Software Engineering
Software Requirements Specification (SRS) Document

[Project Name]

[February 20, 2023]

[1]

By: [Cristian, Collin, Nino]

[WE HAVE ABIDED BY THE UNCG *Academic Integrity Policy* ON THIS ASSIGNMENT]

Table of Contents

1. Introduction

1.1. Purpose

[The goal of your project and the objectives it wishes to accomplish]

The goal of the Restaurant Application is to allow employees of any restaurant access to more efficient modes of management of the workflow. With features such as seating options and arrangement , order status , menu etc.

1.2. Document Conventions

[Full description of the main objectives of this document in the context of your project.
Here's how you should begin this section:
"The purpose of this Software Requirements Document (SRD) is to..."
"In it, we will . . ., . . ., and . . .."]

The Purpose of this Software Requirements Document (SRD) is to allow for Hosts/Waiters to view the seating for a customer and pick according to the count,Set the customers order view its status and Carry out transactions through their view. Chefs will be able to see inventory to know what they can cook from the menu, set the order status, view the orders timer(how long the order has been waiting). The General manager will set the menu items, see and set inventory and be able to order inventory.

1.3. Definitions, Acronyms, and Abbreviations

[Include any specialized terminology dictated by the application area or the product area.
For example:]

Java	A programming language originally developed by James Gosling at Sun Microsystems. We will be using this language to build the Restaurant Manager.
MySQL	Open-source relational database management system.
.HTML	Hypertext Markup Language. This is the code that will be used to structure and design the web application and its content.
SpringBoot	An open-source Java-based framework used to create a micro Service. This will be used to create and run our application.
MVC	Model-View-Controller. This is the architectural pattern that will be used to implement our system.
Spring Web	Will be used to build our web application by using Spring MVC. This is one of the dependencies of our system.
Thymeleaf	A modern server-side Java template engine for our web environment. This is one of the dependencies of our system.
NetBeans	An integrated development environment (IDE) for Java. This is where our system will be created.
API	Application Programming Interface. This will be used to implement a function within the software where the current date and time is displayed on the homepage.

1.4. Intended Audience

[Describe which part of the SRS document is intended for which reader. Include a list of all stakeholders of the project, developers, project managers, and users for better clarity.]

The intended audience is restaurant business owners that are looking for a better way to manage their business.

1.5. Project Scope

[Specify how the software goals align with the overall business goals and outline the benefits of the project to business.]

The goal of the software is to provide an easy-to-use interface for all customers, employees, and managers of a restaurant, as well as provide customers with flexibility to meet their needs. This aligns with the overall business goals of a restaurant as a restaurant requires fast and efficient service in order to fulfill the needs of its customers.

The benefits of the project to business include:

- Relieving stress and pressure from employees and managers as customers are given the opportunity to request services when needed.
- Increasing pleasure to customers as they are given more power when they want to order rather than having to wait for an employee to ask for their order.
- Reducing the amount of time that a customer needs to wait; therefore, increasing the amount of customers that are able to be served in the restaurant within a day.

1.6. Technology Challenges

[Any technological constraints that the project will be under. Any new technologies you may need to use]

New technologies will be our new found knowledge of Spring Boot, SQL ,Thyme Leaf which have recently been introduced to us in a software development environment.

1.7. References

[Mention books, articles, web sites, worksheets, people who are sources of information about the application domain, etc. Use proper and complete reference notation. Give links to documents as appropriate. You should use the APA Documentation model (Alred, 2003, p. 144).]

Ntini, Sunny (Spring 2023) CSC 340-01 Software Engineering.

2. General Description

2.1. Product Perspective

[Describe the context and origin of the product.]

Our Restaurant App found its way through our former teammate Tommy's idea to better make a restaurant function with the help of technology.

2.2. Product Features

[A high-level summary of the functions the software would perform and the features to be included.]

The functions of the product will include a typical login log out with the General Manager setting the menu, seeing inventory and being able to order inventory and other administrative abilities such as adding and

deleting users. Waiters/Host and Chefs seeing and setting and orders status and Waiters/Hosts processing its transaction and seeing seating.

2.3. User Class and Characteristics

[A categorization and profiling of the users the software is intended for and their classification into different user classes]

Our application expects users to have knowledge of the ordering process in a restaurant and no in depth knowledge of operating a computer.

2.4. Operating Environment

[Specification of the environment the software is being designed to operate in.]
The application is designed to operate in any dine-in restaurant.

2.5. Constraints

[Any limiting factors that would pose challenge to the development of the software. These include both design as well as implementation constraints.]

Limited knowledge in the field of software engineering allows for basic backend development to implement necessary functions as well as limiting time constraints.

2.6. Assumptions and Dependencies

[A list of all assumptions that you have made regarding the software product and the environment along with any external dependencies which may affect the project]

The software will be dependent on Spring Web and Thymeleaf in order to create and execute the MVC architecture that will be developed within NetBeans. The application will also use the World Time API (http://worldtimeapi.org/) that will display the current date and time on the home dashboard for everyone to see.

3. Functional Requirements

[Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.]

3.1. Primary

[All the requirements within the system or sub-system 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.]

- FR0: The system will allow the user to lookup of vehicle owner information based on license plate number. This information will contain owner's permit number, assigned lot, and previous violations including tow history.
- FR1: The system will allow the user to enter a new vehicle into the vehicle violation database.
- FR2: The system will allow the user to issue a ticket. The ticket information will be issued in electronic and paper form.

- FR0: The system will allow the user to see, order and input inventory items
- FR1: The System will allow the user to see and change the menu

- FR2: The system will allow the user to create an order, see order status, all orders ,and times
- FR3: The system will allow the user to view seating
- FR4: The system will allow the user to process transactions.

3.2. Secondary

[Some functions that are used to support the primary requirements.]

- Password protection for information only accessible to employees, managers, and each individual table.
- Authorization scheme so that customers can only alter and see their orders and no other customers' orders.

4. Technical Requirements

4.1. Operating System and Compatibility

[The environments that will be needed to operate the system]

The application will be compatible with any operating system that is able to view and to interact with traditional web pages.

4.2. Interface Requirements

4.2.1. 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.]

4.2.2. 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 web application will run on any hardware device that has access to the internet, the ability to display webpages, and the ability to interact with web pages. This includes, but is not limited to, smartphones, tablets, desktop computers, and laptops.

4.2.3. Communications Interfaces

[Determination of all the communication standards to be utilized by the software as a part of the project]

It must be able to connect to the internet as well as the local database on phpMyAdmin. The communication protocol, HTTP, must be able to connect to the World Time API and return the current date and time.

4.2.4. Software Interfaces

[The interaction of the software to be developed with other software components such as frontend and the backend framework to the used, the database management system and libraries describing the need and the purpose behind each of them.]

We will use React and Spring Boot ThymeLeaf to help build the frontend, as well as JPA for the backend database functionality. We will also use Spring Boot with Java to connect the frontend to the backend.

5. Non-Functional Requirements

[Constraints on the services or functions offered by the system (e.g., timing constraints, constraints on the development process, standards, etc.). Often apply to the system as a whole rather than individual features or services.]

5.1. Performance Requirements

[The performance requirements need to be specified for all the functional requirements.]

- NFR0(R): The local copy of the vehicle violation database will consume less than 20 MB of memory
- NFR1(R): The system (including the local copy of the vehicle violation database) will consume less than 50MB of memory
- NFR2(R): The novice user will be able to create and print a ticket in less than 5 minutes.
- NFR3(R): The expert user will be able to create and print a ticket in less than 1 minute.

5.2. Safety Requirements

[List out any safeguards that need to be incorporated as a measure against any possible harm the use of the software application may cause.]

5.3. Security Requirements

[Privacy and data protection regulations that need to be adhered to while designing of the product.]

- NFR4(R): The system will only be usable by authorized users.

5.4. Software Quality Attributes

[Detailing on the additional qualities that need to be incorporated within the software like maintainability, adaptability, flexibility, usability, reliability, portability etc.]

5.4.1. Availability

[Details]

5.4.2. Correctness

[Details]

5.4.3. Maintainability

[Details]

5.4.4. Reusability

[Details]

5.4.5. Portability

[Details]

5.5. Process Requirements

5.5.1. Development Process Used

[Software Process Model]

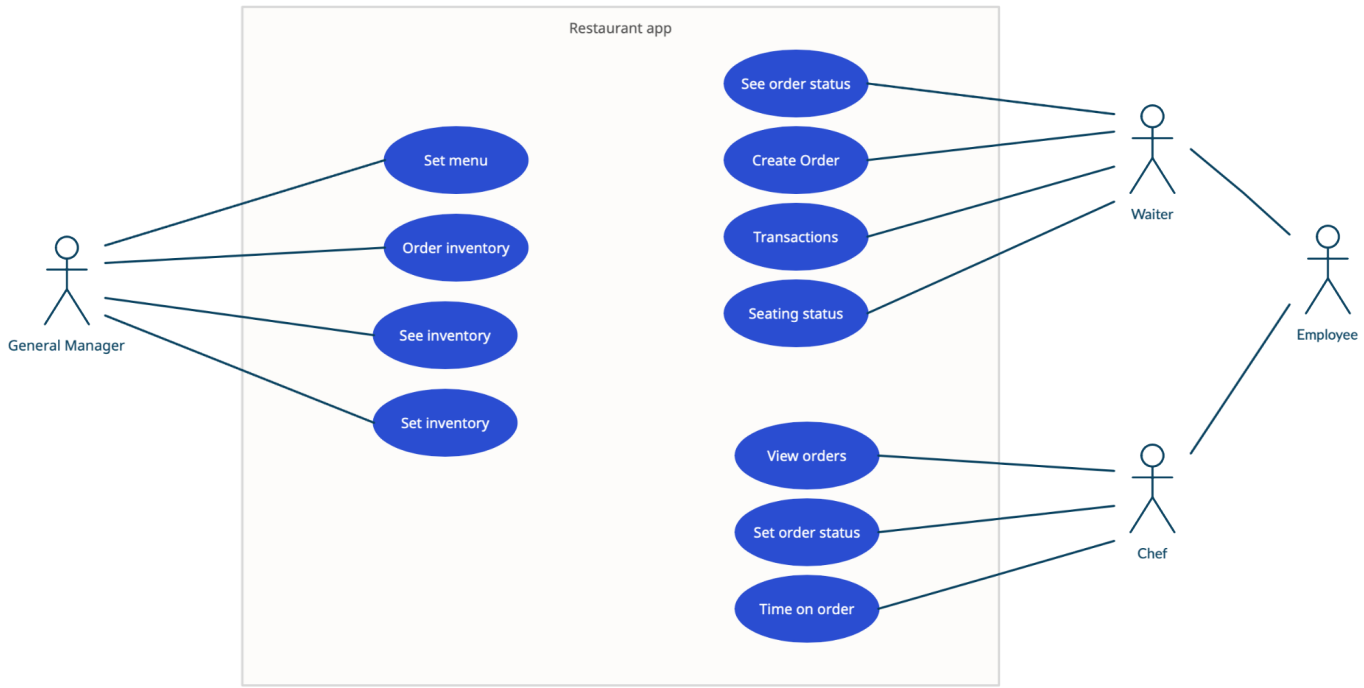
5.5.2. Time Constraints

5.5.3. Cost and Delivery Date

5.6. Other Requirements

TBD

5.7. Use-Case Model Diagram



5.8. Use-Case Model Descriptions

5.8.1. Actor: General Manager (Cristian)

- **Set menu**: [General Manager sets the menu for the restaurant.]
- **Order inventory**: [General manager orders more supplies to restock.]
- **See inventory**: [General Manager can view the current inventory]
- **Set inventory**: [General Manager can set the amount of inventory.]

5.8.2. Actor: Waiter (Nino)

- **See order status**: [Views status of order]
- **Create order**: [Creates a new order]

- **Transactions**: [Shows the transaction]
- **Seating status**: [Views the seating status]

5.8.3. Actor: Chef (Collins)

- **View orders**: [View the current orders]
- **Set order status**: [Sets the current order status]
- **Time on order**: [View the time on the order]

5.9. Use-Case Model Scenarios

5.9.1. Actor: Chef (Collins)

- **Use-Case Orders**:
    - **Initial Assumption**: The Chefs will be able to view the orders submitted by Waiters and update the status of the order for the waiters to view and pick up. The chefs will also be able to see the timer of an order to then prioritize latter orders.
    - **Normal**: Chef can see order status, each order's details and the time set on an order.
    - **What Can Go Wrong:** if the chef set the wrong order status to an order causing orders that are not ready to be pushed, and orders that are ready to be late
    - **Other Activities**: Chef setting order status should notify waiter
    - **System State on Completion**: Order Complete
- **Use-Case See Inventory**:
    - **Initial Assumption**: Chef should be able to see the inventory so they can determine what they can cook and if they have enough ingredients for the work day.
    - **Normal**: See Inventory
    - **What Can Go Wrong**: during the workday a certain key ingredient goes down the chef would have to notify the general manager that there is no longer a key ingredient
    - **Other Activities**: Emergency Loss of a certain item in the inventory should notify the manager (so the manager would make the action of setting the menu accordingly)
    - **System State on Completion**: Inventory View

5.9.2. Actor: Waiter (Nino)

- **Use-Case orders**:
    - **Initial Assumption**: The waiter will be able to input the customers order into the system so that the chefs can see. The waiter will be able to cancel the orders. The waiter will be able to see when the orders will be completed.
    - **Normal**: The waiter will be able to see customers orders and the bills.
    - **What Can Go Wrong**: If the waiter mis inputs, that will cause the wrong customer to be billed. Overcharging and undercharging customers.
    - **Other Activities**: The waiter will be able to check with the chef if an item is available.
    - **System State on Completion**: The customer will be ready for their bill.
- **Use-Case Checkout**
    - **Initial Assumption**: The waiter will be able to go in and charge the customer. The waiter will be able to cancel and refund transactions.
    - **Normal**: The waiter will handle payment forms of customers.
    - **What Can Go Wrong**: Charge the wrong customer.

- **Other Activities**: The waiter will be able to give discounts and rewards to customers
- **System State on Completion**: When the transaction is done the customer will be satisfied and the restaurant will be paid. The waiter will be able to view completed orders.

5.9.3. Actor: General Manager (Cristian)

- **Use Case Set Menu**:
  - **Initial Assumption**: The general manager has logged into their account and navigated to the set menu tab.
  - **Normal**: The general manager will be able to set the menu of the restaurant.
  - **What Can Go Wrong**: The manager inputs the wrong amount for an item.
  - **Other Activities**: An item can be updated or set a new amount.
  - **System State on Completion**: The menu is set successfully. It has developed a menu for the restaurant. The menu would be available to view for customers.
- **Use Case Order Inventory**:
  - **Initial Assumption**: The general manager has logged into their account and navigated to the order inventory.
  - **Normal**: The manager will be able to make an order for a resupply of the current inventory.
  - **What Can Go Wrong**: Miscalculation in the amount of inventory. Inputting the wrong amount.
  - **Other Activities**: An order can be edited based on the item.
  - **System State on Completion**: The order for the resupply is sent successfully. The manager can view the statement of the order and previous orders.

6. Design Documents

6.1. Software Architecture

6.2. High-Level Database Schema

6.3. Software Design

6.3.1. State Machine Diagram: Actor Name (Responsible Team Member)

6.3.2. State Machine Diagram: Actor Name (Responsible Team Member)

6.3.3. State Machine Diagram: Actor Name (Responsible Team Member)

6.4. UML Class Diagram

7. Scenario

7.1. Brief Written Scenario with Screenshots