INSTITUTE FOR ADVANCED COMPUTING AND SOFTWARE DEVELOPMENT, AKURDI, PUNE

"MealStack-Canteen Management System"

PG-DAC August 2025

Submitted By:

Group No: 62

Roll No.	**Name of Student**
258100	Vaibhav Pophale
258218	Vrushabh Bhandalkar

Mr. Vaibhav Verulkar
Project Guide

Mr. Anil Sharma
Centre Coordinator

ABSTRACT

MealStack is a full-stack, web-based meal and food ordering management system developed to automate and optimize daily food service operations in environments such as institutional canteens and cafeteria systems. The application addresses key operational challenges including meal inventory tracking, daily menu management, order processing, and secure user access through a centralized digital platform. The system is designed to reduce manual intervention, ensure data consistency, and improve overall operational efficiency.

The frontend of the application is implemented using React, enabling a component-based architecture and dynamic user interface for efficient interaction with system functionalities. The backend is developed using Spring Boot, which exposes RESTful APIs to handle business logic, data processing, and communication between system components. Authentication and authorization are implemented using JWT (JSON Web Token) based security integrated with Spring Security, providing stateless, role-based access control for administrators and users.

MealStack utilizes MySQL as the relational database management system to store and manage structured data related to users, meals, daily menu items, and orders. Object-relational mapping is achieved using JPA and Hibernate, ensuring efficient data persistence and retrieval. The application follows a layered architecture consisting of controller, service, and repository layers, promoting modularity, scalability, and maintainability.

The system supports core functionalities such as CRUD operations on menu items, daily inventory handling, order placement, and administrative management through secure API endpoints. By integrating frontend and backend components through RESTful communication and enforcing security at multiple levels, MealStack demonstrates a scalable, secure, and robust solution for real-world food service management using modern full-stack development technologies.

ACKNOWLEDGEMENT

I take this occasion to thank God, almighty for blessing us with his grace and taking our endeavor to a successful culmination. I extend my heartfelt thanks to our esteemed guide, **Mr. Vaibhav Verulkar** for providing me with the right guidance and advice at the crucial juncture and showing me the right way. I sincerely thank our respected Centre Co- Ordinator, **Mr. Anil Sharma**, for allowing us to use the available facilities. I would also like to thank the other faculty members at this occasion. Last but not least, I would like to thank my friends and family for the support and encouragement they have given me during our work.

Vaibhav Sunil Pophale :250841220199

Vrushabh Bhandalkar:250841220214

Table of Contents

Sr.No	Description	Page No.
1	Introduction	1
2	SRS	6
3	Diagrams	11
3.1	ER Diagram	11
3.2	Use Case Diagram	12
3.3	Data Flow Diagram	13
3.4	Activity Diagram	14
3.5	Class Diagram	18
3.6	Sequence Diagram	19
4	Database Design	23
5	Snapshots	27
6	Conclusion	34
7	References	35

1.INTRODUCTION

MealStack is a comprehensive, web-based meal and food service management system designed to automate and streamline daily operations in institutional canteens, corporate cafeterias, and similar food service environments. The system integrates meal inventory management, daily menu handling, order processing, and user management into a unified platform, enabling efficient coordination between administrators and end users. By digitizing traditionally manual processes, MealStack reduces operational overhead, minimizes errors, and improves overall service efficiency.

The application is developed using React for the frontend, providing a responsive, component-based user interface that enhances usability and interaction. The backend is implemented using Spring Boot, which offers a robust and scalable RESTful API architecture to handle business logic, data processing, and system workflows. Secure authentication and authorization are enforced using JWT (JSON Web Token) integrated with Spring Security, ensuring stateless and role-based access control across the system.

MealStack is designed to support real-time meal availability tracking, daily menu resets, automated order handling, and secure user interactions. The system caters to the operational requirements of food service administrators while delivering a seamless ordering experience to users. By combining modern full-stack technologies with a layered architectural approach, MealStack provides a scalable and maintainable solution for efficient food service management.

MealStack is specifically tailored to meet the unique needs of educational canteen environments, ensuring real-time menu updates, automated stock management, wallet-based transactions, and a user-friendly interface for both students and administrators. This integration of inventory management with a digital ordering system helps canteens reduce manual errors, eliminate cash handling, improve operational efficiency, and provide superior customer service to students.

In the MealStack system, both Student and Admin roles play crucial roles in the management and operation of the platform. Each role comes with specific responsibilities, permissions, and access levels to ensure that the system operates smoothly and securely.

System Roles:

Student Role: Students can register, login, browse daily menus, place orders using wallet balance, track order status, view order history, recharge wallet, and manage profile.

Admin Role: Administrators manage student accounts, maintain item catalog, set daily menus with quantities, process orders, mark orders as served, and view reports and analytics.

Key Features:

• Real-time menu browsing with availability

• Wallet-based cashless payments

• Automated inventory tracking

• Order processing workflow

• Comprehensive reporting

• Mobile-responsive design

Security: JWT-based authentication, BCrypt password hashing, role-based access control, SQL injection prevention, XSS protection, and CORS configuration.

1.1 PURPOSE

MealStack digitizes canteen operations to:

For Students:

• Enable cashless, convenient ordering

• Reduce waiting times

• Provide transparent pricing and availability

• Maintain digital transaction records

For Administrators:

• Streamline order processing

• Eliminate cash handling

- Provide real-time inventory visibility
- Enable data-driven decisions
- Reduce food wastage

Overall Goals:

- Complete digitization of canteen operations
- Paperless, eco-friendly system
- Improved efficiency and customer satisfaction
- Scalable for growing populations

1.2 Scope

Functional Modules:

1. User Management: Registration, authentication, profile management
2. Menu Management: Item catalog, categorization, pricing
3. Inventory Management: Daily stock, real-time tracking, alerts
4. Order Management: Cart, placement, tracking, history
5. Wallet & Payment: Balance management, recharge, transactions
6. Reporting: Sales, inventory, and analytics

Technical Stack:

- Frontend: React.js, Redux, Material-UI, Axios
- Backend: Spring Boot, Spring Security, JPA/Hibernate, MySQL
- Security: JWT, BCrypt, CORS
- Integration: Payment gateway, Email service

Users: Students and Canteen Administrators

Deployment: Single institution, cloud or on-premise

Out of Scope: Physical equipment, payroll, supplier management

1.3 Objectives:

1. Efficient Inventory Management: Real-time tracking, automated stock updates, low-stock alerts, wastage minimization

2. Streamlined Order Processing: Instant processing, elimination of manual errors, automated calculations

3. Cashless Transactions: Digital wallet system, payment gateway integration, complete transaction history

4. Enhanced User Experience: Intuitive interfaces, quick browsing, mobile-responsive design

5. Robust Security: JWT authentication, role-based access, data encryption, vulnerability prevention

6. Comprehensive Analytics: Sales reports, inventory analysis, student spending patterns, business insights

7. Operational Efficiency: Reduced manual effort, automated tasks, optimized workflows

1.4 KEY FUNCTIONALITIES

User Management:

• Student registration with email validation

• JWT-based login and authentication

• Profile management and password change

• Role-based access control (Student/Admin)

Menu & Inventory (Admin):

• Item master catalog with categories (Breakfast, Lunch, Snacks, Dinner, Beverages)

• Cuisine types (South Indian, North Indian, Oriental, Maharashtrian)

• Daily menu setup with quantities

• Real-time stock tracking and updates

• Low-stock alerts and manual adjustments

Order Management:

• Browse menu with real-time availability

• Shopping cart with quantity management

• Order placement with balance validation

• Unique transaction ID generation

• Order status tracking (Pending/Served)

• Complete order history

Wallet & Payment:

• View wallet balance

• Recharge through payment gateway

• Automatic payment deduction

• Transaction history with receipts

Admin Functions:

• Student account management

• Pending order processing

• Mark items/orders as served

• Sales and inventory reports

• Dashboard with key metrics

Additional Features:

• Responsive design for all devices

• Email notifications

• Data export (Excel/PDF)

• Audit logging

• Performance optimization

2. SOFTWARE REQUIREMENT SPECIFICATION

2.1 FUNCTIONAL REQUIREMENTS

1: User Management

• Student registration with name, email, password, mobile, DOB, course

• Email uniqueness validation and BCrypt password hashing

• JWT token generation on login (24-hour expiry)

• Profile view/update and password change

• Role-based access control (STUDENT/ADMIN)

2: Menu Item Management (Admin)

• Add items with name, price, category, genre, image (unique name constraint)

• Update item details and pricing

• Delete items (with validation for active references)

• Search, filter, and sort items

3: Daily Inventory (Admin)

• Set daily menu from master catalog with quantities

• Unique constraint on (date, item_id)

• Real-time stock tracking (Available = Initial - Sold)

• Automatic quantity reduction on orders

• Manual inventory adjustments with logging

• Low-stock alerts

4: Order Management

• Shopping cart with add/modify/remove items

• Order placement with balance and stock validation

• Atomic transaction: deduct balance, create order, update inventory

• Generate unique transaction ID

• View pending/completed orders

• Mark orders as served

• Order history with filtering

5: Wallet & Payment

• View wallet balance

• Recharge with min/max limits through payment gateway

• Instant balance update on payment

• Transaction history with receipts

• Automatic payment deduction for orders

6: **Student Management (Admin)**

• View all students with details

• Add/edit/delete student accounts

• Search and filter by course/balance

• View student statistics

7: Reporting (Admin)

• Daily sales reports (item-wise, category-wise)

• Inventory status and turnover reports

• Student spending analytics

• Dashboard with key metrics

8: Security

• JWT authentication with role validation

• BCrypt password hashing

• Input validation and sanitization

• Audit logging

2.2 Non-Functional Requirements

Performance:

• Response time < 2 seconds

• Support 100 concurrent users

• Process 50 orders/minute during peak

• Database queries < 100ms

Reliability:

• 99.5% uptime

• Daily automated backups

• Transaction rollback on failures

• Graceful error handling

Usability:

• Intuitive navigation (max 3 clicks)

• Mobile responsive design

• No training required for students

• Clear error messages

Security:

• Strong password policy (min 8 chars)

• HTTPS for all communications

• SQL injection and XSS prevention

• CSRF protection

• Audit trails

Maintainability:

• Modular MVC architecture

• 70%+ code coverage

• Comprehensive documentation

• Version control with Git

Scalability:

• Horizontal scaling support

• Database sharding capability

• Caching for frequent queries

• Load balancing ready

2.3 Hardware and Software Requirements

HARDWARE

Server (Minimum):

• Processor: Intel i5/AMD Ryzen 5 (Quad-core, 2.5 GHz)

• RAM: 8 GB

• Storage: 50 GB SSD

• Network: 100 Mbps

Client (Minimum):

• Processor: Dual-core 1.5 GHz

• RAM: 4 GB

• Display: 1024x768

• Network: 4G/10 Mbps

SOFTWARE

Backend:

• OS: Ubuntu 22.04/Windows Server 2019

• JDK: 17+

• Database: MySQL 8.0

• Server: Embedded Tomcat (Spring Boot)

• Build: Maven 3.8+

Backend Frameworks:

• Spring Boot 3.9.18

• Spring Security 6.x

• Spring Data JPA 3.x

• Hibernate 6.x

• JWT, Lombok, Swagger

Frontend:

• Browsers: Chrome 100+, Firefox 95+, Edge 100+, Safari 15+

• Node.js 18.x, npm 8.x

Frontend Frameworks:

• React 18.x

• Redux 4.x

• Material-UI 5.x

• React Router 6.x

• Axios 1.x

Development Tools:

• IDE: VS Code/IntelliJ/Eclipse/Spring Suite

• Version Control: Git

• API Testing: Postman/Swagger ui

• Database: MySQL Workbench

Additional:

• Payment Gateway: Razorpay/Stripe

• Email: SMTP (Gmail/SendGrid)

• SSL: Let's Encrypt

3.DIAGRAMS

3.1 Entity Relationship Diagram:

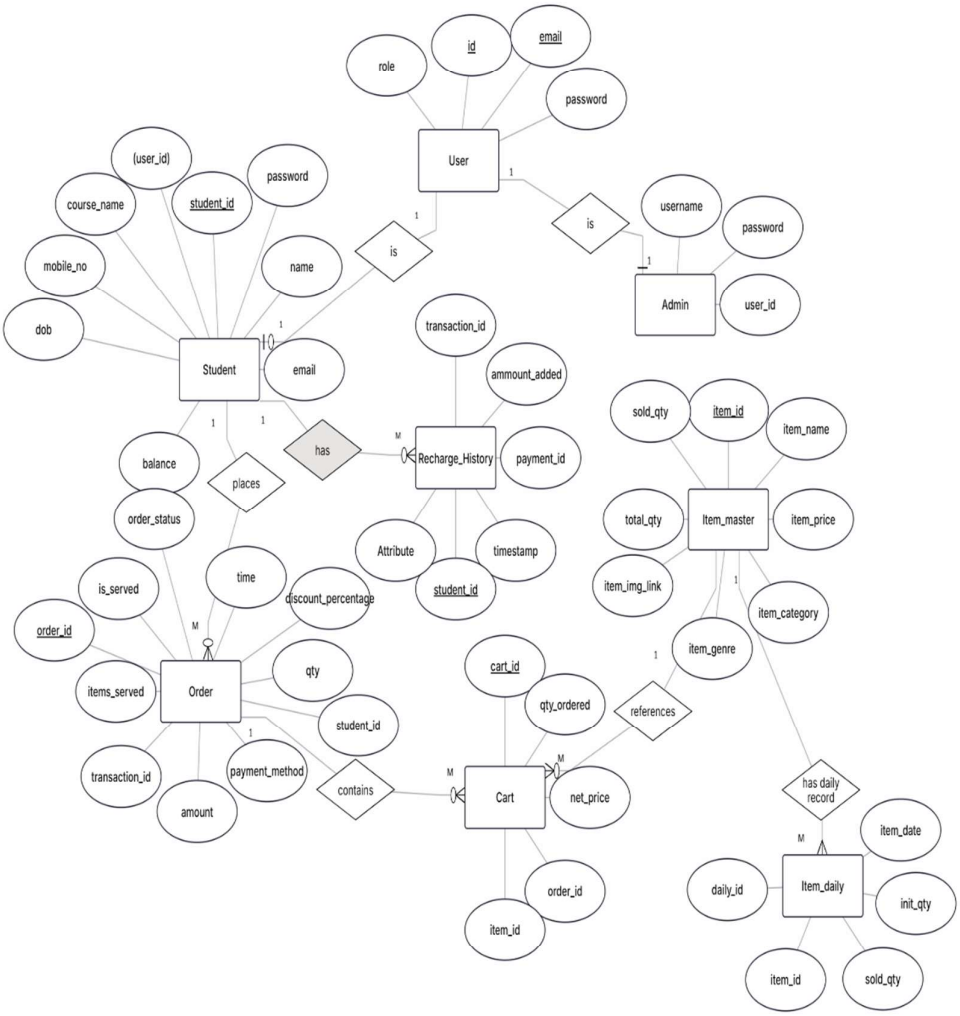


Figure 1 ER Diagram for MealStack-Canteen management System

3.2 Use Case Diagram:

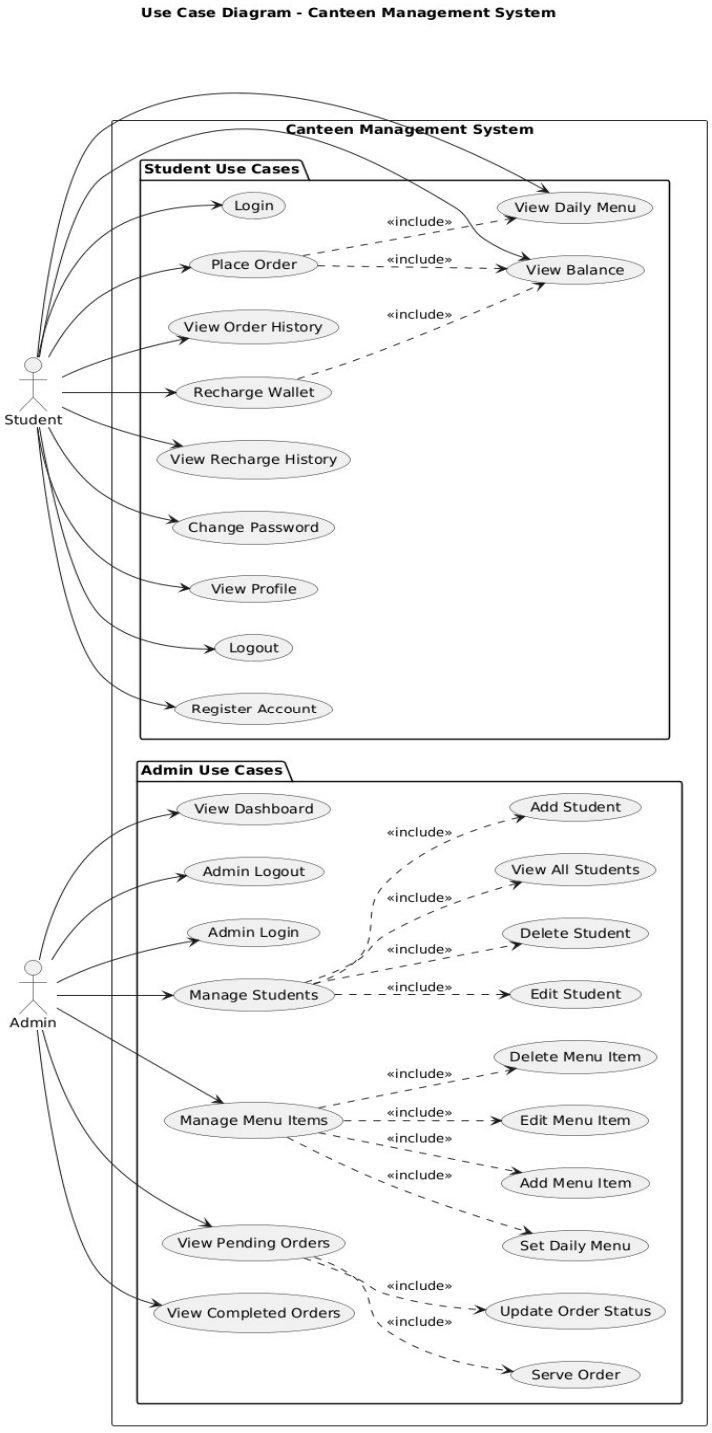


Figure 2 Use Case Diagram for MealStack-Canteen Management System

3.3 Data Flow Diagram:

DFD Level 0:

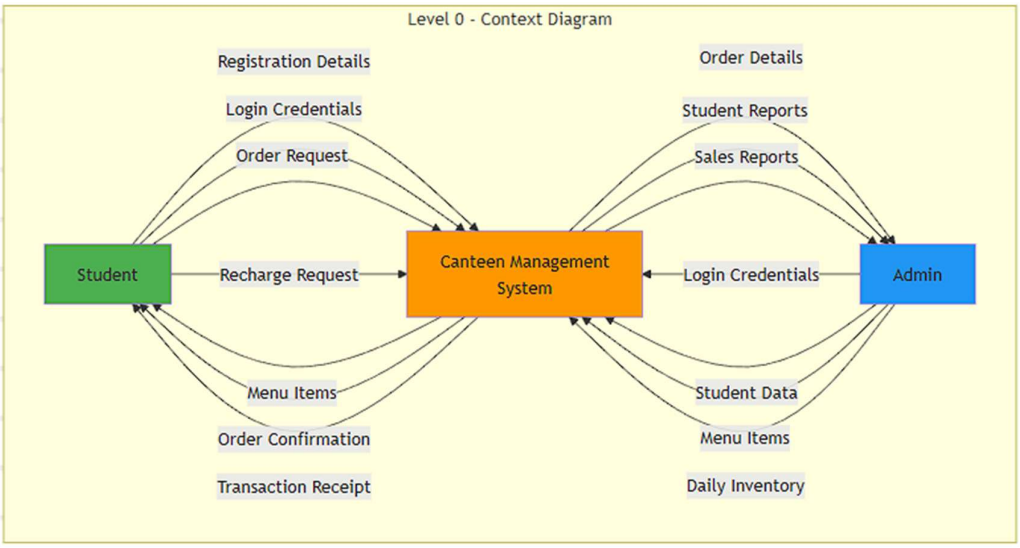


Figure 3 Data Flow 0

DFD Level 1:

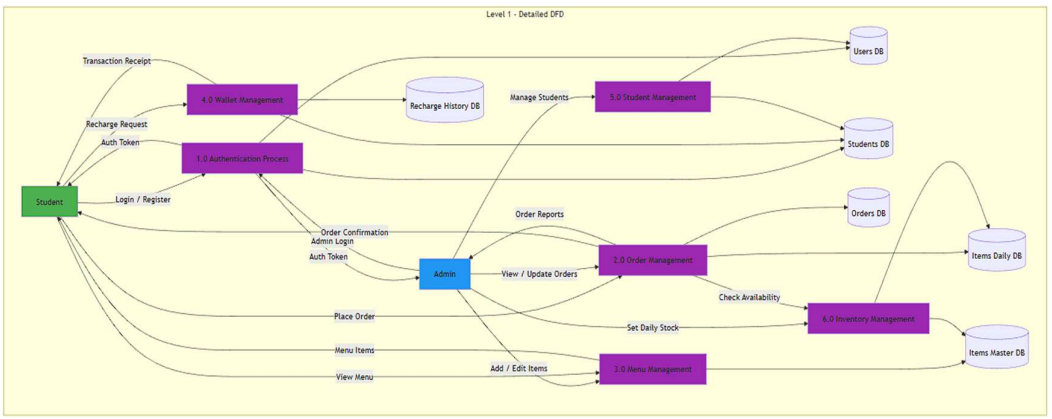


Figure 4 Data flow 2

3.4 Activity Diagram:

1.Login Activity Diagram

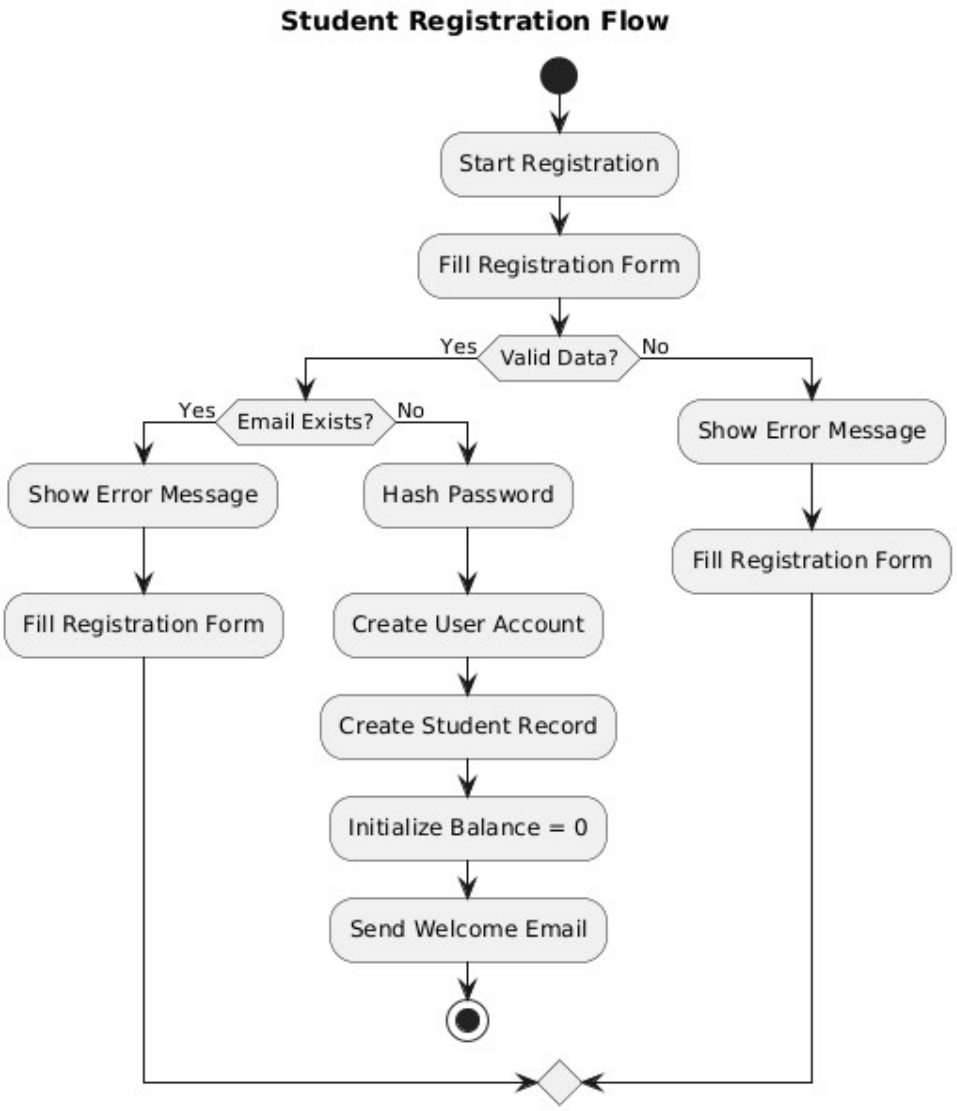


Figure 5 Login Activity

2.Admin Activity Diagram:

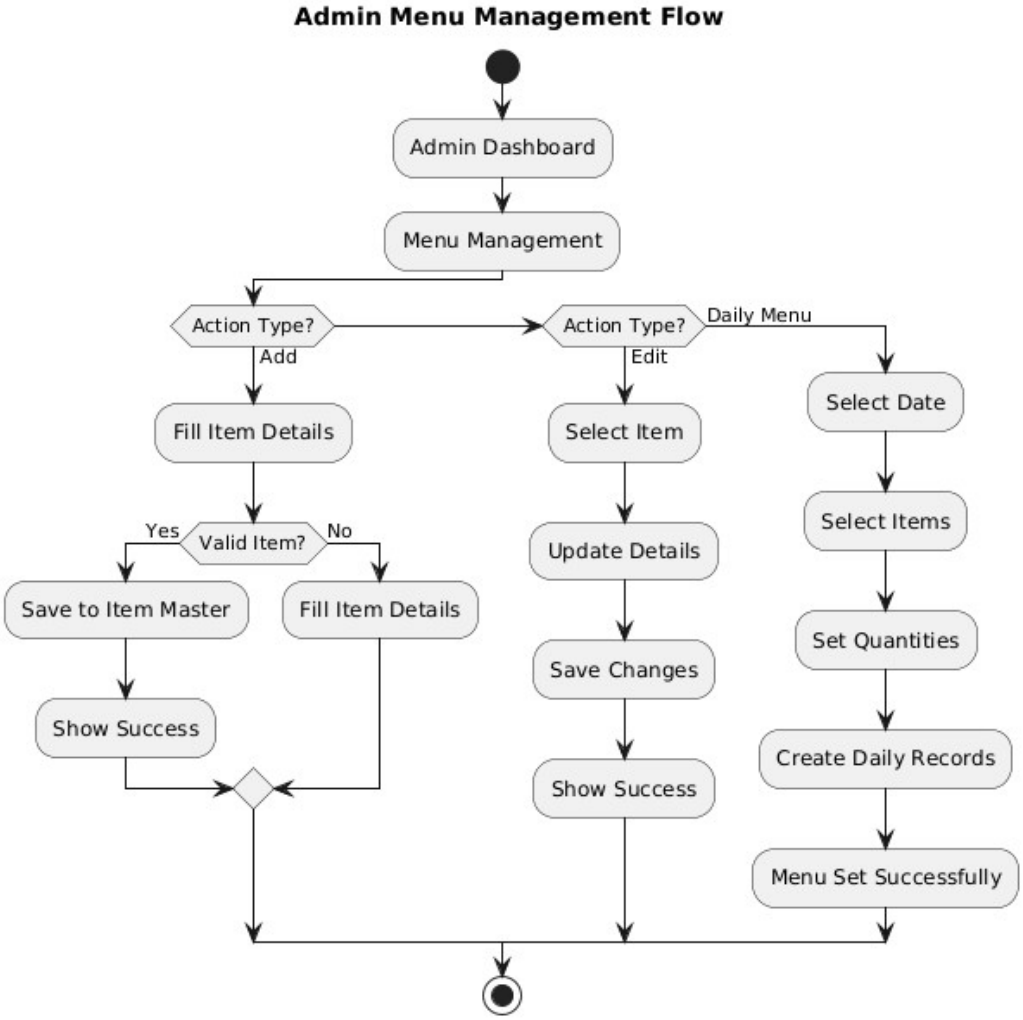


Figure 6 Admin Activity

3.Admin Order Processing Activity Diagram

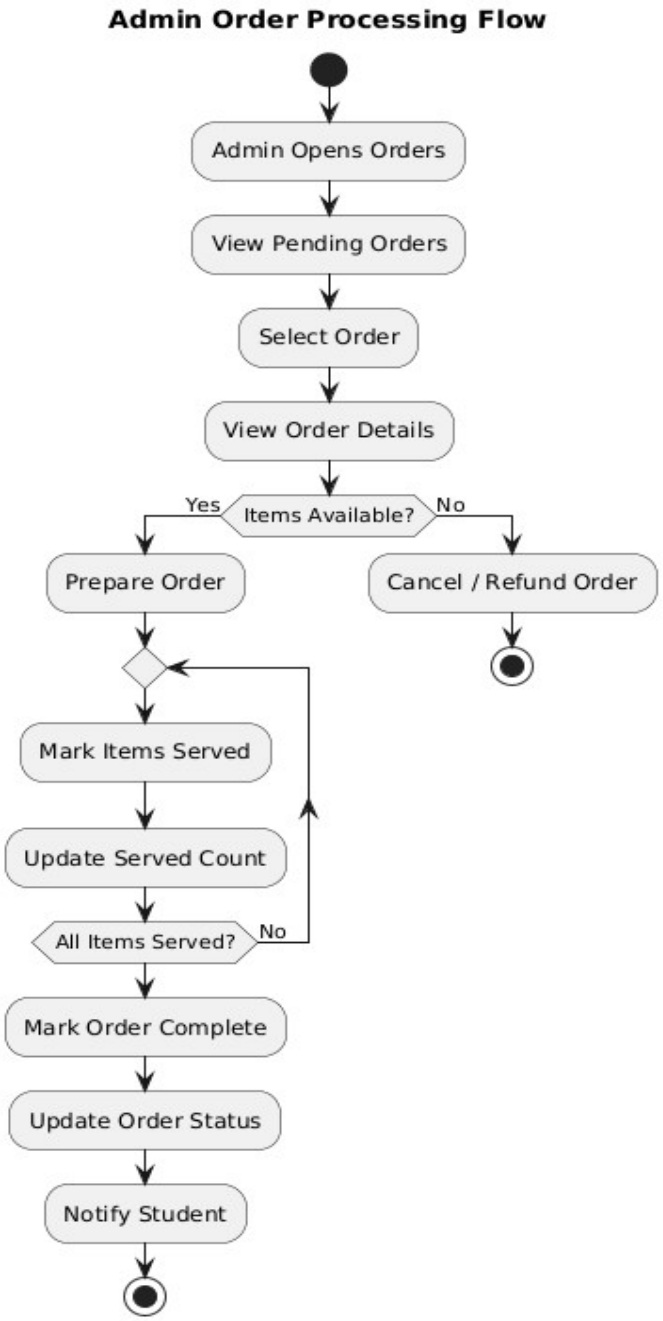


Figure 7 Admin Order

4.Student Order Activity Diagram

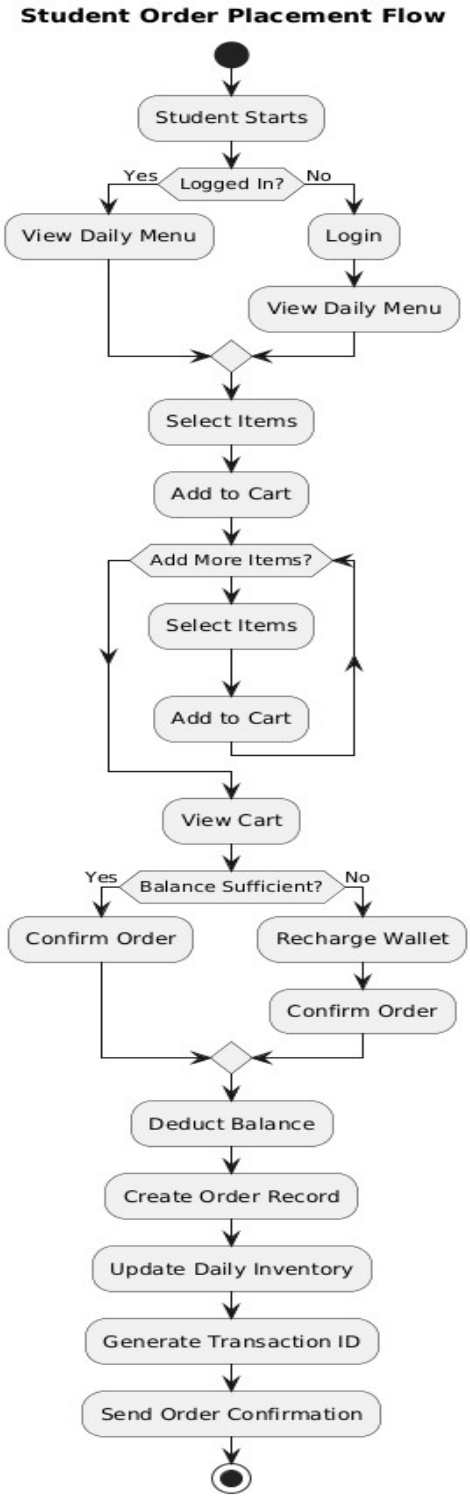
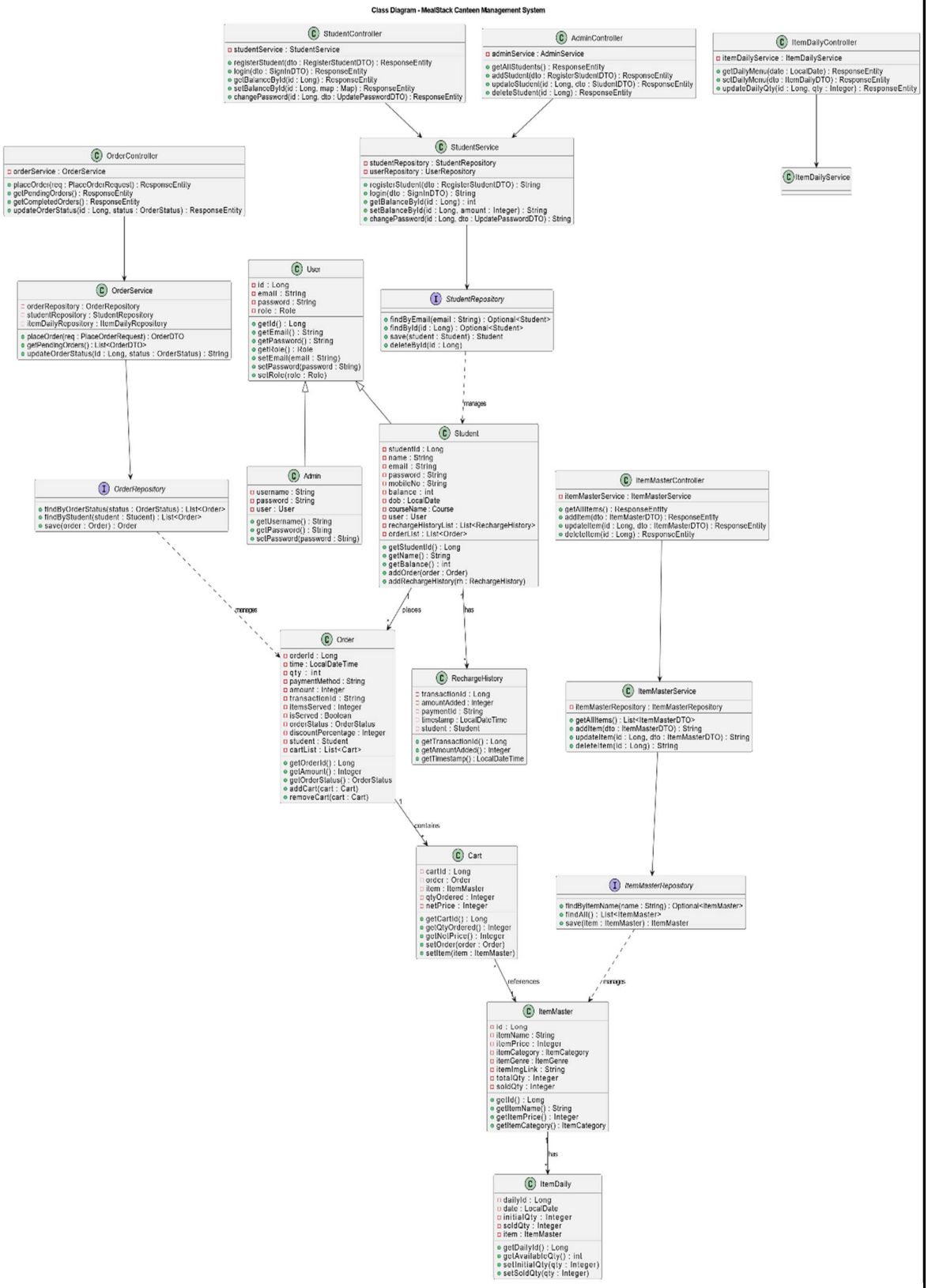


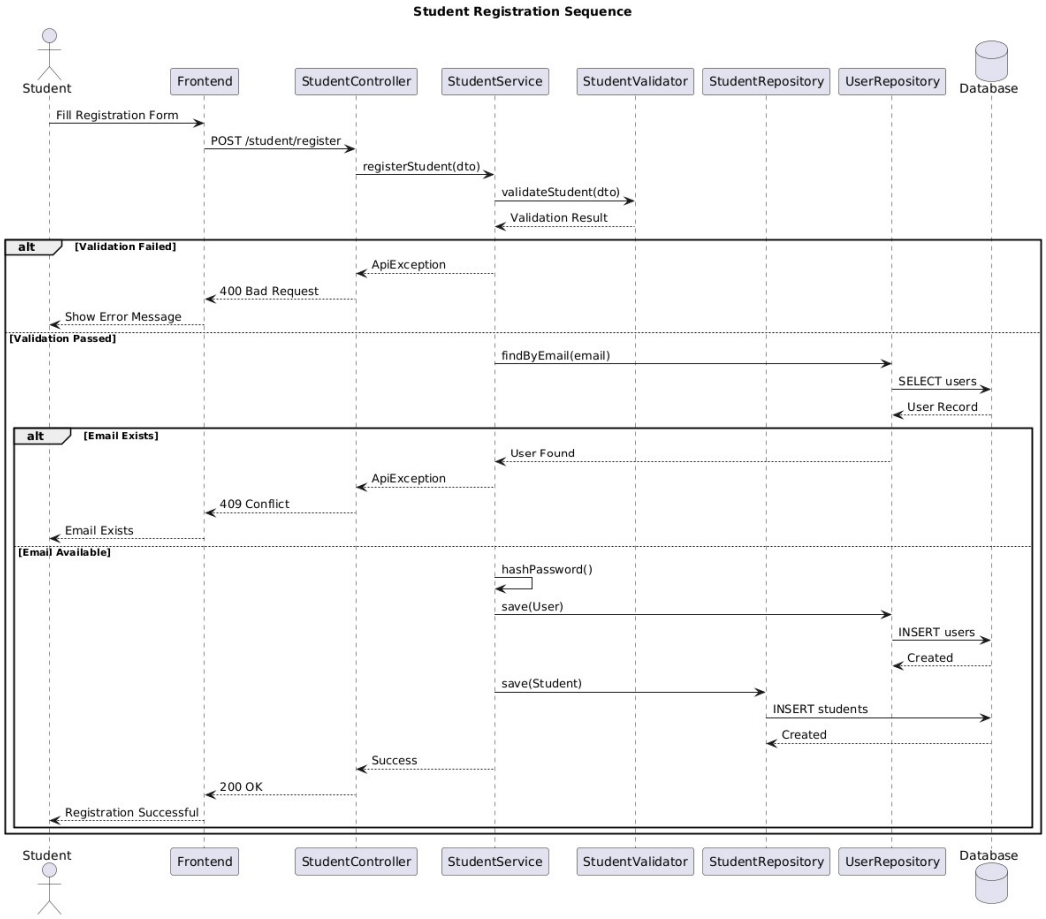
Figure 8 Student Order

3.5 Class Diagram:

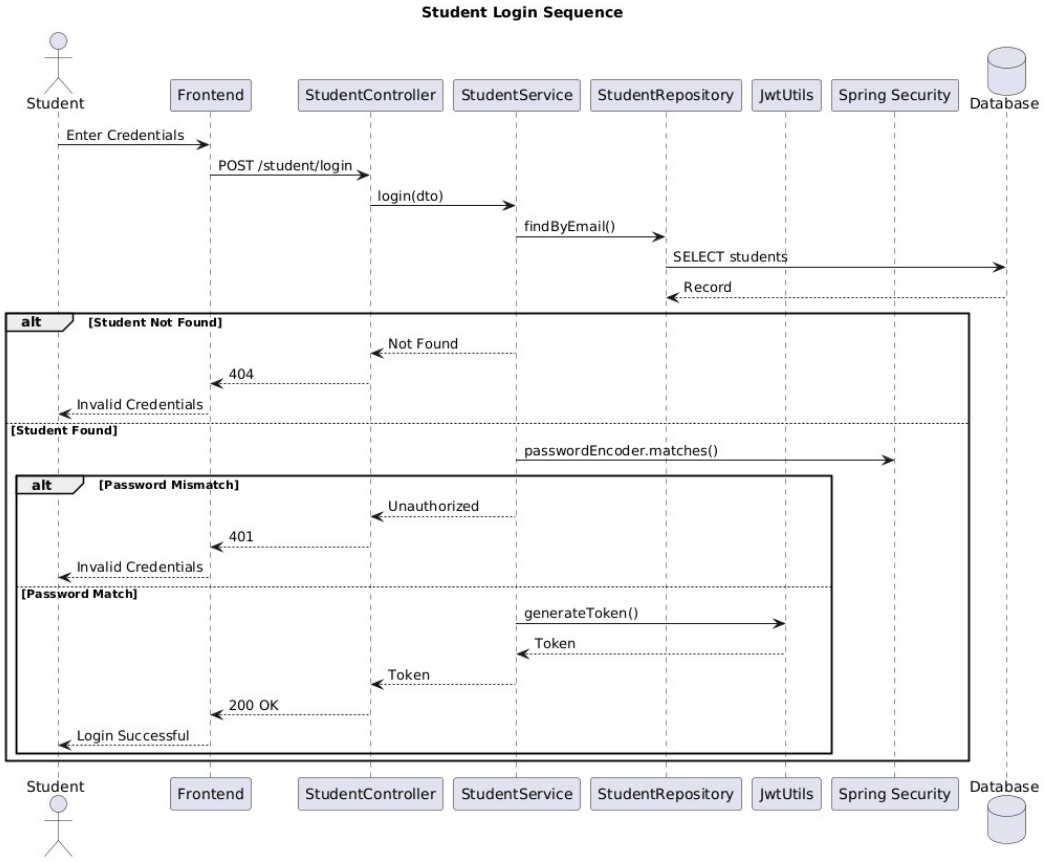

Class Diagram - MealStack Canteen Management System
StudentController
studentService : StudentService
registerStudent(dto : RegisterStudentDTO) : ResponseEntity
login(dto : SignInDTO) : ResponseEntity
getBalanceById(id : Long) : ResponseEntity
setBalanceById(id : Long, map : Map) : ResponseEntity
changePassword(id : Long, dto : UpdatePasswordDTO) : ResponseEntity
AdminController
adminService : AdminService
getAllStudents() : ResponseEntity
addStudent(dto : RegisterStudentDTO) : ResponseEntity
updateStudent(id : Long, dto : StudentDTO) : ResponseEntity
deleteStudent(id : Long) : ResponseEntity
ItemDailyController
itemDailyService : ItemDailyService
getDailyMenu(date : LocalDate) : ResponseEntity
setDailyMenu(dto : ItemDailyDTO) : ResponseEntity
updateDailyQty(id : Long, qty : Integer) : ResponseEntity
OrderController
orderService : OrderService
placeOrder(req : PlaceOrderRequest) : ResponseEntity
getPendingOrders() : ResponseEntity
getCompletedOrders() : ResponseEntity
updateOrderStatus(id : Long, status : OrderStatus) : ResponseEntity
StudentService
studentRepository : StudentRepository
userRepository : UserRepository
registerStudent(dto : RegisterStudentDTO) : String
login(dto : SignInDTO) : String
getBalanceById(id : Long) : int
setBalanceById(id : Long, amount : Integer) : String
changePassword(id : Long, dto : UpdatePasswordDTO) : String
ItemDailyService
OrderService
orderRepository : OrderRepository
studentRepository : StudentRepository
itemDailyRepository : ItemDailyRepository
placeOrder(req : PlaceOrderRequest) : OrderDTO
getPendingOrders() : List<OrderDTO>
updateOrderStatus(id : Long, status : OrderStatus) : String
User
id : Long
email : String
password : String
role : Role
getId() : Long
getEmail() : String
getPassword() : String
getRole() : Role
setEmail(email : String)
setPassword(password : String)
setRole(role : Role)
StudentRepository
findByEmail(email : String) : Optional<Student>
findById(id : Long) : Optional<Student>
save(student : Student) : Student
deleteById(id : Long)
manages
Student
studentId : Long
name : String
email : String
password : String
mobileNo : String
balance : int
dob : LocalDate
courseName : Course
user : User
rechargeHistoryList : List<RechargeHistory>
orderList : List<Order>
getStudentId() : Long
getName() : String
getBalance() : int
addOrder(order : Order)
addRechargeHistory(rh : RechargeHistory)
OrderRepository
findByOrderStatus(status : OrderStatus) : List<Order>
findByStudent(student : Student) : List<Order>
save(order : Order) : Order
Admin
username : String
password : String
user : User
getUsername() : String
getPassword() : String
setPassword(password : String)
ItemMasterController
itemMasterService : ItemMasterService
getAllItems() : ResponseEntity
addItem(dto : ItemMasterDTO) : ResponseEntity
updateItem(id : Long, dto : ItemMasterDTO) : ResponseEntity
deleteItem(id : Long) : ResponseEntity
manages
places
has
Order
orderId : Long
time : LocalDateTime
qty : int
paymentMethod : String
amount : Integer
transactionId : String
itemsServed : Integer
isServed : Boolean
orderStatus : OrderStatus
discountPercentage : Integer
student : Student
cartList : List<Cart>
getOrderId() : Long
getAmount() : Integer
getOrderStatus() : OrderStatus
addCart(cart : Cart)
removeCart(cart : Cart)
RechargeHistory
transactionId : Long
amountAdded : Integer
paymentId : String
timestamp : LocalDateTime
student : Student
getTransactionId() : Long
getAmountAdded() : Integer
getTimestamp() : LocalDateTime
ItemMasterService
itemMasterRepository : ItemMasterRepository
getAllItems() : List<ItemMasterDTO>
addItem(dto : ItemMasterDTO) : String
updateItem(id : Long, dto : ItemMasterDTO) : String
deleteItem(id : Long) : String
1
contains
Cart
cartId : Long
order : Order
item : ItemMaster
qtyOrdered : Integer
netPrice : Integer
getCartId() : Long
getQtyOrdered() : Integer
getNetPrice() : Integer
setOrder(order : Order)
setItem(item : ItemMaster)
ItemMasterRepository
findByItemName(name : String) : Optional<ItemMaster>
findAll() : List<ItemMaster>
save(item : ItemMaster) : ItemMaster
references
manages
ItemMaster
id : Long
itemName : String
itemPrice : Integer
itemCategory : ItemCategory
itemGenre : ItemGenre
itemImgLink : String
totalQty : Integer
soldQty : Integer
getId() : Long
getItemName() : String
getItemPrice() : Integer
getItemCategory() : ItemCategory
has
ItemDaily
dailyId : Long
date : LocalDate
initialQty : Integer
soldQty : Integer
item : ItemMaster
getDailyId() : Long
getAvailableQty() : int
setInitialQty(qty : Integer)
setSoldQty(qty : Integer)

3.6 Sequence Diagram

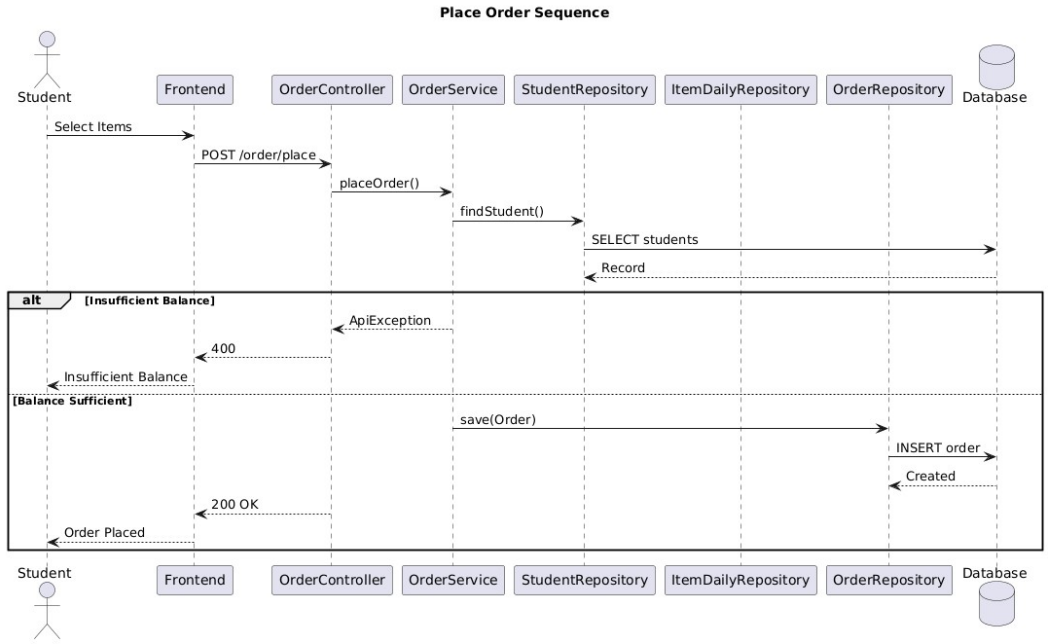
1.Student registration Sequence



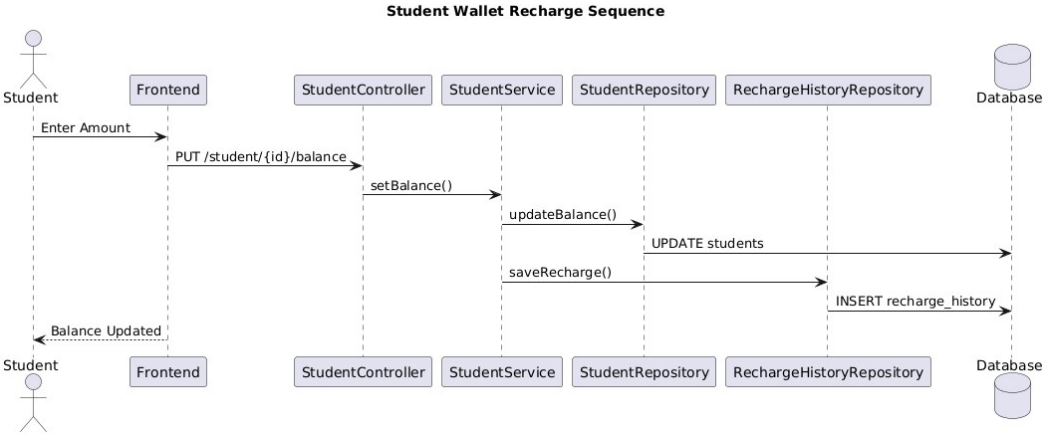
2.Student Login Sequence



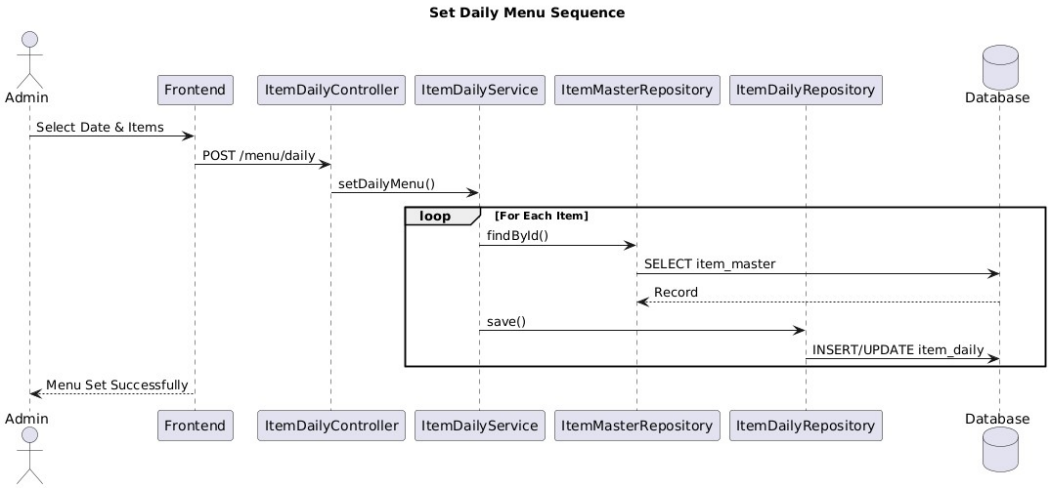
3.Place order sequence



4.Wallet sequence

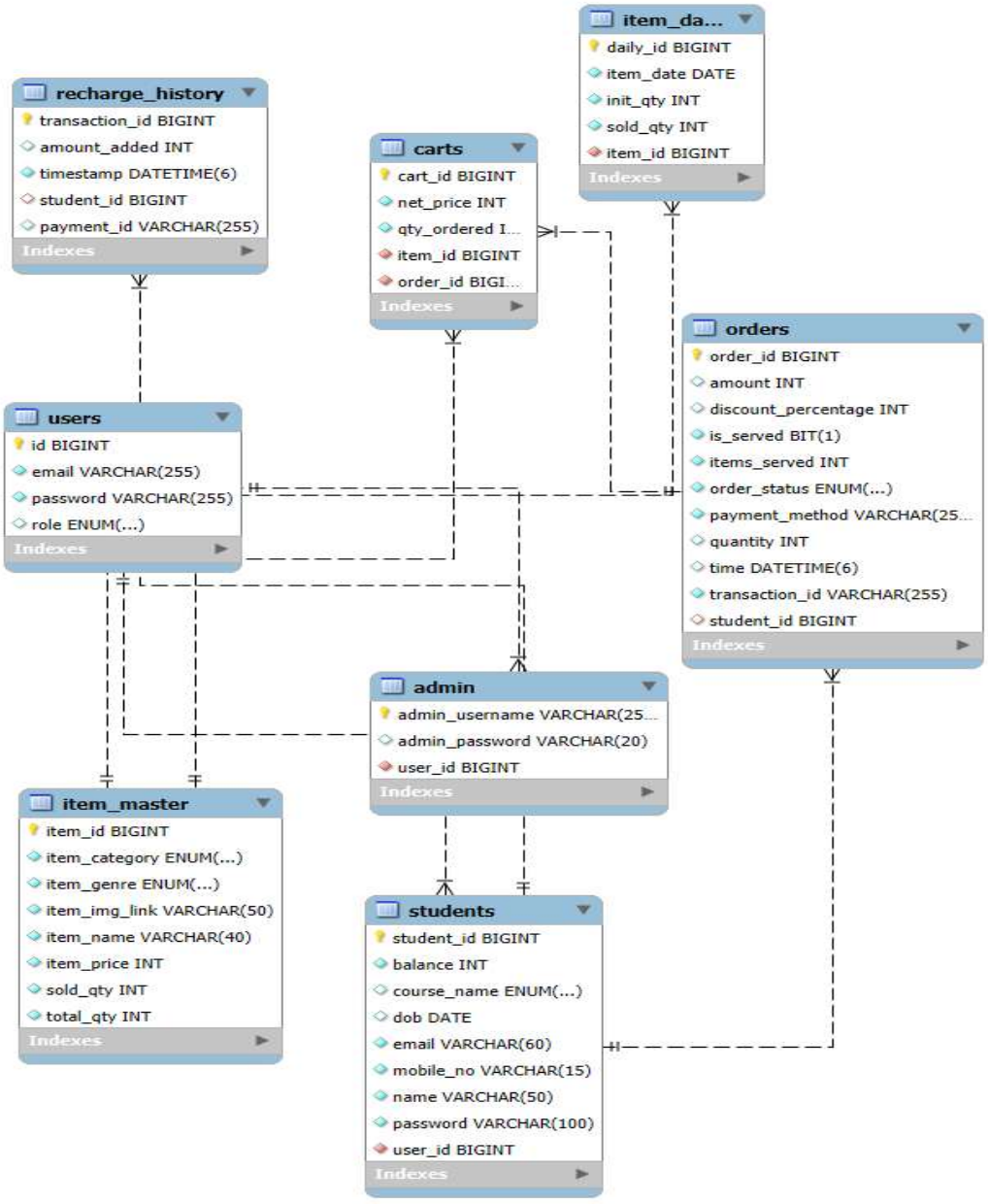


5.Set daily sequence



4. DATABASE DESIGN

4.1 Design:



4.2 Tables:

The following table structures depict the database design

Table 1 Tables

```
mysql> show tables;
+------------------+
| Tables_in_cms    |
+------------------+
| admin            |
| carts            |
| item_daily       |
| item_master      |
| orders           |
| recharge_history |
| students         |
| users            |
+------------------+
8 rows in set (0.03 sec)
```

Table 2 Users

```
mysql> desc users;
+----------+-------------------------+------+-----+---------+----------------+
| Field    | Type                    | Null | Key | Default | Extra          |
+----------+-------------------------+------+-----+---------+----------------+
| id       | bigint                  | NO   | PRI | NULL    | auto_increment |
| email    | varchar(255)            | NO   | UNI | NULL    |                |
| password | varchar(255)            | NO   |     | NULL    |                |
| role     | enum('STUDENT','ADMIN') | YES  |     | NULL    |                |
+----------+-------------------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

Table 3 Students

```
mysql> desc students;
+-------------+-----------------------------------+------+-----+---------+----------------+
| Field       | Type                              | Null | Key | Default | Extra          |
+-------------+-----------------------------------+------+-----+---------+----------------+
| student_id  | bigint                            | NO   | PRI | NULL    | auto_increment |
| balance     | int                               | NO   |     | NULL    |                |
| course_name | enum('DAC','DBDA','DAI','DITISS') | YES  |     | NULL    |                |
| dob         | date                              | YES  |     | NULL    |                |
| email       | varchar(60)                       | NO   | UNI | NULL    |                |
| mobile_no   | varchar(15)                       | NO   |     | NULL    |                |
| name        | varchar(50)                       | NO   |     | NULL    |                |
| password    | varchar(100)                      | NO   |     | NULL    |                |
| user_id     | bigint                            | NO   | UNI | NULL    |                |
+-------------+-----------------------------------+------+-----+---------+----------------+
```

Table 4 admin

```
mysql> desc admin;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| admin_username | varchar(255) | NO   | PRI | NULL    |       |
| admin_password | varchar(20)  | YES  |     | NULL    |       |
| user_id        | bigint       | NO   | UNI | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Table 5 orders

```
mysql> desc orders;
+--------------------+-------------------------+------+-----+---------+----------------+
| Field              | Type                    | Null | Key | Default | Extra          |
+--------------------+-------------------------+------+-----+---------+----------------+
| order_id           | bigint                  | NO   | PRI | NULL    | auto_increment |
| amount             | int                     | YES  |     | NULL    |                |
| discount_percentage | int                    | YES  |     | NULL    |                |
| is_served          | bit(1)                  | NO   |     | NULL    |                |
| items_served       | int                     | NO   |     | NULL    |                |
| order_status       | enum('PENDING','SERVED') | NO  |     | NULL    |                |
| payment_method     | varchar(255)            | NO   |     | NULL    |                |
| quantity           | int                     | YES  |     | NULL    |                |
| time               | datetime(6)             | YES  |     | NULL    |                |
| transaction_id     | varchar(255)            | NO   | UNI | NULL    |                |
| student_id         | bigint                  | YES  | MUL | NULL    |                |
+--------------------+-------------------------+------+-----+---------+----------------+
11 rows in set (0.00 sec)
```

Table 6 recharge

```
mysql> desc recharge_history;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| transaction_id | bigint       | NO   | PRI | NULL    | auto_increment |
| amount_added   | int          | YES  |     | NULL    |                |
| timestamp      | datetime(6)  | NO   |     | NULL    |                |
| student_id     | bigint       | YES  | MUL | NULL    |                |
| payment_id     | varchar(255) | YES  |     | NULL    |                |
+----------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

Table 7 item_daily

```
mysql> desc item_daily;
+-----------+--------+------+-----+---------+----------------+
| Field     | Type   | Null | Key | Default | Extra          |
+-----------+--------+------+-----+---------+----------------+
| daily_id  | bigint | NO   | PRI | NULL    | auto_increment |
| item_date | date   | NO   | MUL | NULL    |                |
| init_qty  | int    | NO   |     | NULL    |                |
| sold_qty  | int    | NO   |     | NULL    |                |
| item_id   | bigint | NO   | MUL | NULL    |                |
+-----------+--------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

Table 8 item_master

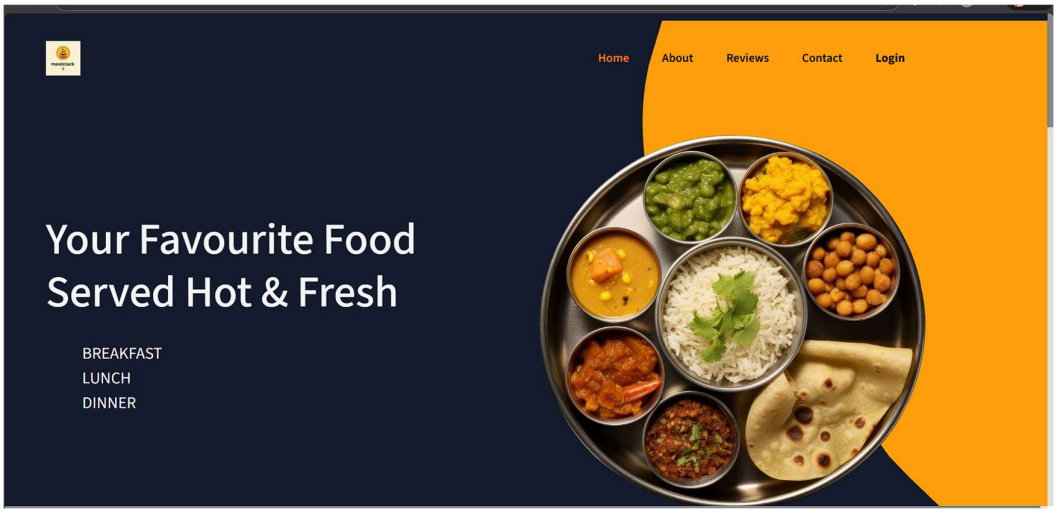
```
mysql> desc item_master;
+--------------+-----------------------------------------------------------------+------+-----+---------+----------------+
| Field        | Type                                                            | Null | Key | Default | Extra          |
+--------------+-----------------------------------------------------------------+------+-----+---------+----------------+
| item_id      | bigint                                                          | NO   | PRI | NULL    | auto_increment |
| item_category | enum('Breakfast','Lunch','Snacks','Dinner','Beverages')        | NO   |     | NULL    |                |
| item_genre   | enum('SouthIndian','Oriental','NorthIndian','Maharashtrian')    | NO   |     | NULL    |                |
| item_img_link | varchar(50)                                                    | NO   | UNI | NULL    |                |
| item_name    | varchar(40)                                                     | NO   | UNI | NULL    |                |
| item_price   | int                                                             | NO   |     | NULL    |                |
| sold_qty     | int                                                             | NO   |     | 0       |                |
| total_qty    | int                                                             | NO   |     | 0       |                |
+--------------+-----------------------------------------------------------------+------+-----+---------+----------------+
8 rows in set (0.00 sec)
```

Table 9 carts

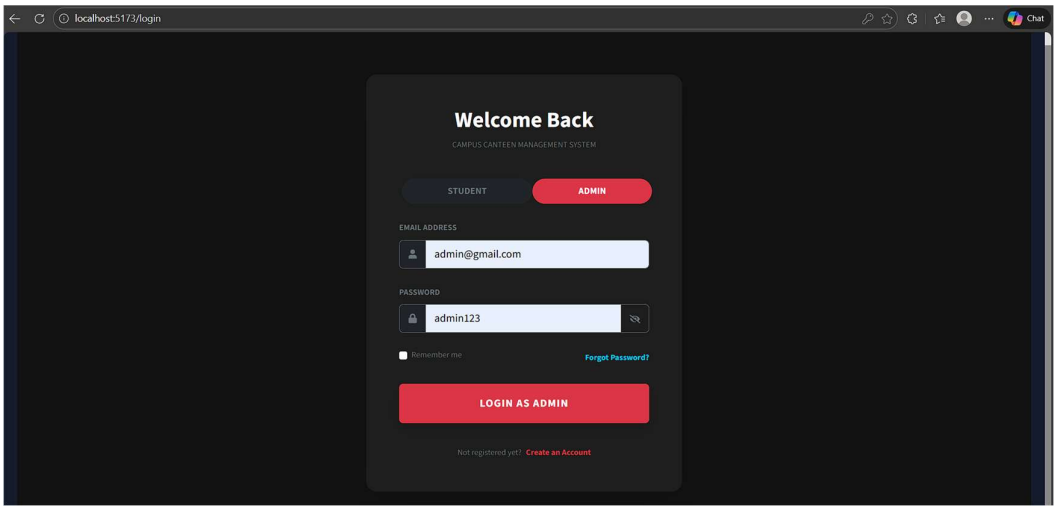
```
mysql> desc carts;
+-------------+--------+------+-----+---------+----------------+
| Field       | Type   | Null | Key | Default | Extra          |
+-------------+--------+------+-----+---------+----------------+
| cart_id     | bigint | NO   | PRI | NULL    | auto_increment |
| net_price   | int    | NO   |     | NULL    |                |
| qty_ordered | int    | NO   |     | NULL    |                |
| item_id     | bigint | NO   | MUL | NULL    |                |
| order_id    | bigint | NO   | MUL | NULL    |                |
+-------------+--------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

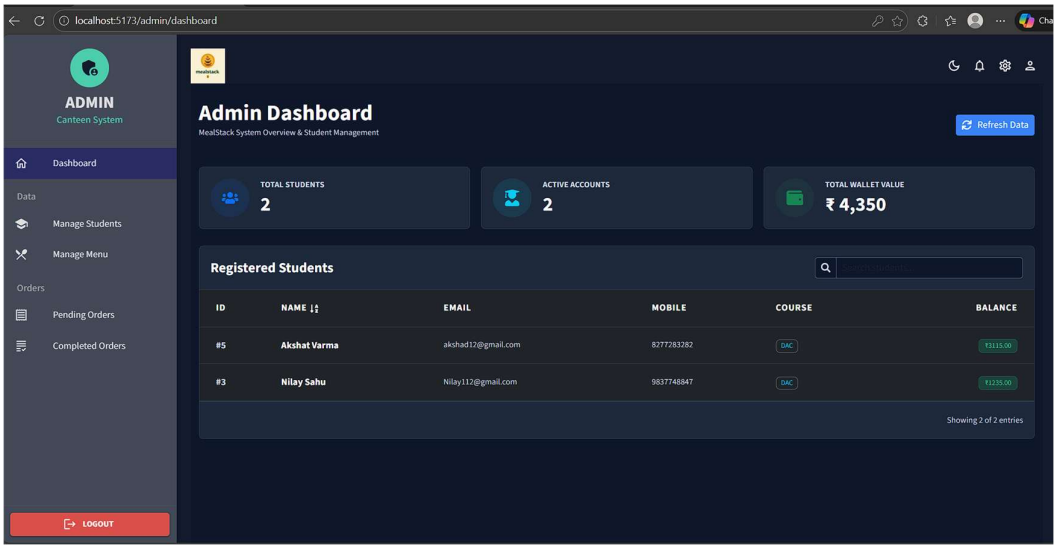
5.SNAPSHOTS

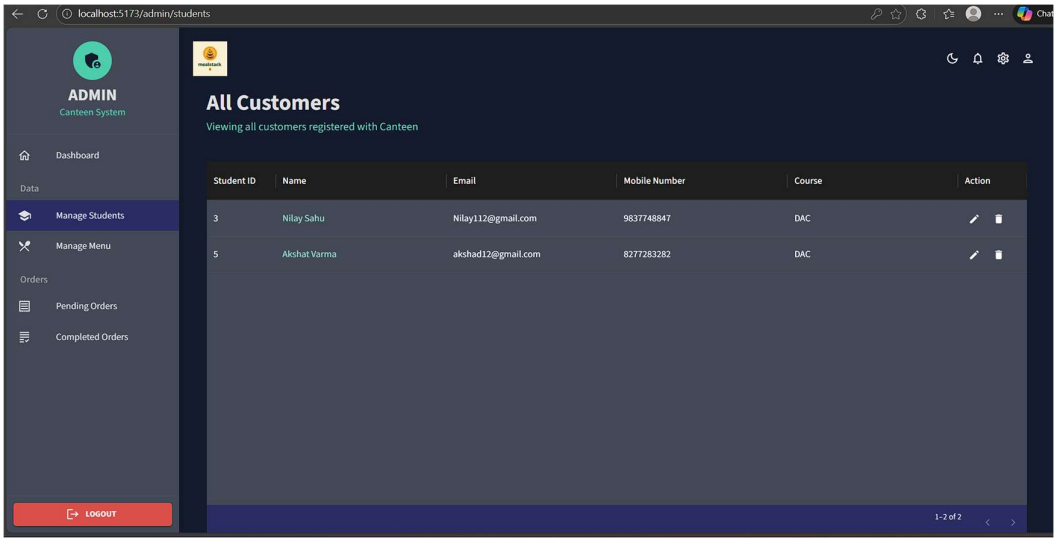
Home page:

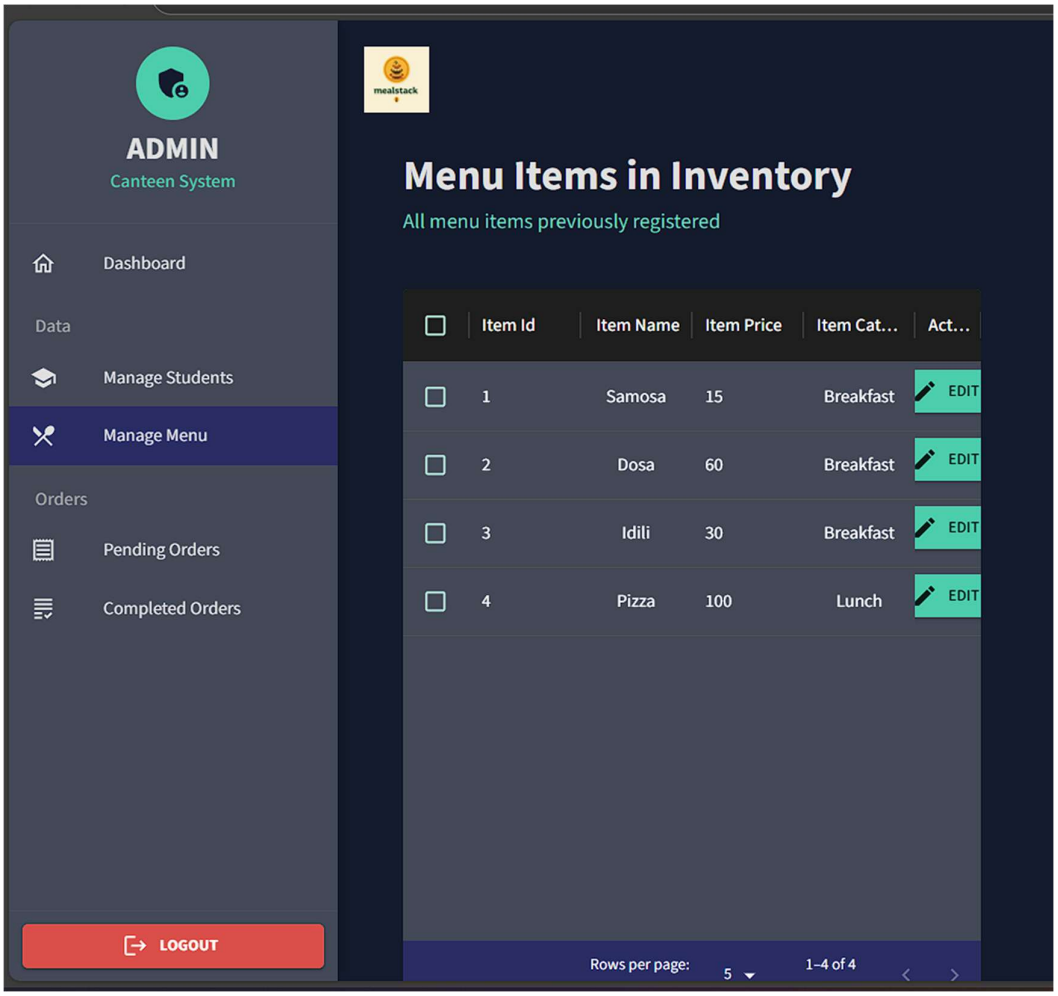


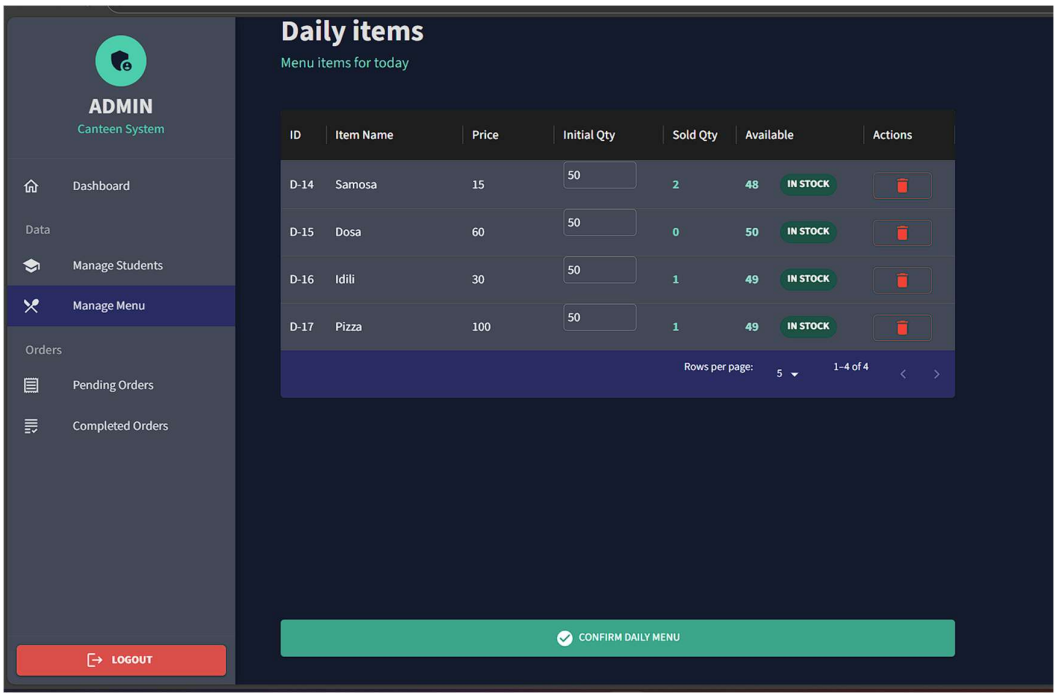
Admin DashBoard:

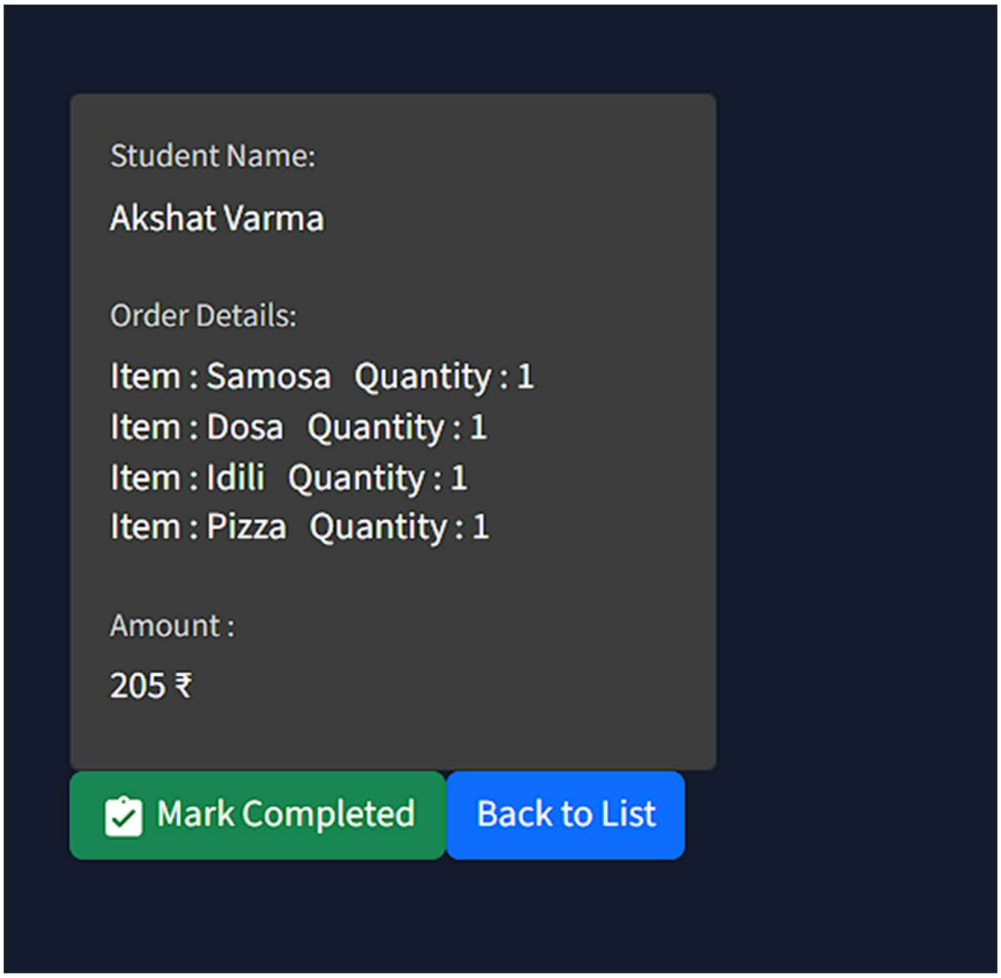


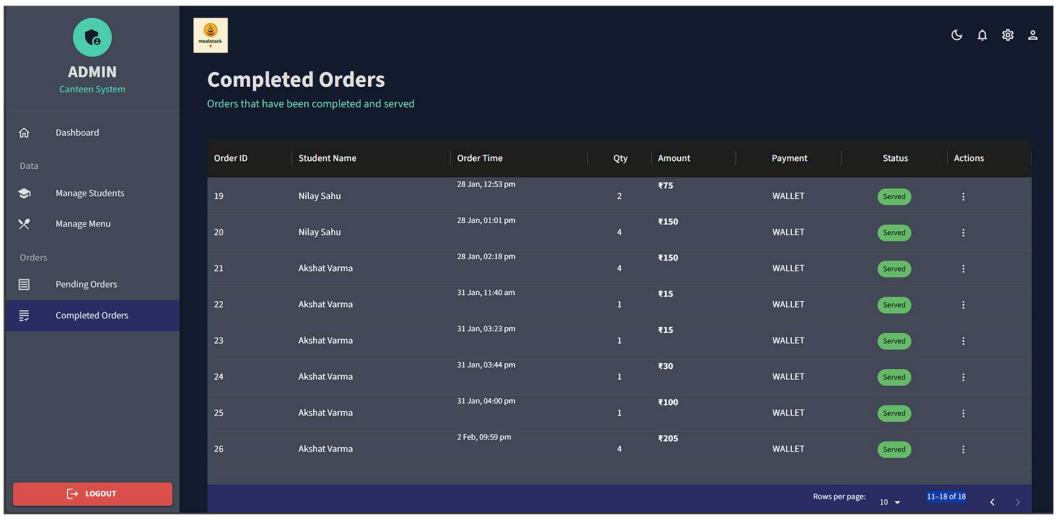

localhost:5173/admin/dashboard
ADMIN
Canteen System
Dashboard
Data
Manage Students
Manage Menu
Orders
Pending Orders
Completed Orders
LOGOUT
Admin Dashboard
MealStack System Overview & Student Management
Refresh Data
TOTAL STUDENTS
2
ACTIVE ACCOUNTS
2
TOTAL WALLET VALUE
₹ 4,350
Registered Students
ID
NAME
EMAIL
MOBILE
COURSE
BALANCE
#5
Akshat Varma
akshad12@gmail.com
8277283282
DAC
₹3115.00
#3
Nilay Sahu
Nilay112@gmail.com
9837748847
DAC
₹3235.00
Showing 2 of 2 entries


localhost:5173/admin/students
ADMIN
Canteen System
Dashboard
Data
Manage Students
Manage Menu
Orders
Pending Orders
Completed Orders
LOGOUT
All Customers
Viewing all customers registered with Canteen
Student ID
Name
Email
Mobile Number
Course
Action
3
Nilay Sahu
Nilay112@gmail.com
9837748847
DAC
5
Akshat Varma
akshad12@gmail.com
8277283282
DAC
1–2 of 2

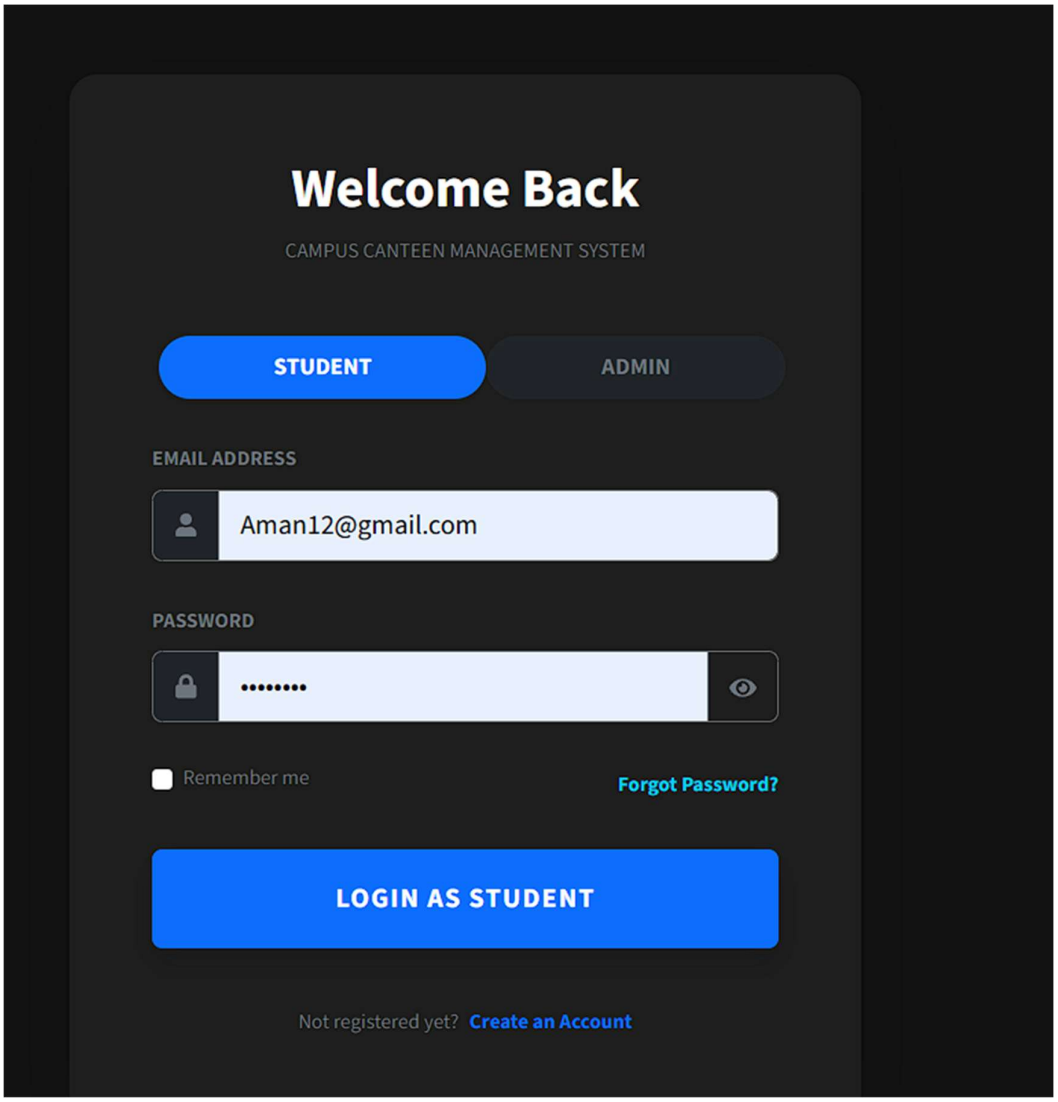

ADMIN
Canteen System
Dashboard
Data
Manage Students
Manage Menu
Orders
Pending Orders
Completed Orders
LOGOUT
Menu Items in Inventory
All menu items previously registered
Item Id
Item Name
Item Price
Item Cat...
Act...
1 Samosa 15 Breakfast EDIT
2 Dosa 60 Breakfast EDIT
3 Idili 30 Breakfast EDIT
4 Pizza 100 Lunch EDIT
Rows per page: 5
1–4 of 4

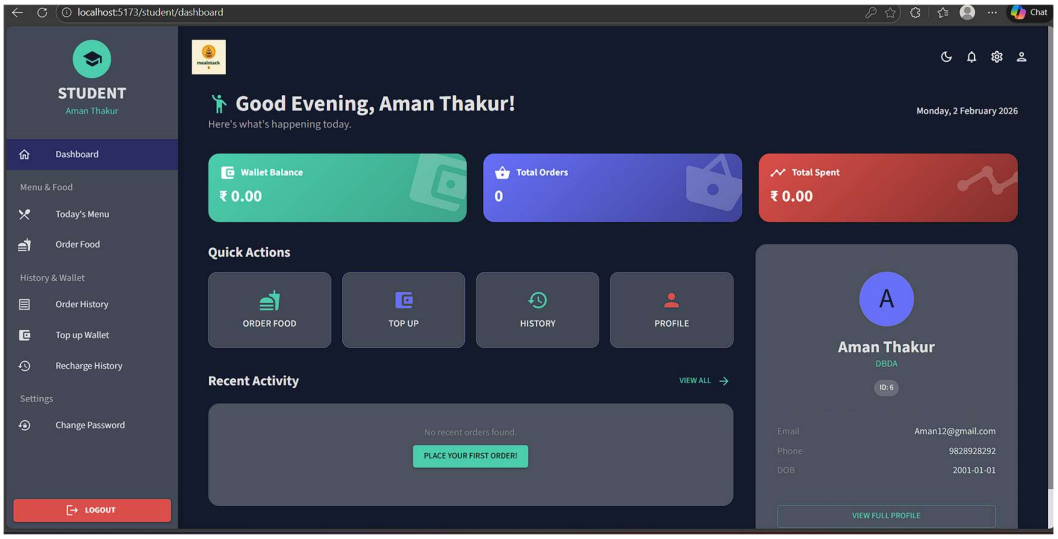

ADMIN
Canteen System
Dashboard
Data
Manage Students
Manage Menu
Orders
Pending Orders
Completed Orders
LOGOUT
Daily items
Menu items for today
ID
Item Name
Price
Initial Qty
Sold Qty
Available
Actions
D-14 Samosa 15 50 2 48 IN STOCK
D-15 Dosa 60 50 0 50 IN STOCK
D-16 Idili 30 50 1 49 IN STOCK
D-17 Pizza 100 50 1 49 IN STOCK
Rows per page: 5
1–4 of 4
CONFIRM DAILY MENU

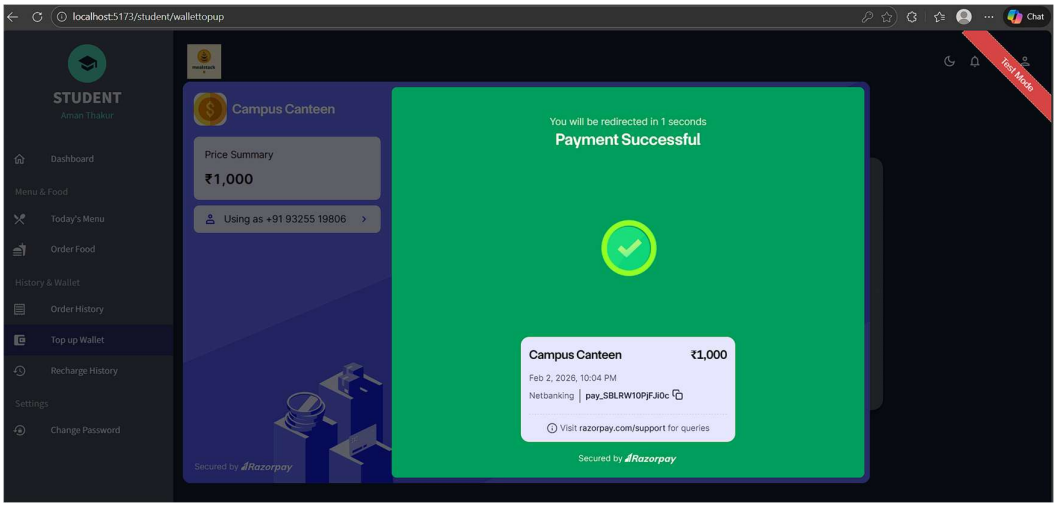

Student Name:
Akshat Varma
Order Details:
Item : Samosa Quantity : 1
Item : Dosa Quantity : 1
Item : Idili Quantity : 1
Item : Pizza Quantity : 1
Amount :
205 ₹
Mark Completed
Back to List

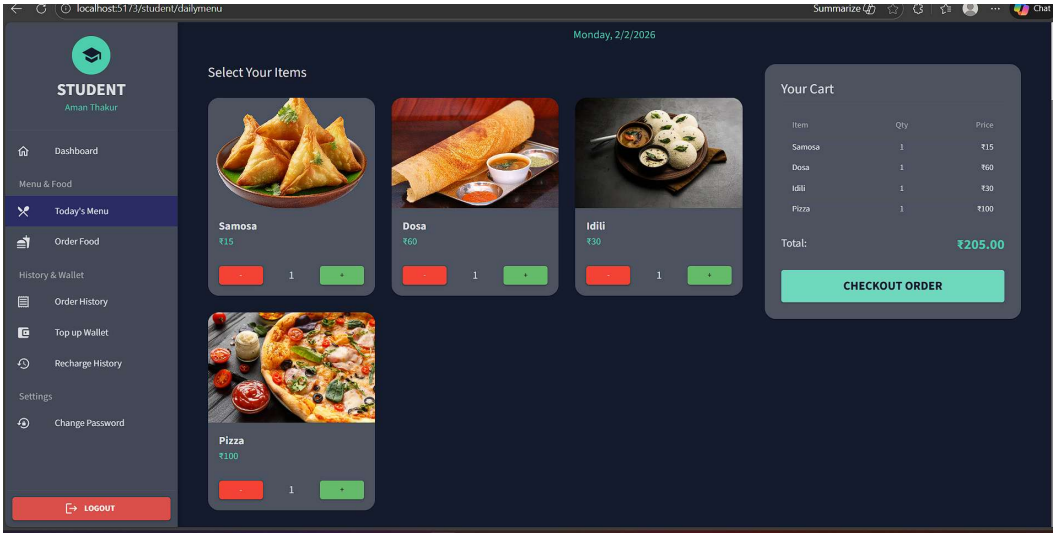

ADMIN
Canteen System
Dashboard
Data
Manage Students
Manage Menu
Orders
Pending Orders
Completed Orders
LOGOUT
Completed Orders
Orders that have been completed and served
Order ID | Student Name | Order Time | Qty | Amount | Payment | Status | Actions
19 | Nilay Sahu | 28 Jan, 12:53 pm | 2 | ₹75 | WALLET | Served
20 | Nilay Sahu | 28 Jan, 01:01 pm | 4 | ₹150 | WALLET | Served
21 | Akshat Varma | 28 Jan, 02:18 pm | 4 | ₹150 | WALLET | Served
22 | Akshat Varma | 31 Jan, 11:40 am | 1 | ₹15 | WALLET | Served
23 | Akshat Varma | 31 Jan, 03:23 pm | 1 | ₹15 | WALLET | Served
24 | Akshat Varma | 31 Jan, 03:44 pm | 1 | ₹30 | WALLET | Served
25 | Akshat Varma | 31 Jan, 04:00 pm | 1 | ₹100 | WALLET | Served
26 | Akshat Varma | 2 Feb, 09:59 pm | 4 | ₹205 | WALLET | Served
Rows per page: 10
11–18 of 18

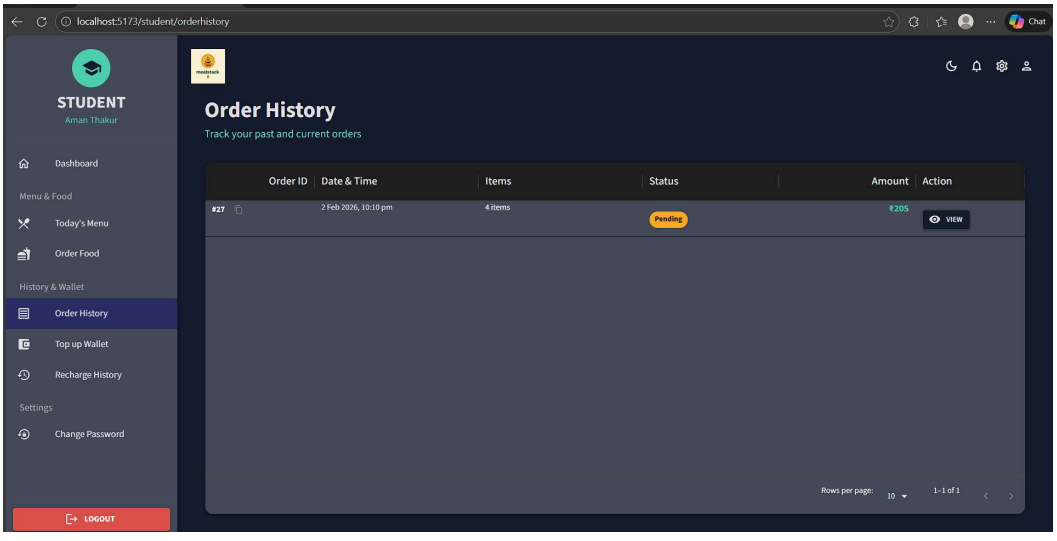
Student Dashboard:

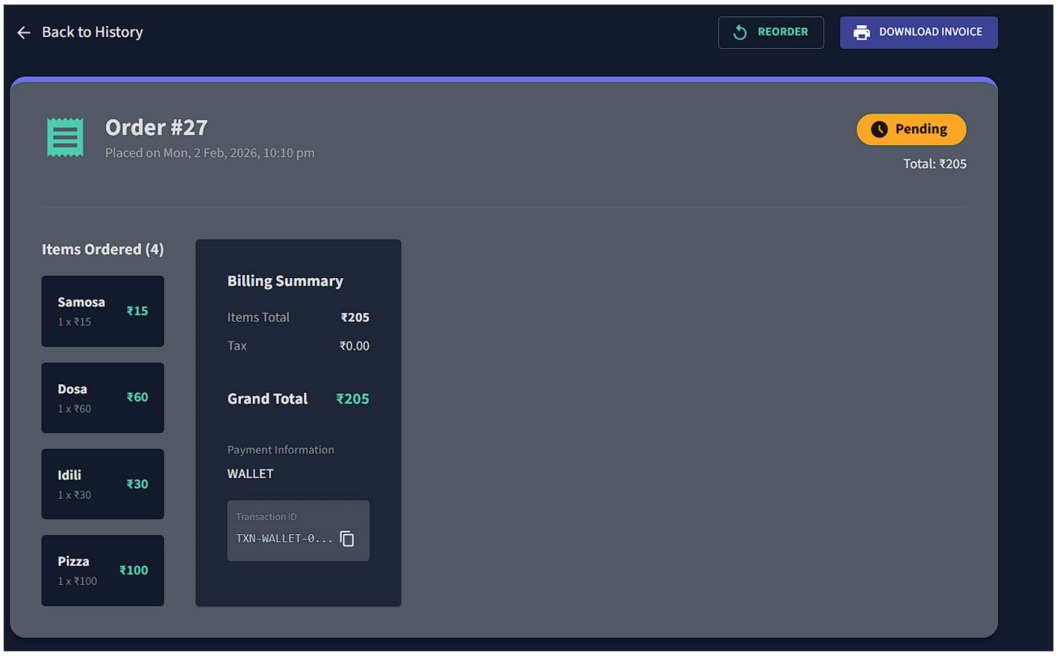


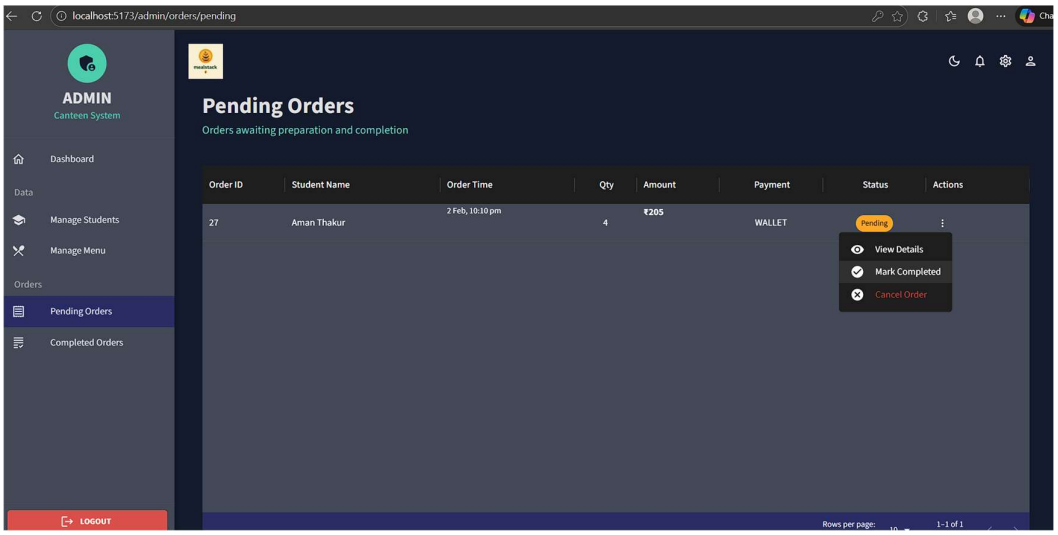


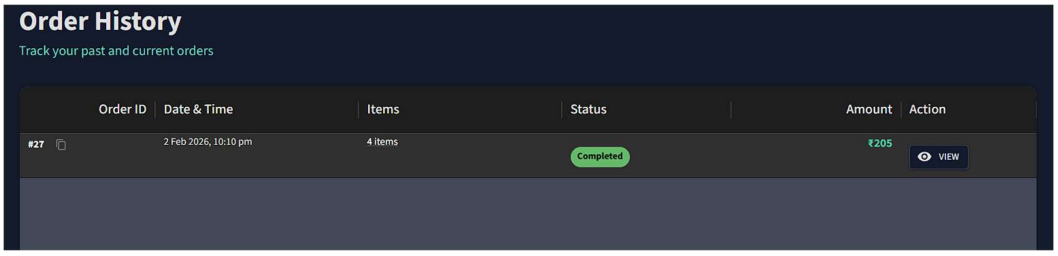

localhost:5173/student/wallettopup
STUDENT
Aman Thakur
Dashboard
Menu & Food
Today's Menu
Order Food
History & Wallet
Order History
Top up Wallet
Recharge History
Settings
Change Password
Campus Canteen
Price Summary
₹1,000
Using as +91 93255 19806
Secured by Razorpay
You will be redirected in 1 seconds
Payment Successful
Campus Canteen ₹1,000
Feb 2, 2026, 10:04 PM
Netbanking | pay_SBLRW10PjFJi0c
Visit razorpay.com/support for queries
Secured by Razorpay
Test Mode


localhost:5173/student/dailymenu
STUDENT
Aman Thakur
Dashboard
Menu & Food
Today's Menu
Order Food
History & Wallet
Order History
Top up Wallet
Recharge History
Settings
Change Password
LOGOUT
Monday, 2/2/2026
Select Your Items
Samosa
₹15
Dosa
₹60
Idili
₹30
Pizza
₹100
Your Cart
Item Qty Price
Samosa 1 ₹15
Dosa 1 ₹60
Idili 1 ₹30
Pizza 1 ₹100
Total: ₹205.00
CHECKOUT ORDER


localhost:5173/student/orderhistory
STUDENT
Aman Thakur
Dashboard
Menu & Food
Today's Menu
Order Food
History & Wallet
Order History
Top up Wallet
Recharge History
Settings
Change Password
LOGOUT
Order History
Track your past and current orders
Order ID
Date & Time
Items
Status
Amount
Action
#27
2 Feb 2026, 10:10 pm
4 items
Pending
₹205
VIEW
Rows per page: 10
1–1 of 1


Back to History
REORDER
DOWNLOAD INVOICE
Order #27
Placed on Mon, 2 Feb, 2026, 10:10 pm
Pending
Total: ₹205
Items Ordered (4)
Samosa
1 x ₹15
₹15
Dosa
1 x ₹60
₹60
Idili
1 x ₹30
₹30
Pizza
1 x ₹100
₹100
Billing Summary
Items Total
₹205
Tax
₹0.00
Grand Total
₹205
Payment Information
WALLET
Transaction ID
TXN-WALLET-0...


localhost:5173/admin/orders/pending
ADMIN
Canteen System
Dashboard
Data
Manage Students
Manage Menu
Orders
Pending Orders
Completed Orders
LOGOUT
Pending Orders
Orders awaiting preparation and completion
Order ID
Student Name
Order Time
Qty
Amount
Payment
Status
Actions
27
Aman Thakur
2 Feb, 10:10 pm
4
₹205
WALLET
Pending
View Details
Mark Completed
Cancel Order
Rows per page:
1–1 of 1


Order History
Track your past and current orders
Order ID
Date & Time
Items
Status
Amount
Action
#27
2 Feb 2026, 10:10 pm
4 items
Completed
₹205
VIEW

6.CONCLUSION

MealStack successfully digitizes canteen operations, providing an efficient platform for food ordering and inventory management in educational institutions.

Key Achievements:

• Complete digitization eliminating manual processes and cash handling

• Order processing reduced from minutes to seconds

• Real-time inventory control with automatic updates

• Secure JWT authentication and role-based access

• Intuitive, mobile-responsive interface

• Comprehensive analytics for data-driven decisions

Impact:

For Students: Convenient cashless ordering, transparent pricing, reduced wait times, complete transaction history

For Administrators: Reduced manual effort, better inventory control, valuable business insights, eliminated cash reconciliation

For Institution: Improved satisfaction, operational efficiency, successful digital transformation

Technical Success:

Effective integration of Spring Boot, React, Redux, Material-UI, and MySQL demonstrating modern full-stack development with RESTful architecture and modular design.

Future Enhancements:

QR code pickup, push notifications, AI recommendations, nutrition tracking, mobile apps, multi-canteen support, predictive analytics, ERP integration

MealStack demonstrates successful application of software engineering principles to real-world problems, exceeding functional requirements in usability, security, and performance. The modular architecture ensures easy maintenance and future enhancements.

7.REFERENCES

1. Spring Boot Documentation - https://docs.spring.io/spring-boot/

2. Spring Security Reference - https://docs.spring.io/spring-security/

3. React Documentation - https://react.dev/

4. Material-UI Documentation - https://mui.com/

5. "Spring Boot in Action" by Craig Walls, Manning Publications

6. "Learning React" by Alex Banks, O'Reilly Media

7. "Database Systems" by Garcia-Molina, Ullman, Widom

8. Codd, E.F. (1970) "Relational Model of Data", ACM

9. Fielding, R.T. (2000) "REST Architecture", UC Irvine

10. MySQL Documentation - https://dev.mysql.com/doc/

11. JWT.io - https://jwt.io/

12. OWASP Security - https://owasp.org/

13. Baeldung Spring Tutorials - https://www.baeldung.com/

14. MDN Web Docs - https://developer.mozilla.org/

15. Razorpay Documentation - https://razorpay.com/doc