Software Requirements Specification

for

Smart Dine-in System

Version 1.0 approved

Prepared by Mayank Jain & Mohit Kalra

The NorthCap University

11 January 2019

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

To automate the current Dine-in Systems in restaurants and make it more effective, feasible and improve the services offered in a cheaper way.

## Document Conventions

Font size: 11

Font face: Arial

Important points have been highlighted in **bold**.

## Intended Audience and Reading Suggestions

The intended audience is the food market and outlets which have dine in and al a carte services available. This will help in cost cutting and technological advancement. The overview of the SRS is mentioned as the Table of Contents.

## Product Scope

The goal of the smart dine-in software is to incorporate current technological advancements in the restaurants. This will help in cost cutting, smart interfaces and faster, efficient services.

# Overall Description

## Product Perspective

The proposed system is an improvement over the current dine-in systems being used and can come up as a replacement in the near future. It involves all the aspects of dining like placing the order, delivering the order and creating the final bill of the customer.

## Product Functions

The major functions of the product include placing the order using an Android application installed on the user’s smartphone, using channels and mechanics to deliver the order to the customer’s table using the shortest available path for the bot and then generating a computerized bill as per the order given by the user. The main technologies used are Android, Internet of Things, electronics and mechanics.

## User Classes and Characteristics

The smart dining system can be deployed on all kinds of food outlets. Its application will be unique to each outlet depending on the number of tables, layout of the restaurant, area occupied, size of tables and the frequency of the customers which visit each day.

But, this system will be most beneficial for large restaurants so as to reduce the number of waiters and hence reduce labor cost and increase profits.

## Operating Environment

The software will be deployed on Android platforms which include smartphones, tablets etc. supporting versions 4.0 and above.

This ensures that most of the devices will be covered, excluding only really old devices.

The deployment of hardware will be done using channels and mechanics which will depend on the size of the restaurant and layout of tables.

RFID will be used by the restaurant, which will be scanned by the customer so as to let the restaurant know the table number the customer is sitting at.

## Design and Implementation Constraints

The hardware implementation will depend on:

1. Size of the food outlet.

2. Layout of the outlet.

3. Number of tables.

4. Size of each table.

5. Area of ceiling of restaurant.

6. Number of orders at a particular time.

7. Number of occupied and unoccupied tables.

The more the size of the restaurant, the more difficult it will be to create a grid for the smart delivery system to work.

The software implementation will depend on the Android version of the customer’s device, whether it can support the application or not.

# External Interface Requirements

## User Interfaces

The user will interact with the system using an Android application which will have to be installed on the customer’s device. This will enable him/her to identify the table number, place order, give payment using different methods and leave a review regarding the service offered by the restaurant. It will be a graphical and interactive user interface supported by a secure login so as to preserve user information for future occurrences.

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## 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.>

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# 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

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

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: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>