***Software Engineering***

***Software Requirements Specification***

***(SRS) Document***

**Allerfence**

**2/17/2024**

**1.0**

**By: Kevin Ruiz, Dylan Muroki, Lindsey Flores**

**[We all followed the UNCG Honor Code]**

**Table of Contents**

[1. Introduction 3](#_Toc156255120)

[1.1. Purpose 3](#_Toc156255121)

[1.2. Document Conventions 3](#_Toc156255122)

[1.3. Definitions, Acronyms, and Abbreviations 3](#_Toc156255123)

[1.4. Intended Audience 4](#_Toc156255124)

[1.5. Project Scope 4](#_Toc156255125)

[1.6. Technology Challenges 4](#_Toc156255126)

[1.7. References 4](#_Toc156255127)

[2. General Description 5](#_Toc156255128)

[2.1. Product Features 5](#_Toc156255129)

[2.2. User Class and Characteristics 5](#_Toc156255130)

[2.3. Operating Environment 5](#_Toc156255131)

[2.4. Constraints 5](#_Toc156255132)

[2.5. Assumptions and Dependencies 5](#_Toc156255133)

[3. Functional Requirements 6](#_Toc156255134)

[3.1. Primary 6](#_Toc156255135)

[3.2. Secondary 6](#_Toc156255136)

[3.3. Use-Case Model 6](#_Toc156255137)

[3.3.1. Use-Case Model Diagram 6](#_Toc156255138)

[3.3.2. Use-Case Model Descriptions 7](#_Toc156255139)

[3.3.2.1. Actor: Manager (Alice) 7](#_Toc156255140)

[3.3.2.2. Actor: Actor Name (Responsible Team Member) 7](#_Toc156255141)

[3.3.2.3. Actor: Actor Name (Responsible Team Member) 7](#_Toc156255142)

[3.3.3. Use-Case Model Scenarios 7](#_Toc156255143)

[3.3.3.1. Actor: Manager (Alice) 7](#_Toc156255144)

[3.3.3.2. Actor: Actor Name (Responsible Team Member) 7](#_Toc156255145)

[3.3.3.3. Actor: Actor Name (Responsible Team Member) 8](#_Toc156255146)

[4. Technical Requirements 9](#_Toc156255147)

[8.1. Interface Requirements 9](#_Toc156255152)

[8.1.1. User Interfaces 9](#_Toc156255153)

[8.1.2. Hardware Interfaces 9](#_Toc156255154)

[8.1.3. Communications Interfaces 9](#_Toc156255155)

[8.1.4. Software Interfaces 9](#_Toc156255156)

[9. Non-Functional Requirements 10](#_Toc156255157)

[9.1. Performance Requirements 10](#_Toc156255158)

[9.2. Safety Requirements 10](#_Toc156255159)

[9.3. Security Requirements 10](#_Toc156255160)

[9.4. Software Quality Attributes 10](#_Toc156255161)

[9.4.1. Availability 10](#_Toc156255162)

[9.4.2. Correctness 10](#_Toc156255163)

[9.4.3. Maintainability 10](#_Toc156255164)

[9.4.4. Reusability 10](#_Toc156255165)

[9.4.5. Portability 10](#_Toc156255166)

[9.5. Process Requirements 10](#_Toc156255167)

[9.5.1. Development Process Used 10](#_Toc156255168)

[9.5.2. Time Constraints 10](#_Toc156255169)

[9.5.3. Cost and Delivery Date 10](#_Toc156255170)

[9.6. Other Requirements 10](#_Toc156255171)

[10. Design Documents 11](#_Toc156255172)

[10.1. Software Architecture 11](#_Toc156255173)

[10.2. High-Level Database Schema 11](#_Toc156255174)

[10.3. Software Design 11](#_Toc156255175)

[10.3.1. State Machine Diagram: Actor Name (Responsible Team Member) 11](#_Toc156255176)

[10.3.2. State Machine Diagram: Actor Name (Responsible Team Member) 11](#_Toc156255177)

[10.3.3. State Machine Diagram: Actor Name (Responsible Team Member) 11](#_Toc156255178)

[10.4. UML Class Diagram 11](#_Toc156255179)

[11. Scenario 11](#_Toc156255180)

[11.1. Brief Written Scenario with Screenshots 11](#_Toc156255181)

# Introduction

## Purpose

Our application aims to help users find which restaurants are allergy friendly, as well as be able to effortlesly create an order with special requirements as instructions to the restaurant to make sure that they are not ingesting foods which they are allergic to, We also allow for drivers to sign up and deliver food, and for restaurants to input their menu items.

## Document Conventions

The purpose of this Software Requirements Document (SRD) is provide clarity in how Allerfence is meant to wrork. We will highlight user functions ranging from drivers, costumers and restaurants. We will also highlight the technology used to achieve this, providing clarity and malleability to our project in order to be able to be improved on in the future.

## Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| Java | JavaScript, often abbreviated as JS, is a programming language and core technology of the World Wide Web, We will use this to add functionality to our webapp |
| MongoDB | Open-source relational database management system. |
| .HTML | Hypertext Markup Language. This is the code that will be used to structure the web application and its content.. |
| Express | An open-source framework to run our back end server with |
| Node Js | Node.js is a cross-platform, open-source JavaScript runtime environment. Node.js runs on the V8 JavaScript engine, and executes JavaScript code outside a web browser. |
| Visual Code | An integrated development environment (IDE) for many languages which we will use for JS, HTML, and CSS |
| CSS | Cascading Style Sheets is a style sheet language used for specifying the presentation and styling of a document written in a markup language such as HTML or XML, will be used to add styling to our webapp |
| Bcrypt | 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. |

## Intended Audience

Developers

Project Managers

Restaurants

Costumers

## Project Scope

The goal of the software is to provide an easy-to-use interface for all customers, drivers, and restaurants, as well as provide customers with the ability to order their food without the fear of eating something they may be allergic to. This aligns with the overall business goals of a restaurant, because they need to provide food that is safe for consumers and avoid getting complains from costumers

The benefits of the project to business include:

⦁ Relieving stress and pressure from costumers by ensuring that their food will be sage to eat

⦁ Increasing the amount of people that can order from a restaurant because they will know if it is safe for them to eat there

⦁ Allowing people to gain extra money from delivering food to costumers from restaurants

## Technology Challenges

A working phone or computer, that connects to the internet

## References

N/A

# General Description

## Product Features

Our product will be able to provide costumers with a painless food ordering experience, they will not worry about eating food that will give them allergies and they won’t have to type special requests ever again. Our app will provide restaurants with an easy way to list all their food items and ingredients making a more transparent interaction with costumers, our app will allow for people to earn money through food delivery as well as aliviate the burden of having drivers as staff in restaurants

## User Class and Characteristics

Our Costuemrs do not need to have any background knowledge to use the app just know how to use a computer or a phone, however our Vendors, will need to know how to manage items add and remove, as well as receive orders etcs... And well our drivers need to know how to drive

## Operating Environment

The application is designed to operate on the web across any divice that can connect to the internet

## Constraints

Our app has to be connected to the internet

## Assumptions and Dependencies

Our software depends on NojeJs, Express, and Nodemon to be able to run, as well as our data base MongoDB. We assume that our server has all of these things intigrated

# Functional Requirements

## Primary

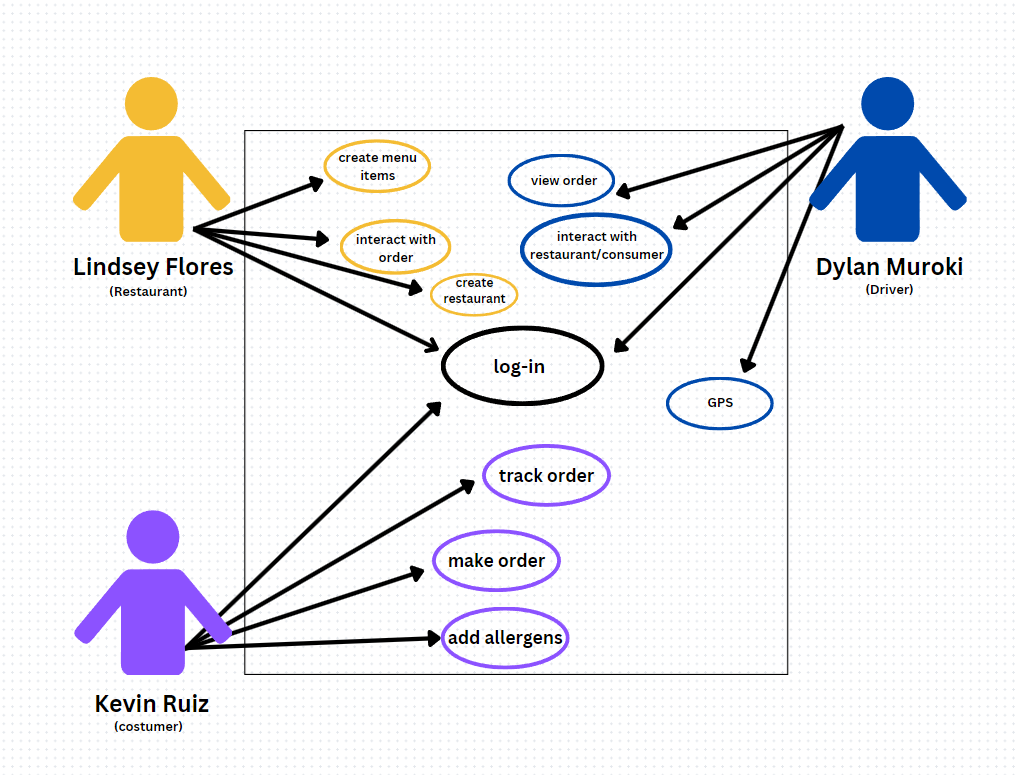
* The system will allow the customer to lookup restaurants and their order as well as how far the driver is from their house
* The system will allow the restaurant to create menu items, update their hours of business and view and print incoming orders
* The system will allow the driver so see active order that requre delivery in the nearby area, and how far the customer is from the restaurant.

## Secondary

Only allow accounts that are designed for specific actions to do, a customer should be able to create menu items for example

## Use-Case Model

## Use-Case Model Diagram



## Use-Case Model Descriptions

### Actor: Costumer (Kevin Ruiz)

* The costumer can input their allergies, order from restaurants, cancel and view order also pay

### Actor: Restaurant (Lindsey Flores)

* Login: [Allows user to login to access their information and use-rights]
* **Creates menu items**: [The restaurant user will be able to add menu items with an ingredients list ]
* **Create Restaurant** : [Restaurant will be able to create their restaurant’s profile]
* Interaction: will allow restaurant to interact with the client users comments to adreess comments and concerns.

### Actor: Actor Name (Responsible Team Member)

* **Use-Case Name**: [Brief Use-Case Description]
* **Use-Case Name**: [Brief Use-Case Description]

## Use-Case Model Scenarios

### Actor: (Kevin Ruiz)

* **Use-Case Name**: Make order
  + **Initial Assumption**: User can log in and order
  + **Normal**: The user can make their order
  + **What Can Go Wrong: The user selects an item from a restaurant that is out of stock, the order never went through because of a connection issue**
  + **Other Activities**: Cancel order/Change Order
  + **System State on Completion**: The order is created and the restaurant can update the time it takes for completion.
* **Use-Case Name**: Track order
  + **Initial Assumption**: Order was recieved by the restaurant
  + **Normal**: Costumer can see how long it will take the restaurant to make order, and how long the driver will take to deliver it
  + **What Can Go Wrong**: Driver didn't notify system of pick up time
  + **Other Activities**:
  + **System State on Completion**: The costumer can track their order in real time
  + Use-Case Name: Add Allergens
* Initial Assumption: User created an account
* Normal: Costumer can add their allergies and a message to send to a restaurant if the items they order have those allergies in them, to remove them if possible
* What Can Go Wrong: Restaurnat doesn't have their menu items ingredients listed,
* Other Activities:
* System State on Completion: User has a profile with their allergies, can order without having to worry about typing special instructions

### Actor: Restaurant (Lindsey Flores)

* **Use-Case Name**: Login
  + **Initial Assumption**: Allows user to login to access their information and use-rights
  + **Normal**: User logs into their assigned restaurant profile
  + **What Can Go Wrong**:
    - User logs into wrong account (i.e client) or user cant login at all
  + **Other Activities**: Signup/Logout
  + **System State on Completion**: User successfully logged in
* **Use-Case Name**: Create Menu Items
  + **Initial Assumption**: The restaurant user will be able to add menu items with an ingredients list
  + **Normal**: User successfully adds a menu item with the correct ingredient
  + **What Can Go Wrong**:
    - User is unable to create menu items correctly
    - User adds wrong ingrediants to wrong menu item
  + **Other Activities**: Remove menu items
  + **System State on Completion**: Menu Item is added to the menu with the listed ingrediants
* **Use-Case Name: Interaction**
  + **Initial Assumption**: This will allow restaurant to interact with the client users comments to adreess comments and concerns.
  + **Normal**:
    - Restaurant is able to reply to comments to the desired user when they want, or in real time
  + **What Can Go Wrong**:
    - Restaurant is unable to see the comments
    - Restaurant is unable to reply to comments
    - Rest. Replies to the wrong user comment
  + **Other Activities**: Unable to delete message
  + **System State on Completion**: Restaurant’s comment is successfully posted.

3.333 **Actor: Actor Name (Dylan Muroki)**

* **Use-Case Name**: View order
* **Initial Assumption**: The customer has placed their order and the restaurant has passed along all of the necessary information
* **Normal**: The driver can accept or deny the order, fully up to their discretion but key factors will be in place such as distance and pay for each order
* **What Can Go Wrong**: Restaurant doesn’t pass along the order information or the customer orders something outside the scope of the driver
* **Other Activities**:
* **System State on Completion**: Driver can see all orders near them in real time
* **Use-Case Name**: Interact with customer and restaurant
* **Initial Assumption**: there is a portal or window that is linked to a customers account and another one for a resturant
* **Normal**: The restaurant can communicate with driver and the customer can also communicate with driver
* **What Can Go Wrong**: A thread cannot be established to the right endpoint so someone gets a message that is no meant for them.
* **Other Activities**:
* **System State on Completion**: All parties have a secure chat line that will be sent along in real time
* **Use-Case Name**:GPS
* **Initial Assumption**: Google Api will be used to gather drivers and restaurant locations
* **Normal**: The Driver gets directions to the restaurant location and real time the customer know how far away the driver is
* **What Can Go Wrong**: Incorrect Directions: Inaccurate mapping data or GPS errors could result in the driver receiving incorrect navigation directions, leading to delays or the driver getting lost
* **Other Activities**:
* **System State on Completion**:Upon successful completion of the use case, the system updates the order status to "delivered," records relevant delivery metrics (e.g., delivery time, driver performance), and releases the driver for the next assignment. The customer receives confirmation of delivery and may be prompted to rate their experience.

# Technical Requirements

## Interface Requirements

### User Interfaces

Our webapp will use button, forms, and images for user to be able to interact with the page, as well as visuals indicating progress of food delivery andorder progress

### Hardware Interfaces

The web application will run on any device that can connect and it connected to the internet they also have to be able to load webpages. Like laptops, and phones

### Communications Interfaces

It must be able to connect to the internet as well as the cloud database on MongoDB/Atlas

The communication protocol, HTTP, must be able to connect to the google maps API, and return restaurants near the adress the user chose

### Software Interfaces

We will use HTML, CSS, and JS to add styling to our user interface and create a responsinve web app, we will also use Express, NodeJS, and Nodemon to run our backend and we will store the information on MongoDB cloud server

# Non-Functional Requirements

## Performance Requirements

Restaurants need to recieve new orders instantly and the ticket should be able to be printed instantly as well

Costumers need to be able to view the items the restaurant has listed in real time

## Safety Requirements

Make sure that the food items the restaurant provide are accurate description of all the ingredients to avoid people having an allergic reaction, when they thought the food they ordered was safe. Drivers need to follow a code of conduct of not tampering with the food and delivering it a timely matter

## Security Requirements

We will use Bcrypt to ensure that passwords as hashed

## Software Quality Attributes

### Availability

Available to anyone in the United States

### Correctness

Ensure that information displayed in true

### Maintainability

Required server Maintence as well as Database Maintence

### Reusability

N/A

### Portability

Portable in any device that the internet can acess via website

## Process Requirements

### Development Process Used

**Waterfall Model**

### Time Constraints

Provided by Professor

### Cost and Delivery Date

$0 Delivery Date, End of Semester?

## Other Requirements

N/A

# Design Documents

## Software Architecture

## High-Level Database Schema

## Software Design

### State Machine Diagram: Actor Name (Responsible Team Member)

### State Machine Diagram: Actor Name (Responsible Team Member)

### State Machine Diagram: Actor Name (Responsible Team Member)

## UML Class Diagram

# Scenario

## Brief Written Scenario with Screenshots