

College Placement Management System

Submitted for the Partial Fulfillment of the Requirements for the degree of
Bachelor of Technology
in
Computer Science and Engineering
By

Yogesh Nade
UI22CS51

Vikas Sharma
UI22CS81

Datt Patel
UI22CS58



To

Ms. Nancy Sukhadia
&
Ms. Ankita Parmar

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SURAT-394190
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

April, 2024

College Placement Management System

Submitted for the Partial Fulfillment of the Requirements for the degree of
Bachelor of Technology

in

Computer Science and Engineering

By

Yogesh Nade
UI22CS51

Vikas Sharma
UI22CS81

Datt Patel
UI22CS58



To

Ms. Jiby Babin
&
Mr. Rishi Sharma

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SURAT-394190
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

April, 2024

Indian Institute of Information Technology Surat
Computer Science and Engineering Department



CERTIFICATE

This is to certify that candidate Datt Patel bearing Roll No: UI22CS58 of B.Tech. IV, 4th Semester has successfully carried out the work on “College Placement Management System” for the partial fulfillment of the degree of Bachelor of Technology(B.Tech.)

1. Subject Co-ordinator 1: Ms. Nancy Sukhadia : Sign:

2. Subject Co-ordinator 2: Ms. Ankita Parmar : Sign:

Indian Institute of Information Technology Surat
Computer Science and Engineering Department



CERTIFICATE

This is to certify that candidate Yogesh Nade bearing Roll No: UI22CS51 of B.Tech. IV, 4th Semester has successfully carried out the work on “College Placement Management System” for the partial fulfillment of the degree of Bachelor of Technology(B.Tech.)

1. Subject Co-ordinator 1: Ms. Nancy Sukhadia : Sign:

2. Subject Co-ordinator 2: Ms. Ankita Parmar : Sign:

Indian Institute of Information Technology Surat
Computer Science and Engineering Department



CERTIFICATE

This is to certify that candidate Vikas Sharma bearing Roll No: UI22CS86 of B.Tech. IV, 4th Semester has successfully carried out the work on “College Placement Management System” for the partial fulfillment of the degree of Bachelor of Technology(B.Tech.)

1. Subject Co-ordinator 1: Ms. Nancy Sukhadia : Sign:

2. Subject Co-ordinator 2: Ms. Ankita Parmar : Sign:

Indian Institute of Information Technology Surat
Computer Science and Engineering Department



CERTIFICATE

This is to certify that candidate Datt Patel bearing Roll No: UI22CS58 of B.Tech. IV, 4th Semester has successfully carried out the work on “College Placement Management System” for the partial fulfillment of the degree of Bachelor of Technology(B.Tech.)

1. Subject Co-ordinator 1: Ms. Jiby Babin : Sign:.....

2. Subject Co-ordinator 2: Mr. Rishi Sharma : Sign:.....

Indian Institute of Information Technology Surat
Computer Science and Engineering Department



CERTIFICATE

This is to certify that candidate Yogesh Nade bearing Roll No: UI22CS51 of B.Tech. IV, 4th Semester has successfully carried out the work on “College Placement Management System” for the partial fulfillment of the degree of Bachelor of Technology(B.Tech.)

1. Subject Co-ordinator 1: Ms. Jiby Babin : Sign:.....

2. Subject Co-ordinator 2: Mr. Rishi Sharma : Sign:.....

Indian Institute of Information Technology Surat
Computer Science and Engineering Department



CERTIFICATE

This is to certify that candidate Vikas Sharma bearing Roll No: UI22CS86 of B.Tech. IV, 4th Semester has successfully carried out the work on “College Placement Management System” for the partial fulfillment of the degree of Bachelor of Technology(B.Tech.)

1. Subject Co-ordinator 1: Ms. Jiby Babin : Sign:.....

2. Subject Co-ordinator 2: Mr. Rishi Sharma : Sign:.....

DECLARATION

This is to certify that

- (i) This report comprises my original work towards the degree of Bachelor of Technology in Computer Science and Engineering at Indian Institute of Information Technology (IIIT) Surat and has not been submitted elsewhere for a degree.
- (ii) Due acknowledgement has been made in the text to all other material used.

Yogesh Nade

Vikas Sharma

Datt Patel

ACKNOWLEDGEMENTS

I would like to express my gratitude and appreciation to all those who gave me the possibility to complete this Project Report. Though I have made efforts for completion of the project, it would not have been possible without the support and guidance of many individuals and my team. I would like to extend my sincere thanks to all of them.

Special thanks are due to my Faculty Mentor at IIIT Surat - Ms. Nancy Sukhadia, Ms. Jiby Babin whose help, stimulating suggestions, and encouragement helped me a lot while working on this project. I extend my deepest gratitude to our esteemed college director Prof. J. S. Bhat for their unwavering commitment to excellence, visionary leadership, and dedication to fostering a thriving learning environment. I am indebted to friends and family for their utmost support and optimistic approach which let me work with sheer determination and focus throughout the entire time. I extend my sincere thanks to all of them who supported me, guided me throughout the duration of this semester.

ABSTRACT

Focus on automating conventional training and placement management systems. Application for Training & Placement Officers in colleges. Manages student information for placements and provides assistance through a portal. Students can post queries to TPOs and coordinators via the assistance portal. Student login allows updating personal and educational information. Information is added to the database, and students can upload resumes. Provides preparation materials for placements. Includes a Company Tab to assist companies in shortlisting eligible students. Reduces manual work, minimizes paperwork, and enhances efficiency.

Contents

Certificate ii Declaration iii Acknowledgements iv Abstract

1. Introduction to project

1.1 Introduction	1
1.1.1 What the system will do	1
1.2 Scope	1
1.3.Out of scope	1
1.4 Functional Requirements	2
1.4.1 User registration and Authentication:.	2
1.4.2 User Profiles	2
1.4.3 Search and filter	2
1.4.4 User(seller)	2
1.5 Non functional requirements	3
1.5.1 Performance	3
1.5.2 Security	3
1.5.3 Reliability	3
1.5.4 Usability	3
1.5.1 Performance	3

2. Tools/Technologies

2.1 Technologies	4
2.1.1 MERN stack with Redux	4
2.1.2 firebase.	4
2.1.3 Git	4
2.1.4 Insomnia	5

2.2 Libraries	6
2.2.1 js-cookie	6
2.2.2 Tailwind CSS	6

3. Proposed Systems

3.1 Required Software Specifications	7
3.1.1 Operating System	7
3.1.2 Web Browser	7
3.1.3 Database Management System	7
3.1.4 Node.js Runtime Environment	7
3.1.5 Development Tools	8
3.2 Hardware Specifications	8
3.2.1 Processor	8
3.2.2 Memory (RAM)	8
3.2.3 Storage	8
3.2.4 Network Connectivity	9

4. Design

4.1 System Architecture	10
4.2 System Design	10
4.3 Diagrams	11

5. Implementation

5.1 Development Environment	17
5.2 Coding Practices	17
5.3 Integration	18

5.4 Work demo	19
-------------------------	----

6. Testing and Experimental Results

6.1 Testing Methodology	26
6.1.1 Unit Testing	26
6.1.2 Integration Testing	28
6.1.3 Functional Testing	28
6.1.4 Performance Testing	31
6.1.5 Security Testing	31
6.1.6 User Acceptance Testing (UAT)	32
6.2 Experimental Results	32
6.2.1 Unit Testing Results	32
6.2.2 Integration Testing Results	33
6.2.3 Functional Testing Results	33
6.2.4 Performance Testing Results	33
6.2.5 Security Testing Results	33
6.2.6 User Acceptance Testing Results	33

List of Figures

4.3	Class Diagram of Placement Management System	
20	4.3 Use Case Diagram of Placement Management System	
22	4.3 Sequence Diagram of Placement Management System	
24	4.3 Deployment Diagram of Placement Management System	
25	4.3 Component Diagram of Placement Management System	
26	6.4 Sending link to the User for joining the Interview Figure	
30	6.4 Figure of Interviewer and Student	
31	9.2 Testing the response in Headers	
33	9.2 Checking the Actual Response	
34	9.2 Checking for Styling	
35	9.2 Local Storage	
35	9.2 Time Taken by the Request	
36	9.2 Lighthouse Result	36

1. Introduction

1.1 About

College Placement Management System converts the placement process into a smooth Digitized process. This Project aim to shift Placement process into Online Mode

1.2. System Overview

The system comprises the following key components:

- User Authentication
- Dashboard
- Registration
- Communication(Group wise)

1.2.1 User Authentication

- Users can register for a new account or log in with existing credentials.
- Authentication mechanisms ensure secure access to the system.

1.2.2 Dashboard

- Provide Notification of Upcoming Companies
- Provide Notification of already registered Companies regarding further Process

1.2.3 Registration

- Student can register if they are full fill the criteria of Company
- Here they have to upload their Detail with resume

1.2.4 Communication(Group wise)

- Admin can communicate with the Student groups base on Company
- He/She can give instruction or suggestion for group of Student who applied in particular company

2. Tools/Technologies

2.1. Programming Languages:

2.1.1 HTML :

Hypertext Markup Language is the standard language used to create web pages. It provides the structure and content of a webpage through a series of markup tags. These tags define the elements within the page such as headings, paragraphs, images, links, forms, and more.

2.1.2 CSS :

CSS, which stands for Cascading Style Sheets, is a style sheet language used to control the presentation, layout, and formatting of HTML documents. It enables web developers to define how HTML elements should appear on a webpage, including aspects such as colors, fonts, spacing, alignment, and more.

2.1.3 JAVASCRIPT :

JavaScript is a versatile and widely used programming language primarily known for its role in web development. It's a core technology for building dynamic and interactive websites and web applications.

2.2.2 NODE JS

Node.js is an open-source, cross-platform JavaScript runtime environment built on Chrome's V8 JavaScript engine. It allows developers to run JavaScript code on the server-side, enabling the development of scalable and high-performance network applications. Node.js was created by Ryan Dahl in 2009 and has since gained widespread adoption due to its lightweight, event-driven architecture and extensive ecosystem of libraries and frameworks.

2.2.3 EXPRESS JS

Express.js, commonly referred to as Express, is a minimal and flexible web application framework for Node.js. It provides a robust set of features for building web and mobile applications, APIs, and server-side applications. Express.js was inspired by the Sinatra framework for Ruby and is known for its simplicity, speed, and opinionated nature.

2.2.4 BOOTSTRAP

Bootstrap is a free and open-source front-end framework for developing responsive and mobile-first web projects. It was originally created by Mark Otto and Jacob Thornton at Twitter and was later released as an open-source project. Bootstrap provides a collection of CSS and JavaScript components, along with pre-built templates and themes, to help developers quickly build modern and visually appealing websites and web applications.

2.3 Database System

2.3.1 MONGODB ATLAS

→ MongoDB is a popular open-source NoSQL database management system that is designed for flexibility, scalability, and performance. Unlike traditional relational databases, MongoDB stores data in flexible, JSON-like documents, making it easy to store and retrieve complex, hierarchical data structures.

→ MongoDB's document-oriented model allows for dynamic schemas, enabling developers to quickly iterate and evolve their data models without the need for predefined schemas. MongoDB is known for its horizontal

scalability, allowing it to handle large volumes of data and high throughput applications with ease.

→It supports features such as automatic sharding, replication, and failover, ensuring data availability and reliability. MongoDB's query language and indexing capabilities enable fast and efficient data retrieval, making it suitable for a wide range of use cases, including web applications, content management systems, real-time analytics, and more.

→Additionally, MongoDB offers a rich ecosystem of tools, libraries, and integrations, making it easy for developers to work with MongoDB in their preferred programming languages and environments. Overall, MongoDB's combination of flexibility, scalability, and ease of use has made it a popular choice

for modern application development.

2.4 Development Tools and Environments

2.4.1 Visual Studio Code

Visual Studio Code (VS Code) is a free, open-source source code editor developed by Microsoft for Windows, macOS, and Linux. It has quickly become one of the most popular code editors in the developer community due to its versatility, performance, and extensive set of features.

2.4.2 Git

Git is a distributed version control system (DVCS). It is designed to track changes to files and coordinate work among multiple contributors in a collaborative environment. Git operates locally on a developer's machine, allowing for fast and efficient operations even with large codebases.

2.4.3 Github

GitHub is a web-based platform built on top of Git that provides hosting for Git repositories and a range of collaboration and project management features. It was founded in 2008 by Chris Wanstrath, PJ Hyett, and Tom Preston-Werner and has since become the largest and most popular platform for hosting open-source projects and collaborating on software

development.

2.5 Cloud Platforms

2.5.1 Vercel

Vercel is a cloud platform designed to help developers deploy and host websites, web applications, and serverless functions with ease. Vercel focuses on providing a seamless developer experience and high-performance hosting infrastructure.

2.5.2 Render

Render is an excellent choice for deploying backend applications, offering managed infrastructure services that simplify the deployment and management of server-side code.

2.6. Security

2.6.1 OpenSSL

OpenSSL is an open-source toolkit implementing the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols, as well as

14

cryptographic functions and utilities. It provides a comprehensive set of tools for encryption, decryption, digital signatures, and certificate management.

→ Encryption: OpenSSL supports various encryption algorithms, including symmetric (e.g., AES, DES) and asymmetric (e.g., RSA, DSA) encryption. It allows developers to encrypt and decrypt data using cryptographic keys, providing confidentiality and privacy for sensitive information.

2.6.2 bcrypt

bcrypt is a password-hashing function designed specifically for securely hashing passwords. It is based on the Blowfish cipher and is widely used in web applications and software systems for password storage.

→ Password Hashing: bcrypt hashes passwords by applying a cryptographic hash function multiple times (known as key stretching) with a salt value to protect against rainbow table attacks and brute-force attacks. This

makes it computationally expensive for attackers to crack hashed passwords, even with the use of powerful hardware and algorithms.

3. External interface requirements

3.1 User interface requirements

- Be intuitive and user-friendly, requiring minimal training for users to navigate. Support multiple languages for global accessibility.
- Incorporate responsive design principles to ensure compatibility across various devices (desktop, mobile, tablet).
- Provide clear instructions and guidance throughout the interview preparation Process.

3.2 Hardware Interface Requirements

Device (Computer/laptop) Requirements:

- Memory (RAM): Minimum 2GB RAM
- Processor: Minimum 1GHZ; Recommended 2 GHZ or more.
- Hard disk: 40 GB; Recommended 64 GB or more.
- Ethernet connection (LAN) or a wireless adapter (Wi-Fi)

3.3 Software Interface Requirements

- Support web browsers such as Chrome, Mozilla Firefox, etc.
- Compatible with Windows, Linux, macOS (32-bit and 64-bit).

3.4 Communication Interface Requirements

- Secure data transmission protocols (e.g., HTTPS) to protect user information during interactions with the system.
- APIs and web services for facilitating communication between the front-end and back-end components of the system.
- Email notifications for account-related activities (e.g., registration confirmation, password reset).
- Real-time chat or messaging functionality for users to interact with mentors or peers during their preparation.

4. Design

4.1 System Architecture

- ⇒ The system architecture of the software is designed to be modular and scalable, allowing for flexibility and extensibility in the future. The architecture

consists of several key components, including:

- **Presentation Layer** : This layer comprises the user interface components responsible for interacting with users, including web pages, forms, and user controls. Technologies such as NextJS and Tailwind CSS are utilized to create responsive and visually appealing user interfaces.
- **Application Layer** : The application layer contains the business logic and processing components of the software. NestJS is employed to develop robust and scalable backend services, handling tasks such as order processing, inventory management, and user authentication.
- **Data Layer** : The data layer is responsible for managing and storing data used by the software. MongoDB, a NoSQL database, is chosen for its flexibility and scalability, allowing for efficient storage and retrieval of structured and unstructured data.

4.2 System Design

⇒ The system design of the software is based on a client-server architecture, where clients interact with the server to access and manipulate data. The following design principles and patterns are employed:

- **Model-View-Controller (MVC)** : The MVC pattern is used to separate the presentation, business logic, and data access layers, promoting modularity and maintainability.
- **Dependency Injection** : Dependency injection is employed to manage component dependencies and promote loose coupling between modules, facilitating easier testing and maintenance.
- **RESTful API** : A RESTful API is designed to provide a standardized interface for clients to interact with the software, enabling seamless integration with external systems and services.

4.3 Diagrams

4.3.1 Class Diagram :

- A class diagram in software engineering is a graphical representation of the classes, interfaces, associations, and collaborations within a system. It's a fundamental aspect of object-oriented design and is commonly used during

the early stages of software development to visualize the structure of the system and its components.

Classes: Classes represent the blueprint for creating objects. They encapsulate data (attributes or properties) and behaviors (methods or functions). Each class is depicted as a rectangle with three compartments: the top compartment contains the class name, the middle compartment contains the attributes, and the bottom compartment contains the methods.

Interfaces: Interfaces define a contract for classes to follow. They specify a set of methods that implementing classes must provide. In a class diagram, interfaces are depicted similar to classes but with a dashed line border.

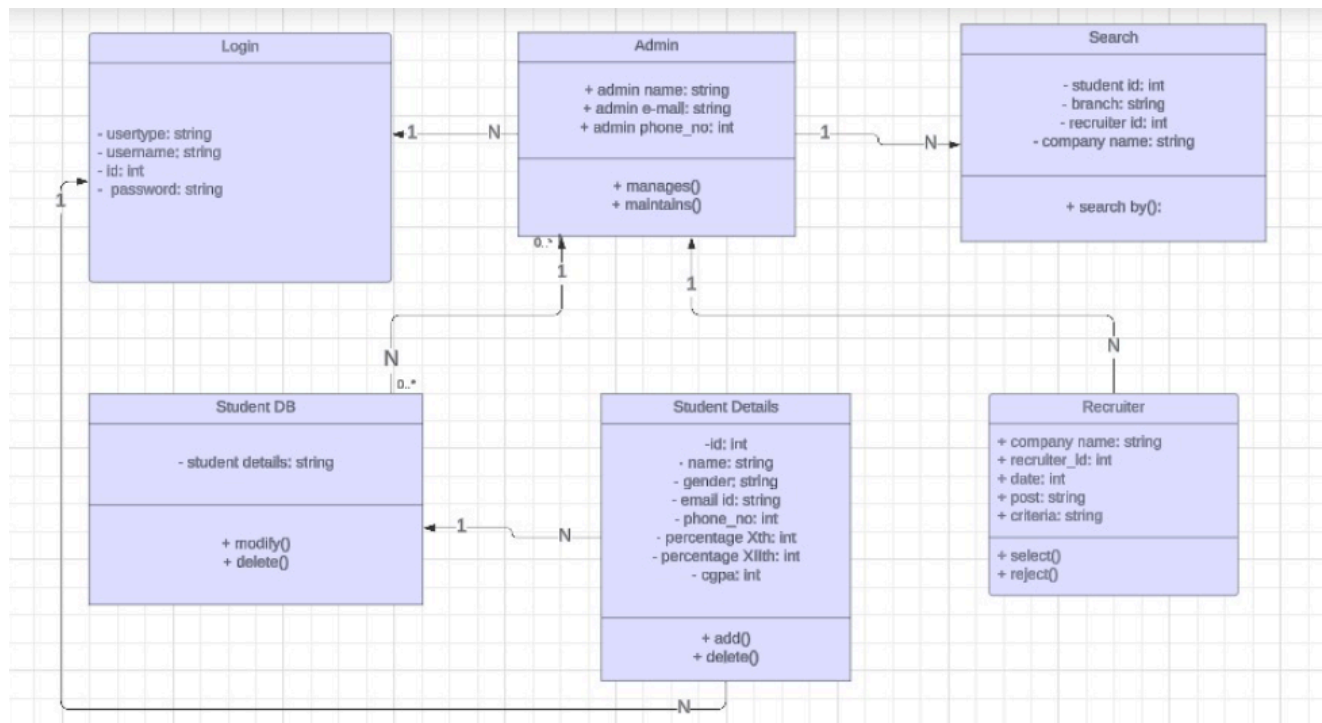
Associations: Associations represent relationships between classes. They indicate how classes are connected or interact with each other. Associations can be one-to-one, one-to-many, or many-to-many. They're represented by a line connecting the related classes, with optional arrows indicating the direction of the relationship.

Inheritance/Generalization : Inheritance denotes an "is-a" relationship between classes, where one class (subclass or derived class) inherits attributes and methods from another class (superclass or base class). It's depicted by a solid line with a hollow triangle pointing from the subclass to the superclass.

Composition: Composition represents a strong "has-a" relationship between classes, where one class owns or contains instances of another class. It's depicted by a solid diamond at the containing class end of the association line.

Aggregation: Aggregation is a weaker form of composition, indicating a "part-of" relationship between classes. Unlike composition, the parts can exist independently of the whole. It's depicted by an empty diamond at the containing class end of the association line.

Dependencies: Dependencies represent a relationship where one class depends on another class in some way. It indicates that a change in one class might affect another class. Dependencies are typically represented by a dashed arrow from the dependent class to the class it depends on.



4.3.2 Use Case Diagram :

- A Use Case diagram is a visual representation of the functional requirements of a system from the perspective of its users.
- **Purpose:** Use Case diagrams are used to capture the functional requirements of a system and depict how users interact with it to achieve certain goals or tasks.
- **Components:**
 - ⇒ **Actors:** Represent users or external systems interacting with the system being modeled. Actors are typically drawn as stick figures.
 - ⇒ **Use Cases:** Represent individual functionalities or tasks that users can perform within the system. Use cases are depicted as ovals or ellipses.

- Relationships :

- ⇒ **Association:** Shows the relationship between actors and use cases, indicating which actors are involved in executing specific use cases. ⇒

Inclusion: Indicates that one use case includes another, meaning the included use case is part of the behavior of the including use case. ⇒

Extension: Represents that one use case can extend another, meaning additional functionality is added to the base use case under certain conditions.

⇒ System Boundary: Use Case diagrams typically include a boundary box to encapsulate the system being modeled, distinguishing it from external actors and systems.

- Use Cases in Software Development :

⇒ Requirement Analysis: Use Case diagrams help stakeholders understand the functional requirements of the system and ensure that all user interactions are considered.

⇒ Communication: They serve as a communication tool between developers, designers, and clients, providing a common understanding of system functionality.

⇒ Design: Use Case diagrams inform system design by identifying major functionalities and guiding the creation of more detailed designs. -

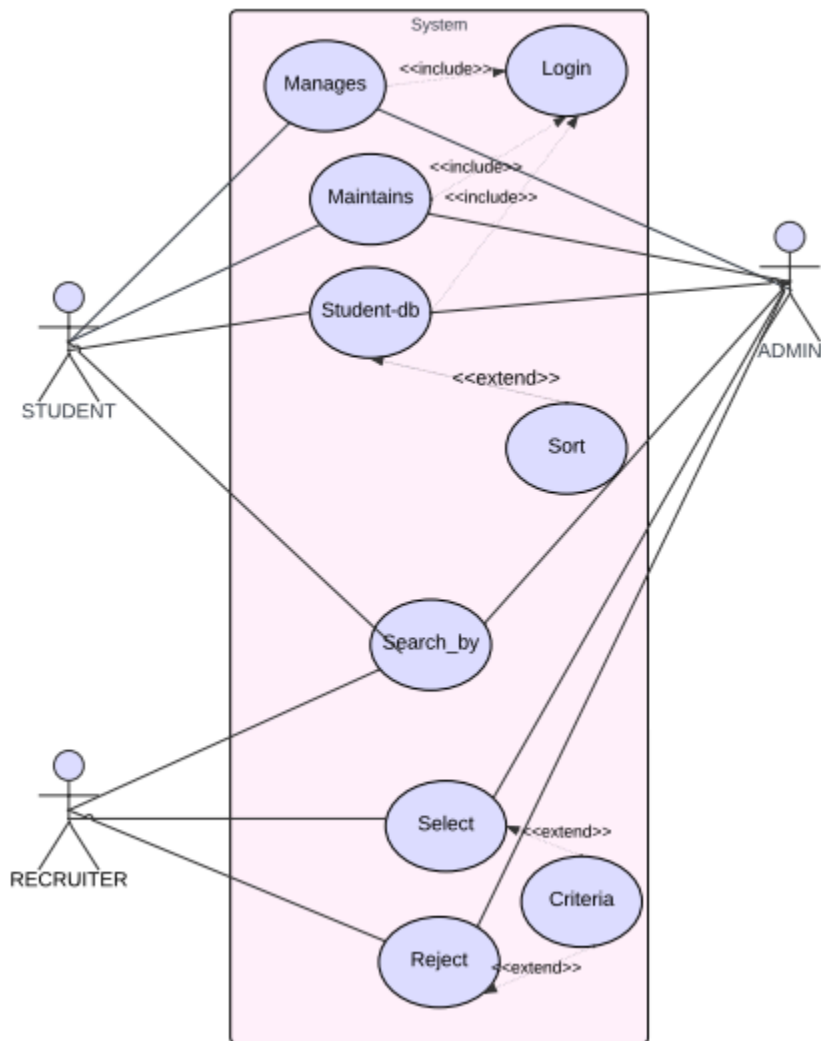
Benefits:

⇒ Clarity : Use Case diagrams provide a clear visualization of system functionality and user interactions.

⇒ Understanding : They aid in understanding the scope and requirements of the system from a user's perspective.

⇒ Validation : They help validate requirements with stakeholders and ensure that all user needs are addressed.

⇒ Prep-Tech Use-case Diagram :

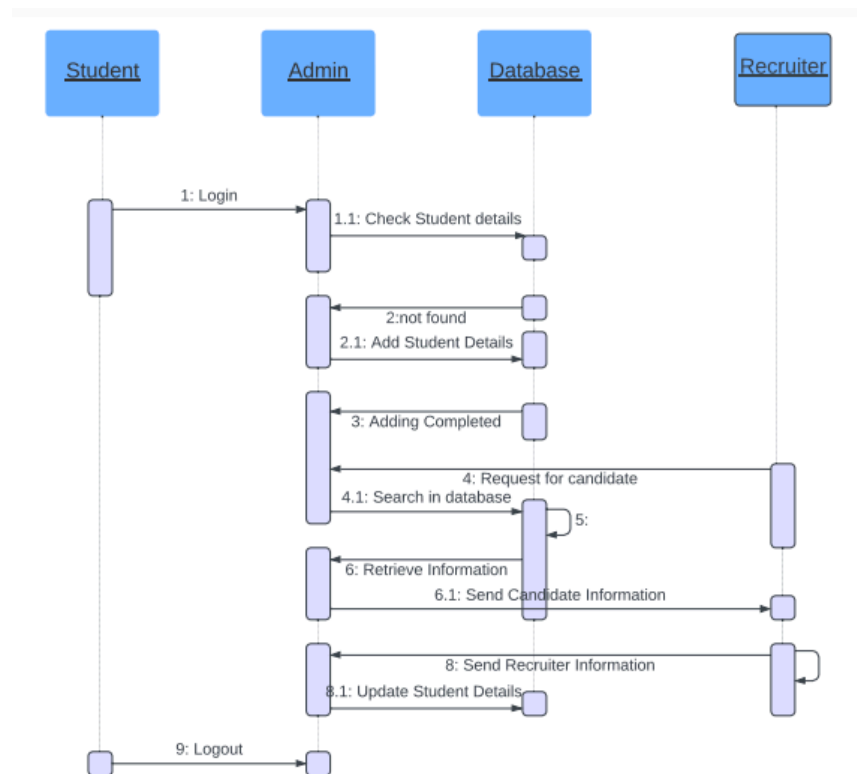


4.3.3 Sequence Diagram :

- A sequence diagram is a type of interaction diagram in UML (Unified Modeling Language) used primarily to show the interactions between objects or components in a system in a sequential order. Here's a brief overview:
- **Purpose** : Sequence diagrams visualize the interactions between various components or objects in a system over a specific period. They provide a dynamic view of how objects collaborate to achieve a certain functionality.
- **Components** : Sequence diagrams consist of several key components: -

Objects : Represent instances of classes or components participating in the interaction.

- **Lifelines** : Vertical lines that represent the lifespan of an object during the interaction.
- **Messages** : Arrows indicating communication between objects, representing method calls, or data exchanges.
- **Activation Bars**: Horizontal bars on lifelines showing the period during which an object is active or executing a particular operation.
- **Optional elements** : Such as return messages, self-messages, and loops, to capture more complex interactions.
- **Syntax** : Sequence diagrams are typically drawn from top to bottom, with time progressing downwards. Objects are listed vertically, and messages are represented by arrows between lifelines. The sequence of messages indicates the order of interactions.
- **Design**: They aid in designing the system architecture by detailing how components interact.
- **Analysis**: They help analyze the behavior of a system by visualizing the flow of messages.
- **Documentation**: They serve as documentation for understanding system behavior, especially during the development process.
- **Clarity**: Sequence diagrams offer a clear visualization of interactions between objects.
- **Communication**: They facilitate communication between stakeholders, including developers, designers, and clients.



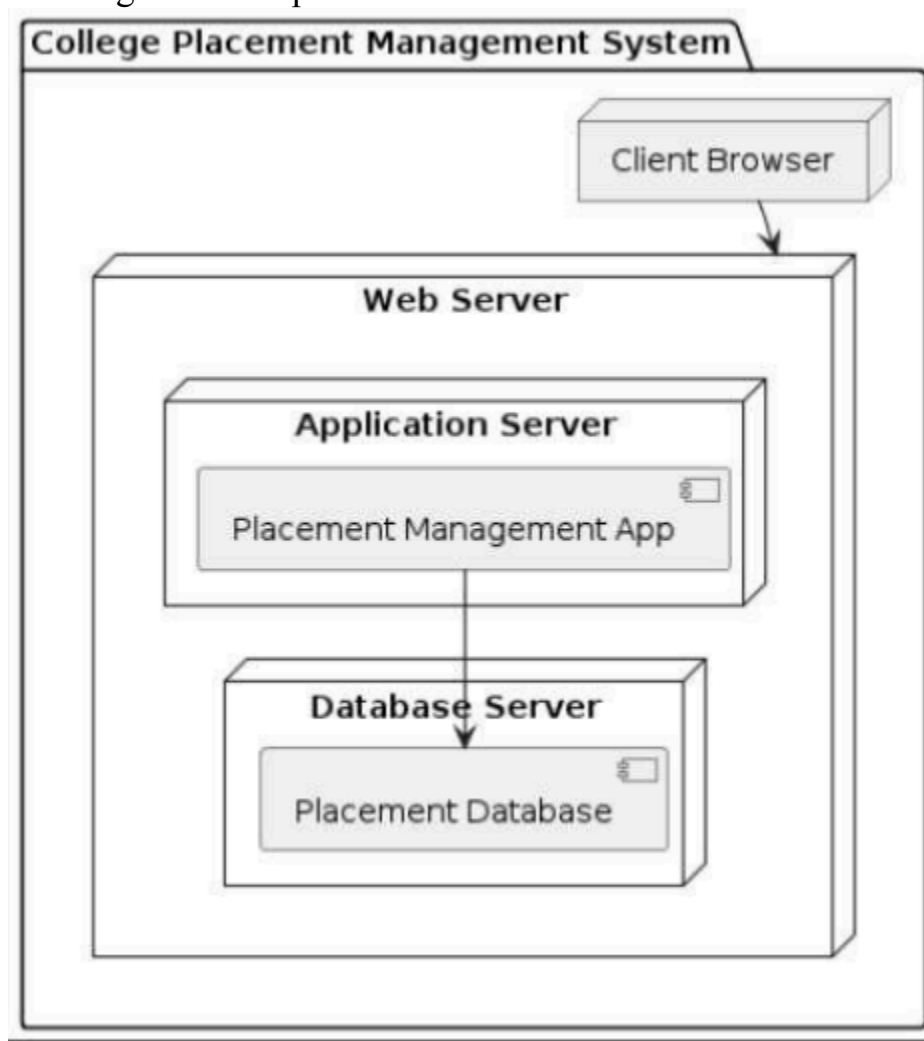
4.3.4 Component and Deployment Diagram:

- Deployment Diagram :

⇒ A deployment diagram in UML provides a detailed view of the physical deployment of software components on hardware nodes. It shows how software artifacts are distributed across different hardware environments, such as servers, routers, PCs, or other devices. In a deployment diagram, nodes represent these hardware entities, while artifacts represent the software components, executables, libraries, or other files that are deployed onto these nodes.

⇒ Deployment diagrams are vital for system architects and developers as they offer insights into the actual deployment configuration of a system. They help in understanding the infrastructure requirements, including hardware resources, network connections, and other dependencies needed for system operation. Additionally, deployment diagrams aid in planning the deployment process, ensuring efficient utilization of resources and facilitating scalability and maintenance.

⇒ Deployment Diagram of Prep Tech



- Component Diagram :

⇒ A component diagram in UML provides a high-level view of the architecture of a software system, focusing on the components that constitute the system and their relationships. Components represent modular units with well-defined interfaces that encapsulate functionality and can be independently developed, deployed, and replaced. These components may correspond to classes, packages, modules, or even larger subsystems in the system.

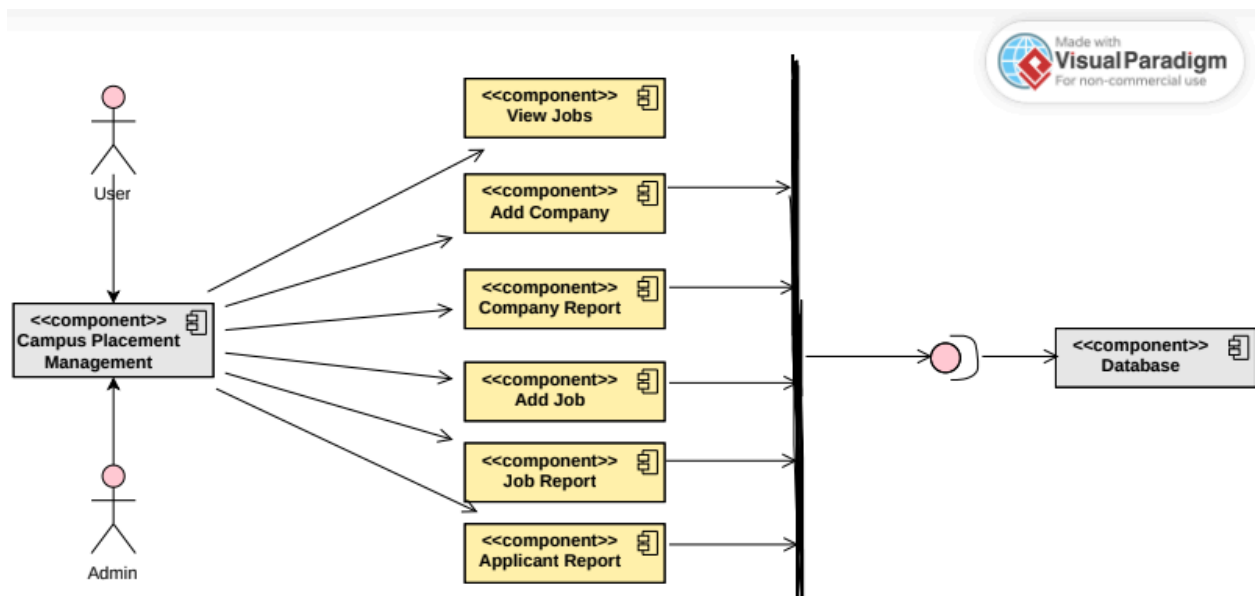
⇒ Component diagrams are essential for software designers and architects as they help in visualizing the structural organization of a system. They highlight the modularization of the system into reusable and interchangeable components, promoting maintainability, scalability, and reusability in software design. By illustrating the dependencies and relationships between components, component diagrams facilitate

24

communication among team members and support the analysis of system structure and behavior.

⇒ In summary, deployment diagrams focus on the physical deployment of software artifacts on hardware nodes, while component diagrams focus on the structural organization of the software system in terms of its modular components and their relationships. Together, these diagrams provide a comprehensive understanding of both the physical deployment and architectural structure of a system.

⇒ Component Diagram



5. Non Functional Requirements

5.1 Performance

- The system should respond quickly to user interactions, minimizing loading times and latency.
- Scalability should be considered to accommodate a growing user base.

5.2 Security

- User data should be encrypted and securely stored to prevent unauthorized access.
- Access controls should be implemented to restrict sensitive functionalities to authenticated users only.

5.3 Usability

- The user interface should be intuitive and user-friendly, catering to users of varying levels of technical proficiency.
- Accessibility guidelines should be followed to ensure the system is usable by all users, including those with disabilities.

5.4 Reliability

- The system should be highly reliable, with minimal downtime and robust error handling mechanisms in place.
- Data backups and disaster recovery plans should be implemented to prevent data loss.

6. Implementation

⇒ Implementation of the project has undergone via all the steps and methods mentioned in previous chapters along with mentioned tools and technologies. The Following Objectives shall be developed for the complete implementation of this project.

6.1 Development Environment

- The development environment for implementing the software consists of the following key components:
 - ⇒ IDE: Visual Studio Code (VS Code) is chosen as the primary Integrated

Development Environment (IDE) for its lightweight nature and extensive support for web development technologies.

⇒ Version Control: Git is used for version control to manage changes to the source code efficiently. The Git repository is hosted on platforms such as GitHub or Bitbucket to facilitate collaboration and code sharing among team members.

⇒ Project Management: Asana is utilized as the project management tool to track tasks, manage sprints, and prioritize work items. It provides a centralized platform for team communication and task coordination.

6.2 Pseudocode

6.2.1 Tour implementation

⇒ First we have utilized npx-create-react app which is simply used to create web application

⇒ After that simply creating Navbar, registration and Communication Components.

⇒ Using the sidebar for login and signup purposes.

⇒ Utilizing useRef for referencing the elements inside the Interview sub parts or in the Roadmap in respective components or paths.

⇒ useState for changing the states of the components like whether the user has turned his/her mic on/off.

⇒ sending the video audio data over the socket and peer-2-peer.

6.2.2 Rolewise permissions implementation

⇒ The main purpose of using Role Wise permission is to allow only the interviewer to call and maintain a video call while the interview is running

⇒ to do so simply utilizing localStorage of the browser to check whether the user is Interviewer or Student which simply allows us to maintain the accessibility of the Interview.

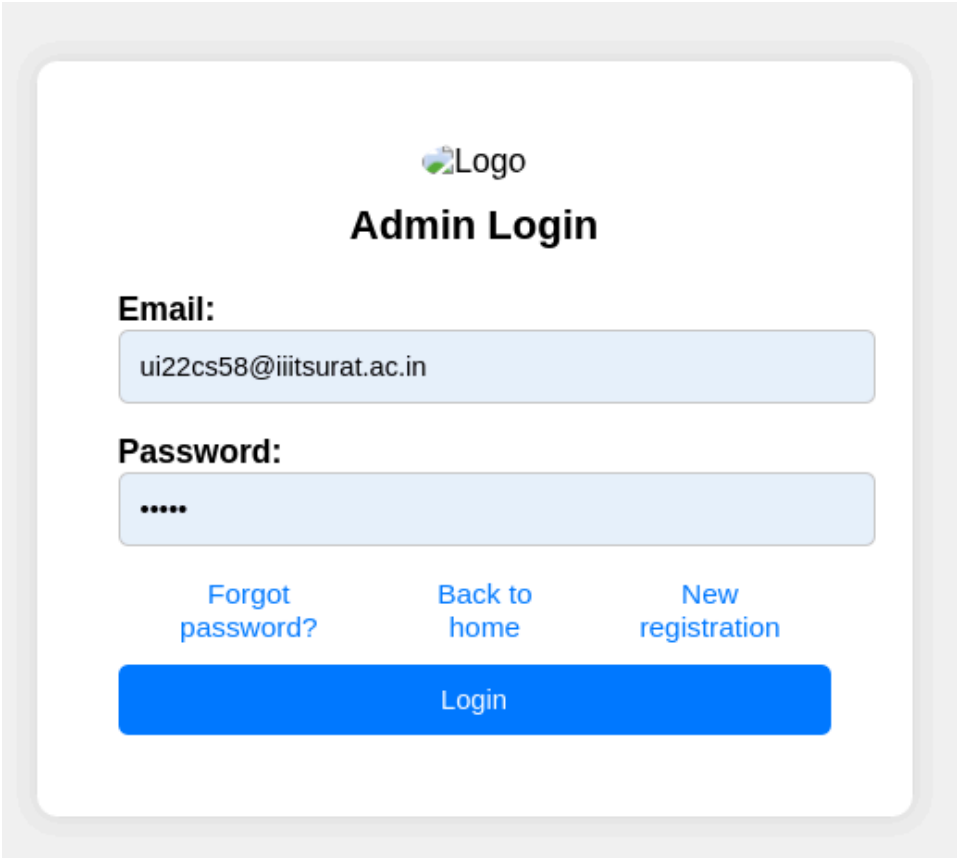
⇒ Apart from that the Interviewer will not be able to see the Roadmap as he/she is there for hiring purposes which will prevent them from going on the roadmap.

⇒ If a user tries to go outside the defined path within the application then they will get a 404 PAGE NOT FOUND error.

6.3 Integration :

- The implementation of the software involves the integration of various components and technologies, including:
 - ⇒ **Frontend Technologies:** Bootstrap CSS are utilized to build responsive and visually appealing user interfaces for the software. These frontend technologies are seamlessly integrated with the backend services using RESTful APIs.
 - ⇒ **Backend Services:** NodeJs employed to develop backend services, including API endpoints, data processing, and business logic implementation. NodeJS provides a scalable and maintainable architecture for building robust server side applications.
 - ⇒ **Database Integration:** MongoDB is used as the database management system for storing and managing data used by the software. The integration of MongoDB with the backend services is achieved using Mongoose, a MongoDB object modeling tool designed for Node.js.
- The integration of these components ensures the seamless functioning of the ERP software, providing users with a reliable and efficient solution for managing business operations.

6.4 Work Demo : Of Authentication

A screenshot of an 'Admin Login' web form. At the top center is a placeholder for a logo, labeled 'Logo' with a small icon. Below it is the title 'Admin Login' in bold. The form contains two input fields: 'Email:' with the value 'ui22cs58@iiitsurat.ac.in' and 'Password:' with masked characters '.....'. Below the password field are three links: 'Forgot password?', 'Back to home', and 'New registration'. At the bottom is a large blue 'Login' button.

Logo

Admin Login

Email:

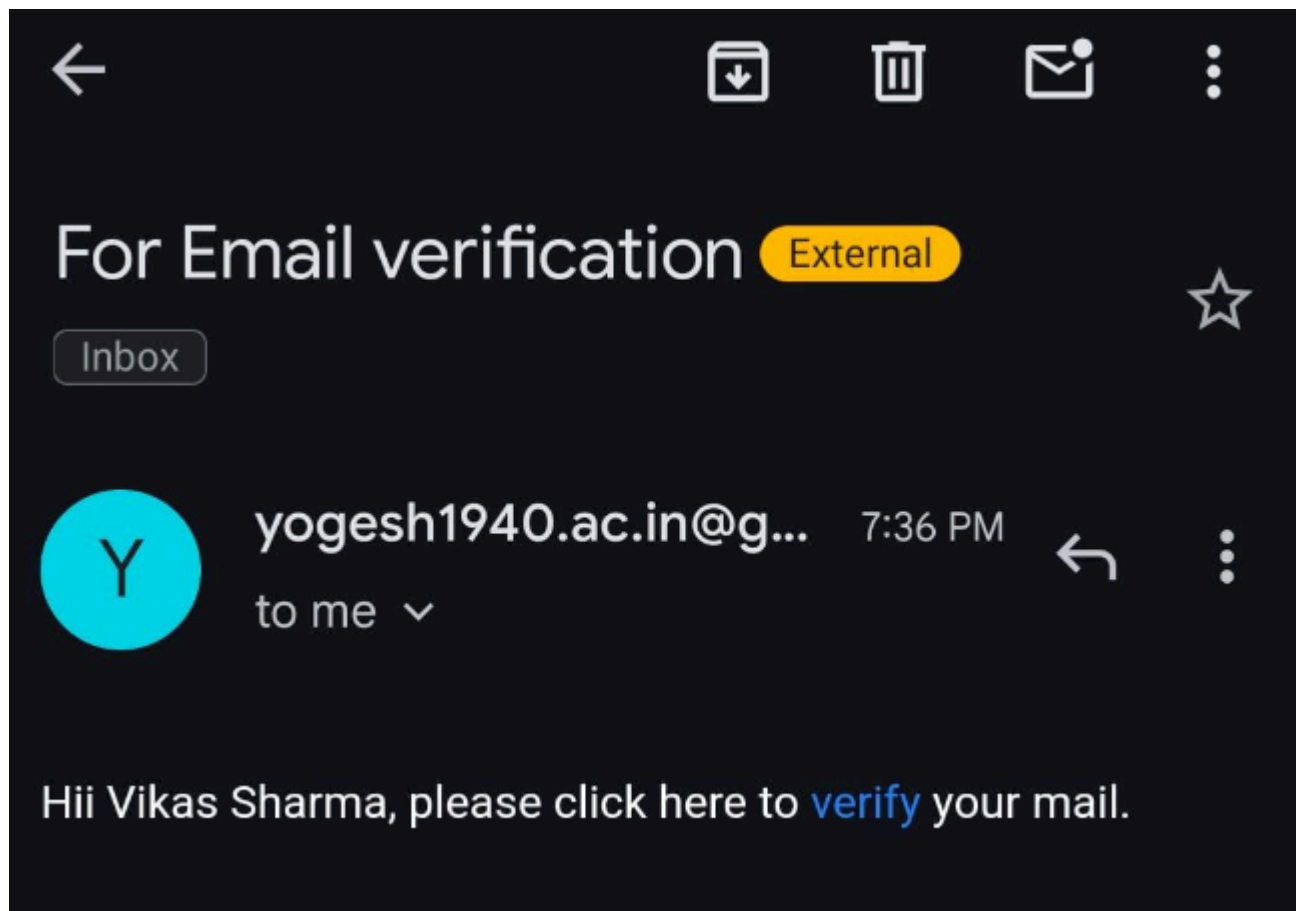
ui22cs58@iiitsurat.ac.in

Password:

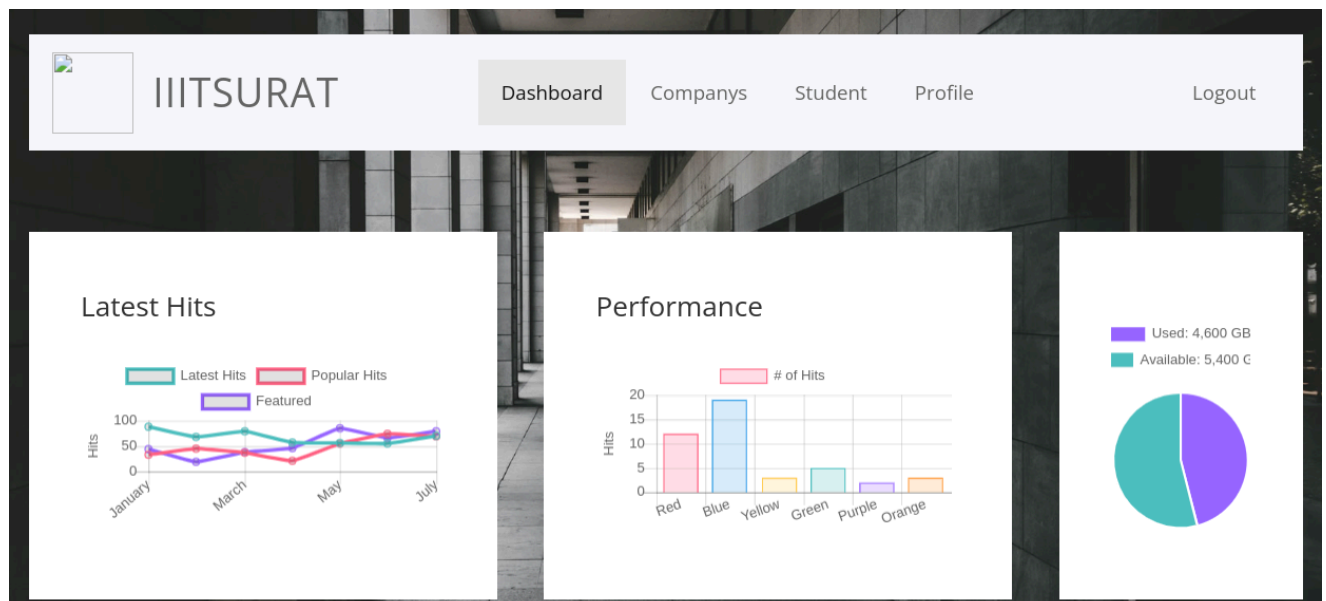
.....

[Forgot password?](#) [Back to home](#) [New registration](#)

Login



Verification Mail





Admin Dashboard




Export users

List of registered Student

Student Name	Email	Branch	Image	Status	
Datt Patel	ui22cs58@iiitsurat.ac.in	CSE		Not Placed	Edit
Vikas Sharma	ui22cs86@iiitsurat.ac.in	CSE		Not Placed	Edit

List of Student



Datt Patel
UI22CS58
CSE
[Edit Profile](#)

Information

Email	Phone
ui22cs58@iiitsurat.ac.in	9499886758
CGPA	Gender
8.3	Male

Current Status

Status	Applied
Not Placed	0
Offer	
0	

7. Constraints

- The system should comply with relevant legal and regulatory requirements, including data protection and privacy laws.
- Budget and resource constraints may limit the scope and timeline of development.

8. Assumptions

- Users have access to a stable internet connection and compatible devices to access the system.
- Users are responsible for preparing appropriate hardware and software environments for mock interviews, such as webcams and microphones.

9. Testing

9.1 Testing Methodology

The testing phase of Prep-Tech involves comprehensive testing methodologies to ensure the reliability, performance, and functionality of the software. The following

testing approaches are employed:

9.1.1 Unit Testing

Unit testing is performed to validate individual units or components of the software in isolation. It helps identify and fix bugs early in the development cycle, ensuring code correctness and robustness.

9.1.2 Integration Testing

Integration testing is conducted to verify the interactions and interfaces between different modules or components of the Software. It ensures seamless integration and interoperability across the entire system.

9.1.3 Functional Testing

Functional testing focuses on validating the functional requirements of the software, including features, user interfaces, and workflows. It ensures that the software meets the specified functional criteria and behaves as expected.

9.1.4 Performance Testing

Performance testing assesses the responsiveness, scalability, and stability of the software under various load conditions. It helps identify performance bottlenecks and optimize system resources for efficient operation.

9.1.5 Security Testing

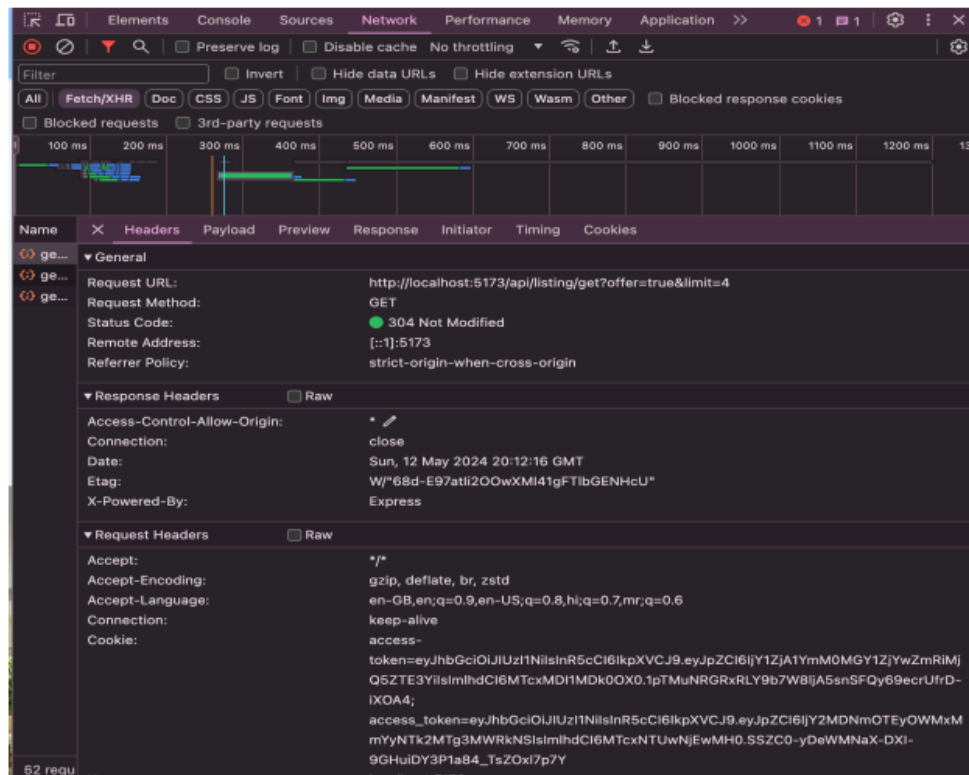
Security testing is conducted to identify and mitigate potential security vulnerabilities and threats in the software. It includes penetration testing, vulnerability scanning, and security code reviews to ensure data confidentiality, integrity, and availability.

9.1.6 User Acceptance Testing

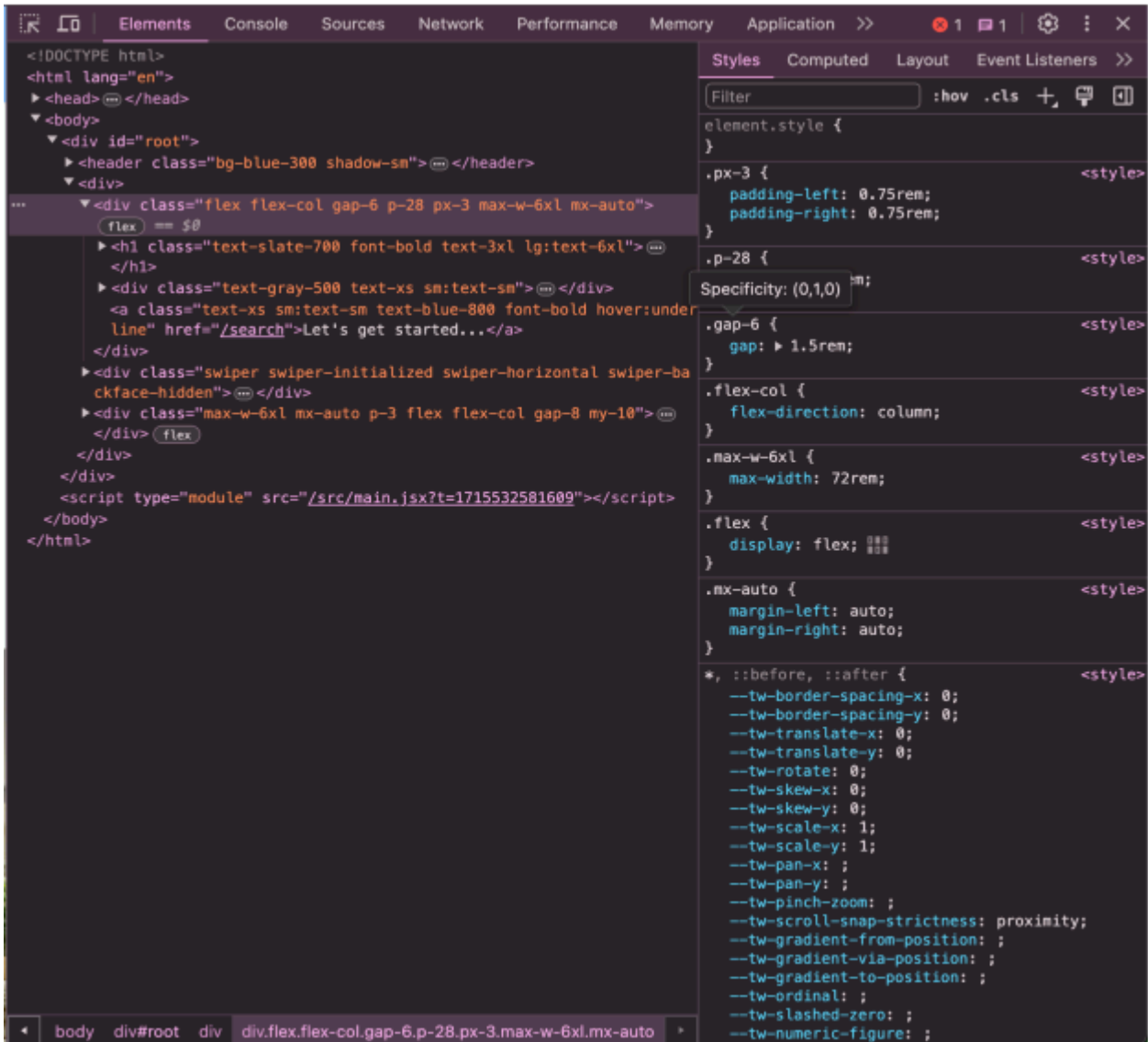
User acceptance testing involves real-world testing by end-users to validate software against their requirements and expectations. It ensures that the software meets user needs and delivers a satisfactory user experience.

9.2 Experimental Results

The experimental results from the testing phase provide valuable insights into the performance and reliability of Dealsify. The following key findings and observations are noted:



9.2.1 Unit Testing Results Testing the response in Headers



Checking the actual response

9.2.2 Integration Testing Results

Integration testing demonstrated seamless integration between different modules of Prep Tech, with minimal issues related to interface compatibility or data exchange.

9.2.3 Functional Testing Results

Functional testing confirmed the correct implementation of features and workflows in Prep Tech, with all functional requirements met according to specifications.

Application

Manifest

Service workers

Storage

Storage

- Local storage
- Session storage
- IndexedDB
- Cookies
 - http://localhost:5173
- Private state tokens
- Interest groups
- Shared storage
- Cache storage

Background services

- Back/forward cache
- Background fetch
- Background sync
- Bounce tracking mitigation
- Notifications
- Payment handler
- Periodic background sync
- Speculative loads
- Push messaging
- Reporting API

Frames

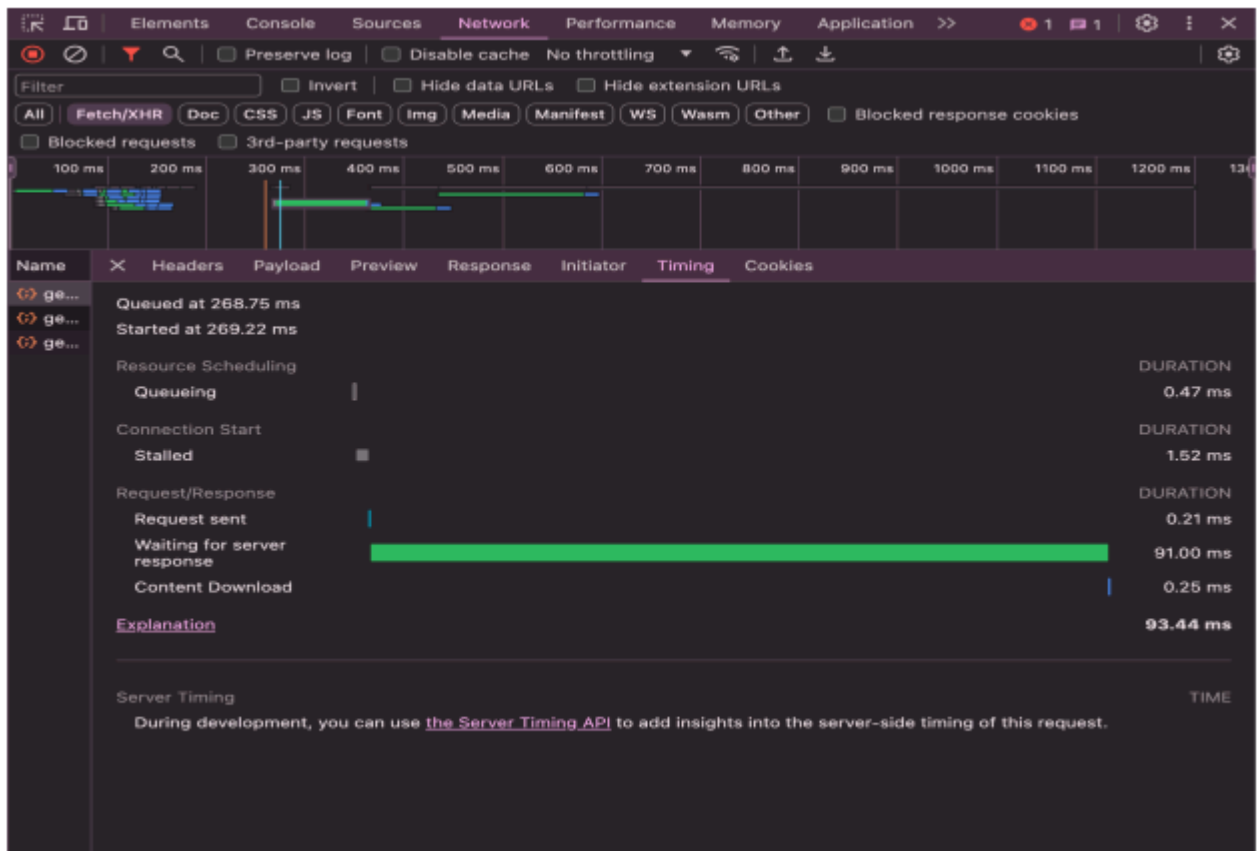
- top

Filter

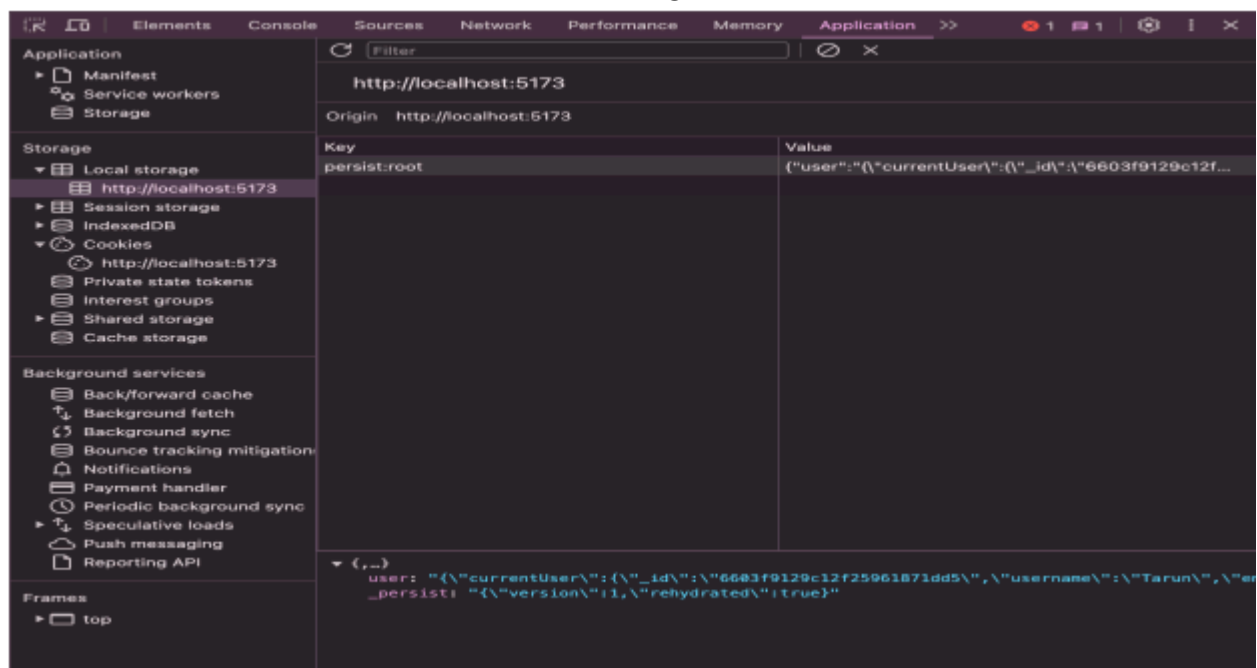
Only show cookies with an issue

Name	Value	Do...	Path	Ex...	Size	Ht...	Se...	Sa...	Pa...	Pri...
aces...	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpX...	lo...	/	Se...	160	✓				Me...
aces...	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpX...	lo...	/	Se...	160					Me...

Checking for styling



Local Storage



Checking the time taken by the Request

9.2.4 Performance Testing Results

Performance testing indicated satisfactory response times and scalability of Prep Tech under normal and peak load conditions, with no critical performance bottlenecks identified.

9.2.5 Security Testing Results

Security testing identified and addressed several security vulnerabilities in Prep Tech, including authentication flaws and data exposure risks, ensuring robust security measures are in place.

9.2.5 User Acceptance Testing Results

User acceptance testing received positive feedback from end-users, with the majority expressing satisfaction with the functionality, usability, and overall performance of Placement Management System

10. Conclusion and Future Scope

10.1 Conclusion

The College Placement Management System stands as a testament to streamlined efficiency in the realm of online placement registration. By minimizing physical paperwork and embracing a fully digital approach, it has redefined the landscape of placement processes, making them smoother and more accessible than ever before.

10.2 Future Scope

Introducing a new "Company" module as the third pillar in our College Placement Management System. Aim: to expedite and optimize workflow processes for enhanced efficiency.

Frontend development will be implemented using React framework. Rationale: React offers superior user-friendliness and interactivity.