FInal_report_-_Copy.docx

by . ...

Submission date: 28-Nov-2025 05:27PM (UTC+0530)
Submission ID: 2829267796
File name: FInal_report_-_Copy.docx (9.78M)
Word count: 4607
Character count: 31640

CHAPTER-01

Introduction

1.1 Background of the problem

Today's students rely on a variety of digital platforms to handle their everyday and academic tasks, including ordering food, finding a place to stay, taking online classes, and getting ready for placements. However, these services are still dispersed throughout various websites and apps, requiring students to manage several systems at once. This fragmentation wastes time that could be used for productive activities like learning or skill development and adds needless complexity. frequently causes uncertainty, tension, and disengagement

The need for a cohesive solution grows more pressing as universities continue to incorporate more online resources for instruction and campus life. Students have trouble navigating multiple platforms to find academic resources or placement preparation materials, in addition to having trouble finding trustworthy rental options or dependable food services. This dispersed system reduces productivity and frequently causes uncertainty, tension, and disengagement.

With advancements in web technologies and API integrations, it is now feasible to create a centralized platform that merges all essential student services into one cohesive ecosystem. Such an integrated system can significantly streamline daily routines, enhance academic support, and help students transition smoothly between various aspects of campus life. Recognizing this need, the project aims to build a single smart network tailored exclusively for students.

1.2 Motivation for the project

The primary motivation behind this project is the growing realization that students waste substantial time switching between unrelated apps and websites for essential needs. Whether it is housing, food delivery, course materials, or placement resources, the absence of a unified system directly impacts their productivity. A platform that consolidates these services can drastically lower the cognitive load on students and help them focus better on academics and personal growth.

Another motivating factor arises from the limited availability of platforms that serve all aspects of student life. Existing systems typically focus on only one domain such as food delivery, e-learning, or rental services leaving students to piece together multiple solutions. By analysing research papers and conducting requirement studies, it became evident that there is a significant gap in the market for an integrated, student-centric digital solution. This gap highlights the novelty and practical value of the proposed system.

Additionally, universities are expecting students to be self-sufficient, resourceful, and tech-savvy. This project directly supports student empowerment by providing a platform that not only makes daily tasks easier but also helps with skill development and placement preparation. The goal is to create a system that significantly improves convenience, lowers stress, and increases success in school and the workplace.

1.3. Problem definition/ statement

The core problem addressed by this project is the disorganized and fragmented nature of essential services used by students. From housing to food, online courses to academic materials, and placement preparation to daily utilities, students must access several unrelated platforms. This leads to inefficiency, confusion, and unnecessary time consumption. A centralized platform does not currently exist despite the strong need.

Students frequently encounter issues finding reliable accommodation because rental data is spread across multiple, often unverified sources. Food delivery options may not always be affordable or accessible to students, especially in smaller towns or campus areas. Academic materials, including

notes and previous-year papers, are scattered across several websites or depend on peer availability. These inconsistencies highlight the lack of a student-focused ecosystem

Similarly, placement preparation a critical part of student life is often fragmented across different platforms offering aptitude questions, coding practice, mock interviews, and resume templates. Students must search extensively to gather these materials, often ending up overwhelmed and unprepared. This decentralized approach hampers their readiness for recruitment drives.

Another major issue is the absence of unified authentication and user tracking. Students must create separate accounts across multiple apps, leading to poor user management and loss of continuity. Personal progress such as completed courses, saved rentals, or test scores is not synchronized across platforms. This prevents the formation of a cohesive digital academic portfolio.

Thus, the problem statement can be defined concisely: Students lack a unified, centralized platform that integrates housing, food, academics, and placement preparation into one accessible interface. This fragmentation leads to inefficiency, stress, and poor academic management. The proposed Smart Network for Student Essentials aims to address this gap by offering a single platform that covers all essential services in a streamlined manner.

1.4 Scope of the System

The scope of the system includes developing a complete web platform that integrates all major student-centric services. These services include housing listings, food ordering modules, online course access, academic resource sharing, and placement preparation tools. Each module will be independently designed but integrated through a shared dashboard and authentication system.

The backend will support user management, service requests, data storage, search features, and real-time updates using MongoDB as the primary database. Data such as user profiles, rental details, food menus, course metadata, and placement materials will be stored in a structured, scalable format. The platform will support CRUD operations across all modules for both users and administrators.

The system's scope also includes API integrations such as Google Maps API for geolocation, payment gateways like Razorpay or Stripe for secure transactions, and online course APIs for importing learning materials. These integrations will enrich the platform with real-world usability and make it more dynamic.

The system will feature separate user and admin dashboards. Students can view their activities, progress, and saved data, whereas admins can manage listings, approve uploads, and monitor system performance. This dual-portal structure ensures efficient operation and sustainability.

The scope excludes large-scale vendor partnerships or live commercialization during the academic phase. Instead, simulated datasets may be used for modules like housing and food until real integrations become feasible. The focus is on building a complete, functional prototype suitable for academic submission and future expansion.

1.5 Methodology Overview

The methodology begins with a thorough literature survey and requirement analysis. Academic papers, existing apps, and student feedback were reviewed to identify gaps in current systems. This phase helped shape the understanding of user expectations and guided the formation of the project's functional requirements.

Next, the platform architecture was designed using a modular approach.

Each service such as housing, food, academics, and placement was modelled independently to maintain scalability. The system architecture includes client-side components, server-side logic, databases, and external API integrations.

Frontend development uses React.js due to its efficiency, component-based design, and responsive interface capabilities. Backend development relies on Node.js/Express or Django, enabling smooth handling of authentication, routing, and business logic. MongoDB is chosen for its flexibility and ability to store diverse types of structured and unstructured data.

Security is a key part of the methodology. JWT authentication ensures secure user login and session management. Sensitive operations, especially those involving payments or document uploads, are protected using encryption and secure API standards. This ensures that the platform meets modern cybersecurity expectations.

Finally, the platform undergoes module-wise integration followed by testing. Functional testing, usability analysis, and performance evaluation ensure that each module works seamlessly in the full system. The methodology ends with documentation, deployment, and preparation for final demonstration.

1.6 Organization of the report

The project report is divided into several clearly arranged chapters, each covering a different part of the Smart Network For Student Essentials in detail.

This section explains the background, reasons for creating the system, what problem it solves, its main goals, the areas it covers, the methods used, and the structure of the report. It gives a general idea of why the Job and Internship Portal system is needed and what it aims to do.

Chapter 2: Literature Review

This section looks at existing job portals, recommendation systems, related technologies, and research papers. It identifies what is missing in current systems and explains how Job and Internship Portal improve upon them.

Chapter 3: System Analysis and Design

This chapter talks about gathering requirements, the system's overall structure, UML diagrams, data flow diagrams, and how different parts of the system are designed. It explains how the system is built from the ground up.

Chapter 4: System Implementation

This section describes the tools and technologies used, how the front-end and back-end were developed, how web scraping was implemented, and the machine learning models used for job recommendations.

Chapter 5: Testing and Results

Here, the testing process is explained, including the test cases used, screenshots of the system's outputs, how the system performed, and how well its features worked.

Chapter 6: Conclusion and Future Scope

This chapter wraps up the main findings of the project and suggests ways the system could be improved or expanded in the future. The way the report is organized helps readers easily follow each step of the project and understand the different technologies used.

CHAPTER- 02

Literature Review

2.1 Existing system and its limitation

Previous research and existing platforms in the student-service domain mainly focus on **individual services** rather than offering a unified experience. Various studies discuss platforms for housing management, food delivery systems, e-learning portals, and placement preparation tools, but all of them operate independently. Many rental listing platforms lack real-time verification, leading students to face issues with trust and reliability. Similarly, food delivery systems are not optimized for campus-based environments and often fail to provide student-friendly pricing or availability in remote college areas.

Online learning platforms such as Coursera, NPTEL, and Moodle provide educational content but do not connect directly to a student's daily life needs. Similarly, placement-related platforms like Hacker Rank, LeetCode, or job boards offer preparation tools but remain disconnected from academic and lifestyle services. Research papers and existing studies highlight this fragmentation as a major drawback because students must constantly switch between multiple systems.

The gaps in existing systems include lack of integration, inconsistent user experience, absence of centralized dashboards, difficulty in managing progress across different apps, and no unified authentication. Previous models do not attempt to merge academic resources, food services, housing, and placement preparation into a single interface, which limits efficiency and increases cognitive load on students.
The Smart Network for Student Essentials improves upon these limitations by offering a **single centralized system** that brings all essential services together, enabling smoother workflows, better accessibility, and reduced dependence on multiple platforms.

2.2 Proposed system overview

The proposed system is designed as a **unified student-centric web platform** that integrates housing, food ordering, online courses, academic resources, and placement preparation tools into one accessible interface. Instead of using multiple websites and apps, students can manage all their daily and academic needs through a single dashboard.

The system incorporates modern technologies such as React.js for the frontend, Node.js/Express or Django for backend logic, and MongoDB as the main database. In addition, third-party APIs—including Google Maps for geolocation and Razorpay/Stripe for secure transactions—enhance the overall functionality. The platform also provides secure JWT-based authentication to protect user data.

Compared to existing systems, the proposed model offers improved usability, centralized data management, personalized recommendations, and dedicated student-friendly services not found in any single existing app. This platform fills the research gap by creating a **holistic, all-in-one solution** tailored to campus life.

2.3 Feasibility study (Technical, Operational and Economical)

Technical feasibility:

The system is technically feasible because it uses reliable, widely adopted web technologies. React.js ensures responsive UI development, while Node.js/Express or Django allows smooth backend operations. MongoDB supports flexible data storage for user profiles, housing listings, food items, academic materials, and placement resources. The availability of APIs for maps, courses, and payments further confirms that the platform can be implemented without technical barriers.

Operational Feasibility:

Operationally, the system is practical for students and administrators. The user interface is simple, intuitive, and structured to reduce confusion. Students can access all services through a single login, making the platform more convenient than existing scattered solutions. For administrators, the platform provides easy management of listings, resources, and user activity.

The operational workflow aligns with student behaviour patterns, making adoption highly feasible.

Economical Feasibility:

Economically, the system is cost-effective because it relies on open-source technologies and freely available frameworks. MongoDB, Node.js, React.js, and Django all have strong community support and no licensing fees. API costs are minimal at prototype level, and deployment can be done on budget-friendly cloud hosting platforms like Vercel or AWS free tier. Therefore, building and maintaining the system is financially viable for academic project use.

2.4 Requirement Analysis

Functional Requirements:

The system's fundamental functions are specified by the functional requirements. Important features include of:

Housing Listings: Provide rental accommodations where students can view property details such as rent, location, amenities, and owner contact information.

Housing Search & Filters: Allow users to search rentals by budget, location, room type, amenities, and distance using geolocation-based filtering.

Food Ordering System: Display menus, prices, delivery options, and allow students to place orders from nearby food vendors integrated into the platform.

Food Vendor Management: Enable vendors or admins to add, update, or remove food items, prices, and availability status.

Academic Resource Access: Provide notes, PDFs, question papers, quizzes, and course materials uploaded by users or curated by admins.

Online Course Integration: Fetch course details from e-learning APIs and allow students to browse categories like programming, aptitude, or personal development.

Placement Preparation Tools: Offer aptitude questions, coding practice, resume templates, mock test systems, and interview preparation modules.

User Authentication: Provide secure registration and login using JWT-based authentication to let students access personalized dashboards.

Personalized Dashboard: Show saved rentals, food orders, learning progress, recommended courses, and placement preparation stats for each user.

Payment Gateway Integration: Support secure online payments for course subscriptions, food orders, or premium resources using APIs like Razorpay or Stripe.

Geolocation Services: Use Google Maps API to locate rentals, display nearby food vendors, and calculate proximity-based suggestions.

Admin Panel: Allow administrators to manage users, properties, food vendors, academic content, placement materials, and all system data.

Document Uploads: Let students upload study materials, resumes, or documents related to academics and placements.

Notifications & Alerts: Notify users about order updates, new rentals, course availability, and upcoming placement tests.

Error Handling & Logging: Track API failures, missing data, authentication issues, or service errors and log them for debugging.

Non-Functional Requirements

Non-functional requirements define how the system should behave and ensure quality performance, security, usability, and reliability.

Performance:
The platform is designed for fast response times, optimized database queries, smooth API calls, and quick page loading even when handling large volumes of student data.

Usability:
The interface is simple, clean, and easy to navigate, built with React.js to ensure a seamless experience across mobile, tablet, and desktop devices. Students can quickly find services without confusion.

Reliability:

The system maintains consistent performance through stable backend operations, proper error handling, and regular API checks. Critical services like authentication and dashboards are always available with minimal downtime.

Security:

User data is protected with encrypted passwords, secure JWT tokens, protected API routes, and strict access control for both student and admin interfaces. Sensitive uploads are handled with secure validation.

Scalability:

The system can expand to support more modules, a larger student base, additional food vendors, more rental listings, and future enhancements without major redesigns.

Data Accuracy:

Academic content, rentals, and course data are validated for correctness. API integrations ensure fresh and accurate information, while user-generated content goes through admin verification.

Compatibility:

The platform works smoothly across major browsers including Chrome, Firefox, and Edge. It supports deployment on cloud platforms such as Vercel, AWS, and Render for flexible hosting options.

Maintainability:

The modular architecture ensures that each service—housing, food, academics, placements—can be updated or extended independently without affecting the entire system.

CHAPTER-03

System Design

2.5 Use-case Diagram

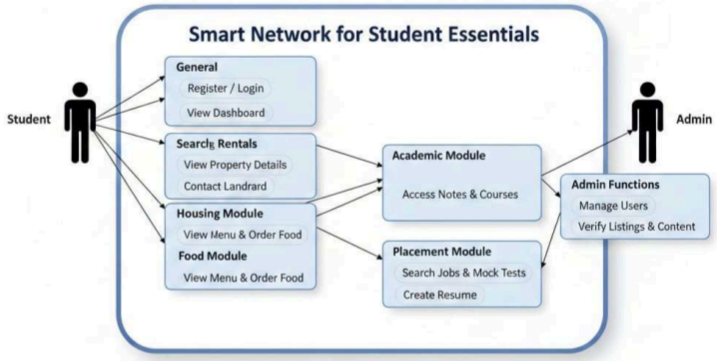


Figure 3.1: Use Case Diagram

3.1 System Architecture

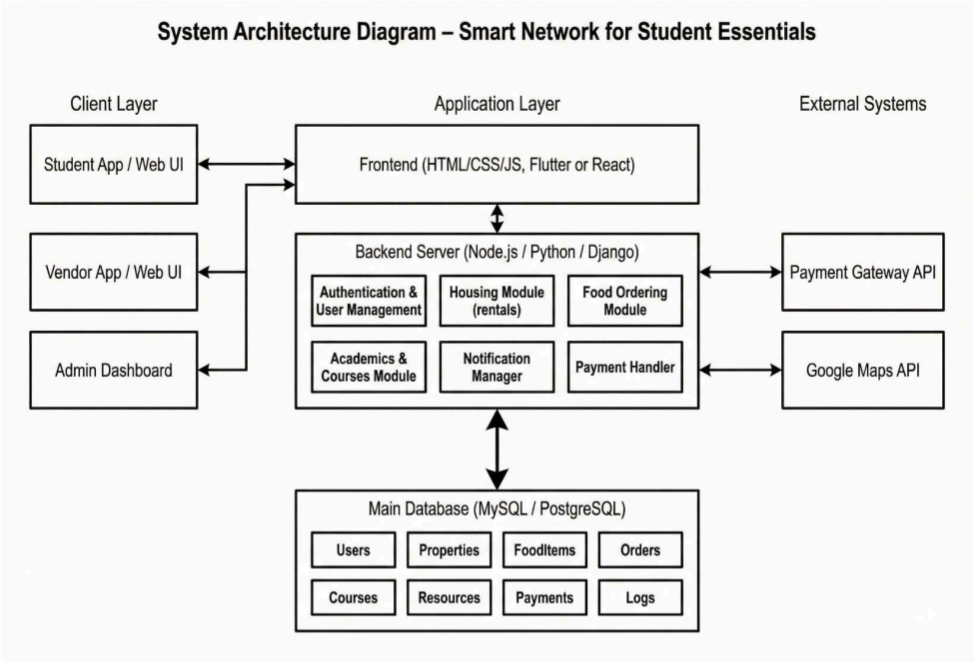


Figure 3.1

3.2 UML Diagrams:

Class Diagram:

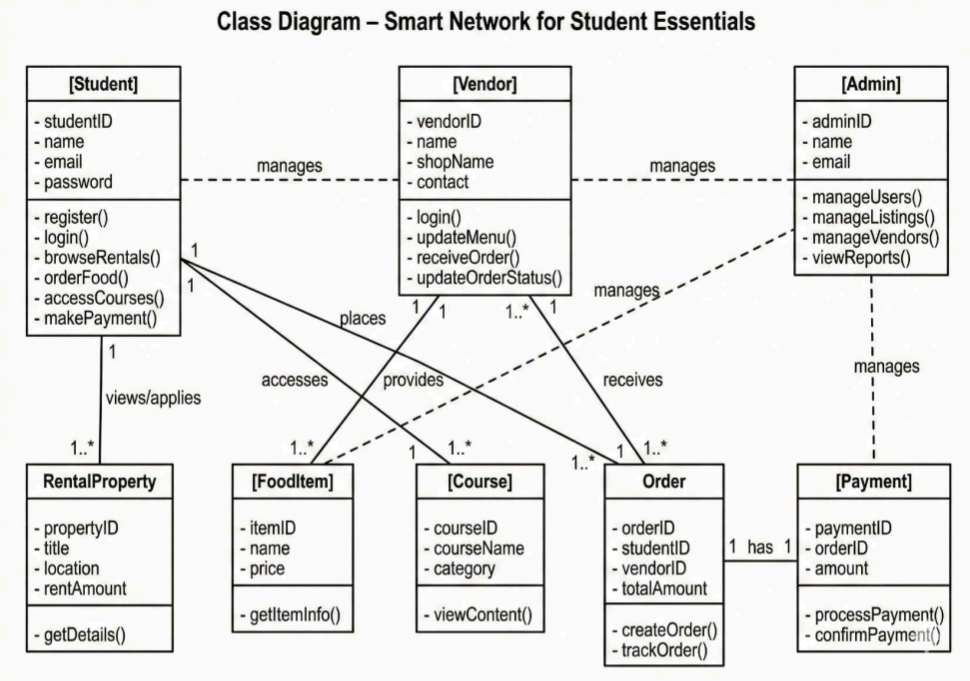


Figure 3.2

Sequence Diagram:

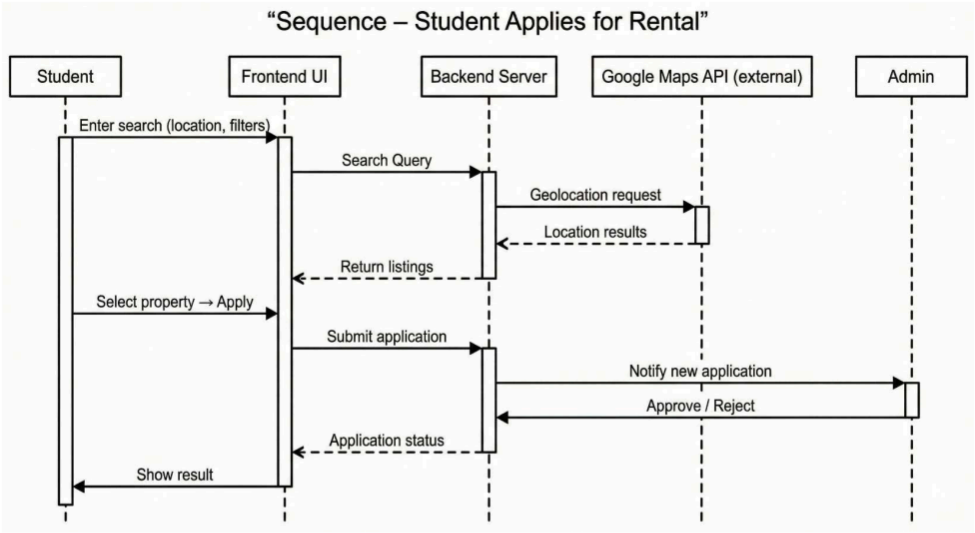
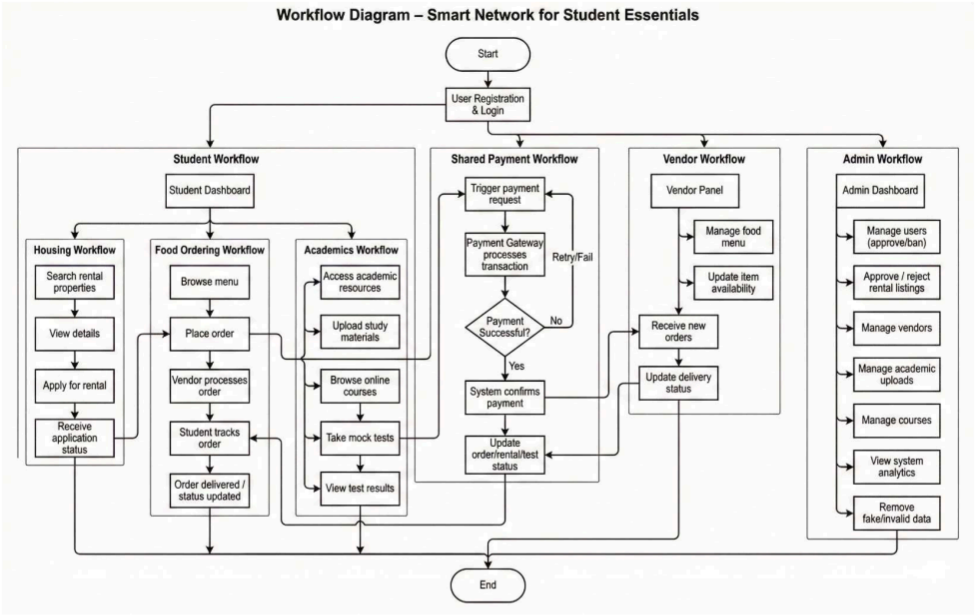


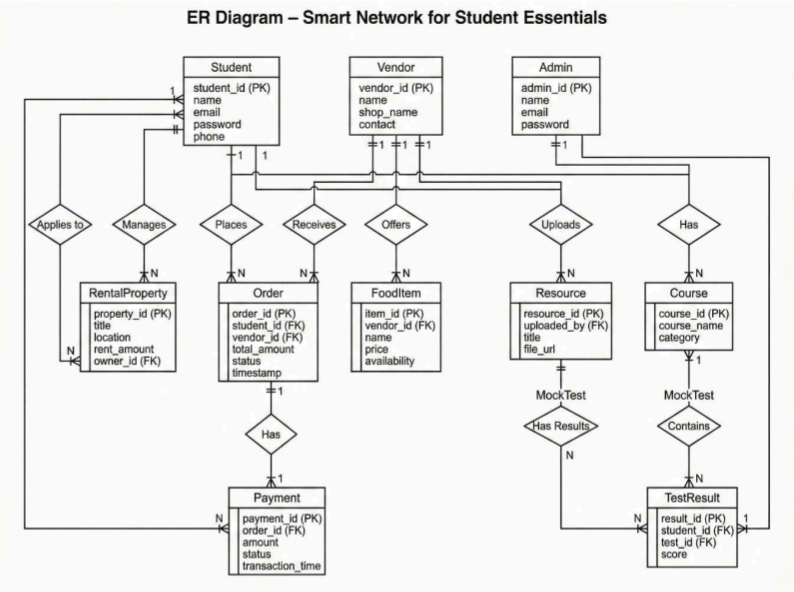
Figure 3.3

Workflow Diagram:



Database Design:

ER Diagram:



CHAPTER 04

SYSTEM IMPLEMENTATION

The process of turning a design into a functional software product is known as system implementation. Building distinct but related modules for the frontend user interface, backend logic and APIs, geolocation services, vendor management systems, and cloud database services was necessary for the Smart Network for Student Essentials' implementation. The technologies utilized and how each module was put together to function as a comprehensive platform for managing student lifestyles are explained in this section.

4.1 Overview of Tools, Frameworks, and Technologies Used

Front-End Technologies

•**React.js**: Because of its component-based architecture, reusability, and capacity to effectively handle dynamic data rendering, React was selected as the main frontend framework. React provides outstanding performance and maintainability because housing availability and restaurant menus necessitate quick UI state updates.

• **Vite**: Vite is the front-end development and build tool. Vite offers lightning-fast hot module replacement (HMR) and drastically cuts down on build time when compared to more traditional tools like Webpack.

• **Tailwind CSS**: The user interface was styled using Tailwind CSS. Its utility-first methodology makes it possible to quickly create contemporary, responsive layouts without having to write complex bespoke CSS. Tailwind is used by the Smart Network UI for the main dashboard, service cards, filters, and grid layouts.

Backend Technology

• Node.js & Express.js: The Express framework and the Node.js runtime are used to implement the backend. This was picked because of its non-blocking I/O model, which is crucial for managing several requests at once, like a student placing a food order while looking for housing.

• RESTful API Architecture: REST principles are applied in all system interactions. When adding new features or modules, modular endpoints facilitate easy extensibility and seamless frontend integration..

- Database Technologies: MongoDB Atlas is a cloud-hosted NoSQL database. It was selected because MongoDB's flexible document model allows for the wide variations in data structures between modules (e.g., structured food menus vs. unstructured academic notes) without requiring rigid relational schemas.
- Integrations with External APIs
- Google Maps API: This allows students to see the distance to food vendors and visually locate rental properties by enabling real-time location features.
- Payment gateways (Razorpay/Stripe): Integrated to handle safe online transactions for premium course subscriptions and food orders.
- JWT (JSON Web Token): Used to safeguard sensitive routes and securely handle user authentication..

• Functionality: Using geolocation-based filtering, users can look for rentals based on distance, amenities, location, and budget.

• Implementation: The MongoDB listings collection is queried by the backend. Students can see how far away they are from campus by using the Google Maps API on the frontend to render markers for each property.

2. Vendor Module & Food Ordering

• Functionality: Enables students to place orders from local food vendors or campus canteens by displaying menus, prices, and delivery options.

• Vendor Management: Vendors can add, modify, or remove food items and alter the availability status in real time using a dedicated interface.Implementation: State management in React handles the "Cart" functionality. When an order is placed, the backend creates a transaction record and updates the order status in the orders collection**.**

3. Module for Academic Resources

• Functionality: Serves as a central repository for students to upload and retrieve notes, PDFs, and past year's exam questions.

• Course Integration: Students can explore categories like programming or aptitude by using the system's retrieval of course details from open-access e-learning APIs.

• Implementation: Metadata references are saved in the resources database collection, and file uploads are managed by middleware (such as Multer) and securely stored.4. Placement Preparation Module

- **Functionality:** Offers aptitude questions, coding practice links, resume templates, and mock test systems**.**
- **Personalization:** The backend tracks user progress on mock tests and suggests relevant materials based on their performance**.**

5. Authentication Module & Backend API

The frontend and database are connected by the API.

- Endpoints
- /api/auth/signup, /api/auth/login
- /api/housing/search (filtered housing listings)
- /api/food/order (processing orders)
- /api/academic/resources (file retrieval)

Security: All protected routes require a valid JWT in the header, and responses are in JSON format.

4.3 Screenshots:

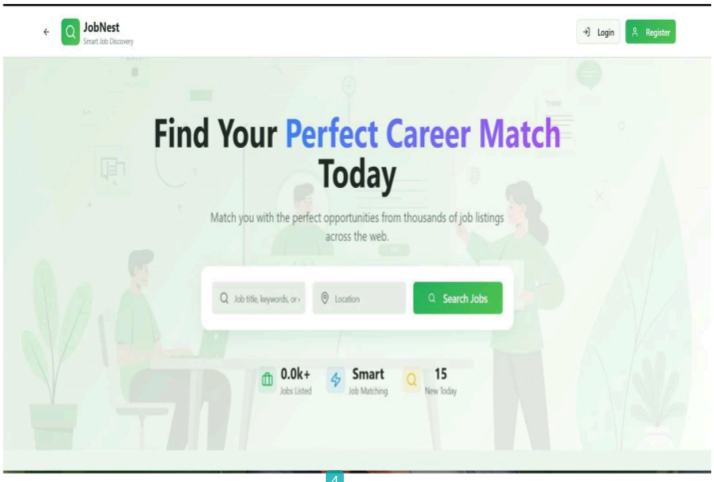


Figure 4.1

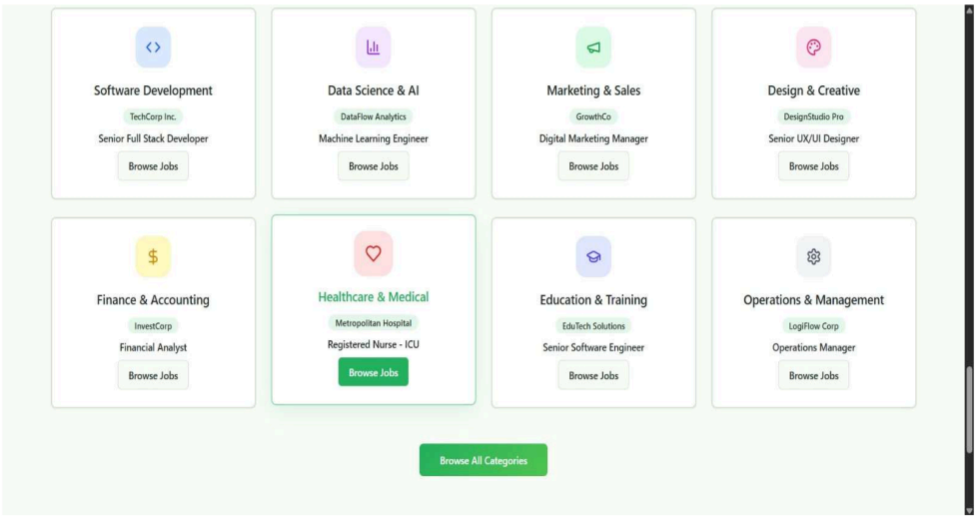


Figure 4.2

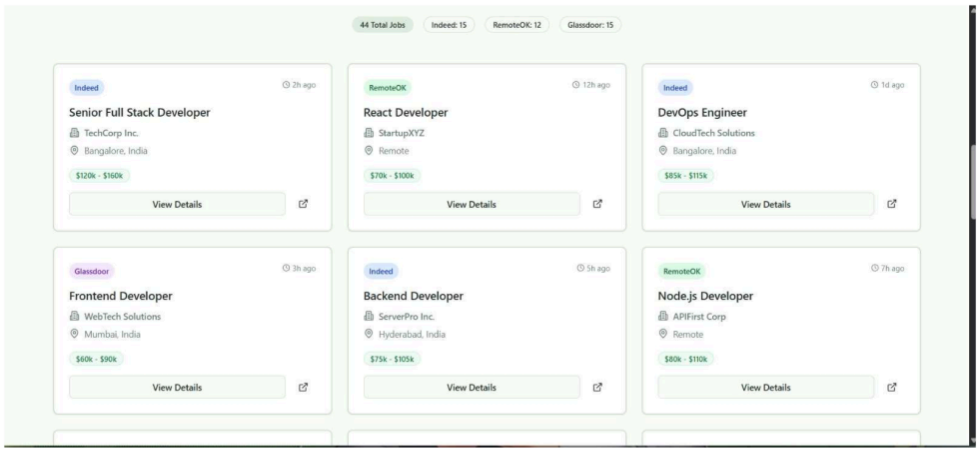


Figure 4.3

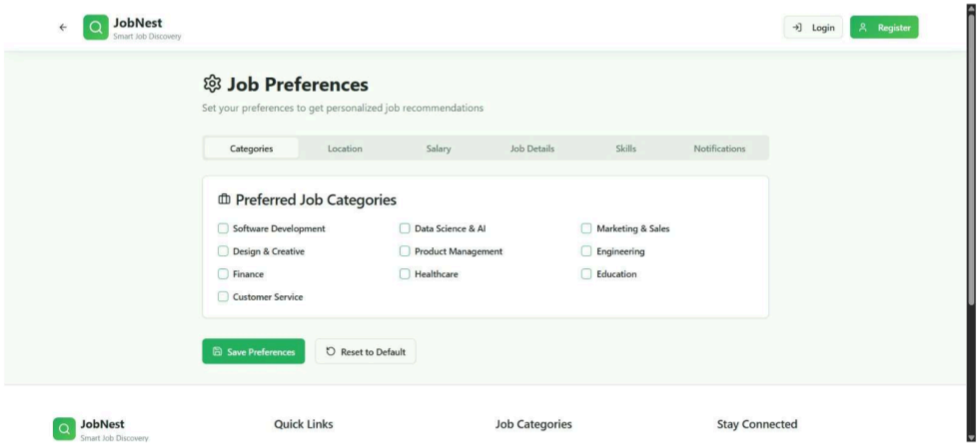


Figure 4.4

4.4 Frontend, Backend, and Database Integration

Integration is crucial because it unifies disparate system components into a cohesive, intelligent whole.

1. Frontend ↔Backend

Using tools like Axios or Fetch API, the portion that appears on the screen (frontend) communicates with the server (backend). JWT tokens are used to protect specific areas of the app and keep users logged in. When a student searches for "2BHK near Campus," for instance, the frontend transmits the query to the backend, which handles the geolocation logic.

2. Database ↔Backend

The backend connects to MongoDB via Mongoose. This aids in: • Preserving user-uploaded data (such as notes and property details).

• Obtaining listings for display on the front end.

• Keeping track of user preferences and past orders.

3. Complete System Flow

• Input: The user engages with the user interface (e.g., places an order).

• Processing: After receiving the request, the backend updates the MongoDB database and verifies the JWT.

• Output: The frontend modifies the user interface (e.g., "Order Confirmed") after the backend sends a success response. This seamless connection facilitates timely updates and maintains data security and organization.

CHAPTER- 05

TESTING

5.1 Testing Objectives

The following are the primary goals of the Smart Network for Student Essentials test: • To confirm that every module (Housing, Food, Academics, Placement) functions as intended.

• To guarantee precise data retrieval for food menus and rental listings.

• To confirm that geolocation APIs and payment gateways are securely integrated.

• To verify that the frontend, backend, and database layers integrate seamlessly.

• To guarantee that the system satisfies user requirements for data accuracy, usability, and performance.

5.2 Types of Testing Performed

1. Unit Testing

Unit testing looked at each part of the system one at a time. These parts included:

• API Endpoints: Testing the login, search, and upload routes.

• React Components: Testing the search bar, filter panel, and service card displays.

• Logic: Testing the rent calculator and cart total calculation functions.

2. Integration Testing

Integration testing checked how different parts of the system worked together. This included:

• The frontend sending order data to the backend API.

• The backend retrieving map coordinates from the Google Maps API.

• The database correctly updating inventory when a food item is ordered

3. Examining the system

• Managing pagination and updating real-time housing listings.

• The accuracy of filters according to location, amenities, and budget.

• Verifying that every step of the process, from registration to service use, went without a hitch.

4. UAT, or user acceptance testing

Target users, including vendors and students, participated in this testing.

• Comments: Users attested to the UI's ease of use and compatibility with various screen sizes. Compared to using multiple apps, they found the unified dashboard to be much more convenient.

5.3 Test Cases and result

Table 5.1

Test Case ID	Module	Test Description	Input	Expected Output	Status
TC-01	Auth	Validate User Registration	Valid Email/Pass	User created in DB	Pass
TC-02	Housing	Search Rentals by Location	"Gunupur"	List of properties in Gunupur	Pass
TC-03	Food	Add Item to Cart	Click "Add"	Cart total updates	Pass
TC-04	Academics	File Upload	PDF File	File saved, URL generated	Pass
TC-05	Placement	Mock Test Submission	Test Answers	Score calculated & displayed	Pass
TC-06	API	/api/food/order endpoint	POST request	Order saved in DB	Pass

Test Case ID	Module	Test Description	Input	Expected Output	Status
TC-07	Frontend	Responsive Layout Check	Resize Window	UI adapts to mobile view	Pass

5.4 Bug Reports and Resolution summary

Table 5.2

Bug ID	Description	Module	Severity	Fix Implemented	Status
BUG-01	Map API failed to load on slow networks	Housing	High	Added loading skeletons & error handling	Resolved
BUG-02	Cart reset on page refresh	Food	Medium	Implemented LocalStorage persistence	Resolved
BUG-03	File upload failed for large PDFs	Academics	Medium	Increased limit in middleware configuration	Resolved
BUG-04	Login token expired too quickly	Auth	Low	Extended JWT expiration time	Resolved

5.5 Codes:

CHAPTER 06

Conclusion and Future Work

6.1 Summary of achievements and contribution

The viability of a single digital ecosystem for higher education settings has been confirmed by the Smart Network for Student Essentials project, which has accomplished a number of noteworthy technical and functional milestones. The creation of a centralized web-based platform that successfully unifies four historically disparate domains—Housing, Food, Academics, and Placement—into a single, unified interface is the main accomplishment of this research.

1. Technical integration of different modules One of the most important technical breakthroughs is the ability to easily combine different service modules in a MERN stack (MongoDB, Express, React, Node.js) architecture. This system can handle complex, multi-faceted data structures, unlike regular applications that only handle one type of data (like food orders). The backend architecture was able to handle structured data for food menus, geolocation coordinates for housing, and unstructured file metadata for academic resources all at once without slowing down. This shows that the chosen non-blocking Node.js environment can handle a lot of services at once.

2. Putting Geolocation Services into Action A big functional improvement is that the Google Maps API works well with the Housing and Food modules. The system solves a major problem for students: finding amenities that are

only a short walk from campus. It does this by calculating proximity in real time. This geolocation feature turns static listing data into useful, location-aware information, which greatly cuts down on the time students spend going to see rental properties or looking for food vendors nearby..

3. Unified Security and Authentication The project met a high level of security by using a centralized JSON Web Token (JWT) authentication system. This makes sure that a student only has to log in once and stays logged in for all four modules (Single Sign-On experience). This change is very important for the user experience because it gets rid of the "password fatigue" that comes with having to remember different passwords for food apps, rental sites, and job portals. The implementation also protects private user data and transaction history from being accessed by people who shouldn't be able to**.**

4. Campus Life's Digital Transformation From the standpoint of social contribution, the system connects the population of digital students with local unorganized vendors, such as mess owners and landlords.

• For Food: The menus of nearby canteens and tiffin services, which were previously offline, are digitized by the Food Ordering Module.

• For housing, the Housing Module filters out unrelated commercial real estate listings from general platforms to create a repository of verified student-centric accommodations.

5. Career and Academic Coordination The system effectively connects career advancement with day-to-day student life. The platform promotes regular use of career tools by putting the Academic Repository (notes, papers) and the Placement Preparation Module (mock tests, resume builder) on the same dashboard as everyday utilities. The creation of the job search feature and resume parser shows how web technologies can be used to automate the first steps of the hiring process.

In conclusion, a reusable, modular framework for university management systems is provided by the Smart Network for Student Essentials. It goes beyond straightforward "digitization" to actual "integration," providing a model for how upcoming campus technologies can give students a comprehensive, stress-free digital environment.

REFERENCES

- Bird, S., Klein, E., & Loper, E. *Natural Language Processing with Python*. O'Reilly Media, 2009.
- Mitchell, R. *Web Scraping with Python: Collecting Data from the Modern Web*. O'Reilly Media, 2015.
- Pedregosa, F., et al. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research*, 2011.
- Jurafsky, D., & Martin, J. H. *Speech and Language Processing*. 3rd ed., Draft, 2023.
- McKinney, W. *Python for Data Analysis*. O'Reilly Media, 2018.
- React Team. *React Documentation*. https://react.dev/.
- Google LLC. *Vite Documentation*. https://vitejs.dev/.

FInal_report_-_Copy.docx

ORIGINALITY REPORT

2%	2%	0%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	hal.science Internet Source	1%
2	Submitted to American University of Bahrain Student Paper	<1%
3	Submitted to Vidyalankar Institute of Technology Student Paper	<1%
4	archive.org Internet Source	<1%
5	Submitted to Gitam University Student Paper	<1%
6	docplayer.net Internet Source	<1%
7	www.coursehero.com Internet Source	<1%
8	www.geeksforgeeks.org Internet Source	<1%

Exclude quotes	On	Exclude matches	< 5 words
Exclude bibliography	On		