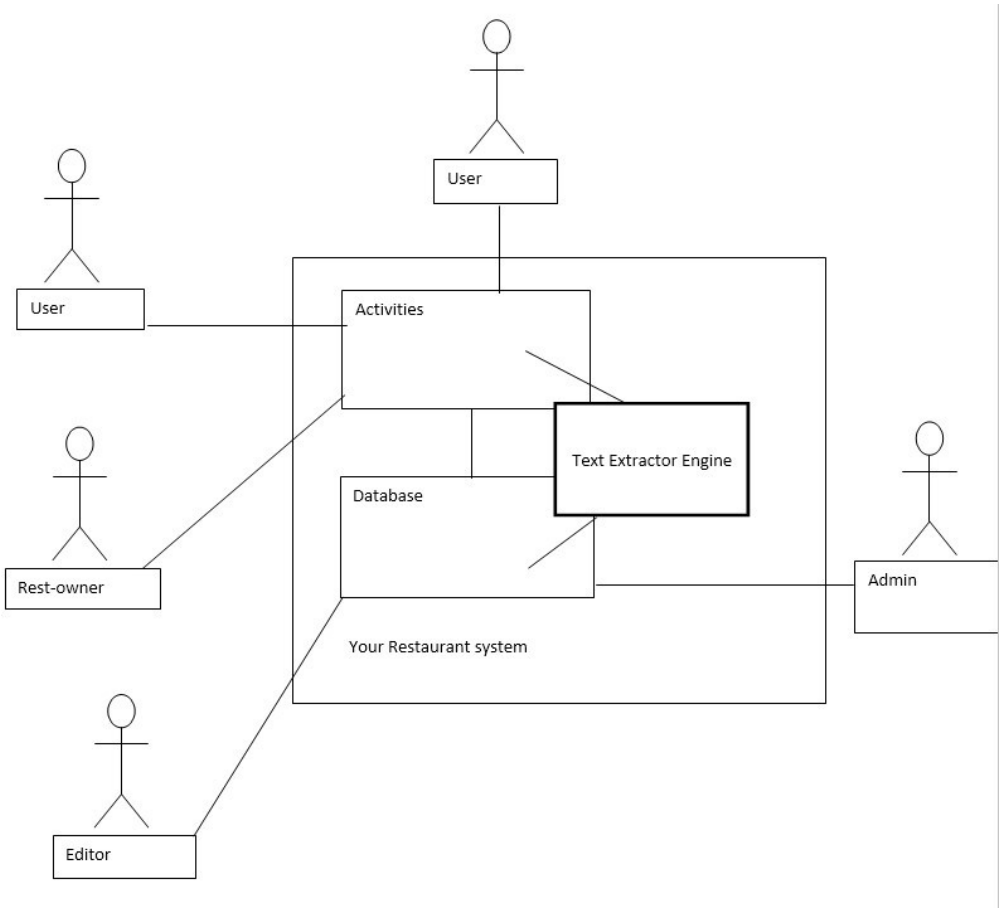


Figure 3.1: System Modules

Activity Module

This module provides the functionality for customers to display the local restaurant names and supply necessary details.

Users of the system, namely restaurant customers, may be provided the following functionality:

- Create an account.
- Manage their account.
- Log in to the system.

Users of the system, namely restaurant customers, must be provided the following functionality:

- Navigate the restaurant's menu.
- Select an item from the menu.
- Review their current restaurant.
- Remove an item/remove all restaurant referenced items.
- View the restaurant selected location.

Data base Module

This module provides functionality for the power user-Administrator only. It will not be available to any other users of the system Customers. Using a graphical interface, it will allow an Admin to manage the menus that is displayed to users.

- Add/update/delete food category to/from the menu.
- Add /update/delete food item to/from the menu.
- Update price for a given food item.
- Update additional information (description, photo, etc.) for a given food item.
- View the restaurant selected location.

Before customers can actually use this system, functionality provided by this component will have to be configured first.

Once the initial configuration is done, this will be the least likely used component as menu updates are mostly seasonal and do not occur frequently.

Text extractor engine Module

Users (restaurant owners) can use this Machine Learning based engine to extract data from menus and sent it to our servers for processing without the need to write it by hand which allows our users to have more accessibility and ease of use in using our app.

Chapter 4

System Models

Diagram Search.jpeg

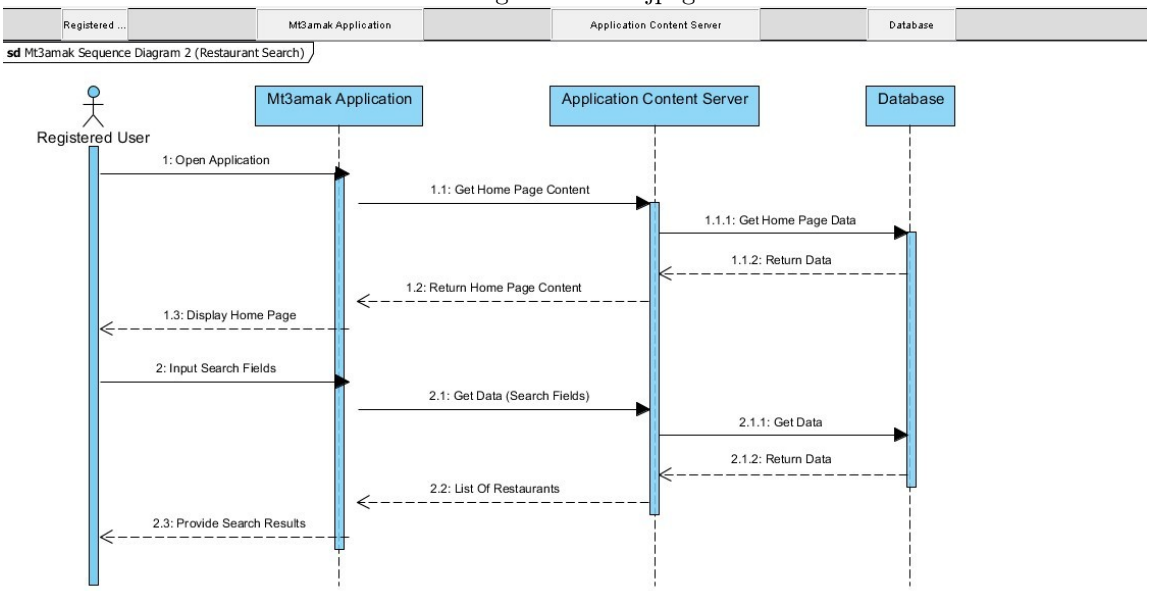


Figure 4.1: Restaurant Search

Diagram Login.jpeg

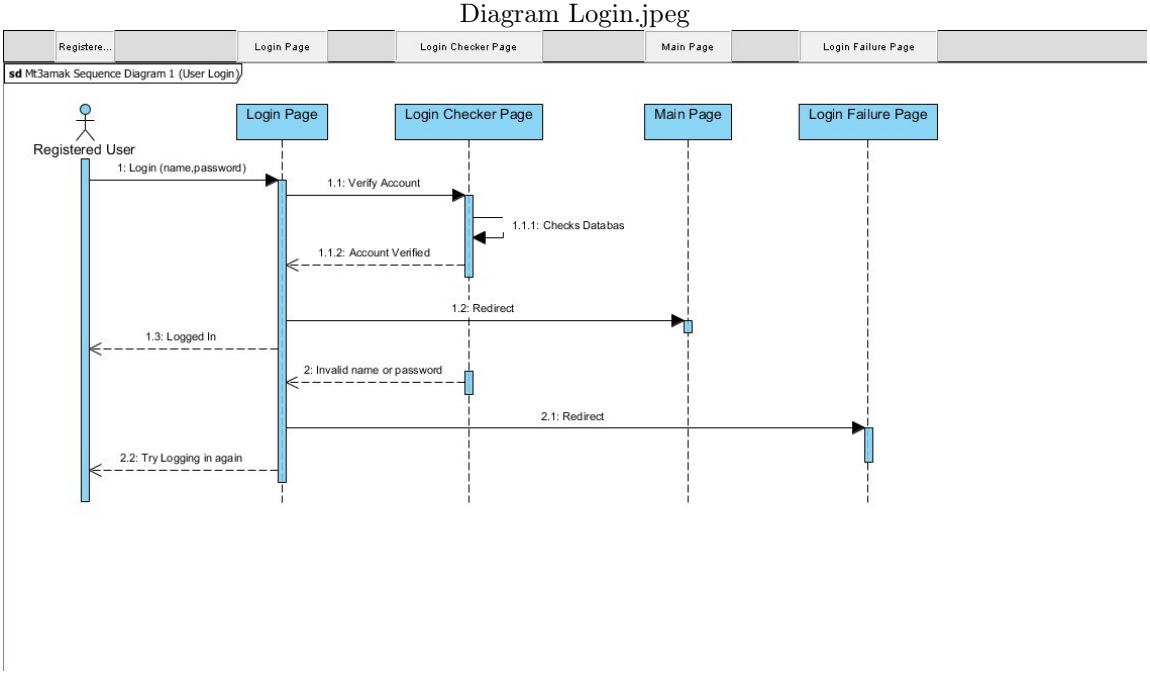


Figure 4.2: User Log-in

Diagram.jpeg

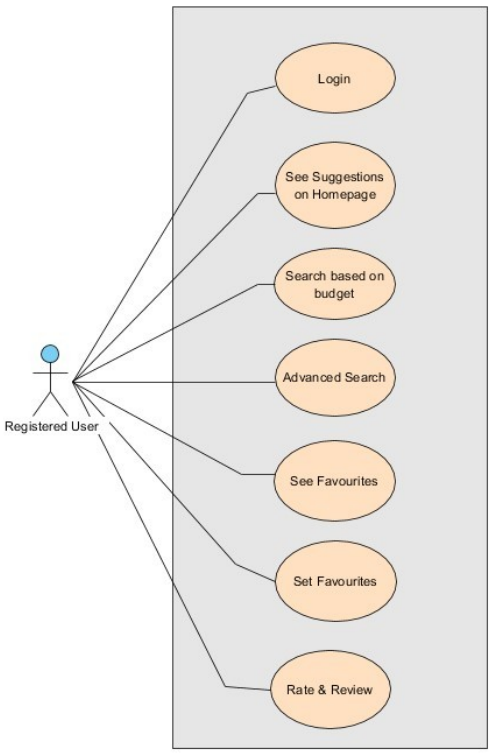


Figure 4.3: Use Case Diagram

Chapter 5

System Features

5.1 Search up nearby Restaurants

- Description: User can check out nearby restaurants based on his / her location. User can search up a specific food item.
- Inputs: they input either partial or full name of the food. For example, you can just type "Pizza" and search for all Pizza items from all restaurants in the current location or in every location possible.
- Outputs: The search result will contain a list of different restaurants.
- Pre-conditions: internet connection ,GPS permission.
- Post-conditions: internet connection.

5.2 Getting Food Menus

- Description: When a user launches the application, a list of restaurants within the current location will be displayed in the phone. Further detailed information of a restaurant will be displayed as the user selects a restaurant from the list.
- Inputs: select a restaurant from the list.
- Outputs: the user can get the entire menu of the selected restaurant. Each food contains detail information which helps user to have better idea so that they can order the food from the phone.
- Pre-conditions: internet connection ,GPS permission.
- Post-conditions: internet connection.

5.3 Adding Comments

- Description: With the picture, rating and comment features available on the phone, this phone application is better suited for users to give instant feedback about the food items.
- Inputs: select a food item or restaurant.
- Outputs Outputs: Users can read and write any comments for food items in the system making an overall better experience for every user.

- Pre-conditions: internet connection, Firebase, ML Kit .
- Post-conditions: internet connection.

5.4 Chat Rooms

- Description: Creating chat rooms allows the user to interact with live sessions with the restaurant giving him / her more experience about the food menu and items, also the user can take a picture with the phone and upload it to our server
- Inputs: users select a restaurant and its food category.
- Outputs: they can take and upload pictures to the server. Allowing users to have this feature can motivate users to use our food application more frequently.
- Pre-conditions: internet connection, Firebase, ML Kit .
- Post-conditions: internet connection .

5.5 Barcode Scanning

- Description: Scan for all supported Barcode formats at once, without having to specify the format you're looking for. Or, boost scanning speed by restricting the detector to only the formats you're interested in.
- Inputs: Input images must contain Barcodes that are represented by sufficient pixel data. In general, the smallest meaningful unit of the Barcode should be at least 2 pixels wide (and for 2-dimensional codes, 2 pixels tall).
- Outputs: Structured data stored using one of the supported 2D formats are automatically parsed. Supported information types include URLs, contact information, calendar events, email addresses, phone numbers, SMS message prompts, ISBNs, WIFI connection information, geographic location, and AAMVA standard driver information.
- Pre-conditions: Authentication from the camera, Firebase, ML Kit.
- Post-conditions: Structured data for the Database and support for on sale discounts.

5.6 Text Extraction

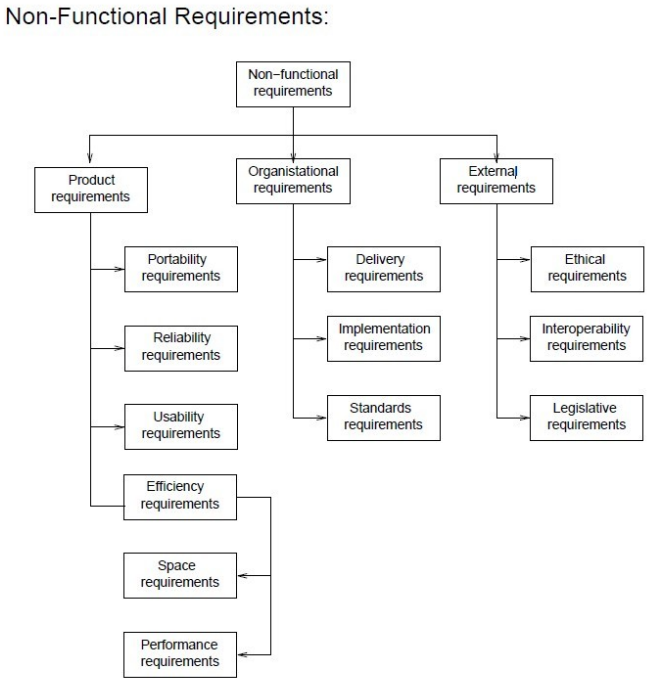
- Description: Recognize text in any Latin-based language to automate tedious data entry for credit cards, Menus, and business cards.
- Inputs: input images must contain text that is represented by sufficient pixel data. Ideally, for Latin text, each character should be at least 16x16 pixels. For Chinese, Japanese, and Korean text, each character should be 24x24 pixels.
- Outputs: Text Parsed and Formatted in Block, Paragraph, Word, and Symbol way for accessing or storing in the database.
- Pre-conditions: Authentication from the camera, Firebase, ML Kit
- Post-conditions: Text Extraction in Real Time.

5.7 Multi Language Support

- Description: This application will support two languages (Arabic & English) for a better user interface
- Inputs: choose the language
- Outputs:Users will be able to search and write with the available languages so it can be more convenient for them.
- Pre-conditions: Authentication from the camera, Firebase, ML Kit
- Post-conditions: Text Extraction in Real Time.

Chapter 6

Other Nonfunctional Requirements



Functional Requirements.jpeg

Figure 6.1

6.1 Product requirements

6.1.1 Portability requirements

Our platform will be available as an android application which can be installed with one click through the app store

6.1.2 Reliability requirements

The application should be very reliable for our users. The data base must be filled with all the top restaurant franchises. The team will also add some popular local restaurants. Restaurant owners can also have an account on our system to add their own businesses.

6.1.3 Usability requirements

The UI of the application will be simple but elegant. It provides easy navigation and rich content and functions for the users.

6.1.4 Efficiency requirements

Space requirements

To increase portability, the entire data base will be stored remotely on a server. However, the application might store local cache on the user device to allow easy accessibility if there is no internet access.

Performance requirements

The application should not be hardware intensive. It mainly depends on the user internet connection speed. Some features might require some processing power like text recognition.

6.2 Organizational requirements

6.2.1 Delivery requirements

The source code, data base and PSDs used in the app should be delivered to the project manager. After its validation, it can be published online on the app store.

6.2.2 Implementation requirements

Git version control system should be used by the developers. Only team leaders are responsible for merging branch conflicts if they occur.

6.2.3 Standards requirements

The application must be well documented stating the role of each developer in the development process.

6.3 External requirements

6.3.1 Ethical requirements

- Both male and female developers must be treated equally.
- Private customer data should not be shared publicly.
- All user comments on our application should be monitored.

6.3.2 Interoperability requirements

- Photoshop designs must be convertible to XML.
- Servers and the application should be seamlessly integrated. Delay between data entry and data base modification must be minimized.

6.3.3 Legislative requirements

Publishing any restaurant related content on the application should be accepted by the business owners. Business owners that register through the application should visit our office and provide the official government license and permit of their restaurant.

Chapter 7

System Interfaces

7.1 User Interface

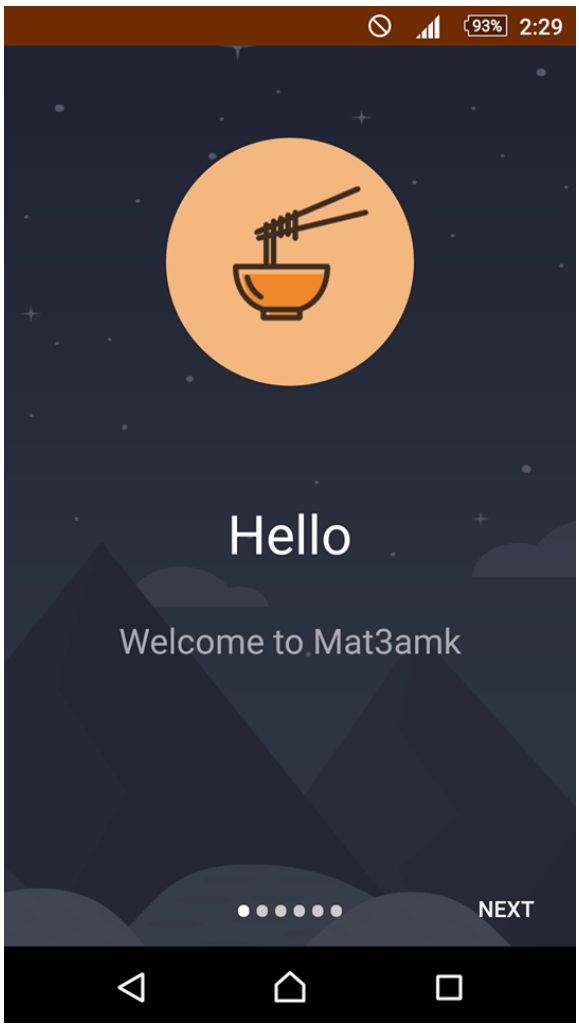


Figure 7.1: Welcome to Mat3amk

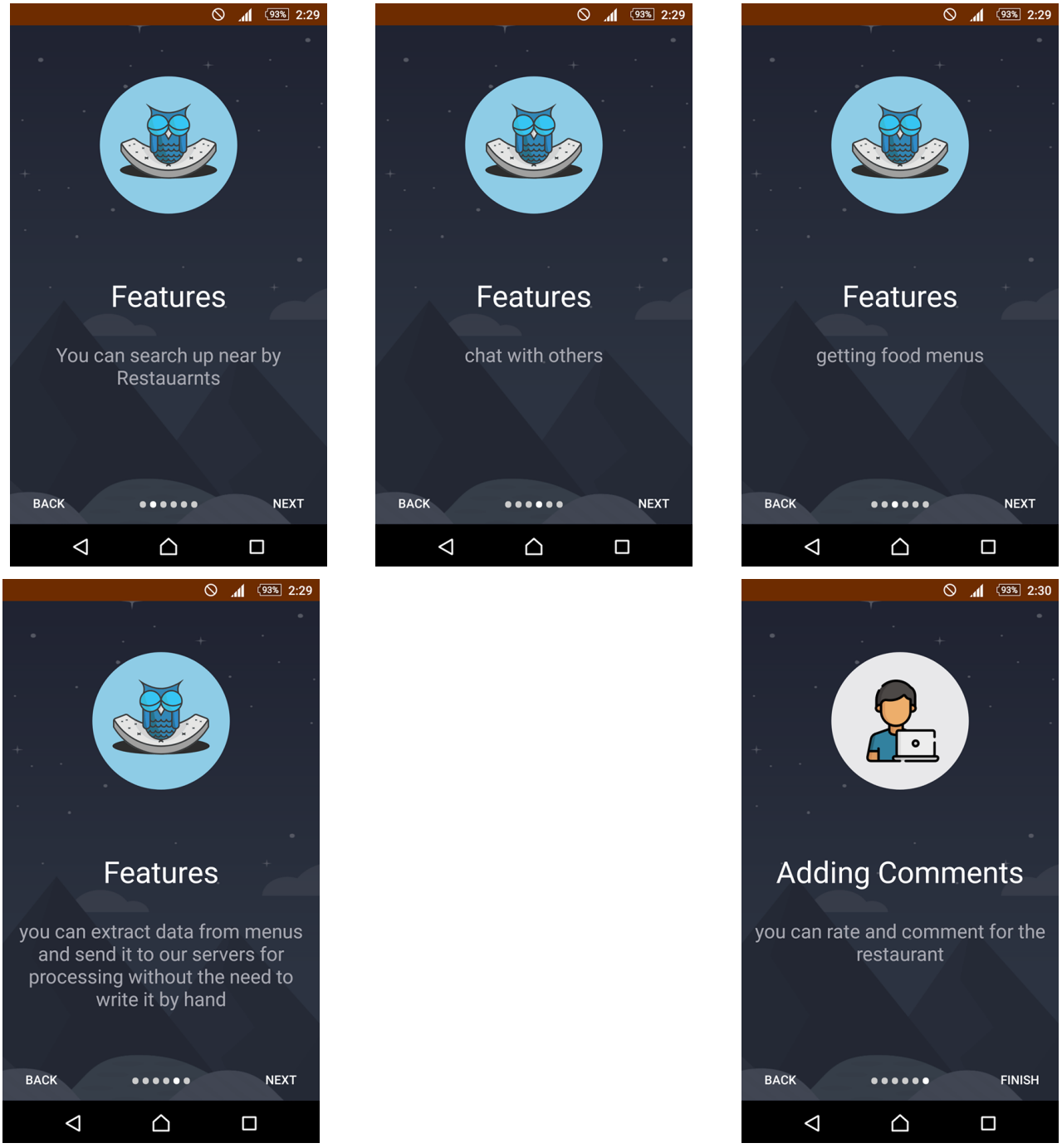


Figure 7.2: Introduction Interface

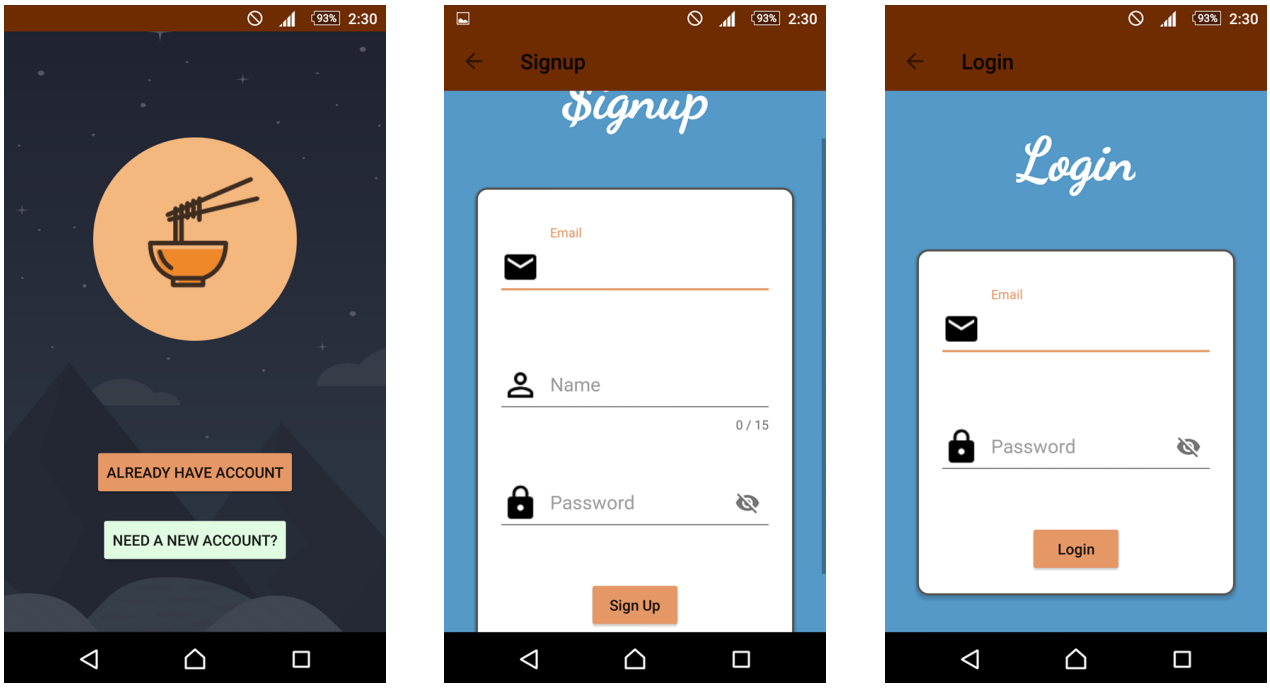


Figure 7.3: Sign-Up & Log-in Interface

Appendices

Appendix A

REFERENCES

C. Larman, APPLYING UML AND PATTERNS An Introduction to Object-Oriented Analysis and Design and Iterative Development, 3rd ed., Massachusetts: Pearson Education, 2005. D. Carrington, CSSE3002 Course Notes, School of ITEE University of Queensland, 2008 IEEE Recommended Practice for Software Requirements Specifications, IEEE Standard 830, 1998.

Appendix B

PROCESS EVALUATION

In order to improve an engineering process, it must be recorded and analysed. This appendix seeks to collate time data based on both the planned schedule and the actual time spent during the process. Furthermore, it discusses the problems encountered during the process of developing the subject SRS and suggests remedies to prevent these problems from arising in future iterations.

Development Process

The team's original plan for developing the Software Requirements Specification (SRS) was straightforward. Each team member was to review lecture notes and related textbook chapters to ensure familiarity with the process (background research). They were to also research existing computerised restaurant menu/ordering systems to gather an idea of what already currently exists and what the requirements of the system would be (requirements elicitation). The list of requirements gathered from this were then to be refined in a meeting with all team members, which would result in a consistent and complete set of requirements defining the system. This would include a use case model, use case descriptions, sequence diagrams, activity diagrams, UML class analysis diagram and state chart diagrams. From this set of requirements, the final SRS was to be developed written as a formal SRS as per the IEEE template. The requirements gathered in previous phases were collated and each section was to be written in turn making sure that later sections were consistent with previously written sections and then reviewed by all team members to check for any errors or inconsistencies. The completed SRS once agreed to by all team members was then to be submitted.

Problems Encountered

The most serious problem encountered was that of finding the time in which everyone could meet outside of university, work hours and other commitments. With all of the team members having busy timetables, as well as other commitments, we were limited in the times that we were able to meet and therefore had to spend a lot of time working via email. Second was the inaccurate estimation of time required to complete the writing of the SRS. The time required for requirements elicitation was underestimated, delaying the schedule and leaving less time available for diagram production and report writing. This miscalculation occurred because the team had no previous metrics by which to estimate the time needed to complete the requirements elicitation phase.

Suggested Process Changes

The primary change to the SRS development process that this report suggests pertains to time estimation. It is not ideal to plan the development of a SRS with no previous metrics on which to base the time estimations. It is recommended that a longer window of time be allocated to requirements elicitation, diagram production and the writing of the final report.

Time Estimation/Expenditure Comparison

The differences between the submitted SRS plan and the actual process undertaken have been summarised below. a) Preparation of SRS Plan b) Proof reading of the SRS Plan c) Research d) 'Requirements Elicitation' meeting e) Complete analysis/use case modelling f) Complete sequence, activity, UML class analysis, and statechart diagrams g) Review requirements h) Complete first draft i) Proof read first draft j) Complete final report draft