Online Pub or Café House

Jianjie Yan
Software Engineering
Maynooth University
Maynooth, Ireland
jian.yan.2017@mumail.ie

Micheal Ruane
Software Engineering
Maynooth University
Maynooth, Ireland
michael.ruane.2017@mumail.ie

Ziyan Jin
Software Engineering
Maynooth University
Maynooth, Ireland
ziyan.jin.2020@mumail.ie

***Abstract*—The Covid-19 pandemic has disrupted many aspects of life, including the way people get access food. An unprecedented number of restaurants have shuttered across the County Kildare, Ireland. Many restaurants have been closed for dine-in services. Under this situation, this web application called Online Pub or Café is developed to help people to enjoy food at home, meanwhile, the business of restaurants can be boosted. Our application offers a platform, which helps people who are living in Kildare can order deliveries from pub/café nearby.**

Keywords—web application, delivery，covid-19

I. INTRODUCTION

The Lockdown by the covid-19 pandemic has influenced peoples life significantly. People have not enjoyed food from local pubs and cafes for a long time. This web application is an online pub or café house for the County Kildare. Specifically, we developed a website that allows individuals to order takeaway food and drinks from their local pub/café. Online platform offers the freedom to make an order for delivery without getting in touch with other people. Customers are able to order food from nearest restaurants through this system across County Kildare. They can select time slots when they prefer to get delivery.

II. TECHONOLOGY

During the development of this web application, several technologies have been utilized as below:

1. Front-end: HTML5, CSS3, Bootstrap, JavaScript (including ECMAScript 6 syntax) and jQuery.
2. Back-end: Node.js, Mongoose, MongoDB and Express.

The front-end focuses on static data which can be viewed by customers. We used HTML to create basic web pages, adding CSS files to style the pages, applying JavaScript to program the behavior of web pages (JavaScript Tutorial, 2021). The back-end, which is not directly accessed by the user, typically responsible for storing and manipulating data. MongoDB, a NoSQL database, is used here because of its robust schema architecture and JSON format storage. Node.js, a JavaScript framework which helped us build our webserver. All the development dependencies were installed using the node package manager.

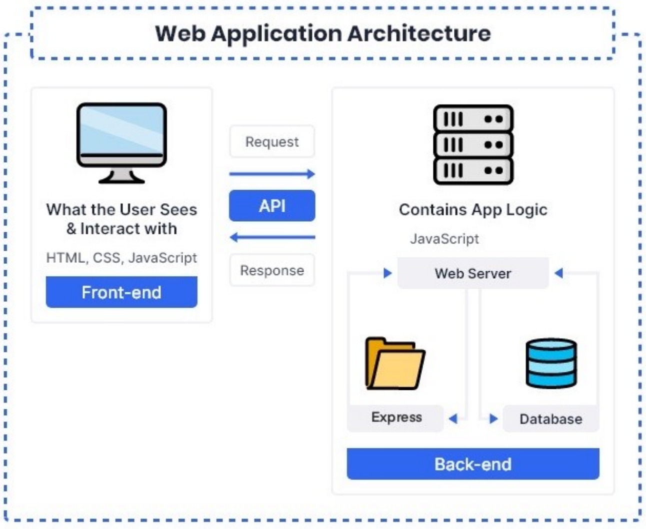


Fig1: Basic architecture

III. PROPOSED SYSTEM

The web application, Online pub or café house, is aiming to provide a system of online booking that includes making order about the food and the food delivery. People can register an account and select a region in Kildare. After they have logged into the web application successfully, they will have options to choose restaurants, then menus will be shown, and they may start to order the food they want. Items in menu can be sorted based on price. After food has been added to the cart, customers can check the shopping cart, select delivery time slot and fill in the delivery address. Then clicking the button of "order now" to finish order. The button of "Logout" will bring back users to the login page.

The proposed system consists of following modules.

Module 1: Login Module:

The login module allows customers to login the system and order delivery. Each customer is registered with a unique e-mail address.

Module 2: Regsitrastion Module:

The registration module allows customers to create accounts with their email address, phone number and other details to gain access to services. In this section, people need to choose location, therefore to check pub/café across their region.

Module 3: Fetch Location Based Business Module:

Fetch business location module fetches all the pub/café according to customer's selected region.

Module 4: Fecth Business Menu Module:

Fetch business menu module shows items of selected restaurant. In this module users can view, and sort items based on price or name.

Module 5: Fetch Business Delivery Module:

Fetch business delivery module fetches present day's available delivery time of the business and allows customers to select delivery time slot, then the selected time slot is set to be reserved for the customer and the business.

Module 6: Add Cart Items Module:

This module allows user to add items in their cart. The selected cart items can then be submitted as an order after providing the delivery address and selecting the preferred delivery time-slot.

IV. DESIGN ARCHITECTURE

In this section, more details about this web application's architecture will be discussed, including the frontend, the backend, and overall processes. We developed the application in two separate projects front-end and back-end.

Software Architecture

This web application is designed using several functional connected folders. We will talk more details about how this application works based on these folders below.

The customer-frontend folder involves user view pages, which consists by data from Rest API in JSON format. There are three HTML files, index, menus and welcomePage, where serves static web page contents. The CSS folder, which is used to style the HTML document. CSS describes how HTML elements should be displayed. In this CSS folder, toastr file and welcome file are used to style welcomePage and menus page, while style file is used to style index page. Js folder is used to program the behavior of webpages. Img folder contains all the pictures used in this application.

The customer-backend folder index.js is the entry point to the server. In this file, MongoDB is set up through mongoose. Model folder contains all the database schemas in this system. Routes folder includes two JavaScript files, customer and café, which consists of Express code that associates an HTTP verb (GET, POST, PUT, DELETE, etc.), a function that is called to handle that pattern. The last two files are Procfile and .gitignore. Procfile specifies the commands that Heroku needs to run the application on its server. The .gitignore file contains a series of subfolders and files to ensure that certain files not tracked by git and not published on BitBucket .

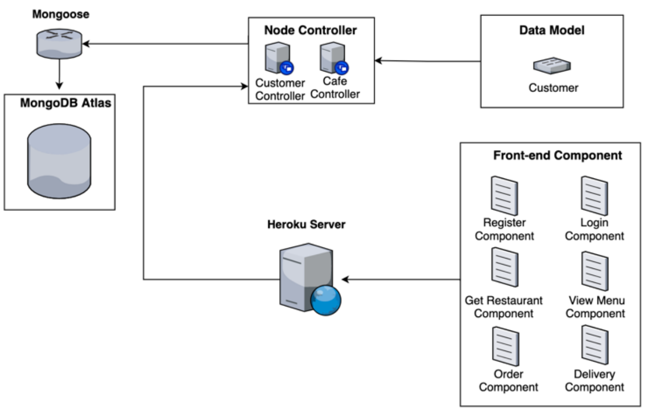
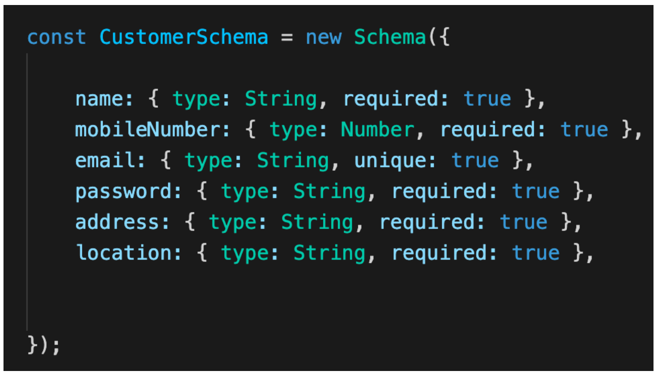


Fig.2. the flow of data

Models

For the development of our API we required only one schema. The CustomerSchema, which is our primary schema.



```
const CustomerSchema = new Schema({

    name: { type: String, required: true },
    mobileNumber: { type: Number, required: true },
    email: { type: String, unique: true },
    password: { type: String, required: true },
    address: { type: String, required: true },
    location: { type: String, required: true },


});
```

Fig.3. Customer Schema

Routes

After understanding our functionalities, we would like to add them in our web application, for that we need to create routes (URL handlers) to handle an HTTP request/response. The routes folder includes two files, customer.js and café.js.

In customer file, we have register and login functions. When a customer signing in, the server will first check if the email is in use, if not then the password will be encrypted using bcrypt package. Once successfully registered, message 'Customer Registered!!' will be thrown by server in response body, otherwise the server will throw an error 'Email already registered'. Same goes for the login part, when a customer is logging in, if account does not exist, the server will send 'No account registered with this email' otherwise, users will see 'Logged In'.

In café.js, we have actions called getNearby and getCurrentDayDeliveryTime. In action getNearby, we used 'axios', which is a client HTTP API based on the XMLHttpRequest interface provided by browsers. Using this library, we can get all restaurants in one area from the owner side of this application. While signing up customer has already chosen his location. Based on their location, a

POST request is sent to the URL provided by owner side and in its response body we get all the restaurants for a particular location. The getCurrentDayDeliveryTime works in the same way, it enables customers to select delivery time slot for present day. If the delivery time is not present in the database, an alert will be thrown 'No delivery on ' + dayName + ' available'.

Components

Using components help in separating the large project into smaller components and reduce the complexity of the project. We separated this project into five components shown in Fig.4. The signup component requires customers to select location, which will be used to fetch business in the particular area. Other components allow customers to select business in the area, fetch the business menu and sort menu items based on the price and the name, fetch delivery time slots, add item to shopping cart and make order. When a customer selects a delivery slot, the slot will be reserved, so that other customers cannot use this time slot.

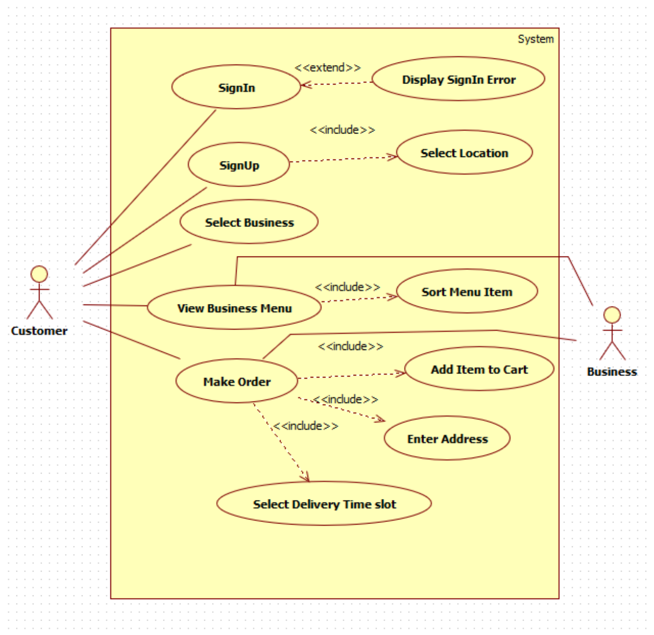


Fig.4. Use case diagram

V. KEY IMPLEMENTATIONS

In this part, we highlight some key implementations of our web application.

A. Display business in selected region

This implementation displays all the businesses in selected region. We use card to represent all the businesses with their name, location, email address and phone number. A ‘VIEW MENU’ button is placed on the same card with the business, so that customers can select businesses as they prefer.

B. Display and Sort Menu

This implementation allows customers to sort the menu items according to the name and price. An ‘ADD TO CART’ button is displayed under each item. Therefore, when customers click on the button, the item will be added to cart.

C. Shopping Cart

There is a shopping cart icon displayed on the top right of the page. When customers click on the icon, shopping cart section will pop up with the delivery time slots and items selected display on the section. There is a delivery address input section and a button ‘ORDER NOW’ underneath. Once the button is clicked, the data will be sent to database.

D. Delivery

Delivery slots are displayed in the shopping cart section. Each delivery time slot represents 15 minutes from 12:00 to 20:00. There is a switch button after each time slot, so customers can switch on the button to choose the delivery time.

REFERENCES

https://docs.mongodb.com/drivers/node/current/J.

https://developer.mozilla.org/en-US/docs/Learn/Server-

https://www.irjet.net/archives/V7/i4/IRJETV7I41009.pdfhttp://iaetsdjaras.org/gallery/14-jaras-327-december.pdf

https://www.w3schools.com/js/DEFAULT.asp

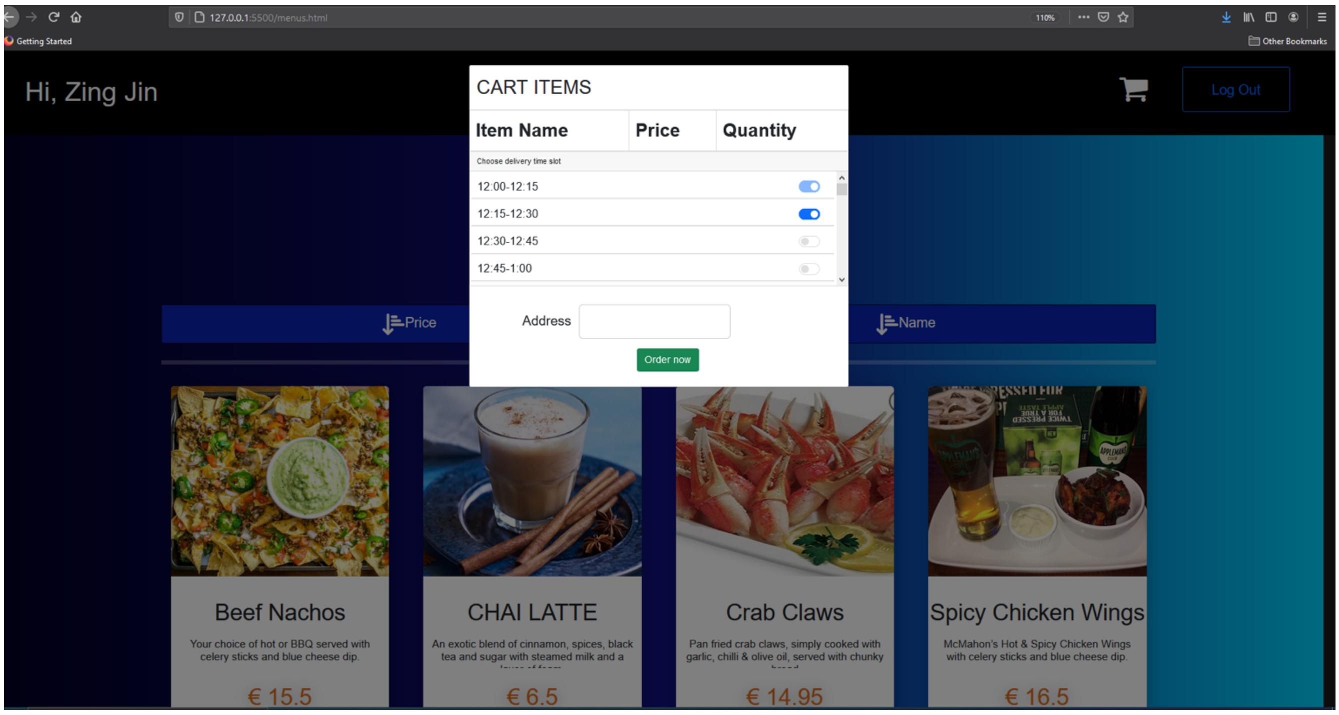
https://www.npmjs.com/package/node-fetch

https://www.w3schools.com/js/DEFAULT.asp

https://getbootstrap.com/docs/5.0/getting-started/introduction/

https://blog.logrocket.com/how-to-make-http-requests-like-a-pro-with-axios/

Appendix



Hi, Zing Jin

CART ITEMS

Item Name	Price	Quantity

Choose delivery time slot

12:00-12:15	✓
12:15-12:30	✓
12:30-12:45	
12:45-1:00	

Address

Beef Nachos

Your choice of hot or BBQ served with celery sticks and blue cheese dip.

€ 15.5

CHAI LATTE

An exotic blend of cinnamon, spices, black tea and sugar with steamed milk and a

€ 6.5

Crab Claws

Pan fried crab claws, simply cooked with garlic, chilli & olive oil, served with chunky

€ 14.95

Spicy Chicken Wings

McMahon's Hot & Spicy Chicken Wings with celery sticks and blue cheese dip.

€ 16.5