

“ KIOSKMEET ”
Where All Solutions Converge

Enroll Numbers : 21103291, 21104031, 21103279

Name : Manya Jindal, Aryan Gupta, Tanya Gupta

Name of Supervisor : Dr. Tribhuwan Kumar Tewari



May - 2024

**Submitted in Partial Fulfillment of Degree
of Bachelor of Technology in
Computer Science and Engineering & Information Technology**

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING &
INFORMATION TECHNOLOGY**

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

(I)
TABLE OF CONTENTS

Chapter No.	Topics	Page No.
Chapter - 1	Introduction	10-14
1.1.	General Introduction	
1.2.	Problem statement	
1.3.	Novelty	
1.4.	Brief Description of the Solution Approach	
Chapter - 2	Literature Survey	15-18
2.1.	Prerequisite Knowledge	
2.2	Integrated summary of the literature studied	
Chapter - 3	Requirement Analysis and Solution Approach	19-29
3.1	Prerequisite Knowledge	
3.1.1	Databases and its types	
3.1.2	Online Payment Gateway	
3.1.3	Frontend Skills	
3.1.4	Backend Skills	
3.2	Requirement Analysis	
3.2.1	Functional Requirements	
3.2.2	Non-Functional Requirements	
3.3	Implementation Tools	
3.4	Solution Approach	
3.4.1	WebPortal	
3.4.2	Laundry Updates	
3.4.3	Cafeteria	
3.4.4	Photocopy Shop	

Chapter - 4 Modeling and Implementation Details	30-36
4.1 Design Diagrams	
4.1.1 Class Diagrams	
4.1.2 Use Case Diagram	
4.1.3 Sequence Diagrams	
4.2 Implementation details and Features Built	
Chapter - 5 Findings, Conclusions, and Future Work	52-56
6.1 Findings	
6.2 Conclusions	
6.3 Future Work	
References	57

(II)

DECLARATION

We hereby declare that this submission is our own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

Place: JIIT, Sec-62, Noida

Signature:

Date:

Name: Manya Jindal

Enrollment No.: 21103291

Place: JIIT, Sec-62, Noida

Signature:

Date:

Name: Aryan Gupta

Enrollment No.: 21104031

Place: JIIT, Sec-62, Noida

Signature:

Date:

Name: Tanya Gupta

Enrollment No. :21103279

(III)
CERTIFICATE

This is to certify that work titled "**KioskMeet: Where All Solutions Converge**" submitted by "**Aryan Gupta (21104031), Manya Jindal (21103291), Tanya Gupta (2113279)**" in partial fulfillment for the award of degree of **B.Tech** of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other university or institute for the award of this or any other degree or diploma.

Signature of Supervisor

Name of Supervisor Dr. Tribhuwan Kumar Tewari

Designation

Date

(IV)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to everyone who has contributed to the successful completion of our Semester-VI Minor Project, titled “KioskMeet: Where All Solutions Converge”

First and foremost, we extend our heartfelt thanks to Prof. Vikas Saxena, Head of the Department - CSE & IT, for his unwavering support and invaluable insights throughout the course of this project. His guidance has been instrumental in shaping our ideas and pushing us towards excellence.

We would also like to extend our sincere appreciation to our esteemed faculty coordinator, Dr. Tribhuwan Kumar Tewari, for his exceptional guidance and mentorship. His expertise in his respective field has been a constant source of inspiration. Dr. Tewari's insightful feedback and constructive criticism have played a pivotal role in refining our approach and methodology.

We are also grateful to the entire faculty of the department for their continuous support and encouragement. Their expertise and commitment to academic excellence have been a source of motivation for all of us.

Lastly, we would like to express our heartfelt thanks to our student mentor Ms. Shreya Garg for her dedication and hard work throughout this project. This project has been a significant learning experience, and we are thankful to everyone who has been a part of this journey.

(V)
SUMMARY

1. User Login with JSON Web Token (JWT):

- Users enter credentials for login.
- Server validates credentials and generates a JWT.
- JWT contains user information and is signed with a secret key.
- Email authentication process is triggered after successful login.

2. Email Authentication:

- An email with a unique authentication link is sent to the user's email.
- Link/token verifies user authenticity.
- Nodemailer in Node.js is used for sending emails.

3. Student and Teacher Login:

- Students and teachers login with username and password.
- PHP backend validates credentials against the database.
- Successful login redirects to respective dashboards.

4. Photocopy Shop

- Send a custom crafted email with all the necessary customizations for printing along with attachments using Multer and Nodemailer.

5. Laundry Notification System:

- Users should receive notifications regarding the status of their laundry, including when clothes are ready for pickup.

6. Cafeteria Services:

- Admin Interface: Admins manage cafeteria items (add, edit, delete) via a separate dashboard.
- User Interface: Users browse cafeteria items, view details, and add items to cart.
- Payment Process: Users provide payment details for checkout.

(VI)
LIST OF TABLES

Table No.	Title	Page No.
1	Difference between Traditional Method and Updated Version	12

(VII)
LIST OF FIGURES

Fig. No.	Figure Title	Page No.
4.1.1	Class Diagram	28
4.1.2	Use Case Diagram	28
4.1.3	Sequence Diagram of Cafeteria Services	29
4.2.1	Website View	30
4.2.2	Select Role	30
4.2.3	Student Login	31
4.2.4	Student Profile	31
4.2.5	Subject View	32
4.2.6	Admin Dashboard	32
4.2.7	Add Students - Admin Dashboard	33
4.2.8	View Existing Teachers - Admin Dashboard	33
4.2.9	Laundry Admin Panel Login	34
4.2.10	Laundry Admin Panel - Email Sending Dashboard	34
4.2.11	Email Confirmation	35
4.2.12	Emails Received	35

4.2.13	Cafeteria View	35
4.2.14	Menu Items	36
4.2.15	Additional Menu Items	36
4.2.16	Manage Items From Admin Panel	37
4.2.17	Manage Users As Admin	37
4.2.18	Add Item To Menu As Admin	38
4.2.19	Delete Items from Cart	38
4.2.20	Cart Items	39
4.2.21	Checkout Screen	39
4.2.22	Tracking Orders	40
4.2.23	User Gets Logged Out When Tries To Access Admin Dashboard	40
4.2.24	Mongodb Database For Users View	41
4.2.25	Stripe Test Payment Keys	41
4.2.26	Getting Details from user for Printouts	42

Chapter 1 - Introduction

1.1 General Introduction:

KioskMeet is a revolutionary online platform designed to revolutionize the college experience for both students and faculty members. In the bustling environment of academic institutions, time is of the essence, and KioskMeet aims to alleviate the common challenges that often disrupt daily routines.

At its core, KioskMeet is a multifunctional tool that caters to the diverse needs of college communities. It serves as a comprehensive solution, offering a plethora of features tailored to streamline various aspects of campus life. Whether it's tackling the notorious long queues at the cafeteria during peak hours or optimizing the management of queues at the photocopy shop, KioskMeet is equipped to address these pain points effectively.

One of the standout features of KioskMeet is its integration of a secure payment gateway. This allows users to seamlessly conduct transactions for a range of services, including cafe purchases, photocopy services, and laundry facilities. By facilitating cashless transactions, KioskMeet not only enhances user convenience but also promotes a safer and more efficient ecosystem within the college community.

Furthermore, KioskMeet doesn't just stop at optimizing existing processes; it also strives to innovate and adapt to the evolving needs of its users. Through regular updates and enhancements, the platform remains dynamic and responsive, ensuring that it continues to meet the ever-changing demands of college life.

Behind the scenes, KioskMeet operates with a robust administrative framework, providing oversight and management of the platform's operations. This ensures smooth functionality and efficient service delivery, ultimately contributing to a more seamless experience for all users.

1.2 Problem Statement:

It is a common and well-known problem that exists not only in our college but in other colleges which caught our attention. Apart from the WebPortal that does not always work as desired, the college community faces challenges such as long queues at the cafeteria during peak hours and inefficient management of queues at photocopy shop, absence of timely updates from laundry services. These issues hinder the smooth functioning of the academic environment and require urgent attention and solutions.

Table 1.1 - Difference between Traditional Method and Updated Version

	Traditional Method	Updated version
Time Efficiency in Cafeteria	Traditional coupon distribution systems cause long queues during lunch breaks.	KioskMeet cuts out the hassle of long lunch break queues by letting users grab coupons online, saving time and stress.
Time Efficiency in Laundry	Initially, visiting the laundry often resulted in wasted time as clothes weren't ready for collection until another day, causing frustration and inconvenience.	Now, with KioskMeet, we've streamlined the laundry process by onboarding laundry staff who send notifications via email regarding when clothes will be ready for collection. This feature eliminates wasted time and inconvenience, enhancing efficiency for students and faculty.
Saves time at Photocopy Shop	Previously, waiting for your turn at the photocopy shop meant enduring long queues, consuming valuable time. However, with KioskMeet, this hassle is a thing of the past.	Now, through KioskMeet, you can submit your documents for printing and complete the payment process seamlessly without the need to wait in line at the photocopy shop.

1.3 Novelty:

The novelty of this project lies in its comprehensive approach to addressing the specific challenges faced by the college community through the implementation of the KioskMeet platform. Unlike existing solutions that may only tackle individual issues or offer limited functionality, KioskMeet offers a holistic solution that integrates various features tailored to the needs of students, faculty, and administrators.

One key aspect of novelty is the user-friendly login system, which enhances accessibility and ease of use for all members of the college community. By simplifying the login process, KioskMeet ensures that users can quickly access the platform and avail themselves of its many functionalities without unnecessary friction.

Moreover, the implementation of secure processes for data management addresses concerns about privacy and confidentiality, instilling confidence in users regarding the safety of their personal information. This focus on security sets KioskMeet apart as a reliable and trustworthy platform for conducting transactions and accessing sensitive data.

Another innovative feature is the seamless integration of services such as cafe ordering, laundry updates, and photocopying, streamlining these processes and reducing the time and effort required to access essential campus services. By centralizing these functions within a single platform, KioskMeet simplifies campus life and enhances overall efficiency.

By combining these innovative features into a single platform, KioskMeet has the potential to transform the way we access information, and navigate campus life.

1.4 Brief Description of the Solution Approach:

Our comprehensive solution tackles common challenges encountered by college communities, offering streamlined processes and improved user experiences. Through the integration of MongoDB, React, Node.js, HTML, and CSS, we've developed an integrated web kiosk platform. This platform revolutionizes food ordering by enabling users to purchase coupons online, eliminating the need for queue at the cafeteria during peak hours.

Moreover, we've implemented a secure payment gateway, ensuring smooth and secure transactions. Additionally, our solution addresses the inefficiencies of photocopy services by allowing users to submit documents for printing and complete payments online. This eliminates the need to wait in long queues at the photocopy shop, saving valuable time and enhancing efficiency.

Furthermore, we've enhanced laundry services by enabling laundry staff to send email notifications when clothes are ready for collection. This eliminates the need for users to physically visit the laundry, further optimizing time management.

By combining these innovative features with robust technologies, our solution addresses critical pain points within college environments, enhancing efficiency, convenience, and overall user satisfaction.

Chapter 2 - Literature Survey

2.1 Summary of Texts Studied:

2.1.1 PHP v/s JavaScript [1]

<https://www.cloudways.com/blog/php-vs-javascript/>

When embarking on a new web development project, choosing the right technology is paramount. Among the top contenders in the field are PHP and JavaScript, each offering distinct features and advantages.

PHP and JavaScript are fundamental to creating dynamic and interactive web applications, yet they possess unique characteristics and strengths.

In this comprehensive guide, we'll delve into the contrasting attributes of PHP and JavaScript, exploring their capabilities, frameworks, and community support. By examining these factors, we'll equip you with the insights needed to make an informed decision when selecting the optimal technology for your project.

PHP:

Now let's dive deeper into PHP and explore its capabilities, frameworks, compatibility with

databases, and community support.

Disadvantages of Using PHP for Web Development:

- Slower performance compared to other languages
- Security concerns with some commonly used functions
- Limited scalability for larger applications

JavaScript:

Now, let's shift our focus to JavaScript and dive into its client-side scripting capabilities, frameworks, compatibility with different platforms, and community support.

Advantages Associated With JavaScript-Based Web Development:

- Fast runtime performance due to client-side rendering.
- Open-source ecosystem.
- Compatibility with modern web technologies.
- Broad application areas, from building chat apps to fully-fledged enterprise systems.

JavaScript is a client-side scripting language that runs directly on the user's web browser. It makes websites more interactive by allowing users to interact with elements on a web page without refreshing or reloading the entire page.

With its robust libraries and frameworks like React, VueJS, AngularJS, Node.js, etc., JavaScript has become increasingly popular among developers for creating single-page applications (SPAs) and real-time web applications.

Because of its adaptability and versatility, JavaScript has grown in popularity recently. It can be used on both the client side (in web browsers) and server-side (with Node.js), making it a great choice for full-stack development projects.

2.1.2. Test Payment Methods:

<https://docs.stripe.com/testing> [2]

To effectively test your integration with Stripe for payments, you can use a variety of test cards provided by Stripe. These test cards simulate different scenarios, such as successful payments, card errors, authentication challenges, and more. By using these test cards, you can verify that your integration works correctly without actually processing any real transactions.

For interactive testing, you can input test card details like the card number, expiration date, and CVC into your payment form. **An example test card number is 4242 4242 4242 4242, with an expiration date of 12/34 and any three-digit CVC.**

When writing test code, it's recommended to use test PaymentMethods like `pm_card_visa` instead of directly using card numbers. This approach ensures PCI compliance and better security for your code.

Here's a summary of the key points for testing payments with Stripe:

- Use test API keys in all API calls to Stripe.
- Avoid using real card details; instead, utilize the provided test card numbers.
- Test cards simulate various scenarios, including successful payments, card errors, and authentication challenges.
- Test interactively by entering test card details into your payment form.
- When writing test code, prefer using PaymentMethods instead of card numbers directly.
- Ensure that you switch to live API keys when your integration is ready to process live payments.

By following these guidelines and utilizing the provided test cards, you can effectively test your integration with Stripe and ensure its reliability and functionality before going live.

2.1.3 NodeMailer Tutorial: Sending an Email Using NodeMailer 101: [3]

<https://www.turing.com/kb/comprehensive-guide-to-sending-an-email-using-nodemailer>

NodeMailer is a powerful Node.js module that simplifies the process of sending emails from your server. It is widely used in Node.js applications for its ease of use, flexibility, and robust features.

Key Features of NodeMailer:

- 1) **Single Module with Zero Dependencies:** NodeMailer is a standalone module with no external dependencies, making it lightweight and easy to manage.
- 2) **Platform-Independent:** It works seamlessly across different platforms, ensuring consistent performance regardless of the underlying operating system.
- 3) **Unicode Support:** NodeMailer supports Unicode characters, allowing you to use any language or special characters in your emails.
- 4) **Security Focus:** With a heavy focus on security, NodeMailer minimizes the risk of vulnerabilities such as Remote Code Execution (RCE).
- 5) **HTML Content Support:** You can send HTML-formatted emails in addition to plain text, enabling rich email designs and layouts.
- 6) **Attachment Support:** Easily add attachments to your emails, including files such as images, documents, or other media.
- 7) **Secure Delivery:** NodeMailer supports secure email delivery using TLS/STARTTLS encryption protocols, ensuring data integrity during transit.
- 8) **Embedded Images:** Attach images directly to HTML content, allowing for visually appealing email designs with embedded graphics.

By leveraging NodeMailer's capabilities, you can easily integrate email functionality into your Node.js applications, whether it's sending notifications, newsletters, or transactional emails. Its intuitive API and robust features make it a popular choice for developers looking to incorporate email communication into their projects.

2.2 Integrated summary of the literature surveyed

The project aims to streamline college operations and enhance user experience through an integrated web platform. This platform addresses common challenges faced by college communities, such as long queues at the cafeteria, inefficient management of queues at the photocopy shop, and lack of timely updates from laundry services. By integrating functionalities like food ordering, document printing,

and laundry notifications, the platform aims to improve efficiency and convenience for students, faculty, and administrators.

PHP vs JavaScript:

In selecting the appropriate technology stack for the project, the comparison between PHP and JavaScript is crucial. PHP offers compatibility with databases and is commonly used for server-side scripting. However, it may face challenges such as slower performance and security concerns. On the other hand, JavaScript excels in client-side scripting, offering fast runtime performance and compatibility with modern web technologies. Its versatility makes it suitable for both front-end and back-end development, aligning well with the project's goals of creating a dynamic and interactive web platform.

Stripe Test Payment Methods:

For testing payment integration with Stripe, developers can utilize various test cards provided by Stripe. These cards simulate different payment scenarios and allow developers to verify the integration without processing real transactions. By following guidelines and using provided test cards or PaymentMethods, developers can ensure reliable payment functionality within the platform before deploying it live.

NodeMailer Tutorial:

NodeMailer serves as a crucial component for implementing email communication within the platform. Its features, including platform independence, Unicode support, and security focus, make it an ideal choice for sending notifications, updates, and other email communications. By integrating NodeMailer into the project, developers can enhance user engagement and streamline communication processes within the college community.

Overall, by leveraging technologies like JavaScript for front-end development, integrating payment functionality with Stripe, and implementing email communication using NodeMailer, the project aims to create a comprehensive and efficient web platform tailored to the specific needs of college communities.

Chapter 3 - Requirement Analysis and Solution Approach

3.1 Prerequisite Knowledge

3.1.1 Databases and its types:

For the project focused on streamlining college operations and enhancing user experience through an integrated web platform, databases play a crucial role in storing and managing data efficiently. Let's discuss databases and the types of databases (SQL and NoSQL) with respect to the project:

Databases:

Centralized Data Storage: Databases serve as centralized repositories for storing various types of data related to the college community, including user profiles, transaction records, cafeteria orders, laundry notifications, and more.

Data Organization: Databases enable structured organization of data, facilitating efficient retrieval and manipulation of information based on specific criteria or requirements.

Data Security: Proper database management ensures data security and integrity, safeguarding sensitive information such as user credentials and financial transactions.

Types of Databases:

SQL Databases:

Relational Data Model: SQL databases follow a relational data model, where data is organized into tables with predefined schemas, and relationships between tables are established using keys.

ACID Transactions: SQL databases ensure data integrity through ACID (Atomicity, Consistency, Isolation, Durability) transactions, providing reliable and consistent data management.

Structured Query Language (SQL): SQL databases utilize SQL queries for data manipulation, including operations such as SELECT, INSERT, UPDATE, and DELETE, making it easier to retrieve and modify data.

NoSQL Databases:

Schema-less Design: NoSQL databases offer flexibility with a schema-less design, allowing for dynamic and unstructured data storage, which is suitable for handling diverse data types and formats.

Scalability: NoSQL databases are highly scalable, capable of handling large volumes of data and

accommodating the growth of the college community and its associated data.

High Performance: NoSQL databases prioritize performance and scalability, making them ideal for real-time data processing and handling high concurrency.

Relevance to the Project:

SQL Databases: SQL databases like MySQL or PostgreSQL may be suitable for managing structured data such as user profiles, transaction records, and cafeteria orders. They ensure data consistency and integrity, which are essential for financial transactions and user authentication.

NoSQL Databases: NoSQL databases like MongoDB or Redis can be utilized for managing unstructured data such as notifications, chat messages, and user preferences. Their flexibility and scalability are beneficial for handling diverse data types and accommodating the dynamic nature of college operations.

User Profiles and Authentication:

A SQL database is utilized to store user profiles, including student, faculty, and administrative information.

User authentication data such as usernames, passwords (hashed for security), email addresses, and roles are stored in the database.

This database ensures secure access to the platform and manages user permissions based on their roles.

Cafeteria Orders and Transactions:

A NoSQL database stores transaction records related to cafeteria orders, including details such as order IDs, items ordered, quantities, prices, and timestamps.

This database facilitates efficient tracking of cafeteria transactions, enabling users to view their order history and administrators to analyze sales data.

3.1.2 Online Payment Gateway :

In the project aimed at streamlining college operations and enhancing user experience through an integrated web platform, integrating an online payment gateway using Stripe adds significant value by enabling secure and efficient payment processing. Here's a detailed description of how the online payment gateway using Stripe is implemented within the project:

1. Integration Setup:

- The development team sets up a Stripe account and obtains API keys, including publishable and secret keys, required for integrating Stripe with the web platform.

2. Client-Side Integration:

- The frontend development team integrates Stripe's client-side JavaScript library into the web platform's frontend code.
- This library enables the creation of payment forms and handles secure transmission of payment information to Stripe servers.

In our project aimed at enhancing the college community's experience through an integrated web platform, the integration of an online payment gateway using Stripe plays a pivotal role in facilitating seamless and secure transactions for various services offered within the platform. Here's how the implementation of Stripe's payment gateway relates to our project:

3.1.3 Frontend skills :

In our project aimed at enhancing the college community's experience through an integrated web platform, we have utilized various front-end skills to design and develop the user interface (UI) components of the platform. Here's an overview of the front-end skills we have used and how they have been applied in our project:

HTML (Hypertext Markup Language):

HTML forms the backbone of our web platform's structure, defining the layout and content hierarchy of each page.

We have utilized HTML to create semantic markup for different sections, including headers, navigation menus, content areas, and footer.

CSS (Cascading Style Sheets):

CSS is instrumental in styling and visually enhancing the appearance of our web platform.

We have applied CSS styles to HTML elements to achieve consistent typography, colors, spacing, and layout across all pages.

CSS frameworks like Bootstrap or Tailwind CSS may have been utilized to expedite styling and

ensure responsiveness across various devices.

JavaScript:

JavaScript adds interactivity and dynamic behavior to our web platform, enhancing user engagement and functionality.

We have utilized JavaScript to implement client-side form validation, ensuring data integrity and providing real-time feedback to users.

Dynamic features such as dropdown menus, modal dialogs, and interactive elements may have been implemented using JavaScript libraries or frameworks like jQuery or Vue.js.

3.1.4 Backend Skills:

In our project aimed at enhancing the college community's experience through an integrated web platform, we have utilized a range of backend skills to develop the server-side logic, manage data, and ensure the functionality and security of the platform. Here's an overview of the backend skills we have used and how they have been applied in our project:

Node.js:

In projects where Node.js is used as the backend runtime environment, we have leveraged its asynchronous, event-driven architecture to handle concurrent connections efficiently.

We have utilized Node.js frameworks like Express.js to streamline route handling, middleware integration, and HTTP request/response processing.

Database Management Systems (DBMS):

We have worked with relational database management systems (RDBMS) like MySQL as well as NoSQL databases like MongoDB. These DBMS are used for storing and managing structured and unstructured data related to user profiles, transactions, notifications, documents, and more.

3.2 Requirement Analysis:

3.2.1 Functional Requirements :

Functional Requirements for the Integrated College Web Platform:

1) User Authentication and Authorization:

- Users should be able to register for accounts using their email addresses or existing social media accounts.
- Registered users should be able to log in securely to access personalized features and services.
- Different user roles (e.g., students, faculty, administrators) should have distinct permissions and access levels.

2) Profile Management:

- Users should be able to create and manage their profiles, including updating personal information, profile pictures, and contact details.
- Profile information should be securely stored and accessible only to authorized users.

3) Cafeteria Ordering System:

- Users should be able to browse menu items offered by the cafeteria and place orders online.
- The system should support customization of orders (e.g., special instructions, dietary preferences).
- Users should be able to make secure payments for their cafeteria orders through integrated payment gateways.

4) Laundry Notification System:

- Users should receive notifications regarding the status of their laundry, including when clothes are ready for pickup.

5) Attendance Recording:

- Teachers should be able to record attendance for their classes through the web platform.
- The system should allow teachers to mark students as present, absent, or tardy for each class session.
- Attendance records should be time stamped and securely stored for future reference.

3.2.2 Non-Functional Requirements :

Performance:

The web platform should provide fast response times and load pages quickly to ensure a smooth user experience.

Response times for critical actions such as login, accessing documents, and placing orders should be within acceptable limits, even during peak usage periods.

Scalability:

The platform should be scalable to accommodate a growing number of users, documents, and transactions over time.

It should be capable of handling increased loads without degradation in performance or reliability.

Reliability:

The platform should be highly reliable, with minimal downtime and disruptions to services.

It should have built-in redundancy and failover mechanisms to ensure continuous availability, even in the event of server failures or network outages.

Security:

The platform should adhere to industry best practices for security to protect user data and sensitive information.

It should use encryption protocols to secure data transmission, enforce access controls to prevent unauthorized access, and regularly audit for vulnerabilities.

Usability:

The platform should be intuitive and easy to use, with a user-friendly interface that requires minimal training for users to navigate.

It should follow accessibility standards to ensure that users with disabilities can access and interact with the platform effectively.

Compatibility:

The platform should be compatible with a wide range of web browsers and devices, including

desktops, laptops, tablets, and smartphones.

It should render consistently across different browsers and screen sizes, ensuring a consistent user experience for all users.

3.3 Implementation Tools:

- Languages used:
 - HTML
 - CSS
 - JavaScript
- Frameworks Used:
 - React
 - Tailwind CSS
 - Bootstrap
 - EJS
- Database:
 - NoSQL database: MongoDB
 - SQL database: MySQL
- Online payment processing platform: Stripe

3.4 Solution Approach

3.4.1 WebPortal

User Login with JSON Web Token (JWT):

User initiates the login process by entering their credentials (typically username/email and password).

The backend server validates the credentials.

If the credentials are valid, the server generates a JWT containing relevant information about the user (such as user ID, role, etc.). The JWT is signed using a secret key known only to the server.

Email Authentication:

Once the user is logged in and the JWT is generated, the server triggers an email authentication process. This involves sending an email to the user's registered email address with a unique authentication link. The email contains a link with a token or a

unique identifier that the server can use to verify the authenticity of the user. The email may also include instructions for the user to follow to complete the authentication process.

Student Login:

Students access the login page of the system.

Enter username (e.g., student ID) and password.

PHP backend validates the credentials against the database.

Upon successful validation, the student is authenticated and redirected to their dashboard.

Teacher Login:

Teacher accesses the login page of the system.

Enter username (e.g., teacher ID) and password.

PHP backend validates the credentials against the database.

Upon successful validation, the teacher is authenticated and redirected to their dashboard.

Teacher Marks Attendance:

Teacher navigates to the attendance section on their dashboard.

Selects the class/course for which attendance needs to be marked.

PHP backend retrieves the list of students enrolled in the selected class/course from the database.

Teachers mark attendance for each student, typically by selecting options like present, absent, or late.

PHP backend updates the attendance records in the database.

Student Views Attendance:

Students navigate to the attendance section on their dashboard.

PHP backend retrieves the attendance records for the student from the database based on their ID.

3.4.2 Laundry Updates

Sending Emails using Nodemailer:

The backend server uses Nodemailer, a module in Node.js for sending emails.

Configuration: Set up Nodemailer with the SMTP server details (such as host, port,

username, password).

Compose Email: Create an email template or message content to be sent to the user.

Send Email: Use Nodemailer's sendMail function to send the email to the user's email address.

3.4.3 Cafeteria

Admin Interface:

Admins access a separate dashboard with options to manage cafeteria items (add, edit, delete).

Admin authentication is required to access the admin interface.

The admin interface interacts with the backend to perform CRUD operations on cafeteria items

User Interface:

Users browse cafeteria items available for purchase.

They can view details of each item, including name, price, and description.

Users can add items to their cart for purchase.

Payment Process:

When users are ready to checkout, they proceed to the payment stage.

They provide payment details, such as credit/debit card information

3.4.4 Photocopy Shop

Send a custom crafted email with all the necessary customizations for printing along with attachments using Multer and Nodemailer.

Chapter - 4 Modelling and Implementation Details

4.1 Design Diagrams

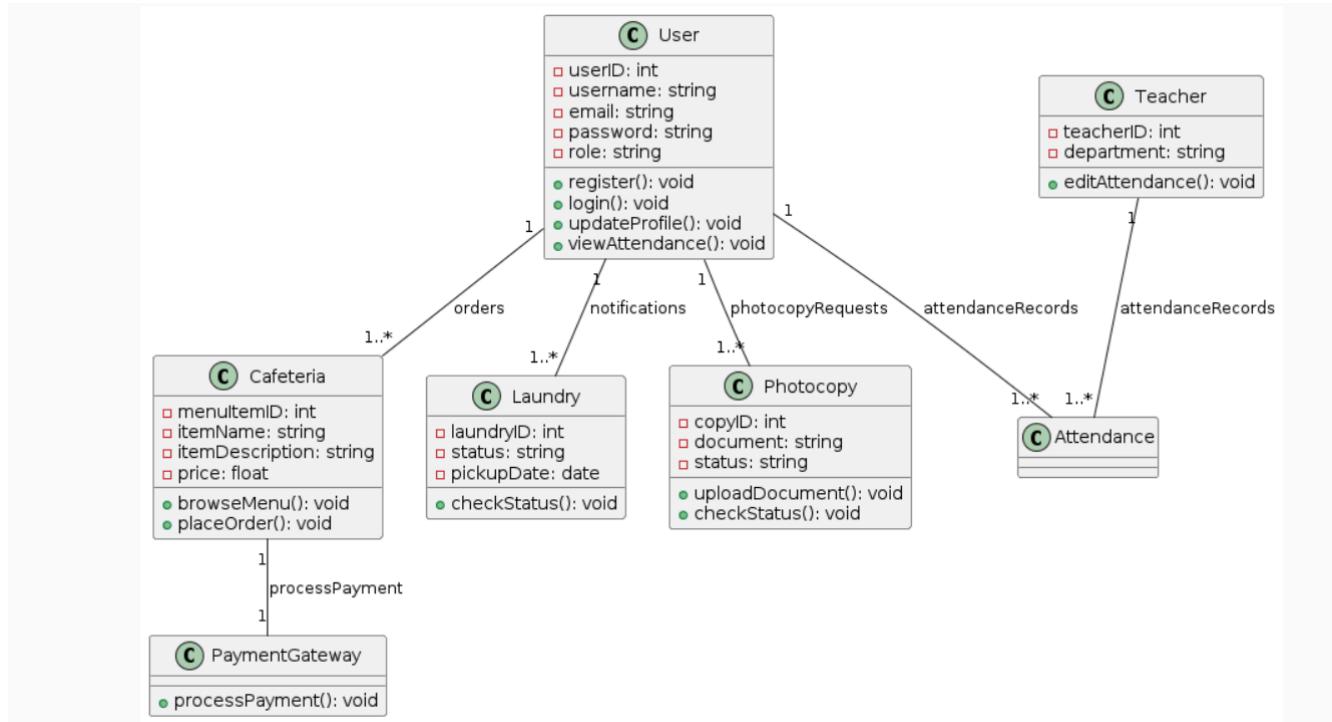


Fig 4.1.1 Class Diagram

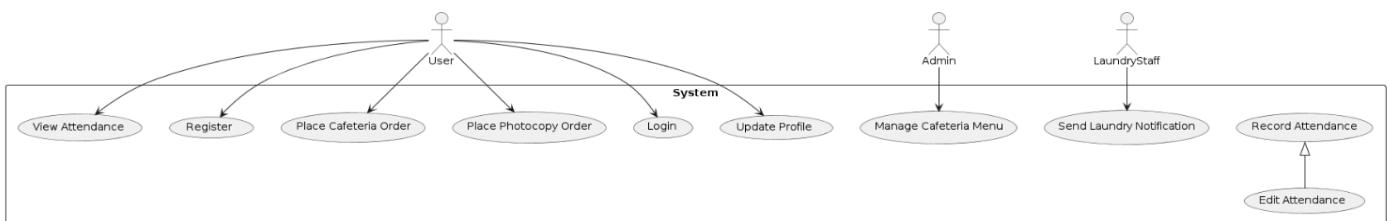


Figure 4.1.2 Use Case Diagram

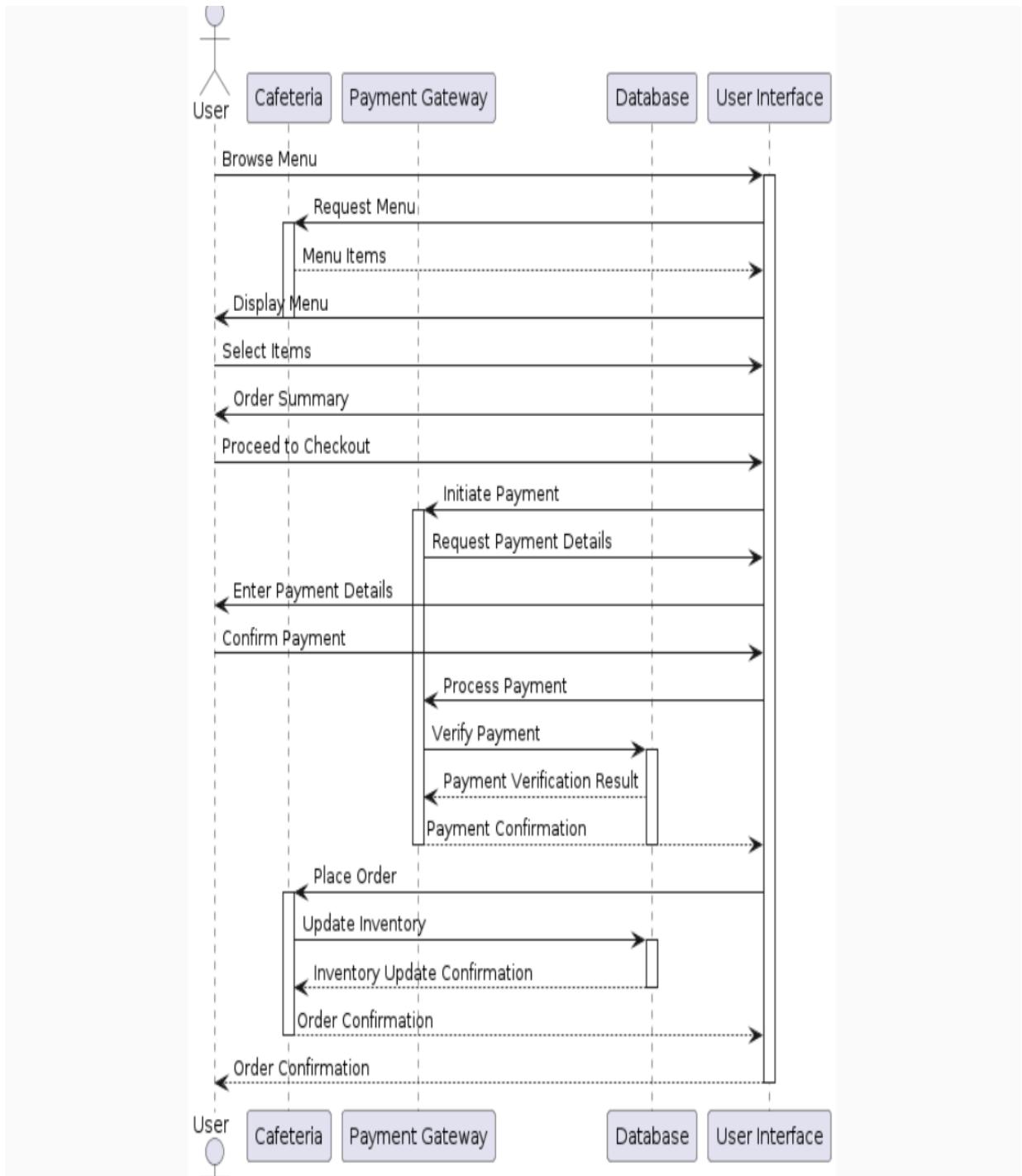


Fig 4.1.3 Sequence Diagram of Cafeteria Services

4.2 Implementation Details and Features Built:

Student and Teacher Login:

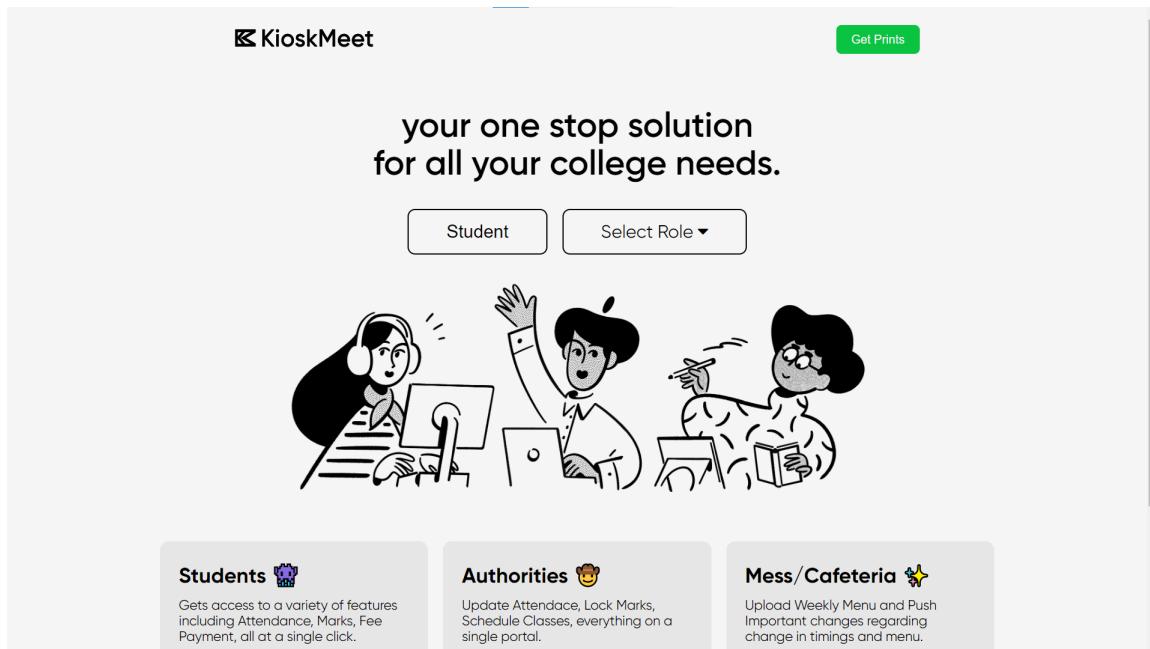


Figure 4.2.1 Website View

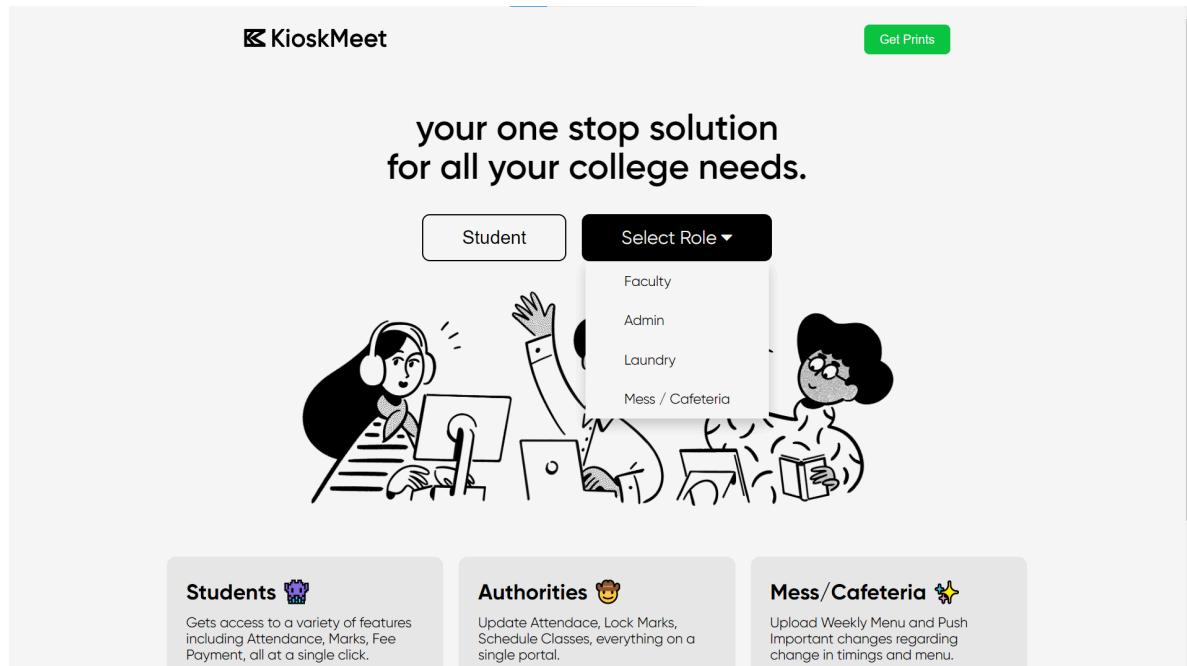


Figure 4.2.2 Select Role

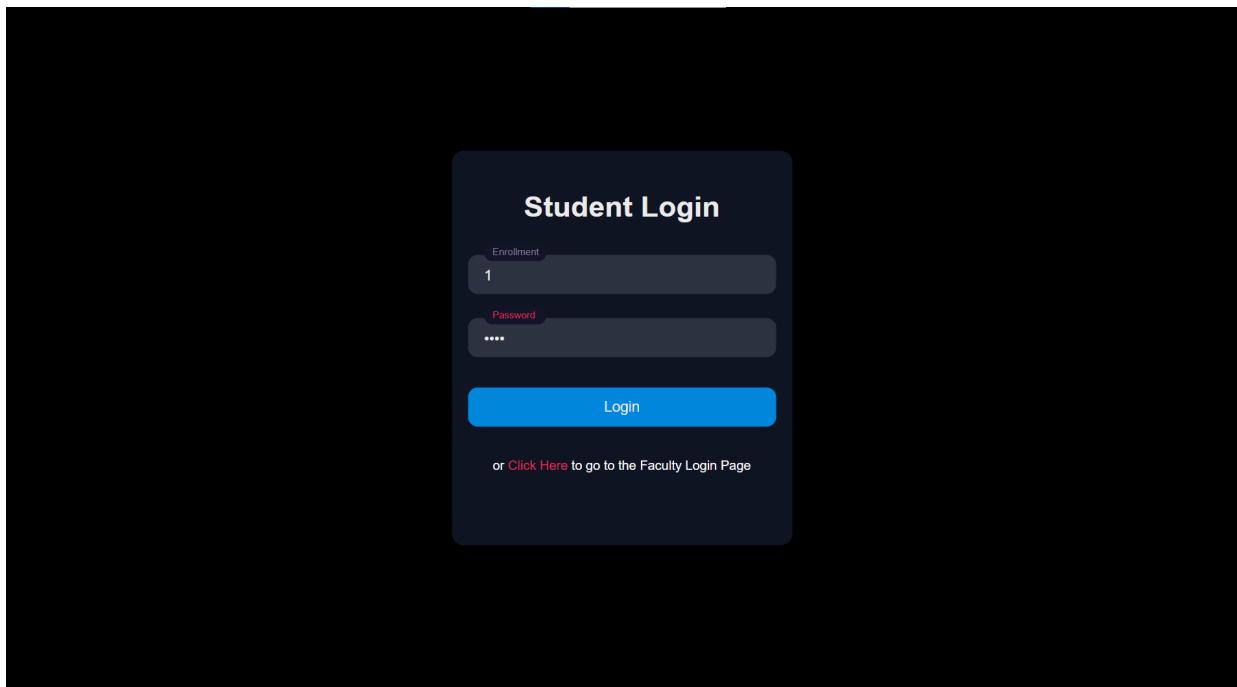


Figure 4.2.3 Student Login

A screenshot of a mobile application titled "KioskMeet". The top navigation bar includes icons for back, profile, attendance, result, and logout. The main content area shows a "Status: Logged in" message above a table with student information. The table rows are: name (Aryan Gupta), dob (2003-05-13), semester (6), batch (B14), and course (CSE). The background features a cartoon illustration of three students and a dog.

Figure 4.2.4 Student Profile

S. NO.	SUBJECT	SEMESTER	GRADE AWARDED
1	Cloud Systems	6	B
1	Data Mining	6	B

Figure 4.2.5 - Subject View

Add Student

Add Teacher

View/Edit Existing Student Data

View/Edit Existing Teacher Data

Figure 4.2.6 Admin Dashboard

ADD STUDENT

ID:

Name:

DOB: dd-mm-yyyy

Semester:

Batch:

Course:

Set Password:

Figure 4.2.7 Add Students - Admin Dashboard

ID	NAME	PHONE NO.	DEPT	EDIT
1	Arpita Jadhav Bhatt	99999999	CSE	<input type="button" value="Edit"/>
2	Tribhuwan Kumar Tewari	987654321	ECE	<input type="button" value="Edit"/>
3	Monali	123455432	BIO	<input type="button" value="Edit"/>
4	Harleen Kaur	2147483647	IT	<input type="button" value="Edit"/>

Figure 4.2.8 View Existing Teachers - Admin Dashboard

Laundry Updates:

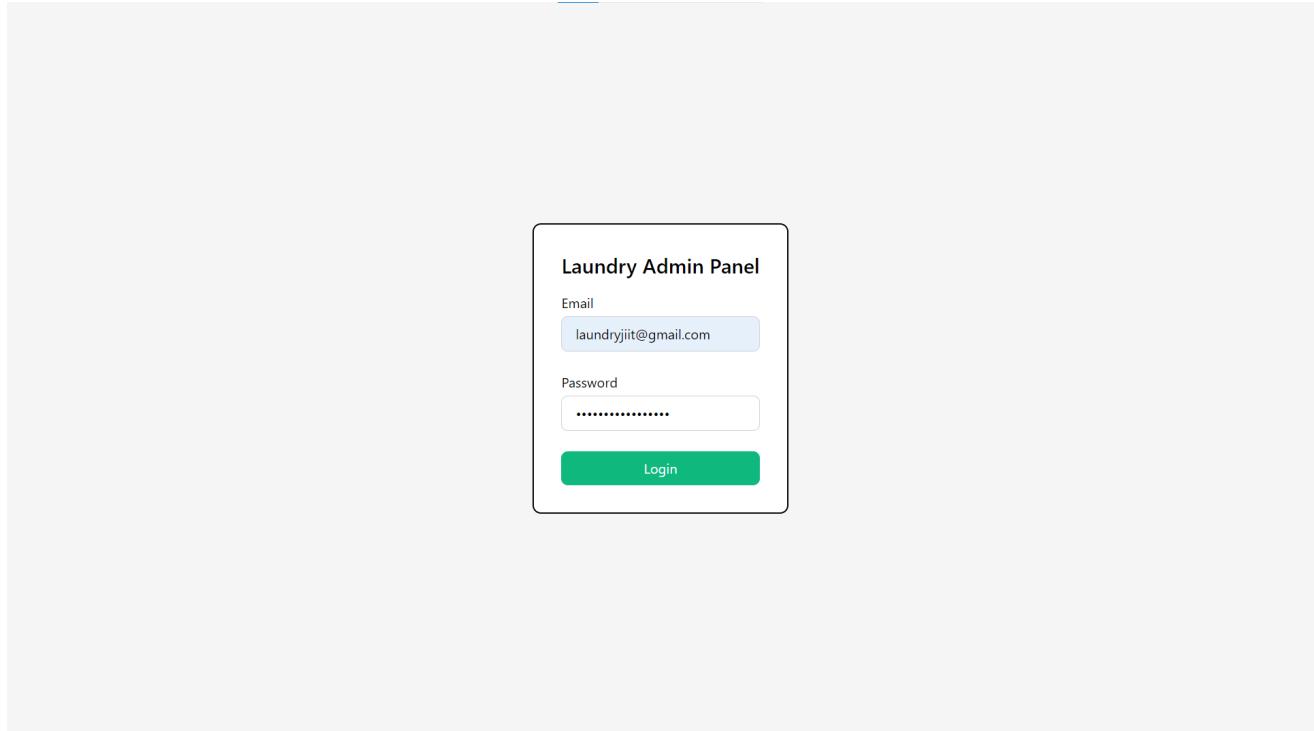


Figure 4.2.9 Laundry Admin Panel Login

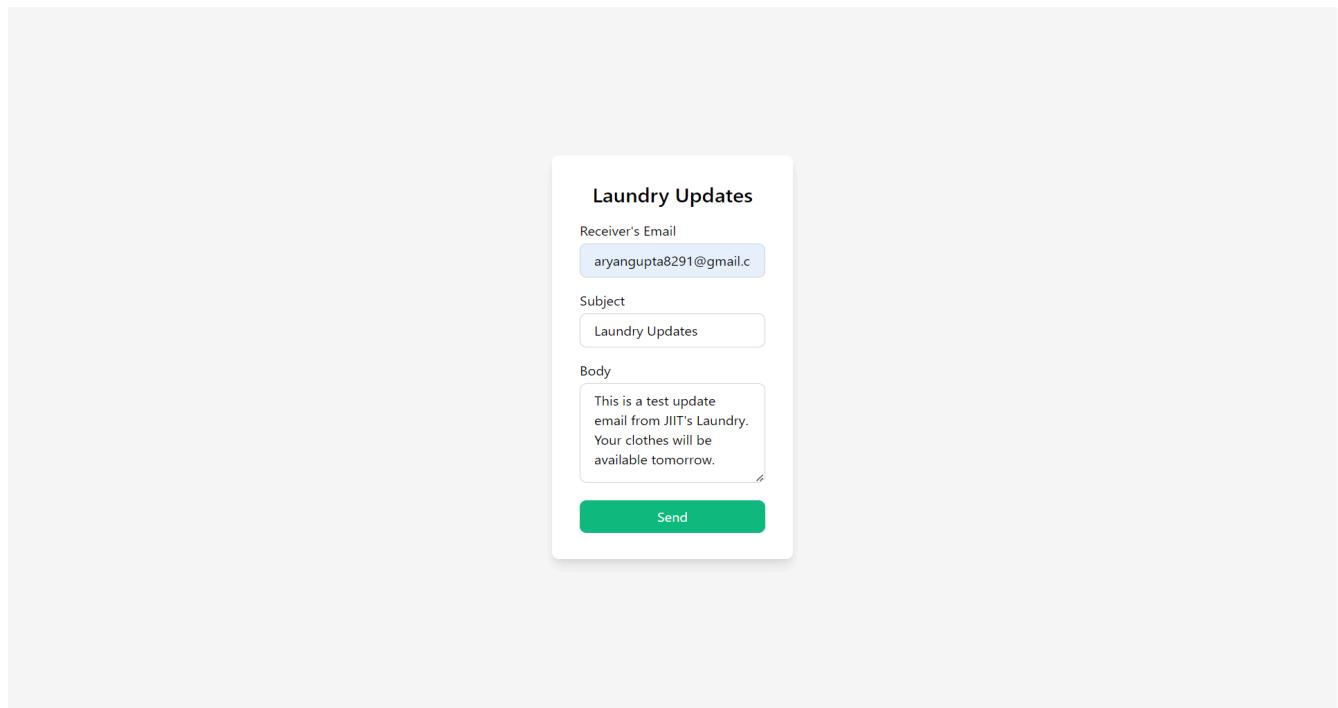


Figure 4.2.10 Laundry Admin Panel - Email Sending Dashboard



Email Sent Successfully

Figure 4.2.11 Email Confirmation

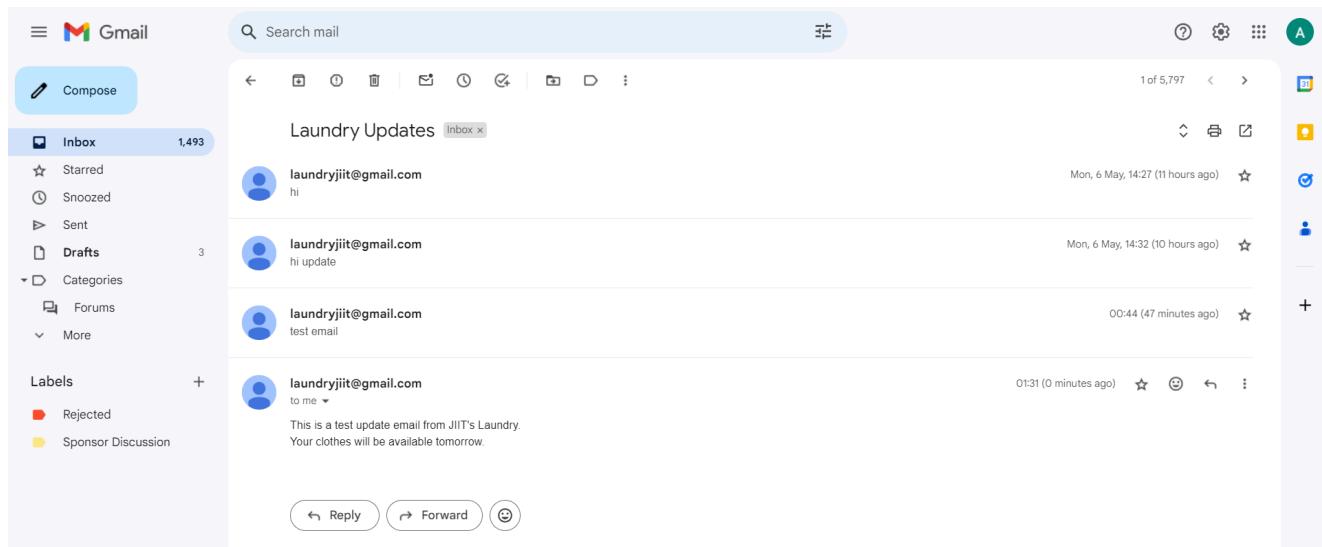


Figure 4.2.12 Emails Received

Cafeteria:

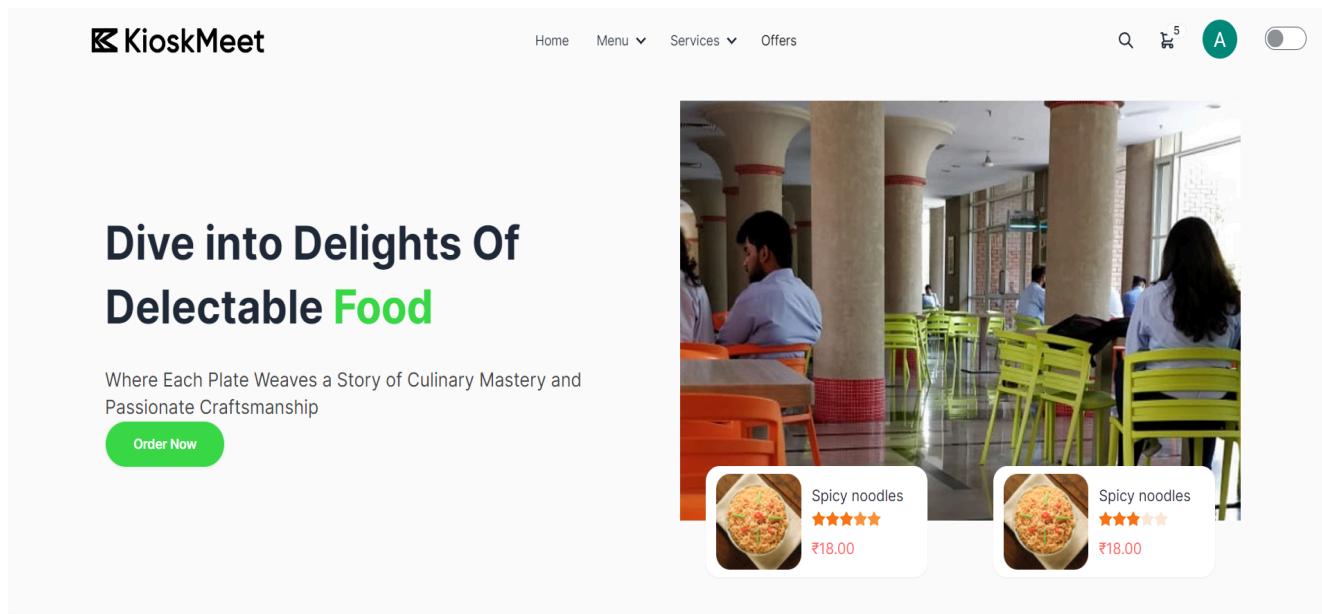


Figure 4.2.13 Cafeteria View

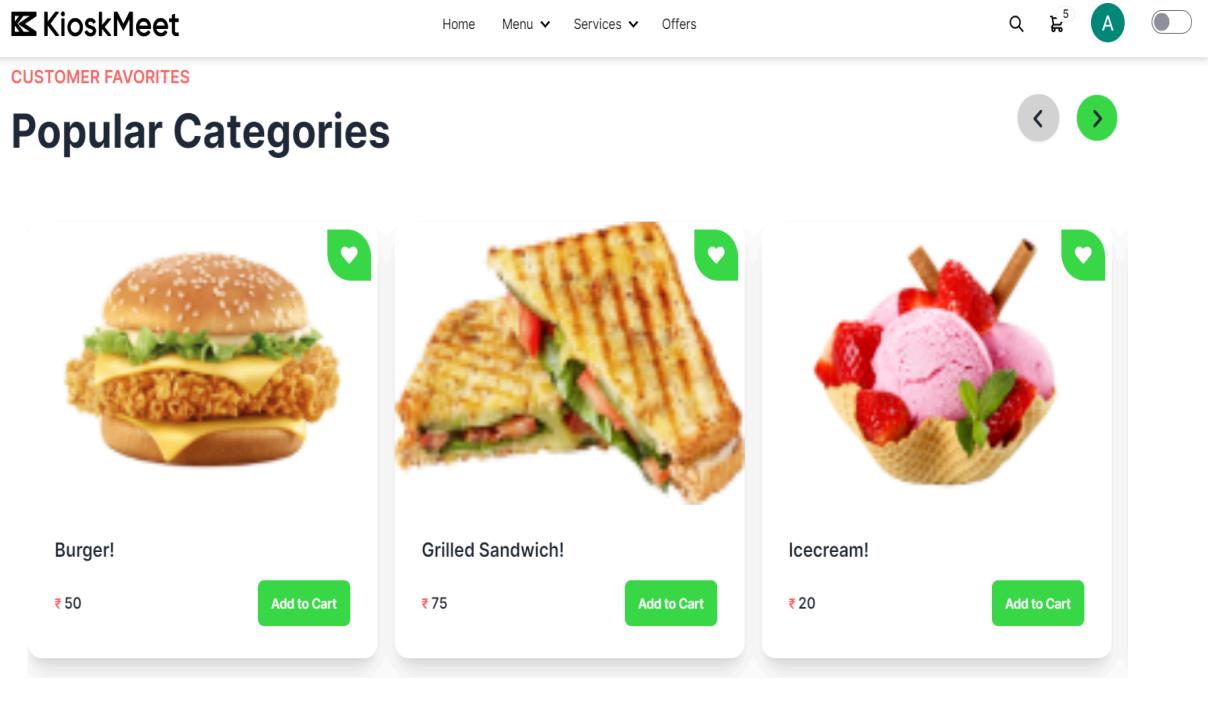


Figure 4.2.14 Menu Items

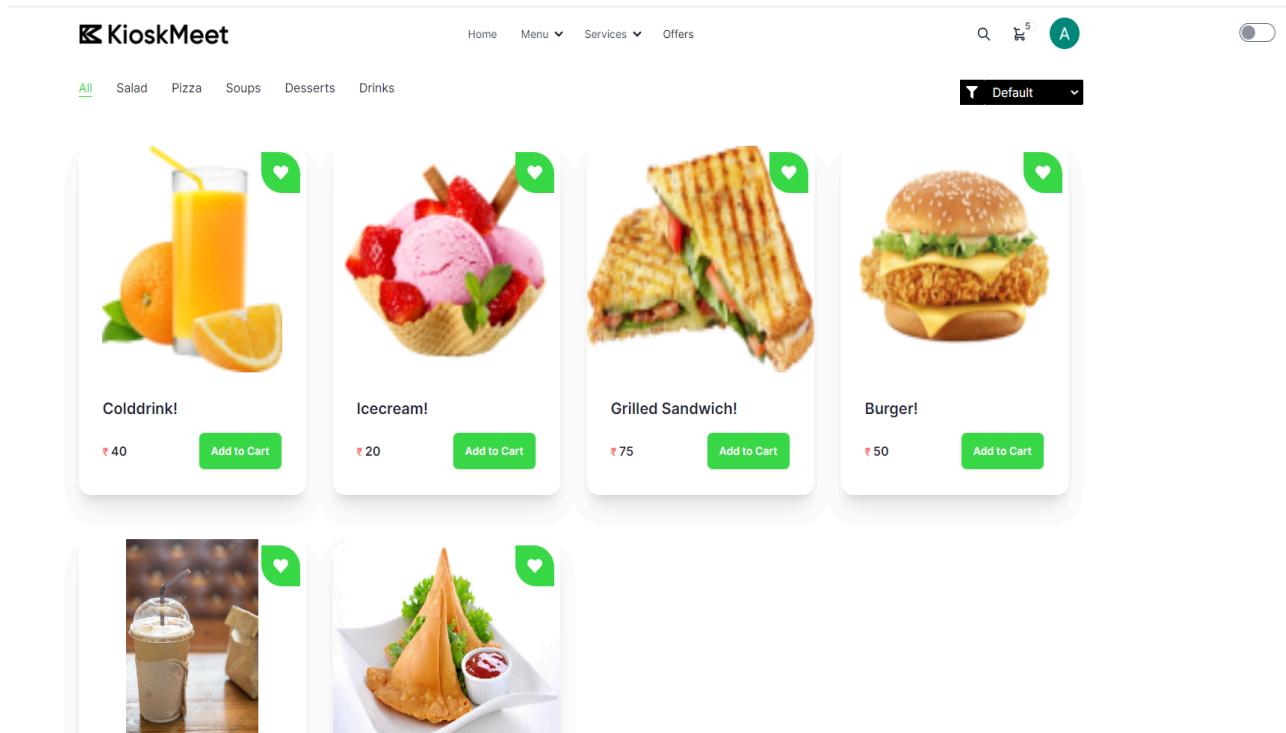


Figure 4.2.15 Additional Menu Items

Manage All Menu Items!

#	Image	Item Name	Price	Update	Delete
1		Colddrink	₹40		
2		Icecream	₹20		
3		Grilled Sandwich	₹75		
4		Burger	₹50		
5		Cold Coffee Frappe	₹40		
6		Samosa	₹10		

[Previous](#) [Next](#)

Figure 4.2.16 Manage Items from Admin Panel

All Users

Total Users: 2

#	Name	Email	Role	Action
1	Sarthak Chauhan	sarthakchauhan093@gmail.com		
2	Aryan Gupta	aryangupta8291@gmail.com	Admin	

Figure 4.2.17 Manage Users as Admin

KioskMeet Admin

Upload A New Menu Item

Recipe Name*

Category*

Price*

Recipe Details

Bio

CHOOSE FILE No file chosen

Add Item !

French Fries is added to the menu.

Figure 4.2.18 Add Item To Menu As Admin

KioskMeet

Home Menu Services Offers

Items Added to The Cart

#	Food	Item Name	Price	Action
1		Cold Coffee Frappuccino	₹40.00	
2		Samosa	₹30.00	
3		Grilled Sandwich	₹225.00	
4		Icecream	₹80.00	
5		Colddrink	₹40.00	

Figure 4.2.19 Delete Items from Cart

The screenshot shows the KioskMeet website with the following interface elements:

- Header:** KioskMeet logo, Home, Menu, Services, Offers, search icon, cart icon (2 items), font size icon, and a toggle switch.
- Title:** Items Added to The Cart
- Table:** A table listing items in the cart.

#	Food	Item Name	Quantity	Price	Action
1		Samosa	<input type="button" value="-"/> 3 <input type="button" value="+"/>	₹30.00	
2		Grilled Sandwich	<input type="button" value="-"/> 3 <input type="button" value="+"/>	₹225.00	
- Customer Details:** Name: Aryan Gupta, Email: aryangupta8291@gmail.com, User_Id: OtdlDoX7KyPBXxV6bGyDQMXaR6i2
- Shopping Details:** Total Items: 2, Total Price: ₹255.00
- Buttons:** Proceed to Checkout (green button) and a large white button below it.

Figure 4.2.20 View Cart

The screenshot shows the KioskMeet website with the following interface elements:

- Header:** KioskMeet logo, Home, Menu, Services, Offers, search icon, cart icon (2 items), font size icon, and a toggle switch.
- Order Summary:** Total Price: ₹85, Number of Items: 2.
- Payment Screen:** A modal window titled "Process your Payment!" containing:
 - Credit/Debit Card input field showing sample card number: 4242 4242 4242 4242, expiration date: 08 / 28, and CVV: 600.
 - A large blue "Pay" button.
 - An orange "Pay with Paypal" button.

Figure 4.2.21 Checkout Screen

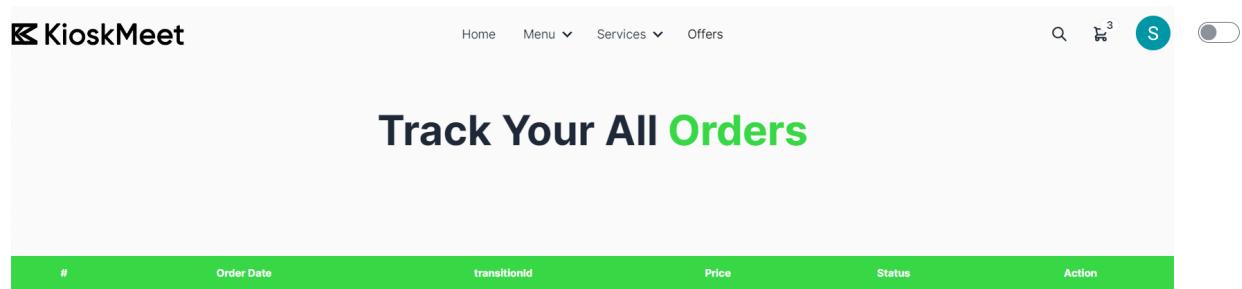


Figure 4.2.22 Tracking Orders

A login form titled "Please Login!". It contains fields for "Email" (with placeholder "email") and "Password" (with placeholder "password"). Below the password field is a "Forgot password?" link. A large green "Login" button is centered at the bottom. Below the button, text reads "Donot have an account? [Signup Now](#)". At the very bottom are three social media sharing icons for Google+, Facebook, and Twitter.

Figure 4.2.23 User Gets Logged Out When Tries To Access Admin Dashboard

Figure 4.2.24 MongoDB Database for User View

BRAND	NUMBER	CVC	DATE
Visa	4242 4242 4242 4242	Any 3 digits	Any future date
Visa (debit)	4000 0566 5566 5556	Any 3 digits	Any future date
Mastercard	5555 5555 5555 4444	Any 3 digits	Any future date
Mastercard (2-series)	2223 0031 2200 3222	Any 3 digits	Any future date
Mastercard (debit)	5200 8282 8282 8210	Any 3 digits	Any future date

Figure 4.2.25 Stripe Test Payment keys

Photocopy View:

The screenshot shows a user interface titled "Photocopy Shop". It includes fields for "Sender's Email" (with placeholder "Enter from email"), "Sender's Password" (with placeholder "Enter password"), "Receiver's Email" (with placeholder "Enter to email"), "Subject" (with placeholder "Enter subject"), and "Body" (with placeholder "Enter email body"). Below these are "Print Options" (set to "Black & White"), "Pages per print" (set to "1"), and "Print Type" (set to "Single Sided"). At the bottom is a "Upload Files" button.

Figure 4.2.26 Getting Details from user for Printouts

Chapter - 6 Findings and Conclusion

6.1 Findings

- The combination of NodeJs and ExpressJS was found to be more robust and scalable as compared to its counterpart in PHP and MySQL.
- Stripe Test API produces authentication issues in Test Mode due to guidelines of Government of India requiring the completion of KYC of the demanding individual.

6.2 Conclusion

In crafting KioskMeet as the ultimate solution for all college needs, we've meticulously addressed every facet of student and faculty life. Through the seamless integration of functional and non-functional requirements, we've constructed a digital ecosystem where convenience and efficiency converge.

From the moment users authenticate, they step into a world tailored to their roles and needs.

Profile management empowers individuals to curate their digital identities securely, while our cafeteria ordering system brings the entire dining experience online, offering both choice and convenience at their fingertips. Meanwhile, our laundry notification system ensures that the mundane task of laundry becomes a hassle-free experience, with timely updates on the status of their clothes.

For educators, attendance recording becomes a breeze through our web platform, streamlining administrative tasks and freeing up valuable time for meaningful engagement with students. And behind the scenes, our platform boasts impeccable performance, scalability, reliability, security, usability, and compatibility, ensuring that every interaction is smooth, secure, and accessible across devices and browsers.

As we conclude this journey, KioskMeet stands not just as a project but as a testament to our dedication to enhancing the college experience. With its comprehensive suite of features and unwavering commitment to excellence, KioskMeet truly embodies the ethos of a one-stop solution for all your college needs. Welcome to the future of campus life, where convenience, innovation, and empowerment intersect seamlessly.

References:

- [1] PHP v/s Javascript [as on 5th May, 2024]
<https://www.cloudways.com/blog/php-vs-javascript/>
- [2] <https://docs.stripe.com/testing>
- [3] <https://www.turing.com/kb/comprehensive-guide-to-sending-an-email-using-node-mailer> [as on 54th May, 2024]
- [4] <https://www.npmjs.com/package/nodemailer>
- [5] Design Inspiration
<https://www.figma.com/file/bD49Zi9jedJfHPkvAMGJfu/Figma-File?type=design&node-id=0%3A1&mode=design&t=mYV6PNNmZFeJOghe-1>