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

1.1. Purpose

Our AllerFence application serves as a solution for users seeking restaurant options tailored to their dietary needs. It facilitates the process of discovering dining establishments and allows users to place orders with customized instructions ensuring they avoid allergens that may pose health risks. Additionally, AllerFence provides a platform for delivery drivers to register and fulfill food orders.

1.2. Document Conventions

The purpose of this Software Requirements Document (SRD) is to provide an understanding of AllerFence's functionality. This document will highlight the key user interactions for drivers, customer, and restaurants, providing clarity in the system's intended operations. Additionally, 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.

1.3. 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.

1.4. Intended Audience

Developers
Project Managers
Restaurants
Costumers

1.5. Project Scope

The goal of the software is offering a user-friendly interface that is accessible to customer, drivers, and restaurants. Furthermore, it aims to empower customers by enabling them to place food orders confidently, without concerns of their allergies. This object aligns seamlessly with the goals of the restaurant in mind, since it emphasizes the provision of safe and satisfactory dining experiences minimizing customer complaints.

The benefits of the project to business include:

- Relieving stress and pressure from costumers by ensuring that their food will be safe to eat.
- Increasing the amount of traffic for restaurants since they are providing safe meals.
- Allowing people to gain extra money from delivering food to costumers from restaurants.

1.6. Technology Challenges

A working phone or computer, that connects to the internet.

1.7. References

N/A

2. General Description

2.1. Product Features

Our product will be able to provide customers 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 effortless 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 alleviate the burden of having drivers as staff in restaurants

2.2. User Class and Characteristics

Our Costumers 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 edit their menu item in their profile and navigate their client orders. And well our drivers need to know how to drive, and how to communicate with the customer that their order is on its way.

2.3. Operating Environment

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

2.4. Constraints

Our app will only work if the user has access to the internet.

2.5. Assumptions and Dependencies

Our software depends on Node.js, Express, and Nodemon to be able to run, as well as our database MongoDB. We assume our server has all these things integrated.

3. Functional Requirements

3.1. Primary

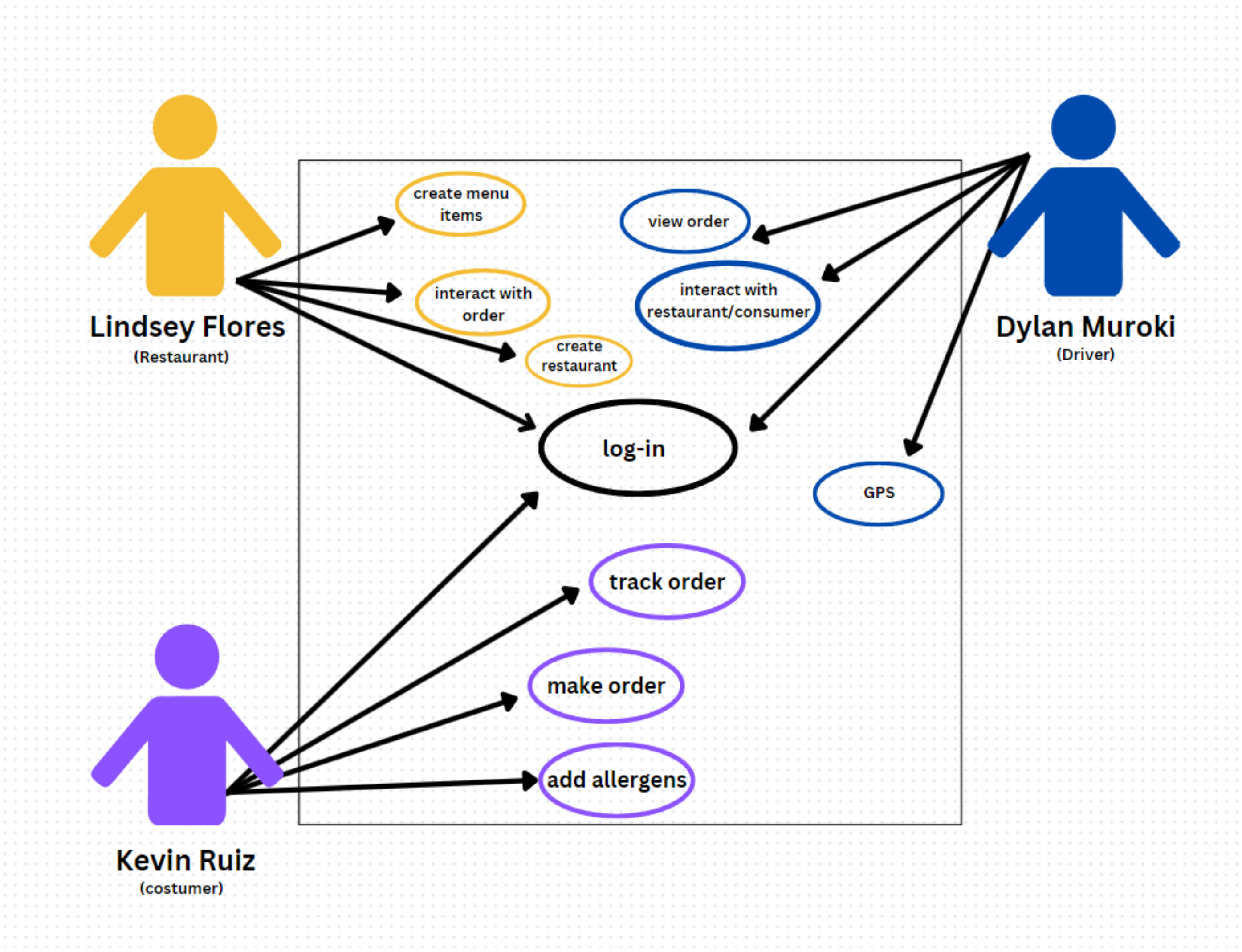
- The system will allow the customer to look up restaurants, view their order, as well as how far the driver is from their house.
- The system will allow the restaurant to create menu items and view incoming orders.
- The system will allow the driver to see the order they picked up, allow drivers to communicate with customers on their order.

3.2. Secondary

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

3.3. Use-Case Model

3.3.1. Use-Case Model Diagram



3.3.2. Use-Case Model Descriptions

3.3.2.1. Actor: Costumer (Kevin Ruiz)

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

3.3.2.2. Actor: Restaurant (Lindsey Flores)

- **Create An Account:** [Allows user to login to access their information and use-rights]

- **Login:** Allows users to login to their existing account with their restaurant information.
- **Creates menu items**: [The restaurant user will be able to add menu items with an ingredients list.]
- **Interact with order:** Allows Restaurant to be able view their incoming orders.

3.3.2.3. Actor: Driver (Dylan Muroki)

- **Interact with Order:** Will be able to interact with the order they picked up.
- **Communicate with Customer**: [Will be able to interact with customer regarding their order]
- **GPS**: GPS to keep customer aware of order status

3.3.3. Use-Case Model Scenarios

3.3.3.1. Actor: (Kevin Ruiz)

- **Use-Case Name**: Make order
  - ∉ **Initial Assumption**: User can log in and create 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 notify when the order is complete.
- **Use-Case Name**: Track order
  - ∉ **Initial Assumption**: Order was received 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 did not communicate their item was picked up.
  - ∉ **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 to let restaurant beware of their allergens, that way those allergens can be removed if possible.
- ∉ What Can Go Wrong: Restaurant doesn't have all their menu items ingredients listed.
- ∉ Other Activities: Customer will be able to communicate with their delivery driver
- ∉ System State on Completion: User has a profile with their allergies, can order without having to worry about typing special instructions

3.3.3.2. Actor: Restaurant (Lindsey Flores)

- **Use-Case Name**: Create an Account
  - ∉ **Initial Assumption**: Allows restaurants to be able to create their account, they will be redirected to create their restaurant profile, then they will be able to create menu items in user profile.
  - ∉ **Normal**: Restaurant manager creates an account, then creates their restaurant and is able to navigate their pages.
  - ∉ **What Can Go Wrong**:

        - User has internet issues creating an account which would have them restart the process of creating an account
    - ∉ **Other Activities**: Creating their restaurant account.
    - ∉ **System State on Completion**: User successfully creates account
- **Use-Case Name**: Login
    - ∉ **Initial Assumption**: The restaurant manager will be able to access their restaurant account.
    - ∉ **Normal**: User logs into their assigned restaurant profile
    - ∉ **What Can Go Wrong**:
        - User is not able to log into their account via email.
        - Forgot Password
    - ∉ **Other Activities**: Updating their Menu via profile
    - ∉ **System State on Completion**: User successfully logs into existing account.
- **Use-Case Name: Create Menu Item**
    - ∉ **Initial Assumption**: Restaurant is able to add a menu item to their menu
    - ∉ **Normal**:
        - Restaurant is able to update their menu items by adding a new menu.
    - ∉ **What Can Go Wrong**:
        - Restaurant is unable to add a menu item
    - ∉ **Other Activities**: Able to view full menu/profile information
    - ∉ **System State on Completion**: Restaurant's menu successfully displays.
- **Use-Case Name: Interact with incoming order**
    - ∉ **Initial Assumption**: This will allow restaurants to view incoming orders.
    - ∉ **Normal**: Restaurant is able to see incoming orders
    - ∉ **What Can Go Wrong**:
        - Restaurant is unable to see an order and it gets lost in DB.
    - ∉ **System State on Completion: Restaurant can view incoming order.**

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

- **Use-Case Name**: Get order
- **Initial Assumption**: The customer has placed their order, and the restaurant has completed order.
- **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 restaurant
- **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.

4. Technical Requirements

Interface Requirements

4.1.1. 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

4.1.2. Hardware Interfaces

The web application will run on any device that can connect to internet and load webpages (i.e laptops and phones)

4.1.3. 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

4.1.4. 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

5. Non-Functional Requirements

5.1. Performance Requirements

Restaurants need to view their new orders instantly. Customers need to be able to view the menu items the restaurant has listed in real time. Drivers must be able to communicate with customers that their order is on their way.

5.2. Safety Requirements

Ensuring that the restaurant provided full menu items, making sure that the food item the restaurant provide are accurate description of all the ingredients to avoid people having an allergic reaction. Customers need to correctly place their allergies when creating profile. Drivers need to follow a code of conduct of not tampering with the food and delivering it a timely matter.

5.3. Security Requirements

We will incorporate Bcrypt to ensure that passwords as hashed.

5.4. Software Quality Attributes

5.4.1. Availability

Available to anyone in the United States

5.4.2. Correctness

Ensure that information displayed in true

5.4.3. Maintainability

Required server Maintence as well as Database Maintence

5.4.4. Reusability

N/A

5.4.5. Portability

Portable in any device that the internet can acess via website

5.5. Process Requirements

5.5.1. Development Process Used

Waterfall Model

5.5.2. Time Constraints

4 months to fully complete a food service application.

5.5.3. Cost and Delivery Date

This project cost nothing out of the pocket of the creators. The delivery date is April 30, 2024.

5.6. Other Requirements

N/A

6. Design Documents

6.1. Software Architecture

Our software architecture leverages a design to ensure flexibility and efficiency in accommodating future enhancements and/or production.

6.2. High-Level Database Schema

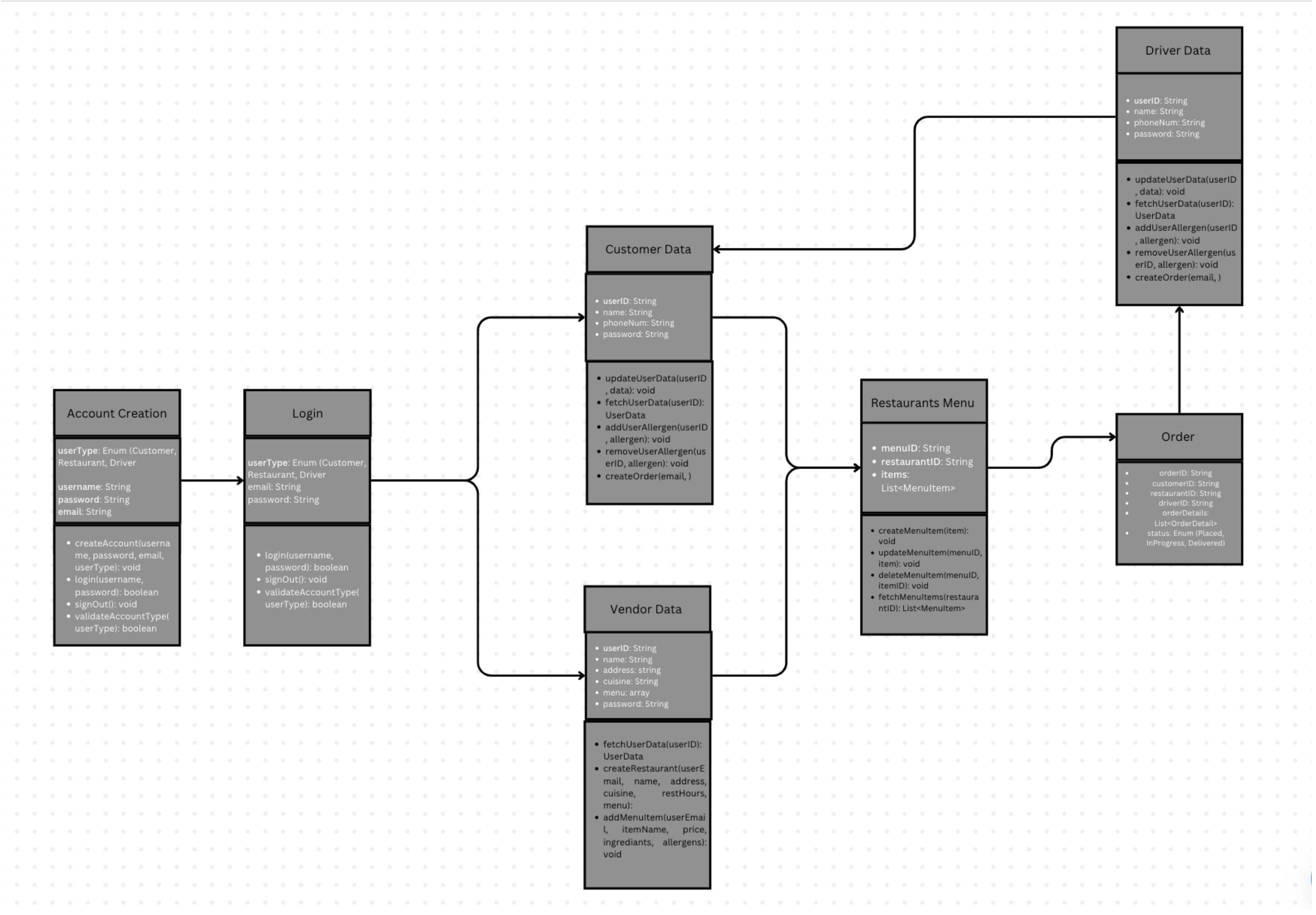
6.3. Software Design

6.3.1. State Machine Diagram: Customer (Kevin Ruiz)

6.3.2. State Machine Diagram: Restaurant (Lindsey Flores)

6.3.3. State Machine Diagram: Driver (Dylan Muroki)

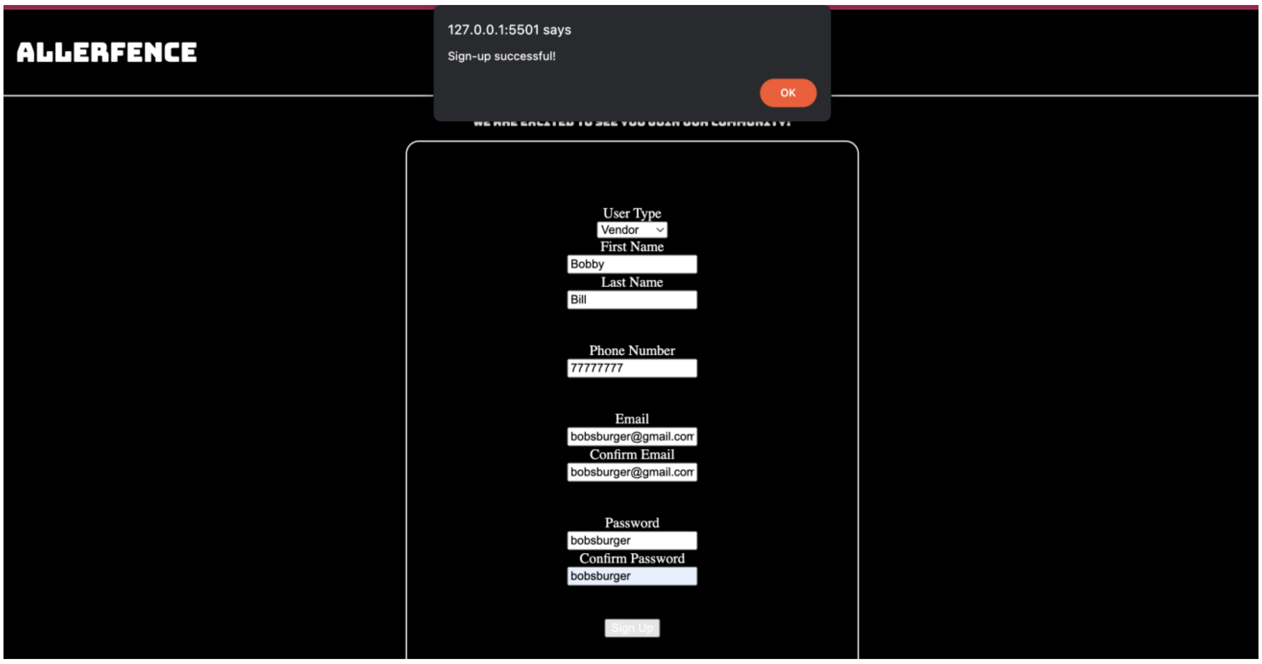
6.4. UML Class Diagram


Account Creation
userType: Enum (Customer, Restaurant, Driver
username: String
password: String
email: String
createAccount(username, password, email, userType): void
login(username, password): boolean
signOut(): void
validateAccountType(userType): boolean
Login
userType: Enum (Customer, Restaurant, Driver
email: String
password: String
login(username, password): boolean
signOut(): void
validateAccountType(userType): boolean
Customer Data
userID: String
name: String
phoneNum: String
password: String
updateUserData(userID, data): void
fetchUserData(userID): UserData
addUserAllergen(userID, allergen): void
removeUserAllergen(userID, allergen): void
createOrder(email,)
Vendor Data
userID: String
name: String
address: string
cuisine: String
menu: array
password: String
fetchUserData(userID): UserData
createRestaurant(userEmail, name, address, cuisine, restHours, menu):
addMenuItem(userEmail, itemName, price, ingrediants, allergens): void
Restaurants Menu
menuID: String
restaurantID: String
items: List<MenuItem>
createMenuItem(item): void
updateMenuItem(menuID, item): void
deleteMenuItem(menuID, itemID): void
fetchMenuItems(restaurantID): List<MenuItem>
Order
orderID: String
customerID: String
restaurantID: String
driverID: String
orderDetails: List<OrderDetail>
status: Enum (Placed, InProgress, Delivered)
Driver Data
userID: String
name: String
phoneNum: String
password: String
updateUserData(userID, data): void
fetchUserData(userID): UserData
addUserAllergen(userID, allergen): void
removeUserAllergen(userID, allergen): void
createOrder(email,)

7. Scenario

Let’s say the restaurant manager of small business LittleEats find our application. The manager can create an account, then be taken to the create your restaurant page where they will create their resturant with a manager email. Afterwards, the manager of LittleEats can go to their profile account and create their menu by adding menu items.
Lets say Ellie find our application creates an account with us and decided shes really craving a burger, but is allergic to gluten. Ellie creates an account with AllerFence, lists her allergies/dietary restriction. She then searches LittleEats (her favorite eating spot downtown) and finds that the burger she loves can be delivered to the comfort of her home without needing to call or contact the restaurant about her allergies. Her order of the burger comes through to the restaurant and restaurant accepts, once Ellie’s order is complete, the restaurant gives the clients location to their store delivery driver. The driver contacts the client that they are on their way via the chat and customer is able to track their order via a GPS. Client must pay in cash at this stage of the application.

Screenshots:
Restaurant is created:


ALLERFENCE

127.0.0.1:5501 says

Sign-up successful!

OK

WE ARE EXCITED TO SEE YOU JOIN OUR COMMUNITY!

User Type: Vendor

First Name: Bobby

Last Name: Bill

Phone Number: 77777777

Email: bobsburger@gmail.com

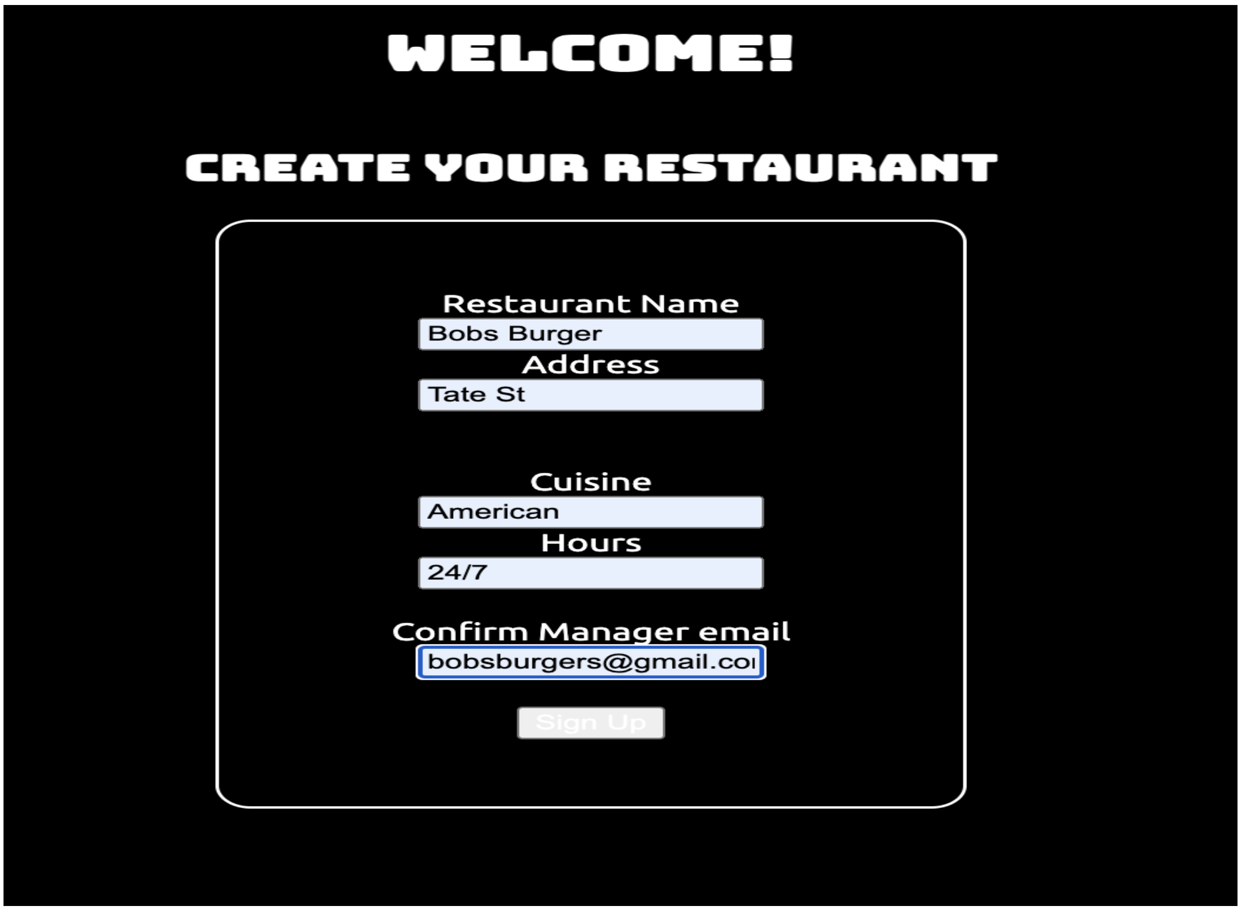
Confirm Email: bobsburger@gmail.com

Password: bobsburger

Confirm Password: bobsburger

Sign Up

Restaurant is created:


WELCOME!

CREATE YOUR RESTAURANT

Restaurant Name: Bobs Burger

Address: Tate St

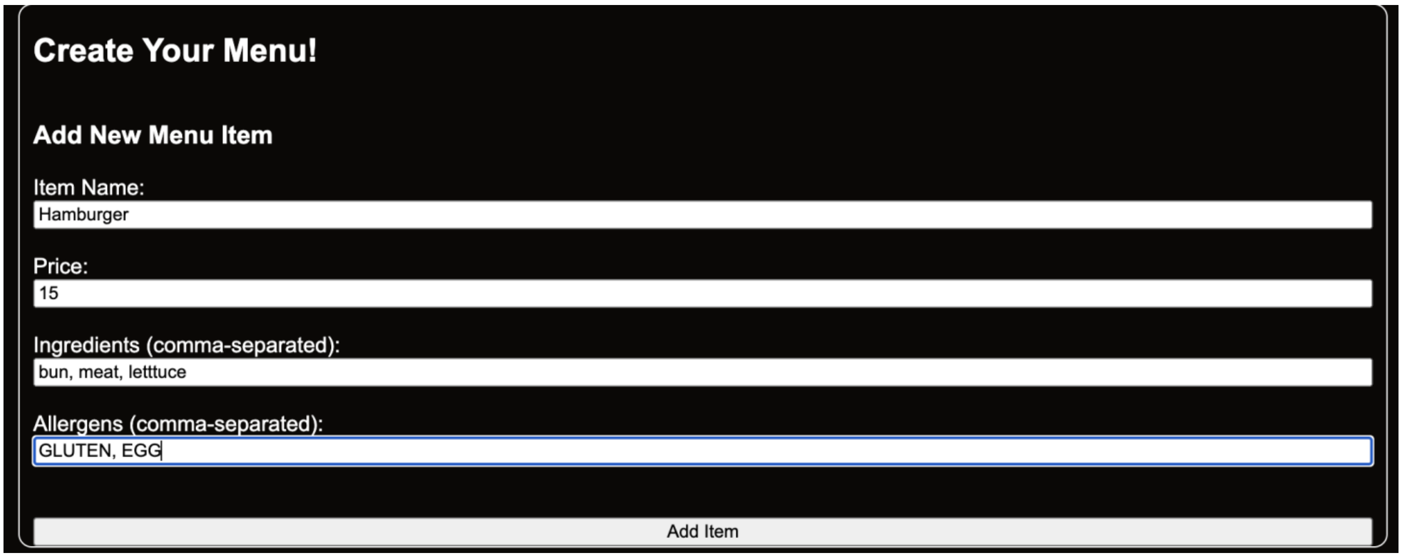
Cuisine: American

Hours: 24/7

Confirm Manager email: bobsburgers@gmail.co

Sign Up

Adds Menu:


Create Your Menu!

Add New Menu Item

Item Name: Hamburger

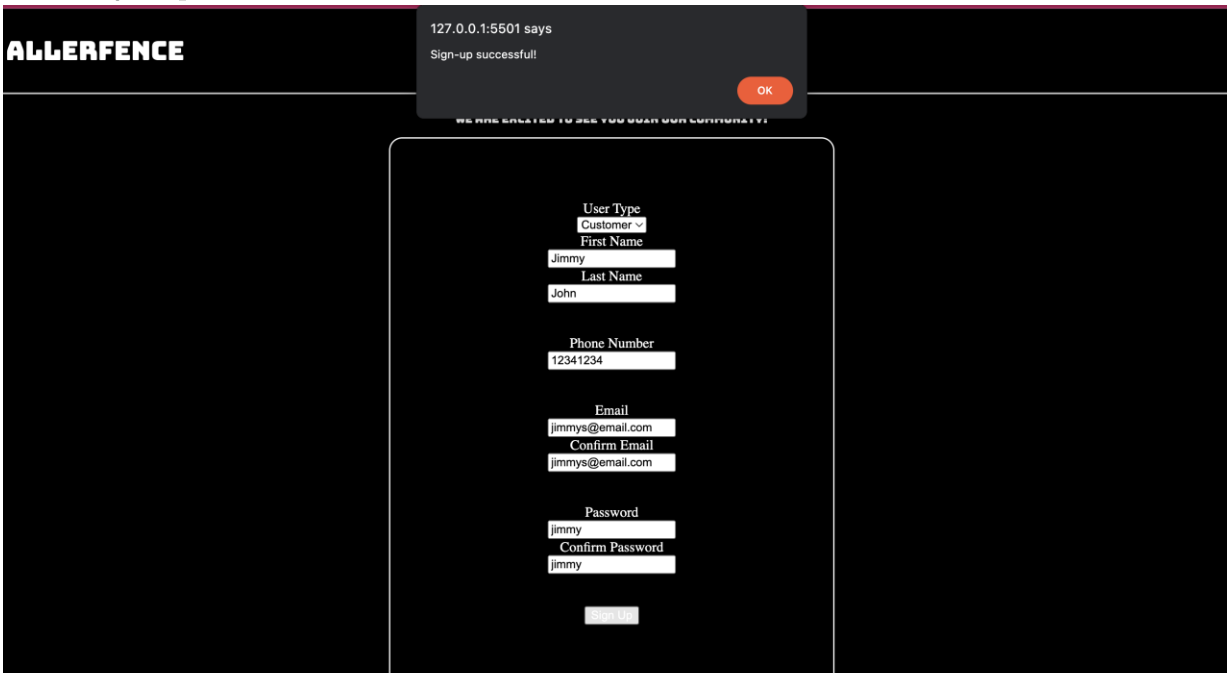
Price: 15

Ingredients (comma-separated): bun, meat, letttuce

Allergens (comma-separated): GLUTEN, EGG

Add Item

Client signs up:


ALLERFENCE

127.0.0.1:5501 says

Sign-up successful!

OK

User Type: Customer

First Name: Jimmy

Last Name: John

Phone Number: 12341234

Email: jimmys@email.com

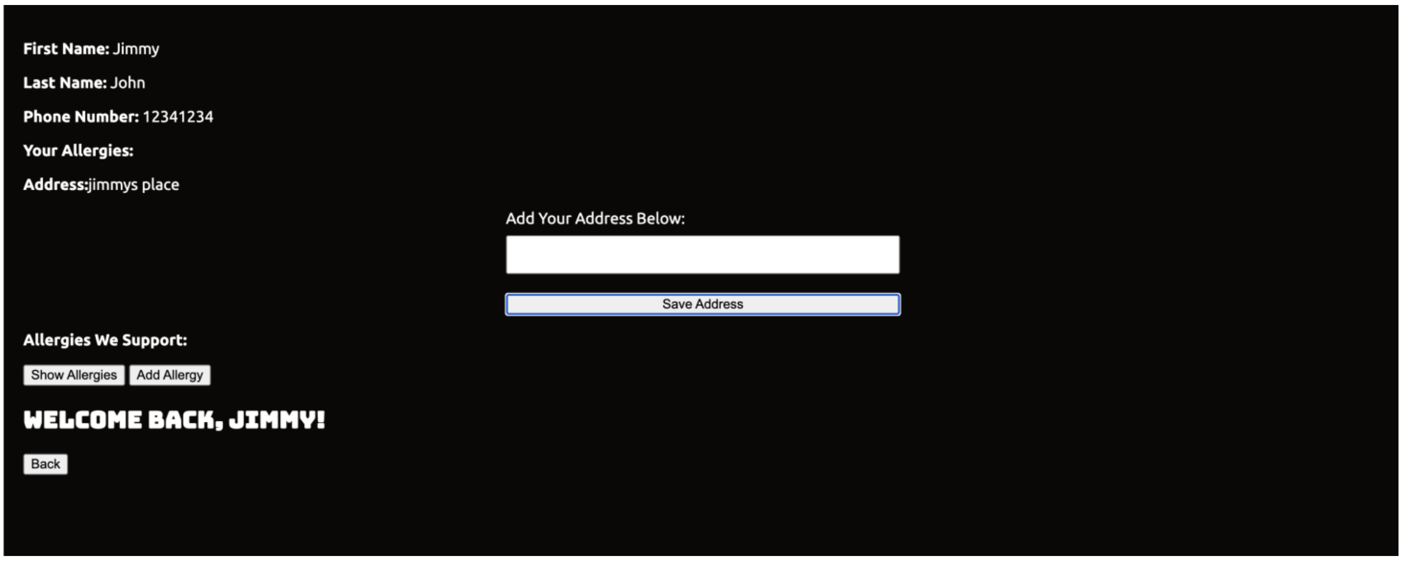
Confirm Email: jimmys@email.com

Password: jimmy

Confirm Password: jimmy

Sign Up

User adds address:


First Name: Jimmy

Last Name: John

Phone Number: 12341234

Your Allergies:

Address:jimmys place

Add Your Address Below:

Save Address

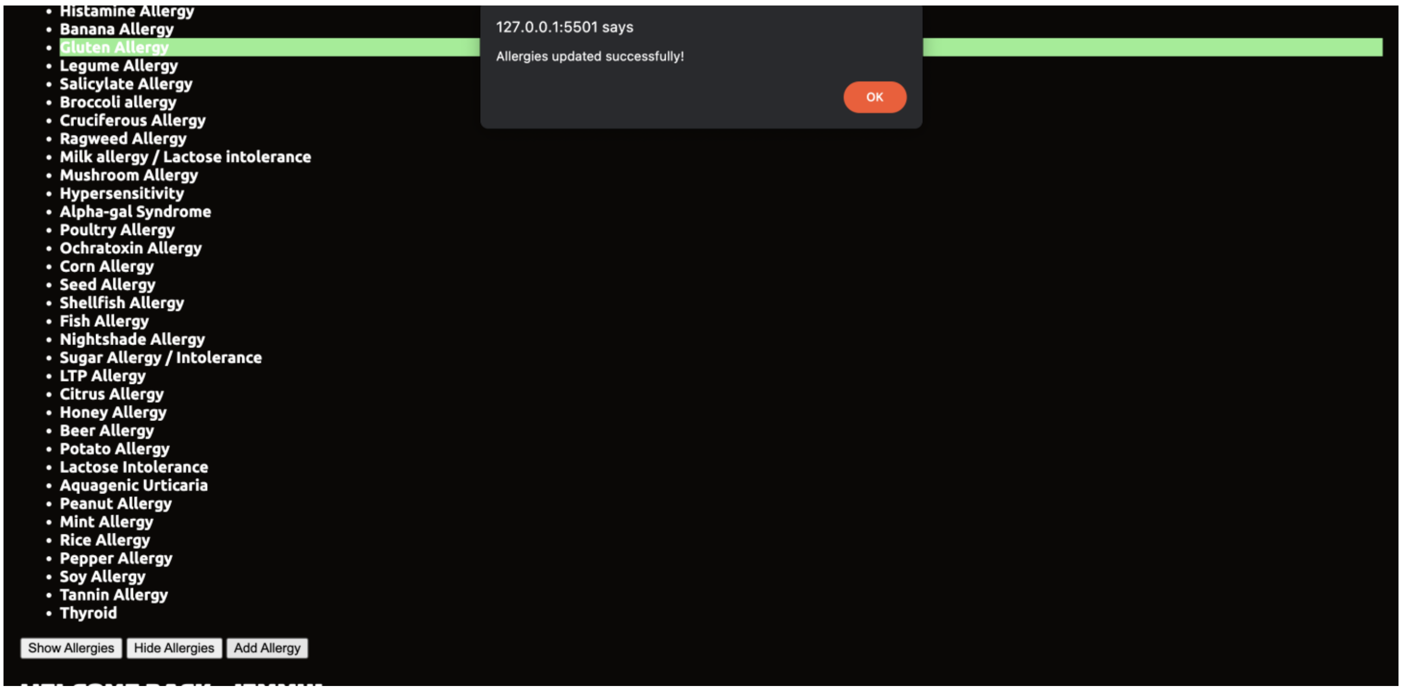
Allergies We Support:

Show Allergies Add Allergy

WELCOME BACK, JIMMY!

Back

User selects allergy:

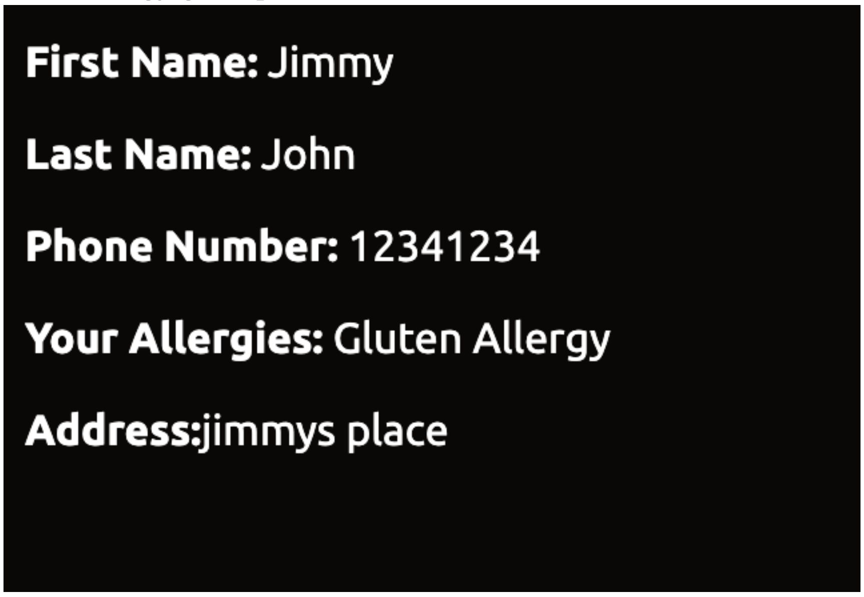


127.0.0.1:5501 says

Allergies updated successfully!

- Histamine Allergy
- Banana Allergy
- Gluten Allergy
- Legume Allergy
- Salicylate Allergy
- Broccoli allergy
- Cruciferous Allergy
- Ragweed Allergy
- Milk allergy / Lactose intolerance
- Mushroom Allergy
- Hypersensitivity
- Alpha-gal Syndrome
- Poultry Allergy
- Ochratoxin Allergy
- Corn Allergy
- Seed Allergy
- Shellfish Allergy
- Fish Allergy
- Nightshade Allergy
- Sugar Allergy / Intolerance
- LTP Allergy
- Citrus Allergy
- Honey Allergy
- Beer Allergy
- Potato Allergy
- Lactose Intolerance
- Aquagenic Urticaria
- Peanut Allergy
- Mint Allergy
- Rice Allergy
- Pepper Allergy
- Soy Allergy
- Tannin Allergy
- Thyroid

Users allergy gets updated:



First Name: Jimmy

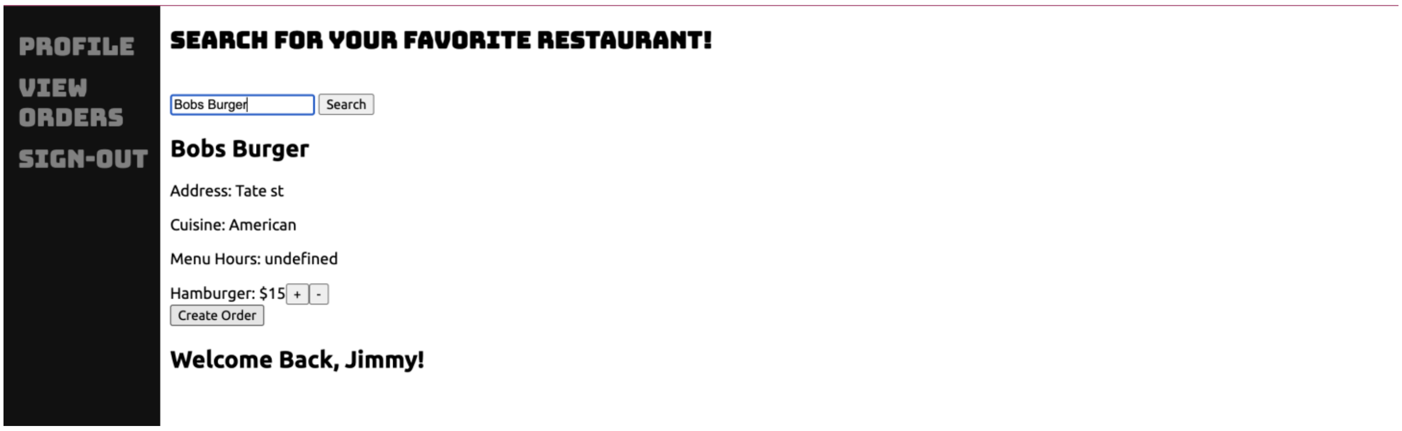
Last Name: John

Phone Number: 12341234

Your Allergies: Gluten Allergy

Address:jimmys place

User creates order:



SEARCH FOR YOUR FAVORITE RESTAURANT!

Bobs Burger

Bobs Burger

Address: Tate st

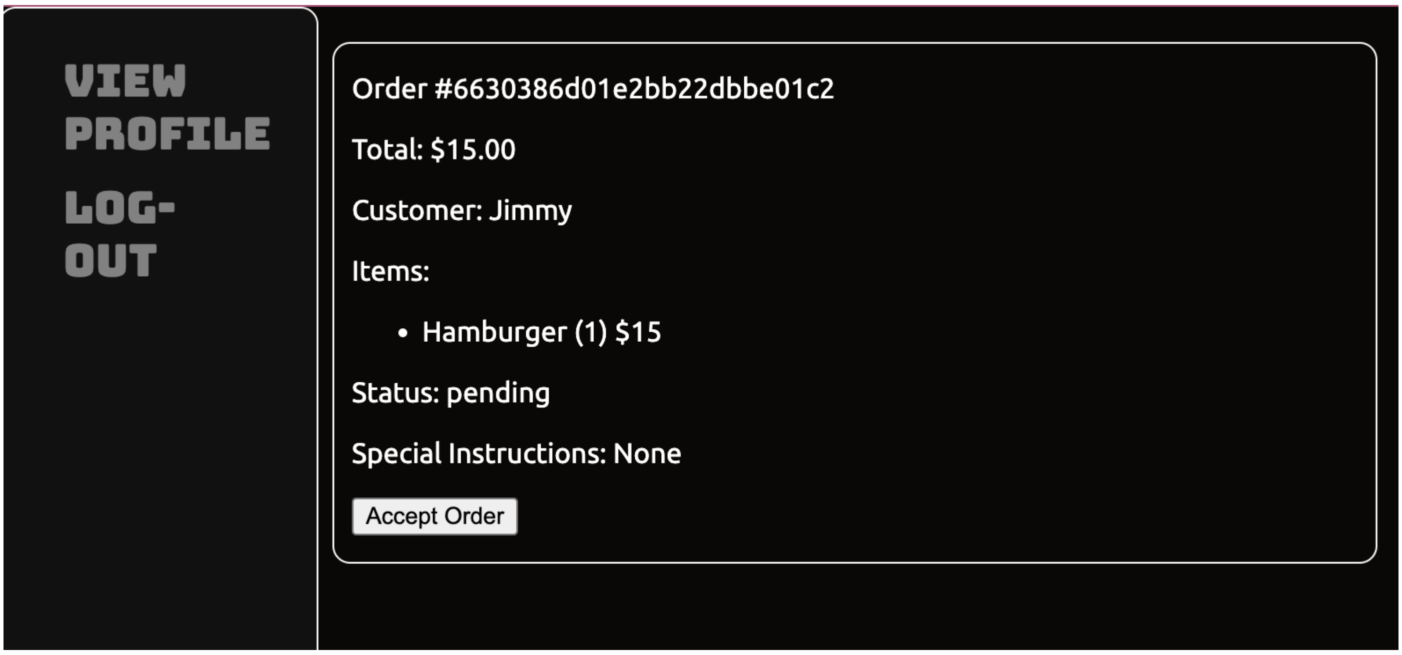
Cuisine: American

Menu Hours: undefined

Hamburger: $15

Welcome Back, Jimmy!

Restaurant gets order:



Order #6630386d01e2bb22dbbe01c2

Total: $15.00

Customer: Jimmy

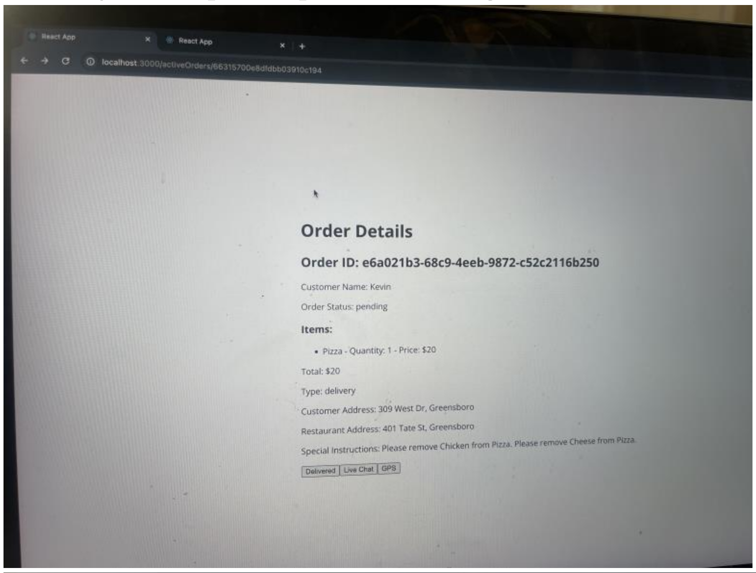
Items:

- Hamburger (1) $15

Status: pending

Special Instructions: None

Delivery driver picks up order and begins to communicate to customer



Order Details

Order ID: e6a021b3-68c9-4eeb-9872-c52c2116b250

Customer Name: Kevin

Order Status: pending

Items:

- Pizza - Quantity: 1 - Price: $20

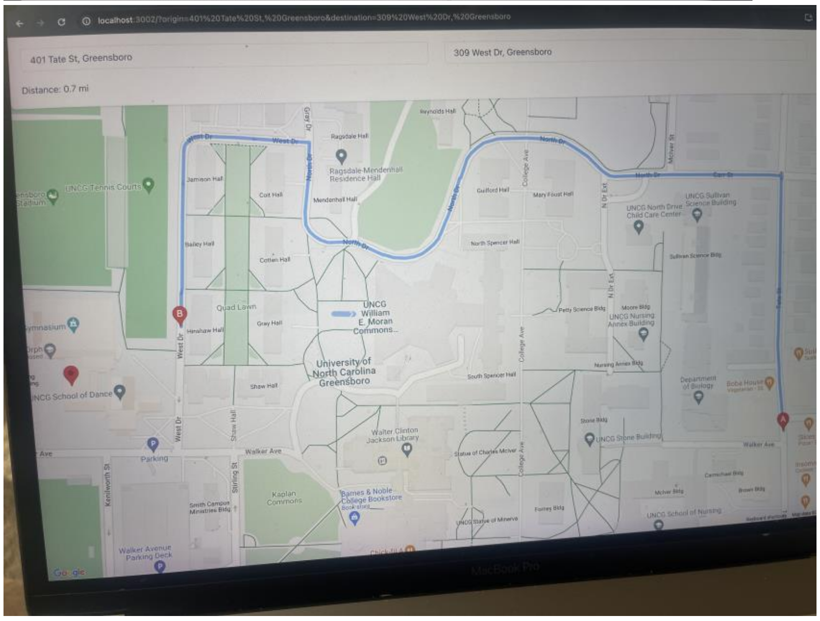
Total: $20

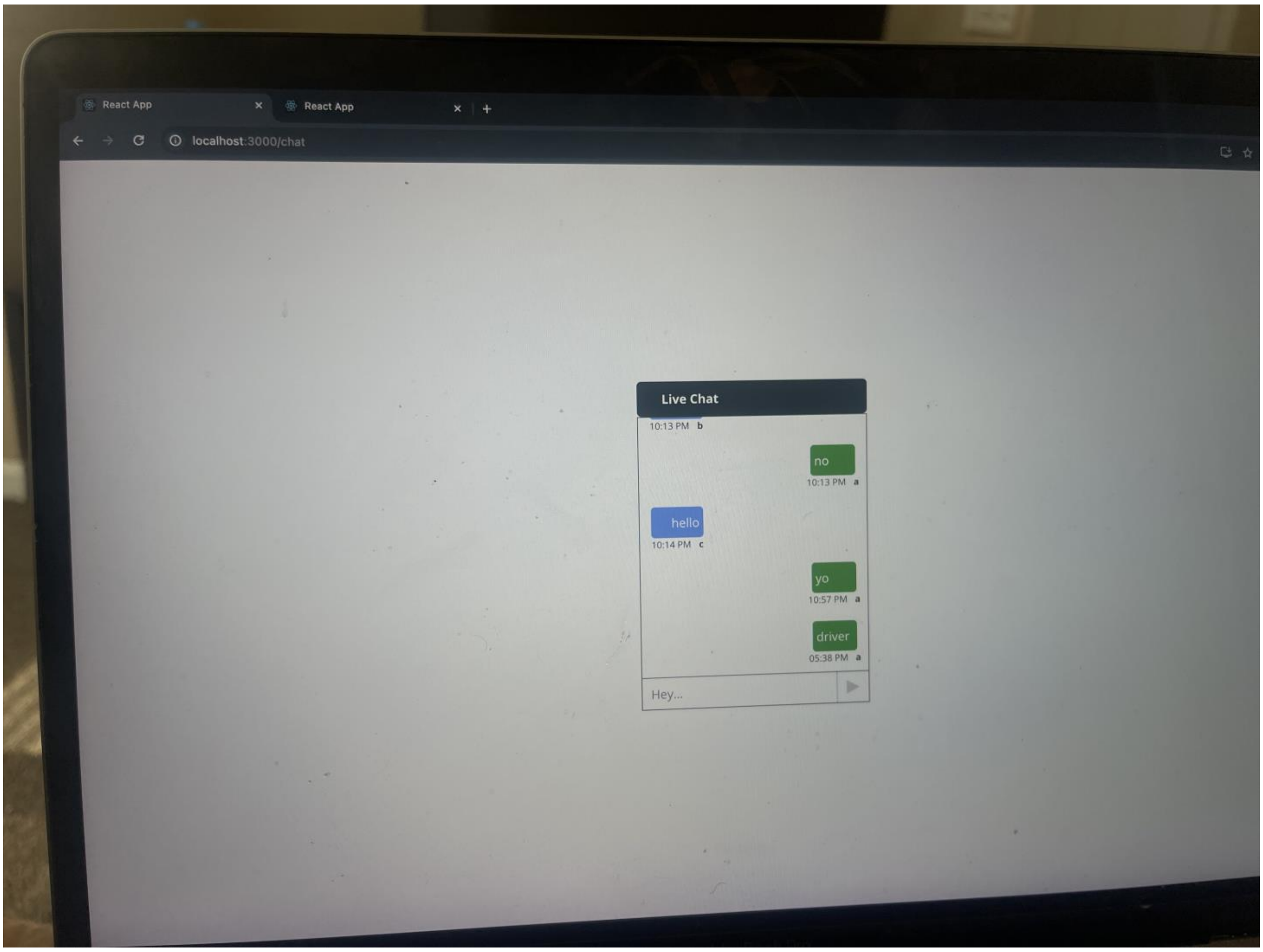
Type: delivery

Customer Address: 309 West Dr, Greensboro

Restaurant Address: 401 Tate St, Greensboro

Special Instructions: Please remove Chicken from Pizza. Please remove Cheese from Pizza.


localhost:3002/?origin=401%20Tate%20St,%20Greensboro&destination=309%20West%20Dr,%20Greensboro
401 Tate St, Greensboro
309 West Dr, Greensboro
Distance: 0.7 mi
West Dr
North Dr
Ragsdale-Mendenhall Residence Hall
UNCG Tennis Courts
UNCG William E. Moran Commons
University of North Carolina Greensboro
Walter Clinton Jackson Library
UNCG School of Dance
UNCG Sullivan Science Building
Walker Ave
College Ave



```
10:13 PM  b
                    no
                    10:13 PM  a
hello
10:14 PM  c
                    yo
                    10:57 PM  a
                    driver
                    05:38 PM  a
Hey...
```