**Software Engineering**

**Software Requirements Specification**

**(SRS) Document**

John

6/11/2023

V 1

| **Revisions** |
| --- |

| **Version** | **Primary Author(s)** | **Description of Version** | **Date Completed** |
| --- | --- | --- | --- |
| 1 | John Spink | The minimal viable product | 6/11//2023 |

| **Review & Approval** |
| --- |

**Requirements Document Approval History**

| **Approving Party** | **Version Approved** | **Signature** | **Date** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

**Requirements Document Review History**

| **Reviewer** | **Version Reviewed** | **Signature** | **Date** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

| **Table of Contents** |
| --- |

[1. Introduction 3](#_gjdgxs)

1.1 Purpose 3

1.2 Document Conventions…………………………………………………………………….3

1.3 Intended Audience…………………………………………………………………………3

1.4 Scope……………………………………………………………………………………….3

1.5 References………………………………………………………………………………….3

[2. General Description 3](#_30j0zll)

2.1 Product Perspective………………………………………………………………………...3

2.2 Product Features 3

2.3 User Class Characteristics 3

2.4 Operating Environment 3

2.5 Constraints 3

2.6 Assumptions and Dependencies 3

[3. System Requirements 3](#_1fob9te)

[4. External Interface Requirements 4](#_3znysh7)

[4.1 User Interfaces 4](#_2et92p0)

[4.2 Hardware Interfaces 4](#_tyjcwt)

[4.3 Communications Interfaces 4](#_3dy6vkm)

[4.4 Software Interfaces 4](#_1t3h5sf)

[5. Non Functional Requirements 4](#_4d34og8)

**1. Introduction**

* 1. **Purpose:** The goal of this project is Gus’s fine eating as an MVP for running a restaurant

.

* 1. **Document conventions:**

MVP - minimum viable product.

OMS - object management system

* 1. **Intended audiences:** the intended audience would be the Software Development Company LLC team and the customers wanting this OMS.
  2. **Scope:** This program's goal is to handle orders taken by servers and get them to the kitchen staff efficiently. It is also handling the table reservations and waiting list.
  3. **References:** A list of other documents that the SRS document refers to including sources such as websites or written literature.

## 2. General Description

**2.1 Product perspective:** Gus’s fine eating is requesting an OMS for the restaurant to run smoothly.

**2.2 Product features:**

There would be 4 prominent features revolving around the front Service, Servers, Back kitchen, and the customers.

The front Service will greet and seat the customers if tables are open or put them on the waiting list if full.

The servers will be getting and delivering the orders and cleaning the tables afterward.

The back kitchen will be receiving and preparing the orders managing inventory and cleaning the cooking area.

The customers would be able to reserve tables or walk into the front service.

* 1. **User class and characteristics:** the users of the application would be the employees and the customers.

The customers would have access to their reservation/waiting list.

The employees would have access to all the other features.

* 1. **Operating environment:** This was not specified possibly in-house and/or on a website/app.
  2. **Constraints:** The MVP would not have the menu options as they would be added after.
  3. **Assumptions and dependencies:** This application would need the tables and orders labeled ahead of time along with an ingredients list.

## 3. System Requirements

**3.1 Functional requirements**

All the requirements within the system or subsystem in order to determine the output that the software is expected to give in relation to the given input. These consist of the design requirements, graphics requirements, operating system requirements, and constraints if any.

Current Date Tracker

Set date and time methods needed to adjust and test the reservations and other systems

Unit Test and QA (but we will work on this later in the next course significantly)

Food management (Back Kitchen)

Intake Orders (Receive new orders from customers)

Prepare Orders (FIFO queue cooking)

Sending orders to the correct table

Manage Ingredient Inventory

Clean cooking area

Front Service

Greet and take the name and party size (Add to the waiting queue)

Make a Reservation System for tables (NO DOUBLE BOOKING)

Walk-in (Remember to use priority queue as reservations come in)

Seat customer at a table

Take payment (process card/cash)

Table Service

Take Order at the table

Serve food to the table

Serve Drinks

Give check

Clean table after the customer finishes

## 4.External Interface Requirements

**4.1 User Interfaces**

The logic behind the interactions between the users and the software. This includes the sample screen layout, buttons, and functions that would appear on every screen, messages to be displayed on each screen and the style guides to be used.

Servers would get a layout of the restaurant with tables to choose from to seat customers.

The tables would open to an order food page to send to the kitchen.

**4.2 Hardware Interfaces**

All the hardware-software interactions with the list of supported devices on which the software is intended to run on, the network requirements along with the list of communication protocols to be used.

The hardware would be tablets with Wi-Fi

Possibly MQTT Message Queuing Telemetry Transport to message the customer's mobile phone and update other servers that a table is full/empty/not clean.

**4.3 Communications Interfaces**

This project should have communications transfer between users reasonably fast and simultaneously to reduce wait times.

**4.4 Software Interfaces**

The software interfaces for the front end: customer reservations, customer waitlist, Front Service check-in, server orders, kitchen tasks, ingredients list, and menu.

## 5. Non-Functional Requirements

**5.1 Performance requirements**

The interconnectivity needs to update all users within the same 5 seconds to ensure that the table information and orders are the latest data for the servers, front service, kitchen, and customers.

**5.2 Safety requirements**

Separated customers and workers.

**5.3 Security requirements**

Payment should be secure and not saved customers' data should not be public.

**5.4 Software quality attributes**

The program needs adaptability for new /changing tables and order items.

The program should also be flexible for the server's orders to have notes.

The program should be reliably in-sink for coordination.

**5.5 Other requirements**

These may include legal requirements, resource utilizations, future updates, etc.

UML diagram

