TITLE OF THE PROJECT:

Canteen Management Portal

2.0 Introduction:

The "Canteen Management Portal" project is a student-led initiative focused on enhancing the efficiency of in-house canteen operations within a corporate environment. This web portal aims to provide a seamless and user-friendly experience for employees, offering features such as streamlined meal bookings, transparent venue reservations, and community engagement tools. Developed as a part of a student's portfolio, the project emphasizes modernization through a responsive design and Progressive Web App (PWA) integration. By addressing the evolving needs of the company's workforce, the "Canteen Management Portal" seeks to optimize internal services, laying the groundwork for future enhancements and improved corporate experiences.

2.1 Project Context:

In response to the evolving needs of the company's workforce, there is a compelling need to modernize and optimize internal services. The existing systems for canteen management and venue booking can be significantly enhanced to better align with the dynamic requirements of the employees. This project is a proactive effort to address these challenges, providing a user-friendly, responsive web portal, and incorporating Progressive Web App (PWA) technology for seamless access across various devices, including smartphones and laptops.

3.0 Project Objectives

The core objectives of the "Canteen Management Portal" project are as follows:

- **Efficient Canteen Management:**
  Streamline day-to-day canteen operations by introducing dynamic menu displays, timely notifications for announcements and menu changes, and real-time updates by the canteen staff.

- **User-Centric Meal Booking:**
  Develop a user-friendly interface for employees to easily book meals, modify bookings within designated timeframes, and maintain a personalized history of meal preferences.

- **Transparent Venue Booking System:**
  Enhance the transparency and efficiency of the venue booking system with features like an availability calendar, streamlined booking requests, and an approval workflow for administrators.

- **Event Planning Capabilities:**
  Introduce features for seamless event planning within the corporate township, offering customization options for seating arrangements and equipment needs.

- **Analytics and Reporting:**
  Implement analytics tools to empower administrators with insights into popular dishes, user preferences, and overall usage patterns, enabling informed decision-making.

- **Community Engagement:**
  Promote community engagement by highlighting and encouraging participation in community events, fostering a sense of unity among the company's employees.

- **Multi-language Support:**
  Ensure inclusivity by incorporating multi-language support, addressing the diverse linguistic backgrounds of the company's workforce.

This project, "Canteen Management Portal," is a student endeavor to contribute to the optimization of internal services for the company. By achieving these objectives, the portal not only meets the current needs of the employees but also lays the groundwork for future enhancements and continuous improvement in corporate services.

4.0 Project Category:

The "Canteen Management Portal" project falls within the domain of Web Development, specifically focusing on Responsive Website and Progressive Web App (PWA) technologies. The project aims to leverage these technologies to create an intuitive and accessible web portal that caters to the diverse needs of users across various devices, including smartphones and laptops.

Technologies

The development of the "Canteen Management Portal" will involve the use of the following technologies:

Frontend:

- HTML5, CSS3, JavaScript
- React.js

Backend:

- Node.js with Express
- Database MySQL

Progressive Web App (PWA):

- Service Workers for offline functionality
- Manifest files for app-like experience

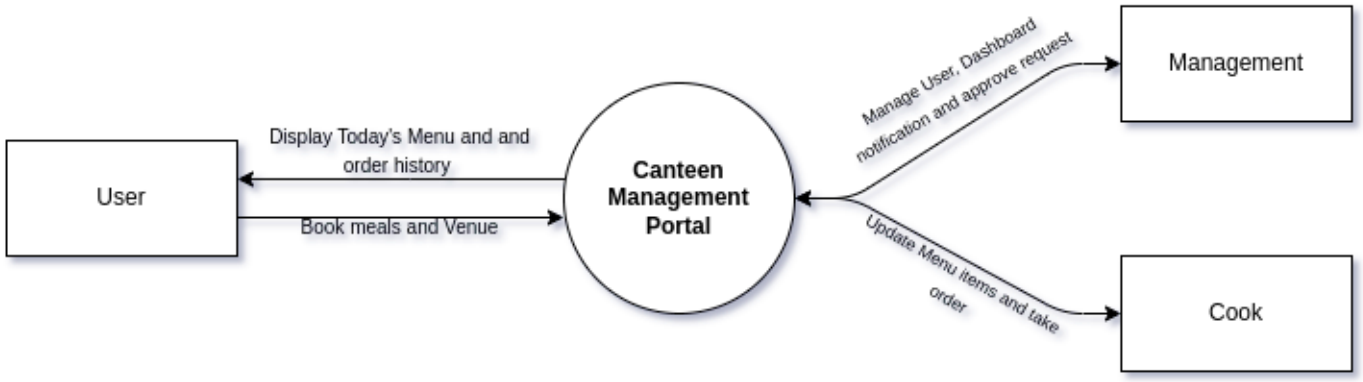
Additional Technologies:

- Version control – Git
- Visual Studio for writing code
- Deployment platforms – AWS / Microsoft Azure / Digital Ocean

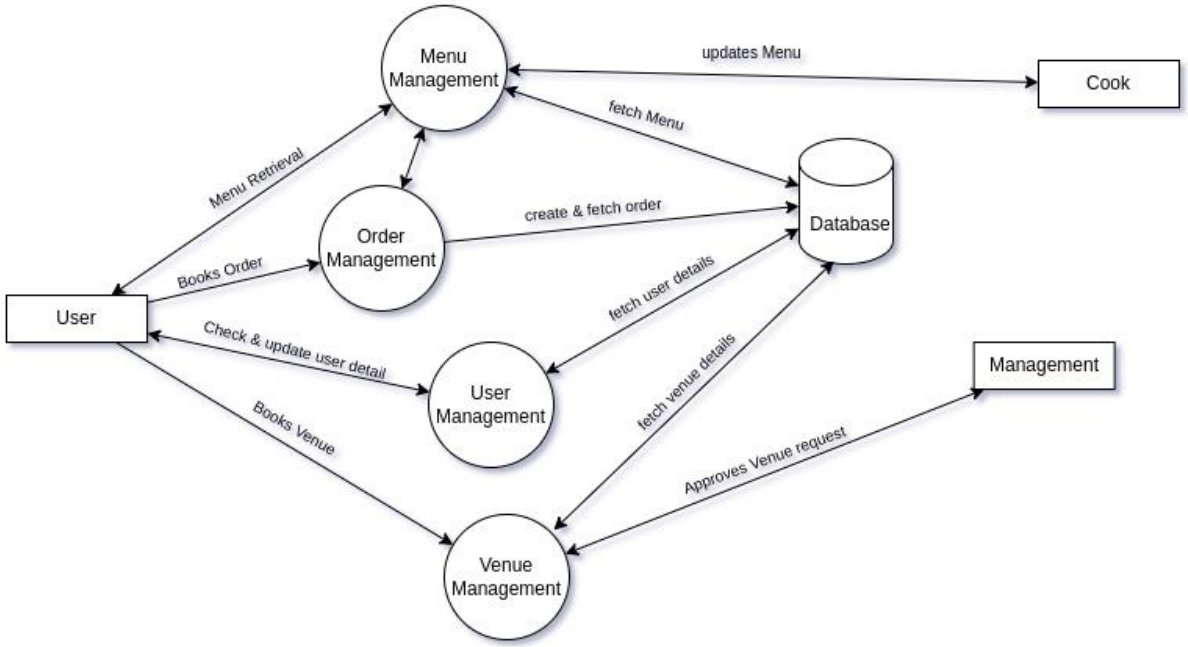
5.0 Analysis:

Data Flow Diagrams (DFDs):

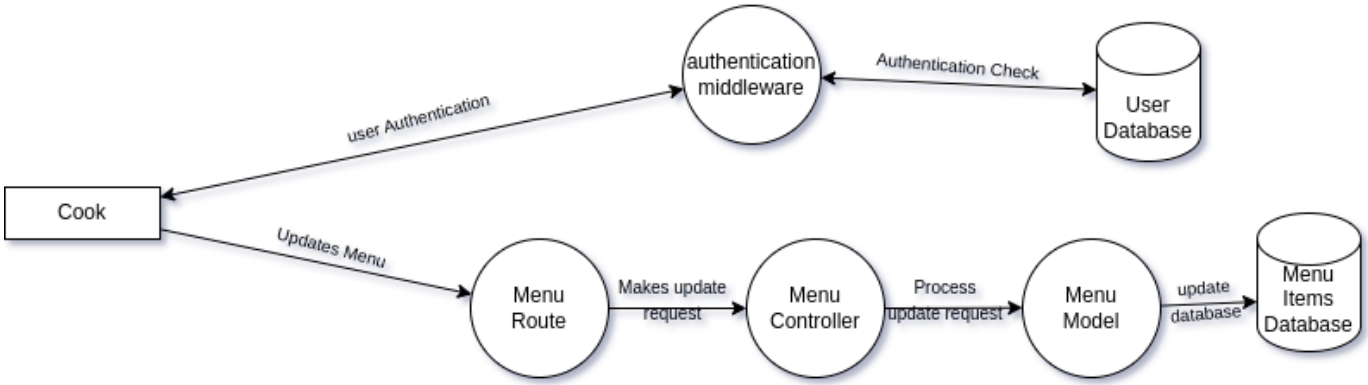
To provide a visual representation of data flow within the "Canteen Management Portal," I develop detailed Data Flow Diagrams (DFDs). These diagrams will illustrate the interaction between various components, showcasing the flow of data from user interactions to backend processes and vice versa.



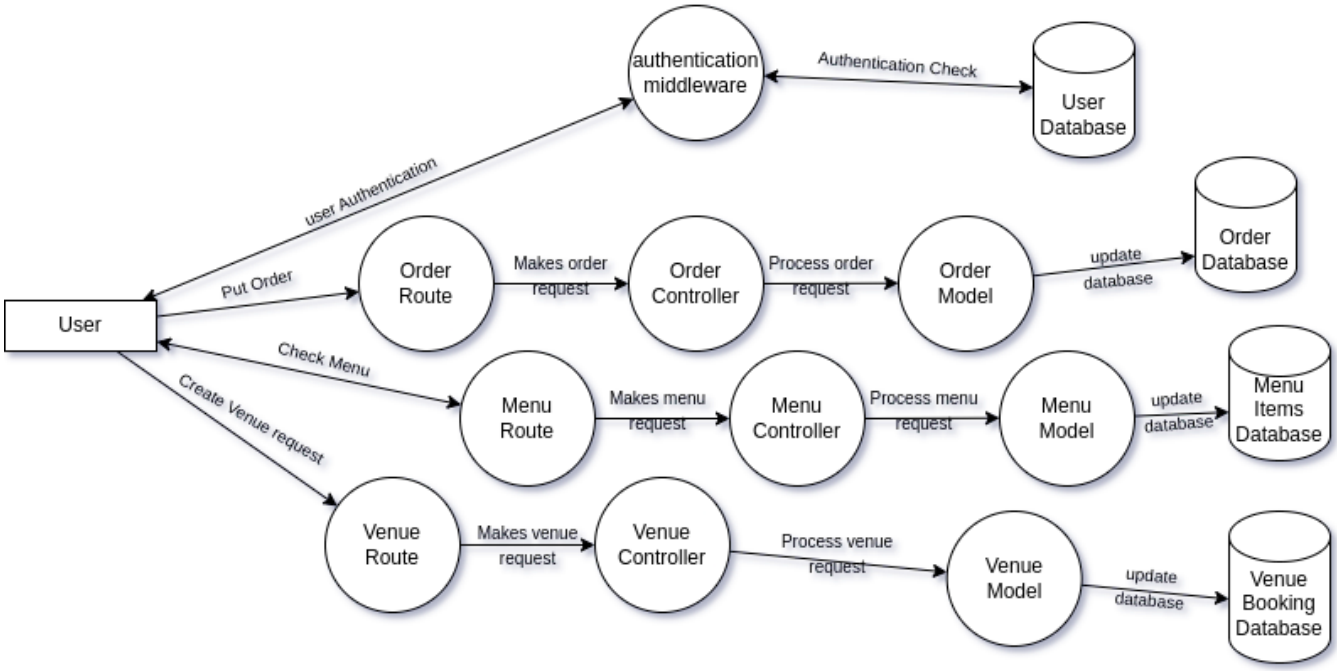
0 - Level DFD



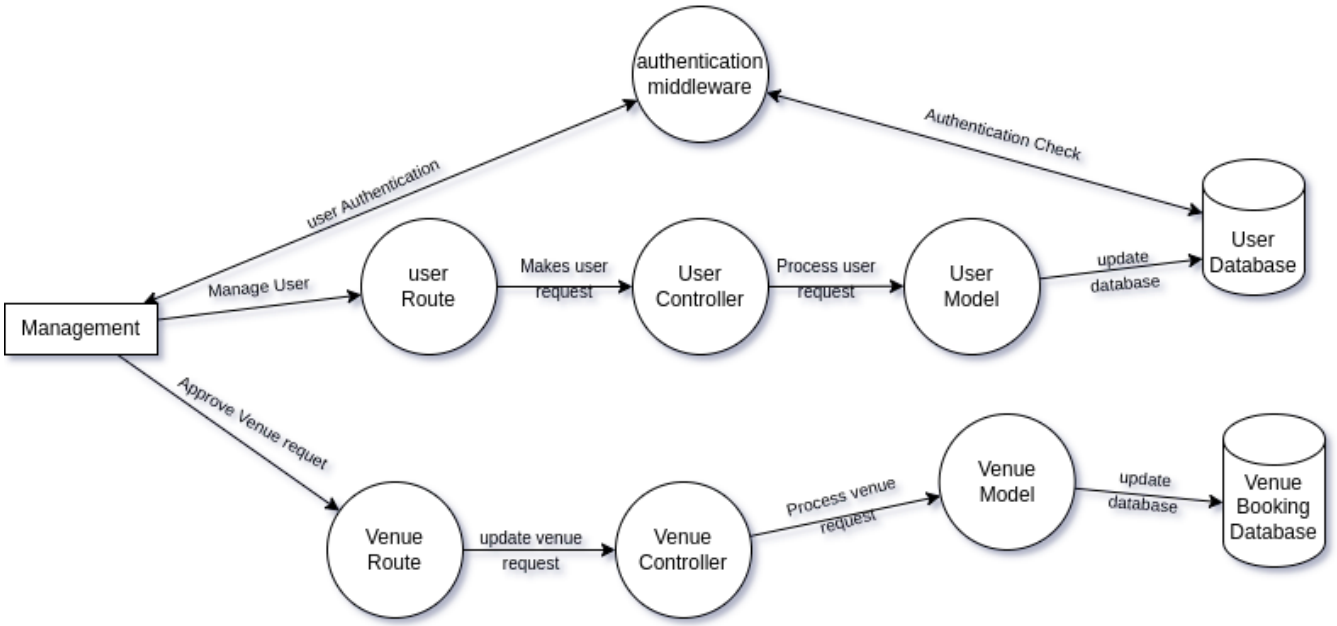
1 - Level DFD



2 - Level DFD(Cook)



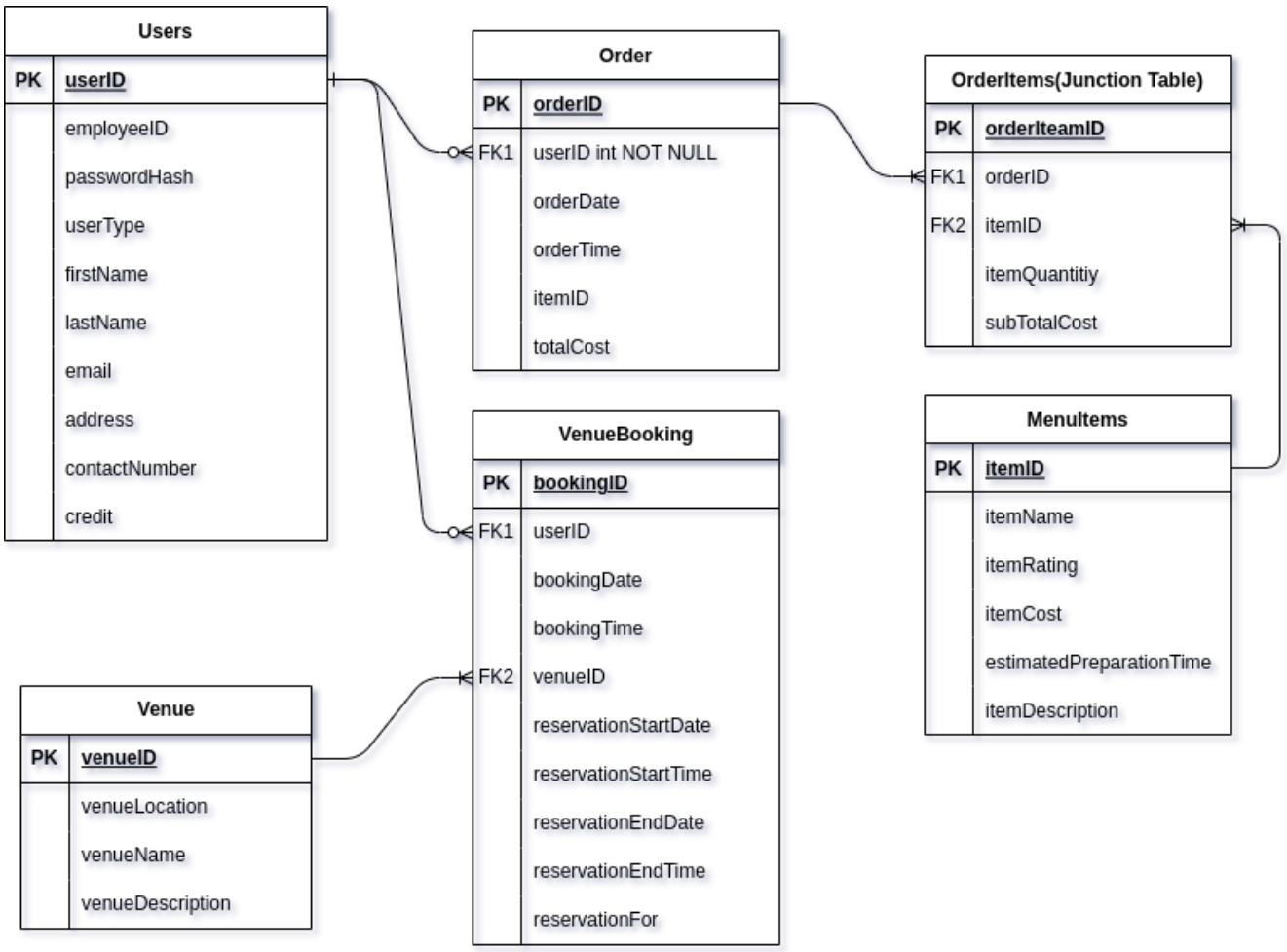
2 - Level DFD(User)



2 - Level DFD(Management)

ER Diagrams/Database Design

For efficient data management, we'll employ a relational database, specifically MySQL. ER (Entity-Relationship) diagrams will be crafted to depict the entities, their relationships, and attributes within the database. This step is crucial for establishing a robust foundation for data storage and retrieval.



ER - Diagram

Entity-Relationship (ER) Diagram Description:

Users:

Attributes:

- **userID** (Primary Key): Unique identifier for each user.
- **employeeID**: Identification number for the employee.
- **passwordHash**: Hashed password for security.
- **userType**: Type of user (e.g., regular employee, admin).
- **firstName**, **lastName**: User's first and last name.
- **email**: User's email address.
- **address**: User's residential address.
- **contactNumber**: User's contact number.
- **credit**: Credit balance for canteen transactions.

Order:

Attributes:

- **orderID** (Primary Key): Unique identifier for each order.
- **userID** (Foreign Key): Links to the Users table to identify the ordering user.
- **orderDate**, **orderTime**: Date and time of the order.
- **itemID**: Identifier for the ordered item.
- **totalCost**: Total cost of the order.

OrderItems (Junction Table):

Attributes:

- **orderitemID** (Primary Key): Unique identifier for each order item.
- **orderID** (Foreign Key): Links to the Order table.
- **itemID** (Foreign Key): Links to the MenuItems table.
- **itemQuantity**: Quantity of the ordered item.
- **subTotalCost**: Subtotal cost for the specific item in the order.

MenuItems:

Attributes:

- **itemID** (Primary Key): Unique identifier for each menu item.
- **itemName**: Name of the menu item.
- **itemRating**: Rating assigned to the item by users.
- **itemCost**: Cost of the menu item.
- **estimatedPreparationTime**: Time required for item preparation.
- **itemDescription**: Description of the menu item.

VenueBooking:

Attributes:

- **bookingID** (Primary Key): Unique identifier for each venue booking.
- **userID** (Foreign Key): Links to the Users table to identify the booking user.
- **bookingDate**, **bookingTime**: Date and time of the booking.
- **venueID** (Foreign Key): Identifier for the booked venue.
- **reservationStartDate**, **reservationStartTime**, **reservationEndDate**, **reservationEndTime**: Start and end date/time of the venue reservation.
- **reservationFor**: Purpose of the venue reservation.

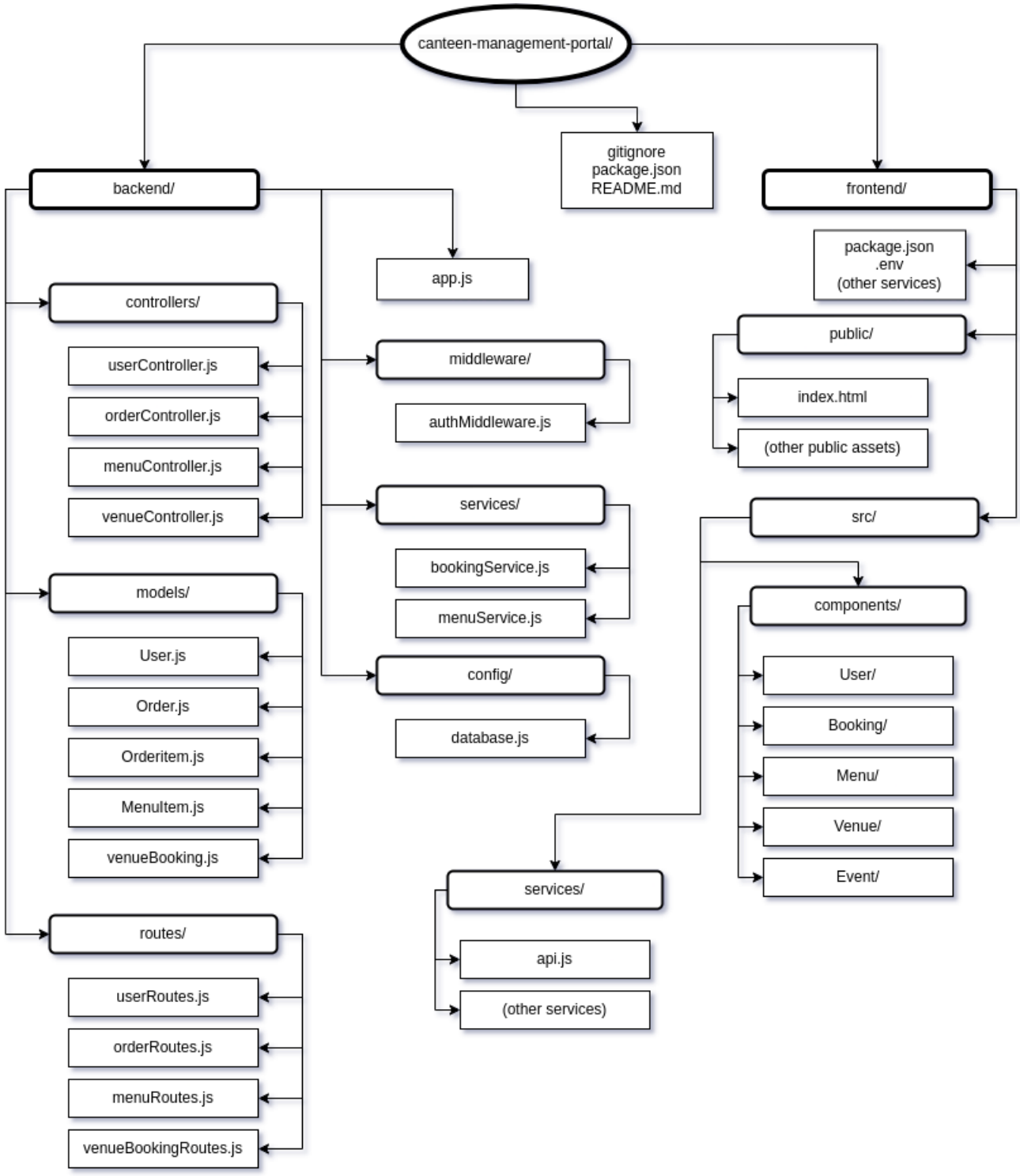
System Architecture

The overall system architecture will be designed with a modular approach, considering various components such as frontend, backend, and the database. This modular design ensures flexibility, ease of maintenance, and scalability as the "Canteen Management Portal" evolves.

Prototyping

To better visualize the user interface and gather early feedback, we'll create prototypes of key sections within the portal. Prototypes aid in refining the user experience, allowing us to make informed decisions on layout, navigation, and overall design before full implementation.

6.0 Project Structure and Logic:



Backend Structure:

Controllers:

userController.js:

Logic:

- Manages user-related operations, including registration, login, and profile management.
- Utilizes services to interact with the database for user-related functionalities.

orderController.js:

Logic:

- Handles order-related operations such as placing orders and calculating costs.
- Interacts with the Order and OrderItem models for database operations.

menuController.js:

Logic:

- Manages menu-related operations, including fetching menu items and handling ratings.
- Interacts with the MenuItem model for database operations.

venueBookingController.js:

Logic:

- Handles venue booking operations.
- Interacts with the VenueBooking model for database operations.

Models:

User.js:

- **Attributes:** Represents the User model with attributes from the ER diagram.
- **Relationships:** Handles relationships with Order and VenueBooking models.

Order.js:

- **Attributes:** Represents the Order model with attributes and relationships as per the ER diagram.
- **Relationships:** Connects with the User and OrderItem models.

OrderItem.js:

Attributes:

- Represents the OrderItem model for the many-to-many relationship between Order and MenuItem.
- Connects Order and MenuItem models.

MenuItem.js:

- **Attributes:** Represents the MenuItem model with attributes from the ER diagram.
- **Relationships:** Connects with the OrderItem model for the many-to-many relationship.

VenueBooking.js:

- **Attributes:** Represents the VenueBooking model with attributes and relationships.
- **Relationships:** Connects with the User model.

Routes:

userRoutes.js:

- **Endpoints:** Manages user-related routes for registration, login, and profile functionalities.

orderRoutes.js:

- **Endpoints:** Handles routes for placing orders and viewing order history.

menuRoutes.js:

- **Endpoints:** Manages routes for fetching menu items and submitting ratings.

venueBookingRoutes.js:

- **Endpoints:** Handles routes for venue bookings.

Config:

database.js:

- **Functionality:** Configures the database connection, typically using Sequelize or another ORM.

Middleware:

authMiddleware.js:

- **Functionality:** Implements authentication middleware to secure routes requiring user authentication.

Frontend Structure:

Components:

UserRegistration.js:

Logic:

- Component for user registration.
- Utilizes userService.js for making API calls related to user registration.

UserLogin.js:

Logic:

- Component for user login.
- Utilizes userService.js for making API calls related to user login.

UserProfile.js:

Logic:

- Component for managing user profiles.
- Utilizes userService.js for making API calls related to user profile management.

OrderForm.js:

Logic:

- Component for placing orders.
- Utilizes orderService.js for making API calls related to order placement.

OrderHistory.js:

Logic:

- Component for viewing order history.
- Utilizes orderService.js for making API calls related to order history.

MenuItemsList.js:

Logic:

- Component for displaying menu items.
- Utilizes menuService.js for making API calls related to fetching menu items.

VenueBookingForm.js:

Logic:

- Component for venue booking.
- Utilizes venueBookingService.js for making API calls related to venue booking.

Services:

userService.js:

- **Functionality:** Service for handling user-related API calls.

orderService.js:

- **Functionality:** Service for order-related API calls.

menuService.js:

- **Functionality:** Service for menu-related API calls.

venueBookingService.js:

- **Functionality:** Service for venue booking API calls.

Main Entry Points:

App.js:

- **Functionality:** Orchestrates the overall structure and routing of the React application.

index.js:

- **Functionality:** Entry point for the React application.

6.1 Testing Overview:

Unit Testing:

- **Objective:** Ensure individual components (controllers, models, services) function as intended.
- **Process:** Use Jest for backend unit tests and Jest/Enzyme for frontend tests.

Integration Testing:

- **Objective:** Validate the seamless interaction between integrated components.
- **Process:** Employ Supertest for backend API testing and ensure proper data flow.

End-to-End (E2E) Testing:

- **Objective:** Verify complete workflow from user interaction to system response.
- **Process:** Leverage Cypress or Selenium for comprehensive E2E testing scenarios.

User Acceptance Testing (UAT):

- **Objective:** Involve end-users to validate the system meets their requirements.
- **Process:** Create test scenarios based on user stories and gather feedback.

Performance Testing:

- **Objective:** Assess system responsiveness and stability under varied conditions.
- **Process:** Use JMeter or Gatling to measure response times and resource utilization.

Security Testing:

- **Objective:** Identify and address potential security vulnerabilities.
- **Process:** Perform penetration testing and validate encryption mechanisms.

Usability Testing:

- **Objective:** Evaluate user-friendliness and accessibility of the application.
- **Process:** Collect feedback on design and navigation.

Automated Testing:

- **Objective:** Improve testing efficiency with automated scripts.
- **Process:** Develop automated scripts for repetitive test cases.

Documentation:

- **Objective:** Maintain comprehensive test plans and results.
- **Process:** Document test cases, results, and any identified issues.

6.2 Reports Generation:

1. **Executive Summary:**
   - Brief overview of the project.
   - High-level achievements and goals.
2. **Introduction:**
   - Background and context of the project.
   - Objectives and purpose of the system.
3. **System Architecture:**
   - Overview of the backend and frontend structures.
   - Explanation of controllers, models, routes, and services.
4. **Database Design:**
   - Description of the database schema.
   - Relationships between different entities.
5. **User Guide:**
   - Instructions for user registration and login.
   - Guide on using different features (ordering, venue booking, etc.).
   - User profile management details.
6. **Feature Implementation:**
   - Detailed explanation of implemented features.
   - User authentication and authorization mechanisms.
   - Menu management and order placement.
   - Venue booking system.
7. **Testing Methodologies and Results:**
   - Overview of testing methodologies employed.
   - Summary of test cases, including unit, integration, and acceptance tests.
   - Test results, including any issues identified and resolutions.
8. **Challenges and Solutions:**
   - Identification of challenges faced during development.
   - Strategies and solutions employed to overcome challenges.
9. **Future Enhancements:**
   - Proposed features for future development.
   - Areas of improvement based on user feedback and testing results.
10. **Conclusion:**
    - Summary of project outcomes.
    - Lessons learned during the development process.
11. **Acknowledgments:**
    - Recognition of contributors and stakeholders.

- Appreciation for guidance and support.

12. References:

- Citations for tools, frameworks, and technologies used.

13. Appendices:

- Additional supporting documentation.
- Code snippets or excerpts for key functionalities.
- Additional resources or materials.

7.0 Tools / Platform, Hardware, and Software Requirement Specifications:

Development Tools:

- **Editors:** Visual Studio Code
- **Version Control:** Git, GitHub
- **Backend:** Node.js, Express.js
- **Frontend:** React.js
- **Database:** PostgreSQL, Sequelize
- **Testing:** Jest, Supertest
- **Containerization:** Docker
- **CI/CD:** Jenkins, GitHub Actions
- **Platform:**
- **Hosting:** AWS, Microsoft Azure
- **PaaS:** Heroku

Database:

- **DBMS:** PostgreSQL
- **ORM:** Sequelize

Backend Server:

- **Runtime:** Node.js
- **Web Framework:** Express.js
- **Frontend Application:**
- **Framework:** React.js
- **State Management:** Redux
- **Styling:** Styled Components/Emotion
- **UI Components:**

- **Library:** Material-UI, Ant Design
- **Database Hosting (Production):**
- **Cloud Service:** Amazon RDS, Azure Database

Security:

- **Encryption:** HTTPS/SSL
- **Security Tools:** Helmet.js

Communication: Slack, Microsoft Teams

- **Version Control Hosting:**
- **Repository:** GitHub

Hardware and Software Requirements:

- **Hardware:** Standard development machines
- **Software:** Browsers (Chrome, Firefox, Safari), Node.js, PostgreSQL (local development)

8.0 Industry/Client Information:

Client Name: Tata Steel Long Products (TSLP)
Formerly Known As: Tata Sponge Iron Limited
Industry: Manufacturing high alloy steel, primarily for the auto sector and wire rope industry.
Capacity: One million tonne capacity, one of the largest specialty steel plants in India in the long product segment.
CIN (Corporate Identity Number): L27102OR1982PLC001091
Registered Office & DRI Plants: Joda, Odisha, P.O. Joda, Dist. Keonjhar, Orissa - 758034, India.

Contact Person: Apurva Rani
Designation: Manager (Electronics and Instrumentation Department)

The project is being developed for Tata Steel Long Products (TSLP) under the guidance of Apurva Rani, who holds the position of Manager in the Electronics and Instrumentation Department. The aim of the project is to create an in-house canteen management system for the company's township, improving the efficiency of managing food services and venue bookings.

9.0 Future Scope and Enhancement:

- **Mobile App Development:**
  Scope: Create Android and iOS apps for seamless mobile access.
- **Enhanced Analytics:**
  Scope: Expand insights with advanced analytics and visualization tools.
- **Integration with Employee Database:**
  Scope: Link directly to employee database for secure user management.
- **Automated Inventory Management:**
  Scope: Implement real-time tracking and automated stock alerts.
- **Enhanced Venue Booking:**
  Scope: Calendar view, detailed event planning options.
- **Multi-Language Support:**
  Scope: Implement language preferences for a diverse workforce.
- **Integration with Payment Systems:**
  Scope: Direct linkage to company payment systems for seamless transactions.
- **Enhanced Feedback System:**
  Scope: Robust user feedback with sentiment analysis.
- **Advanced Security Measures:**
  Scope: Two-factor authentication, regular security audits.
- **Customization of Dietary Preferences:**
  Scope: User-defined dietary preferences and smart recommendations.
- **Integration with Wellness Programs:**
  Scope: Collaborate with wellness programs for health-centric features.
- **Offline Mode and PWA Optimization:**
  Scope: Improved PWA features for optimal offline functionality.
- **Real-Time Notifications:**
  Scope: Implement instant notifications for menu changes and bookings.
- **Scalability and Cloud Integration:**
  Scope: Optimize for scalability and explore cloud-based solutions.
- **Enhanced UI/UX:**
  Scope: Regular updates for modern design and usability.