Software Requirements Specification

for

Campus Craves

Version 0.1

Prepared by

Group 19 **Group Name: Supervised Learners**

Name	Roll no.	E-mail
Aayush Gautam	220020	aayushg22@iitk.ac.in
Abhishek Kumar	210040	abhikumar21@iitk.ac.in
Anand Chutani	220130	anandc22@iitk.ac.in
Anany Dev Choudhary	220135	ananydc22@iitk.ac.in
Chatla Sowmya Sri	200293	srisowmya20@iitk.ac.in
Lokesh Mehra	220591	lokeshm22@iitk.ac.in
Pranshu Mani Tripathi	220800	pranshumt22@iitk.ac.in
Siddharth Pathak	211034	siddharthp21@iitk.ac.in
Sidharth A S	221056	sidharthas22@iitk.ac.in
Sparsh Gupta	221084	gsparsh22@iitk.ac.in
Ujjwal Kumar	211123	ujjwalk21@iitk.ac.in

Course: CS253

Mentor TA: Mr. Ashish Singh

Date: 24 January, 2025

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
0.1	Abhishek Kumar Chatla Sowmya Sri Pranshu Mani Tripathi Siddharth Pathak Sparsh Gupta Ujjwal Kumar	Initial Draft	24/01/2025

1 Introduction

1.1 Product Scope

The “Campus Craves” application allows the digitization of grocery stores and canteens on the IIT Kanpur campus. Canteen and store owners can use the app to advertise their goods, services and menu options online. The online application allows students and campus residents to conveniently make orders online as long as they can access a web browser. In addition, it helps store and canteen owners take and manage orders, keep tabs on stocks, and gain access to some sales information.

The main goals of the product include:

- Accessing services and products more widely.
- Accessing different ways of making purchases.
- Reducing the need to move around, particularly during odd hours or a pandemic.
- Lessening unnecessary interactions for the clientele.
- Increasing the clientele of the business.
- Increasing competitiveness among business owners.
- Offering relevant logistical and accounting information on sales made and stock in hand.
- Enhancing sales in the business.

The application will broadly provide the following features for canteens and grocery stores:

- **Catalog of Products under One Roof:** Store owners can exhibit a catalog of their products and services while handling inventory and price update management in an effortless manner.
- **Ease of Convenience to Customers:** Store owners can offer flexible options of picking up orders or on-door delivery, catering to the needs of varied customers.
- **Simplified Shopping:** Students, like other customers, can view a list of multiple store catalogs, compare prices, place orders, and receive digital bills all through a single platform.
- **Simplified Customer Interaction:** Customers can view sales history, order status, and promo codes offers, while store owners can view and export sales records, make promotional offers, and analyze trends to make business smoother.

There are several benefits associated with the product:

- Greater access to a wider variety of products and services.
- Sales and transaction records organized and maintainable.
- No need to physically visit the stores.
- Price comparison across different stores.
- Customers can have real-time views of the status of their orders.

1.2 Intended Audience and Document Overview

Intended Viewers:

As we start the design, developers would use this document to communicate and proceed. Then at the time of testing, testers would use this document. It can then be used to form a user manual using some subset of the SRS.

Sections and Organisation:

The SRS starts with sections that describe its importance and how to use it, then it moves forward with the overview and functionality of the product. Then it covers a guide towards the interfaces which make it easier for a user to understand how they need to interact with the product, covering some use cases to wind up.

Sequence for Important Sections:

● A Developer or a Tester can start reading the document with focus on overview (2.1), functionality (2.2), design and implementation (2.3), interfaces (3.1) and functional requirements (3.2).
● A user should focus on the scope (1.1), overview (2.1) and functionality (2.2), and then the specific requirements, going through the sections 3.1.1, 3.1.3 and 3.2.

1.3 Definitions, Acronyms and Abbreviations

Term	Definition
ACID	Atomicity, Consistency, Isolation, Durability
AWS	Amazon Web Services
CRUD	Create, Read, Update, Delete
DB	Database
DDB	Distributed Database
ER	Entity Relationship
GUI	Graphical User Interface
HTML	Hypertext Markup Language
JS	JavaScript
SOTA	State of the Arts
SRS	Software Requirements Specifications

1.4 Document Conventions

Formatting Conventions:

- This document is written with an Arial font of size 11 with single spacing and 1-inch margins.
- Words highlighted with bold in the same font space represent terms whose explanations are either given in footnotes or separately in the same section.
- Italics highlight has been used for comments.
- Underline has been used for headings in subsections.
- Bullet point ordering has been used as a listing typesetting tool.

Naming Conventions:

- Buyer: Any potential end customer for the sellers in the application.
- Seller: Any business owner who catalogs products in the application.
- Users: Buyers and Sellers
- Admin: Administrator who gives technological and product support to the users.

1.5 References and Acknowledgments

The following resources were referred to in the process of making this document:

- Use-Case Model - Javatpoint
- Figma
- Creately
- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.
- Explore the UML sequence diagram - IBM

We'd also like to acknowledge the help of our TA, Mr. Ashish Singh, for their valuable input in the creation of this document. We also would like to thank Prof. Indranil Saha for providing the SRS template and teaching the concepts.

2 Overall Description

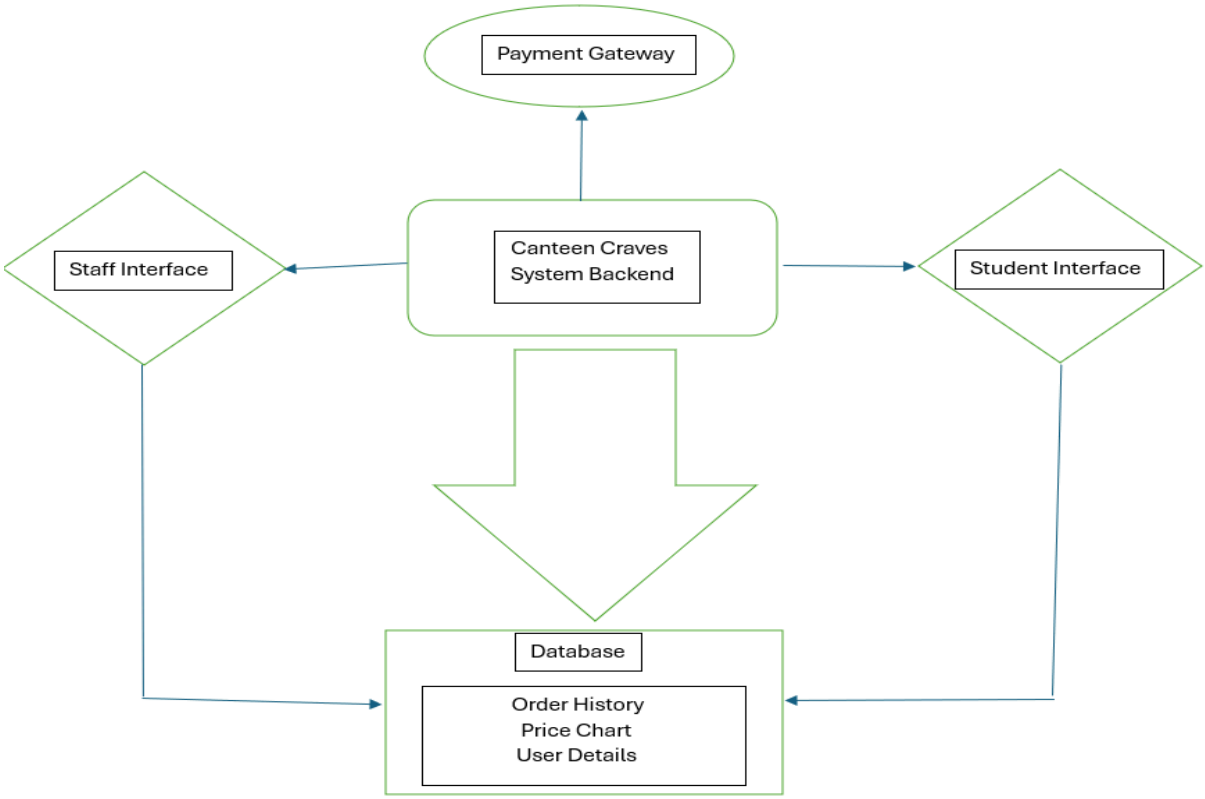
2.1 Product Overview

The "Campus Craves" web-based application is an innovative, standalone offering to enhance the ordering and dining experience at IIT Kanpur. This project is structured to operate as a cohesive platform that links all 16 canteens scattered throughout the campus, thereby making it possible for buyers to discover menus and place orders in an organized and effective way.

2.1.1 Product Perspective:

The Campus Craves application functions autonomously and has a robust database that manages user profiles, orders, and data specific to the canteen. It stands apart as it is purposefully designed to modernize the campus dining experience. The platform boasts user-friendly features allowing effortless menu browsing and order placement. On the other hand, the admin dashboard is crafted specifically for canteen managers, enabling them to efficiently oversee menus, track order statuses, and improve their services based on customer data.

2.1.2 System Overview:



The diagram illustrates the system's major components, showcasing interactions between different user roles and external interfaces. The core components include the Canteen Staff Interface and Student Interface. These interfaces connect to a centralized database that stores information about price charts, user order history, and user details. Students can access their individual data through their interface, while managers and staff can access everyone's data from the database. Note: Staff can access only the bookings section of a student through their roll number.

2.2 Product Functionality

This product is intended to come with the following key functionalities:

- User Management: Role-based accounts for buyers and sellers with password recovery via email.
- Home Page: Buyers can browse menus, search items, and view canteen details.
- Product Search: Search by canteen or category with detailed product information.
- Cart Management: Buyers can add/remove items, adjust quantities, and view cart summaries.
- Order Placement: Buyers can checkout with delivery address and multiple payment options.
- Order History: Buyers can view past orders with details like status, date, and payment method.
- Seller Dashboard: Sellers can manage orders, categories, and products with ease.
- Order Management: Sellers can track and process orders with real-time updates.
- Category Management: Sellers can add, edit, or delete product categories.
- Product Management: Sellers can update inventory and toggle product availability.
- Notifications: Alerts for buyers and sellers about order updates.
- Global Search: Buyers can find stores, items, or services across the platform.
- Payment Integration: Flexible payment options, including UPI, cards, and cash on delivery.
- Reporting and Analytics: Sales reports, product performance insights, and order trends for Sellers and Admins.

2.3 Design and Implementation Constraints

2.3.1 Hardware Limitations

- The system should operate within the hardware constraints of the institution's existing infrastructure. Developers must consider the available computing resources, timing requirements, and memory limitations to ensure optimal performance.
- The Canteen Automation System needs to seamlessly integrate with the institution's existing systems, such as student databases and financial systems. Developers must adhere to specific interfaces and protocols to facilitate smooth data exchange.

2.3.2 Specific Technologies, Tools, and Databases

- **Frontend:** HTML, CSS, Javascript, React.js
- **Backend:** Django
- **Database:** MongoDB
- **Version Control:** Git with GitHub
- **Testing Tools:** Jest and Postman

2.3.3 Parallel Operations

- The system should support simultaneous users (e.g., multiple students ordering from different canteens).
- Implement WebSocket or similar protocols for real-time updates (e.g., order status, menu changes).

2.3.4 Database Management:

- The choice of database technology and its management should align with the institution's database standards. Developers need to consider data backup, recovery strategies, and optimization techniques for efficient database operations.

2.3.5 Design Conventions or Programming Standards

- Follow React component-based design and best practices for maintainability.
- Implement responsive design to support mobile, tablet, and desktop devices.
- Ensure cross-browser compatibility for Chrome, Firefox, Edge, and Safari.

2.4 Assumptions and Dependencies

Assumptions:

- All users (students, canteen operators) will have internet access.
- Canteen operators will provide accurate and timely menu updates.
- College infrastructure will allow integration with its authentication system (e.g., LDAP).
- Third-party payment gateways (e.g., Razorpay) will be reliable and support the required features.(If time permits)
- The application will scale based on anticipated user load without major reengineering.

Dependencies:

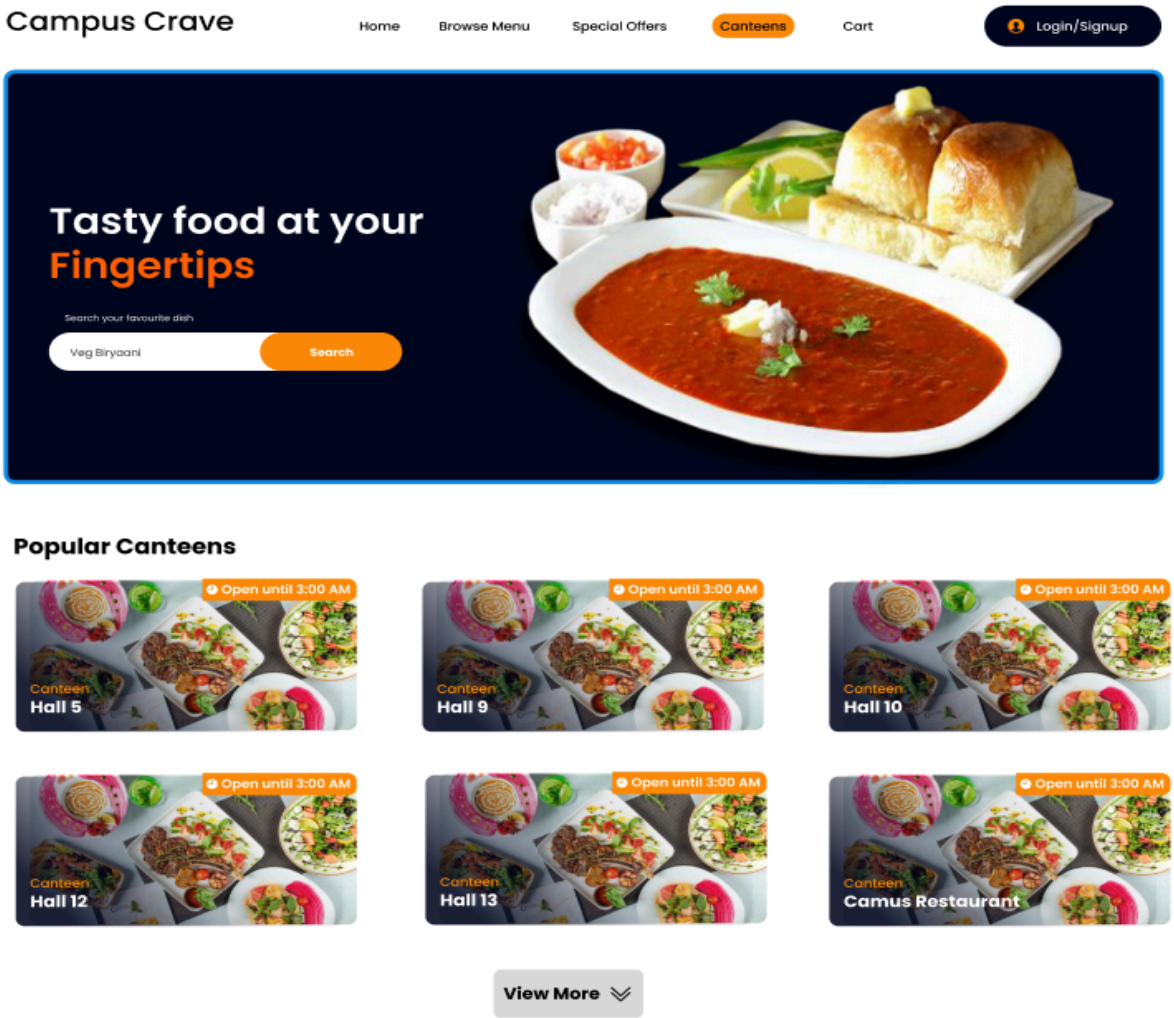
- Reliable performance of MongoDB as the primary database.
- Continuous availability of third-party APIs (payment gateways, notification services).
- Hosting environment (e.g., AWS or Heroku) remains stable and meets performance needs.
- Availability of modern browsers to support the application features.

3 Specific Requirements

3.1 External Interface Requirements

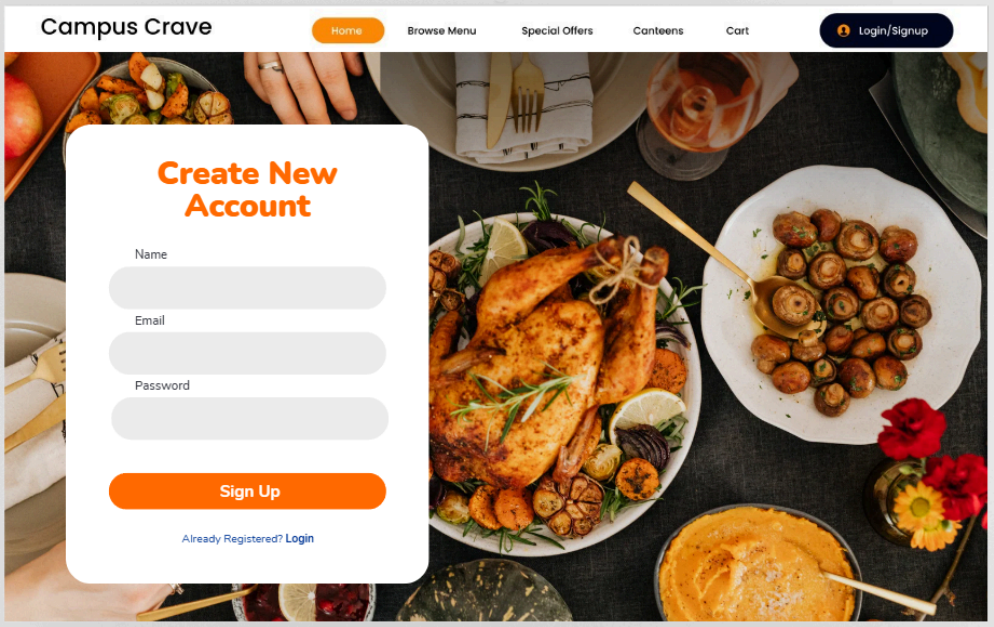
3.1.1 User Interfaces

- **Home Page:** The home page will have the application name and logo on the top left. On the right top, a sign-in button will be visible. Users will be able to click the button to proceed to the sign-in menu and access their accounts. The home page will also carry a global search bar, in which users will be able to search for specific items or services being offered by multiple canteens and grocery stores on Campus Craves. Further, users will see cards, which will represent links and icons of popular stores on the platform, such as Mess Canteen, Mini Market, and Grocery Mart. Users can easily navigate to a respective store's page by clicking on a card.

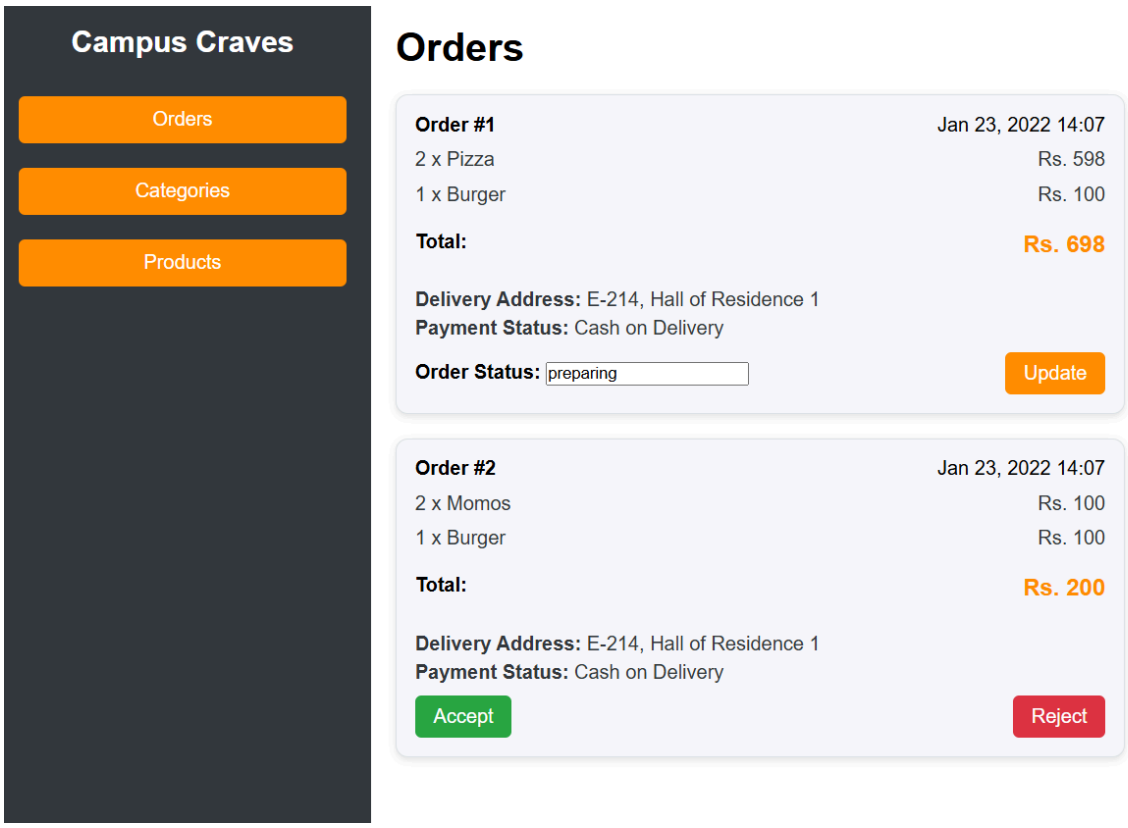


- **Sign-in:** The sign-in window will ask the user to enter their email and password in the given text fields. Once the required details are filled, users can click the Sign-in button at the bottom to access their account. At the time of sign-in, the user will also select his role as a

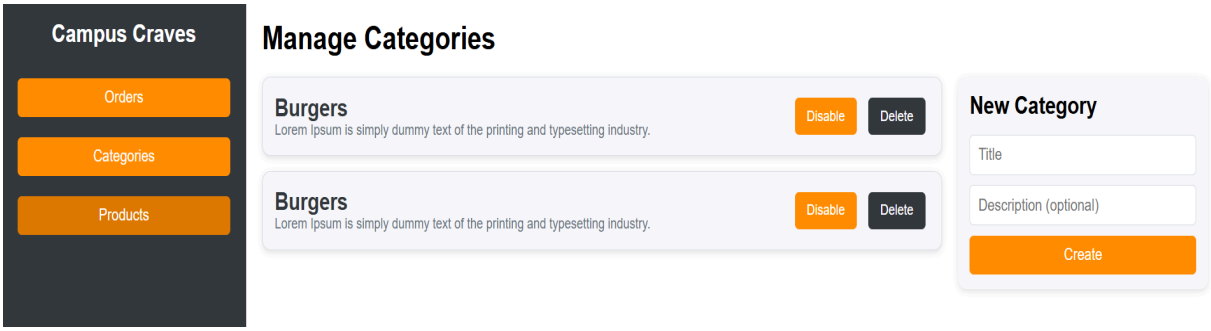
customer or a seller, so that he can go ahead with the appropriate features designed according to his needs.



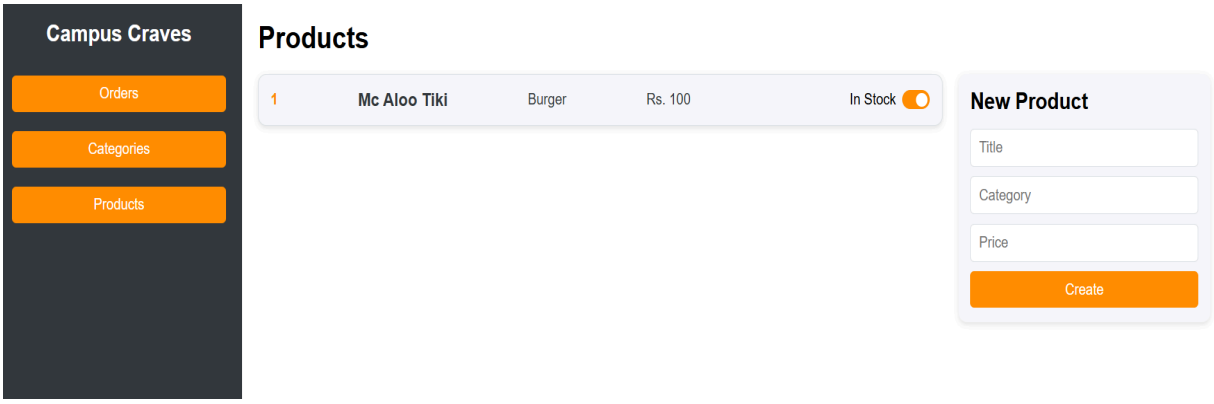
- **Orders View (in Seller's Dashboard):** From the left menu in the seller's dashboard, the "Orders" view is accessible. On this view, sellers are able to monitor both pending and non-accepted orders. For accepted orders, like Order #1, sellers may edit the status as preparing, dispatched, delivered, and can check order details: payment method, delivery address, and the total bill amount. For new orders, for example, Order #2, the sellers can view the total amount, payment method, delivery address, and accept or reject the order by clicking the appropriate buttons.



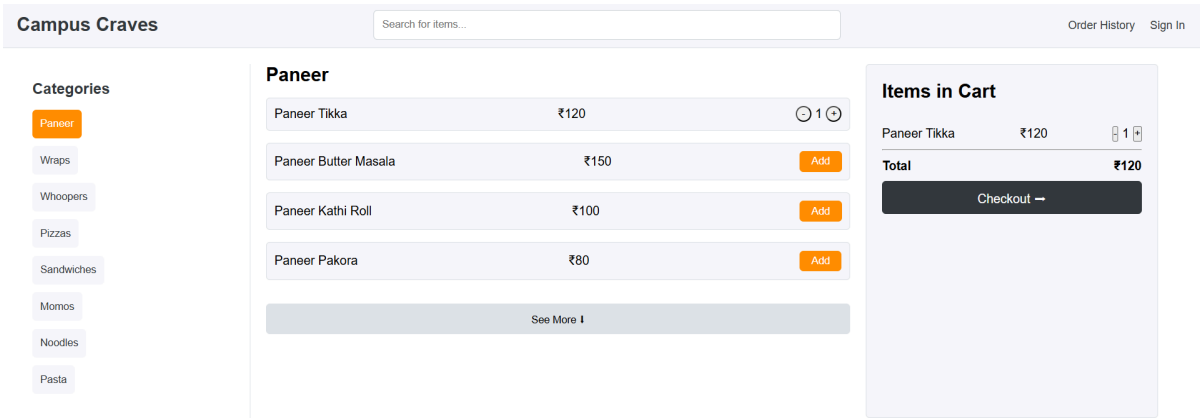
- **Categories View (in Seller's Dashboard):** The "Categories" view enables sellers to display and manage the different product or service categories available in their store. Cards will be displayed with information like title and image of each category. Sellers can deactivate or remove a category by clicking the button on the card of the respective category. In addition, sellers can create a new category by filling in the title, description, and image in the right-hand menu. When the "Create" button is clicked, the newly added category will be shown in the middle section among other categories.



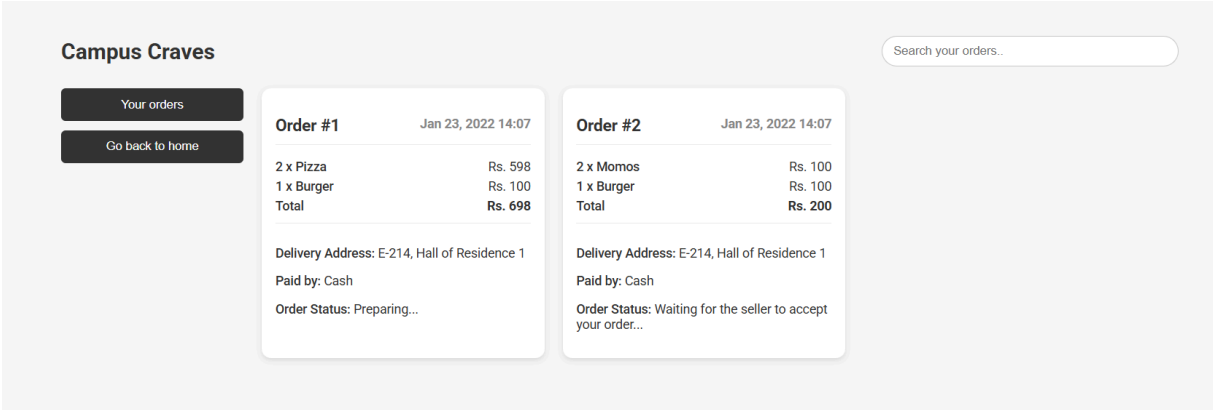
- **Products View (in Seller's Dashboard):** In this view, sellers can manage the products available in their store. The product details are shown in a list format. The "In Stock" button can be toggled to indicate whether a specific product is available or not. They can delete a product by clicking the trash icon in the last column of the product's row. To add a new product, sellers can input product details such as title, category, and price in the text fields provided in the right-side menu and click the "Create" button to add it to the list.



- **Buyer's View:** The top menu displays the store name and a search bar. Users can search for categories or products using the search bar. The left-side menu contains the list of available categories, and users can navigate to a specific category by clicking its name. The center of the view shows details of the products in the selected category. Users can add items to the cart, adjust quantities using the "+" or "-" buttons, and use the "See More" dropdown to explore additional items. The right-hand menu shows the status of the current cart: items in the cart, total amount, and a "Checkout" button to place an order.



- **Order History View:** Users can return to the home page by clicking the "Go Back to Home" button in the left-hand menu. This view displays the store name and a search bar to search through previously placed orders. The middle part of the view displays information on previous orders. These include the order number, date and time, items ordered, total amount, delivery address, payment method, and status of the order. Each order has a button at the bottom that can be used to download a digital copy of the order in PDF format.



3.1.2 Hardware Interfaces

There are no specific hardware interfaces involved in the project. Except the following utilities no additional hardware required:

- Hardware required to connect to the internet. For example, Modem, Network Card, Network Connection, etc.
- A device with an active internet connection. It can be a phone, a tablet, a desktop, anything with internet access, and a browser

3.1.3 Software Interfaces

The software requirements for the project include:

- **Web Browser (with Internet Access):** Essential for debugging, testing, and deploying the application during the development phase.

- **HTML, CSS, JavaScript:** Core technologies for developing the frontend, enabling responsive and visually appealing web components.
- **React Js:** To build the user-friendly frontend interface with dynamic and interactive features, enhancing the user experience for both customers and sellers.
- **Django:** A robust backend framework used to handle server-side logic, manage user authentication, and connect the frontend to the database. It ensures seamless communication between different components of the application and provides built-in security features to safeguard data.
- **MongoDB:** Database technology to store and manage application data, such as product listings, user details, and order history.
- **Visual Studio Code (VS Code):** The integrated development environment (IDE) for writing, testing, and debugging code efficiently.

3.2 Functional Requirements

The **Campus Craves** project has been designed to cater to the needs of both customers and sellers, providing a platform for ordering and managing various campus services. Below are the detailed functional requirements of the project:

3.2.1 User Management

- **Sign-Up:**
  - Customers and sellers can create accounts by entering required details like name, email, password, and contact information.
  - Sellers need to provide additional information about their store (e.g., store name, category, location).
- **Sign-In**
  - Users can log in to their accounts using email and password.
  - A role selection (Customer/Seller) is provided to ensure proper access.
- **Forgot Password**
  - Users can reset their passwords through an email verification process.

3.2.2 Customer Functionalities

- **Home Page:**
  - Customers can browse various stores and services offered on the platform.
  - A **global search bar** allows users to search for specific items, services, or stores.
  - Store cards with store details (e.g., Chocolate Room, Domino's) allow customers to navigate to the desired store.
- **Buyer's View:**
  - **Search Products/Categories**: Customers can search for products or categories within a store.
  - **Category Navigation**: Customers can browse products by selecting a category from the left menu.
  - **Product Details**: Customers can view product details, including price, availability, and descriptions. Users can click "See More" to expand the product list.
- **Cart Management**:
  - Customers can add products to the cart, adjust quantities with "+" or "-", and view the cart summary on the right panel.

    - The total amount and checkout button are displayed for finalizing orders.
- **Order History:**
    - Customers can view the details of past orders, including order number, date/time, delivery address, payment method, and status.
    - A search bar allows users to filter their order history.
    - Users can download a digital copy of the order in PDF format.

3.2.3 Seller Functionalities

- **Dashboard:** Sellers can access their dashboard to manage orders, categories, and products.
- **Orders View**
    - **New Orders**: View details (amount, delivery address, payment method) and accept/reject the order.
    - **Accepted Orders**: Update the order status (e.g., Preparing, Dispatched, Delivered).
    - Sellers can view complete order details, including payment methods, delivery address, and total bill.
- **Categories View:**
    - Sellers can view all categories available in their store.
    - Enter category title, description, and image to create a new category.
    - Disable or delete existing categories using the corresponding buttons.
- **Products View:** Sellers can manage their products through a detailed list view:
    - **In-Stock Toggle**: Update stock availability for products.
    - **Add New Product**: Enter details like title, category, and price in the form.
    - **Delete Product**: Remove products from the list using the delete button.

3.2.4 Cart and Checkout

- **Cart Management**
    - Customers can manage their cart by adding/removing items or adjusting quantities.
- **Checkout**:
    - Customers can review their cart, view the total amount, and place an order.
    - Payment methods and delivery address are entered during checkout.

3.2.5 Notification System

- **Order Updates**:
    - Customers receive notifications when their order status changes (e.g., Accepted, Dispatched, Delivered).
    - Sellers are notified of new orders in real-time.

3.2.6 Payment Integration

- Multiple payment options, such as UPI, credit/debit cards, and cash on delivery, are supported.
- Sellers can view payment methods for each order.

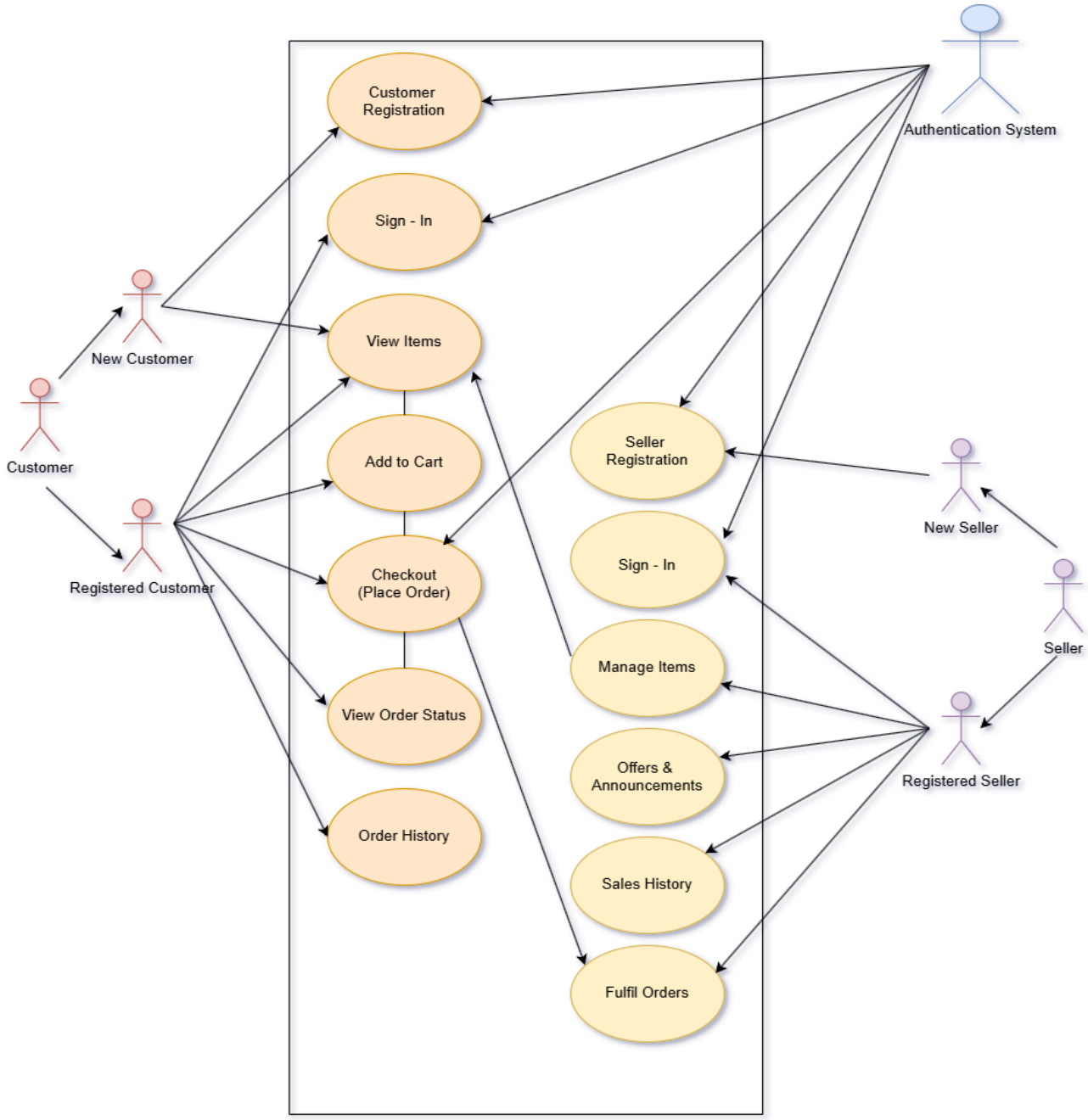
3.2.7 Reporting and Analytics (Seller-Specific)

- Sellers can access a report of total sales, most sold products, and order trends to improve their services.

3.2.8 Search Functionality

- A global search bar allows users to find specific items, services, or stores.
- Customers can search their order history or products within a store.

3.3 Use Case Model



3.3.1 UC1: Seller Registration and Catalogue Management

Author	Siddharth Pathak
Purpose	Seller registration and menu/catalogue creation
Requirements Traceability	Shop Name, Shop ID, List of all menu items with descriptions and prices
Priority	High
Pre-conditions	None
Post-conditions	Seller is registered in the database with a unique Shop ID. The catalogue/menu of the shop is visible to customers.
Actors	Seller, Shop's catalogue
Exceptions	Creating a catalogue for a shop that does not exist in the database
Includes	None
Notes/Issues	Ensure the seller is logged in or registered before creating or modifying the menu

3.3.2 UC2: Browse Options

Author	Siddharth Pathak
Purpose	Browse the menu/catalogue of a specific shop and view all available items.
Requirements Traceability	Shop ID, Item details, and price list of products.
Priority	High
Pre-conditions	Shop must have an active and updated menu in the database.
Post-conditions	Customers can view the shop's menu, item descriptions, and prices.
Actors	Customer, Shop menu
Exceptions	Attempting to view the menu of a shop that does not have an active catalogue.
Includes	None
Notes/Issues	None

3.3.3 UC3: Add to Cart

Author	Siddharth Pathak
Purpose	Add item to cart
Requirements Traceability	Item chosen, Shop ID, Product ID
Priority	Medium
Pre-conditions	Shop must be registered and have an ID. Item(s) chosen must be available.
Post-conditions	Cart now contains the required item.
Actors	User's device, Shop's catalogue
Exceptions	Shop requested is not registered. Product is unavailable.
Includes	Use Case #1
Notes/Issues	Items list must be visible. Ensure completeness when adding products to the shop's item list.

3.3.4 UC4: Customer Registration

Author	Siddharth Pathak
Purpose	Register as a buyer using phone number and OTP verification.
Requirements Traceability	Name and phone number of the buyer
Priority	High
Pre-conditions	None
Post-conditions	Users are now registered in the database with their phone number.
Actors	User's device
Exceptions	Phone number already registered or invalid.
Includes	None
Notes/Issues	OTP might not be sent due to any issue. Validate phone number and check if the user is already registered. If so, log in with a password.

3.3.5 UC5: Checkout and Place Orders

Author	Siddharth Pathak
Purpose	Checkout and place an order
Requirements Traceability	Cart items, User ID, Payment details
Priority	High
Pre-conditions	User must have a valid cart and be logged in.
Post-conditions	Order is successfully placed and saved in the database
Actors	User’s device, Shop’s system
Exceptions	Payment failure or invalid cart items.
Includes	Use Case #3
Notes/Issues	Ensure proper handling of payment failures and notify users of issues.

4 Other Non-functional Requirements

4.1 Performance Requirements

4.1.1 Simultaneous Users

- Requirement: The system should be capable of handling a minimum of 500 simultaneous users without performance degradation.
- Rationale: Ensuring the system can support such high traffic users helps maintain smooth user experiences, preventing slowdowns or crashes during peak usage times.

4.1.2 Response Time

- Requirement: The average response time for any operation (e.g., adding to the cart, placing an order) should not exceed 2 seconds.
- Rationale: Maintaining a quick response time is essential for providing a seamless and efficient user experience, minimizing wait times and preventing user frustration.

4.1.3 Batch Process Efficiency

- Requirement: Batch processes, such as updating product inventory, must complete within 15-20 minutes.
- Rationale: Quick and efficient batch processing is critical to ensuring that the system remains up to date with real-time data, such as inventory changes, without disrupting overall system performance.

4.1.4 Scalability:

- Requirement:
    - The system architecture must scale to accommodate a 100% increase in user traffic during peak times.
    - Adding new stores or categories should require no more than 1 hour of downtime.
- Rationale: The system must be able to handle increased traffic during busy periods without performance degradation and allow seamless updates, such as adding new stores or categories, to ensure continuous operation with minimal disruptions.

4.2 Safety and Security Requirements

4.2.1 User Authentication

- Requirement: All users must complete a secure authentication process using a username and password to access the system.
- Rationale: Ensuring secure user authentication protects accounts from unauthorized access, safeguarding the confidentiality and integrity of user data.

4.2.2 Privacy

- Requirement: User information must not be shared in any form without explicit user consent.
- Rationale: Prioritizing privacy builds trust and ensures compliance with data protection regulations by safeguarding user information.

4.2.3 Data Encryption

- Requirement: All data must be encrypted both during transit and while at rest in the servers.
- Rationale: Ensuring data in transit prevents unauthorized access during transmission, while encryption at rest protects stored data from breaches, ensuring data security and compliance with privacy standards.

4.2.4 Security Audit

- Requirement: Keep an audit trail of user activities and other actions performed.
- Rationale: Regular audits help proactively detect and mitigate potential security risks, ensuring the system remains secure and resilient against breaches.

4.3 Software Quality Attributes

4.3.1 Reliability

- Requirement: The system should ensure a reliability rate of 99.9% for processing and confirming meal bookings.
- Rationale: High system reliability guarantees minimal downtime, ensuring continuous access for users.

4.3.2 Cross-Platform Accessibility

- Requirement: Ensure compatibility with all modern browsers (Chrome, Firefox, Edge, and Safari). Also be accessible through both Android and iOS devices.
- Rationale: Cross-platform compatibility ensures the system can be accessed by a wide range of users, improving reach.

4.3.3 Intuitive User Interface

- Requirement: The user interface must be intuitive and easy to use.
- Rationale: An intuitive and simple interface enhances user experience, reduces the resistance to use the app.

5 Other Requirements

5.1 Legal data requirements: User data collected via the website should be stored in the country's regional data center. This is to comply with data protection and privacy rules that may be enforced by the Government of the nation.

5.2 Internationalization requirements: All strings and locale-specific references from the code should be maintained in a resources file which creates the possibility of using other languages in the future.

Appendix A – Data Dictionary

User Class

Element Name	User
Description	Represents the users of the application, including Buyers and Sellers. Each user registers using an email and password, which is later used to log in.
Attributes	1. email: string (Type: Input) 2. password: string (Type: Input)
Operations	1. register(): Registers a new user in the system. 2. login(): Authenticates a user to log in.
Relationships	Users can either be Buyers or Sellers. The User class is the parent class for both.

Buyer Class

Element Name	Buyer
Description	Represents potential end customers (Buyers) who can browse and purchase products from Sellers in the application.
Attributes	1. customerName: string (Type: Input) 2. phoneNumber: string (Type: Input)
Operations	1. getShoppingCartByStoreID(): Retrieves the shopping cart associated with a specific store.
Relationships	The Buyer class inherits from the User class. It has a one-to-many relationship with the ShoppingCart and Order classes.

Seller Class

Element Name	Seller
Description	Represents business owners (Sellers) who catalog their products for Buyers in the application.
Attributes	1. businessName: string (Type: Input) 2. contactNumber: string (Type: Input)
Operations	1. addProduct(): Adds a new product to the catalog. 2. manageOrders(): Manages incoming orders from Buyers.

Relationships	The Seller class inherits from the User class. It has a one-to-many relationship with the Store and Order classes.

Shopping Cart Class

Element Name	Shopping Cart
Description	Contains items added by Buyers for purchase. Each cart is associated with a Buyer ID and Store ID.
Attributes	1. buyerID: string (Type: Foreign Key) 2. storeID: string (Type: Foreign Key)
Operations	1. checkout(): Processes checkout for the cart. 2. calculateCartValue(): Calculates the total value of items in the cart. 3. addCartItem(): Adds an item to the cart. 4. getItems(): Retrieves the items in the cart.
Relationships	Many-to-one relationship with the Buyer and ShoppingCartItem classes.

Order Class

Element Name	Order
Description	Represents an order placed by a Buyer. Contains information such as Buyer ID, Store ID, payment status, and shipping details.
Attributes	1. buyerID: string (Type: Foreign Key) 2. storeID: string (Type: Foreign Key) 3. shippingInfo: string (Type: Input) 4. currentStatus: string (Type: State Variable) 5. paymentStatus: string (Type: State Variable) 6. amount: int (Type: Input)
Operations	1. updateOrderStatus(): Allows the Seller to update the order status.
Relationships	Many-to-one relationship with Buyer, Seller, and OrderItem classes.

Store Class

Element Name	Store
Description	Represents a store owned by a Seller, offering products to Buyers. Includes the store's name, address, and contact information.
Attributes	1. name: string (Type: Input)

	2. address: string (Type: Input) 3. contact: string (Type: Input) 4. imageURL: string (Type: Input)
Operations	1. addCategory(): Allows the Seller to add a category to the store.
Relationships	One-to-one relationship with the Seller class. One-to-many relationship with the Category class.

Category Class

Element Name	Category
Description	Represents product categories in a store. Each category contains products related to that category.
Attributes	1. title: string (Type: Input) 2. description: string (Type: Input) 3. imageURL: string (Type: Input)
Operations	1. addProduct(): Adds a product under the category.
Relationships	Many-to-one relationship with the Store class. One-to-many relationship with the Product class.

Product Class

Element Name	Product
Description	Represents products available in a store. Each product has a title, description, price, and availability status.
Attributes	1. title: string (Type: Input) 2. description: string (Type: Input) 3. price: int (Type: Input) 4. available: bool (Type: State Variable)
Operations	1. updatePrice(): Allows the Seller to update the product's price. 2. updateAvailability(): Allows the Seller to mark the product as available/unavailable.
Relationships	Many-to-one relationship with the Category class.

Admin Class

Element Name	Admin

Description	Represents the Administrator who provides technological and product support to Buyers and Sellers.
Attributes	1. adminID: string (Type: Input) 2. name: string (Type: Input)
Operations	1. monitorLogs(): Reviews error and order logs for debugging and auditing. 2. manageUsers(): Manages user accounts and permissions.
Relationships	The Admin oversees and supports all Users (Buyers and Sellers).

Appendix B - Group Log

Since the beginning of the project, our entire team has been very enthusiastic. We have formed a Whatsapp group for effective communication. We have created a private repository on GitHub for collaborative software development.

Meeting Time	**Agenda/Discussions**
10 January 2025 (21:00 - 22:00 IST)	Brainstormed different project ideas and finalised 3 ideas that are web application for connecting Canteens and Grocery store, IITK Resources Management, Study Place Availability Tracker
11 January 2025 (16:30 - 17:00 IST)	After the discussion with course Instructor, finalize the idea of web application for connecting Canteens and Grocery stores and named it “Campus Craves”
16 January 2025 (21:00 - 22:00 IST)	Discussed Project requirements, structure and initiate the work of Software Requirement Specification document.
20 January 2025 (21:00 - 23:30 IST)	Worked on a draft of Software Requirement Specification document.
24 January 2025 (10:00 - 11:00 IST)	Meeting with TA. Discussed about different frameworks that can be utilized in the project (pros and cons) and updation in the draft of SRS document.