Software Requirement Specification

Software Engineering BCS 6D

Restaurant Management System

Prepared By:
Muhammad Ahmad 18L-0965
Zulfiqar Chaudhry 18L-1037
Muhammad Daud Mazhar 18L-0919

FAST NU, Lahore

April 30th, 2021

Table of Contents:

1. Introduction:

1.1 Purpose:

The purpose of this project is to streamline the working of a restaurant by facilitating communication between the waiting staff and kitchen staff. Through this, the project aims to make restaurants more efficient so they can serve a larger number of patrons with ease.

1.2 Document Conventions:

There are no special conventions used in this document. The entire script of the document is written in plain English. The standard format for Software Requirements Specification is used as available at www.processimpact.com. The bold items inside this document are mainly highlighted so that the exact wording for those particular items are used when developing the application UI buttons. The priority for each feature is also specified out of a scale of 1-9 with 1 being the lowest priority.

1.3 Intended Audience and Reading Suggestions:

This document is intended for the team that is responsible to develop the software. The software requirements as prescribed by the client are documented in order to give the programming team a guide for development. The overall product descriptions and purposes are mainly intended for the project managers in Chapter 2 so that they have an idea of what the outcome and the final product should look like.

Testers should specifically pay attention to Chapters 3-5 and test the program accordingly. The programmers should also pay special attention to these chapters as they contain vital functional requirements to keep in mind before developing the application.

1.4 Project Scope:

The project would be a mobile phone application made to enhance productivity of a restaurant. The objective of the application would be to save time by providing a hassle free way to communicate orders from the waiting staff to the kitchen and vice versa. Consequently, the goal of the project would be fulfilled as the restaurant would be able to serve more people in a single day and earn more. In addition, the speedy service paired with modern technology is sure to awe the patrons and hook in good ratings for the restaurant on social media.

2. Overall Description:

2.1 Product Perspective:

The project would provide a replacement for the current system used by waiting staff and kitchen staff to communicate orders. In the current system, the waiting staff has to go back and forth from the customers to the kitchen to inform the kitchen of orders and check if any orders have been prepared. However, our project aims to eradicate this moving back and forth by facilitating communication through the use of our mobile phone application. As a result, the waiting staff would be able to save a lot of time and effort.

2.2 Product Features:

The major features of the project are summarized below:

Allowing Incoming Customers to Choose Tables: Incoming customers would be shown a view of the vacant tables by the welcomer from which they can choose the table they want to sit at.

Automatic Employee Salary Calculation: The software would automatically calculate the employees' salary, based on their working hours, and show it to the manager to add bonuses to it and approve it.

Managing Dirty Tables: The app would assign dirty tables to free busboys to clean. After they have cleaned the table, the table would be added to the vacant table list of the welcomer.

Managing Customer Orders: The application would provide an interface to the waiters to send an order to the chefs through the application. After the chefs are done preparing the order they can inform the waiters, through the application too, to come pick it from the kitchen.

2.3 User Classes and Characteristics:

There are 5 different user classes based on the subset of product functions and privilege levels:

Manager: The manager would be able to monitor the flow of data going through the application and how much work everyone is putting in. He/She would also have the power to dispatch the salaries of the employees by verifying the value calculated by the app based on the employee's work.

Welcomer/Greeter: The greeter would be able to see all vacant tables in the application and would allow the new customers to choose one table from them.

Waiter: The waiter would be able to take customer orders through the app and push them to the kitchen staff. Secondly, he/she would also be notified when an order pushed by them is prepared.

Busboy: The busboy would see dirty tables on the app and then declare them clean after he has cleaned them.

Chef: The Chefs would receive the order from specific waiters on the app and they will ping that waiter to pick the order up when it is prepared.

Each user except for the Manager would receive their salary slip on the application after it has been approved by the Manager. Lastly, the favoured user classes are manager, waiting staff and chefs while the others are less favoured.

2.4 Operating Environment:

The application would be installed on android tablets given to the staff. Furthermore, it would be linked with the database server of the restaurant which keeps all customer data.

2.5 Design and Implementation Constraints:

Implementation Constraints: The only implementation constraint would be linking the application with the database server already being used by the restaurant to store customer data.

Design Constraints: The restaurant shouldn't have to waste many resources in teaching its staff to use the application. So, the application GUI should be intuitive, expressive and simple to use.

2.6 User documentation:

After logging into the application for the first time, each type of user would be greeted with a unique interactive tutorial to teach them how to use their portion of the application. Furthermore, a video would be provided to the manager to show to the employees which would explain how each user type can interact with the other via the software.

2.7 Assumptions and Dependencies:

We assume that the tablets provided to the employees by the owner are powerful enough to run the app fluently.

3. System Features:

Below are the major system features provided by the application. Each of these functional requirements is given a priority indicated by the numbers 1-9, from 1 being the lowest priority to 9 being the highest priority.

3.1 Allowing Incoming Customers to Choose Vacant Tables

3.1.1 Description & Priority:

This feature will allow all the incoming customers to view and choose from all the vacant tables in the restaurant. It is one of the key features of the application, which is assigned **priority 8**.

3.1.2. Stimulus/Response Sequences:

As soon as a customer enters the restaurant, the welcomer will hand over a smart device (preferably tablets) to the customer with the screen that shows the vacant tables. The customer will then choose a particular table(s) of his choice and proceed. The welcomer will then lead them to the table selected, and then the menu task shall begin. In case no vacant tables are available, the customers will have an option to wait in the waiting lounge if they wish.

3.1.3 Functional Requirements:

REQ-1: The customer will choose the number of people that need to be seated from a minimum of count button displayed at the bottom of the screen.

REQ-2: Based on the number of people selected, the application will provide a graphical chart of available tables (colored blue) and allow the customer to select a table. If needed, more than one table can be merged for this particular customer.

REQ-3: Once selected, the table will be marked occupied (colored red). Each selected table or group of merged tables will be assigned a unique customer identification number, which will be later on used to track order billing and tracking.

REQ-4: Once all of this is done, the menu interface will be shown.

REQ-5: In case there are no vacant tables to suffice the customer, the customer can be added to the queue of waiting customers if the customers choose to wait.

REQ-6: As soon as a table gets vacant (colored blue), the app will notify the welcomer so that the first waiting customer group can be entertained. This particular customer group will be removed from the waiting queue if they are allotted a table.

3.2 Automatic Employee Salary Calculation:

3.2.1 Description & Priority:

This feature will allow the manager to monitor salary calculation for each employee. At the time of hiring, each employee will be assigned a unique Employee Number and the first working day will be stored for that month. At the end of each month, the employee salary will be calculated based on working hours and wage rate. Any extra bonuses will be included too. This is one of the most important tasks that will be handled by the application so it is given the highest **priority 9**.

3.2.2 Stimulus/Response Sequence:

Manager logins and clicks on the Calculate Salary Button shown on his interface. For each employee, the total salary for that month up to current date will be calculated. Manager will click on the Salary Paid button after he pays the employee his dues. This will reset the salary amount and put the current date as the start of salary calculation for the next month. It will also notify the restaurant owner about this salary payment, and a salary slip will be generated for that particular employee.

3.2.3 Functional Requirements:

REQ-1: Display the list of all employee categories in hierarchical order, starting from the manager to the welcomer (top to bottom order). Each category will then enlist all the employees who fall in that category.

REQ-2: The application will retrieve the data for each employee’s working hours from the salary start date of that particular month. It will display the salary due for the employee.

REQ-3: Additional bonuses can be added for each employee as determined by the manager. It could include ‘employee of the month benefits’ etc. which is completely under the control of the manager.

REQ-4: Once the salary is finalized, the manager can hit the Salary Paid button. This will reset the salary for the employee and start date for that month. Also, this button will generate a salary slip that can be viewed by that particular employee only, and notify the owner regarding the details of this transaction.

REQ-5: IMPORTANT: The manager will have the privilege to calculate salaries for all employees below him hierarchically. The manager’s salary will be calculated and dispatched by the **restaurant owner only**.

REQ-6: The app will also be able to track the time each employee has checked in and checked out on each working. It will be able to track Employee absences as well. This will be recorded in the system database for salary calculation each month.

3.3 Managing Customer Orders:

3.3.1 Description & Priority:

This is the main working feature of the application. This part will handle all the tasks from customer menu selection and preparation of order to the deliverance of order at the customer table. This task is given relatively high **priority 7,** in order to ensure smooth operation, which is the primary goal of this application.

3.3.2 Stimulus/Response Sequence:

First of all, the customers will be shown a menu by the waiter. The customers will select the menu items. Once selected, the menu items needed at that table will be communicated to the kitchen staff via the application. After the order is ready, the kitchen staff will notify the particular waiter that the items needed by the table he is attending are ready. The waiter will then move to the kitchen and bring the items to the customer table. After customers are done eating, the waiter will then proceed with the billing info for that table.

3.3.3 Functional Requirements:

REQ-1: Display all the menu. The menu can be altered by the **manager only** depending on the seasonal/occasional availability of certain dishes and drinks, e.g. soup, fish etc, and stock available during that day.

REQ-2: For each table, add customer’s selected menu items to a list, along with quantity and preference options (if available).

REQ-3: Once the order is finalised, the waiter will hit the **Confirm Order** button and the list will be forwarded to the kitchen staff in a queue in an orderly fashion.The kitchen staff will be able to view the table number, the waiter for that table, and the list of items needed.

REQ-4: The kitchen staff will prepare the food and once the order for a particular table is ready, the kitchen staff will hit the **Order Ready** button. This button will notify the particular waiter that the order for a table he is attending is ready for pickup.

REQ-5: As soon as the waiter picks up an order, the kitchen staff will hit the **Order Dispatched** button. This will add the particular order to the list of orders served and remove it from the current orders queue.

REQ-6: During the time customers are eating, they will be given an option to further add items to their order if needed. The waiter can continue adding items to their order and it will be placed to the kitchen staff as a new order.

REQ-7: The waiter will be able to view all the order items for a particular table he is attending. When the customers are done eating, the waiter can hit the **Calculate Bill** button. This will show the total amount due for all the orders this table has placed.

REQ-8: The waiter will then collect the amount and hit the **Table Cleared** button, which will mark that the customers have left and notify the busboy that this particular table is dirty and needs cleaning.

3.4 Managing Dirty Tables:

3.4.1 Description & Priority:

This feature would make sure that as soon as a customer leaves, their table is cleaned by a busboy and mark it ready for the next waiting customer. The process would be optimized for speed so that maximum number of customers can be seated. This is a less significant, but preferable feature and is assigned **priority 6**.

3.4.2 Stimulus/Response Sequence:

After the waiter has taken the payment from a customer, he/she would mark that table as dirty on the app. The application would inform a free busboy to clean that table. After that busboy has cleaned that table, he/she would mark it as cleaned and the table would appear free to the welcomer so he can seat more patrons.

3.4.3 Functional Requirements:

REQ-1: The application would display a list of all occupied tables to the waiter. When the waiter has taken the bill from a table he can mark that table as dirty.

REQ-2: The application would have a list of free busboys in the background. It would search for a random busboy and send him a ping to go clean the table. If the busboy does not respond in 10 seconds the application would assign the table to another busboy.

REQ-3: The busboy would be declared busy as he cleans the table. After the busboy has cleaned the table, he will declare it as cleaned from the application.

REQ-4: The application would update the free tables array and add the cleaned table to that array. Through this, the welcomer would see that a table has been freed (colored blue).

REQ-5: Lastly, the application would update the free busboy list and add the busboy, who cleaned the table, to that list.

4. External Interface Requirements:

4.1 User Interface:

The following is meant to be a guide in designing the user interface for each employee category.

4.1.1 Login/Main Screen:

First of all, the main login screen should be available to all kinds of users as soon as the application is launched. After a user has logged in with his credentials, then the specific interfaces that employee has privileges over should be displayed. An option to logout should be available for all users on the top right corner, along with his name at all times.

4.1.2 Welcomer:

The welcomer will get the view of the entire floor. If he chooses 'ground floor', all the tables on the ground floor will be displayed just like viewing them from the top. Each vacant table will be displayed in blue color, occupied table in red, and dirty tables that need cleaning will be colored in green.

Each table will be a clickable item. As soon as a customer(s) arrives, he/they will be assigned a customer number. All the tables selected by this customer will hold this customer number for identification purposes.

As soon as a table is selected, it will be colored blue. Any merged tables should also be viewed as merged tables. When the customers empty the merged tables, they should be unmerged again to be displayed as single tables.

The second screen accessible to the welcomer will contain the list/queue of waiting customers sitting in the waiting lounge. As soon as tables are vacant, the welcomer will be notified and he will call waiting customers in order to select their tables.

4.1.3 Waiter:

The waiter will have a list of all the table numbers he is attending on the main page. In case tables are merged, the list will contain the group of tables selected in the same entry. The customer identity number associated with each table/group of tables will be displayed to the waiter in each cell.

At the bottom of the screen, there will be a button **New Table.** This will allow the waiter to attend new incoming customers at the tables they have selected.

Once the waiter clicks on a particular table he is attending, he will have a drop down button on the top left corner to show the menu.

Once the menu is displayed, customers will have the option to select/deselect items, along with quantity and special instructions/options if applicable. At the top right, they will have an option to review their order.

At the bottom will be the **Confirm Order** button that will place the order to the kitchen staff.

For each order ready by the kitchen staff, the waiter will have an option to view all the **Prepared Orders** and also **Pending Orders** that are under preparation by the kitchen staff.

4.1.4 Kitchen Staff:

The chefs would be shown a list of orders that they have to prepare. They can choose an order from the list to open it.

In a single order, they would be shown a **checklist** of the items that the order consists of, allowing them to check off the items they have prepared.

Additionally, at the bottom of the order would be an **Order Ready** button, pressing which would inform the waiter to come pick the order up and would remove the order from the list of orders.

4.2 Hardware Interface:

The application would run exclusively on Android tablets and hence would use **Touch Screen** for taking input. The GUI would be designed to make the best use of the big tablet touch screen. For communication, Wifi or WLAN would be used so that the tablets can wirelessly communicate with each other at high speeds.

4.3 Software Interface:

The application will be developed for Android based Operating System. The messaging system will be managed over the wireless local area network, which includes mainly the communication between waiters and kitchen staff regarding customer orders. The information regarding tables will be shared amongst all the devices connected to the restaurant network. The application will use a local database server located inside the restaurant, and can be accessed over the local internet.

4.4 Communication Interface:

The application would use a local area network to send data between devices. To achieve this, messages would be sent using HTTP protocol and would be SSL encrypted to prevent anyone from spying and altering the data. Furthermore, the application would use a single backend database to synchronize data between all the devices. Lastly, a low data transfer rate would be needed as the messages are of small size enabling the application to work in any network situation.

5. Other Non Functional Requirements:

5.1 Performance Requirements:

The system can have delays upto 10 seconds to update things like free tables, sending orders to the kitchen etc. However, there must be minimum delay (less than 3 seconds) in sending the “order prepared” message from the kitchen to the waiter so that the food does not get cold. Lastly, there must be a delay of less than 5 seconds to retrieve customer data from the database so that the waiter can immediately recognize and inform the customer of any discounts they might have accumulated over the past.

5.2 Safety Requirements:

The employees must be constantly reminded not to walk while looking at the screen on the application. This would be done to prevent the employees from crashing into each other and disrupting the customers. Furthermore, the application should run in a blue light filtering mode (yellow hue screen) so that it has minimal effect on the employees eyesight and eye fatigue as they use it throughout the day.

5.3 Security Requirements:

The calculated salary must only be accessible and dispatchable by the manager. Secondly, all the transactions done to and from the database must be secure. Lastly, no one else, besides the employees, should be able to send signals into the application’s network.

5.4 Software Quality Attributes:

The GUI must be intuitive and easily adaptable. Furthermore, an employee type should only be able to see the part of the application made specifically for them and should not have to worry about the working of another employee type’s section. Lastly, the application should focus on being easy to learn so that the employees can adopt it in no time.

6. Other Requirements:

6.1 Legal Requirements:

The application, in no case, should be distributed outside the restaurant it was developed for. In case of a violation, legal action would be taken against the client in shape of a lawsuit. Moreover, the application is to remain a property of the developers, with a small developer watermark branding at the bottom. Any attempts to hide this watermark would also be treated as a legal violation.

Appendix A: Glossary

Busboy: The person who is responsible for cleaning the tables when they are dirty.
Welcomer: The person who will attend the customers at the entrance of the restaurant.

The rest of the items/wordings used are in basic English and do not need further explanation.

Appendix B: Analysis Models

Not applicable

Appendix C: Issues List

The issues list is not applicable as of yet. Further development on the project might help identify issues that need to be resolved.