



TRIBHUVAN UNIVERSITY
Institute of Science and Technology

Project Report On
“Food Delivery Web Application
with Payment Gateway”

Submitted to:
Department of Computer Science and
Information Technology
AMRIT SCIENCE CAMPUS

*In partial fulfillment of the requirements for the Bachelor's Degree in Computer
Science and Information Technology*

Submitted by:
Biplov Subedi (23134/076)
Bishal Bhatta (23135/076)
Madhav Dhakal (23152/076)

Under the Supervision of
Yuba Raj Devkota

Supervisor's Recommendation

The project work report titled “Food Delivery Web Application with Payment Gateway” is submitted by Madhav Dhakal, Biplov Subedi and Bishal Bhatta in partial fulfillment of the requirements for the degree of Bachelor of Computer Science and Information Technology, Tribhuvan University under my guidance and supervision. To the best of my knowledge, the information presented by him/her in the project report has not been submitted earlier.

.....

Signature of the Supervisor

Name: Mr. Yuba Raj Devkota

Designation: Supervisor

Certificate of Approval

This is to certify that the report is evaluated and recommended to the Department of Computer Science and Information Technology for acceptance of the report entitled “**Food Delivery Web Application with Payment Gateway**” is submitted by **Madhav Dhakal, Biplov Subedi and Bishal Bhatta** in partial fulfillment for the degree of Bachelor of Science in Computer Science and Information Technology (B.Sc.CSIT), Institute of Science and Technology (IOST), Tribhuvan University.

Mr. Binod Adhikari

Project Coordinator

Mr. Yuba Raj Devkota

Project Supervisor

External Examiner

Acknowledgement

Our final year project's success can be attributed to the invaluable assistance provided by our project supervisors and coordinator. **Mr. Yuba Raj Devkota** our assigned supervisor, deserves sincere appreciation for his unwavering support. We would also like to express our gratitude to the Amrit Science Campus (ASCOL) for offering an exceptional platform that facilitated the efforts made in achieving and developing the project.

With the help of **Mr. Yuba Raj Devkota** and the ASCOL team, our group learned a lot about different parts needed to create this system. This experience gave us a good understanding of how complex systems work, preparing us for future projects in this field.

We are immensely thankful to the ASCOL team for their thorough review, approval, and guidance throughout this transformative journey. Special recognition is also extended to the supportive online communities, as well as our friends and families, whose assistance played a crucial role in the proper design, construction, and development of this application. In conclusion, we feel truly honored by the collaborative efforts and support that have significantly contributed to the success of our project.

Abstract

This project presents a comprehensive food ordering and delivery system designed to streamline the process of ordering food from restaurant. The system encompasses various features including user authentication, restaurant management, order tracking, and user profile management. It uses technologies such as Node.js for the backend server, Express.js for routing, and Mongo DB for the database. The frontend is built using React.js to ensure a responsive and intuitive user interface. Notably, the system integrates advanced payment capabilities through PayPal, facilitating secure and hassle-free transactions for users. Moreover, a map system is incorporated to provide users with interactive location services, enhancing their ability to select delivery addresses and explore nearby food options. This project serves as a robust platform that prioritizes user experience, security, and convenience in the domain of online food ordering.

The project involved developing various functionalities for an online food delivery system, including API routes for managing food items, user orders, and user authentication. Additionally, services for integrating PayPal payment processing, Mailgun email notification, and a map interface using Leaflet were implemented. Findings from the project revealed successful integration of these features, enabling users to browse food items, place orders, and track their deliveries efficiently. The PayPal integration facilitated secure payment processing, while Mailgun enabled timely email notifications for order updates. The map interface provided an intuitive way for users to select delivery locations.

Keywords: User authentication, order tracking, Node.js, Express.js, Mongo DB, Routing, React.js

Table of Contents

Supervisor’s Recommendation	ii
Certificate of Approval	iii
Acknowledge ment	iv
Abstract	v
List of Abbreviations	viii
List of Figures.....	ix
List of Tables	x
CHAPTER 1 Introduction	1
1.1.Introduction.....	1
1.2. Problem Statement	1
1.3. Objectives	2
1.4. Scope and Limitation.....	2
1.4.1. Scope of the Project:.....	2
1.4.2. Limitations of the Project:.....	2
1.5. Development Methodology	2
CHAPTER 2 Background Study and Literature Review.....	5
2.1. Background Study	5
2.2. Literature Review	6
CHAPTER 3 System Analysis	9
3.1. System Analysis.....	9
3.1.1. Requirement Analysis.....	9
3.1.2. Feasibility Analysis.....	11
3.1.3. Analysis.....	13

CHAPTER 4 System Design	16
4.1. Design	16
4.1.1. Refinement of Class Diagram	16
4.1.2. Refinement of Sequence Diagram.....	17
4.1.3. Refinement of Activity Diagram	19
4.1.4. Component Diagram.....	22
4.1.5. Deployment Diagram.....	23
CHAPTER 25 Implementation and Testing	25
5.1 Implementation	25
5.1.1. Tools Used	25
5.1.2. Implementation Details of Modules	27
5.2. Testing.....	34
5.2.1. Test Cases for Unit Testing.....	34
5.2.2. Test Cases for System Testing	36
5.3 . Result Analysis	37
CHAPTER 6 Conclusion and Future Recommendation	40
6.1 Conclusion	40
6.1. Future Recommendations	41
REFERENCES	42
APPENDICES	43

List of Abbreviations

CSS	Cascading Style Sheets
HTTP	Hypertext Transfer Protocol
HTML	Hyper Text Markup Language
ICT	Information and Communication Technology
CRUD	Create Read Update and Delete
UML	Unified Modeling Language
OOP	Object-Oriented Programming
MERN	MONGODB EXPRESS REACT NODE

List of Figures

Figure 3.1: Use Case for Authentication.....	10
Figure 3.2: Class Diagram of Food Delivery System.....	14
Figure 3.3: Sequence Diagram of Food Delivery System.....	15
Figure 3.4: Activity Diagram for Food Delivery System.....	16
Figure 4.1: Refined Class Diagram.....	17
Figure 4.2: Refined Sequence Diagram.....	19
Figure 4.3: Activity Diagram for Admin.....	21
Figure 4.4: Refined Activity Diagram for User.....	22
Figure 4.4: Component Diagram for Food Delivery System.....	23
Figure 4.5: Deployment Diagram for Food Delivery System.....	25
Figure 5.1: Implementation of Registration Module	29
Figure 5.2: Implementation of Login Module	29
Figure 5.3: Implementation of Admin Module.....	30
Figure 5.4: Implementation of Food Module.....	31
Figure 5.5: Implementation of Order Module.....	32
Figure 5.6: Implementation of Image Module.....	33
Figure 5.7: Implementation of Payment Module.....	34
Figure 5.8: Implementation of Map Module.....	35

List of Tables

Table 3.1: Schedule Table	12
Table 5.1: Test Case for Registration	35
Table 5.2: Test Case for Login	36
Table 5.3: Test Case for System Testing	37

CHAPTER 1

Introduction

1.1. Introduction

The prevalence of digital technologies has transformed how we engage with a variety of services, including the food industry. Online platforms now provide customers unparalleled access to numerous dining options from home. As this trend continues to increase in popularity, there is an emerging need for efficient and diverse food delivery applications that prioritize user convenience and choice. This project aims to contribute by developing a comprehensive food delivery application utilizing cutting-edge web and technology to simplify ordering from home. Advanced features such as real-time order tracking, secure payment processing, interactive map navigation will enhance users' overall satisfaction with their experience effortlessly browsing menus before selecting desired items alongside timely deliveries provided through modern means close at hand. By building this user-centric platform addressing modern consumers changing preferences while exploring any challenges along the way demonstrates our commitment towards making life easier one meal at a time!

1.2. Problem Statement

The digital age has been in a growing demand for efficient food delivery services, as consumers increasingly opt for convenience when ordering meals from restaurants. However, restaurants face various challenges to provide seamless delivery experiences to customers due to inadequate integration with popular payment methods such as PayPal and the lack of an interactive map system that can streamline the process and boost customer satisfaction. Consequently, developing a user-friendly food delivery application tailored specifically to meet our unique requirements poses the primary challenge. This app will focus on integrating PayPal features into transactions while ensuring they remain smooth and hassle-free; it will also incorporate robust mapping functions so we can track deliveries efficiently throughout every stage of their journey - all with maximum ease-of-use at its core fitting well within our establishment's operations.

1.3. Objectives

The main objectives of this project are as follows:

- To create a food delivery application tailored to our restaurant's needs, focusing on seamless ordering, efficient delivery tracking through integration of PayPal for payments, and an interactive map system.
- To streamline the ordering process, enhance user experience, ensure food safety, and facilitate timely delivery of meals to customers' desired locations.

1.4. Scope and Limitation

1.4.1. Scope of the Project:

The Project aims to create a user-friendly food delivery app where customers can easily browse menus, place orders, track deliveries in real-time, and make payments through PayPal. We'll also include a map feature to help delivery drivers navigate efficiently. The app will be designed to be easy for both customers and restaurant staff to use, ensuring a smooth ordering and delivery process.

1.4.2. Limitations of the Project:

The limitations of this project are as follows:

- Dependency on PayPal integration for payment processing, which may restrict users who prefer alternative payment methods.
- Limited Language Support
- Limited customization options for users, such as dietary preferences or special requests, which may require additional development effort to implement.

1.5. Development Methodology

In crafting the development methodology for our food delivery system, we chose to adopt the Waterfall Model, a traditional and sequential approach that aligns with the project's well-defined objectives. While it's not as commonly used in modern, dynamic projects, it can be adapted for certain contexts.

1. **Planning and Requirements Gathering:** Objectives and features of the food

delivery app are defined based on user needs and market trends. Requirements are collected through stakeholder interviews and market research to establish a clear project scope.

2. **Design and Prototyping:** Wireframes and mockups are created to visualize the app's user interface and experience. The app architecture, database schema, and system components are designed. Key features are prototyped to validate design decisions and gather feedback.
3. **Development and Implementation:** The backend infrastructure is built using Node.js and Express.js, integrating with Mongo DB for data storage. Frontend components are developed using React.js for dynamic user interfaces. Core features such as user authentication, order management, and menu display are implemented. Third-party services like Render.com are integrated for deployment and hosting.
4. **Testing and Quality Assurance:** Unit tests, integration tests, and end-to-end tests are conducted to ensure functionality, performance, and security. Usability testing is performed, and feedback from beta testers is collected to address any issues or usability concerns.
5. **Deployment and Hosting:** The food delivery app is deployed on Render.com for scalable and reliable hosting. Automatic scaling and monitoring are configured to handle fluctuations in traffic and ensure optimal performance. SSL certificates are set up for secure communication between clients and servers.
6. **User Training and Documentation:** User training materials and documentation are provided to guide users on using the app effectively. Support channels such as FAQs, help guides, and customer support are available to assist users with any questions or issues.

7. **Maintenance and Updates:** The app's performance and security are monitored, and patches and updates are applied as needed. User feedback and analytics data are used to identify areas for improvement and new features. Regular updates and enhancements are made to meet evolving user needs and market demands.

CHAPTER 2

Background Study and Literature Review

2.1. Background Study

The online food ordering system is one of the latest services most fast food restaurants in the western world are adopting. With this method, food is ordered online and delivered to the customer. This is made possible through the use of electronic payment system. Customers pay with their credit cards, although credit card customers can be served even before they make payment either through cash or cheque. So, the system designed in this project will enable customers go online and place order for their food. Due to the great increase in the awareness of internet and the technologies associated with it, several opportunities are coming up on the web. So many businesses and companies now venture into their business with ease because of the internet.

One of such business that the internet introduced is an online food ordering system. In today's age of fast food and take out, many restaurants have chosen to focus on quick preparation and speedy delivery of orders rather than offering a rich dining experience. Until recently, most of this delivery orders were placed over the phone, but there are many disadvantages to this system. It is possible for anybody to order any goods via the internet and have the goods delivered at his/her doorsteps. But while trying to discuss the transfer method of the goods and services, attention is focused on the payment mode. In other words, how possible is it to pay for goods and services via the internet? This then leads to the discussion of the economic consequences of digital cash. What are the implementations from the view point of economic? Since the world is fast becoming a global village, the necessary tool for this process is communication of which telecommunication is a key player. A major breakthrough is the wireless telephone system which comes in either fixed wireless telephone lines or the Global System of Mobile communication (GSM). What I propose is an online ordering system originally designed for use in college cafeterias, but just as applicable in any food delivery industry.

The main advantage of this system is that it greatly simplifies the ordering process for both

the customer and the restaurant. The system also greatly lightens the load on the restaurants end, as the entire process of taking orders is automated. Once an order is placed on the webpage that will be designed, it is placed into the database and then retrieved, in pretty much real-time, by a desktop application on the restaurants end. Within this application, all items in the order are displayed, along with their corresponding options and delivery details, in a concise and easy to read manner. This allows the restaurant employees to quickly go through the orders as they are placed and produce the necessary items with minimal delay and confusion. The greatest advantage of this system is its flexibility.

2.2. Literature Review

ONLINE FOOD ORDERING SYSTEM (Dikshya Regmi) [3]:Online Food Ordering System is the system where we can order the food item and reserve table through internet with just one click, which can make our daily life easy and faster. Presently, the customers spend an average of 1 hour per day going to the restaurant, selecting their food and paying. Restaurants have the provision of customers making a call to the restaurant in advance to order an item to be ready for them for pick or to be delivered to them. some of the customers don't always get the selection they want because the restaurants run out of certain items or because there is no provision of ordering custom foods. (Regmi, 2017)

CUSTOMER ORDERING SYSTEM (Suriey Tafar) [4]:In an automated food ordering system is proposed which will keep track of user orders smartly. Basically, they implemented a food ordering system for different type of restaurants in which user will make order or make custom food by one click only. The list of selected preordered items shall be shown on the kitchen screen, and when confirmed, order slip shall be printed for further order processing. By means of web application this system was implemented. The front end was developed using html, CSS, bootstrap and at the backend node.js programming and Mango DB database was used. (Tafar, 2013)

ONLINE FOOD DELIVERY APP 'FOODIE' (H. Kumar, M. Jain, and M.S. Bajwa) [6]: The inception of food delivery services traces back to the utilization of the term 'Dabbawalas,' denoting the delivery of lunches to Mumbai's working professionals. This was followed by

the inauguration of the first food delivery service in Northern California in 1995 by 'World Wide Waiter' (Waiter.com), which collaborated with restaurants to transition menus online. Presently, numerous countries have established their own online food ordering applications, facilitating the connection between small restaurants and consumers at home. The past literature extensively explores the integration of mobile applications, with developers employing platforms such as Java and SQL.

AN EMPIRICAL STUDY OF ONLINE FOOD DELIVERY SERVICES FROM APPLICATIONS PERSPECTIVE (R. Ramesh and S. V. Prabhu) [7]: The food industry has been significantly influenced by the rise of e-commerce, with a multitude of food distribution apps and websites reshaping its landscape. Online food ordering, enabled through web pages or mobile applications, has experienced steady growth, transitioning from traditional telephone ordering to digital platforms. As e-commerce continues to advance rapidly, online platforms address previous obstacles such as long loading times and payment security concerns, thereby opening up new opportunities for attracting consumers. In India, consumers are embracing online food ordering with enthusiasm, seeking the same ease and convenience they experience in other online shopping interactions. Recognized names like Swiggy, UberEats, and Zomato are acknowledged for their swift service and diverse culinary offerings, catering to customers' preferences across different regions. Overall, this literature review explores the transformative impact of online food delivery apps on the food industry, providing insights into their significance and implications for both businesses and consumers.

IMPROVING ONLINE FOOD ORDERING AND DELIVERY SERVICE QUALITY BY MANAGING CUSTOMER EXPECTATIONS (A. Bonfanti, C. Rossato, V. Vigolo, and A. Vargas-Sánchez) [10]: The Covid-19 pandemic, starting in 2020, significantly altered customer purchasing behaviors and methods, prompting businesses to confront new challenges. As a response to new laws, regulations, and repeated closures, restaurants and catering businesses have been compelled to adapt their practices. Many have chosen to enhance their offerings by introducing or improving online food ordering and delivery services, including the delivery of raw materials alongside ready-made foods. Consequently, there has been a notable shift towards online food ordering and delivery services worldwide, driven by the necessity to adhere to safety regulations such as social distancing and cashless

transactions. These changes have been fueled by anxiety, fear, and perceived risk of infection among consumers.

FACTORS AFFECTING BEHAVIOURAL INTENTION OF ONLINE FOOD DELIVERY SERVICE CONSUMERS IN KATHMANDU VALLEY (L. Pokhrel and R. Shah) [9]: The Kathmandu Valley has a thriving market for online food delivery, with popular services like Foodmandu, Bhojdeals, Foodmario, and BhokLagyo leading the way. Foodmandu, established in November 2010, was the first to offer meals from many well-known eateries in Nepal (Rajbhandari, 2022). These platforms typically accept orders through websites or digital channels, targeting millennials, who heavily rely on smartphones for online ordering. As a result, providing online delivery becomes a strategic move for restaurants and the food industry to connect with this demographic. Despite limited research on online food delivery intention in Nepal, given its growing popularity, especially in cities like Kathmandu, it's important to understand what factors drive consumers' decisions. Thus, this paper explores the influences on online food delivery intention among consumers in the Kathmandu Valley.

THE NEW NORMAL: THE ADOPTION OF FOOD DELIVERY APPS (N. W. Madinga, J. Blanckensee, L. Longhurst, and N. Bundwini) [8]: The adoption of food delivery apps surged during the COVID-19 pandemic, driven by lockdown restrictions and a shift towards contactless dining. This study investigates the factors influencing this adoption trend among 282 users in South Africa. Using structural equation modeling, it finds that perceived ease of use significantly impacts perceived usefulness and attitudes, which in turn affect continuous intention to use. However, perceived COVID-19 threat does not influence attitudes, and neither education nor age moderate these relationships. These findings offer valuable insights for restaurants and app developers, informing strategic planning to better meet customer needs in evolving circumstances. This study stands out for its use of the technology acceptance model in analyzing the pandemic's impact on food delivery app adoption, shedding light on situational factors like social pressure and convenience.

CHAPTER 3

System Analysis

3.1. System Analysis

System analysis is a critical component of developing Food delivery system. The main problem of this organization is that there is no any digital system for online food ordering. The organization's staff uses pen and paper to take down customers' orders. Restaurant's tables are not easily available especially during peak hours. Customers have to wait in a queue for their payment. The analysis phase typically includes the following steps:

3.1.1. Requirement Analysis

i. Functional Requirements

The functional requirements in the project which provides the overview of how behaves from the perspectives of its users are mentioned below:

- **User Authentication:**

The Web application enables the users to access the system if they are registered but for the new users, they have to register for taking the service of the system. Once registered, the user can access the system by logging in with the proper credentials to their account. Once the user uses the system service, the user can logout.

In the Figure 3.1 which is a use case diagram for Authentication, users need to register by providing essential information like name, email, address and password. If everything in order, users are prompted to log in. Upon successful login with valid credentials, users gain access to the system. However, if registration details are invalid, users cannot register until correct credentials are provided. Similarly, incorrect login details result in denied access, accompanied by a message "Please enter a correct username and password.

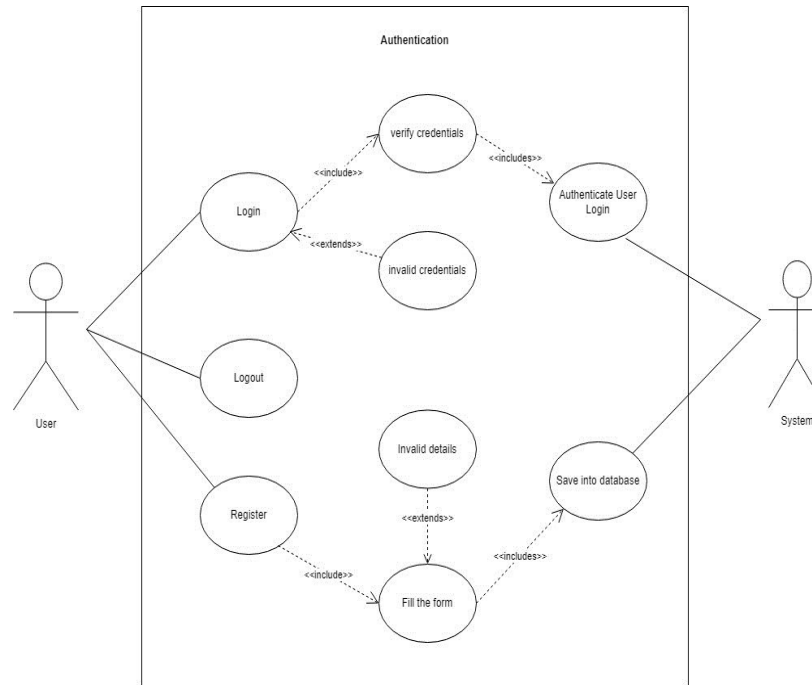


Figure 3.1: Use Case for Authentication

ii. Non-Functional Requirements

The non-functional requirements of our project are:

- **Usability:**

The website is easy to use, with a clear and intuitive interface, fast loading times, and consistent navigation across all pages.

- **Performance:**

The website is responsive and fast, with minimal latency and downtime, even under heavy traffic or high load.

- **Accessibility**

The website is accessible to users with disabilities, complying with web accessibility standards and guidelines, and offering assistive technologies such as screen readers or keyboard navigation.

- **Compatibility:**

The system is compatible with different devices and operating systems to ensure maximum usability for users.

- **Maintainability:**

The website is easy to maintain, with clear documentation, code quality standards, and automated testing and deployment processes.

- **Security:**

The system is secure and protect user data and information, with proper access controls and encryption mechanisms.

3.1.2. Feasibility Analysis

i. Economic Feasibility

The organization has evaluated the cost of software and hardware required for the system including storage of data. Once the software is installed and operated, it will definitely save time of the organization and it does not need additional hardware or software. This would insure nominal expenditure for the organization. Also, since the software is easy to use and do not need skilled manpower to use it. The benefits expected from the system are studied to access the cost due to new system.

ii. Technical Feasibility

The technical issue usually rises during the feasibility stage of the investigation. Online Food Delivery Web App is a system that effectively and efficiently manages the mass numbers of food order at a same time. The current system developed is technically feasible. So, cost of maintenance will reduce. It provides an easy access to the users and it provides the technical guarantee of accuracy and reliability. As all the hardware and software tools required for implementation of the system is under the budget of the organization. The software and hardware requirements for the development of this project are not many and are already available as free as open source. Also, software can be easily upgraded to accomplish specific user requirements. The programming language and database are open source. The system is based on a real-time database that provides real-time updates to the information change in the database. They all are easily available to the public. This system allows for the feasible work operation of the restaurant, where the data may change constantly and assures for the availability of real-time updated data. Hence

technically there is no limitation for the development of the software.

iii. Operational Feasibility

The project's workflow and data collection processes seem manageable, with effective data preprocessing and feature extraction methods identified. We have identified integration opportunities and the proposed system will work in various sectors ensuring that the system can seamlessly fit into existing workflows. The success of the project depends on the ability of the system to collect and classify the news articles from various sources accurately and efficiently. The system's user interface is simple to use and accessible, so this web application is operationally feasible.

iv. Schedule Feasibility

We have calculated the time required for us to complete this project. Time has been separated for each of the work that is to be included into the project so that everything can be compared in time. Hence, the project has been developed according to the following time schedule to make our application schedule feasible.

Table 3.1: Schedule Table

S.No.	Activities	Start Date	End Date	Duration
1.	Project Initiation and Planning	September 12, 2023	September 17, 2023	5
2.	Research and Requirement Gathering	September 18, 2023	October 1, 2023	13
3.	System Design and Basic Implementation	October 2, 2023	October 22, 2023	20
4.	Final Review and Testing	December 28, 2023	January 8, 2024	11
5.	Project Presentation and Submission	January 9, 2024	January 12, 2024	3
6	Documentation	September 12, 2023	January 12, 2024	12

3.1.3. Analysis

i. Object Modeling using Class diagram

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity. Class diagrams are useful in all forms of object-oriented programming (OOP).

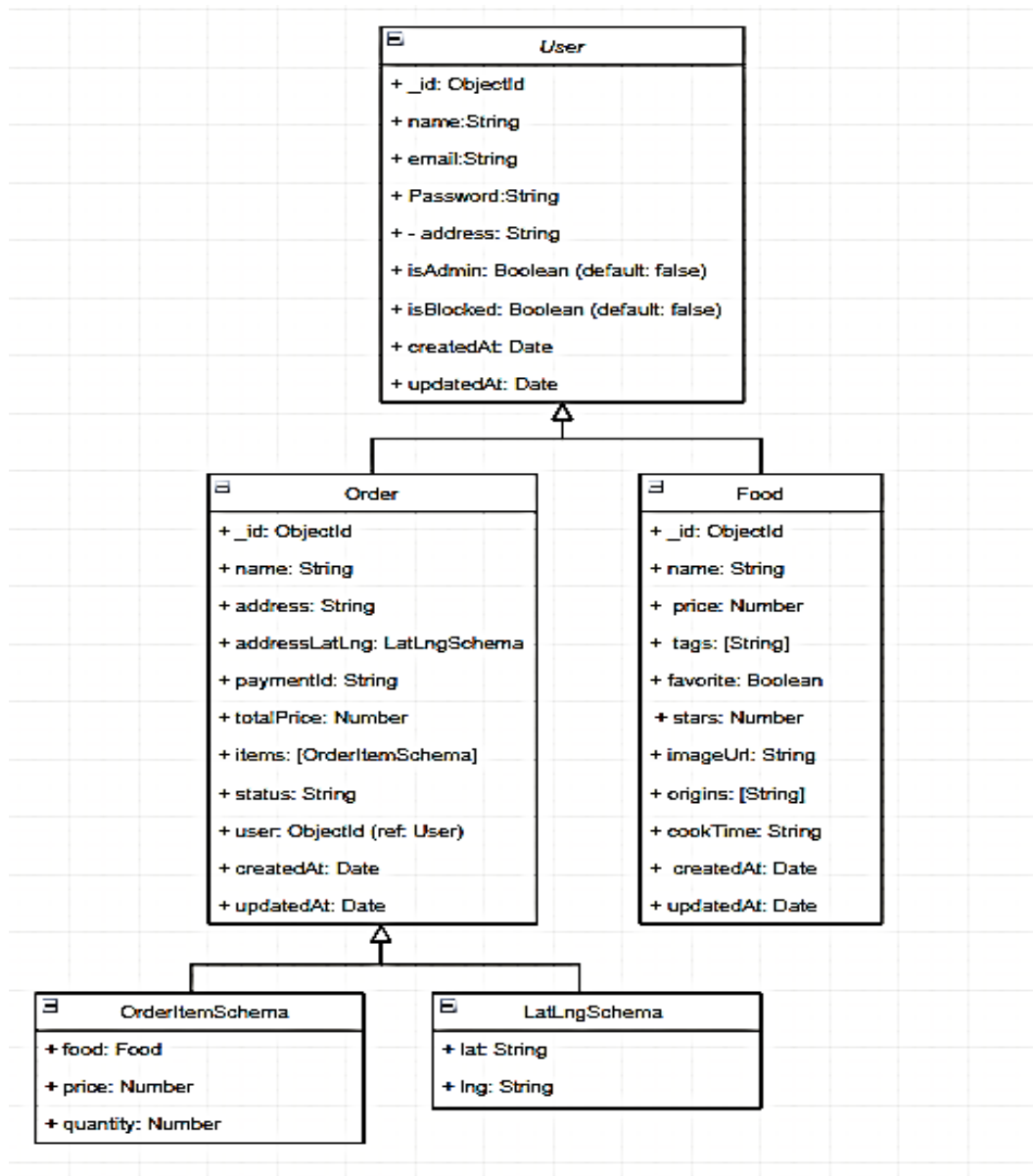


Figure 3.2.: Class Diagram of Food Delivery System

ii. Dynamic Modeling using Sequence diagram

Sequence diagrams for the food delivery system are a type of Dynamic Modeling tool which visually capture the message flow between objects in a system over time. The sequence diagram offers a visual representation of the dynamic behavior of our system during a specific user scenario, helping to understand the interaction flow between various components.

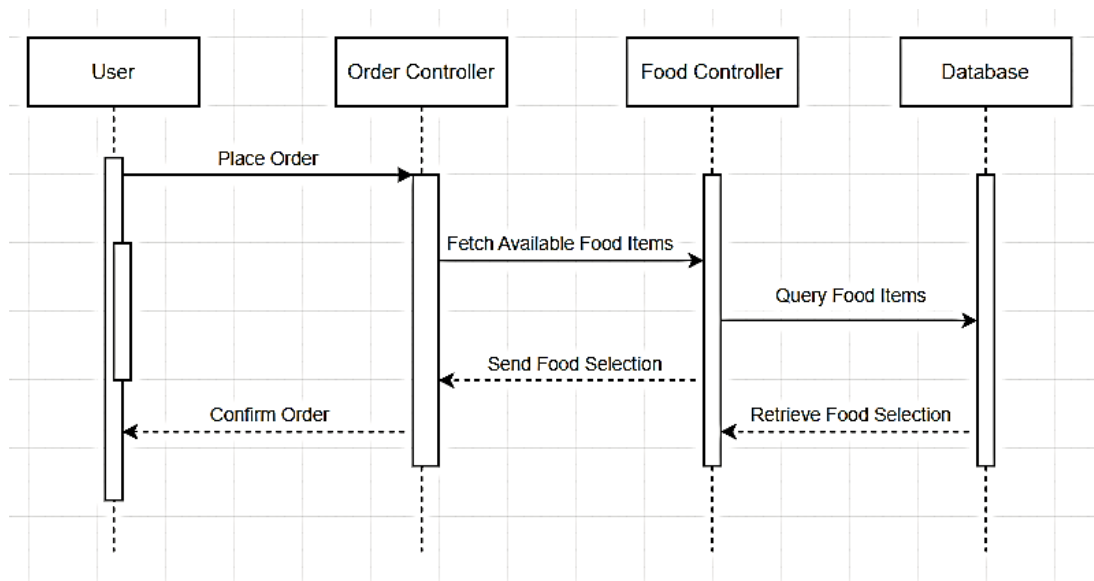


Figure 3.3: Sequence Diagram of Food Delivery System

iii. Process Modeling using Activity Diagram

Activity diagrams act like blueprints for processes, depicting the flow of activities and decisions from start to finish. Similar to a recipe, they map out each step, including potential branches and alternative paths, providing a clear understanding of how a process unfolds.

In our food delivery app's activity diagram, users start by registering or logging in with their chosen username and password. Once logged in, they can navigate the app either as an admin or a customer. Admins have extra privileges like managing food items, while customers can browse the menu and place orders. After selecting items, customers proceed to checkout, and the system processes the order, preparing it for delivery or

pickup. Finally, completed orders are delivered to the customer's location or made available for pickup, completing the transaction. This simple process captures the key steps involved in using our food delivery app, from registration to order fulfillment.

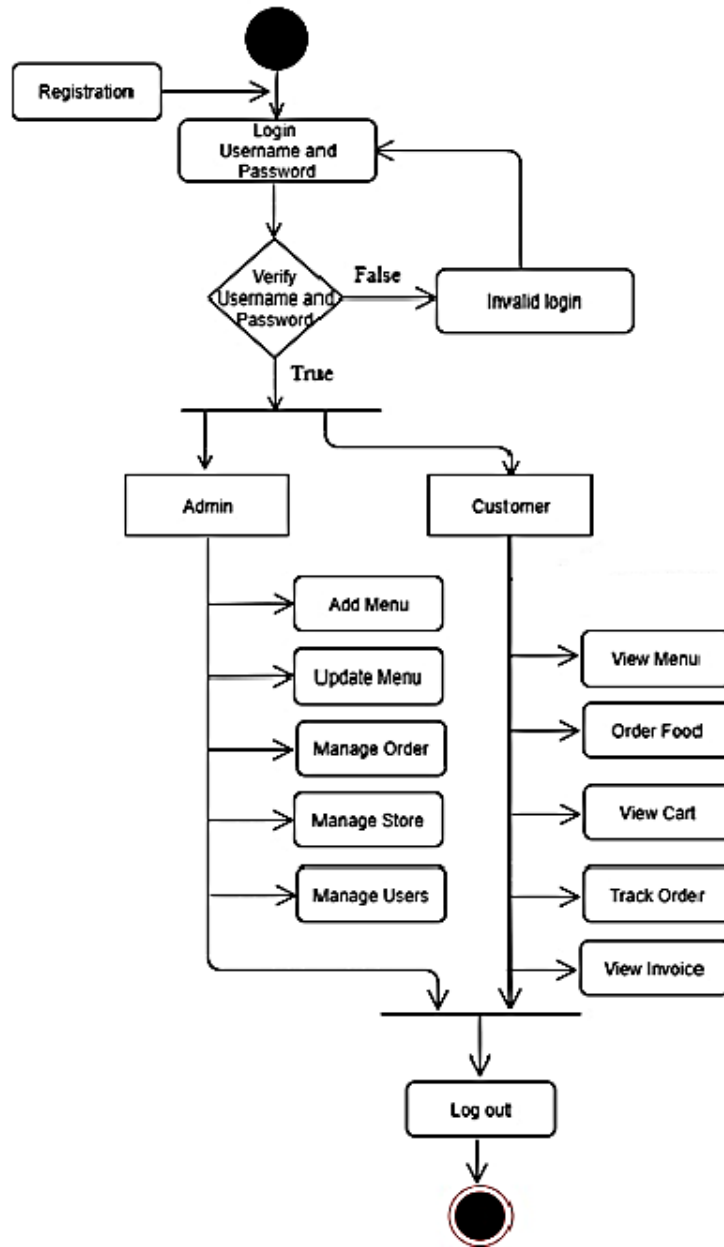


Figure 3.4: Activity Diagram for Food Delivery System

CHAPTER 4

System Design

4.1. Design

System design is the process of defining the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. One of the main components of software design is the software requirement analysis.

4.1.1. Refinement of Class Diagram

One of the main components of software design is the software requirement analysis. In this project there is one database used for store the information of customer who have register their account in system.

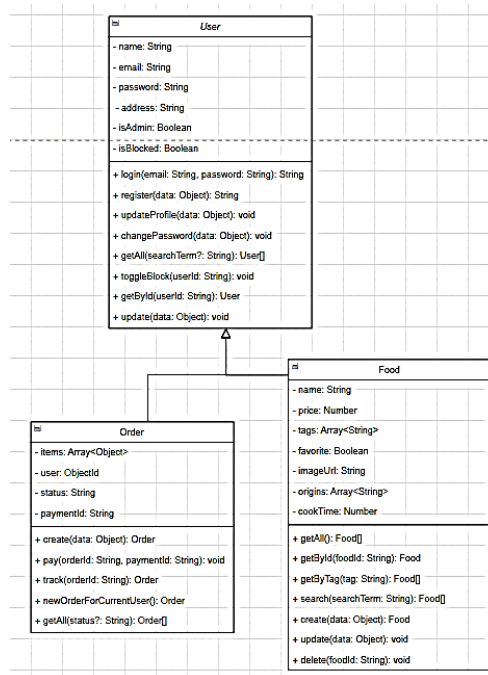
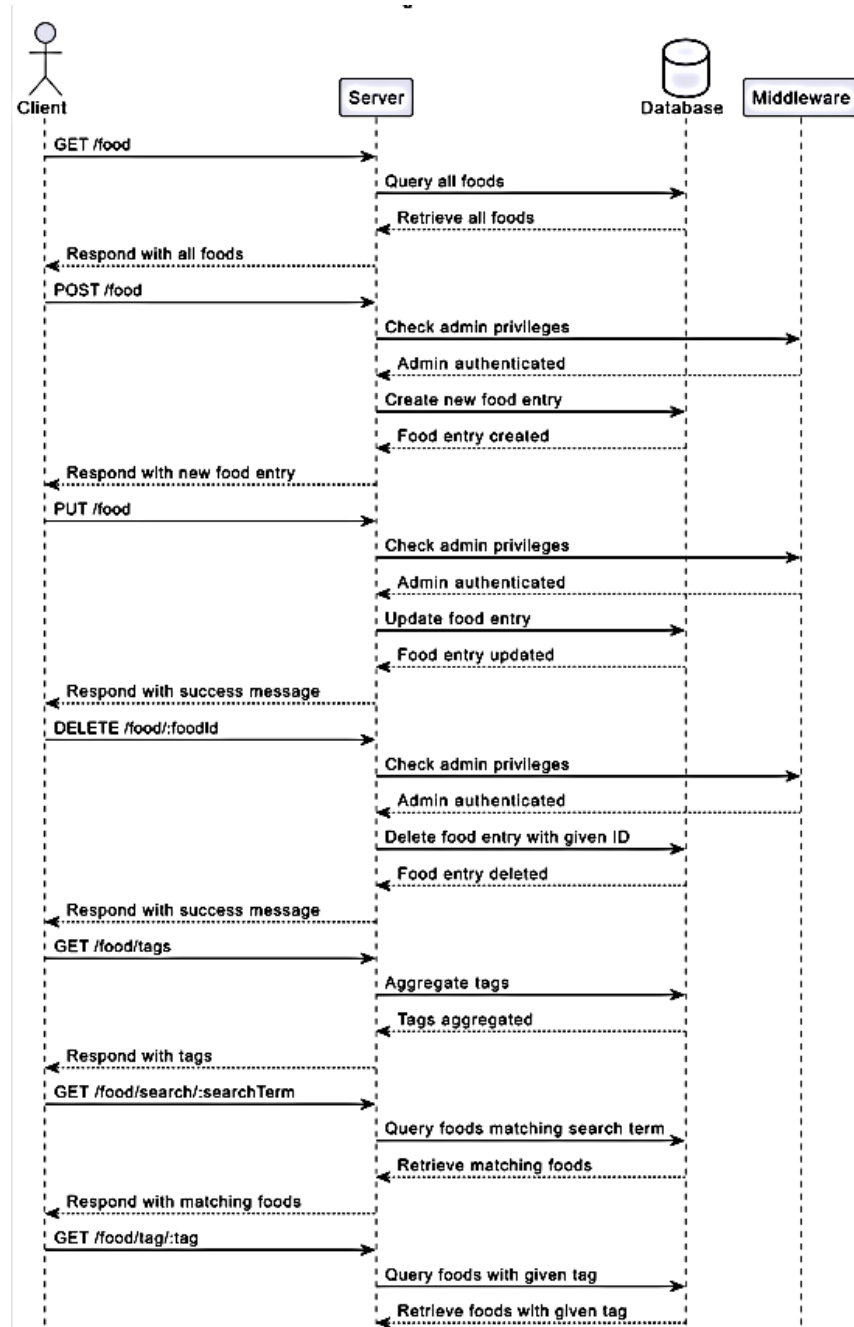


Figure 4.1: Refined Class Diagram

4.1.2. Refinement of Sequence Diagram

A refined sequence diagram of Food delivery system provides a detailed view of how different parts of a system interact. It shows the specific order in which these interactions happen and the messages that are exchanged between different components or objects in the system. It's like a step-by-step visual story of how things happen and how components communicate with each other.



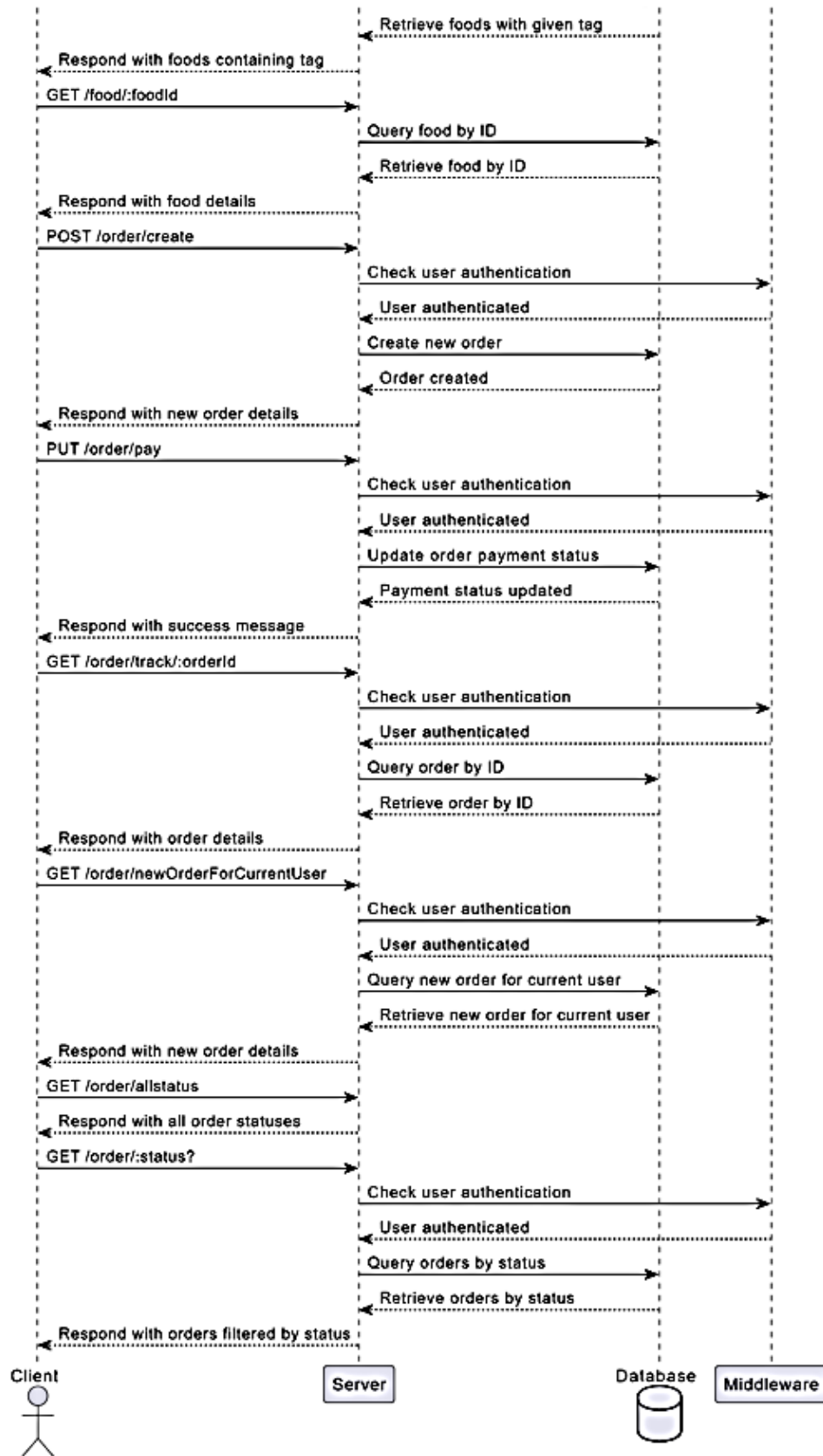


Figure 4.2: Refined Sequence Diagram

In the Figure 4.9, This sequence diagram illustrates how our food delivery app works behind the scenes. Imagine you're using the app and you want to browse the menu. When you open the app, it sends a request to the server to get all the available food items. The server then checks with the database, gathers all the food options, and sends them back to your app so you can see them. Now, let's say you're adding a new item to the menu. You select the option on the app, and it tells the server to add it. The server first checks if you're allowed to add items (like if you're an admin). If everything checks out, it adds the new item to the database and lets you know it's done. Similarly, when you're updating an item or deleting one, the app follows a similar process. This diagram essentially shows the conversation between your app and the server, highlighting how they work together to manage the menu and orders efficiently.

4.1.3. Refinement of Activity Diagram

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities model can be sequential and concurrent. In both cases an activity diagram will have a beginning and an end.

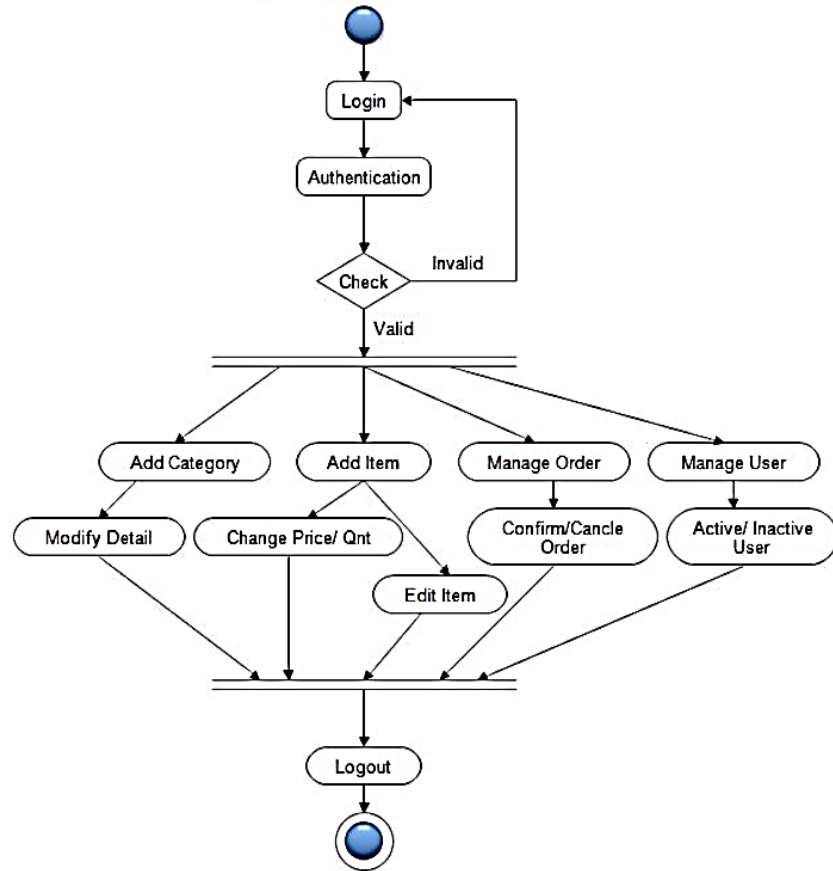


Figure 4.3: Activity Diagram for Admin

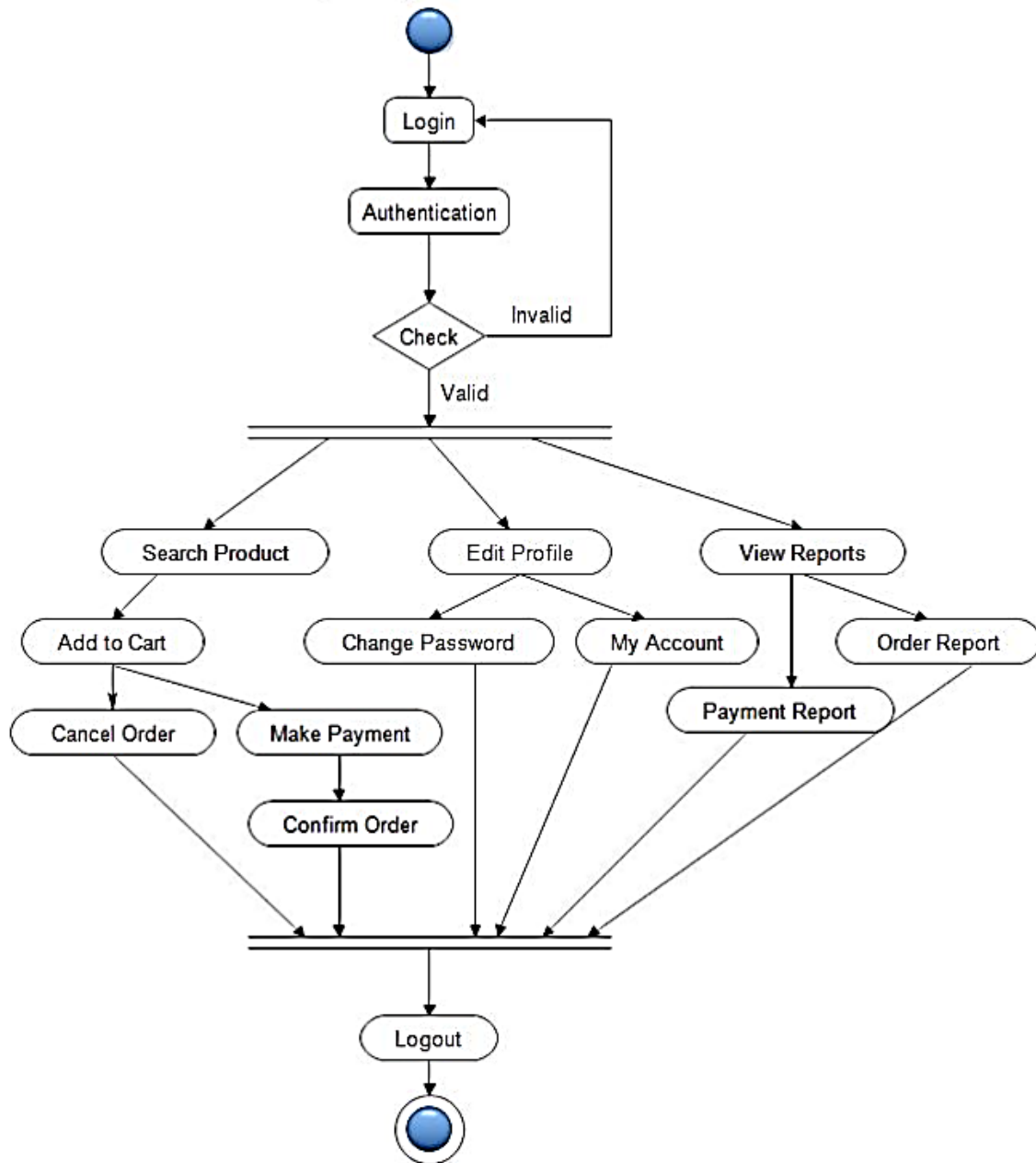


Figure 4.4: Refined Activity Diagram for User

In the Figure 4.10, A refined activity diagram of food delivery system gives a detailed picture of the steps and activities in a process. It goes beyond the basic diagram by adding more details and processes, providing a deeper insight into how the activities unfold. In the above refined activity diagram, the process begins with the user, who has the option to either register for an account or proceed as a guest. If the user opts for registration, they are required to provide account information and set up a password. For users with existing accounts, the login step follows, where they must enter their username and password. The system then validates the entered credentials for accuracy. After successful login, users can either enter the URL of a specific product or browse the store's product selection. The system responds by retrieving information about the product based on the user's input.

4.1.4. Component Diagram

A component diagram for fake news detection is used to break down a large object-oriented system into the smaller components, so as to make them more manageable.

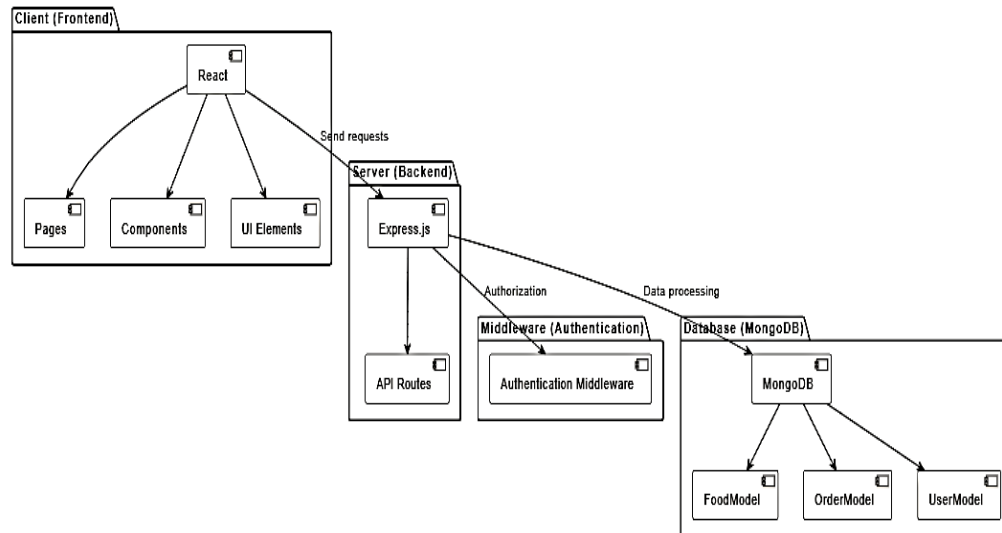


Figure 4.5: Component Diagram

From the Figure 4.11, It illustrates the component diagram of food delivery system, the frontend and backend components of the food delivery application are depicted visually, illustrating their roles and interactions. The Client (Frontend) component represents the user interface developed using React, responsible for user interaction and presentation of data. It communicates with the Server (Backend) component, which comprises two main sub-components: Middleware (Authentication) and Database (MongoDB).

The Middleware component handles user authentication and authorization, ensuring secure access to protected routes. It verifies user credentials and manages user sessions. On the other hand, the Database component represents the MongoDB database, which stores essential collections such as FoodModel, OrderModel, and UserModel. These collections are accessed and manipulated by the backend server to retrieve and store application data.

Overall, the component diagram provides a clear overview of how the frontend and backend components interact to deliver the functionality of the food delivery application. It showcases the flow of data and control between different parts of the system, highlighting the separation of concerns and the roles each component plays in the overall architecture.

4.1.5. Deployment Diagram

The deployment diagram of the food delivery system portrays the static deployment view of a system. It involves the nodes and their relationships.

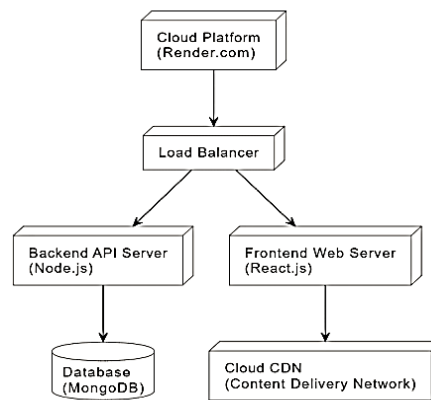


Figure 4.6: Deployment Diagram of Food Delivery System

From the Figure 4.12, Deployment Diagram of food delivery system, In this setup, the food delivery app is hosted on Render.com, providing the necessary foundation for its operation. The Load Balancer ensures that traffic is evenly distributed among multiple backend API servers, making the app more scalable and available.

The Backend API Server, powered by Node.js, handles the core logic of the app, managing data and communicating with the MongoDB database where user details and orders are stored. At the same time, the Frontend Web Server, built with React.js, delivers the app's interface to users and handles their interactions.

The MongoDB Database stores crucial data for the app, ensuring efficient retrieval and management. Plus, a Cloud CDN speeds up the delivery of static content, like images and scripts, by caching them closer to users, leading to faster load times and a smoother user experience.

Overall, this deployment setup leverages cloud technology to make the food delivery app scalable, fast, and reliable, meeting the needs of today's dynamic web applications.

CHAPTER 5

Implementation and Testing

5.1 Implementation

5.1.1. Tools Used

i. Visual Studio Code:

VS Code IDE is used for development of the project. Visual Studio Code (VS Code) is a free source-code editor developed by Microsoft for Windows, Linux, and macOS. It offers a wide range of features for developers, including support for debugging, syntax highlighting, intelligent code completion, code refactoring, and Git integration.

One of the key features of Visual Studio Code is its extensibility, which allows developers to customize and enhance their coding experience through various extensions available in the Visual Studio Code Marketplace. These extensions can add support for additional programming languages, provide new themes, and integrate with other tools and services, and more.

ii. Front End Tools:

1. HTML

HTML serves as the backbone of our project, providing the foundation for structuring web pages and applications. It empowers us to organize content effectively using various elements like headings, paragraphs, lists, and forms. Through HTML, we define the layout of our web pages, ensuring they are well-structured and easy to navigate. With its versatile range of tags and elements, HTML plays a pivotal role in shaping the overall user experience within our project. In our food delivery app, HTML is used to create the layout and structure of the different pages displayed to the users, ensuring a clear and organized presentation of information.

2. CSS

CSS is instrumental in enhancing the visual appeal and consistency of our project. By utilizing CSS, we can effortlessly style HTML and XML documents, creating visually captivating websites. With CSS, we establish rules to control the

presentation of elements, including aspects such as typography, color schemes, spacing, and layout. This allows us to craft a cohesive and polished design across our web pages, ensuring a seamless and engaging user experience.

3. React Js

ReactJS, with its component-based architecture and efficient virtual DOM, is an ideal framework for developing food delivery web applications. By breaking down the UI into reusable components, React enables easier management and maintenance of the codebase, while its dynamic UI updates ensure smooth user interactions such as adding items to the cart or updating delivery details without reloading the entire page. State management capabilities, whether using React's built-in features or external libraries like Redux, facilitate handling user authentication, shopping cart contents, and order details. Integration with backend APIs for functionalities such as user authentication and order processing is simplified through React's support for making HTTP requests. Routing, facilitated by libraries like React Router, allows defining routes for different pages, ensuring proper navigation within the app.

iii. Backend Tools

1. Express

Express.js played a crucial role as the backend tool by serving various functionalities efficiently. Express.js, a popular Node.js framework, handled incoming requests and served corresponding responses seamlessly. It facilitated the creation of API routes for managing food items, user orders, and user authentication. Additionally, Express.js enabled the integration of middleware functions for tasks such as authorization and error handling, enhancing the overall robustness of the application. Through its intuitive routing system and middleware support, Express.js effectively managed the flow of data between the client-side and server-side components, ensuring smooth communication and interaction within the application. Overall, Express.js served as a reliable and versatile backend tool, contributing significantly to the functionality and performance of our project.

2. Node

Node.js played a pivotal role as the backend tool, providing the runtime environment necessary to execute JavaScript code on the server-side. By leveraging Node.js, our application was able to handle server-side operations efficiently, facilitating tasks such as processing HTTP requests, accessing databases, and managing file systems. This allowed us to build a robust and scalable backend infrastructure that powers the functionality of our web application.

iv. Database Tools

1. Mongo dB

MongoDB played a crucial role as the database management system. It stored various types of data, such as user information and food orders, in a structured and efficient manner. This facilitated seamless communication between our Node.js backend and the database, enabling smooth data retrieval and manipulation. Additionally, MongoDB's scalability allowed our application to accommodate increasing data volumes without sacrificing performance, ensuring a reliable and responsive user experience.

5.1.2. Implementation Details of Modules

i. Registration Module

Here Authentication module, which is implemented using Node.js and Express.js, facilitates user authentication and authorization in our food delivery application. It consists of several routes and middleware functions to handle various authentication-related tasks. Users can register by providing their name, email, password, and address. Upon registration, the provided information is stored securely in the database after hashing the password for security. For logging in, users provide their email and password, which are then verified against the stored credentials in the database.

```

router.post(
  '/register',
  handler(async (req, res) => {
    const { name, email, password, address } = req.body;

    const user = await UserModel.findOne({ email });

    if (user) {
      res.status(BAD_REQUEST).send('User already exists, please login!');
      return;
    }

    const hashedPassword = await bcrypt.hash(
      password,
      PASSWORD_HASH_SALT_ROUNDS
    );

    const newUser = {
      name,
      email: email.toLowerCase(),
      password: hashedPassword,
      address,
    };

    const result = await UserModel.create(newUser);
    res.send(generateTokenResponse(result));
  })
);

```

Figure 5.1: Implementation of Registration Module

ii. Login Module

The login module in the system lets registered users log in by entering their username and password. The "User Login" module manages this process, checking and verifying login form submissions. If the credentials are valid, users are redirected to the home page. The "User Logout" module allows users to log out securely when they're done using their accounts.

```

router.post(
  '/login',
  handler(async (req, res) => {
    const { email, password } = req.body;
    const user = await UserModel.findOne({ email });

    if (user && (await bcrypt.compare(password, user.password))) {
      res.send(generateTokenResponse(user));
      return;
    }

    res.status(BAD_REQUEST).send('Username or password is invalid');
  })
);

```

Figure 5.2: Implementation of Login Module

iii. Admin Module

The Certain routes are restricted to users with administrative privileges. Middleware functions are used to verify if a user has admin rights before granting access to these

routes. Admins have additional capabilities, such as retrieving all users' information or toggling user block status.

```
import { UNAUTHORIZED } from "../constants/httpStatus.js";
import authMid from "./auth.mid.js";
const adminMid = (req, res, next) => {
  if (!req.user.isAdmin) res.status(UNAUTHORIZED).send();
  return next();
};
export default [authMid, adminMid];
```

Figure 5.3: Implementation of View Module for Admin

iv. Food (Items) Module

The food module in our Express.js app does a few key things:

1. Fetch All Foods: It gets a list of all available food items from the database.
2. Add New Food: Admins can add new dishes to the menu by providing details like name, price, and tags. These details are saved in the database.
3. Update Food: Admins can modify existing dishes by specifying the changes they want to make, like updating the price or tags.
4. Delete Food: Admins can also remove dishes from the menu by specifying the ID of the dish they want to delete.
5. Get Food Tags: It collects all the unique tags used for categorizing dishes and provides a count of each tag.
6. Search Foods by Name: Users can search for specific dishes by typing in a keyword. The module finds matches based on the dish names.
7. Get Foods by Tag: Users can explore dishes by selecting a particular tag, like "vegetarian" or "Italian."
8. Get Food by ID: If users want more details about a specific dish, they can retrieve it by its unique ID.

In essence, this module manages everything related to food items, from adding and updating dishes to searching and deleting them, providing a smooth experience for both admins and users.

```
router.post(
  "/",
  admin,
  handler(async (req, res) => {
    const { name, price, tags, favorite, imageUrl, origins,
    cookTime } =
      req.body;

    const food = new FoodModel({
      name,
      price,
      tags: tags.split ? tags.split(",") : tags,
      favorite,
      imageUrl,
      origins: origins.split ? origins.split(",") : origins,
      cookTime,
    });

    await food.save();

    res.send(food);
  })
).
```

Figure 5.4: Implementation of Food Module

v. Order Module

The order module helps users in our app to do several things:

1. **Create Order:** Users can make new orders by adding items they want to buy. If the cart is empty, it tells them so.
2. **Pay for Order:** After selecting items, users can pay for their orders. Once paid, it updates the order status and sends an email receipt.
3. **Track Order:** Users can check the status of their orders by providing their order ID.
4. **Get New Order:** Users can see their new orders to review or modify them.

5. Get Order Statuses: It shows all possible order statuses.
6. Get Orders by Status: Users can filter their orders by status, like "NEW" or "PAYED," or see all their orders. Admins can view orders from all users.

So, it's a handy tool for users to manage their orders smoothly, from creation to payment and tracking, making their shopping experience easier.

```
router.post(
  '/create',
  handler(async (req, res) => {
    const order = req.body;

    if (order.items.length <= 0) res.status(BAD_REQUEST).send('Cart
    Is Empty!');

    await OrderModel.deleteOne({
      user: req.user.id,
      status: OrderStatus.NEW,
    });

    const newOrder = new OrderModel({ ...order, user: req.user.
    id });
    await newOrder.save();
    res.send(newOrder);
  })
);
```

Figure 5.5: Implementation of Order Module

vi. Image Module

This module handles image uploads to our app. Users can upload images for various purposes, such as profile pictures or product images. Here's how it works:

1. Image Upload: Users can upload images through a POST request to the specified endpoint.
2. Authorization: Only admins are allowed to upload images, ensuring control over what gets uploaded to the app.
3. Image Processing: The uploaded image is processed using multer middleware, making it available for further handling.
4. Cloud Storage: The image is then uploaded to Cloudinary, a cloud storage service, using the provided configuration.
5. Response: Upon successful upload, the module sends back the URL of the uploaded image, allowing users to access it from the app.

Overall, this module streamlines the process of uploading images to our app, ensuring proper authorization and seamless integration with cloud storage for efficient handling of images.

```
router.post(
  '/',
  admin,
  upload.single('image'),
  handler(async (req, res) => {
    const file = req.file;
    if (!file) {
      res.status(BAD_REQUEST).send();
      return;
    }

    const imageUrl = await uploadImageToCloudinary(req.file?.buffer);
    res.send({ imageUrl });
  })
);

const uploadImageToCloudinary = imageBuffer => {
  const cloudinary = configCloudinary();

  return new Promise((resolve, reject) => {
    if (!imageBuffer) reject(null);

    cloudinary.uploader
      .upload_stream((error, result) => {
        if (error || !result) reject(error);
        else resolve(result.url);
      })
      .end(imageBuffer);
  });
};
```

Figure 5.6: Implementation of Order Module

vii. Payment Module

In our Create React App, we use the PayPal SDK to integrate PayPal payment functionality seamlessly. Here's a simple explanation.

1. Setup: We provide the PayPal client ID in the PayPalScriptProvider component to initialize the PayPal SDK.
2. Buttons: We create PayPal payment buttons using the PayPalButtons component. These buttons are displayed on the checkout page for users to make payments.
3. Payment Handling: When a user clicks on the PayPal button to make a payment, a PayPal order is created using the createOrder function. This includes details like the currency and total amount.
4. Approval: After the user approves the payment on the PayPal website, the

onApprove function is called. Here, we capture the payment and process it on our backend.

5. Error Handling: If any errors occur during the payment process, such as payment failure or capture failure, the onError function handles them by displaying an error message to the user.

```
export default function PaypalButtons({ order }) {  
  return (  
    <PayPalScriptProvider  
      options={{  
        clientId:  
          "ARPPwF7QEW_y-n8-APBz8_VjUIwoAEFnVIbq81U8I983TyR9NtRD_KPPerY0o334pADMFrSQ3fbJeqcb",  
      }}  
    >  
      <Buttons order={order} />  
    </PayPalScriptProvider>  
  );  
}
```

Figure 5.7: Implementation of Order Module

viii. Map Module

1. Setting up the Map: We use a library called react-leaflet to create our map in a React component. We set the initial view of the map and choose how users can interact with it.
2. Displaying the Map Data: We use a component called TileLayer to show the actual map data. In our case, we're using map tiles from OpenStreetMap.
3. Handling User Interactions: We have a button that users can click to find their location. When they click it, we use the browser's geolocation feature to determine their location and show it on the map.
4. Adding a Marker: Once we have the location, we place a marker on the map to represent it. Users can also drag this marker around to adjust the shipping location.
5. Customizing the Marker: We make the marker look nicer by using a custom icon instead of the default one provided by the map library.
6. Updating the Location: As users interact with the map, we keep track of the location using state in our React component. This allows us to update the marker's position and handle any changes made by the user.

```

export default function Map({ readonly, location, onChange }) {
  return (
    <div className={classes.container}>
      <MapContainer
        className={classes.map}
        center={[0, 0]}
        zoom={1}
        dragging={!readonly}
        touchZoom={!readonly}
        doubleClickZoom={!readonly}
        scrollWheelZoom={!readonly}
        boxZoom={!readonly}
        keyboard={!readonly}
        attributionControl={false}
      >
        <TileLayer url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png" />
        <FindButtonAndMarker
          readonly={readonly}
          location={location}
          onChange={onChange}
        />
      </MapContainer>
    </div>
  );
}

```

Figure 5.8: Implementation of Order Module

5.2. Testing

5.2.1. Test Cases for Unit Testing

Following are the tables representing test cases of unit testing:

Table 5.1: Test Case for Registration

ID	Test Case Description	Test Data	Expected result	Actual Result	Test Result
R_1	User enters valid signup information	Username: hello Email: hello@gmail.com Name: Hello Password: asdf@qwer Confirm Password: asdf@qwer	Redirects to Login Page	Redirected to Login page	Pass

R_2	User enters different password and confirm password	Username: hello Email: hello@gmail.com Name: World Password: asdf@qwer Confirm Password: asdfe	Display Error Message	“User name or password does not match”	Pass
R_3	User enters existing username		Display Error Message	“A user already exists”	Pass

In above test case, the system's signup functionality was tested with three scenarios (R_1, R_2, and R_3). In R_1, a user successfully signed up with valid information, leading to a redirection to the login page as expected. R_2 ensured the system correctly identified mismatched passwords and displayed the appropriate error message, ensuring data integrity. Lastly, R_3 confirmed that when attempting to sign up with an existing username system accurately flagged the duplication, displaying the relevant error message. All test cases passed, affirming the system's robustness in handling signup processes.

Table 5.2: Test Case for Login

ID	Test Case Description	Test Data	Expected result	Actual Result	Test Result
L_1	User enters valid login information	Username: hello Password: hello12345	Redirects to Homepage	Redirected to Homepage	Pass
L_2	User enters invalid login information	Username: hello Password: 12345	Display Error Message	“Username or password is invalid”	Pass

L_3	User leaves the login fields empty	Username: hello Password:	Display Error Message	"This field is required"	Pass
-----	------------------------------------	------------------------------	-----------------------	--------------------------	------

In the above test case, the system's login functionality was thoroughly tested with three scenarios (L_1, L_2, and L_3). In L_1, a user successfully logged in with valid credentials, resulting in a seamless redirection to the homepage, as expected. L_2 examined the system's response to invalid login credentials, where it accurately displayed an error message prompting the user to enter correct information, thus maintaining security measures.

Similarly, in L_3, the system appropriately detected empty login fields and prompted the user with an error message, ensuring comprehensive validation. All test cases passed, affirming the system's reliability in facilitating secure login procedures.

5.2.2. Test Cases for System Testing

Table 5.3: Test Case for System Testing

ID	Test Case Description	Test Data	Expected result	Actual Result	Test Result
1	Add Item to Cart	Item details (name, quantity, price)	Item is added to the user's cart	As Expected	Pass
2	Remove Item from Cart	Item details (name, quantity)	Item is removed from the user's cart	As Expected	Pass
3	Place Order	User details, Cart contents	Order is placed successfully	As Expected	Pass

4	View Order History	User's order history	User can view their past orders	As Expected	Pass
5	Update User Profile	User details (name, email, password)	User profile is updated successfully	As Expected	Pass
6	Search for Items	Search query	Relevant items are displayed	As Expected	Pass
7	View Item Details	Item ID	Item details are displayed	As Expected	Pass

5.3. Result Analysis

1. User and Admin Order Placement:

- **User Perspective:** The food delivery app offers a user-friendly interface that simplifies the ordering process for customers. Users can easily navigate through the app, browse diverse food options, and add desired items to their carts. With intuitive features like order customization and secure checkout, users can swiftly place orders with confidence.
- **Admin Perspective:** From an administrative standpoint, the app empowers admins with robust tools for efficient order management. Admins can monitor incoming orders in real-time, track order statuses, and manage inventory effectively. The admin dashboard provides comprehensive insights into order volumes, popular items, and customer preferences, enabling admins to make data-driven decisions to optimize service quality and enhance customer satisfaction.

2. Payment Integration:

- The food delivery app seamlessly integrates multiple payment options, ensuring a hassle-free transaction experience for users. With support for credit/debit cards, digital wallets, and PayPal, users have the flexibility to choose their preferred payment method. The secure payment gateway encrypts sensitive information, safeguarding user data and instilling trust in the

platform.

3. Map Features for Selecting Delivery Location

- **Interactive Mapping:** Leveraging the advanced mapping capabilities of Leaflet, the app provides users with an interactive map interface for selecting delivery locations. Users can explore nearby restaurants, visualize delivery areas, and pinpoint their desired delivery points with precision.
- **Location Selection:** The map feature enhances user convenience by enabling seamless location selection. Users can effortlessly set their delivery addresses by interacting with the map interface, ensuring accurate and efficient order delivery.
- **Enhanced User Experience:** By integrating the map feature into the app, users enjoy an enhanced ordering experience with intuitive navigation and visual cues. The map interface adds a layer of interactivity to the ordering process, making it more engaging and user-friendly.

4. Order Tracking via Mailgun Integration

- The app leverages Mailgun integration to provide users with comprehensive order tracking capabilities. Through automated email notifications, users receive timely updates on order status, estimated delivery times, and delivery confirmations. This proactive communication keeps users informed and reassured throughout the order fulfillment journey.
- **Enhanced Communication:** By incorporating Mailgun integration, the app facilitates seamless communication between users and the platform. Users can easily track their orders, resolve queries, and provide feedback, fostering a positive and interactive user experience.

5. Comprehensive Order Management Features:

- The app offers a suite of order management features designed to streamline operations for both users and admins. Users can access their order history, review past transactions, and repeat orders effortlessly. Admins benefit from advanced analytics, reporting tools, and customer feedback mechanisms to optimize service delivery and drive business growth.

- **Data-driven Insights:** With robust order management capabilities, admins gain valuable insights into user behavior, market trends, and operational performance. These insights empower admins to make informed decisions, refine business strategies, and enhance overall service quality.

In conclusion, the food delivery app project excels in delivering a seamless and intuitive ordering experience while empowering admins with powerful tools for effective order management. Through innovative features like payment integration, map-based location selection, order tracking, and comprehensive analytics, the app sets a new standard for convenience, efficiency, and user satisfaction in the food delivery industry.

CHAPTER 6

Conclusion and Future Recommendations

6.1 Conclusion

In conclusion, Restaurant Management System is a web-based technology that aids the restaurant industry in carrying out tasks effectively and efficiently. It aids in managing cash flow for managers. Managers can view analytics data to assess company growth. The manager can control orders and employee schedules by using this system. The full complement is a restaurant management system. It provides access to the Online Order platform, third-party connector's software, and comprehensive CRM solution, which together cover a sizable portion of your restaurant's requirements. They are not the outdated hardware and software sets for restaurants that were previously offered. They are the hottest things around, smooth, manageable, inexpensive, and quick.

In the "Online Food Ordering Project," every effort is made to meet all the demands of the restaurant. Because it is straightforward and adaptable, the project is successful. A novice user may operate it with ease. Any type of restaurant can utilize our software. By automating meal ordering, billing, and inventory control, the restaurant management system assists the restaurant manager in managing the restaurant more successfully and efficiently. The system handles the transaction and stores the data produced. These data will be used to create reports that assist the restaurant manager in making wise business decisions. For example, the manager can decide whether more waiters, delivery men, delivery carts, and cooks are needed based on how many clients will be present during a specific time period. When this project is finished, all security concerns will be resolved. Additionally, a quick and secure authentication process will be used for record maintenance. Because it automatically pulls information about a consumer from the database on subsequent visits, data entry is quick and easy. As a result, our program will undoubtedly succeed in replacing the manual way of storing secure orders.

6.1. Future Recommendations

Although the project has been completed and able to overcome the problem of the study, even if some problems are solved by this project and still problems and requirement are not implemented by this project which can be solved in upcoming future days. Some of the future enhancements of this project are:

- More interactive user interface can be added.
- Module that allows for the printing of sales report, dishes report can be added.
- Module that allows to tracking the customer location for delivery of foods can be added.
- Online Payment System like E-sewa, Khalti, mobile banking can be added.
- Others necessary module can be added.

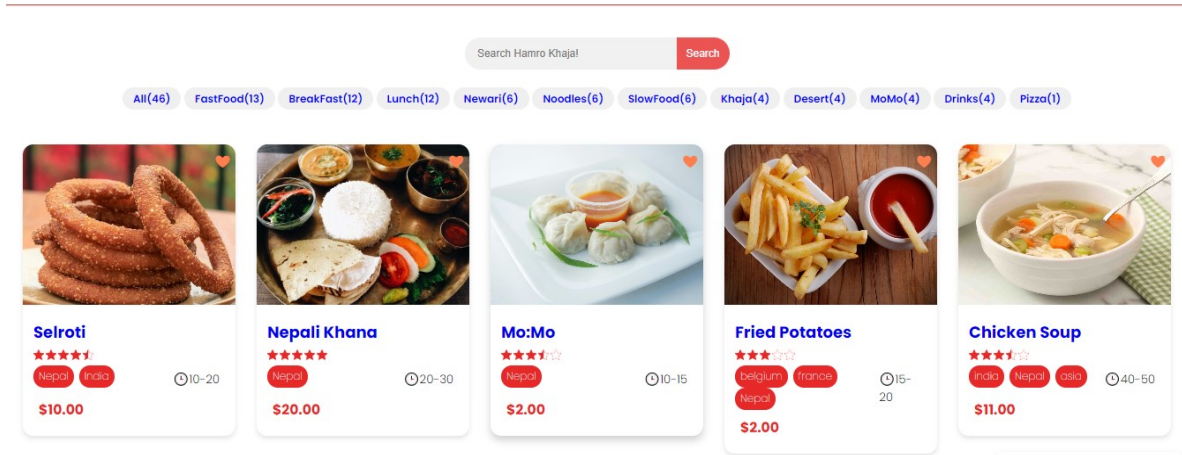
REFERENCES

- [1] Mundo, C. (n.d.). An online food ordering system requirements specification. Retrieved March 03, 2021, from
- [2] https://www.academia.edu/27192023/An_Online_Food_Ordering_System_Requirements_Specification
- [3] Regmi, D. (n.d.). Food order proposal. Retrieved March 03, 2021, from https://www.academia.edu/34223147/Food_order_proposal
- [4] Tafar, S. (2013, September 29). Customer ordering system. Retrieved March 03, 2021, from <https://www.slideshare.net/SuricyTafar/customer-ordering-system>
- [5] Literature review on food delivery services. (n.d.). Retrieved March 04, 2021, from <https://www.scribd.com/presentation/408144487/Literature-Review-on-Food-DeliveryServices>
- [6] H. Kumar, M. Jain, and M. S. Bajwa, "Online Food Delivery App 'Foodie'," *Journal of University of Shanghai for Science and Technology*, vol. 23, no. 7, pp. July 2021.
- [7] R. Ramesh and S. V. Prabhu, "An empirical study of online food delivery services from applications perspective," *Journal Title*, vol. 80, pt. 3, pp. 1751-1755, 2023.
- [8] N. W. Madinga, J. Blanckensee, L. Longhurst, and N. Bundwini, "The new normal: the adoption of food delivery apps," *European Journal of Management Studies*, Nov. 17, 2023.
- [9] L. Pokhrel and R. Shah, "Factors Affecting Behavioural Intention of Online Food Delivery Service Consumers in Kathmandu Valley," *Journal of Business and Social Sciences Research*, vol. VII, no. 2, Dec. 2022.
- [10] A. Bonfanti, C. Rossato, V. Vigolo, and A. Vargas-Sánchez, "Improving online food ordering and delivery service quality by managing customer expectations: evidence from Italy," *British Food Journal* Feb. 6, 2023.

APPENDICES

HAMRO KHAJA

Bishal bhatta Cart



Home Page

HAMRO KHAJA

Login Cart

Login

Email
Email

Password
Password

Login

New user? [Register here](#)

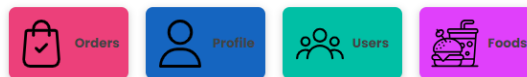
Login page

Register

Name	<input type="text"/>
Email	<input type="text"/>
Password	<input type="password"/>
Confirm Password	<input type="password"/>
Address	<input type="text"/>








[Register](#)

Sign Up page

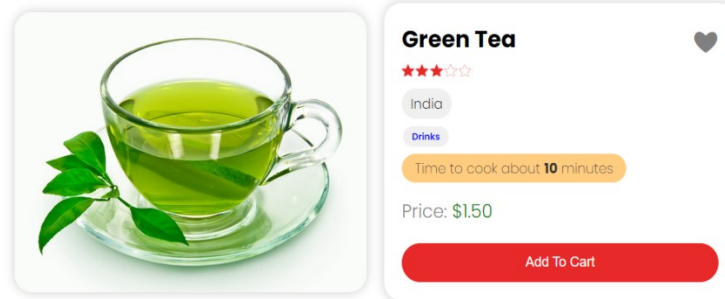


Dashboard

Manage Foods

Search Foods			Search	
Add Food +				
	Sakroti	\$10.00	Edit	Delete
	Nepali Khana	\$20.00	Edit	Delete
	MoMo	\$2.00	Edit	Delete
	Fried Potatoes	\$2.00	Edit	Delete
	Chicken Soup	\$11.00	Edit	Delete
	Vegetables Pizza	\$9.00	Edit	Delete
	Yomari	\$5.00	Edit	Delete

Food Item



Add To Cart

← → ↺ hamrokhaja.onrender.com/checkout

📍 🗺 🔍 ☆ 📄 📱 📖 📧

HAMRO KHAJA

Bishal bhatta Cart 1

Order Form


Name

Bishal bhatta

Address

charikot

Order Items:

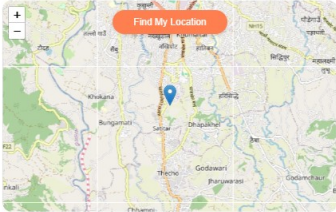
	Green Tea	\$1.50	1	\$1.50
			Total:	\$1.50

Choose Your Location

+

-

Find My Location



Go To Payment

GoTo Payment

Cart Page



Green Tea

1

\$1.50

Remove

Count: 1

Price: \$1.50

Proceed To Checkout

Proceed to checkout

Order Form

Name: Bishal bhatta

Address: charikot

Order Items:



Jalebi

\$3.00

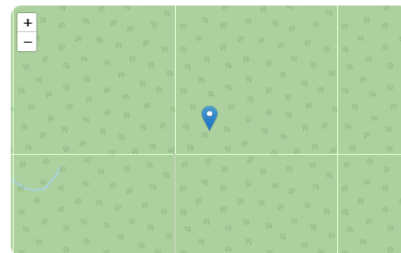
10

\$30.00

Total :

\$30.00

Your Location

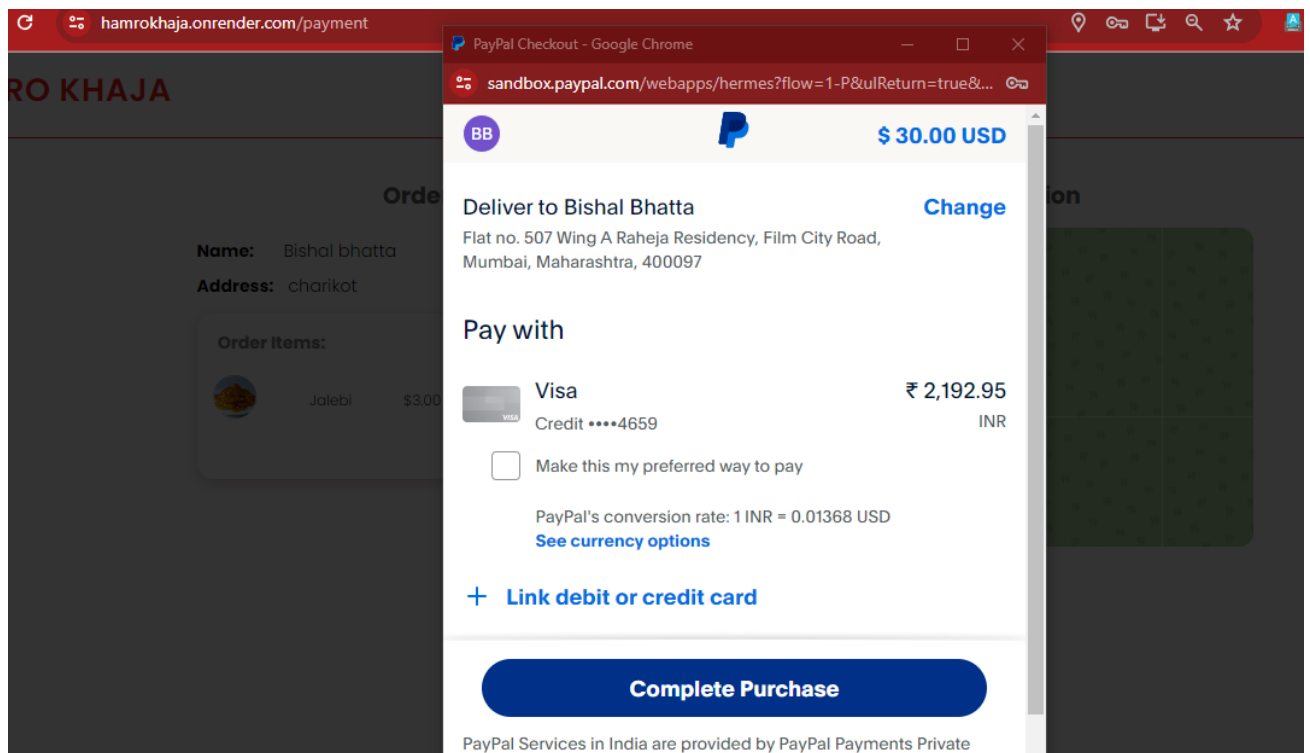


PayPal

Debit or Credit Card

Powered by PayPal

Proceed to checkout



Payment Succeed

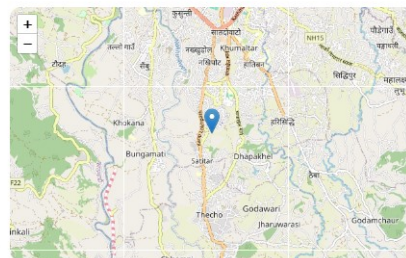
HAMRO KHAJA

Bishal bhatta Cart


Order
#66011654ded267d9ac70b63c

Date Mon, March 25, 2024 at 11:59:44 AM
Name Bishal bhatta
Address charikot
State PAYED
Payment ID 05J620203T470935E

Your Location



Order Items:

	Green Tea	\$1.50	1	\$1.50
Total :				\$1.50

Ordered Succeed

Order Payment Confirmation

Dear Biplov Subedi,

Thank you for choosing us! Your order has been successfully paid and is now being processed.

Tracking ID: 6603a403d703df547297c649

Order Date: 2024-03-27 04:43

Order Details

Item	Unit Price	Quantity	Total Price
Nepali Khana	\$20	7	\$140.00
Total:			\$140

Shipping Address: Tokha 06,Dhapasi

Search in mail

Active

Primary

Promotions 1 new
LinkedIn Job Alerts

Social 2 new
Beekalpa via Messenger, Ascol ...

<input type="checkbox"/>	☆	orders	Order 6601a36ae1b2e65d04f4d602 is being processed - Order Payment Confirmation Dear...	Mar 25
<input type="checkbox"/>	☆	orders	Order 66011690ded267d9ac70b645 is being processed - Order Payment Confirmation Dear...	Mar 25
<input type="checkbox"/>	☆	orders	Order 660041c7fbbe2638b6258058 is being processed - Order Payment Confirmation Dear...	Mar 24
<input type="checkbox"/>	☆	orders	Order 66003f213a168debac79ba5f is being processed - Order Payment Confirmation Dear ...	Mar 24
<input type="checkbox"/>	☆	orders	Order 660030937ddb71c0b4da95d1 is being processed - Order Payment Confirmation Dea...	Mar 24
<input type="checkbox"/>	☆	orders	Order 65feed51aa81e186b8c13454 is being processed - Order Payment Confirmation Dear B...	Mar 23
<input type="checkbox"/>	☆	orders	Order 65fecc729d693ef85082479d is being processed - Order Payment Confirmation Dear ...	Mar 23
<input type="checkbox"/>	☆	orders	Order 65fec6c09d693ef8508246e9 is being processed - Order Payment Confirmation Dear ...	Mar 23
<input type="checkbox"/>	☆	orders	Order 65fdc76c7c1013524910a6f3 is being processed - Order Payment Confirmation Dear B...	Mar 23
<input type="checkbox"/>	☆	orders	Order 65fdbf4d7c1013524910a6b0 is being processed - Order Payment Confirmation Dear ...	Mar 22
<input type="checkbox"/>	☆	orders	Order 65fdb0e7c1013524910a686 is being processed - Order Payment Confirmation Dear ...	Mar 22

Update Profile

Name

Bishal bhatta

Address

charikot

Update

Change Password

Current Password

Current Password

New Password

New Password

Confirm Password

Confirm Password

Change

Update Profile

Manage Users

Search Users

Search

Name	Email	Address	Admin	Actions
Madhav Dhakal	dhakal.madhav@gmail.com	Toronto On	<input checked="" type="checkbox"/>	Edit Block
Bishal bhatta	bishal@gmail.com	charikot	<input checked="" type="checkbox"/>	Edit
MadhavDhakal	satisdhakal4@gmail.com	Butwal,tilattama	<input checked="" type="checkbox"/>	Edit Block
Biplov Subedi	subedi.biplove10@gmail.com	Tokha 05,Dhapasi	<input checked="" type="checkbox"/>	Edit Block

Users