THE NADEEM’S ONLINE FOOD ORDERING SYSTEM

#### A Project Report

Submitted in partial fulfillment of the Requirements for the award of the Degree of

### BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

#### By

Nadeem Salmani

Harsh Pandey

Seat Number: TY-321

Seat Number: TY-339

**Under the esteemed guidance of**

**Prof. Mr. Mayur Tawade**

### Designation



**DEPARTMENT OF INFORMATION TECHNOLOGY**

### SHRI RAJASTHANI SEVA SANGH’S SMT.PARMESHWARIDEVI DURGADUTT TIBERWALA LIONS JUHU COLLEGE OF ARTS, COMMERCE & SCIENCE

***(Affiliated to University of Mumbai)***

**JB NAGAR, ANDHERI EAST, MUMBAI MAHARASHTRA** - **400059 2024** - **25**



**CERTIFICATE**

This is to certify that the project entitled, "**THE NADEEM’S**", is bonafied work of **Mr. NADEEM SALMANI & Mr. HARSH PANDEY** bearing Seat.No: (**TY-321**)(**TY-339**) submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

**Internal Guide Coordinator**

**External Examiner**

**Date: College Seal**

### PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

#### PNR No.: 2022016401487764 Roll No.: 321

#### PNR No.: 2022016401488156 Roll No.: 339

1. Name of the Students

**Mr. NADEEM SALMANI**

**Mr. HARSH PANDEY**

1. Title of the Project

**THE NADEEM’S ONLINE FOOD ORDERING SYSTEM**

1. Name of the Guide

**Mr. MAYUR TAWADE**

1. Teaching Experience of the Guide  **2.5 Years**
2. Is this your first submission? Yes ☐ No ☐

Signature of the Students Signature of the Guide

Date: ……………… Date: ………………

Signature of the Coordinator Date: …………………

# ABSTRACT

**THE NADEEM’S** Online Food Ordering System transforms the dining experience by allowing customers to order food online from anywhere, enhancing convenience and efficiency. Built on the MERN stack (MongoDB, Express.js, React.js, Node.js), this system ensures scalability and real-time access. Customers can easily browse Nadeem's menu, customize orders, and track delivery status, while restaurant staff can manage orders and update the menu through an intuitive dashboard.

This responsive and user-friendly web application reduces administrative workload and fosters a paperless environment. Future expansions may include features like loyalty programs, promotional offers, and integration with popular payment gateways, providing a comprehensive solution for modern dining needs.

# ACKNOWLEDEGEMENT

I would like to express my deepest gratitude to all those who have supported and guided me throughout the development of my project, “**THE NADEEM’S**.”

Firstly, I am profoundly grateful to my project advisor, **Mr. Mayur Tawade**, for their invaluable guidance, insightful feedback, and unwavering support. Their expertise and encouragement have been instrumental in the successful completion of this project.

I would also like to extend my thanks to **Mumbai University/Information Technology Department** for providing the resources and infrastructure necessary for the development of this project. The academic environment and facilities have greatly contributed to my learning experience.

Special thanks to my classmates and colleagues who have provided constructive feedback and assistance throughout the project. Their collaboration and camaraderie have been greatly appreciated.

Finally, I wish to acknowledge my family and friends for their understanding and encouragement during this period. Their patience and support have been a source of strength.

Thank you all for your contributions and support.

# DECLARATION

I hereby declare that the project entitled, “**THE NADEEM’S**” done at **SMT. PARMESHWARIDEVI DURGADUTT TIBREWALA LIONS JUHU COLLEGE**

**OF ARTS COMMERCE & SCIENCE,** has not been in any case duplicated to submit

to any other universities for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of

**BACHELOR OF SCIENCE(INFORMATION TECHNOLOGY)** to be submitted as

fifth semester project as part of our curriculum.

**Name and Signature of the Students**

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Sr. No.** | **Contents** | **Page No.** |
| **Chapter** | **1** | **INTRODUCTION** | **10 - 15** |
|  | 1.1 | Background | 10 - 11 |
|  | 1.2 | Objectives | 11 |
|  | 1.3 | Purpose, Scope, and Applicability | 12 - 13 |
|  | 1.3.1 | Purpose | 12 |
|  | 1.3.2 | Scope | 12 |
|  | 1.3.3 | Applicability | 13 |
|  | 1.4 | Achievements | 14 |
|  | 1.5 | Organisation of Report | 15 |
| **Chapter** | **2** | **SURVEY OF TECHNOLOGIES** | **16** |
| **Chapter** | **3** | **REQUIREMENTS AND ANALYSIS** | **17 - 24** |
|  | 3.1 | Problem Definition | 17 |
|  | 3.2 | Requirements Specifications | 17 - 18 |
|  | 3.2.1 | Functional Requirements | 17 |
|  | 3.2.2 | Non Functional Requirements | 18 |
|  | 3.3 | Planning and Scheduling | 18 - 19 |
|  | 3.3.1 | Development Phases | 18 |
|  | 3.3.2 | Milestones | 19 |
|  | 3.4 | Software and Hardware Requirements | 19 - 20 |
|  | 3.4.1 | Software Requirements | 19 |
|  | 3.4.2 | Hardware Requirements | 20 |
|  | 3.5 | Preliminary Product Description | 20 - 21 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | 3.5.1 | Objectives of the System | 20 |
|  | 3.5.2 | Functions of the System | 21 |
|  | 3.5.3 | System Operation | 21 |
|  | 3.6 | Conceptual Models | 22 - 24 |
|  | 3.6.1 | System Architecture | 22 |
|  | 3.6.2 | User Interaction Flow | 23 |
|  | 3.6.3 | Data Flow | 24 |
| **Chapter** | **4** | **SYSTEM DESIGN** | **25 - 36** |
|  | 4.1 | Basic Modules | 25 - 27 |
|  | 4.2 | Data Design | 28- 31 |
|  | 4.2.1 | Schema Design | 28-29 |
|  | 4.2.2 | Data Integrity and Constraints | 30 - 31 |
|  | 4.3 | Procedural Design | 32 - 33 |
|  | 4.3.1 | Logic Diagrams | 32 |
|  | 4.3.2 | Data Structure | 33 |
|  | 4.3.3 | Algorithms Design | 33 |
|  | 4.4 | User Interface Design | 34 |
|  | 4.5 | Security Issues | 35 |
|  | 4.6 | Test Cases Design | 36 |

**TABLE OF FIGURE**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Sr. No.** | **Contents** | **Page No.** |
| **Chapter** | **3** | **REQUIREMENTS AND ANALYSIS** | **17 - 24** |
|  |  | Fig-3.1 System Architecture | 22 |
|  |  | Fig-3.2 User Interaction Flow | 23 |
|  |  | Fig-3.3 Data Flow Diagram | 24 |
| **Chapter** | **4** | **SYSTEM DESIGN** | **25 - 36** |
|  |  | Fig-4.1.1 User Module | 25 |
|  |  | Fig-4.1.2 Admin Module | 25 |
|  |  | Fig-4.1.3 Gantt Chart | 27 |
|  |  | Fig-4.2.1 Schema Design(ERD) | 28 |
|  |  | Fig-ER Model(ERM) | 29 |
|  |  | Fig-4.2.2Data Integrity & Constraints(Table) | 31 |
|  |  | Fig-4.3.1 Logic Diagram | 32 |
|  |  | Fig-4.3.2 Class Diagram | 33 |
|  |  | Fig-4.4 UI Design | 34 |
|  |  | Fig-4.5 Use Case Diagram | 35 |
|  |  | Fig-4.6 Test Case Diagram | 36 |

**CHAPTER - 1** **INTRODUCTION**

In today’s digital age, the way restaurants manage food orders has undergone a significant transformation. Traditional methods, which often rely on manual processes, are increasingly being replaced by modern, automated systems that enhance efficiency and improve customer satisfaction.

The **Nadeem's Online Food Ordering System** is a web-based application designed to streamline restaurant operations and provide an exceptional dining experience. This system facilitates the management of menu offerings, customer orders, and delivery logistics, allowing for seamless interaction between customers and staff. By leveraging the MERN stack (MongoDB, Express.js, React.js, Node.js), the platform ensures real-time access and scalability, enabling users to browse the menu, customize orders, and track deliveries from anywhere. With these advancements, Nadeem's aims to modernize the food ordering process, making it easier and more efficient for both customers and restaurant personnel.

## Background

Restaurants have long been central to community dining and social interaction, but traditional food ordering processes have often relied on manual methods, leading to inefficiencies and errors. Typically, these processes involve handwritten orders, physical menus, and manual tracking of customer preferences and delivery logistics. As restaurants grow in size and customer volume, these manual systems can become cumbersome, resulting in delays, order inaccuracies, and an overall less satisfying customer experience.

In an era where digitization is reshaping various industries, the restaurant sector is also evolving. There is an increasing demand for automated systems that streamline food ordering and enhance the overall dining experience. Web-based platforms allow customers to explore menus, place orders, and track their deliveries from the comfort of their homes. This digital transformation not only improves accessibility but also significantly reduces human error and enhances operational efficiency.

The **Nadeem's Online Food Ordering System**, addresses these challenges by providing an efficient, scalable, and user-friendly solution for restaurants. This system enables restaurant owners to manage their menus, track orders, and oversee delivery processes with ease, all while reducing reliance on paper records.

### Advantages of website over Traditional Systems:

* **Increased Convenience: Customers can place orders anytime and from anywhere, making dining more accessible and flexible.**
* **Real-Time Order Tracking: Users can track their orders in real time, providing transparency and reducing uncertainty about delivery times.**
* **Reduced Human Error: Automated order processing minimizes mistakes, ensuring customers receive exactly what they ordered and enhancing overall satisfaction.**

These advantages collectively enhance the overall dining experience, driving customer satisfaction and operational efficiency for **Nadeem's**.

## Objectives

This project aims to solve the inefficiencies of traditional restaurant management by offering a user-friendly online platform. By automating core processes, the system enhances convenience and operational efficiency for both customers and staff.

1. **Automating Food Ordering and Delivery Processes**: The system streamlines key operations, including menu browsing, food ordering, and delivery tracking. Previously, these tasks required manual effort, which could lead to errors or delays. With this automated system, users can place orders and track deliveries in real-time, improving both accuracy and speed.
2. **Real-time Menu Access and Updates**: Customers can browse the restaurant’s menu online, with real-time updates reflecting any changes in availability. Additionally, restaurant staff can manage the menu from the backend, adjusting prices, adding new items, or removing unavailable dishes instantly. This ensures that customers always have access to the latest offerings without needing to visit or call the restaurant.
3. **Online Payment and Instant Order Confirmation**: The system supports secure, convenient online payments and offers multiple payment options. Once an order is paid for, customers receive instant confirmation along with order updates, creating a seamless transaction experience.
4. **Order Delivery Tracking**: Customers can follow their orders in real-time, from preparation to delivery. This transparency reduces customer wait-time anxiety, as they know exactly when their order will arrive.
5. **Backend Access for Staff to Manage Operations**: The system includes a backend interface for staff to efficiently manage restaurant operations. Staff can update the menu, change prices, list new items, and modify order statuses as needed. This functionality enables real-time updates for customers and ensures the restaurant maintains full control over its offerings.
6. **Reducing Manual Paperwork and Improving Efficiency**: By digitizing processes like order management, payments, and inventory updates, the system minimizes manual record-keeping and paperwork. This significantly improves operational efficiency and allows for more accurate data tracking and reporting.

## Purpose, Scope, and Applicability

### Purpose

The primary purpose of this project is to design and implement a web-based Online Food Ordering System that simplifies the process of ordering food, reduces operational inefficiencies, and enhances the overall customer experience. This system seeks to ease the burden on restaurant staff by automating tedious tasks such as menu updates, order processing, and delivery tracking. At the same time, customers benefit from real-time access to the menu, secure online payments, and the ability to track their orders from preparation to delivery.

The system also aims to contribute to operational sustainability by digitizing processes, such as order management and payment records, which reduces paper usage and leads to a more environmentally friendly operation.

### Scope

The Online Food Ordering System, while primarily focused on basic ordering and delivery functions, has the potential for future expansion into more advanced features. In the initial implementation, the system will focus on:

* **User Registration**: New users can sign up, create a profile, and manage personal information, including their order history and payment preferences. This provides a personalized and streamlined user experience.
* **Menu Management**: Restaurant staff will have full control over the menu, allowing them to add, update, or remove items. Price adjustments and availability updates can be managed easily through the backend.
* **Menu Browsing**: Customers will be able to browse the restaurant’s menu in real-time. They can view available dishes, including details on pricing and customization options. The search feature will allow users to quickly find items by category or name.
* **Order Placement and Payment**: Customers can efficiently place orders and make payments online through secure methods. Real-time confirmation of order placement and payment ensures smooth transactions, reducing manual intervention by staff.
* **Order Tracking**: The system will provide real-time tracking of orders, from preparation to delivery. This ensures transparency and enhances the customer experience by providing accurate delivery time estimates.

Future extensions of the system could include features such as promotions, loyalty programs, and third-party delivery service integration. This flexibility ensures that the system can grow with changing user needs and business demands.

### Applicability

The Online Food Ordering System has broad applicability across various sectors of the food industry. It can be used by:

* **Restaurants**: Full-service and delivery-focused restaurants can leverage the system to automate online orders and improve customer service.
* **Cafes and Quick-Service Restaurants**: Fast-food establishments and cafes can benefit from quick order processing and delivery management through this system, reducing wait times and increasing efficiency.
* **Catering Services**: Catering companies can use the system to streamline bulk order management and payment processing for clients.

This system can also be adapted for use in specialized food services, food trucks, and other food-related businesses that require efficient online ordering and delivery management.

## Achievements

Upon completing the development of the Online Food Ordering System for THE NADEEM'S, the following key achievements were gained:

* **Full-Stack Development Proficiency**: Developing the system honed my skills in full-stack web development, particularly using the MERN stack (MongoDB, Express.js, React.js, and Node.js). I gained experience in building, integrating, and deploying a complete web application, ensuring that both front and back ends of THE NADEEM'S system worked seamlessly to deliver a smooth user experience.
* **Mastery of Database Management**: By using MongoDB, I learned how to efficiently manage a NoSQL database, structuring collections to manage real-time updates for the restaurant’s menu, customer orders, and transaction data. I worked on performing CRUD (Create, Read, Update, Delete) operations at scale, ensuring data accuracy and reliability for both the customer interface and staff.
* **Enhanced Understanding of API Development**: With Node.js and Express.js, I gained hands-on experience in developing secure, high-performance RESTful APIs. These APIs handled tasks such as order placement, payment integration, and delivery tracking, providing efficient communication between the frontend and backend systems, while ensuring optimal response times for THE NADEEM'S operations.
* **Front-End Expertise**: Creating a dynamic and responsive user interface with React.js refined my front-end development skills. I focused on building reusable components and ensuring state management was optimized for real-time updates. The design also focused on enhancing user engagement, making it easy for customers to browse menus, customize orders, and track delivery statuses directly from their mobile or desktop devices.
* **Efficient Project Structuring and Deployment**: I gained experience in organizing and structuring a complex project for scalability and maintainability. This involved integrating frontend and backend services, deploying the system to a cloud environment, and optimizing it for performance. The deployment process also involved managing environment variables for secure data handling and smooth production operations.
* **Problem-Solving and Debugging Skills**: Throughout development, I faced challenges such as ensuring secure user authentication, optimizing database queries, and implementing real-time order tracking. These tasks helped me sharpen my problem-solving and debugging skills, allowing me to troubleshoot issues in both the frontend and backend effectively and efficiently.
* **Practical Application of Security Best Practices**: Implementing security measures such as JWT-based user authentication, secure payment processing, and data validation was crucial for maintaining the integrity of customer data. These best practices ensured that THE NADEEM'S platform operated in compliance with industry standards for secure web applications, protecting against unauthorized access and data breaches.

## Organization of Report

This report is structured into six chapters, each focusing on a different aspect of the project:

* **Chapter 1: Introduction** - Provides the background, objectives, and scope of the project.
* **Chapter 2: Review of Technologies** - Discusses the technologies used in the development of the project, including MongoDB, Express.js, React.js, Node.js, and other relevant tools.
* **Chapter 3: Project Requirements and Analysis** - Outlines the system requirements and provides an in-depth analysis of the project scope, including functional and non-functional requirements.
* **Chapter 4: System Design** - Covers the system’s design, including the database schema, user interface designs, and overall architecture.
* **Chapter 5: Implementation and Testing** - Presents the implementation process, coding details, and testing methodologies employed to ensure the system functions as intended.
* **Chapter 6: Conclusion and Future Enhancements** - Concludes the report by summarizing the outcomes and discussing potential future improvements.

# CHAPTER - 2 SURVEY OF TECHNOLOGIES

The development of the Online Food Ordering System for *THE NADEEM'S* required careful selection of technologies to ensure scalability, efficiency, and a smooth user experience. Each technology was chosen based on its ability to meet the unique requirements of the project. The following is a detailed survey of the technologies explored and chosen for developing the platform:

1. **React.js: Dynamic and Responsive User Interface**  
   React.js, a powerful JavaScript library for building user interfaces, was selected as the primary front-end technology. React allows the creation of dynamic, interactive, and responsive user interfaces, enabling customers to seamlessly browse the menu, customize orders, and track their deliveries in real-time.
2. **Node.js and Express.js: Back-End Development and RESTful API Creation**  
   Node.js, a JavaScript runtime built on Chrome’s V8 engine, was chosen as the foundation for back-end development. Express.js, a minimal and flexible Node.js framework, was used to efficiently build the server-side logic and create RESTful APIs. These APIs handle crucial functions such as order placement, payment processing, and order tracking, ensuring reliable communication between the client and server.
3. **MongoDB: Scalable and Flexible NoSQL Database**  
   MongoDB, a NoSQL database, was selected for its scalability and flexibility. By storing data in a JSON-like format, MongoDB allows for the efficient handling of customer information, menu details, orders, and delivery statuses. This flexibility enables the system to adapt as new features are introduced or customer requirements evolve.
4. **JSON Web Tokens (JWT): Secure User Authentication and Session Management**  
   For secure user authentication and session management, JSON Web Tokens (JWT) were chosen. JWT allows for secure transmission of information between the frontend and backend, ensuring that only authorized users can access sensitive features like order history, payment details, and account information.
5. **HTML and CSS: Modern User Interface Styling**  
   The frontend of the website was styled using HTML and CSS. These technologies provide the structure and design for the user interface, ensuring that the website is visually appealing and responsive across different devices. By avoiding frameworks like Tailwind, we focused on custom styling to create a unique and personalized experience for *THE NADEEM'S* customers.

# CHAPTER - 3 REQUIREMENTS AND ANALYSIS

## Problem Definition

In a traditional restaurant ordering system, operations such as managing menu items, processing orders, handling customer data, and tracking deliveries are often done manually or through outdated point-of-sale systems. These methods introduce inefficiencies, errors, and challenges, especially when managing high order volumes and providing a smooth user experience. The primary problems to address are:

* **Inefficiency**: Manual processes for taking orders and managing deliveries are time-consuming, prone to human error, and can result in delays, particularly during peak hours.
* **Limited Access**: Traditional systems offer little to no online access, making it difficult for customers to browse menus, place orders, or track delivery statuses remotely. This limits user engagement and convenience.
* **Data Management**: Managing customer data, order histories, and inventory manually increases the risk of inconsistencies, data loss, and difficulty in analyzing business performance or generating insightful reports.

The online food ordering system for *THE NADEEM'S* aims to solve these problems by providing a digital platform that automates restaurant operations, enhances user accessibility, and improves data management, ultimately offering a seamless and efficient customer experience.

## Requirements Specification

### Functional Requirements

Functional requirements specify the features and functionalities *THE NADEEM'S* online food ordering system must provide to meet user and restaurant needs:

* **User Registration and Authentication**: The system must allow users to register, log in, and securely manage their accounts, including viewing past orders and tracking current deliveries.
* **Menu Management**: Restaurant staff must be able to add, update, and remove menu items, as well as change pricing and item availability in real-time through the backend interface.
* **Menu Browsing**: Users should be able to browse and search the online menu, filtering items by category, availability, or price.
* **Order Placement and Payment**: The system must facilitate order placement, allowing users to select items, customize orders, and securely complete payments through various payment methods.
* **Order Tracking and Delivery Updates**: Users should be able to track the status of their orders in real time, receiving updates when the order is confirmed, prepared, and out for delivery.

### Non-Functional Requirements

Non-functional requirements define the system’s quality attributes for THE NADEEM'S online food ordering platform:

* **Performance**: The system should deliver fast response times, allowing users to browse the menu, place orders, and track deliveries without delay, even during peak hours with high user traffic.
* **Security**: Sensitive user data, including personal information and payment details, must be protected against unauthorized access and breaches through secure authentication methods and data encryption.
* **Usability**: The user interface should be intuitive and easy to navigate, ensuring a smooth experience for customers of all technical backgrounds, from first-time visitors to regular users.
* **Reliability**: The platform should be robust and highly available, with minimal downtime to ensure that users can consistently place orders and track their deliveries without disruptions.

## Planning and Scheduling

Planning and scheduling involve outlining the development phases, milestones, and timeline for the project:

### Development Phases

* + - * **Phase 1:** Requirements Gathering and Analysis
        + **Activities:** Define problem, gather requirements, and document specifications, Cost estimation.
      * **Phase 2:** System Design
        + **Activities:** Design system architecture, database schema, and user interface.
      * **Phase 3:** Implementation
        + **Activities:** Develop front-end and back-end components, integrate systems.
      * **Phase 4:** Testing
        + **Activities:** Conduct unit testing, integration testing, and user acceptance testing.
      * **Phase 5:** Deployment and Training
        + **Activities:** Deploy the system and documentation.
      * **Phase 6:** Maintenance and Support
        + **Activities:** Provide support, update the system, and handle user feedback.

### Milestones

* + - * Completion of Requirements Specification
      * Approval of System Design
      * Completion of Core Development
      * Successful Testing and Bug Fixing
      * System Deployment

## Software and Hardware Requirements

### Software Requirements

#### Server-Side

* + - * Operating System: Windows Server
      * Node.js: LTS version
      * Express.js: For RESTful API development
      * MongoDB: NoSQL database management

#### Client-Side

* + - * React.js: For building the user interface
      * CSS: For styling

#### Development Tools

* + - * Code Editor/IDE: Visual Studio Code
      * Version Control: Git and GitHub

### Hardware Requirements

#### Server

* + - * **Processor:** Multi-core CPU
      * **Memory:** 8 GB RAM
      * **Storage:** 512 GB SSD
      * **Network:** High-speed internet connection

#### Client Devices

* + - * **Processor:** Modern multi-core CPU
      * **Memory:** 4 GB RAM
      * **Browser:** Latest versions of Chrome, Firefox, Edge, or Safari

#### Core Features:

* + - * **User Management:** Registration, login, and profile management
      * **Filter:** Search, filter and browse menu
      * **Admin:** Managing user data, Managing orders, New listing,Delivery update

## Preliminary Product Description

* *THE NADEEM'S* online food ordering system is designed to modernize the restaurant’s operations, ensuring efficiency, accuracy, and enhanced user convenience. The primary goal of the system is to automate and streamline core restaurant functions, including managing customer data, updating and organizing the menu, processing online orders and payments, and allowing customers to browse the menu, place orders, and track deliveries online in real-time.

### Objectives of the System:

 **Automate Restaurant Operations**: Minimize manual errors and streamline efficiency by digitizing key tasks like order processing, menu management, and delivery tracking.

 **Enhance User Experience**: Provide a user-friendly interface for customers to easily browse the menu, customize meals, and track orders in real time.

 **Improve Accessibility**: Enable online access to the menu, allowing customers to place orders, make secure payments, and monitor delivery statuses from any device.

 **Efficient Data Management**: Utilize a centralized database to manage customer information, order histories, and menu details, ensuring data integrity and scalability.

 **Real-Time Updates**: Keep customers informed with real-time updates on order status and menu changes, providing current information to enhance satisfaction.

### Functions of the System:

 **User Management:** The system allows for user registration and login with secure authentication. Users can manage their profiles, view their order history, and track the status of current orders.

 **Menu Browsing:** The website provides a comprehensive menu that users can browse, search, and filter. It includes detailed information about the dishes, such as name, description, price, and availability.

 **Food Ordering and Delivery Tracking:** Users can order available dishes from the menu and track their order status in real-time. The system also manages delivery updates to ensure users are informed about their order progress.

 **Payment Integration:** The system allows users to make secure online payments for their orders. Multiple payment methods can be integrated, including credit/debit cards and digital wallets.

 **Administrative Features:** Restaurant staff can manage the backend, including adding new menu items, updating dish details, changing prices, and removing dishes. The system also provides real-time order and delivery status management.

### System Operation:

THE NADEEM'S online food ordering system operates as a web-based platform, accessible via any browser on devices with an internet connection. It utilizes a React.js front-end for a dynamic and responsive user interface, while the back-end is powered by Node.js and Express.js to handle server-side operations and API requests. The system manages all data through a MongoDB database, enabling flexible and scalable data management.

By implementing these functionalities, THE NADEEM'S food ordering system addresses the inefficiencies of traditional ordering methods and provides a user-friendly, robust solution for modern dining needs.

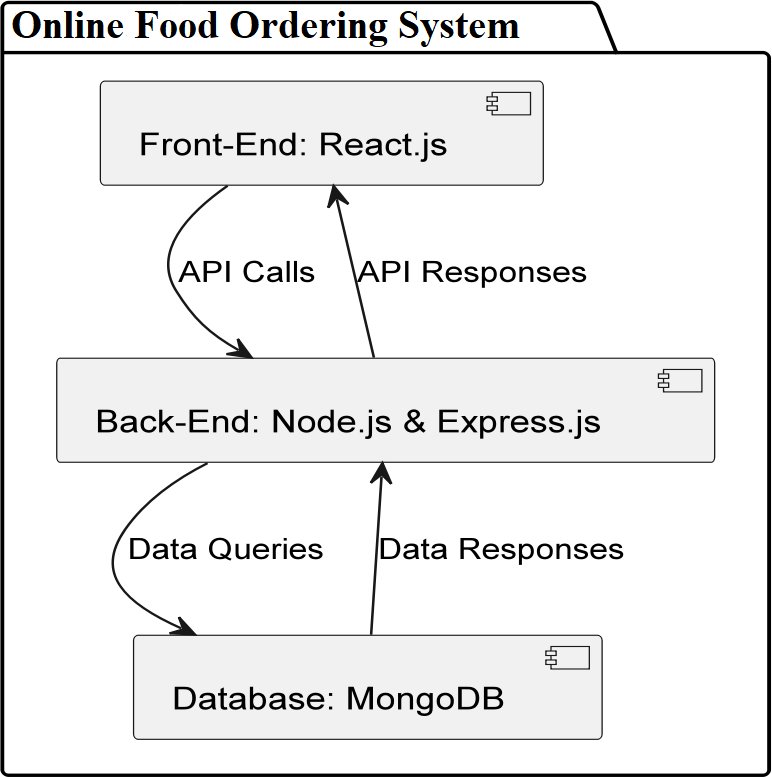
## Conceptual Model

The conceptual model illustrates the high-level design and interaction between different components of the website:

### System Architecture

* + - * **Front-End:** Built with React.js, providing a dynamic and responsive user interface.
      * **Back-End:** Developed with Node.js and Express.js, handling server-side logic and API endpoints.
      * **Database:** MongoDB for flexible and scalable data storage.

Fig-3.1 System Architecture



### User Interaction Flow

* **User Registration/Login**: Customers interact with the registration or login page to create or access their accounts. The system communicates with the back-end for user authentication to ensure secure access.
* **Menu Browsing**: Customers browse through the restaurant’s menu, which is dynamically fetched from the back-end. They can search for specific dishes, filter by categories (e.g., appetizers, main courses, desserts), and view details such as ingredients, prices, and availability.
* **Order Placement**: Customers add items to their cart and proceed to place an order. The system communicates with the back-end to validate the order, update stock levels for menu items, and calculate the total amount for payment.
* **Payment**: The system directs customers to a secure payment page where they can complete the transaction using various payment options. Once the payment is processed, the order is confirmed, and a receipt is sent to the customer.
* **Order Management**: Restaurant staff use the admin panel to update the menu, change prices, add or remove items, and manage orders. The system updates the order status (e.g., preparing, out for delivery) in real time.
* **Delivery Status**: Once the order is confirmed, customers can track the status of their delivery through the system. Updates are provided in real-time, from order preparation to dispatch and delivery.
* **Order History and Reordering**: Customers can view their past orders and choose to reorder items. The system retrieves the history from the database and facilitates quick reordering of favourite items.

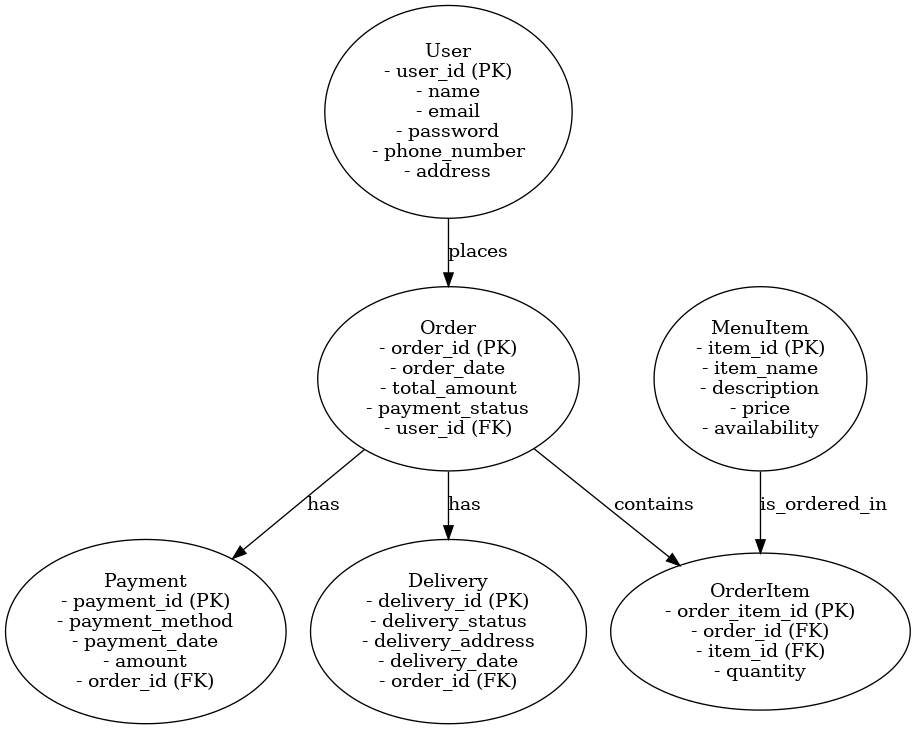


Fig-3.2 User Interaction Flow

### Data Flow

* + - * **Client Requests:** User actions on the front-end generate requests sent to the server via API endpoints.
      * **Server Processing:** The server processes requests, interacts with the database, and returns responses to the client.
      * **Database Operations:** Data is stored, retrieved, and updated in MongoDB as required by the system.

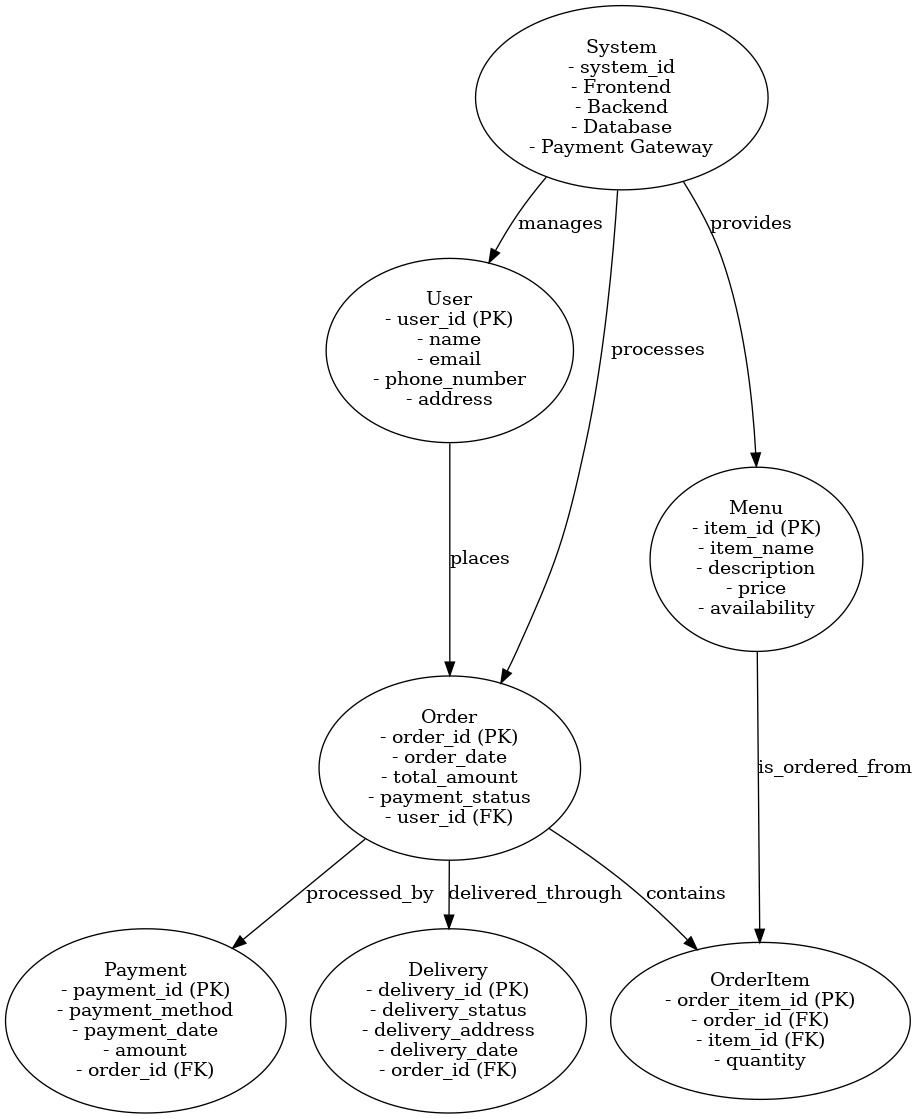


Fig-3.3 Data Flow Diagram

**4.1Basic Modules**

# CHAPTER - 4 SYSTEM DESIGN

The system is divided into the following basic modules:

1. **User Module:** Allows users to register, login, browsing food items, order food and track delivery and manage their profile.

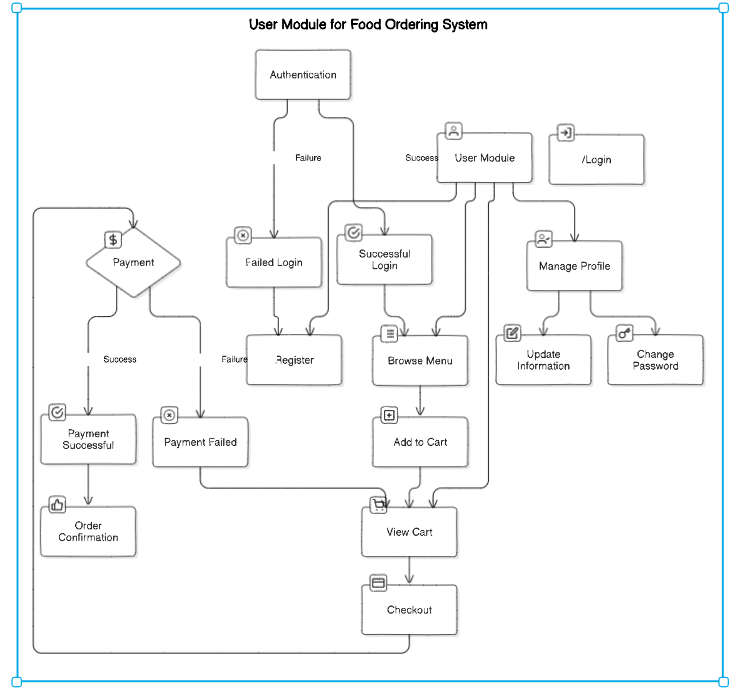


Fig-4.1.1 User Module

1. **Admin Module:** Provides the admin with the ability to manage users data, menu and handle other administrative tasks such as managing food items and viewing user orders updating delivery status.

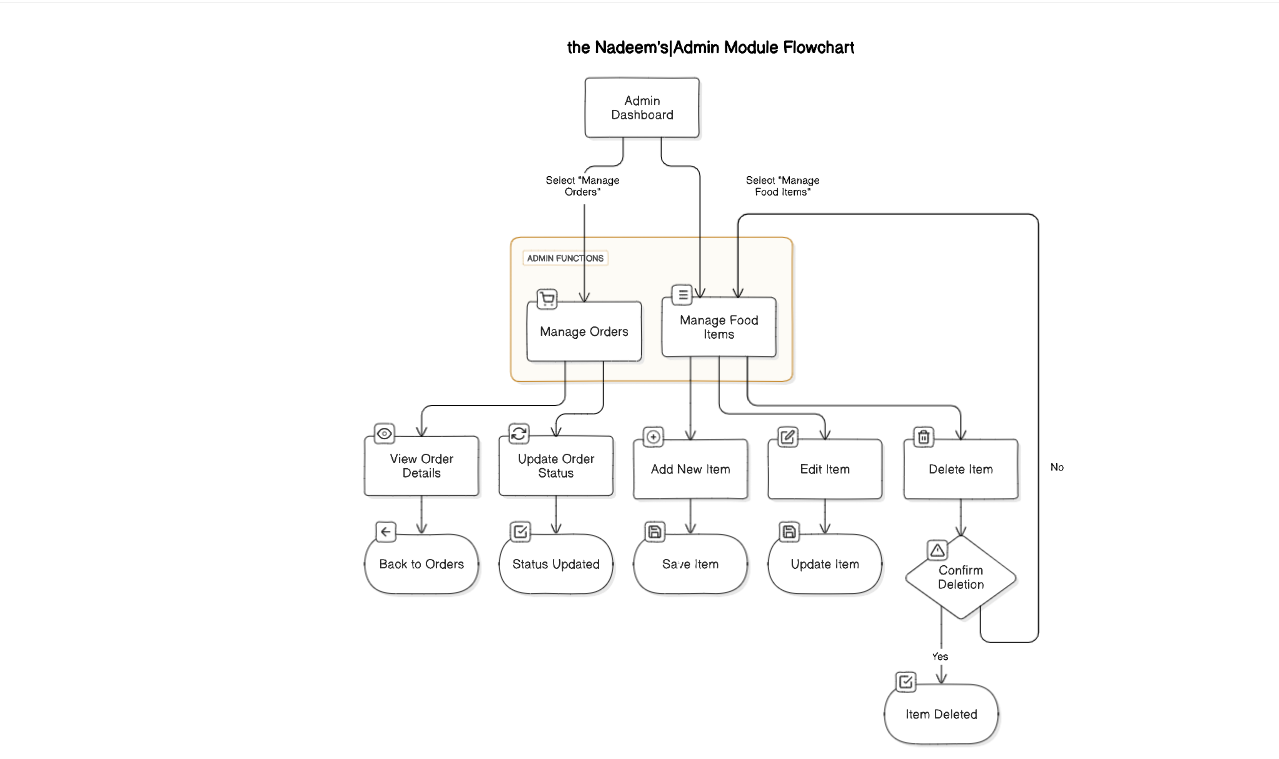
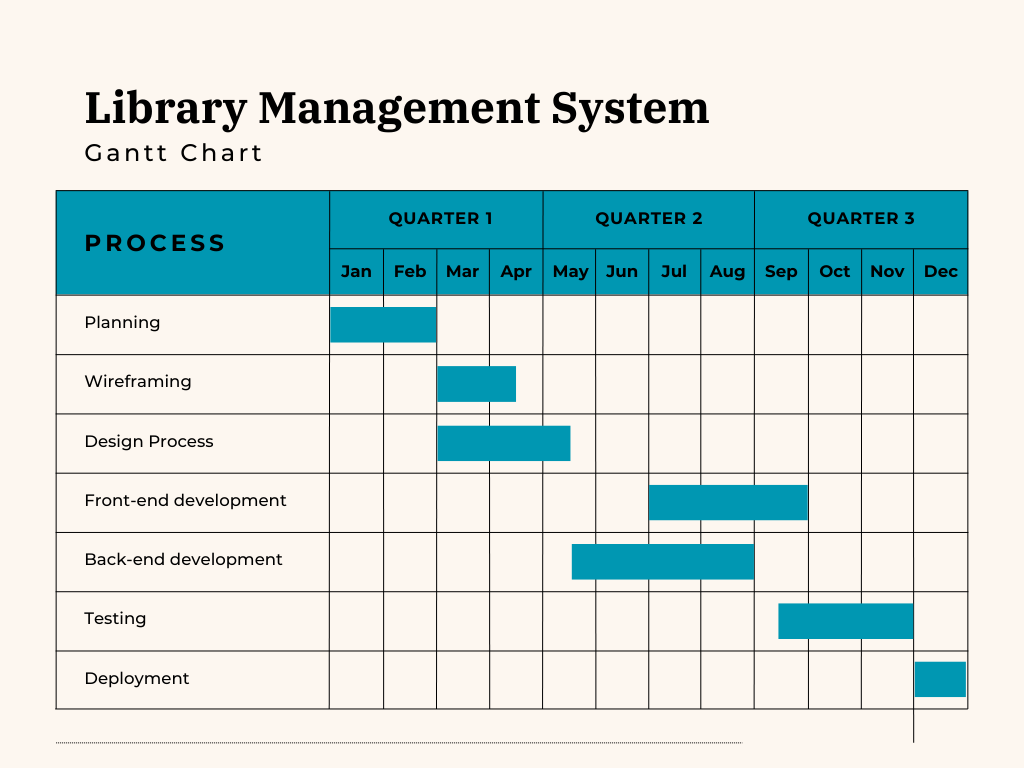


Fig-4.1.2 Admin Module

* + **Project Timeline:** The following Gantt chart shows the development timeline for each of the basic modules mentioned above, including their design, implementation, and testing phases.



**THE NADEEM’S FOOD ORDERING SYSTEM**

Fig-4.1.3 Gantt Chart Design

## Data Design

### 4.2.1 Schema Design

### The database design for THE NADEEM'S online food ordering system includes several entities, such as Users, Orders, Food Items, Payments, and Deliveries. The relationships between these entities are shown in the ER diagram. Each user can place multiple orders, each order contains multiple food items, and payments and deliveries are associated with each order.

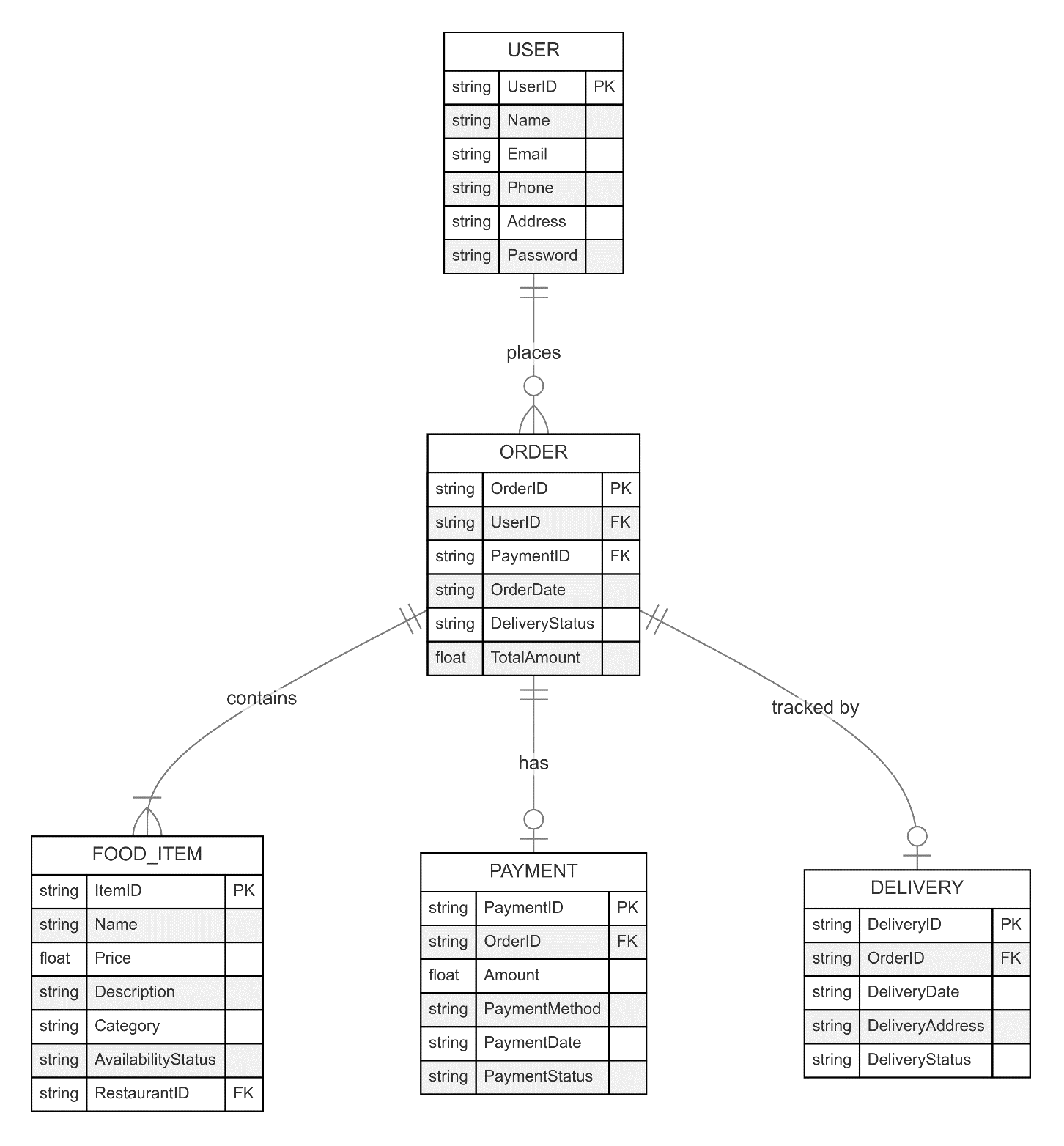


Fig-4.2.1 Schema Design(ERD)

### ER Model

The ER Model outlines the entities, attributes, and relationships within the

THE NADEEM’S Online Food Ordering:

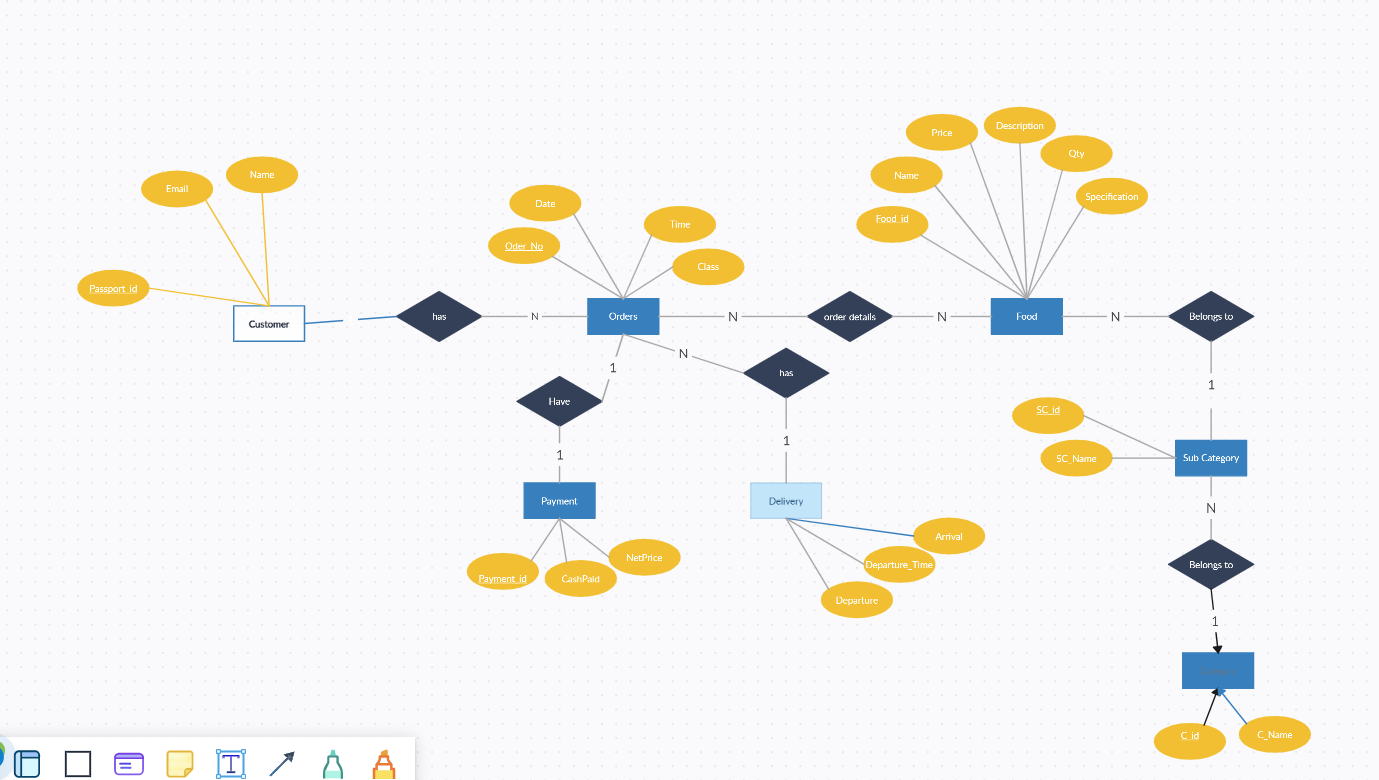


Fig-4.2.1.2 ER Model

### Data Integrity and Constraints

To ensure data integrity within **THE NADEEM'S** online food ordering system, several constraints are implemented across different tables. The following table outlines the entities, their attributes, and the constraints applied to maintain data integrity:

Fig-4.2.2 Data Integrity and Constraints

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity** | **Attribute** | **Constraint** | **Description** |
| **User** | UserID | **Primary Key, Not Null, Unique** | Ensures each user has a unique identifier. |
|  | Email | **Not Null, Unique** | Guarantees no duplicate email addresses for users. |
|  | Phone | **Not Null, Unique** | Ensures a unique phone number is associated with each user. |
|  | Password | **Not Null** | Enforces a password for user account security. |
|  | Address | **Not Null** | Requires an address for delivery purposes. |
| **Order** | OrderID | **Primary Key, Not Null, Unique** | Ensures each order is uniquely identified. |
|  | UserID | **Foreign Key (references User.UserID), Not Null** | Links each order to a registered user. |
|  | TotalAmount | **Not Null, Check (TotalAmount > 0)** | Ensures the order total is always a positive value. |
|  | OrderDate | **Not Null, Default (current date)** | Automatically records the date when an order is placed. |
|  | DeliveryStatus | **Not Null, Default ('Pending')** | Default status for orders awaiting delivery. |
| **Food Item** | ItemID | **Primary Key, Not Null, Unique** | Each food item has a unique identifier. |
|  | Name | **Not Null** | Requires a name for each food item. |
|  | Price | **Not Null, Check (Price > 0)** | Ensures a positive price for each food item. |
|  | AvailabilityStatus | **Not Null, Default ('Available')** | Default status indicates that food items are available. |
| **Payment** | PaymentID | **Primary Key, Not Null, Unique** | Ensures each payment transaction is uniquely identified. |
|  | OrderID | **Foreign Key (references Order.OrderID), Not Null** | Links each payment to a specific order. |
|  | Amount | **Not Null, Check (Amount = Order.TotalAmount)** | Ensures payment matches the total order amount. |
|  | PaymentDate | **Not Null, Default (current date)** | Automatically records the date of payment. |
|  | PaymentMethod | **Not Null** | Requires the method of payment (e.g., card, cash). |
| **Delivery** | DeliveryID | **Primary Key, Not Null, Unique** | Each delivery instance has a unique identifier. |
|  | OrderID | **Foreign Key (references Order.OrderID), Not Null** | Links the delivery to a specific order. |
|  | DeliveryDate | **Not Null, Default (current date)** | Automatically records the date of delivery. |
|  | DeliveryStatus | **Not Null, Default ('In Transit')** | Tracks the delivery status, e.g., 'In Transit' or 'Delivered'. |
|  | DeliveryAddress | **Not Null** | Requires a valid delivery address for each order. |

## Procedural Design

### Logic Diagrams

Logic diagrams include flowcharts and activity diagrams to represent the process flow of key functions such as user login, menu management, and order processing.  
For example, the flow of placing an order will involve selecting items from the menu, confirming availability, processing the payment, and updating the order status for delivery.

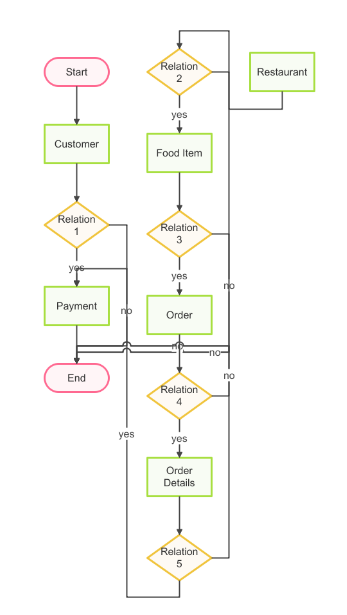
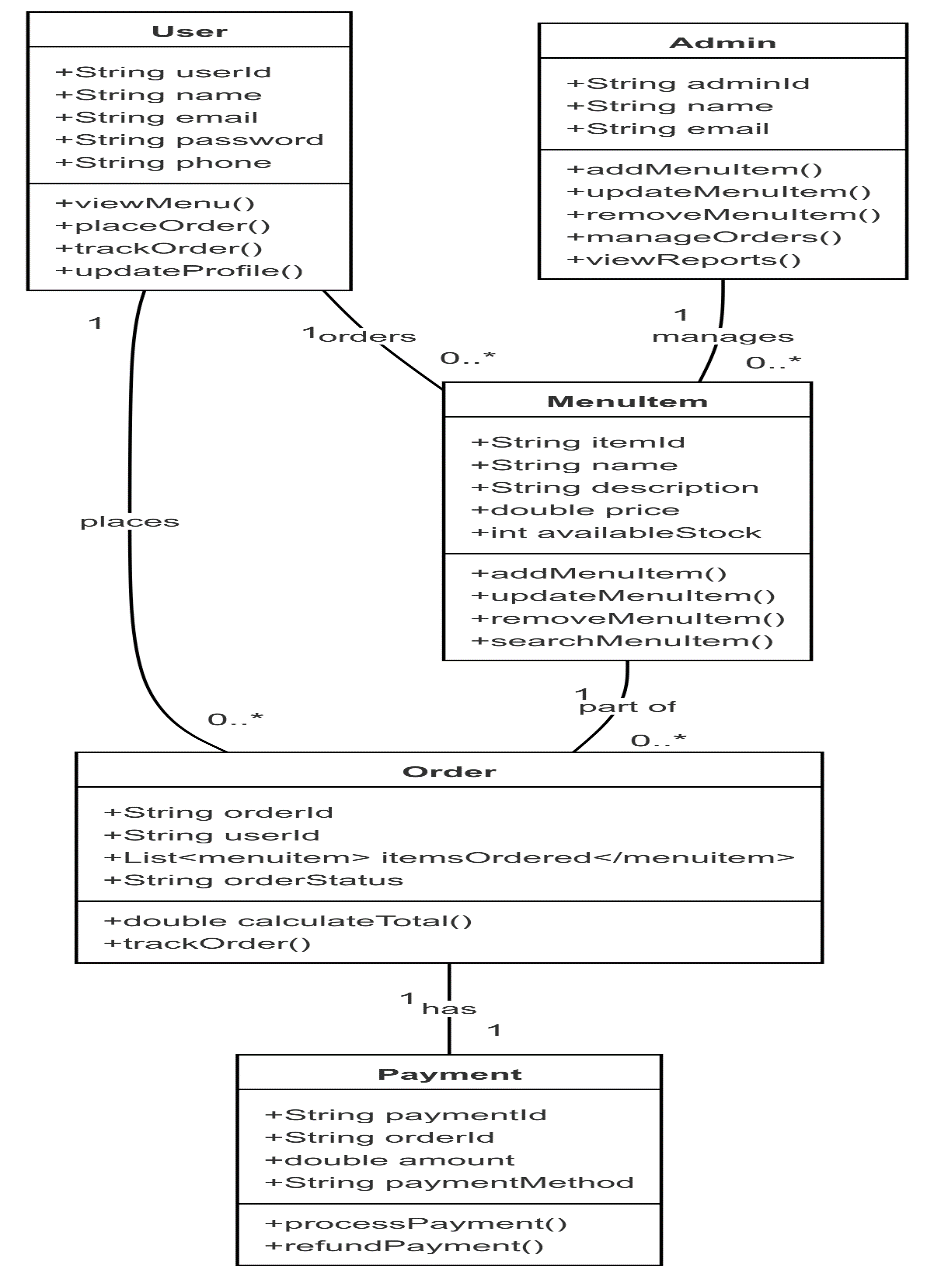


Fig-4.3.1 Logic Diagram

### Data Structures

The data structures used in the system are optimized for managing a large collection of menu items, users, and order records. Hash maps and arrays are utilized for efficient storage and retrieval of data, ensuring that search and filtering operations on the menu can be performed quickly.



### Algorithms Design

Fig-4.3.2 Class Diagram

The algorithms implemented in the system handle operations such as searching for menu items, managing food orders, and processing payments. The menu search algorithm is optimized to filter items based on name, category, or price. The payment processing algorithm calculates the total cost based on selected items, applicable taxes, and discounts. Additionally, the order tracking algorithm updates users with real-time delivery status.

## User Interface Design

The user interface is designed to be intuitive and user-friendly. Wireframes are created to demonstrate the layout of the login page, menu browsing, order placement, and admin management dashboard. Responsive design principles are applied to ensure compatibility across different devices and screen sizes, providing a seamless experience for users whether on desktop or mobile.

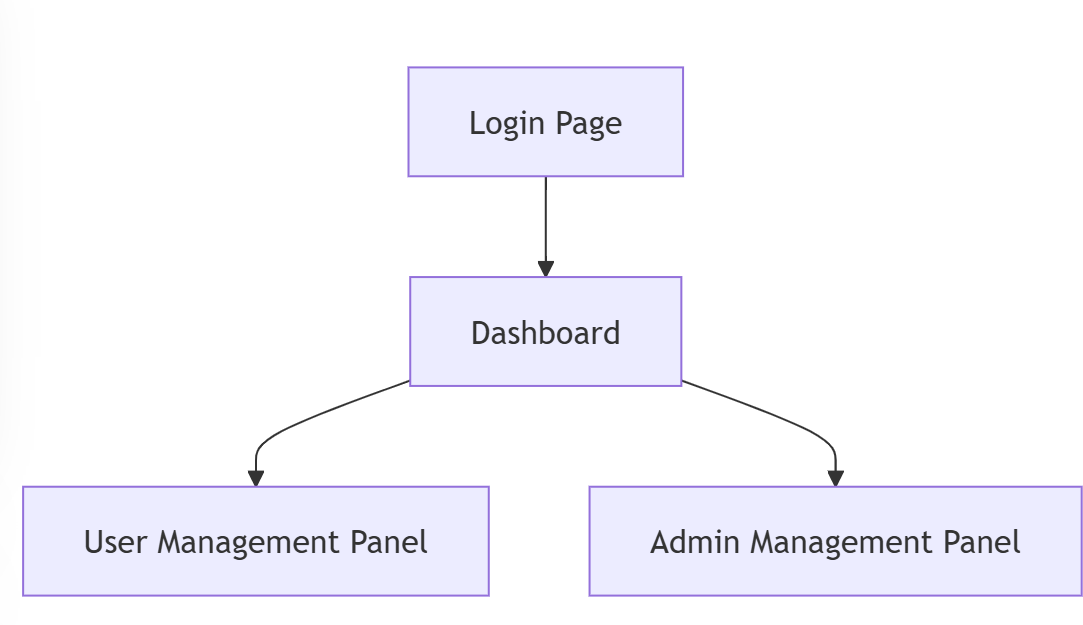
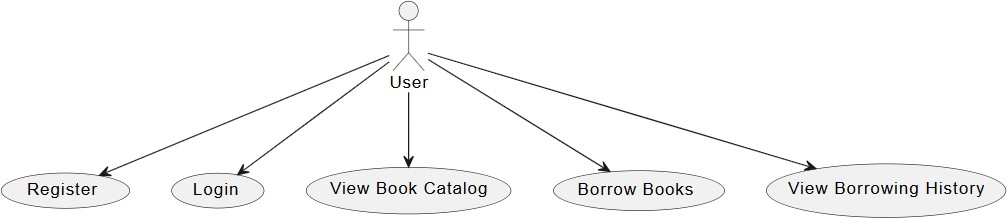


Fig-4.4 UI Design

## Security Issues

The system incorporates several security measures to protect user data and prevent unauthorized access. Authentication is handled using JSON Web Tokens (JWT), ensuring that only authenticated users can access restricted areas of the system. Role- based access control (RBAC) is implemented to differentiate between admin and user functionalities. Use case diagrams demonstrate how different roles interact with the system for secure operations.



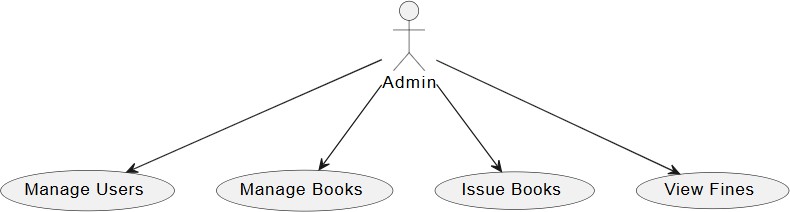


Fig-4.5Use Case Diagram

## Test Cases Design

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case ID | Test Case Description | Input & Steps | Expected Output |
|  |  |  |  |
| TcID0001 | Check customer Login with valid data. | Username: [ns814167@gmail.com](mailto:ns814167@gmail.com)  Password: \*\*\*\*\*\* | User should Login into an Application. |
| TcID0002 | Check customer Login with invalid data. | Username: [harsh243p@gmail.com](mailto:harsh243p@gmail.com) Password: \*\*\*\*\*\* | User should not Login into an Application. |
| TcID0003 | Check if the loading page loading efficient for the customer. | Click on the “Login” button | Welcome to the Home Page. |
| TcID0004 | Customer can see the products. | Customer click on the “Menu” button | Product list will be open. |
| TcID0005 | Customer can add the product in “Add to Cart” section. | Clicks on “Add to Cart” | Product will be added in cart. |
| TcID0006 | Customer can Buy the products. | Clicks on “Check Out” | Go on payment gateway. |
| TcID0007 | Second time, when customer try to buy products. | Go on “Add to Cart” section and click on “Check Out” | Go on payment gateway. |

Test cases are designed to verify the functionality and performance of each module. Unit testing is conducted to ensure individual components are working as expected, and integration testing verifies that the modules work together seamlessly. Sample test cases include testing the user registration process, verifying the menu search functionality, and checking the order total calculation algorithm. Additionally, tests are conducted to ensure accurate delivery status updates and payment processing for successful transactions.

Fig-4.6 Test Case Diagram