B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Information Science and Engineering

Vidyayāmruthamashnuthe

22ISE136 – Web Technologies

Mini-Project Report

on

FOOD ORDERING SYSTEM

Submitted in partial fulfillment of the requirement
for the award of the degree of

Bachelor of Engineering

in

Information Science & Engineering

by

Anusha Sanjana(1BG22IS006)
Divyashree V (1BG22IS014)

Academic Year 2023 – 2024

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Vidyayāmruthamashnuthe

CERTIFICATE

Certified that the Mini-project entitled **Food Ordering System** is carried out by Ms. **Anusha Sanjana (1BG22IS006)** and **Divyashree V (1BG22IS014)** the bonafide student of **B.N.M Institute of Technology** in partial fulfillment for the award of **Bachelor of Engineering** in **Information Science & Engineering** of the **Visvesvaraya Technological University**, Belagavi during the year 2023-2024. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library. The mini-project report has been approved as it satisfies the academic requirements in respect of mini-project prescribed for the said Degree.

Mrs Kavya N L
Asst. Prof, Dept. of ISE
BNMIT

Dr. S Srividhya
Prof & Head, Dept. of ISE
BNMIT

Name & Signature of the Examiners with date:

1.

2.

ACKNOWLEDGEMENT

We consider it a privilege to express through the pages of this report, a few words of gratitude to all those distinguished personalities who guided and inspired me in the completion of this mini project.

We would like to thank **Shri. Narayan Rao R Maanay**, Secretary, BNMEI, Bengaluru for providing an excellent academic environment in college.

We would like to thank **Prof. T.J. Rama Murthy**, Director, BNMIT, Bengaluru for having extended his support and encouragement during the course of work.

We would like to thank **Dr. S.Y. Kulkarni**, Additional Director, BNMIT, Bengaluru for his extended support and encouragement during the course of work.

We would like to express my gratitude to **Prof. Eishwar N Maanay**, Dean, BNMIT, Bengaluru for his relentless support, guidance, and encouragement.

We would like to thank **Dr. Krishnamurthy G.N**., Principal, BNMIT, Bengaluru for his constant encouragement.

We would like to thank **Dr. S. Srividhya**, Professor and Head of the Department of Information Science and Engineering, BNMIT, Bengaluru, for her support and encouragement towards the completion of the mini project.

We would like to express my gratitude to my guide **Mrs Kavya N L,** Assistant Professor Department of Information Science and Engineering, BNMIT, Bengaluru, who has given us all the support and guidance in completing the mini project successfully.

We would like to thank course coordinator **Mrs Kavya N L,** Assistant Professor, Department of Information Science and Engineering, BNMIT, for being the guiding force towards the successful completion of the mini project.

Anusha Sanjana
(1BG22IS006)

Divyashree V
(1BG22IS014)

CHAPTER 1

INTRODUCTION

In the dynamic realm of modern convenience, the integration of technology into our daily lives has revolutionized the way we approach even the simplest tasks. The introduction of a user-friendly food ordering system, crafted with the seamless combination of HTML and CSS, epitomizes this evolution in the culinary landscape. Picture a virtual gateway to a gastronomic world where users can navigate through an aesthetically pleasing interface, carefully designed with HTML to ensure a robust and responsive structure. CSS, with its styling prowess, adds the visual charm, elevating the user experience to a delightful level.

This innovative food ordering system not only simplifies the process of selecting and customizing culinary delights but also enhances the overall engagement through an intuitive and visually appealing design. As we embark on this journey, let the amalgamation of HTML and CSS pave the way for a revolutionary gastronomic adventure, where technology meets taste with unparalleled sophistication.

1.1 Objective

The primary objectives of developing a food ordering system using HTML and CSS are to create an efficient, user-friendly, and visually appealing interface for users to browse and place orders seamlessly. The system aims to enhance the overall user experience and streamline the food ordering process. Here are the key objectives:

- User-friendly Interface
- Responsive Design
- Attractive Visual Design
- Menu Presentation
- Order Customization
- User Authentication and Profile Management
- Real-time Updates
- Feedback Mechanism
- Cross-browser Compatibility

1.2 Scope

- HTML and CSS allow for the creation of visually appealing and responsive user interfaces.
- HTML can be used to structure the menu content, while CSS can style it for clarity and attractiveness.
- CSS can be leveraged to implement responsive design, ensuring that the food ordering system is accessible and user-friendly
- HTML enables the creation of navigation elements, while CSS styles them for easy navigation through different sections of the food ordering system, such as the menu, cart, and checkout pages.

1.3 Motivation

Designing a food ordering system using HTML and CSS can be an exciting and rewarding project that brings numerous benefits to both users and businesses. Motivation for such a system stems from the desire to streamline and enhance the overall dining experience. By leveraging the power of web technologies, we aim to create a user-friendly platform that simplifies the process of ordering food.

One of the key motivations is to provide convenience to customers. In today's fast-paced world, people often seek efficient solutions that save time and effort. A well-designed food ordering system allows users to browse through menus, select their favorite dishes, and place orders with just a few clicks. This seamless experience not only caters to the on-the-go lifestyle but also encourages users to explore a variety of food options without any hassle**.**

1.4 Application Development Need & Importance

The development of a food ordering system through HTML and CSS brings significant advantages and addresses crucial needs in the modern world. As the food industry rapidly embraces digital transformation, an online ordering system becomes indispensable for restaurants and customers alike. One key need that an HTML and CSS-based food ordering system fulfill is the growing demand for convenience. In today's fast-paced lifestyle, people seek efficient ways to access services, including ordering food. HTML and CSS, the foundational technologies for web development, provide a user-friendly.

Table of Contents

List of Figures

CHAPTER 2

METHODOLOGY

Creating a food ordering system using HTML and CSS involves several steps to ensure a well-organized and visually appealing interface. Below is a suggested methodology described in paragraph form: To begin, establish a clear understanding of the system's requirements and user flow. Define the key features such as menu display, item selection, cart management, and order confirmation. Once the requirements are outlined, start by designing a wireframe or mockup to visualize the layout and structure of the food ordering system.

2.1 Techniques Used

Creating a food ordering system using HTML and CSS involves several key techniques to ensure a user-friendly and visually appealing interface. Here are some essential techniques explained in paragraph form:

- **HTML Structure:** Begin by structuring your HTML document with semantic elements. Use headings, sections, and divs to organize different parts of the page, such as the header, menu, order summary, and footer. This provides a clear and logical structure for your food ordering system.

- **Form Elements**: Utilize HTML form elements to collect user input for food orders. Employ labels, input fields, checkboxes, and radio buttons to capture relevant information, such as item selections, quantities, and special instructions. Use the <form> tag to wrap these elements for better organization.

- **CSS Styling:** Apply CSS styles to enhance the visual appeal of your food ordering system. Use a consistent color scheme, typography, and spacing to create a cohesive design. Employ CSS classes and IDs to target specific elements and ensure a uniform style across your entire application. Consider responsive design principles to make your system adaptable to various screen sizes.

- **Grid Layouts**: Implement CSS grid layouts to structure your page in a grid system. This allows you to create responsive and flexible designs, ensuring that your food ordering system looks good on different devices. the alignment of elements and make it easier to create a visually appealing and organized interface.

2.2 Tools Used

Creating a food ordering system involves more than just HTML and CSS. You'll need to use additional technologsies such as JavaScript for interactivity and possibly a server-side language (like PHP, Node.js, Python, etc.) to handle the backend functionality. Additionally, you might use a database to store and retrieve order information.

To develop a food ordering system using React and Spring Boot, you would need a variety of tools and technologies. Here's a list of some essential tools and technologies you might consider using:

- **React**: A JavaScript library for building user interfaces.
- **Spring Boot**: An opinionated framework for building production-ready Spring applications quickly.
- **Node.js**: A JavaScript runtime environment that executes JavaScript code server-side.
- **Create React App:** A tool for quickly setting up a new React project with a sensible default configuration.
- **Spring Framework:** Provides comprehensive infrastructure support for developing Java applications.
- **Spring Data JPA:** Part of the larger Spring Data project, it provides easy implementation of JPA-based repositories.
- **Spring Security:** For securing your Spring-based applications.
- **Hibernate:** An ORM (Object-Relational Mapping) framework for the Java language, which provides mapping of Java classes to database tables.
- **MySQL** or another relational database: For storing application data.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

Designing a food ordering system involves both frontend (HTML and CSS) and backend development. In this response, I'll provide you with a basic HTML and CSS structure for the frontend part of a food ordering system. Keep in mind that for a complete and functional system, you'll also need JavaScript for interactivity and a backend language like PHP, Node.js, or Python for processing orders.

3.1 Software Requirements

Creating a food ordering system using HTML and CSS is a good starting point, but for a fully functional system, you'll also need to incorporate JavaScript for interactivity and possibly a server-side language (such as PHP, Python, or Node.js) for processing orders and managing data. Here are the basic software requirements for a simple food ordering system:

- **Text Editor:** Choose a text editor to write your HTML, CSS, and JavaScript code. Popular choices include Visual Studio Code, Sublime Text, Atom, or any other editor of your preference.
- **Web Browser**: A modern web browser (Google Chrome, Mozilla Firefox, Safari, etc.) for testing and debugging your HTML and CSS code.
- **HTML (Hypertext Markup Language):** Use HTML to structure your web pages. Include elements like forms for order details, buttons for submission, and divs for organizing content.
- **CSS (Cascading Style Sheets):** Apply CSS for styling your HTML elements, making your food ordering system visually appealing and user-friendly.
- **JavaScript:** Implement JavaScript to add interactivity to your system. This can include form validation, dynamic content updates, and handling user interactions.
- **Backend Language** (Optional, but recommended for a complete system): If you want to store and process orders, you'll need a server-side language. Common choices include:
- **PHP:** Great for server-side scripting and widely used in web development.
- **Node.js** (JavaScript on the server): Useful for creating scalable network applications.

3.2 Hardware Requirements

Creating a food ordering system using HTML and CSS is primarily focused on the front-end development, which involves designing and styling the user interface. However, keep in mind that a complete food ordering system also requires back-end development for processing orders, managing user accounts, and handling transactions. Below are the hardware requirements for the front-end development using HTML and CSS:

- **Computer:** A standard personal computer or laptop with a modern web browser (e.g., Google Chrome, Mozilla Firefox, Safari) is sufficient.
- **Processor and RAM:** Since HTML and CSS are lightweight technologies, any modern computer with a decent processor (e.g., Intel Core i3 or equivalent) and at least 4GB of RAM should be adequate for front-end development.
- **Storage:** Adequate storage space for saving project files, images, and other assets. Given the nature of HTML and CSS development, storage requirements are generally not intensive.
- **Graphics Card:** A dedicated graphics card is not necessary for HTML and CSS development. The integrated graphics found in most modern processors are sufficient.
- **Internet Connection:** A stable internet connection is essential for accessing online resources, testing responsive design, and ensuring compatibility with different browsers.
- **Display:** A standard display with a resolution of 1366x768 pixels or higher is recommended for comfortable coding and design work.

3.3 Functional Requirements

Functional requirements for a food ordering system using HTML and CSS typically involve defining the user interface, user interactions, and data handling. Here's a basic set of functional requirements for such a system:

- **User Registration and Login:** Users should be able to register for an account. Users should be able to log in with their credentials.
- **Browse Menu:** Display a categorized menu with food items. Include images, names, and prices for each item.

- **Add to Cart**: Users should be able to add items to their shopping cart. Display the current contents of the shopping cart. Show the total price of items in the cart.
- **Modify Cart:** Allow users to update the quantity or remove items from the cart. Update the total price dynamically as items are added or removed.
- **Checkout**: Users should be able to proceed to checkout. Collect user details for delivery (address, contact information). Display the final order summary and total price.
- **Order Confirmation**: Provide a confirmation message after a successful order placement. Generate and display an order confirmation number.

3.4 Non - Functional Requirements

Non-functional requirements are aspects of a system that describe how it should perform, rather than what functions it should accomplish. For a food ordering system using HTML and CSS, non-functional requirements might include:

- **Performance:** The system should load within 3 seconds on standard internet connections. The response time for placing an order should be less than 5 seconds. The system should be able to handle a concurrent user load of at least 1000 users.
- **Scalability:** The system should be easily scalable to accommodate a 20% increase in user base within a year. It should handle a sudden surge in orders, such as during promotional events or holidays.
- **Reliability:** The system should have an uptime of at least 99.9%. It should be able to recover gracefully from server failures or crashes.
- **Availability**: The food ordering system should be available 24/7, allowing users to place orders at any time. Scheduled maintenance, if required, should be communicated to users in advance.
- **Security**: User data, including personal information and order history, should be securely stored and encrypted. Payment transactions should comply with industry-standard security protocols (e.g., HTTPS).
- **Compatibility**: The system should be compatible with major web browsers (Chrome, Firefox, Safari, and Edge). It should be responsive and provide a consistent user experience across different devices (desktops, tablets, and mobile phones).

CHAPTER 4

SYSTEM DESIGN AND DEVELOPMENT

Designing and developing a food ordering system involves several components, including the user interface (UI), backend logic, and database management. In this example, I'll provide you with a basic outline using HTML and CSS for the frontend. Keep in mind that a complete system would also require server-side scripting (e.g., PHP, Python, Node.js) for backend logic and a database for storing orders and user information.

4.1 Architectural Design

Extend the HTML and CSS as needed based on the features you want to implement.
Use appropriate class names and IDs for better organization and styling. may incorporate JavaScript for dynamic interactions and AJAX for handling backend requests as shown in Fig 4.1

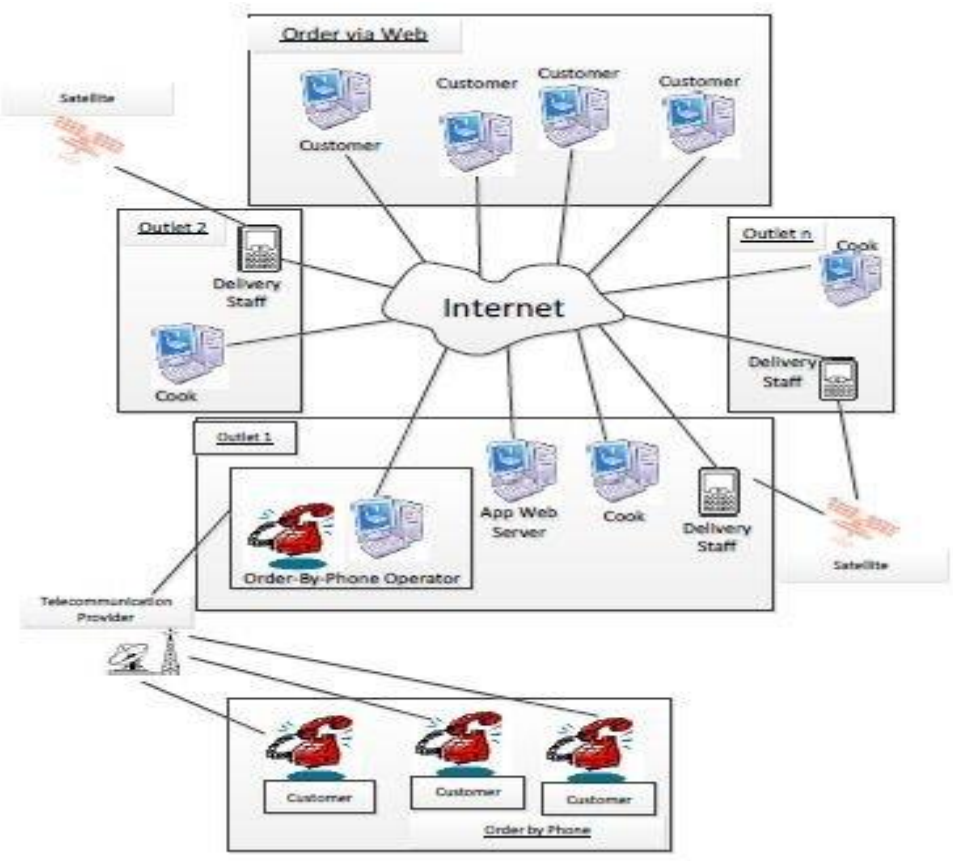

Fig 4.1 Architecture

4.2 Development

Building a food delivery system using React for the frontend and Spring Boot for the backend involves several steps. Below is a high-level outline of the process:

Backend (Spring Boot):

- Setup Spring Boot Project: Use Spring Initialize or your preferred method to create a new Spring Boot project.
- Include dependencies such as Spring Web, Spring Data JPA, and a database driver (e.g., H2, MySQL, PostgreSQL).
- Define Entity Classes: Create Java classes to represent entities like User, Restaurant, Menu, Order, etc.
- Use annotations like @Entity, @OneToMany, @ManyToOne to define relationships.
- Implement Repository Interfaces: Create repository interfaces extending JpaRepository to perform CRUD operations on entities.
- Service Layer: Implement service classes to handle business logic and interact with repositories.
- RESTful API Endpoints: Create controllers with methods to handle HTTP requests (GET, POST, PUT, DELETE).
- Map these methods to specific URL paths and HTTP methods.
- Security: Implement security measures using Spring Security to protect sensitive endpoints.
- Use authentication and authorization mechanisms.
- Testing: Write unit and integration tests for your backend services and controllers.
- Documentation: Generate API documentation using tools like Swagger or Spring RestDocs.
- Database Configuration: Configure the application properties for the chosen database.

Frontend (React):

- Setup React Project: Use Create React App or another method to initialize a new React project.
- Component Structure: Design the component structure for your application. Consider components for user authentication, restaurant listing, menu display, order processing, etc.

- API Integration: Use fetch or a library like Axios to interact with the backend API.
- Implement functions to handle data fetching, posting, and updating.
- State Management:

- Choose and implement a state management solution (e.g., React Context, Redux) for managing the application state.
- Routing: Use React Router to handle navigation between different pages or components.
- User Authentication: Implement user authentication features using JWT or OAuth.
- Forms and User Input: Create forms for user input (e.g., placing an order) and handle form submission.
- UI/UX Design: Design and style your components for a user-friendly experience.
- Consider using a UI library or framework for a consistent look and feel.
- Testing: Write unit tests for React components and integration tests for API calls.
- Deployment: Choose a hosting solution for your frontend (e.g., Netlify, Vercel) and deploy the application.

Integration:

- CORS Configuration: Ensure that your Spring Boot backend allows Cross-Origin Resource Sharing (CORS) from your React frontend.
- Connect Frontend and Backend: Update the API endpoint URLs in your React application to point to the deployed backend.
- Testing: Perform end-to-end testing to ensure seamless communication between the frontend and backend.
- Deployment: Deploy both the frontend and backend to their respective hosting platforms.
- By following these steps, you can create a functional food delivery system using React for the frontend and Spring Boot for the backend. Adjustmesnts and additional features can be added based on specific project requirements.

CHAPTER 5

IMPLEMENTATION

5.1 Modules Implemented

Index.html: The following HTML code describes the structure of the webpage. First of all it has an header tag to display the name of the company which is “Online FoodShop”. The navigation bar contains tabs of “Home”, “About”, “Menu” and “ContactUs”. The navigation bar also consists of the logo of the company followed by the background image with some more details of the “Online FoodShop” along with the featured products.

Contact.html: The following HTML code describes us what contents are present in the “ContactUs” tab in the navigation bar. We have used input tags to enter the type of “query”, “name”, “email-id”, “phone number” and “explanation” of the query. We have also used radio buttons to enquire about the customer’s membership for “online FoodShop “. The form also provides “Submit” and “Reset” buttons.

Style.css: This is the stylesheet block which gives us the proper look of the webpage.

Header styling: It consists of two “head” tags. We have used *font-style* “Ubuntu” to style the header and we also used proper colors to make some text look attractive.

Navigation bar styling: It basically consists of logo styling and content styling. First of all we have considered the navigation bar as a flexbox and assigned the *flex-direction* to “column”. We have centered the logo image by *display:block*. We have used the *margin:auto* property to center the image. Then we considered the content of the navigation bar as a flexbox and then we have positioned it properly.

CHAPTER 6

TESTING and RESULTS

6.1 Snapshots of the project and description

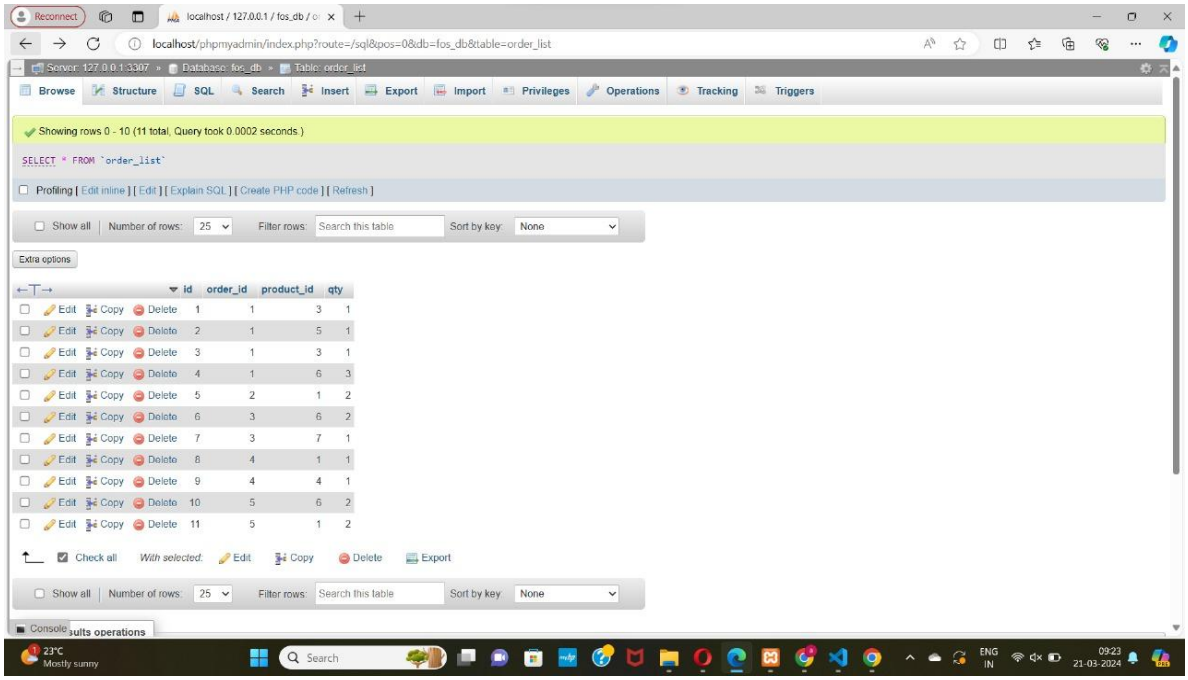

Fig 6.1 Front Page

6.2 Observation

This is the front page of the project as shown above in fig 6.1 . here we can just see the ambience of the website and we can add persons with names and we can add to cart and also we can search any food items in the website. We can finally order our favourite food**.**

This is the page of registration as shown in Fig 6.2 here the user can register with our name, email id, and phone number .

If we are the new customer it will shown as new or if we are already registered then it will show already have an account.

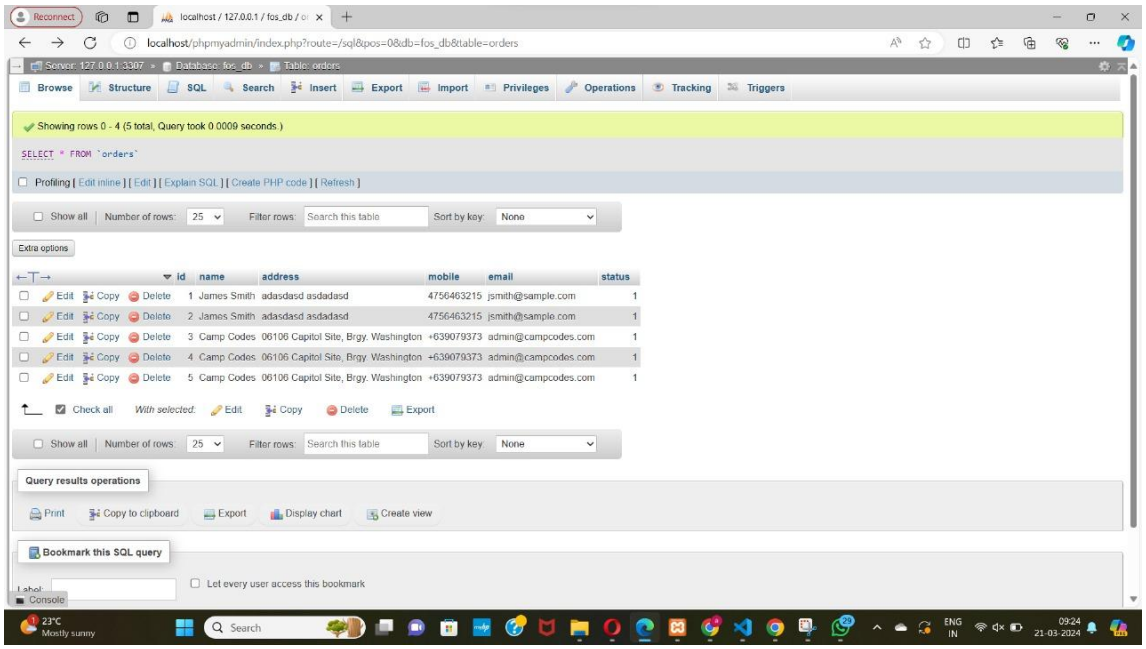

Fig 6.2 Page of Registration

The Food Ordering System project is a comprehensive web application developed using React for the frontend and Spring Boot for the backend. This system aims to streamline and enhance the process of ordering food from various restaurants, providing users with a user-friendly and efficient platform. In the frontend, React, a popular JavaScript library for building user interfaces, is employed to create a responsive and dynamic user interface. Users can easily navigate through the application, browse restaurant menus, and place orders with a seamless and intuitive experience. The use of React components facilitates code reusability, making the application modular and maintainable.

On the backend, the Spring Boot framework is utilized to build a robust and scalable server-side application. Spring Boot simplifies the development of Java-based applications by providing a convention-over-configuration approach. It supports the creation of RESTful APIs, allowing smooth communication between the frontend and backend. The backend is responsible for handling user authentication, managing orders, interacting with the database, and coordinating communication with external payment systems. The system incorporates a secure user authentication mechanism to ensure that only authorized users can place orders and access personal information. Additionally, it integrates a database to store and retrieve information about restaurants, menus, and user orders. This ensures data consistency and persistence.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 Conclusion

In conclusion, the development of the Food Ordering System using React for the frontend and Spring Boot for the backend has proven to be a successful and efficient solution for streamlining the online food ordering process. The React frontend provides a user-friendly and responsive interface, allowing customers to easily browse through menus, place orders, and track their deliveries with a seamless experience. On the other hand, the Spring Boot backend ensures robust and secure handling of data, facilitating efficient communication between the client and server. The use of React and Spring Boot synergistically enhances the overall performance and scalability of the system. React's component-based architecture enables modular development, fostering code reusability and maintainability.

7.2 Future Enhancement:

The future enhancement for the food ordering system, developed using React for the frontend and Spring Boot for the backend, could involve several key aspects to improve user experience, functionality, and overall system efficiency.

- **Real-time Order Tracking:** Implementing a real-time order tracking feature would enhance user satisfaction. Users should be able to track the status of their orders, from confirmation to preparation and delivery. This could involve integrating geolocation services to provide accurate delivery estimates.
-
- **Personalized Recommendations**: Introduce machine learning algorithms to analyze user preferences and ordering history. By offering personalized food recommendations, the system can enhance user engagement and encourage repeat orders.

- **Enhanced Payment Options**: Integrate additional payment options, such as digital wallets, cryptocurrency, and other emerging methods. This not only caters

to a broader user base but also improves the flexibility and convenience of the ordering process.

- **Multi-Language Support**: To reach a wider audience, implement multi-language support in both the frontend and backend. This allows users to navigate the application and place orders in their preferred language, enhancing accessibility and user satisfaction.

- **Optimized Performance:** Continuously optimize the system's performance by implementing caching mechanisms, load balancing, and other strategies. This ensures that the application remains responsive, even during peak usage times, providing a seamless experience for users.

- **Social Media Integration**: Enable users to share their food orders, reviews, and experiences on social media platforms directly from the application. This can serve as a marketing tool and also create a sense of community among users.

References

https://www.divaportal.org/smash/get/diva2:1756935/FULLTEXT01.pdf

https://www.researchgate.net/publication/368659794_Food_Spoilage_Detection_Using_IoT

https://www.smec.ac.in/assets/images/research/ece/1920/79.food%20monitoring%20paper%20NEW.pdf

https://www.ijcrt.org/papers/IJCRTICLI027.pdf