Report on

Cloud Restaurant

Prepared for

Dr. Taslim Taher **Assistant Professor** **Department of CSE**	**Ms. Tasnuva Binte Rahman** **Lecturer** **Department of CSE**

Course No: CSE 3224
Course Name: Information System Design & Software Engineering Lab

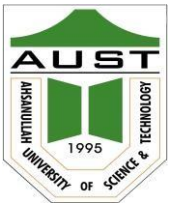
Prepared by

Lab Section: A1

Group No: 03

Submitted by -
20210104005 : Naeema Jannat
20210104008 : Mostafiz Fahim
20210104009 : Asadut Jaman

Date: 17.07.2024



Department of Computer Science and Engineering

List of Contents:

Summary of the Project:

The "Cloud Restaurant" project is an online platform designed to streamline the food ordering process for a virtual restaurant. Customers can explore the digital menu, read item descriptions, check prices, place orders, pay through various online payment options, and monitor the status of their orders from preparation to delivery. For the restaurant administrators, the system offers real-time order management, status updates, and easy menu modifications, such as adding or removing dishes and adjusting prices. Delivery staff receive detailed order information, including customer addresses and ordered items, and can update the delivery status upon completion. This system aims to enhance the efficiency and convenience of ordering, preparing, and delivering food for all participants.

Background Study:

1. Shifting Consumer Behavior:
    - Evolution in food consumption patterns due to changing lifestyles and preferences.
    - Increased demand for food delivery services attributed to urbanization andhectic schedules.
    - Growing preference for hassle-free meal solutions, driving reliance on food delivery platforms.
2. Technological Advancements in the Food Industry:
    - Significantly impacted by widespread smartphone usage and internet accessibility.
    - Enhanced customer experiences facilitated by intuitive interfaces and personalized recommendations.
3. Economic Opportunities and Market Growth:
    - Restaurants benefit from expanded customer reach and increased revenue without significant infrastructure investment.
    - Delivery partners and platform operators generate income through various revenue models.
4. Competitive Landscape:
    - Characterized by intense competition among aggregator platforms, restaurant chains, and independent delivery services.
    - Focus on key competitive factors such as customer experience, deliveryefficiency, menu diversity, and pricing strategies.

Requirement Analysis:

Functional:

1. User Registration and Authentication:
    - Customers should be able to register by providing basic information likename, email, phone number, and delivery address.
    - Secure login authentication for accessing their accounts and performingactions like

browsing the restaurant's menu and placing orders.

2. Profile Management:
   - Customers should have the ability to create and manage their profiles, including preferences like favorite dishes, dietary restrictions, and payment methods.
3. Menu Listings with Detailed Descriptions:
   - The website should display a comprehensive menu with detailed descriptions, images, prices, and any special offers.
   - Dietary information should be clearly indicated for each menu item.
4. Order Placement and Customization:
   - Customers should be able to add items to their cart and specify delivery preferences.
5. Payment Integration:
   - Integration with secure payment gateways for processing online payments, including credit/debit cards and digital wallets.
6. Order Tracking and Management:
   - order tracking functionality allowing customers to monitor the status of theirorders from preparation to delivery.
   - Order history and reorder functionality for quick and easy reordering of favorite dishes.
7. Rating and Feedback System:
   - Ability for customers to rate dishes based on taste, presentation, and overall experience.
   - Restaurant management should have the option to respond to feedback and address any issues raised by customers.

Non-Functional Requirements:

1. Performance:
   - Ensure fast loading times and responsiveness, especially during peak hours,to provide a seamless browsing and ordering experience.
   - Optimize server-side and client-side performance to handle concurrent user requests efficiently.
2. Security:
   - Implement encryption for user data and payment information to ensuresecure transactions.
3. Scalability:
   - Design the website architecture to accommodate increasing user traffic andmenu items without compromising performance.
   - Implement scalable database solutions and caching mechanisms to handlegrowth effectively.
4. Usability:
   - Create an intuitive user interface with clear navigation and visually appealingdesign to enhance user experience.

- Provide descriptive labels and tooltips to guide users through the orderingprocess seamlessly.

5. Reliability:
   - Minimize downtime by implementing robust error handling mechanisms and monitoring tools to detect and resolve issues promptly.
   - Ensure data integrity and consistency to prevent loss of user data or menu information.
6. Compatibility:
   - Ensure compatibility with a wide range of web browsers and devices to reacha broader audience.

Test the website thoroughly on different platforms and screen sizes to ensure consistent performance across devices.

Feasibility Analysis:

Economic Feasibility:

1. Low-cost Development:
   - The project adopts cost-effective development strategies, utilizing existing technologies and minimizing expenses related to software development and infrastructure setup. This approach helps in optimizing resources and reducinginitial investment.

2. Short Time to Market:
   - Leveraging established web development frameworks and tools allows for accelerated development timelines, ensuring a swift launch to capitalize on market opportunities. This quick entry into the market helps in gaining early traction and competitive advantage.
3. Cost Scalability:
   - The project's cost structure remains flexible and scalable, aligning expenses with revenue growth. As the platform expands its user base and features, costsincrease gradually, ensuring sustainability and profitability in the long term.

Technical Feasibility:

1. Existing Technology Availability:
   - The project relies on readily available technologies and frameworks, eliminating the need for new or specialized technology. This enhances technical feasibility by leveraging proven solutions and minimizing development complexities.
2. No New Technology Required:
   - By utilizing existing technologies, the project reduces risks associated with technology adoption and compatibility issues. This approach ensures smoother

development processes and reduces the likelihood of technical challenges

.

Operational Feasibility:

1. Low Maintenance:
    - The platform will be designed for low maintenance, with automated processes and robust systems reducing the need for frequent manual intervention. This enhances operational efficiency and minimizes ongoing maintenance costs.
2. Minimal Moderation:
    - With a streamlined user experience and automated content moderation tools, the platform requires minimal moderator intervention. This reduces operational overhead and ensures a smooth user experience, enhancing overall operational feasibility.

Risk Analysis :

1. **Market Competition:** The shop may face stiff competition from established online food delivery platforms and other local restaurants offering similar cuisine, making it challenging to attract and retain customers.
2. **Technical Issues:** The online ordering system could encounter technical issues such as website crashes, slow loading times, or payment processing errors, leading to a poor user experience and customer frustration.
3. **Data Security:** Storing customer data such as personal information and payment details may pose security risks, including the possibility of data breaches or cyberattacks, resulting in financial losses and reputational damage.
4. **Supply Chain Disruptions:** Dependency on suppliers for ingredients and delivery services for timely order fulfillment exposes the shop to risks such as ingredient shortages, delivery delays, or quality issues, impacting customer satisfaction and revenue.
5. **Inventory Management:** Not enough synchronization between online orders and inventory levels could result in out-of-stock items being displayed on the website, leading to customer annoyance and order cancellations.
6. **Staff Training:** Poor training for restaurant staff on how to manage online orders and utilize the website's features effectively could lead to order processing errors and customer dissatisfaction.
7. **Device Compatibility:** Different web browsers and devices may interpret HTML, CSS, and JavaScript code differently, leading to inconsistencies in website appearance and functionality.
8. **Financial Risks:** Fluctuations in food prices, operating costs, and revenue may impact the shop's profitability, especially during slow periods or economic downturns.

9. **Customer Feedback and Reputation:** Negative reviews or complaints about food quality, delivery times, or customer service could harm the shop's reputation and deter potential customers from ordering in the future.

Challenges:

1. **Integration of Multiple Technologies:** Combining HTML, CSS, C# and MySQL requires careful coordination to ensure compatibility and functionality across the website.
2. **Managing Complex Data Interactions:** Handling customer orders, payments, and inventory updates in real-time while maintaining data integrity poses challenges in managing complex data interactions.
3. **User Experience Design:** Designing a user-friendly interface that is easy to navigate for both customers placing orders and restaurant staff managing the platform is critical for a successful project.
4. **Balancing Security and Usability:** Ensuring the security of sensitive customer information without sacrificing usability presents a challenge, as maintaining a high-security standard while providing a seamless user experience can be difficult.
5. **Meeting Project Deadlines:** Unexpected technical issues may arise, requiring effective project management to ensure that project deadlines are met despite these challenges.

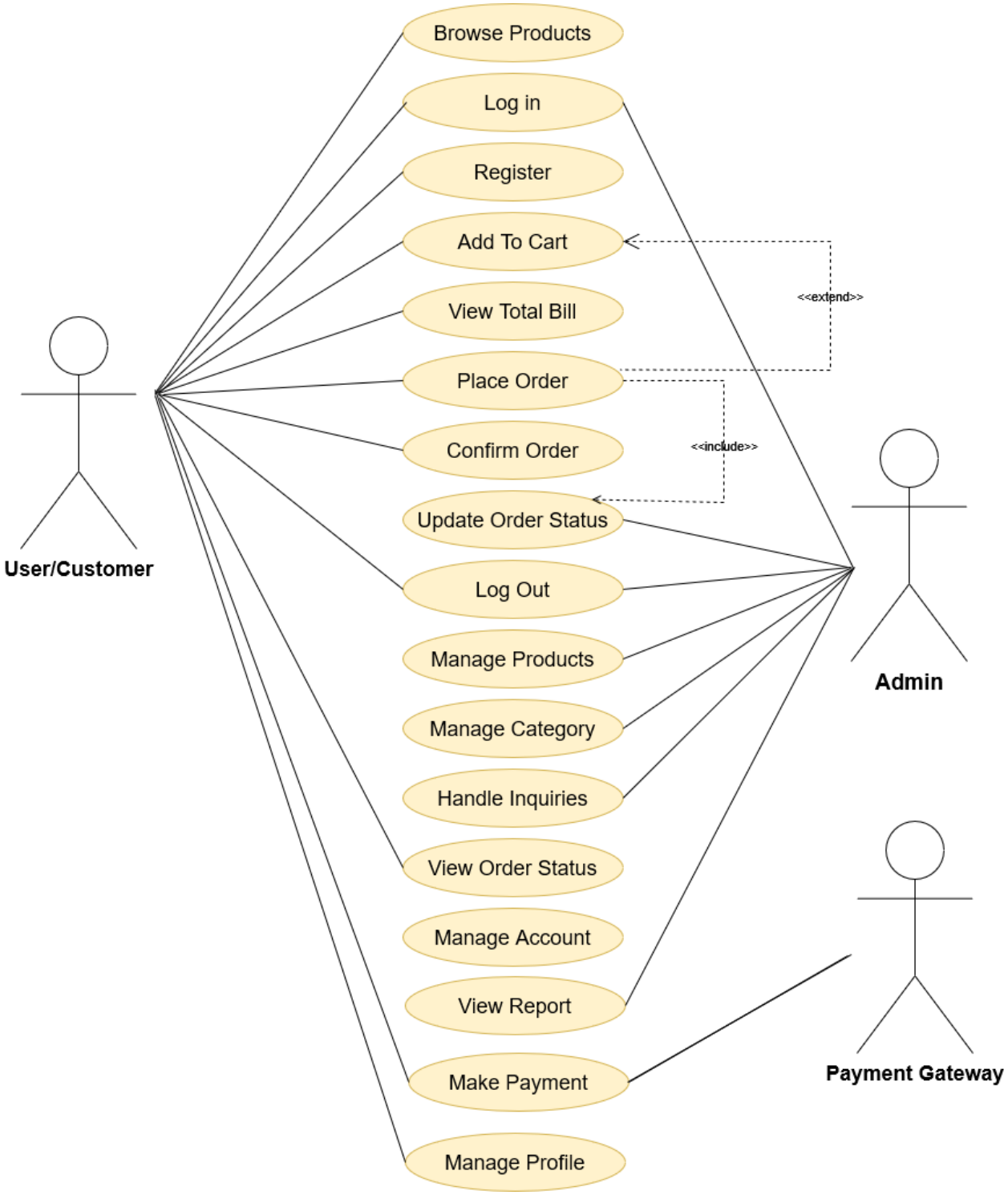
Project Goals:

The selection of an "Online food ordering website" project named Cloud Restauranthas driven by a combination of factors, including market demand, competitive landscape, convenience and accessibility, technological advancements, revenue potential, and customer experience. Research indicating a growing trend in online food ordering can justify the project's selection. Providing customers with a convenient way to order food from a restaurant, leveraging cutting-edge technologyfor seamless ordering and delivery, and identifying multiple revenue streams such as transaction fees and partnerships are key considerations. Creating new job opportunities is another crucial factor to consider when selecting an online food ordering project. Additionally, prioritizing a user-friendly interface and efficient delivery logistics to enhance the customer experience can further validate the decision to pursue the project.

Used Technologies for the project:

- Frontend: HTML, CSS, C# ASP.NET
- Backend: C# ASP.NET
- Database: MS SQL Server
- Platform: Web Application

Use Case Diagram:



The detailed use cases for the online restaurant system ensure a comprehensive understanding of the necessary functionalities for both customers and administrators. Key features include product browsing, cart management, account handling, order processing, and inquiry management. Prioritizing these functions and incorporating dependencies (like <<include>> and <<extend>>) clarifies the interactions and flow within the system. This design approach aims to create a seamless and efficient user experience, ultimately ensuring the system's success and user satisfaction.

Detailed Descriptions of Different Scenarios:

Browse Products:

- Actors: Customer/User
- Description: The customer browses the menu to view available dishes and their details.
- Priority: High
- Prerequisite: The customer must have access to the restaurant's website.

View Product Details:

- Actors: Customer/User
- Description: The customer views detailed information about a specific product.
- Priority: High
- Prerequisite: The customer must have browsed the products and selected a specific product to view.

Add Product to Cart:

- Actors: Customer/User
- Description: The customer adds a selected product to their shopping cart.
- Priority: High
- Prerequisite: The customer must have viewed the details of the product they wish to add to the cart.

Manage Cart:

- Actors: Customer/User
- Description: The customer reviews and modifies the contents of their shopping cart.
- Priority: High
- Prerequisite: The customer must have added at least one product to the cart.

Checkout:

- Actors: Customer/User
- Description: The customer proceeds to checkout to finalize their order.
- Priority: High
- Prerequisite: The customer must have products in their cart.

Make Payment:

- Actors: Customer/User, Payment Gateway
- Description: The customer completes the payment for their order using the integrated payment gateway.
- Priority: High
- Prerequisite: The customer must have proceeded to checkout and confirmed their order details.

Register Account:

- Actors: Customer/User
- Description: The customer registers an account to use the online restaurant services.
- Priority: Medium
- Prerequisite: The customer must have access to the restaurant's website.

Login:

- Actors: Customer/User
- Description: The customer logs in to access their account.
- Priority: High
- Prerequisite: The customer must have a registered account.

Manage Account:

- Actors: Customer/User
- Description: The customer updates their profile information.
- Priority: Medium
- Prerequisite: The customer must be logged in.

Place Order:

- Actors: Customer/User
- Description: The customer places an order after reviewing the cart and making the payment.
- Priority: High
- Prerequisite: The customer must have completed the checkout process and made a payment.

View Order Status:

- Actors: Customer/User
- Description: The customer checks the status of their placed order.
- Priority: Medium
- Prerequisite: The customer must have placed an order and be logged in.

Make Inquiry:

- Actors: Customer/User
- Description: The customer makes an inquiry regarding products, orders, or other services.
- Priority: Medium
- Prerequisite: The customer must be logged in.

Manage Categories:

- Actors: Admin
- Description: The admin manages the categories of products available on the menu.
- Priority: High
- Prerequisite: The admin must be logged in to the admin panel.

Manage Products:

- Actors: Admin
- Description: The admin manages the products available on the menu.
- Priority: High
- Prerequisite: The admin must be logged in to the admin panel.

Handle Inquiries:

- Actors: Admin
- Description: The admin reviews and responds to customer inquiries.
- Priority: Medium
- Prerequisite: The admin must be logged in to the admin panel.

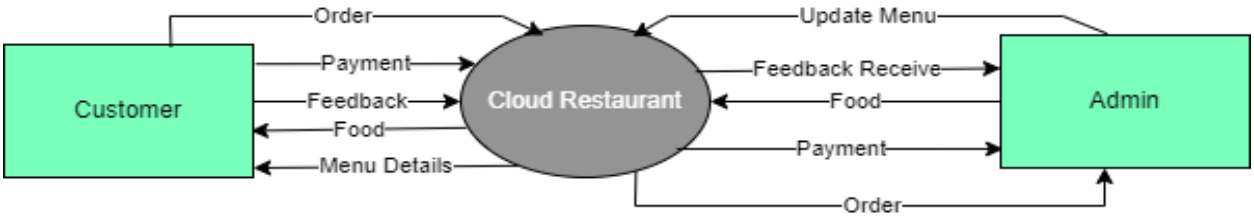
Data Flow Diagram:

The Data Flow Diagram (DFD) provides a visual representation of how information flows within the Cloud Restaurant System. It consists of different levels, each offering a more detailed view of the processes involved. This hierarchical structure of the DFD ensures a clear and comprehensive understanding of the Cloud Restaurant System's functionality, aiding in the system's design, development, and maintenance. The DFD is divided into the following components:

1. Context Level Diagram
2. Level 0 Diagram
3. Level 1-n Diagrams

Context Level Diagram

This highest-level diagram provides a broad view of the entire system, illustrating the interaction between external entities (such as Customers and Admins) and the Cloud Restaurant System.



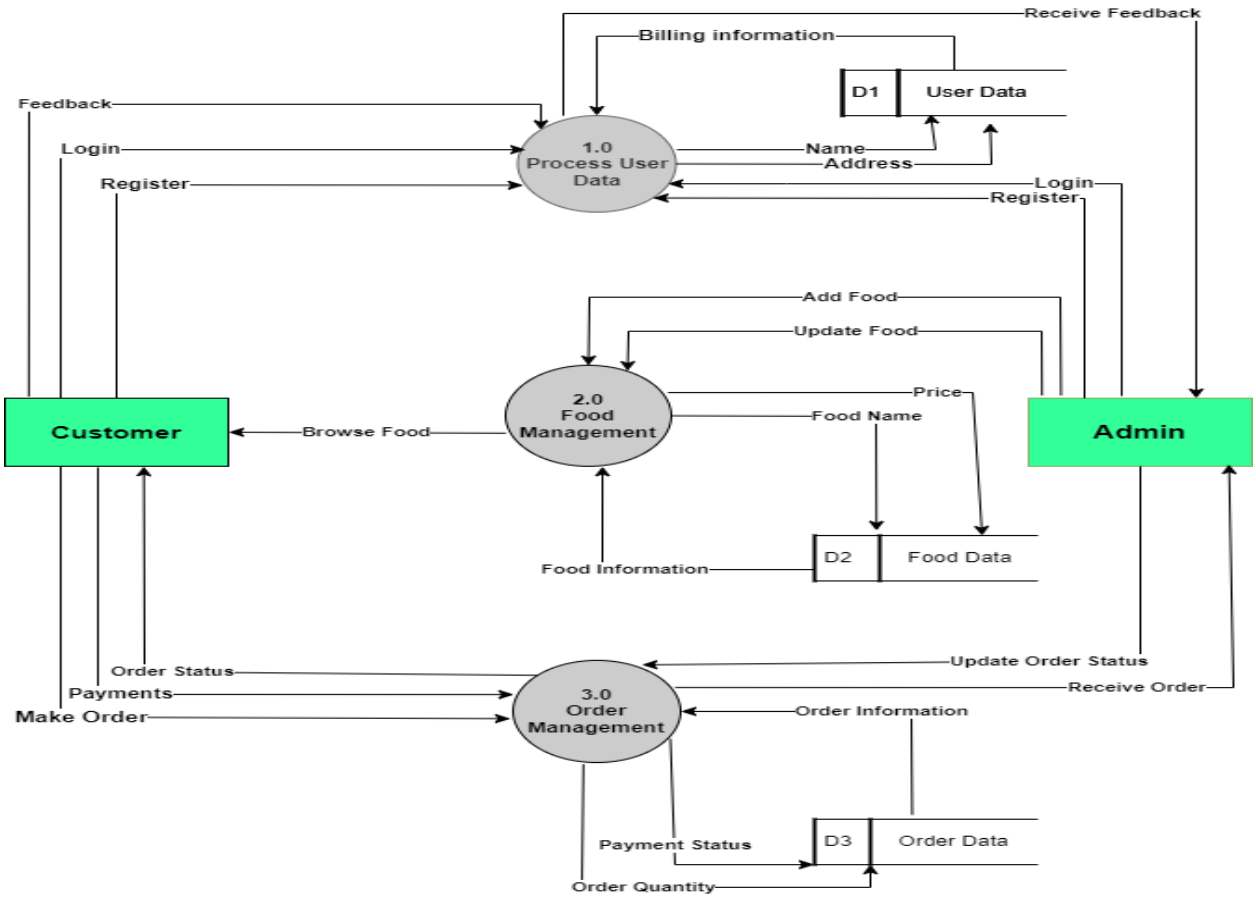
Explanation:

Customer The customer is the primary user of the Cloud Restaurant system. They interact with the system to browse the menu, place orders, make payments, and provide feedback.

Admin: The admin manages the Cloud Restaurant system. They are responsible for maintaining the menu, processing orders, and managing customer feedback.

Main System (Cloud Restaurant): The Cloud Restaurant system processes the data received from the customers and admins to perform its core functions. It consists of various internal processes such as managing user data, handling food items, and processing orders.

Level 0 Diagram



Explanation:

Process User Data (1.0): This process handles user registration and login, as well as storing user information like address and billing information.

Food Management (2.0): Handles menu management, including adding, updating, and retrieving food items.

Order Management (3.0): Manages order placement, processing, and payment.

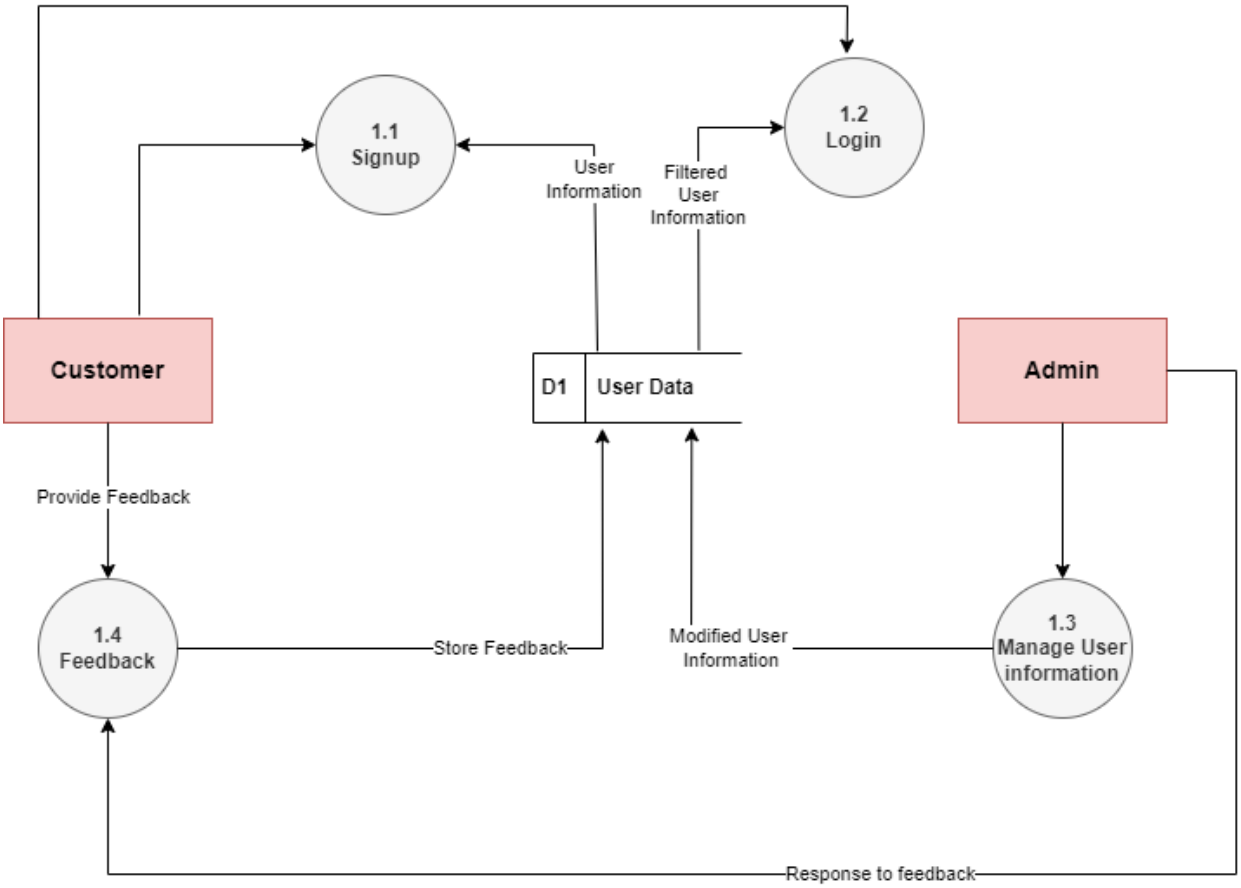
Data Stores:

User Data(D-1): Stores user-related information including names, addresses, and billing details.

Food Data (D2): Stores information about menu items such as names, descriptions, prices, and availability.

Order Data (D3): Stores order-related information including order details, quantities, prices, customer information, payment details, and order status.

Level 1 Diagrams



Explanation:

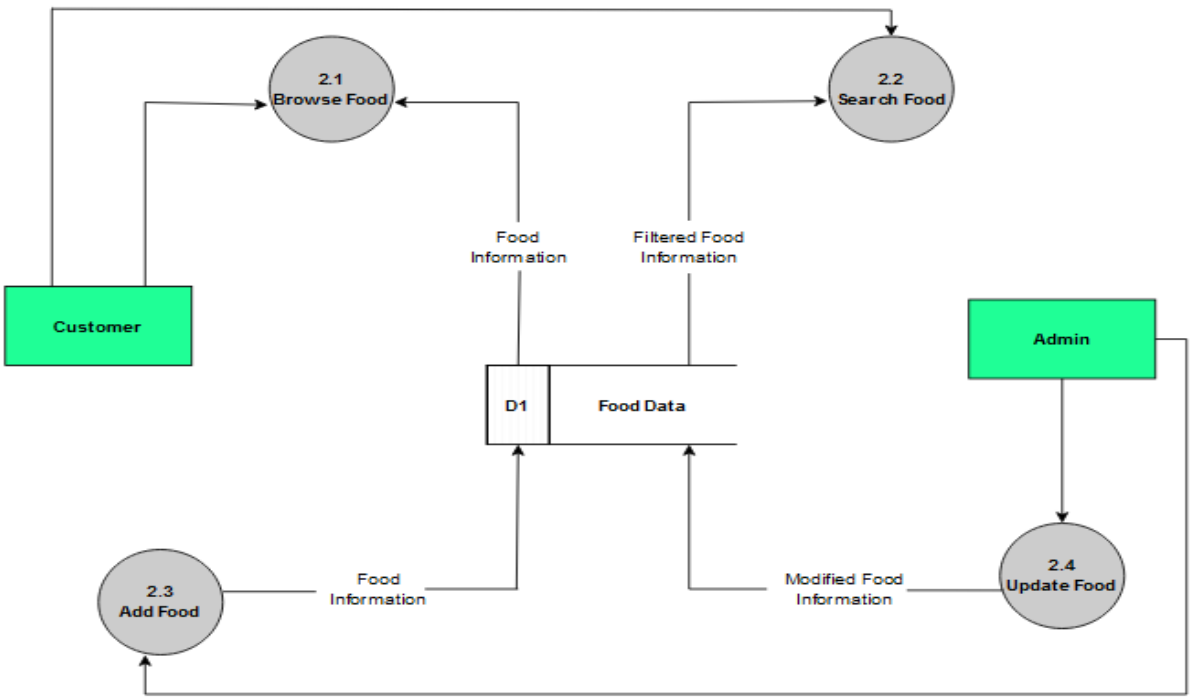
1. **Signup (1.1)**:
    - **Function**: Allows new users to register and create an account.
    - **Inputs**: User registration data (name, email, password, address).
    - **Outputs**: Registration confirmation.
    - **Data Store**: D1 - User Data
2. **Login (1.2)**:
    - **Function**: Enables existing users to log in using their credentials.
    - **Inputs**: Login credentials (username, password).
    - **Outputs**: Login confirmation.
    - **Data Store**: D1 - User Data
3. **Update User Information (1.3)**:
    - **Function**: Allows users to update their profile information.
    - **Inputs**: Updated user information (new address, billing details).

- **Outputs**: Update confirmation.
- **Data Store**: D1 - User Data

4. **Feedback (1.4)**:
   - **Function**: Collects customer feedback and ratings about their orders and overall experience.
   - **Inputs**: Feedback and ratings from customers.
   - **Outputs**: Feedback acknowledgment.
   - **Data Store**: D1 - User Data

Data Stores:

D1 - User Data: Stores user-related information, including names, addresses, emails, passwords, billing information, and customer feedback.



Browse Food (2.1): Allows customers to view available food items.

Search Food (2.2): Enables searching for specific food items based on criteria.

Add Food (2.3): Admin adds new food items to the system.

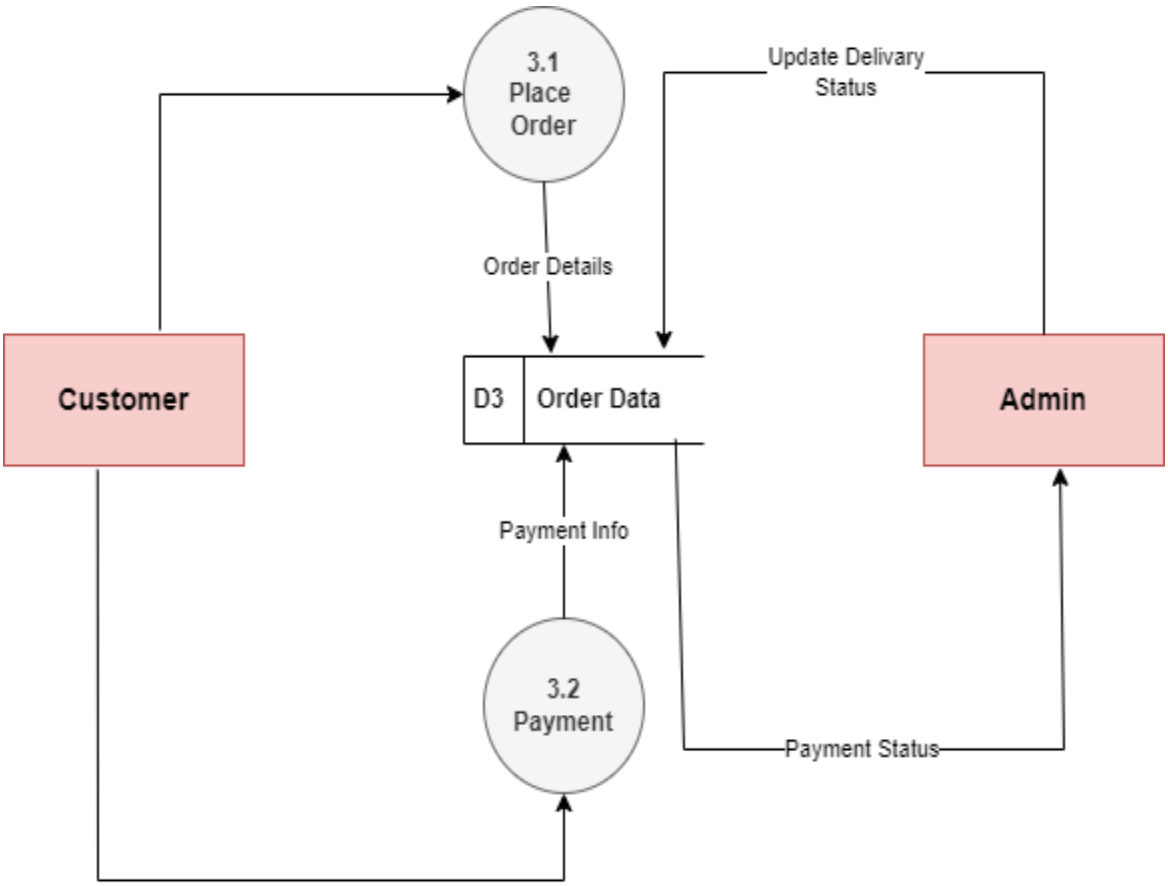
Update Food (2.4): Admin updates existing food item details.

Data Stores:

D1 - User Data: Stores user-related information including names, addresses, and billing information.

D2 - Food Data: Stores food-related information such as food names, prices, and descriptions.

D3 - Order Data: Stores order-related information including order details, quantities, and payment status.



1. **Place Order (3.1)**:
    - **Function**: Handles the placement of orders by customers.
    - **Inputs**: Order details (selected food items, quantities, delivery address, special instructions).
    - **Outputs**: Order confirmation.
    - **Data Store**: D3 - Order Data
2. **Make Payment (3.2)**:
    - **Function**: Processes payment for orders.
    - **Inputs**: Payment information (credit card details, payment method).
    - **Outputs**: Payment receipt.

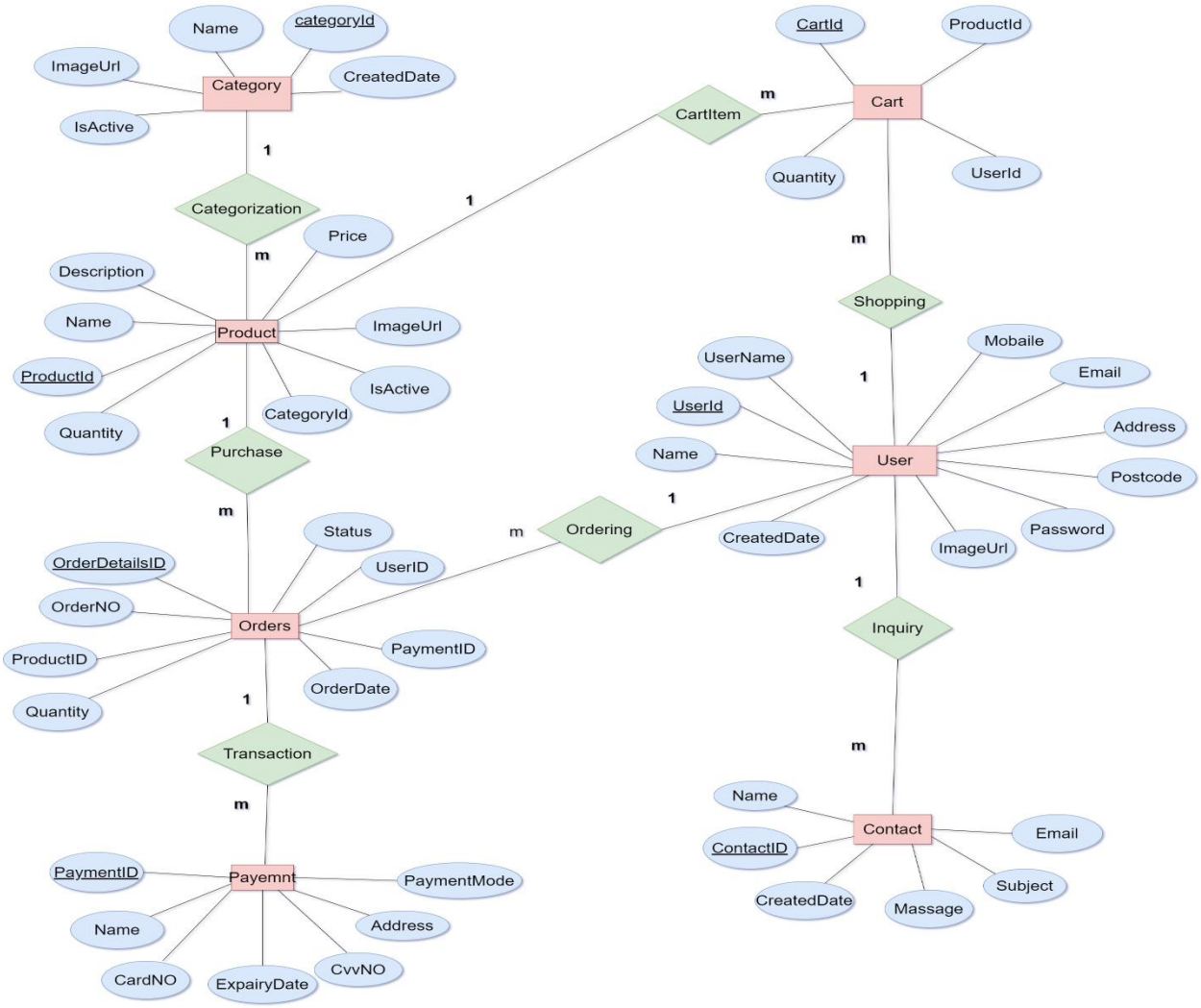
    - **Data Store**: D3 - Order Data
3. **Order Status Updates (3.3)**:
    - **Function**: Updates the status of orders (e.g., confirmed, preparing, out for delivery).
    - **Inputs**: Order status updates from the admin.
    - **Outputs**: Order status notifications to customers.
    - **Data Store**: D3 - Order Data
4. **Rating and Feedback (3.4)**:
    - **Function**: Collects customer feedback and ratings for their orders.
    - **Inputs**: Customer feedback and ratings.
    - **Outputs**: Feedback acknowledgment.
    - **Data Store**: D3 - Order Data

Data Stores:

D3 - Order Data: Stores order-related information, including:

- Order details (items, quantities, prices)
- Customer information (delivery address, contact details)
- Payment information (payment status, receipts)
- Order status updates (order progress, delivery status)
- Customer feedback and ratings

Entity Relation Diagram:



Entities and Attributes

- **Users**

- **UserID**: Primary Key, unique identifier for each user. (int, AUTO_INCREMENT)
- **UserName**: Name of the user. (varchar(255))
- **Email**: Email of the user. (varchar(255))
- **Password**: Password of the user. (varchar(255))
- **Mobile**: Mobile number of the user. (varchar(20))
- **Address**: Address of the user. (varchar(255))

- **Postcode**: Postal code of the user. (varchar(20))
- **ImageUrl**: Profile image URL of the user. (varchar(255))
- **CreatedDate**: Date the user was created. (datetime)

- **Products**

  - **ProductID**: Primary Key, unique identifier for each product. (int, AUTO_INCREMENT)
  - **Name**: Name of the product. (varchar(255))
  - **Description**: Description of the product. (text)
  - **Price**: Price of the product. (decimal(10, 2))
  - **Quantity**: Available quantity of the product. (int)
  - **ImageUrl**: Image URL of the product. (varchar(255))
  - **CategoryID**: Foreign Key, identifier for the product's category. (int)
  - **IsActive**: Status of the product, whether it is active or not. (boolean)
  - **CreatedDate**: Date the product was created. (datetime)

- **Categories**

  - **CategoryID**: Primary Key, unique identifier for each category. (int, AUTO_INCREMENT)
  - **Name**: Name of the category. (varchar(255))
  - **ImageUrl**: Image URL of the category. (varchar(255))
  - **IsActive**: Status of the category, whether it is active or not. (boolean)
  - **CreatedDate**: Date the category was created. (datetime)

- **Carts**

  - **CartID**: Primary Key, unique identifier for each cart. (int, AUTO_INCREMENT)
  - **UserID**: Foreign Key, identifier for the user who owns the cart. (int)
  - **ProductID**: Foreign Key, identifier for the product in the cart. (int)
  - **Quantity**: Quantity of the product in the cart. (int)

- **Orders**

  - **OrderID**: Primary Key, unique identifier for each order. (int, AUTO_INCREMENT)
  - **OrderNO**: Order number. (varchar(50))
  - **UserID**: Foreign Key, identifier for the user who placed the order. (int)
  - **OrderDate**: Date the order was placed. (datetime)
  - **Status**: Status of the order. (varchar(50))

- **OrderDetails**

  - **OrderDetailsID**: Primary Key, unique identifier for each order detail. (int, AUTO_INCREMENT)
  - **OrderID**: Foreign Key, identifier for the order. (int)
  - **ProductID**: Foreign Key, identifier for the product. (int)

    - **Quantity**: Quantity of the product in the order. (int)

- **Payments**

    - **PaymentID**: Primary Key, unique identifier for each payment. (int, AUTO_INCREMENT)
    - **OrderID**: Foreign Key, identifier for the order being paid. (int)
    - **PaymentMode**: Mode of payment (e.g., credit card, PayPal). (varchar(50))
    - **CardNO**: Card number for the payment. (varchar(20))
    - **ExpiryDate**: Expiry date of the card. (varchar(10))
    - **CVVNO**: CVV number of the card. (varchar(10))

- **Contacts**

    - **ContactID**: Primary Key, unique identifier for each contact inquiry. (int, AUTO_INCREMENT)
    - **Name**: Name of the person making the inquiry. (varchar(255))
    - **Email**: Email of the person making the inquiry. (varchar(255))
    - **Subject**: Subject of the inquiry. (varchar(255))
    - **Message**: Message content of the inquiry. (text)
    - **CreatedDate**: Date the inquiry was created. (datetime)

Relationships

- **Users - Carts**

    - **One-to-Many**: One user can have multiple carts.
    - **Foreign Key**: `UserID` in the `Carts` table references `UserID` in the `Users` table.

- **Users - Orders**

    - **One-to-Many**: One user can place multiple orders.
    - **Foreign Key**: `UserID` in the `Orders` table references `UserID` in the `Users` table.

- **Users - Payments**

    - **One-to-Many**: One user can make multiple payments.
    - **Foreign Key**: `UserID` in the `Payments` table references `UserID` in the `Users` table.

- **Products - Categories**

    - **Many-to-One**: Many products belong to one category.
    - **Foreign Key**: `CategoryID` in the `Products` table references `CategoryID` in the `Categories` table.

- **Carts - Products**

  - **Many-to-One**: Many cart items can be of one product.
  - **Foreign Key**: `ProductID` in the `Carts` table references `ProductID` in the `Products` table.

- **Orders - OrderDetails**

  - **One-to-Many**: One order can have multiple order details.
  - **Foreign Key**: `OrderID` in the `OrderDetails` table references `OrderID` in the `Orders` table.

- **Products - OrderDetails**

  - **Many-to-One**: Many order details can be of one product.
  - **Foreign Key**: `ProductID` in the `OrderDetails` table references `ProductID` in the `Products` table.

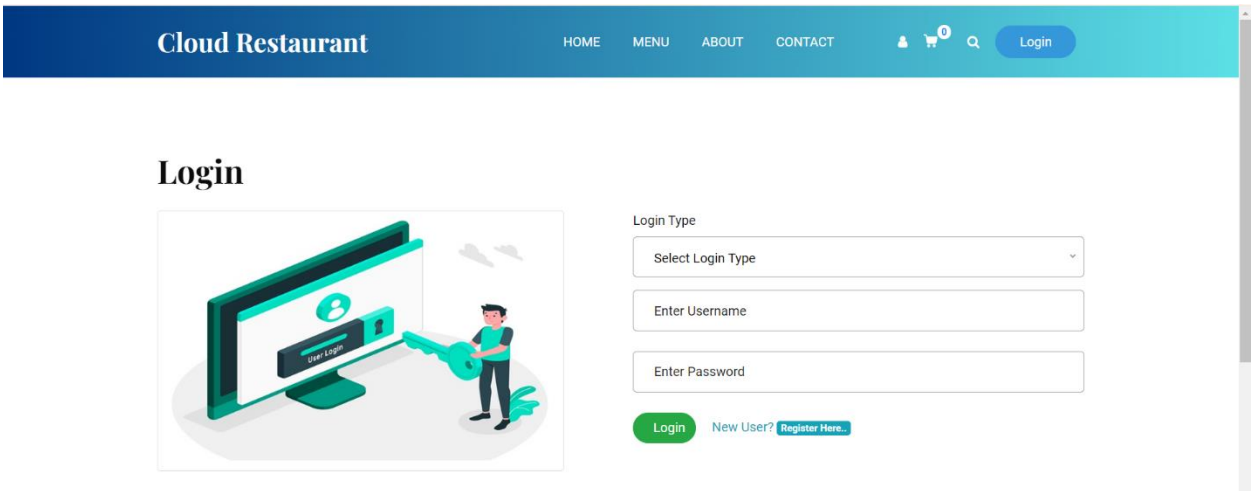
- **Orders - Payments**

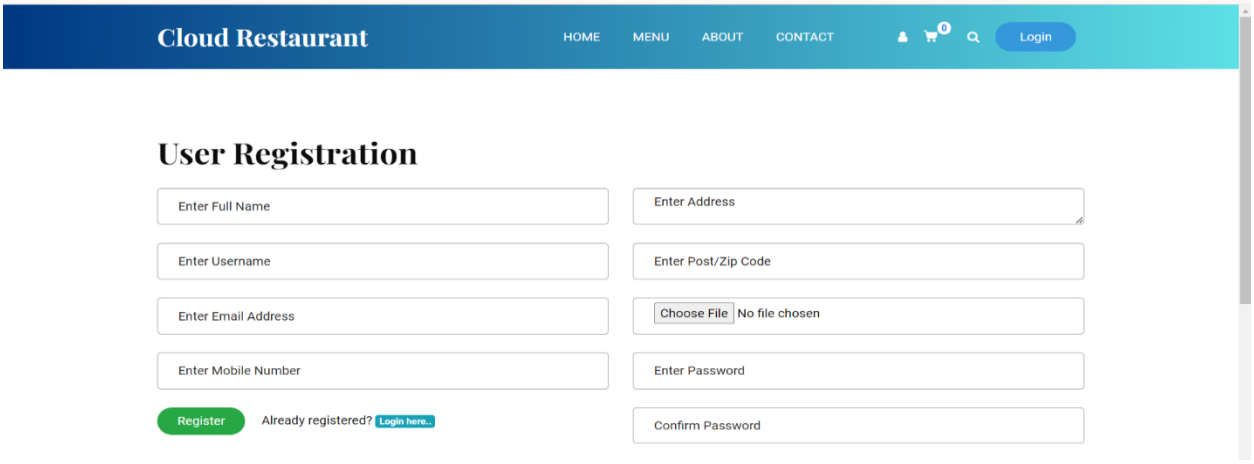
  - **One-to-Many**: One order can have multiple payments (e.g., split payments).
  - **Foreign Key**: `OrderID` in the `Payments` table references `OrderID` in the `Orders` table.

Project Features:

- User Registration and Login:

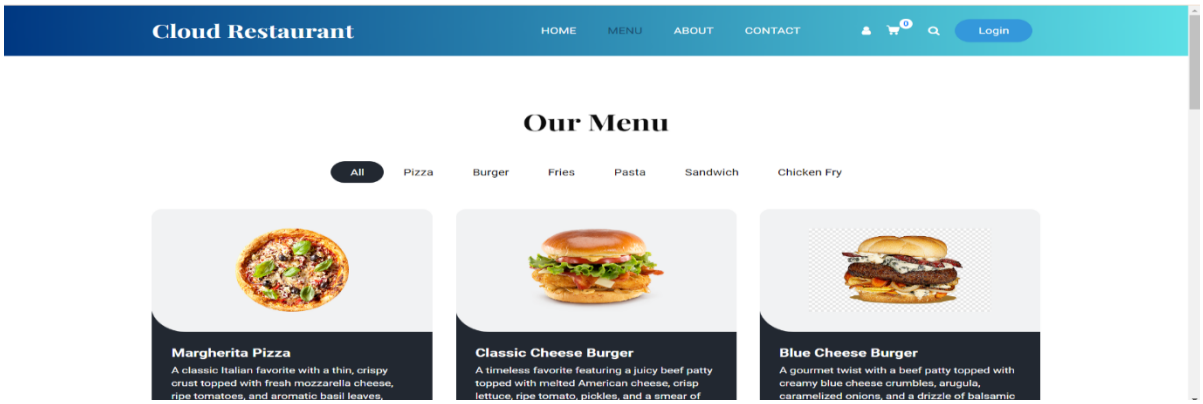
  Users and restaurant admins can create accounts and log in to the system. This ensures that each user has a personalized experience and their information is secure.





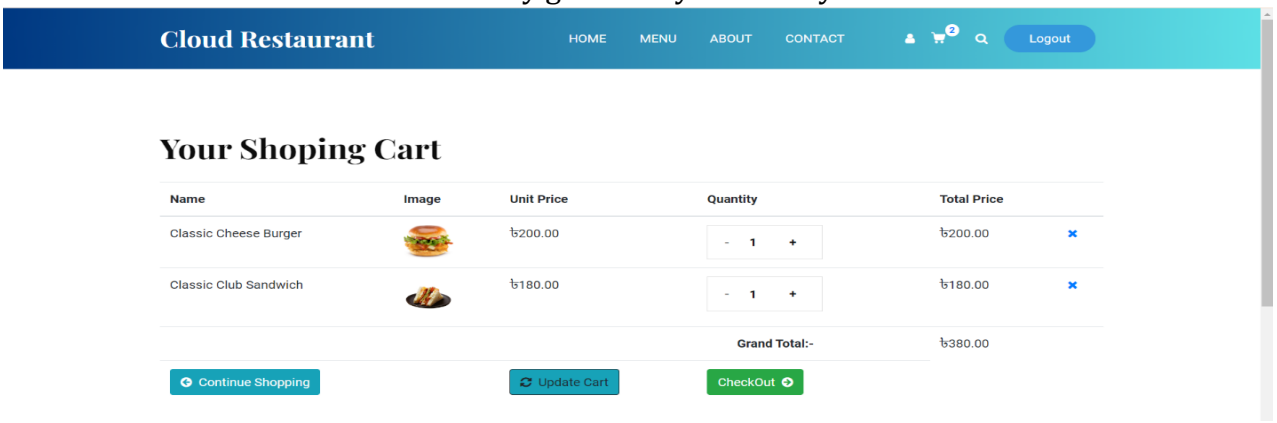
- Menu Browsing:

  Users can easily browse through the restaurant's menu. Each item includes a description, price, and possibly an image, making it easy for users to choose what they want to order.

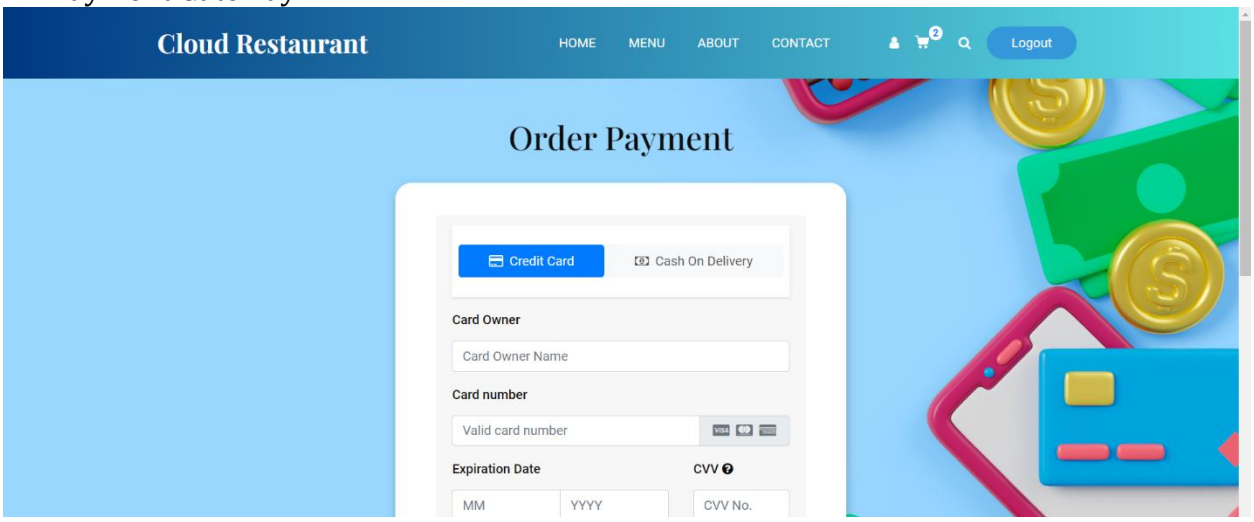


- Order Placement:

  Users can select items from the menu, add them to their cart, and customize their orders. This ensures they get exactly what they want.



- Payment Gateway:



- Admin Dashboard:

The restaurant admin has access to a dashboard where they can manage all orders, update their status, and modify the menu (e.g., adding new items or changing prices). This helps keep the restaurant's operations running smoothly.

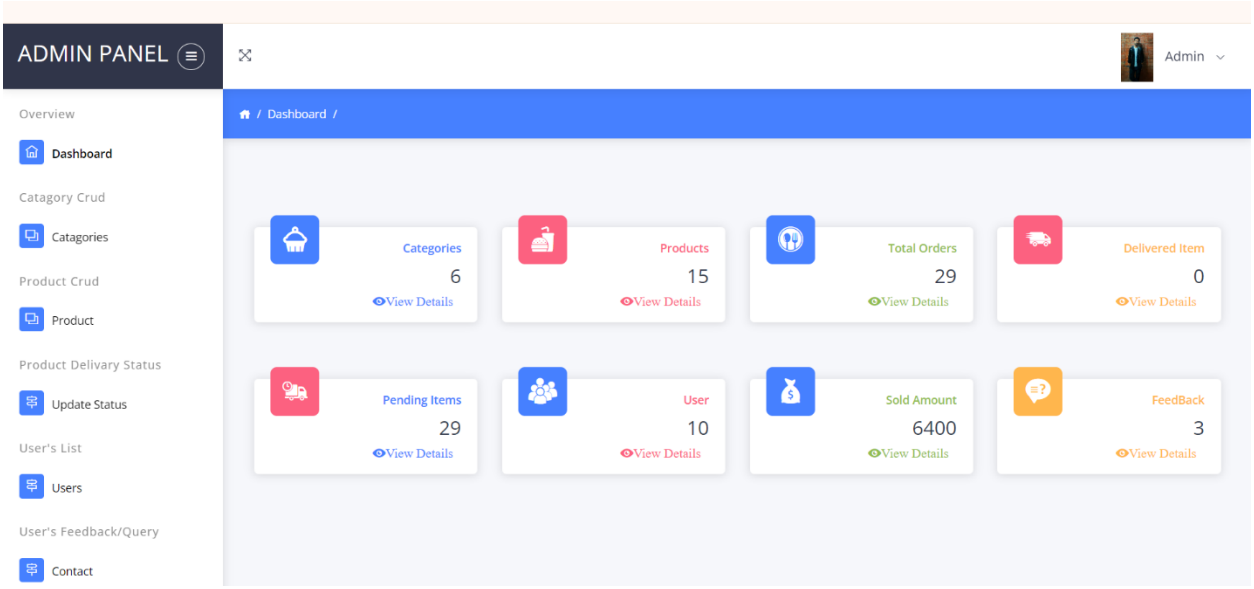


Fig: Admin Dashboard

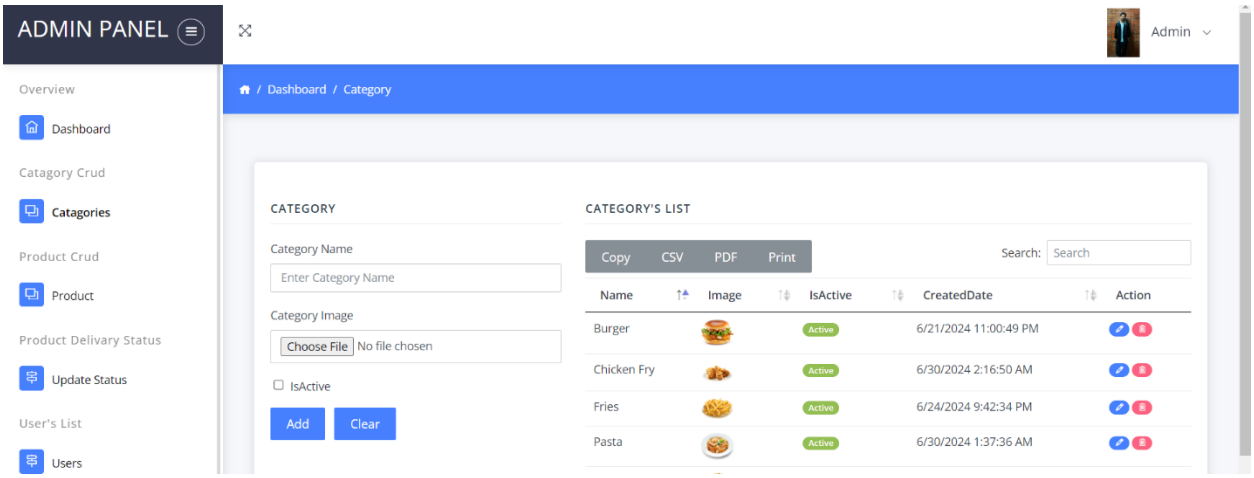


Fig: Manage Category

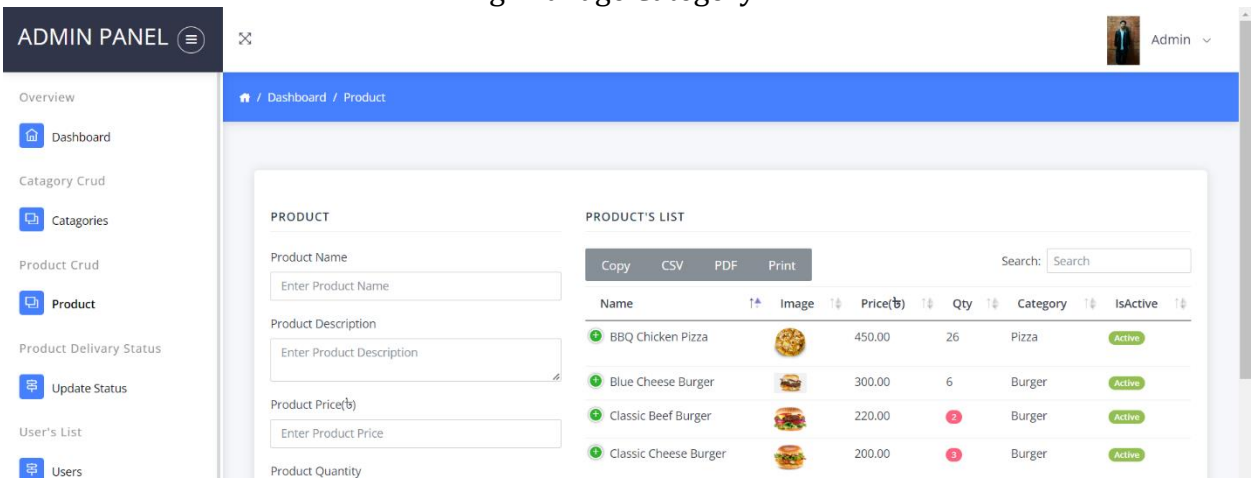


Fig: Manage Products

Opportunities of Development:

1. **User Experience Enhancement:**

    - **Personalization:** Implement algorithms to tailor recommendations and offers based on user behavior and preferences, enhancing engagement.
    - **User Interface:** Design intuitive interfaces across web and mobile platforms for seamless navigation and a visually appealing experience.
    - **Real-time Updates:** Provide customers with instant updates on order status and delivery tracking to improve transparency and satisfaction.

2. **Operational Efficiency:**

    - **Inventory Management:** Deploy advanced systems to monitor stock levels, predict demand, and automate replenishment processes, minimizing shortages.
    - **Order Management:** Streamline order processing to handle peak times efficiently, reducing errors and improving overall service speed.
    - **Automation:** Utilize automation tools for routine tasks like scheduling, billing, and reporting to increase accuracy and free up staff time.

3. **Data Analytics:**

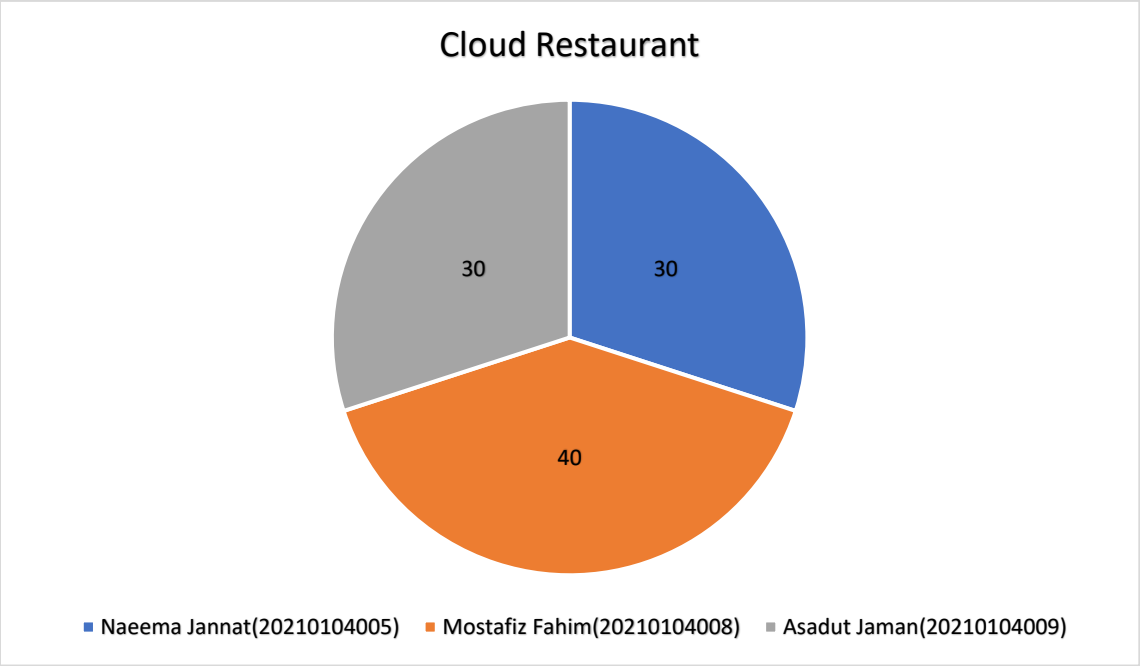
    - **Customer Insights:** Analyze customer data to understand preferences, behaviors, and feedback, enabling targeted marketing and menu adjustments.
    - **Sales Analysis:** Monitor sales trends and item performance to optimize inventory levels, pricing strategies, and promotional efforts.
    - **Performance Metrics:** Track key indicators such as delivery times and customer satisfaction scores to continually enhance service quality.

4. **Integration with Third-party Services:**

    - **Payment Gateways:** Integrate multiple payment options for customer convenience and security, including digital wallets and online banking.
    - **Delivery Services:** Partner with reliable third-party services to expand delivery coverage and ensure timely service during peak demand.
    - **Marketing Platforms:** Connect with social media and marketing channels to engage customers, gather feedback, and drive promotions effectively.

These areas collectively contribute to improving operational effectiveness, customer satisfaction, and market competitiveness of our cloud restaurant project.

Contribution:



Conclusion:

The Cloud Restaurant System streamlines food ordering by providing a user-friendly interface for customers to place orders seamlessly, efficient order management tools for administrators to track and handle incoming orders, and easy status updates for delivery personnel to ensure timely deliveries. Through its intuitive design and robust functionality, the system enhances the overall dining experience by fostering smooth communication and coordination between stakeholders, ultimately leading to heightened satisfaction for all involved parties.