Project Title: Online Food Ordering Web Application

Team Members:

1. Muhammad Usman
2. Aravind Reddy
3. Ritwik Reddy
4. Damodar Reddy

The Story:

Motivation behind developing this application is to facilitate the people with this online application. During the Covid-19 time the e-commerce companies like amazon, boosted their business more than we think about can be and not only it facilitates the business owners to boost their business, also it facilitates the customers to get the thing by just clicking a few buttons. Our application is facilitating the customers in the same way, during and after the covid time. Let's understand the customers. We are targeting both restaurant owners and as well as food buyers. For restaurant owners our app will help them to boost their business by maximizing the reach of their hotel miles away to customers. Unlike Social media marketing, where the customers will only get to know about the business. Our application will not only marketize their business, also it will allow their target audience to buy the food from their restaurants. For food buyers, this will help allow them to get their favorite food by sitting in their home or in the school and office. In this way buyers will be saving their time and commute cost. This application is having the best features that will results in both sellers and buyers benefit.

The Data:

The data we used in our application is mostly string format data that is the name of customers and restaurants and their corresponding menus. All the restaurant data was collected from the google by visiting each restaurant website.

We used NoSQL database, so our data is non-relational data. To give the high-level overview of over data schema, we have attached the schema of one restaurant from our database.


```
{
    "id": "66330",
    "name": "Jimmy John's Sandwiches",
    "address": "9201 E State Rte 350, Raytown, MO 64133, USA",
    "cuisines": "Gourmet Sandwiches",
    "rating": "4.4",
    "reviews": "865",
    "feature_image": "https://img.cdn4dd.com/cdn-cgi/image/fit=cover,wi
    "thumbnail_image": "https://img.cdn4dd.com/cdn-cgi/image/fit=cover,
    "menu": [{
        "id": 663307,
        "name": "SPICY EAST COAST ITALIAN",
        "desc": "A tasty Italian sandwich, made with a double serving o
        "price": 12
    }, {
        "id": "899166",
        "name": "BEACH CLUB",
        "desc": "Made with tasty hand-sliced turkey breast, provolone c
        "price": "20"
    }]
```

Application Screen Shots.

1. Login/Registration

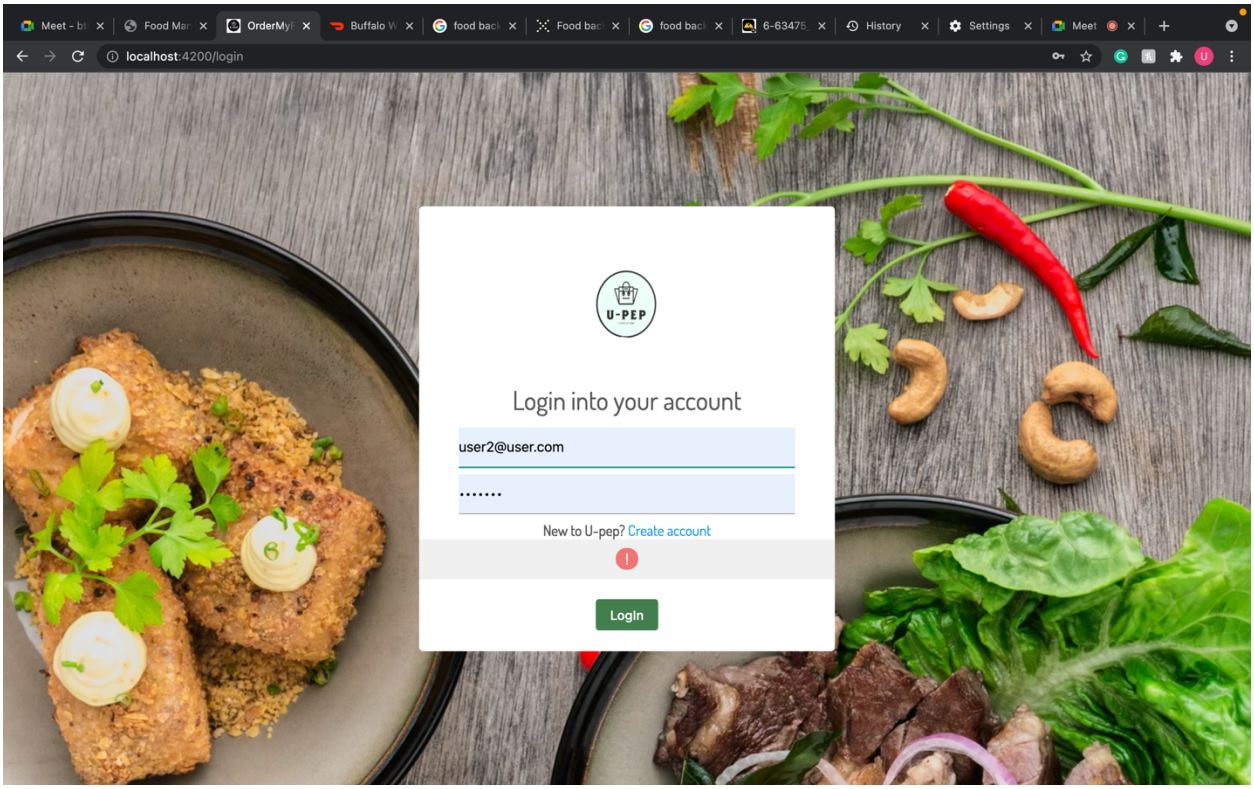


Figure 1: User Login/Registration

2. Home Page

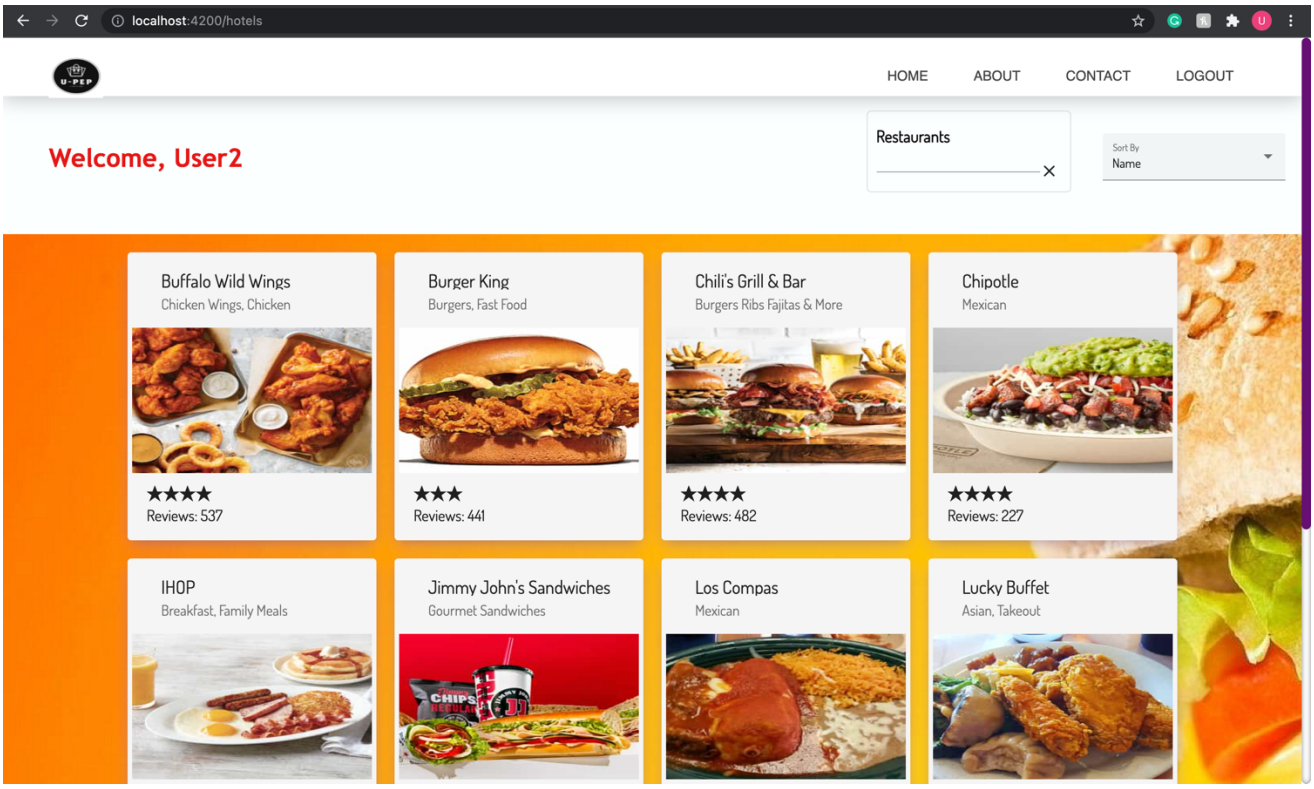


Figure 2: Available Hotels

3. Menu Page

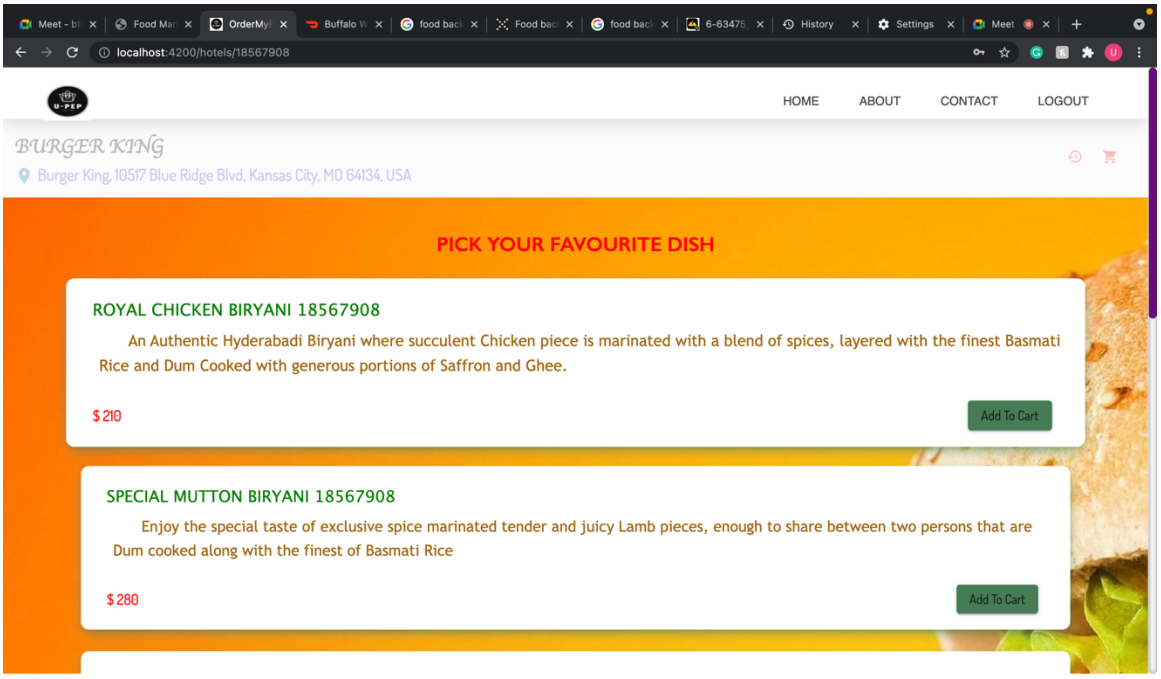


Figure 3: List of menus

4. Cart

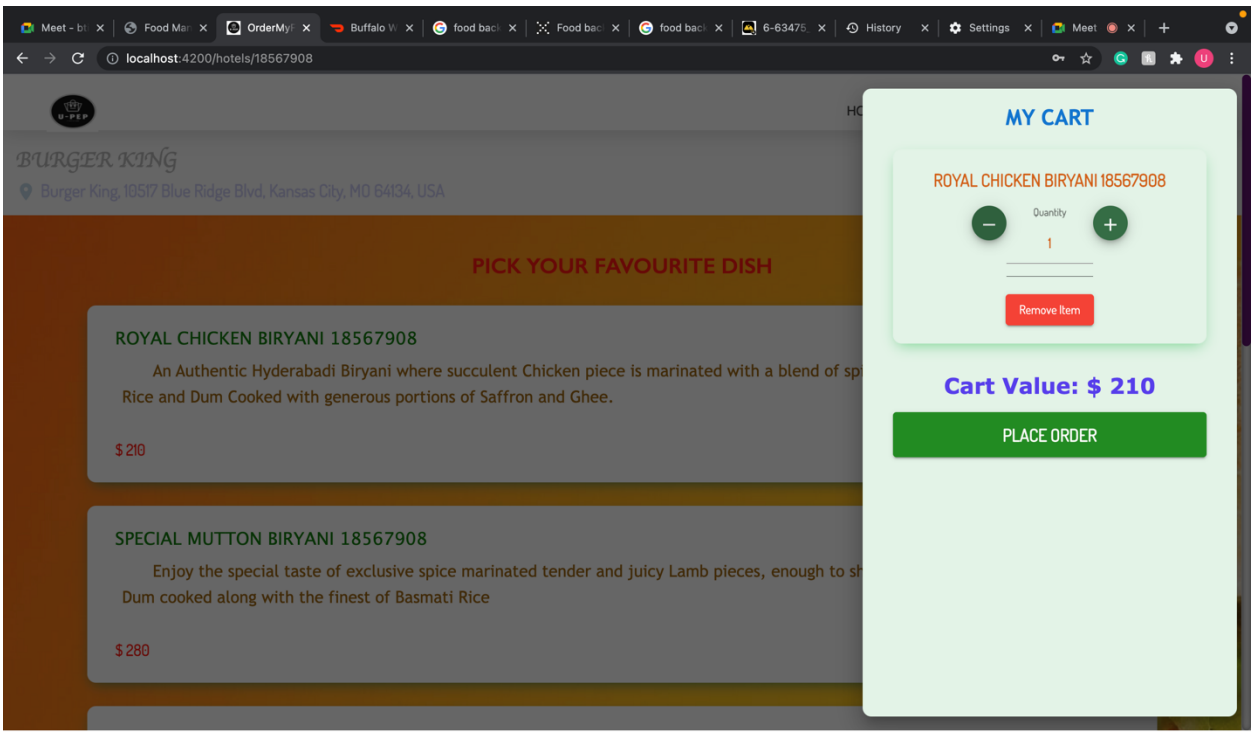


Figure 4: Cart

5. Chat Bot

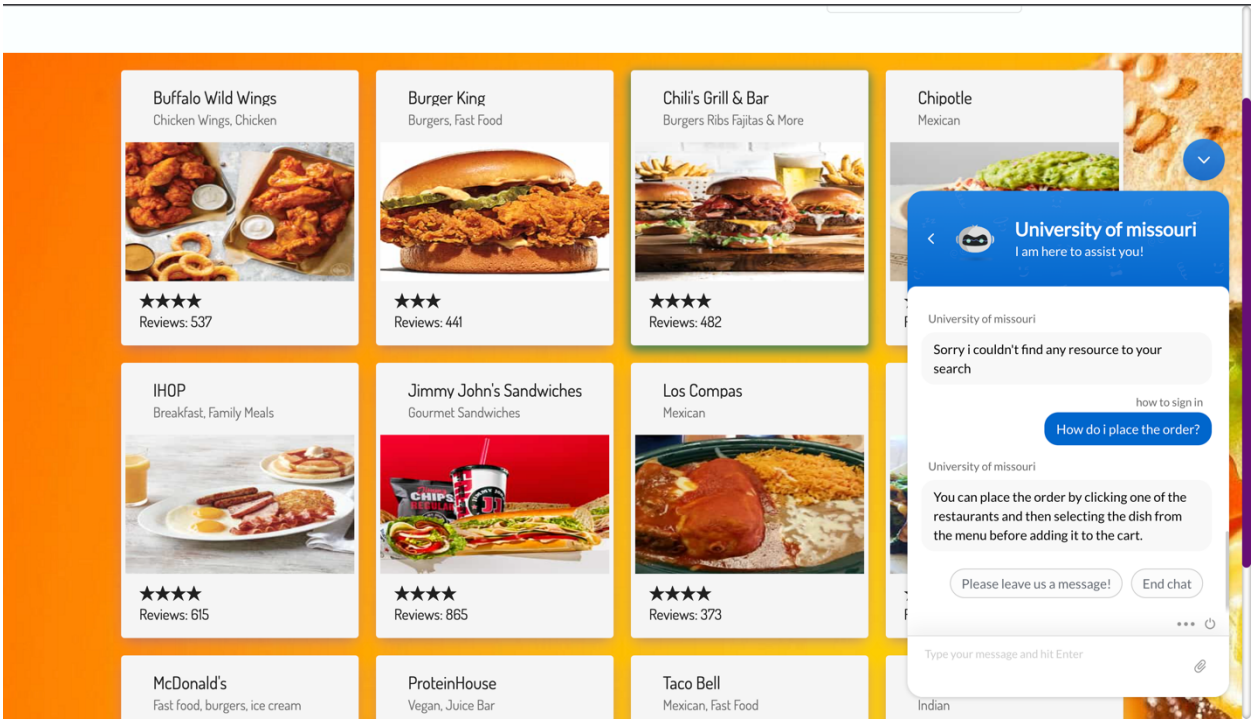


Figure 5: Chatbot

6. Order Confirmation

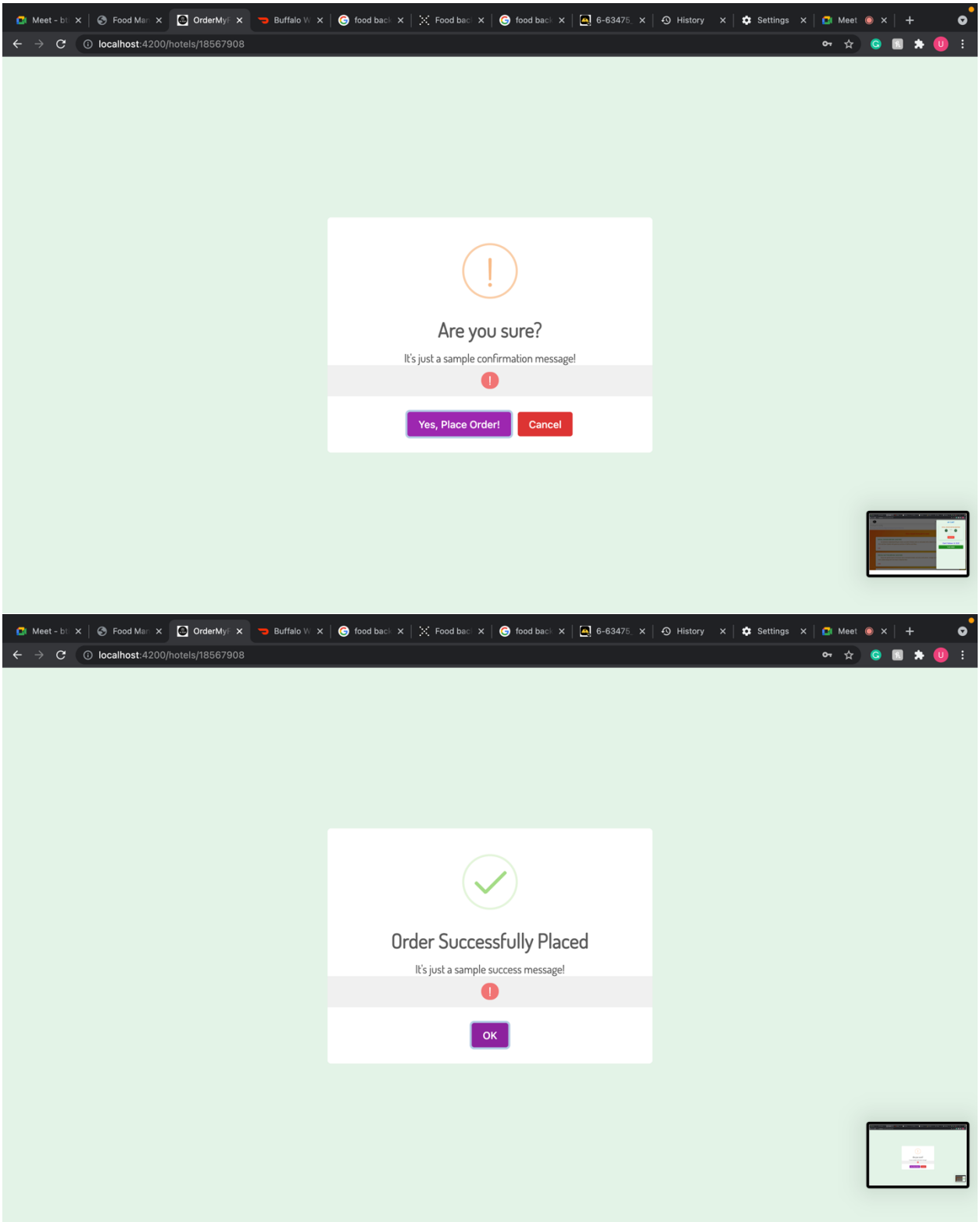


Figure 6: Order Confirmation

7. Admin Panel

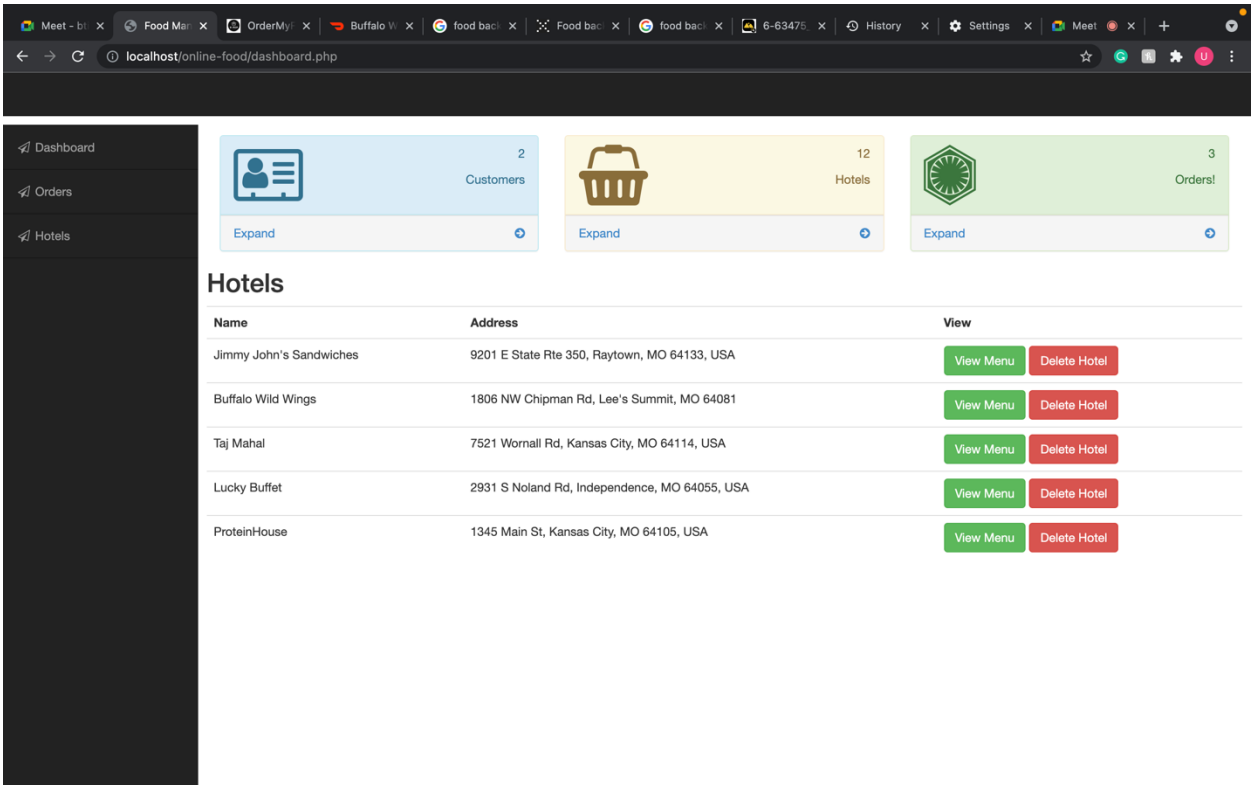


Figure 7: Admin Panel Dashboard

8. Order Management Page

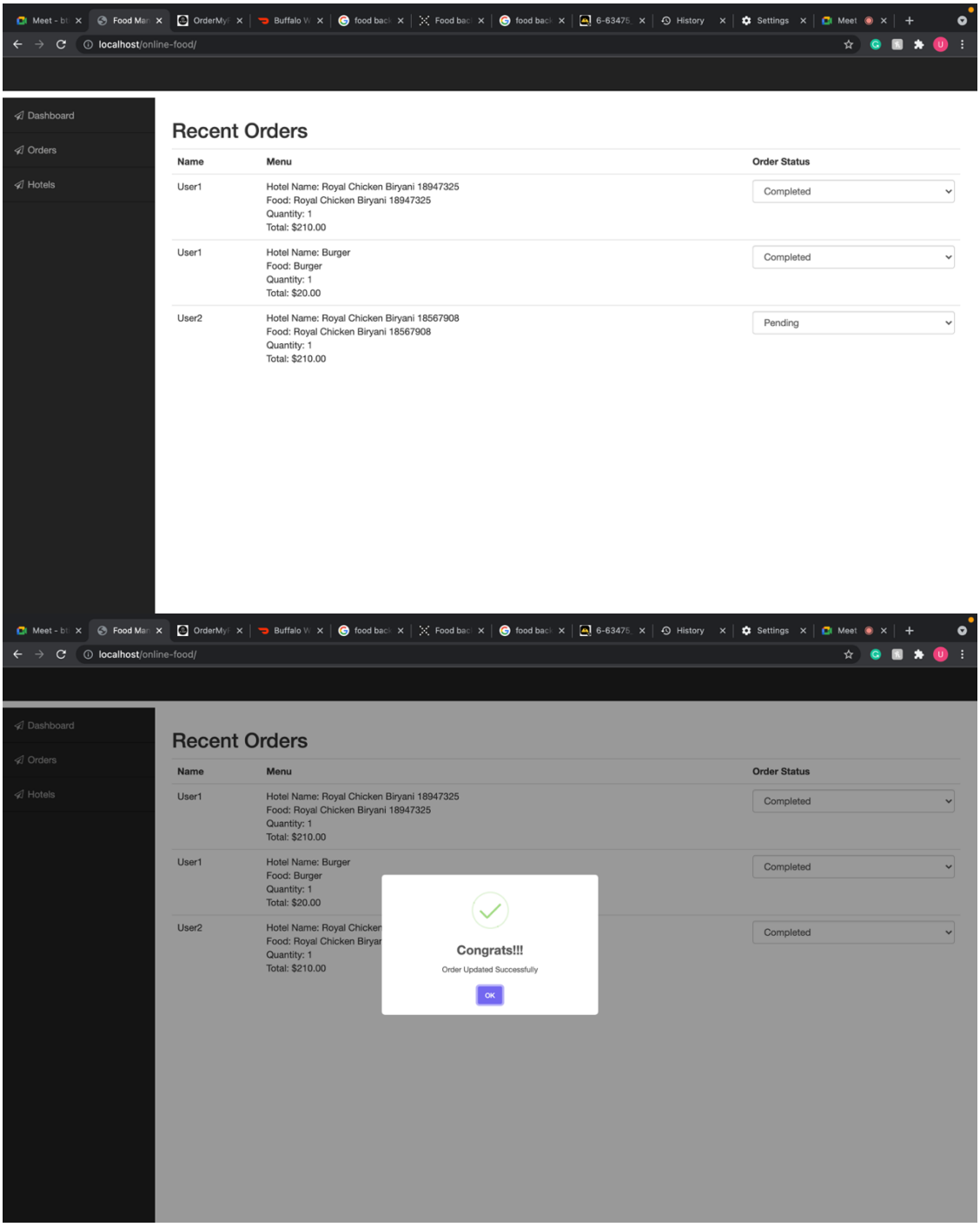


Figure 8: Order Accept/Reject

Description: From this screen Admin can manage the orders from the various users. Managing order includes, accepting and rejecting the incoming orders.

9. Hotels Managing Page

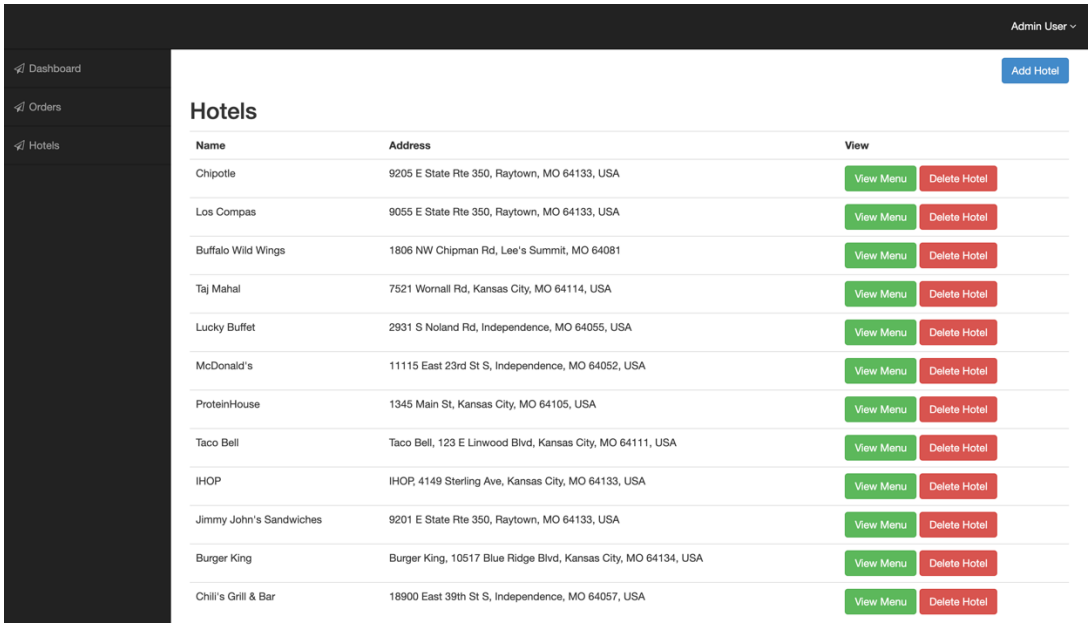


Figure 9: List of Hotels

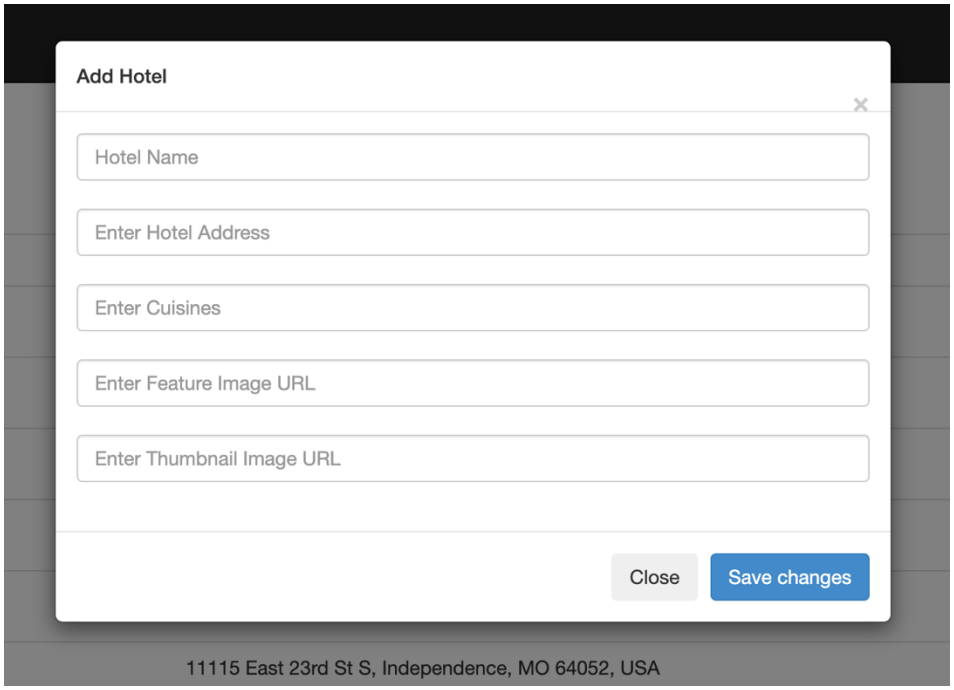


Figure 10: Add Hotel

Description: Admin can view and add the hotel by clicking add new button.

10. Menu Managing Page

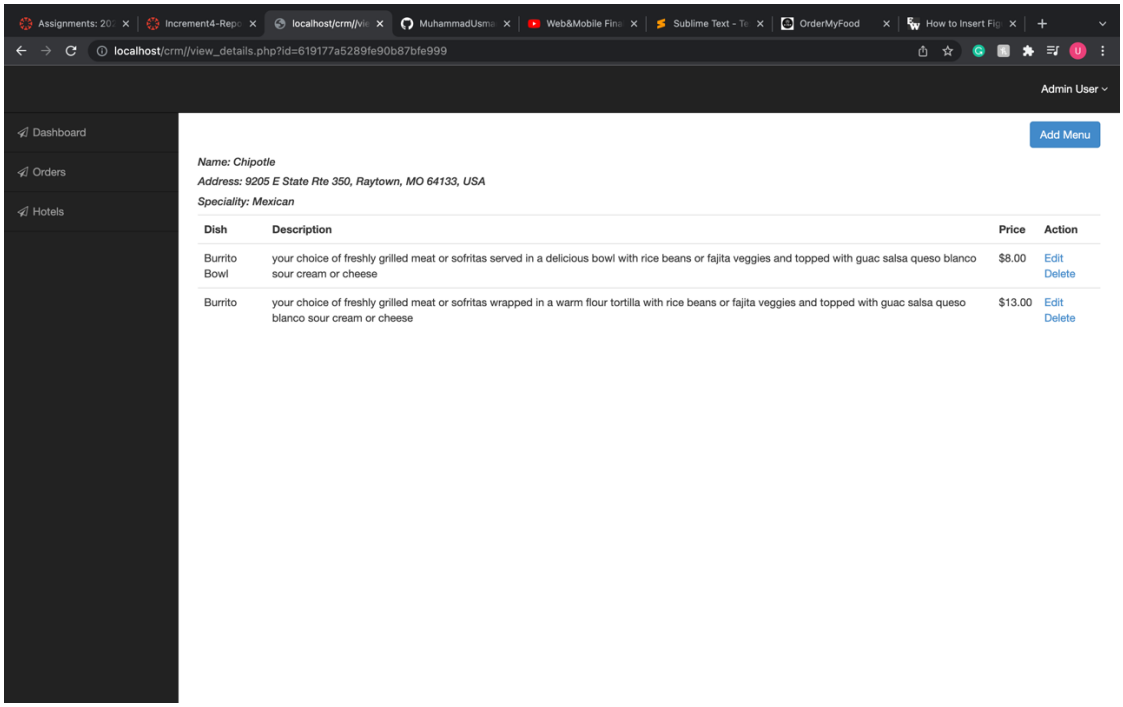


Figure 11: Menu List

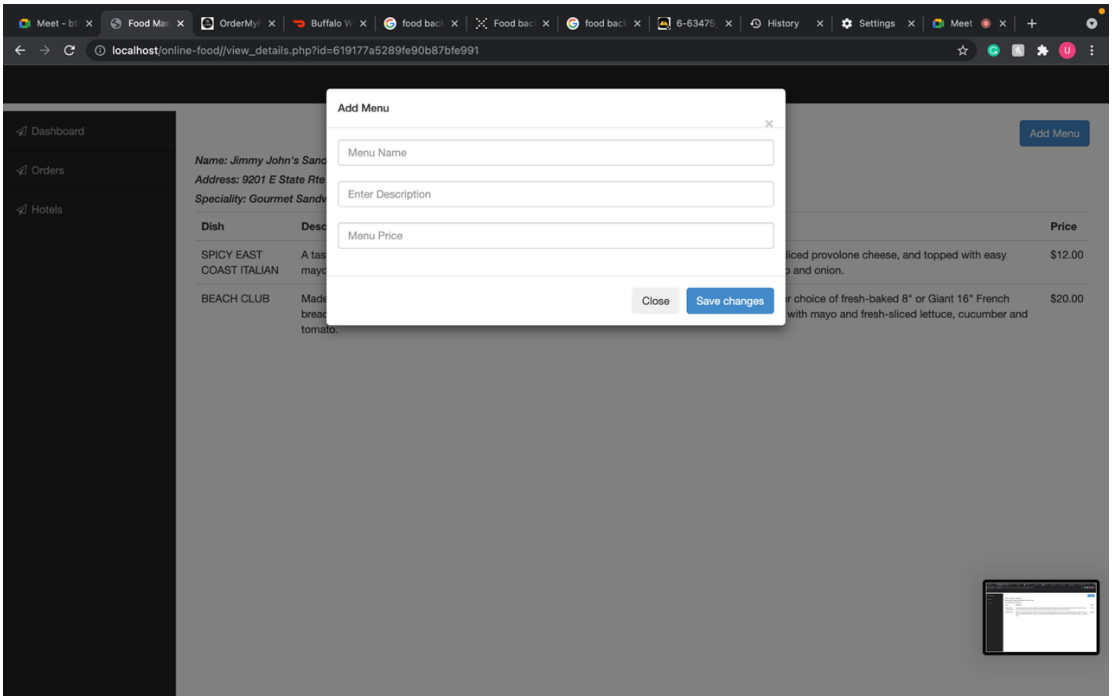


Figure 12: Add menu

Description: Admin can view and add the menu for the specific hotel.

Improvement:

In this increment, we mainly fixed some bugs in the admin panel and add new functionality that were not present in the previous sprint. We have discussed the improvement parts below.

1. The total of the order has been fixed in this sprint by using the correct sum formula.
2. In this sprint we have add the functionality, where user can add the restaurants using admin panel. In the previous sprint user must insert the new restaurant detail by visiting the database.
3. User can now delete the menu from the admin panel, whereas in the previous sprint user cannot delete the menu from the admin panel and if he wanted to delete it, he must need to visit the database to do that.
4. We have also added the chat bot into our website.
5. Refactoring of the whole code and we also added appropriate comments.

Code Snippets:

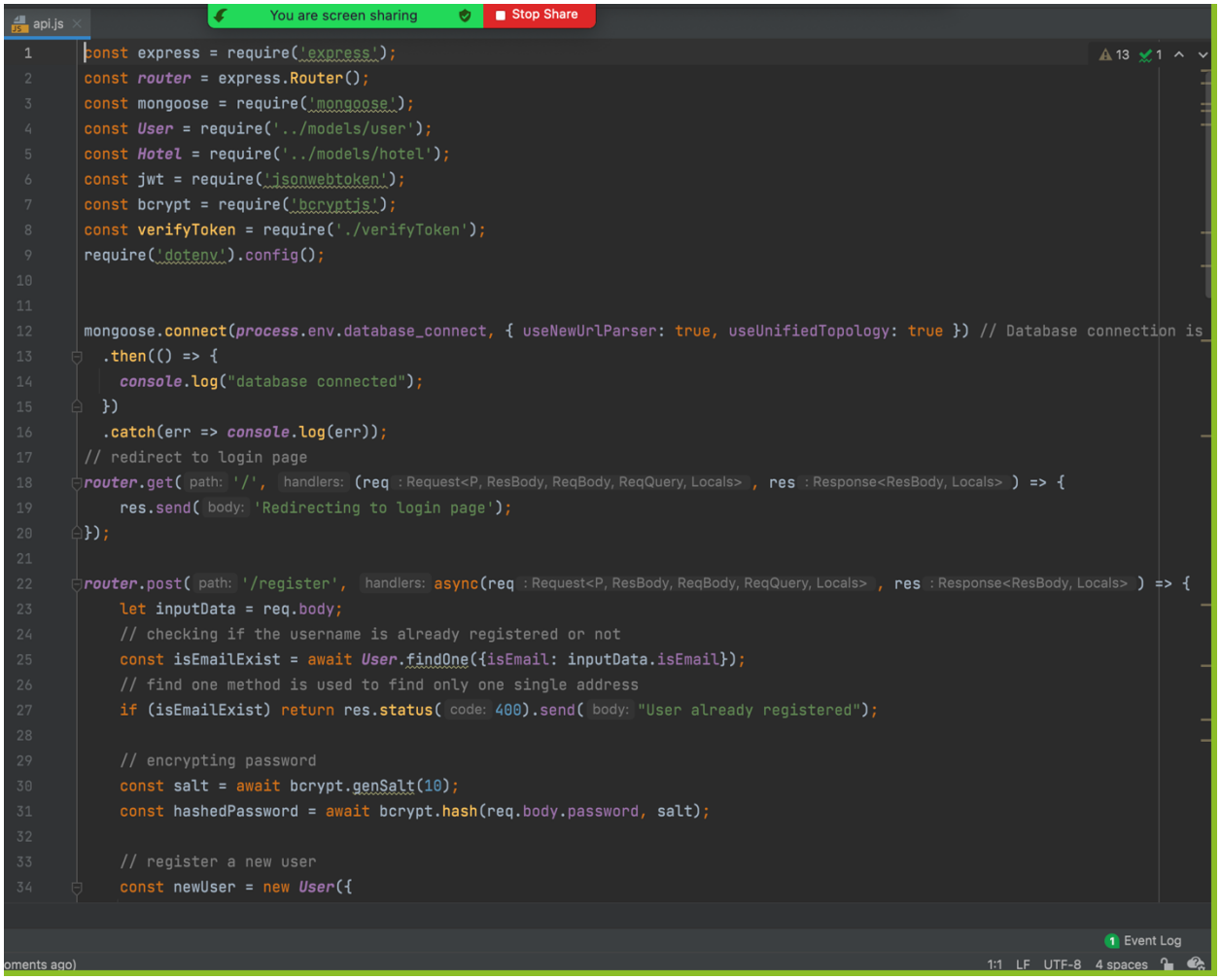


Figure 13: Backend API Code

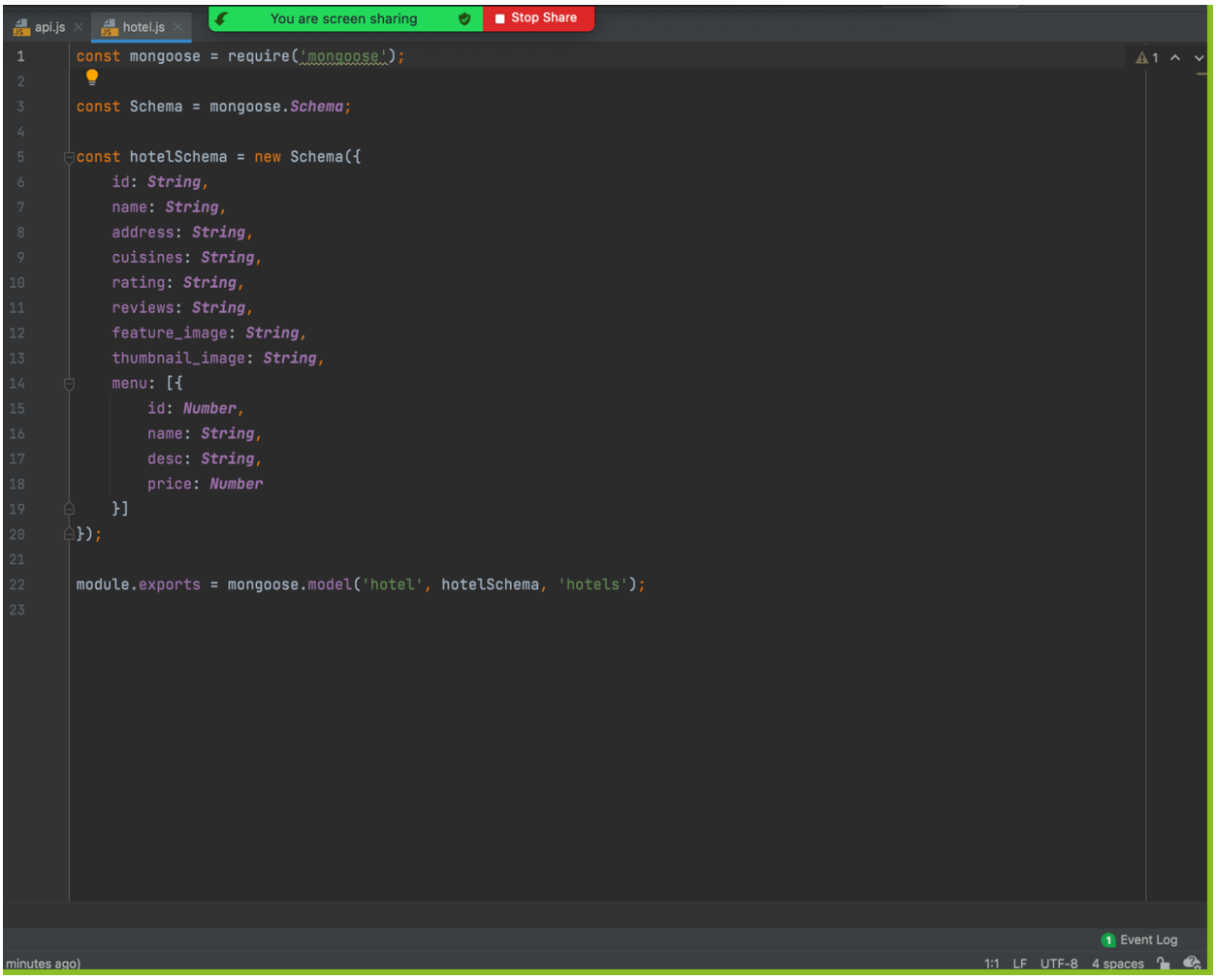
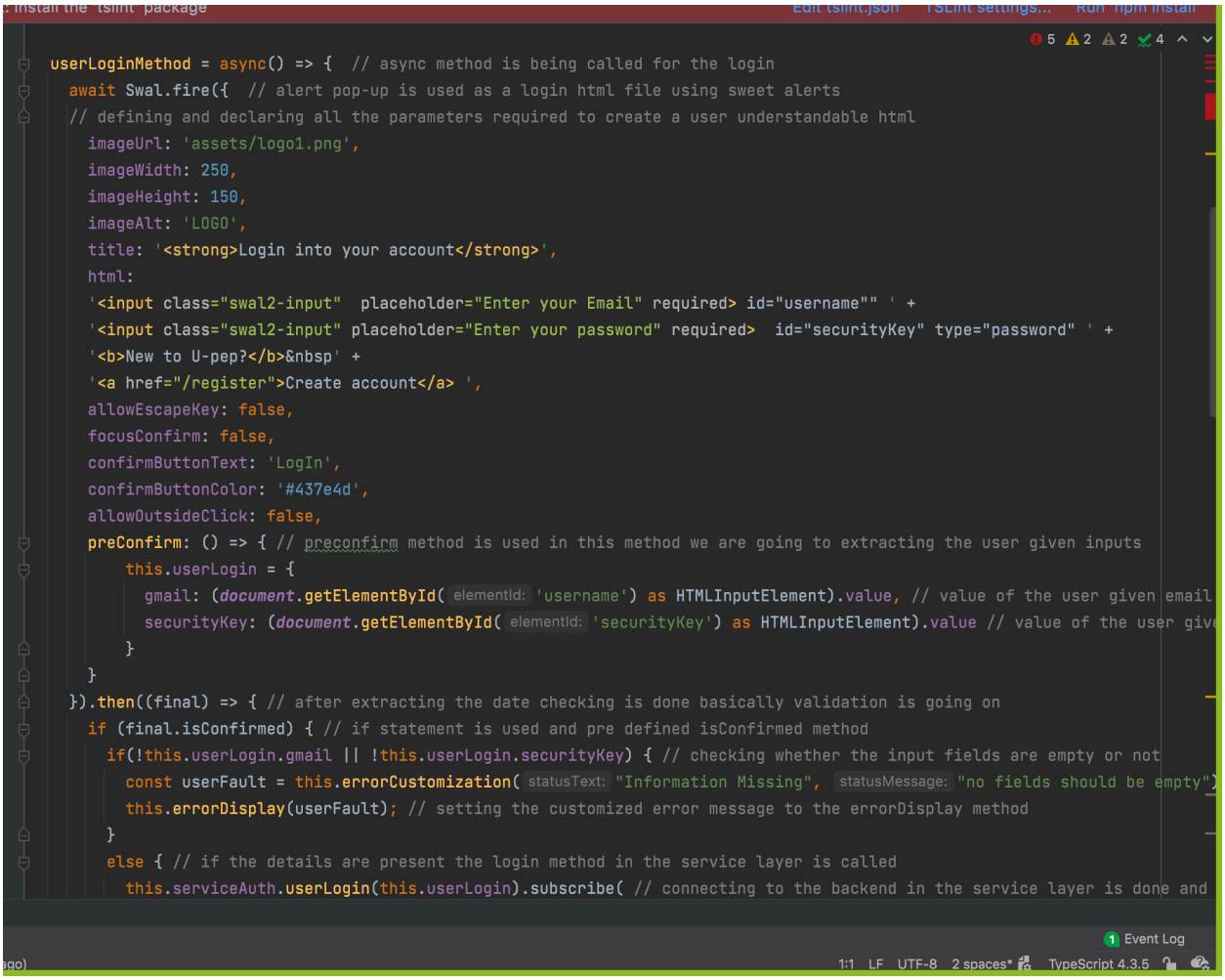


Figure 14: Hotel Schema

```
// getting a list of hotels from the database
router.get( path: '/hotels', verifyToken, async (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<any, Record<string, any>>
    try{
        const hotels = await Hotel.find();
        res.json(hotels);
    } catch(err) {
        res.status( code: 404).send( body: `request is not processed - ${err}`);
    }
});

// finding selected hotel using hotel ID
router.get( path: '/hotels/:hotelId', verifyToken, async (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<any, Record<
    try{
        const hotel = await Hotel.findOne({id: req.params.hotelId});
        res.json(hotel);
    } catch(err) {
        res.status( code: 404).send( body: `request is not processed - ${err}`);
    }
});
```

Figure 15: Getting Hotels List



```
userLoginMethod = async() => {  // async method is being called for the login
  await Swal.fire({  // alert pop-up is used as a login html file using sweet alerts
  // defining and declaring all the parameters required to create a user understandable html
    imageUrl: 'assets/logo1.png',
    imageWidth: 250,
    imageHeight: 150,
    imageAlt: 'LOGO',
    title: '<strong>Login into your account</strong>',
    html:
    '<input class="swal2-input"  placeholder="Enter your Email" required> id="username"" ' +
    '<input class="swal2-input" placeholder="Enter your password" required>  id="securityKey" type="password" ' +
    '<b>New to U-pep?</b> ' +
    '<a href="/register">Create account</a> ',
    allowEscapeKey: false,
    focusConfirm: false,
    confirmButtonText: 'LogIn',
    confirmButtonColor: '#437e4d',
    allowOutsideClick: false,
    preConfirm: () => { // preconfirm method is used in this method we are going to extracting the user given inputs
        this.userLogin = {
          gmail: (document.getElementById( elementId: 'username') as HTMLInputElement).value, // value of the user given email
          securityKey: (document.getElementById( elementId: 'securityKey') as HTMLInputElement).value // value of the user give
        }
    }
  }).then((final) => { // after extracting the date checking is done basically validation is going on
    if (final.isConfirmed) { // if statement is used and pre defined isConfirmed method
      if(!this.userLogin.gmail || !this.userLogin.securityKey) { // checking whether the input fields are empty or not
        const userFault = this.errorCustomization( statusText: "Information Missing",  statusMessage: "no fields should be empty")
        this.errorDisplay(userFault); // setting the customized error message to the errorDisplay method
      }
      else { // if the details are present the login method in the service layer is called
        this.serviceAuth.userLogin(this.userLogin).subscribe( // connecting to the backend in the service layer is done and
```

Figure 16: Login Page

Work Distribution:

- Muhammad Usman & Ritwik Reddy taking care of the Backend framework and managing overall project.
- Aravind Reddy & Damodar Reddy is working on the front-end design of the website.
- All team are working on collecting the data and completing the documentation.

Issue & Blockages with the Project:

1. Due to lack of experience of our team members and with short time period, it was difficult to include all the features in our applications. However, we included all the major features that is required for online food ordering app.
2. Second challenge was to develop the admin panel. We first tried the auto generated Adminbros admin panel that is design for node applications, and we successfully deployed that admin panel. However, there was a problem with iterating the nested array from the database and after discussing with professor Vijaya, we created the admin panel in PHP which was easy, and it was easily connected with the mongodb database.
3. In the beginning we are planning to add online grocery stores in our application, but the business logic of online grocery store was not matching with the online food ordering application. If we add grocery stores in our application, then it would become difficult for grocery store owners to maintain their stocks in the database system. Due to this issue, we excluded this feature from our application.

GitHub Link:

https://github.com/MuhammadUsman94/Web-Mobile_Project

Video Link:

https://youtu.be/RzBavUCulIA

Presentation Link:

https://github.com/MuhammadUsman94/Web-Mobile_Project/blob/main/Power%20Point/Web%20and%20Mobile%20PPT.pptx

References:

- https://www.spaceotechnologies.com/create-online-food-ordering-system-guide/
- https://www.youtube.com/watch?v=WEqOr-clmMM

- **https://www.youtube.com/watch?v=dMQFT270axQ**
- **https://www.youtube.com/watch?v=UrfhqE7I-3o**