Software Requirements Specification

for

Credit Snap

Version 1.0

Prepared by

Group 6 : **Group Name:** ***Bug Dealers***

Kasavajjala Sai Shreyas	**240530**	ksshreyas24@iitk.ac.in
Gade V S S R K Tejas Reddy	**240390**	tejasreddy24@iitk.ac.in
Palagiri Sathish Reddy	**240717**	psreddy24@iitk.ac.in
Kondreddy Sai Haneesh Reddy	**240551**	haneesh24@iitk.ac.in
Yerraiahgari Ram Charan Goud	**241212**	ramcharang24@iitk.ac.in
Golla Lekha Harshaa	**240402**	lekhahg24@iitk.ac.in
Yash Raj	**241202**	yashr24@iitk.ac.in
Korada Jayavardhan	**240554**	jkorada24@iitk.ac.in
Boppudi Sai Chaitanya	**240283**	boppudis24@iitk.ac.in
Panchaneni Ashwin Rao	**240727**	ashwinr24@iitk.ac.in

Course: **CS253**

Mentor TA: **George T L**

Date: **January 24, 2026**

CONTENTS

REVISIONS

Version	Primary Author(s)	Description of Version	Date Completed
v1.0	Kasavajjala Sai Shreyas Gade V S S R K Tejas Reddy Palagiri Sathish Reddy Kondreddy Sai Haneesh Reddy Yerraiahgari Ram Charan Goud Golla Lekha Harshaa Yash Raj Korada Jayavardhan Boppudi Sai Chaitanya Panchaneni Ashwin Rao	First version of the Software Requirement Specification (SRS) Document	24/01/26

1. INTRODUCTION

1.1 Product Scope

The project aims to modernize the traditional operations of campus hall canteens by digitizing manual sales and credit ledger systems. Currently, reliance on physical notebooks leads to calculation errors, long queues, and inefficiencies during peak hours. Credit Snap addresses these issues by providing a unified web platform with two primary interfaces: a student portal for browsing menus, placing digital orders, and availing credit facilities, which reduces counter wait times; and an administrative dashboard for canteen owners to manage incoming orders and efficiently monitor financial records.

A key focus of the platform is the “Credit Ledger” system, which automates credit management by maintaining transparent records of student debts and enforcing dynamic limits to prevent financial discrepancies. Additionally, the system empowers canteen owners with an analytics suite that visualizes revenue data, transforming raw transaction logs into actionable business insights. By bridging the gap between digital ordering and financial management, Credit Snap ensures a seamless, transparent, and error-free dining experience for the entire campus community.

1.2 Intended Audience and Document Overview

1.2.1 Intended Audience

- **Users:**
  The primary customers of the software consist of IITK Hall Canteen Owners and Students.
- **Software Developers:**
  Who will design and implement the platform as per the requirements specified in this document, primarily the group members.
- **Project Managers:**
  Who will supervise the planning and execution of the software development procedure, specifically the TAs and the course instructor.
- **Testers and Approvers:**
  Who will perform quality checks on the designed software and provide feedback on the interface, areas for improvement, etc., primarily the TAs , the course instructor and other teams who intend to beta-test the application

1.2.2 Document Overview

1. **Revisions:** This section contains information about the various versions that this document has gone through.
2. **Introduction:** This section provides basic information useful for reading the SRS, such as document conventions, abbreviations, etc. The reader may choose to skip the section if they are familiar with the basic terminologies. In any case, this section will serve as a helpful collection of information to clarify any confusion that may occur while reading the document.
3. **Overall Description:** This section offers an overall view of the software system and its functionalities, assumptions, and dependencies. This will be a useful read for those seek-

ing to familiarize themselves with the system at a quick glance. A reader is encouraged to read this part as it provides a good basis for understanding the next section of the SRS.

4. **Specific Requirements:** This section is the core of the document, detailing functional requirements and use case models.Functional Requirements describes the specific features and functionalities,such as editing the menu,pre ordering before coming to canteen.Use case models include user scenarios and workflows to demonstrate how the system behaves in different situations.
5. **Other Non-Functional Requirements:** Important non-functional requirements are expounded here such as; Specifies performance metrics, safety and security needs, and software quality attributes like scalability and reliability.

1.3 Definitions, Acronyms, and Abbreviations

API	Application Programming Interface
Billing Cycle	Rigid time intervals defined by Canteen owners for credit settlement
Debt Ceiling	The maximum monetary limit of credit allowed to a particular student
HTTPS	Hypertext Transfer Protocol Secure
IITK	Indian Institute of Technology Kanpur
JWT	JSON Web Token
On-spot Ordering	Order placed by a student at the canteen counter
Pre ordering	Digital request generated by a student before coming(10 to 40 minutes prior) to the canteen counter
QR Code	Quick Response Code
SRS	Software Requirements Specification
UI	User Interface

1.4 Document Conventions

- The document is written using LaTeX (LaTeX)to ensure the consistency of the document and to make it easier to maintain.
- The document is written with the **Arial** font of size 11 with 1-inch margins.
- Bullet point ordering has been used as a listing typesetting tool.
- Important keywords are highlighted in bold font and comments are italicized.
- Whenever we use the term ”Canteen Owner” in this document we mean all the registered IITK Hall Canteen Owners and other food stall owners in IITK campus who are using our website.

1.5 References and Acknowledgments

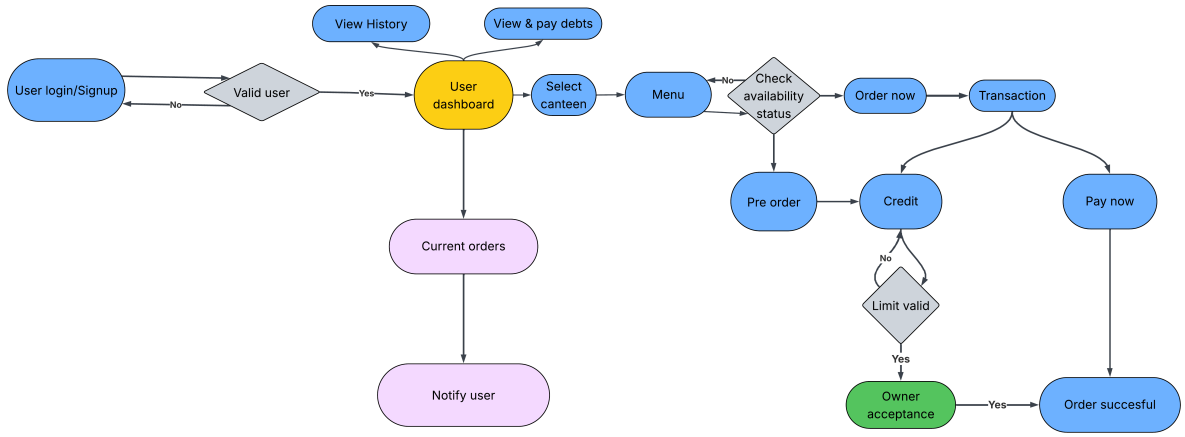
- We would like to acknowledge the help of our TA, Mr. George TL, and our course instructor, Prof. Indranil Saha, for guiding us through the document and providing a template for the Software Requirements Specification document.
- We utilised Figma, Canva and Lucidchart to craft visually compelling graphs and flowcharts, effectively translating our ideas into a concise and impactful pictorial representation.

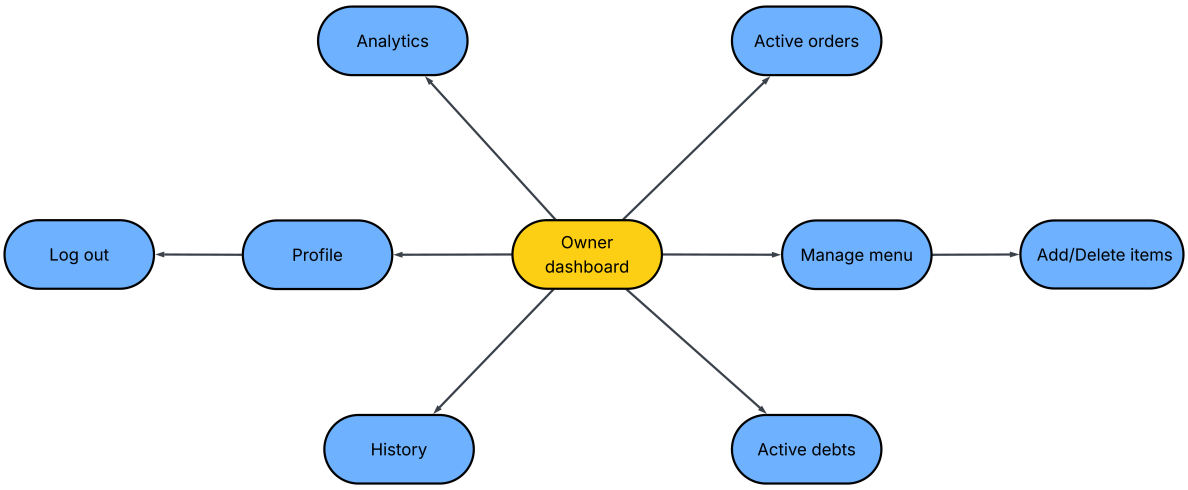
2. OVERALL DESCRIPTION

2.1 Product Overview

CreditSnap is a specialized, self-contained full-stack web solution designed to modernize and streamline the credit management workflows within campus canteens. It replaces antiquated, paper-based manual bookkeeping with a centralized digital platform, effectively bridging the trust gap between canteen owners and students by providing clear transparency into personal debt records. The software operates independently as a central hub that facilitates the authorization of credit requests of students.

The software utilizes an efficient digital system for the storage and retrieval of student credit ledgers and real-time item availability. This centralized approach empowers canteen owners to manage their credit operations with high accuracy and financial oversight, while providing students with a secure interface to monitor their spending and credit status.





2.2 Product Functionality

This product acts as a centralized credit management system for campus canteens, providing specific administrative tools for owners and transparent access for students. It includes the following functionalities:

User Management

1. **Role-Based Login:** Provides distinct interfaces and permissions for Students and Canteen Owners.
2. **Registration & Security:** Includes a verification link generation which is sent to registered email addresses for secure account creation.
3. **Account Recovery:** Users can recover their account using the forgot password tool.

Canteen Owner Management Tools

1. **Credit Parameter Setting:** Canteen Owners can set rigid financial rules for each student, including Debt Ceilings (e.g., 2,500) and specific Billing Cycles.
2. **Real-Time Menu Control:** Canteen Owners can toggle item availability to “unavailable” to prevent out-of-stock item requests, add any item to the menu and also increase the price for any item present in the menu.
3. **Request Authorization:** This interface for the canteen owners is to accept or reject the student credit request.
4. **Digital Ledger Maintenance:** Enables canteen owners to view and manage student debt records, including the ability to manually clear outstanding balances when payments are made outside the website.
5. **Analytics:** Enables Canteen owner to view the analytics of Credit managed during the previous months.

Student Credit & Requesting Features

1. **Canteen Selection & Browsing:** Students select a specific canteen from a list to view its active items.
2. **Payment Mode Selection:** Students must choose between “Add to Credit” (Credit Request) or “Pay Now” (Digital Payment).
3. **Automatic Debt Validation:** The system blocks credit requests if the student’s current debt exceeds the pre-set ceiling.
4. **Transparency Hub:** A UI for students to view their total debts and canteen-wise breakdown.

Monitoring & Alerts

1. **Live Status Tracking:** Real-time visibility of request progress through “Pending,” “Accepted,” and “Rejected” stages.
2. **Automated Notifications:** Students receive alerts when an owner accepts or rejects a request.
3. **Financial Oversight:** Owners can view “Total Outstanding Debt” and “Overdue Accounts” through a specialized dashboard.

4. **Automated History Archiving:** Upon successful debt clearance, the student's transaction record is automatically moved from the "Active Debts" list to the "History" archive in the Owner's dashboard, ensuring the active ledger remains clutter-free.

2.3 Design and Implementation Constraints

Based on the technical requirements for campus financial systems, CreditSnap is subject to several design and implementation constraints that limit developer options and define the operational environment.

Memory and Timing Requirements

- The application's memory usage must be highly optimized, particularly when processing large datasets within the database.
- Efficient memory management is mandatory to prevent performance bottlenecks and ensure smooth operation during high user activity periods.
- Users expect quick response times; any significant delays in loading pages or processing credit authorization requests will lead to poor user experience.
- The system must be designed to efficiently handle and store data related to debt ledgers and user profiles as the database grows over time.

Parallel Operations and Concurrency

- **High Traffic Handling:** The system uses Node.js, which is designed to efficiently handle thousands of connections at once. This ensures that even during busy lunch hours, multiple students can place orders simultaneously without the server crashing or slowing down.
- **Background Tasks:** Time-consuming tasks, like sending order confirmation emails via NodeMailer, run in the background. This means the user doesn't have to wait for the email to be sent before seeing their "Order Successful" screen—the system stays fast and responsive.

Security Considerations

- **Institute Restriction :** Registration is restricted to Institute students through **mandatory IITK email-based authentication.**
- **Secure Logins:** User sessions are managed using **JSON Web Tokens (JWT)**. This is a modern, secure way to keep a user logged in without storing sensitive session data directly on the server, making the system faster and safer.
- **Password Safety:** We never store actual passwords. Instead, we use **bcrypt.js** to "hash" (scramble) the passwords before saving them. Even if someone steals the database, they cannot read the user's actual password.
- **Transaction Security :**All transactions must be processed through a secure payment gateway to ensure the encryption and protection of financial information.
- **Safe Communication:** All data flowing between the student's phone and our server is **encrypted using HTTPS,** ensuring that no one on the campus Wi-Fi can intercept private details like payment info.

Programming Conventions and Standards

- **Organized Code Structure:** The development team shall follow a strict structural hierarchy, keeping the user interface (Frontend) logic completely separate from the server-side (Backend) logic. This separation ensures that changing the look of the website does not accidentally break the core business functions, like calculating debts.
- **Readability and Comments:** All source code will be written with clarity as a priority. Developers are required to include clear, English-language comments explaining complex logic .This ensures that future students or instructors can easily read and understand the project logic.
- **Consistent Naming:** The team will adhere to a consistent naming convention for all variables and files

2.4 Assumptions and Dependencies

2.4.1 Assumptions

- **User Connectivity:** It is assumed that all users have access to a stable and reliable internet connection while interacting with the platform to ensure real-time credit updates and authorization.
- **Registration Credentials:** All users are assumed to possess a valid campus email address and phone number for secure account verification.
- **Hardware Availability:** It is assumed that canteen owners possess a desktop, mobile device or tablet to manage incoming requests and toggle item availability in real-time.
- **Single User Access:** It is assumed that each particular account will be used by only one person at a time to maintain account security and transaction integrity.
- **User Accountability:** Students and canteen owners are assumed to be held accountable for the accuracy of their profiles and secure use of their credentials.
- **Scalability:** The total number of students is expected no to exceed 15,000 users across the campus .

2.4.2 Dependencies

- **React Library:** The frontend interface is dependent on the React ecosystem for building a responsive, component-based user architecture that handles dynamic updates.
- **Node.js & Express.js:** The project relies on the Node.js runtime and Express.js framework to provide a scalable server-side environment for managing API requests.
- **MongoDB Database:** The system depends on MongoDB as its primary NoSQL database for the flexible storage of Digital credit ledgers and transaction histories.
- **Socket.io:** The platform is dependent on Socket.io to enable real-time, bi-directional communication for immediate order status notifications.
- **JWT (JSON Web Tokens):** The security architecture depends on JWT to facilitate secure, token-based authentication and authorized login sessions.
- **Nodemailer:** The system relies on the Nodemailer module to handle the automated delivery of emails, such as verification link for password resets.
- **RESTful APIs:** The project utilizes standard API protocols to facilitate seamless data exchange between the frontend and the backend server.

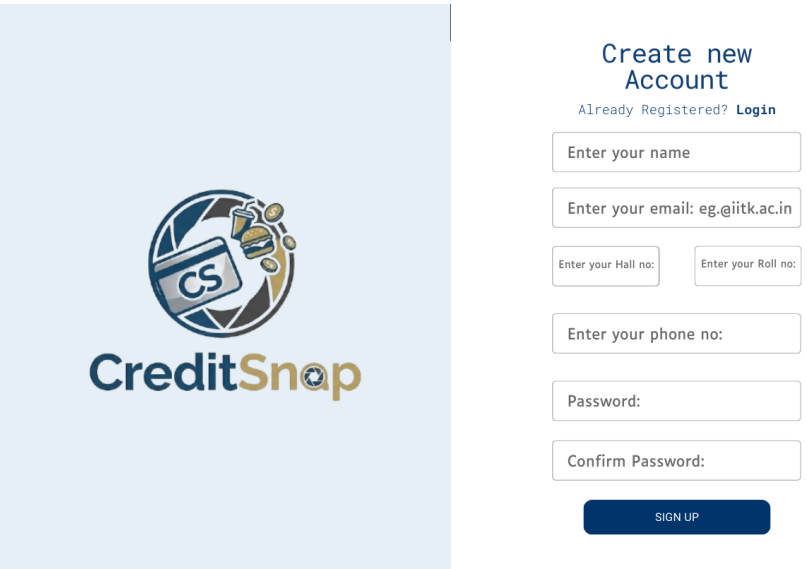
3. SPECIFIC REQUIREMENTS

3.1 External Interface Requirements

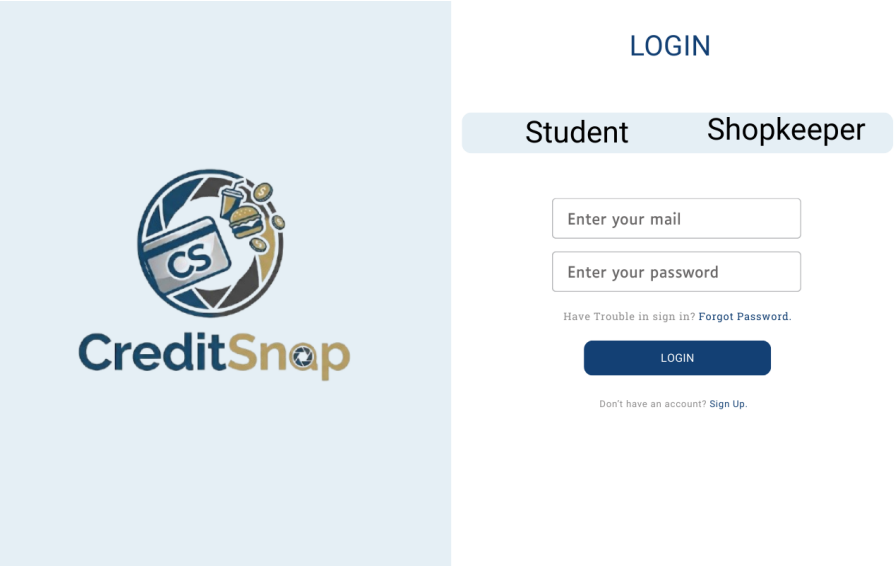
3.1.1 User Interfaces :

- The system provides a role-based login for Canteen Managers and Students. New users must complete a registration process followed by a mandatory verification step to gain access.

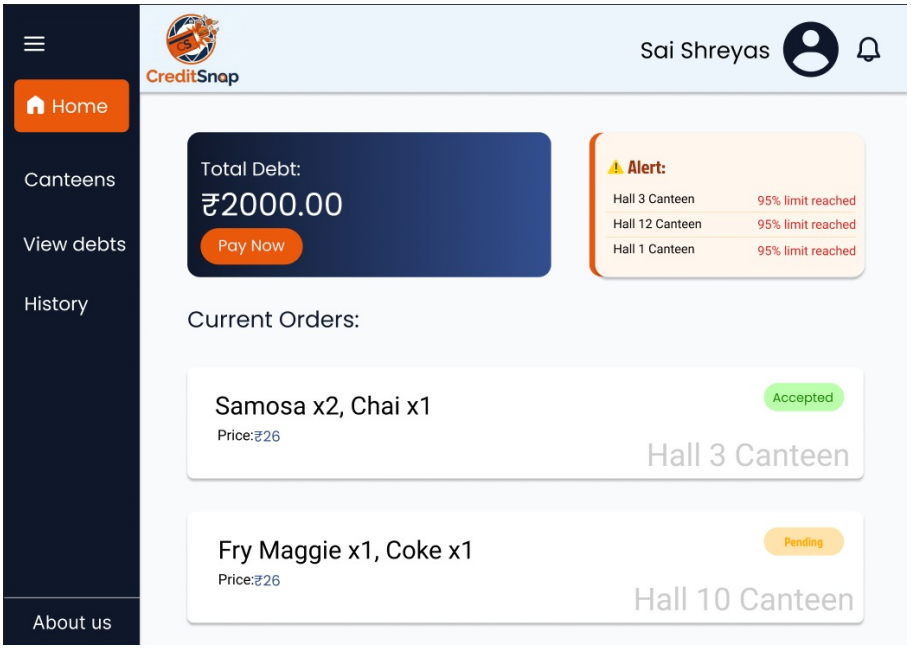
- **Create a new account page**



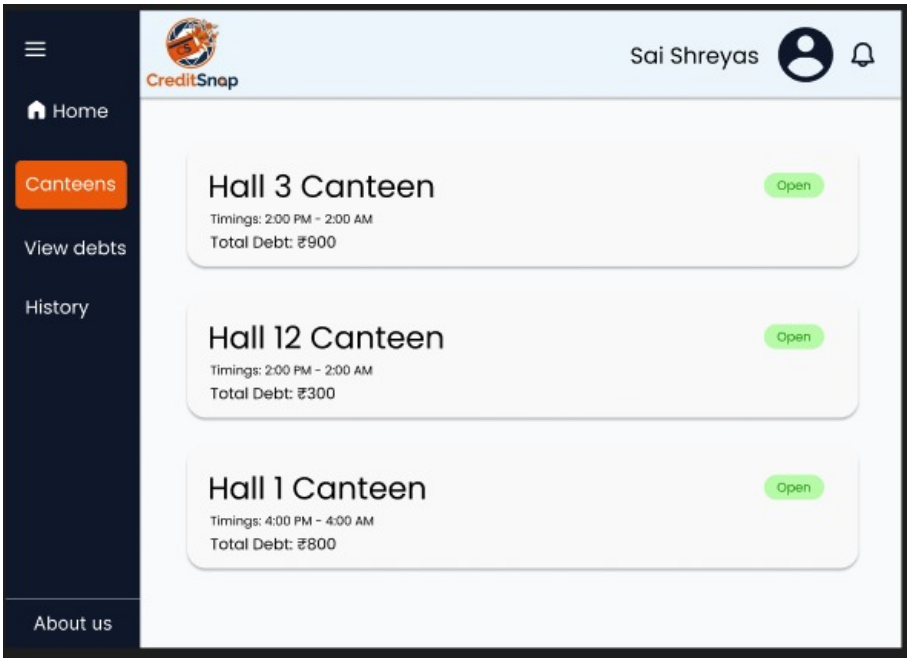
- **Login Page**



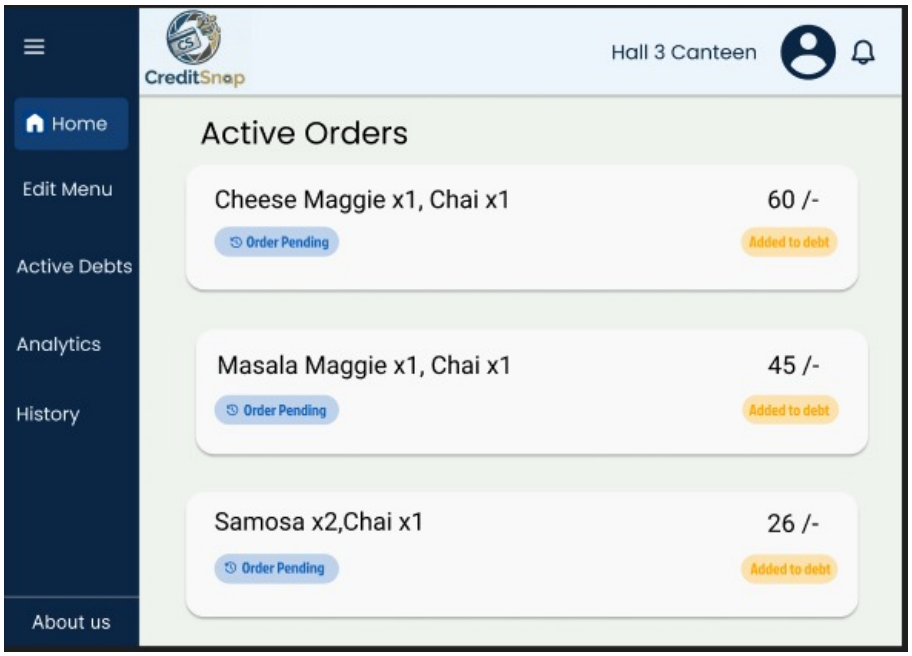
- The home screen for students features an order card that shows real-time status updates of current food order like order accepted,rejected or pending.The total outstanding debt of student is also shown here along with Alerts showing credit limits. Additionally, a notification bell in the header also alerts users about credit warnings.
- **Student Home Page**



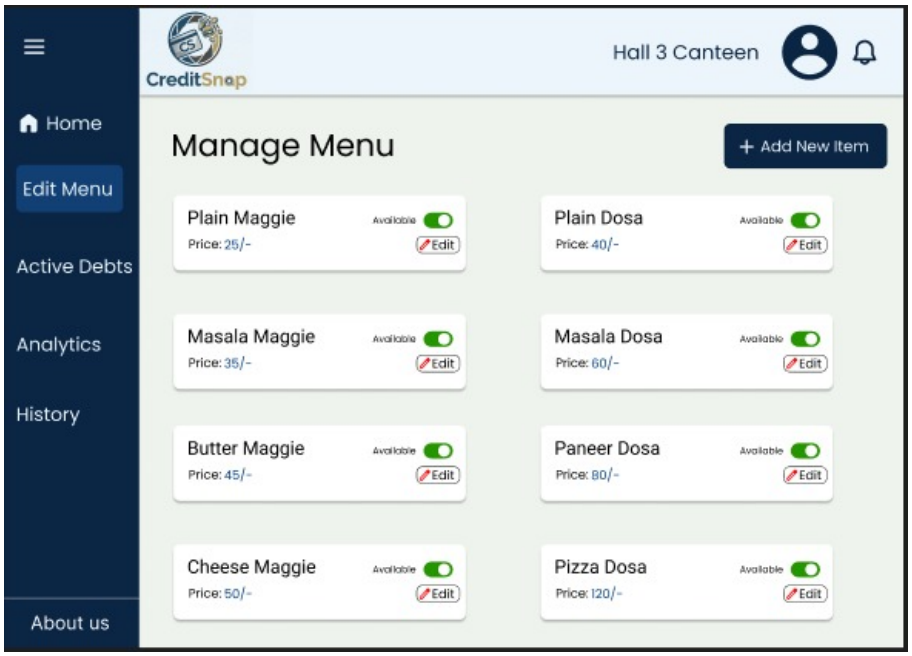
- The Canteen Selection interface serves as a real-time directory, allowing students to browse all active campus eateries while maintaining a clear overview of their debt balance at each location. Upon selecting a specific canteen, the system transitions to a dynamic menu view where items are filtered by current availability, price, Veg/Non-Veg items.
- **Student's Canteen page**



- The Shopkeeper Home Page serves as a live Active Orders dashboard, allowing canteen owners to monitor incoming requests and manage their status in real-time. Each card provides a summary of the order, the total cost, and a status badge indicating the status of order.This centralized hub digitizes traditional paper-based bookkeeping, providing owners with the financial oversight needed to manage student credit requests accurately.
- **Shopkeeper home page**



- The Manage Menu interface gives owners total control to add new items or modify existing ones. Owners can instantly toggle an item's "Available" status or update its price, which syncs immediately with the student-facing menu. This real-time synchronization ensures students only order currently available items, significantly reducing order rejections.
- **Shopkeeper's Edit Menu page**



3.1.2 Hardware Interfaces :

- **Student Devices :** The platform is a web-based application accessible on any device with internet, such as smartphones, tablets, and laptops. This ensures that students can place orders or check their debt status from anywhere on campus, making the process hassle-free.
- **Canteen Manager Devices :** Canteen owners can manage the platform using a Desktop, mobile device or tablet. A larger screen is recommended for the Owner Dashboard to easily view the live order queue, manage credit ledger.

3.1.3 Software Interfaces :

- **Web Browser Support:** The website is designed to run smoothly on all standard web browsers. It does not require users to install any special software or plugins—just a working browser.
- **Operating Systems:** Since the system is web-based, it works on any device with an internet connection.
- **External Payment Gateway :** To handle online money transfers, the system connects to a trusted Third-Party Payment Service (like Razorpay). When a student chooses to pay online, the system hands over the transaction details to this service securely and waits for a ”Success” or ”Failure” confirmation.
- **Database System:** The application connects to a secure Database Server to store all critical information, such as Students' data, Canteen Menus,Transactions History and Order History. This ensures that data is saved permanently and can be retrieved whenever a user logs in.
- **Email Service:** The system connects to an Email Server to automatically send important notifications. This allows the application to email students when they register effectively or when their order is confirmed, without a human needing to send it manually.

3.2 Functional Requirements

This section specifies the detailed functional requirements of the *Credit Snap* system. These requirements describe the behavior of the system from the perspective of both students and canteen owners and define how the platform shall respond to user actions. The functional requirements ensure correct authentication, secure order processing, transparent credit handling, and efficient canteen management. Each requirement is designed to minimize manual intervention while maintaining accountability and real-time synchronization between users.

3.2.1 Authentication and Authorization Functionalities

- The system shall provide a **secure authentication mechanism** to ensure that only authorized users are allowed to access the platform.
- The system shall allow **student sign-up only**, while canteen manager accounts shall be **created, verified, and maintained exclusively by the system administrator**.
- The system shall restrict student registration strictly to **valid IIT Kanpur email addresses (@iitk.ac.in)** to prevent unauthorized access from non-campus users.

- During the registration process, the system shall require students to provide mandatory details including:
  - Full name
  - Institutional email address
  - Roll Number
  - Hall Number
  - Phone number
  - Password
- The system shall enforce **basic password security rules**, such as minimum length and non-empty credentials, to prevent weak authentication.
- Upon successful submission of the registration form, the system shall **generate a verification link** and send it to the registered email address.
- The system shall activate the student account **only after successful email verification**, ensuring that fake or incorrect email addresses are not registered.
- The system shall store authentication credentials in a **secure and encrypted format** to protect user data.
- The system shall provide **separate login interfaces** for students and canteen owners to ensure role-based access control.
- The system shall validate login credentials and redirect users to their respective dashboards upon successful authentication.
- The system shall provide a **"Forgot Password" functionality**, allowing users to reset their passwords using email-based verification.
- The system shall invalidate old passwords after a successful reset and update the authentication database accordingly.

3.2.2 Student (Customer) Functionalities

Canteen Browsing and Menu Viewing:

- The system shall allow students to **browse a list of all active campus canteens** available on the platform.
- The system shall display basic canteen information such as name, List of items, and student-specific outstanding debt (if any).
- Upon selecting a canteen, the system shall display a **dynamic menu interface** containing all available food items.
- The system shall display each menu item with relevant details including:
  - Item name
  - Price
  - Availability status
- The system shall ensure that **unavailable items are clearly marked**, preventing students from placing invalid orders.

Order Placement and Payment Selection:

- The system shall allow students to **select one or more items**, specify quantities, and add them to a virtual order cart.
- The system shall calculate the **total payable amount** in real time as items are added or removed.
- The system shall allow students to provide **optional special instructions** related to food preparation.
- Before placing an order, the system shall require students to **select a payment method**, either:
  - Online payment
  - credit-based payment
- The system shall validate the selected payment method before forwarding the order to the canteen.

Credit Management:

- The system shall maintain a **digital credit ledger** for each student, tracking outstanding balances across all canteens.
- Before allowing a Cred-based order, the system shall automatically **check the student's current outstanding debt**.
- The system shall compare the current debt with the **maximum debt limit set by the canteen owner**.
- If the debt exceeds the allowed limit, the system shall:
  - Disable the debt payment option
  - Display a warning message indicating the exceeded limit
- If the student is within the allowed limit, the system shall allow the credit request to be submitted to the canteen owner for approval.
- The system shall update the student's debt ledger **only after the order is approved by the canteen owner**.
- Student can make partial or complete payment to a hall at any time.

Order Tracking and Notifications:

- The system shall allow students to **track the real-time status** of placed orders.
- Order statuses shall include:
  - Pending
  - Accepted
  - Rejected
  - Ready
- The system shall update the order status immediately upon any action taken by the canteen owner.

- The system shall notify students whenever:
  - An order is accepted or rejected
  - An order is marked ready
  - A payment is successfully completed
- The system shall display **credit limit warnings** when a student's debt approaches the maximum allowed limit.

Profile and History Management:

- The system shall allow students to **view and edit their profile information**, including name and contact details.
- The system shall validate profile inputs and prevent saving invalid data.
- The system shall maintain a **complete order history**, allowing students to view previous transactions.
- The system shall allow students to view **payment history and settled debts** for transparency.
- The system shall allow students to securely **log out** of the platform at any time.

3.2.3 Canteen Owner Functionalities

Order Management:

- The system shall notify canteen owners in **real time** when a new order is placed.
- The system shall display all active orders in a centralized dashboard.
- For each order, the system shall display:
  - Ordered items
  - Total cost
  - Payment mode
  - Student identifier
- The system shall allow owners to **accept or reject orders** based on availability and payment validity.
- Upon rejection, the system shall notify the student and update the order status accordingly.

Menu Management:

- The system shall allow canteen owners to **add new menu items** to their canteen.
- The system shall allow owners to **update item prices and availability**.
- The system shall synchronize menu updates in **real time** with the student interface.
- The system shall prevent students from ordering items marked as unavailable.

Debt and Financial Management:

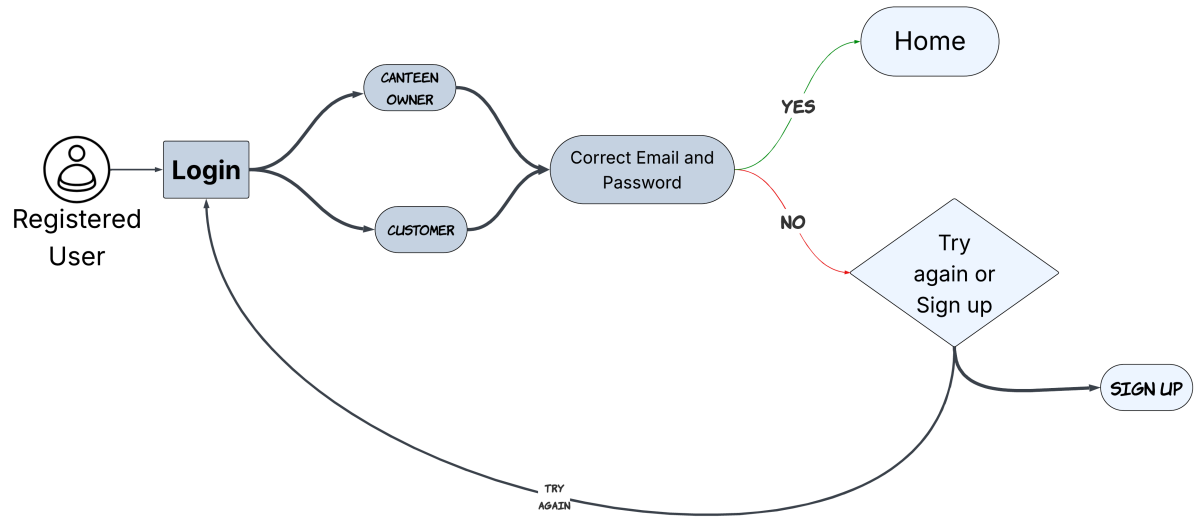
- The system shall allow canteen owners to **set default and student-specific debt limits**.
- The system shall maintain a **canteen-wise digital ledger** of all outstanding debts.
- The system shall allow owners to view:
  - Student-wise outstanding balances
  - Total pending credit
- The system shall allow owners to manually clear debts in cases of offline payments.
- The system shall record all transactions in a **persistent transaction database**.

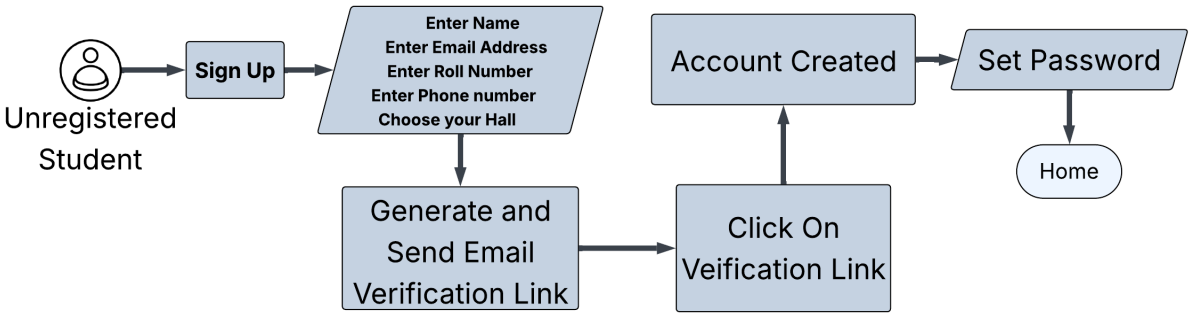
Reports and Account Management:

- The system shall generate **basic analytics and reports** related to orders and earnings.
- The system shall allow owners to view historical transaction records.
- The system shall allow owners to **edit profile and contact information**.
- The system shall allow owners to securely **log out** of the platform.

3.3 Use Case Model

3.3.1 Use Case #1 (Signup/Login)





Author - Yash raj, Sathish

Purpose - This use case is for the login and signup of the user.

Requirements Traceability - Profile page interface, signup, and login interface.

Priority - High.

Preconditions - Valid iitk email for sign-up. Valid iitk email and password for login.

Post Conditions - The user is logged in and able to interact with the server.

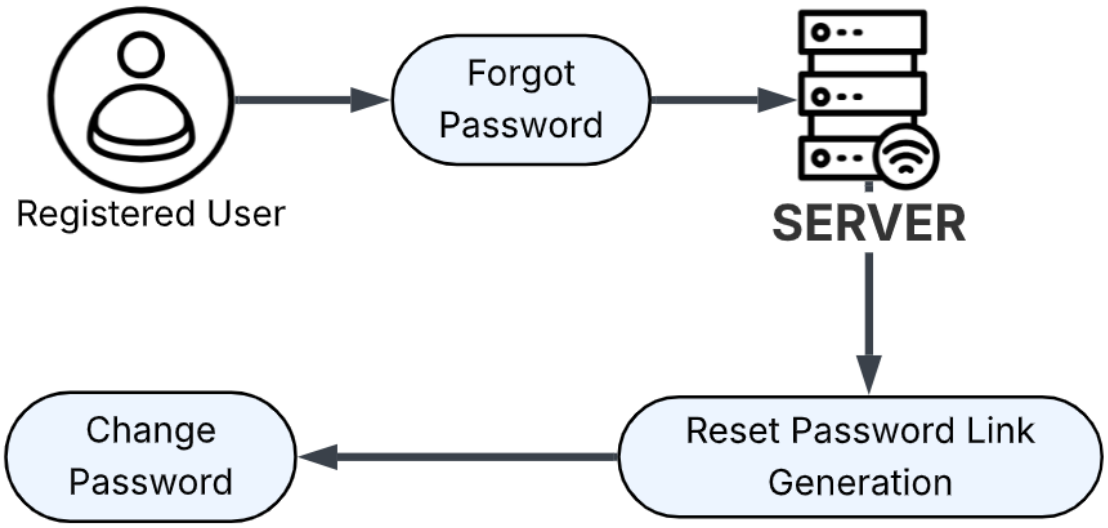
Actors - User of the website, and server.

Exceptions - The user may forget the password and go for the forgot password option.

Includes - Only one account is created for a mail ID

Notes - Here the user will have to enter some necessary details(name, email, password, etc.) during signup/login.

3.3.2 Use Case #2 (Forgot Password)



Author - Jayavardhan

Purpose - This use case is to reset passwords.

Requirements Traceability - Profile page, User Database

Priority - High.

Preconditions - Valid Email-ID of the registered user.

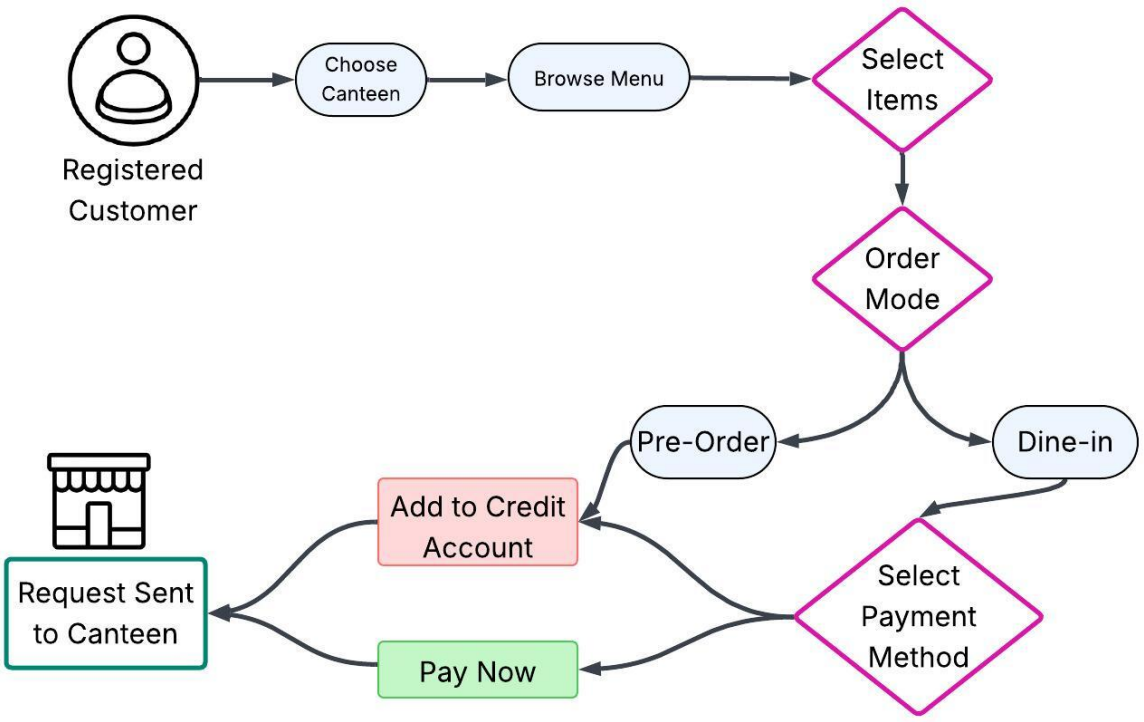
Post Conditions - The user gets a computer-generated reset link to his mail and the User's new password is updated.

Actors - User of the website, and server.

Exceptions - If the user doesn't receive the reset password link,they may go for the resend link option.

Includes - None.

3.3.3 Use Case #3 (Place Order)



Author - Tejas, Yash raj.

Purpose - This use case is to place a canteen order with a choice of payment methods (Credit Account or Immediate Payment).

Requirements Traceability - Menu Page, User Database, Canteen Management System.

Priority - High.

Preconditions - The customer is a registered user and is logged into the platform.

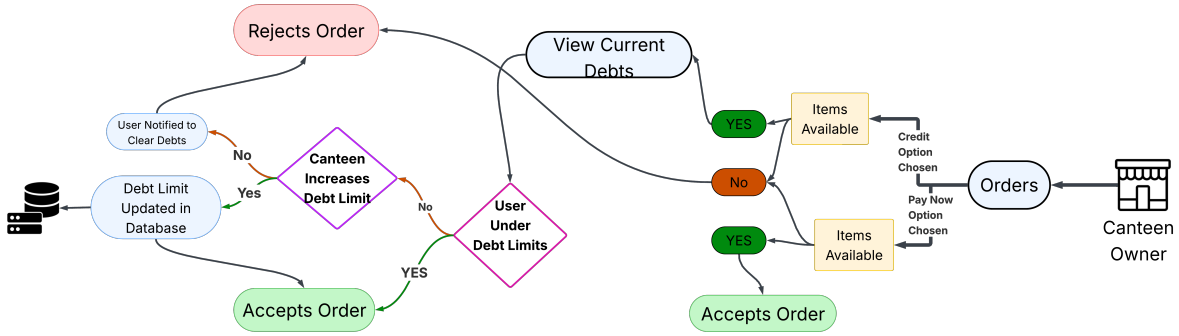
Post Conditions - The order request is successfully sent to the canteen, and the transaction is recorded in either the digital ledger (Khata) or via immediate payment.

Actors - Registered Customer, Canteen System/Owner..

Exceptions - The credit request may be blocked if the user exceeds the pre-set debt ceiling.
An item may be marked as unavailable by the owner during the browsing stage.

Includes - None.

3.3.4 Use Case #4 (Receive Order)



Author - Shreyas, Chaitanya, Lekha harshaa

Purpose - To manage the backend authorization of orders, specifically verifying item availability and validating student credit limits.

Requirements Traceability - Canteen Management System, User Database, Payment Portal.

Priority - Medium.

Preconditions - A registered user has initiated an order from the menu and selected a payment method (Credit or Pay Now).

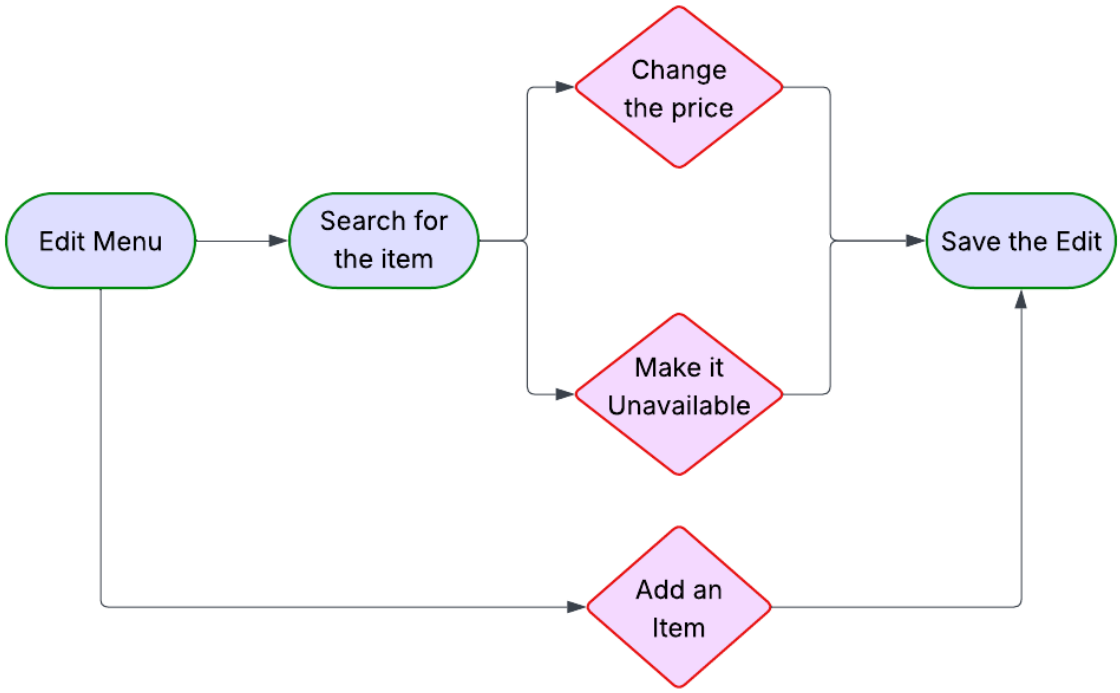
Post Conditions - The order status is updated to “Accepted” or “Rejected,” and the database is updated with new debt levels or payment records.

Actors - Canteen Owner/System, Database, Payment Gateway.

Exceptions - If items are unavailable, the order is automatically rejected.
If the user is over the limit and the owner refuses to increase it, the student is notified to clear existing debts, and the order is rejected.

Includes - None.

3.3.5 Use Case #5 (Edit Menu)



Author - Tejas, Jayavardhan

Purpose -To allow canteen owners to manage their digital menu by adding new products or updating existing item details such as pricing and availability.

Requirements Traceability - Canteen Management Interface, Menu Database.

Priority - Medium.

Preconditions - The Canteen Owner must be logged into their administrative account with valid credentials.

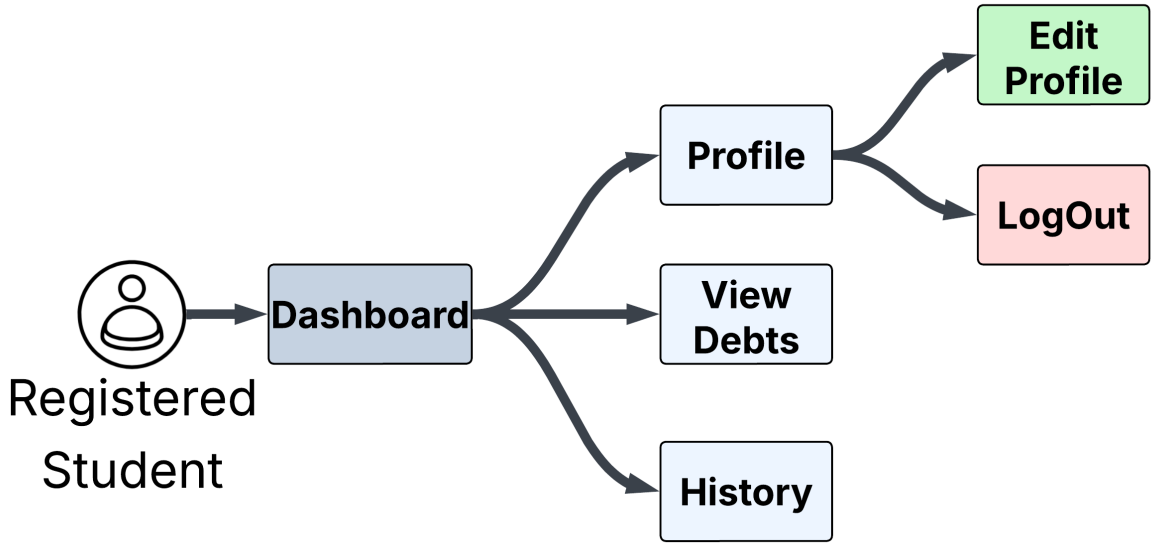
Post Conditions - The menu database is updated, and changes are reflected in real-time for students browsing the canteen menu.

Actors - Canteen Owner/Administrator, Menu Database.

Exceptions - 1) If a search yields no results, the owner must be notified that the item does not exist.
2) If ”Save the Edit” fails due to a network error, the owners should be prompted to retry to ensure the database updates.

Includes - None.

3.3.6 Use Case #6 (Student Account Navigation)



Author - Chaitanya, Haneesh

Purpose -To allow registered students to access and edit their profiles , log out of their accounts, view canteen wise debts, and check the transaction history of their payments and settled debts.

Requirements Traceability - Student Portal Interface, User Account Database , Transactions Database.

Priority - Low.

Preconditions -The user must be a Registered Student and must have successfully navigated to the initial access point (Sign Up/Login).

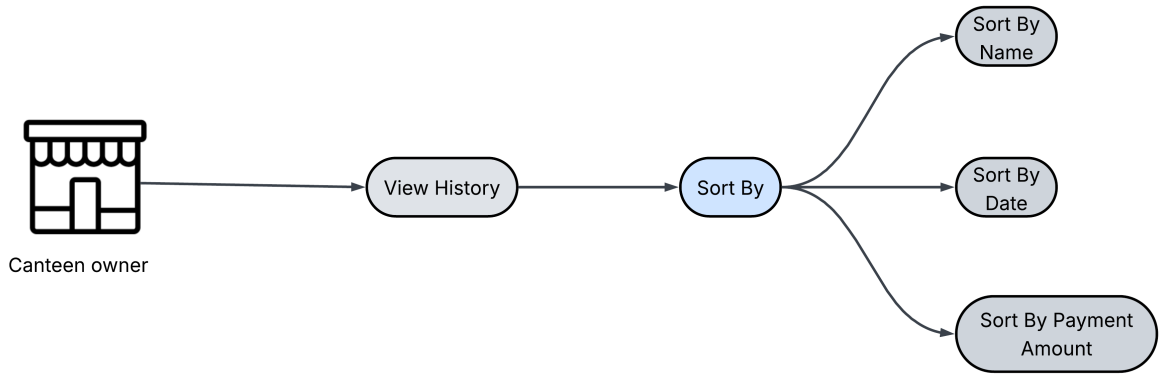
Post Conditions -The student has successfully accessed their desired information (Profile, Debts, or History) or has logged out of the system.

Actors - Registered Student.

Exceptions - In the ”Edit Profile” section, If the student attempts to save an invalid phone number (e.g., less than 10 digits) or leaves the ”Name” field blank, The system blocks the save operation.

Includes : None.

3.3.7 Use Case #7 (Financial Records Management.)



Author - Ashwin, Lekha harshaa

Purpose - To allow the canteen owner to manage and view all financial records in one place. This includes viewing transaction history (payments and cleared debts).

Requirements Traceability - Canteen Management Interface, Financial Database, Student Database, Transaction Log.

Priority - High.

Preconditions - The Canteen Owner must be logged into their administrative account with valid credentials.

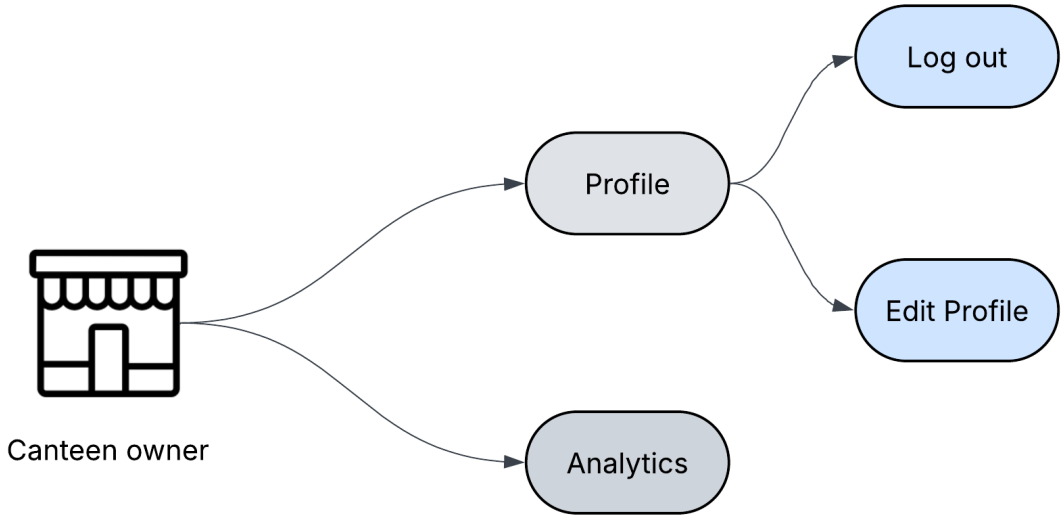
Post Conditions - The requested data (Transaction Logs and payment history) is retrieved from the database and displayed to the user in real-time.

Actors - Canteen Owner/Administrator, Financial Database, Student Database.

Exceptions - If a database connection fails during any query, prompt the user to refresh the dashboard.

Includes - None.

3.3.8 Use Case #8 (Canteen Owner Account Navigation)



Author - Ram Charan, Ashwin

Purpose - To allow the Canteen Owner to access and edit their profile information, log out of the system, and view analytics regarding canteen operations and sales.

Requirements Traceability - Canteen Owner Interface, Owner Account Database, Analytics Database.

Priority - Low.

Preconditions - The user must be a registered Canteen Owner and must have successfully navigated to the initial access point (Sign Up/Login).

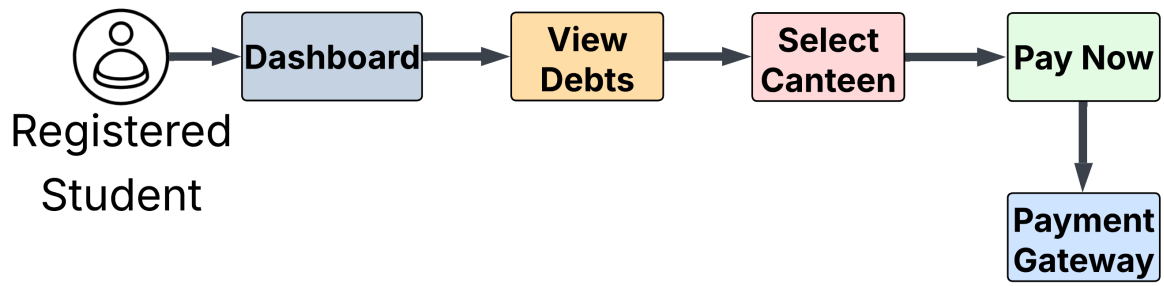
Post Conditions - The Canteen Owner has successfully accessed their desired information (Profile or Analytics) or has logged out of the system.

Actors - Canteen Owner.

Exceptions - In the ”Edit Profile” section, if the owner attempts to save an invalid phone number or leaves mandatory fields blank, the system blocks the save operation.

Includes - None.

3.3.9 Use Case #9 (Student Debt Settlement)



Author - Sathish, Shreyas

Purpose - To allow a registered student to view their outstanding debts for a specific canteen and process a payment through a secure gateway to clear their debts due for that selected canteen.

Requirements Traceability -Student Dashboard, Transactions Database, Payment Gateway Interface..

Priority - High.

Preconditions - The user must be a Registered Student and logged into the system. They must have an active internet connection to reach the payment gateway..

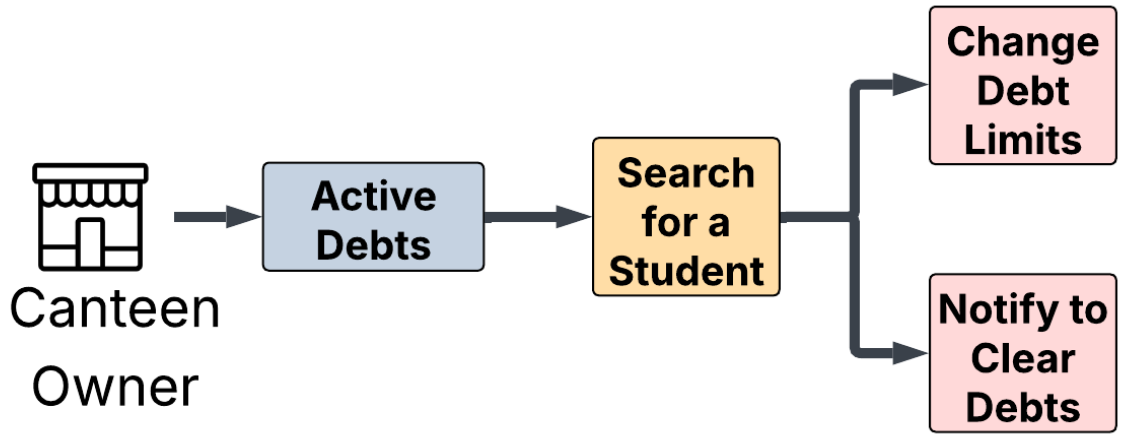
Post Conditions - The student's debt record is updated in the database, and a payment confirmation is generated for both the student and the canteen.

Actors - Registered Student, Payment Gateway, Transaction Database.

Exceptions - 1)If the student navigates to "View Debts" but has no outstanding dues, the "Select Canteen" and "Pay Now" options are disabled, and a "No Debts Found" status is displayed.
2)If the external payment gateway fails to respond within the allotted time after clicking "Pay Now," the system displays an error message: "Payment service is currently unavailable. Please try again in a few minutes."

Includes - None.

3.3.10 Use Case #10 (Student Debt Management and Notification)



Author - Ram charan, Haneesh

Purpose -To allow the canteen owner to manage debt limits of a particular student by either adjusting his/her individual debt limits or send notifications to clear outstanding balances.

Requirements Traceability -Student Dashboard, Transactions Database, Payment Gateway Interface.

Priority - High.

Preconditions - The Canteen Owner must be logged into their administrative account with valid credentials and have navigated to the “Active Debts” in the navigation sidebar of his UI.

Post Conditions - The student’s debt limit is successfully updated in the database, or a notification is dispatched to the student’s registered email/ SMS to their Phone number to clear their dues.

Actors - Canteen Owner, Students Database, Notification System.

Exceptions - 1)If the search query does not match any registered student in the ”Active Debts” list, the system displays a ”No matching student found” message and prompts for a corrected ID or name.

Includes - None.

4. OTHER NON-FUNCTIONAL REQUIREMENTS

4.1 Performance Requirements

Page Load Latency :

- For the Student Menu and Owner Dashboard, the system aims for a page load time of under 3 seconds on standard networks. Minimizing this latency is crucial during peak dining hours to prevent **digital queues** from forming, ensuring students can quickly browse and order without causing congestion at the counter.
- The system should ensure that critical interactions, such as **Pay Now** or **Add to Debt,** in near real-time (ideally under 5 seconds). This immediate confirmation is necessary to prevent user confusion (e.g., double-ordering) and to keep the service flow smooth for the canteen owner.

Live Menu Synchronization :

- The system must remain stable and responsive even when multiple students are placing orders and the canteen owner is updating the **Live Menu** simultaneously.
- Minimizing this synchronization delay is essential to prevent overselling situations where a student orders an item that effectively went out of stock seconds ago, thereby reducing the need for manual refunds and order cancellations.

Concurrency:

- The backend shall support at least 50 concurrent active users (simulating a rush hour in a single Hall canteen) without server timeout or crash, and shall be scalable up to 500 concurrent users without any significant performance degradation. It must remain stable and capable of managing a high volume of API calls, ensuring continuous operation and providing a reliable, smooth user experience even under heavy load.

Database Query Optimization :

- The database design should be capable of storing an increasing volume of transaction history and student profiles over the semester without negatively impacting query speeds or report generation times.

4.2 Safety and Security Requirements

Account Verification:

- All student users must strictly register using a **valid IITK email address (@iitk.ac.in)**. This ensures that only verified campus students can access the credit facility and place orders.
- **Canteen Owner accounts are manually provisioned** by system administrators and do not have a public sign-up page. This prevents unauthorized users from creating fake Canteen owner profiles or impersonating legitimate canteen owners.

Data Privacy :

- All user passwords must be securely encrypted in the database to prevent unauthorized access. Additionally, a student's debt profile and order history are strictly private and no other student can view them.

Debt Limit Safeguard:

- To protect both the canteen owner and the student from unmanageable dues, the system automatically disables the **Credit request option** if a student's pending debt exceeds the **Maximum Debt Limit** (default ₹2500) set by the canteen owner.

Role-Based Access Control :

- The system must strictly enforce authorization checks on every API request. A user with the role student must be restricted from accessing any canteen owner specific endpoints (e.g., modifying menu prices or clearing debts manually).
- A student shall only be able to view their own transaction history and debt profile. Accessing another student's data via ID manipulation must be blocked by the server

Payment Gateway Security :

- The system shall never store sensitive banking details (Credit Card numbers, UPI PINs, etc.) on the application servers.

4.3 Software Quality Attributes :

4.3.1 Reliability

The system must operate consistently without errors, ensuring seamless performance under all conditions. It should be resilient to failures and capable of recovering quickly from any issues that arise, providing users with a stable and dependable experience at all times.

4.3.2 Usability

- Students should find it easy to check and pay their debts,order an item searching in the menu, and his history to streamline this process.
- Canteen owner should find it easy to update his profile, update the menu, accept or reject any order, increase or decrease debt limit .

4.3.3 Portability

The application functions flawlessly across all major platforms, making it responsive and adaptable as a progressive web application .

4.3.4 Flexibility :

The platform shall be designed to adapt to the dynamic nature of a mess/canteen environment. Canteen owner shall have full control to customize their offerings in real-time—whether it's changing the price of an item, adding a new "Special Dish" for the day, or setting different debt limits for specific trusted students. This flexibility ensures the software supports the business rather than restricting it.

4.3.5 Maintainability :

The platform shall be designed to run smoothly and efficiently. Its well-organized structure and detailed documentation make maintenance straightforward. Minor security fixes shall be addressed within a few days, whereas updates or improvements, including testing and documentation, shall be typically implemented within a week.

4.3.6 Adaptability :

The application shall automatically keeps up with the changes.For example, when Canteen owner marks an item as ”Unavailable” in their dashboard, the item is immediately removed out of the Canteen’s menu .This real-time adaptability shall prevent ”overselling” and ensures the information displayed to users is always accurate and up-to-date.

4.3.7 Security :

Passwords shall be securely stored using advanced hashing techniques. If the user forgets the password, resetting it is easy with a secure link sent directly to the user’s registered email ID.

5. OTHER REQUIREMENTS

5.1 Session Timeout

To prevent unauthorized access to a student's credit facility (e.g., if a student forgets to log out on a public lab computer), the system shall automatically terminate the user session after 30 minutes of inactivity.

5.2 Financial Security

To maintain the integrity of the "Credit Ledger", the system shall enforce a strict **Immutable Ledger Policy**. Once an order status is marked as **"Accepted"** by the canteen owner, the transaction record shall become permanent and **cannot be modified or deleted by any user**. This prevents financial discrepancies and ensures that students cannot erase their debt records after receiving food.

Appendix A-Data Dictionary

A.1 User Class

Variable Name	Variable Type	Description	Example
User ID	String	System-generated unique identifier for each user.	`usr_98765`
Full Name	String	Stores the user's name entered during registration.	`Rahul Sharma`
Email	String	IITK email address. Used for login and verification.	`abc24@iitk.ac.in`
Role	String	Defines the user's access level (Student or Owner).	`STUDENT`
Mobile No	String	10-digit phone number for urgent contact.	`9123456780`
Current Debt	Positive Decimal	Total money currently owed to canteens.	`450.00`
Debt Limit	Positive Decimal	Maximum credit limit set by the canteen owner.	`2500.00`

A.2 Menu Item Class

Represents the food items available in a specific Hall Canteen

Variable Name	Variable Type	Description	Example
Item ID	String	Unique identifier for the food item.	`itm_501`
Canteen ID	String	Links the item to a specific Hall Canteen.	`hall_03`
Item Name	String	The name of the food product.	`Egg Maggi`
Price	Positive Decimal	Selling price per unit (in INR).	`45.00`
Category	String	Food classification (Veg, Non-Veg, Beverages).	`Non-Veg`
Is Available	Boolean	Toggle for "In Stock" (True) or "Sold Out" (False).	`True`

A.3 Order Class

Stores details of every order placed by a student

Variable Name	Variable Type	Description	Example
Order ID	String	Unique identifier for the order transaction.	`ord_10023`
Student ID	String	Reference to the student who placed the order.	`usr_98765`
Total Amount	Decimal	Final bill amount after calculation.	`140.00`
Order Status	String	Current state of preparation.	`PREPARING`
Timestamp	DateTime	Exact date and time when the order was placed.	`24-01-2026 13:45`

A.4 Transaction Class

Handles the financial side: Payments and Debts

Variable Name	Variable Type	Description	Example
Txn ID	String	Unique ID for the payment record.	`txn_778899`
Payment Mode	String	Method used to settle the bill.	`ONLINE` or `CREDIT`
Payment Status	String	Confirmation of success or failure.	`SUCCESS`
Sender or Receiver Name	String	Name of sender and the receiver.	`Yash Raj`

Appendix B – Group Log

Sl. No.	Date	Timings	Venue	Description
1	07/01/2026	22:30-00:30	Hall-3	Brainstormed project ideas emphasizing practical solutions designed to improve campus life.
2	09/01/2026	15:00-20:00	RM building	We shortlisted two projects,discussed with professor and finalized our project
3	11/01/2026	21:00-23:30	KD	Finalised the idea about the project and submitted google form
4	13/01/2026	21:30-23:00	RM building	Studied the SRS template given and discussed various technical doubts related to our software.
5	15/01/2026	16:30-18:30	Hall-3	Discussed some more features after meeting one of our client
6	17/01/2026	21:00-23:00	RM building	Distributed the work for SRS document
7	18/01/2026	15:00-17:00	RM building	Explored some more functionalities and progressed with the SRS document
8	20/01/2026	14:30-16:30	RM building	A meeting was conducted with the TA to address and clarify ambiguities in the document and discussed among ourselves the UI designs
9	21/01/2026	16:00-20:30	RM building	Developed the final UI images using Figma, ensuring they aligned with our project's design requirements .
10	22/01/2026	14:30-16:30	RM building	Redistributed pending work and Finalized the SRS document
11	23/01/2026	18:30-19:30	RM building	Reviewed the entire SRS document and corrected formatting and content errors.