**Software Engineering**

**Software Requirements Specification**

**(SRS) Document**

**Joe’s Ristorante Online Ordering System**

**4/21/2020**

**1**

**By: DeveloperCorp**

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

[1. Introduction 3](#_Toc244519333)

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](#_Toc244519334)

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](#_Toc244519335)

[4. External Interface Requirements 4](#_Toc244519336)

[4.1 User Interfaces 4](#_Toc244519337)

[4.2 Hardware Interfaces 4](#_Toc244519338)

[4.3 Communications Interfaces 4](#_Toc244519339)

[4.4 Software Interfaces 4](#_Toc244519340)

[5. Non Functional Requirements 4](#_Toc244519341)

1. Introduction

* 1. **Purpose:** This Software Requirements Specification (SRS) document describes the functions and details of Joe’s Ristorante Online Ordering System. This document should serve as the basis for a software team to create the system.
  2. **Document conventions:**

|  |  |
| --- | --- |
| DB | Database |
| JROOS | Joe’s Ristorante Online Ordering System |
| JSON | JavaScript Object Notation |
| REST | Representational State Transfer |
| Ticket | An order of pizzas by a customer |

* 1. **Intended audience:**  This document is to be sent to all members of the Scrum development team. All members should read this document to gain an overview of what must be achieved in this application. The product owner of the application is Mr. Adam C Ranieri who is reachable at adam.ranieri@revature.com.
  2. **Scope:** This application is to be marketed to potential customers in areas nearby Joe’s Ristorante. JROOS will allow customers to place orders online for pickup at our location.
  3. **References:** 
     1. **Figure A (JROOS ERD)** [**https://drive.google.com/open?id=1T3XK9vlmYu1XO2XJPL9Mhs\_9Mm03NP8l**](https://drive.google.com/open?id=1T3XK9vlmYu1XO2XJPL9Mhs_9Mm03NP8l)
     2. **User Stories** [**https://docs.google.com/spreadsheets/d/1N6Kwct5xGqUcVIyhIT0rKA-1dmG7A\_KbyN6f8jMy-hw/edit?usp=sharing**](https://docs.google.com/spreadsheets/d/1N6Kwct5xGqUcVIyhIT0rKA-1dmG7A_KbyN6f8jMy-hw/edit?usp=sharing)
     3. **Figure B GUI layouts** [**https://drive.google.com/open?id=1SGkFqIJWPvEIcbTbuk8Myx7T-XPrLulI**](https://drive.google.com/open?id=1SGkFqIJWPvEIcbTbuk8Myx7T-XPrLulI)

## 2. General Description

**2.1 Product perspective:** Joe’s Ristorante is a small family owned restaurant in Morgantown, West Virginia. The restaurant has been established for over 60 years. Increasing popularity means that answering phone calls to place orders takes a considerable amount of time for employees. Lack of an online website makes it difficult for customers to know the entire menu. During the busiest hours tickets are lost or information misinterpreted. This leads to an overall decline in consumer experience.

To maximize workflow and increase efficiency Joe’s Ristorante wants JROOS. This application will allow the employees to graphically see pending orders. This should minimize the chances of an order being mishandled or fulfilled incorrectly. Additionally, many of their younger customers expect and prefer an online ordering experience.

**2.2 Product features:** This application will consist of two parts. First a web application that will allow users to interface with the system. This application will be the primary way both customers and employees interact with the system. Second, there should be a RESTful web service for the tickets and querying the database. This web service will be used by an inventory management systems to automatically notify when a recorder of supplies must be placed.

* 1. **User class and characteristics:** 
     1. Unregistered Customer – A new customer without an account
     2. Registered Customer – A customer with an account
     3. Employee – An employee of Joe’s Ristorante
  2. **Operating environment:** This application is to be run on an AWS EC2 running on either a Linux or Windows OS.
  3. **Constraints:** Joe’s Ristorante lacks IT infrastructure aside from a few tablets and laptops. The entirety of the system should be accessible over the internet with no on premise infrastructure.
  4. **Assumptions and dependencies:** Joe’s Ristorante has high speed internet within its establishment. All customers of the application will have an internet connection and the ability to read screens and menus without impairment.

## 3. System Requirements

**3.1 Functional requirements**

The following should serve as a high-level overview of functionalities to be performed by JROOS. Detailed user stories with the acceptance criteria can be found here, [user stories](https://docs.google.com/spreadsheets/d/1N6Kwct5xGqUcVIyhIT0rKA-1dmG7A_KbyN6f8jMy-hw/edit?usp=sharing), as well as the referenced materials section.

An unregistered customer should have the following functionalities:

* Register for an account
* Create a ticket
* Leave a note with their ticket order for special instructions

A registered customer should have the following functionalities:

* Create a ticket
* View all past tickets
* Leave a note with their ticket order for special instructions

An employee should have the following functionalities:

* View all tickets
* Organize tickets based on status and time received
* Change the status of tickets

The RESTful API should have the following functionalities:

* Retrieve all tickets
* Retrieve tickets by username
* Retrieve all toppings
* Retrieve toppings by order popularity( the highest topping ordered )
* CRUD operations on toppings
* CRUD operations on tickets

**3.2 Design Requirements**

* Front-end
  + The front end must be developed with Angular
  + Deployment of the front end should be to an S3 bucket
* Back-end
  + Must be written in Java
  + Employs Spring Boot and the following Spring modules
    - Spring WEB
    - Spring DATA
* Database
  + Must adhere to ERD specified in figure A
    - Modifications can created as system expands
    - All pizzas have a topping of Basic Cheese which serves as the minimum cost of any pizza.
  + RDBMS should be MariaDB

**3.3 Testing Requirements**

* JUnit5 tests of all services within the application
* Postman collection of API endpoints

## 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.

4.2 Hardware Interfaces

There are no hardware interfaces for this web application.

4.3 Communications Interfaces

Communication between the front end and back end should be entirely in JSON.

4.4 Software Interfaces

This application will be used in a browser by customers and employees. This application must be accessible and display correctly via Chrome and Firefox.

## 5. Non-Functional Requirements

**5.1 Performance requirements**

* No one page of the web application should take longer than 3 seconds to load.
  + We want the user experience to be as straightforward as possible

**5.2 Safety requirements**

* We do not anticipate any safety requirements for this application

**5.3 Security requirements**

* There is no sensitive information stored in the database
* There is no identifiable information in the database
* Passwords may be stored in plain text
* Site may be accessed over http

**5.4 Software quality attributes**

* Application source code should be readily available in a public repository on GitHub
* Features of the application should be added via branching and pull requests to create a detailed history of development
* Static code analysis should be run at the end of every sprint to identify code smells

**5.5 Other requirements**

API endpoints should be documented with OpenAPI 3.0. This will allow future applications to easily consume the web service aspect of this application.