PLACEMENT MANAGEMENT SYSTEM

A PROJECT REPORT

Submitted by

BHAVANA SHREE B S
(Reg. No: 24MCR011)

ELAMUGIL S
(Reg. No: 24MCR023)

FATHIM NISHA S
(Reg. No: 24MCR024)

in partial fulfilment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

DEPARTMENT OF COMPUTER APPLICATIONS

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE – 638 060

DECEMBER 2024

DEPARTMENT OF COMPUTER APPLICATIONS

KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE – 638 060

DECEMBER 2024

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **“PLACEMENT MANAGEMENT SYSTEM”** is the bonafide record of project work done by **BHAVANA SHREE B S (24MCR011), ELAMUGIL S (24MCR023)** and **FATHIM NISHA S (24MCR024)** in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications of Anna University, Chennai during the year 2024-2025.

SUPERVISOR **HEAD OF THE DEPARTMENT**

Date: **(Signature with seal)**

Submitted for the mini project viva voce examination held on _________________

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We affirm that the project entitled **“PLACEMENT MANAGEMENT SYSTEM”** being submitted in partial fulfilment of the requirements for the award of Master of Computer Applications is the original work carried out by us. It has not formed the part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidates.

BHAVANA SHREE B S
(Reg. No: 24MCR011)

ELAMUGIL S
(Reg. No: 24MCR023)

FATHIM NISHA S
(Reg.No:24MCR024)

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date: **Name and Signature of the Supervisor**
[Dr.M.PYINGKODI]

ABSTRACT

The Placement Management System is a web-based application created to simplify and improve placement-related activities for students. It provides a centralized platform where students can manage all placement tasks efficiently. This system shares important resources, and encourages collaboration among students. It acts as a one-stop solution for students to access everything they need to prepare for placements, making the process smoother and more effective.

One of the standout features of the system is its ability to store, manage, and retrieve a variety of placement-related resources. For example, it allows students to access aptitude questions and interview experiences contributed by their peers who have participated in placement drives for companies like Wipro, Cognizant,. These resources act as a knowledge base that helps students prepare thoroughly for their upcoming interviews and tests.

The feedback module is another important part of the system. This feature enables students to share their insights and suggestions after attending placement drives. They can contribute details about the interview rounds they went through, the technical questions they were asked, and the HR interactions they had. This feedback provides valuable real-world insights that benefit other students preparing for similar opportunities. This ensures that students can easily access the material they need for targeted preparation

Overall, the Placement Management System is a practical and innovative solution to the challenges students face during placements. By offering a centralized platform for accessing interview resources, company information, placement schedules, and feedback, it makes the entire process more efficient and less stressful.

ACKNOWLEDGEMENT

We respect and thank our correspondent, **Thiru.A.K.ILANGO B.Com., MBA., LLB.,** and our Principal **Dr.V.BALUSAMY BE(Hons)., MTech., PhD.**, Kongu Engineering College, Perundurai for providing us with the facilities offered.

We convey our gratitude and heartfelt thanks to our Professor and the Head of the Department, Professor **Dr.A.TAMILARASI Msc., MPhil., MTech., PhD.,** Department of Computer Applications, Kongu Engineering College, for her perfect guidance and support that made this work to be completed successfully.

We would like to express our gratitude and sincere thanks to our project coordinator **Dr.T.KAVITHA MCA., MPhil., PhD.,** Assistant professor(Sr.G) Department of Computer Applications, Kongu Engineering College, who have motivated us in all aspects for completing the project in the scheduled time.

We Would like to express our gratitude and sincere thanks to the project supervisior **Dr.M.PYINGKODI MCA., MPhil., PhD** Assistant Professor, Department of Computer Applications , Kongu Engineering College , for giving her valuable guidance and suggestions which helped us in successful completion of the project.

We owe a great deal of gratitude to our parents for helping overwhelm in all proceedings. We bow our hearts and head with heartfelt thanks to all those who thought us their warm services to succeed and achieve our work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	**EXPLANATON**
AI	Artificial Intelligence
API	Application Programming Interface
BAAS	Backend as a Service
DB	Database
JS	JavaScript
FW	Frame Work
SRS	Software Requirement Specification
SB	Subabase
UI	User Interface

CHAPTER 1

INTRODUCTION

ABOUT THE PROJECT

The Placement Management System is a web-based application created to simplify and improve placement-related activities for students. It provides a centralized platform where students can manage all placement tasks efficiently. This system shares important resources, and encourages collaboration among students. It acts as a one-stop solution for students to access everything they need to prepare for placements, making the process smoother and more effective.

One of the standout features of the system is its ability to store, manage, and retrieve a variety of placement-related resources. For example, it allows students to access aptitude questions and interview experiences contributed by their peers who have participated in placement drives for companies like Wipro, Cognizant,. These resources act as a knowledge base that helps students prepare thoroughly for their upcoming interviews and tests.

The feedback module is another important part of the system. This feature enables students to share their insights and suggestions after attending placement drives. They can contribute details about the interview rounds they went through, the technical questions they were asked, and the HR interactions they had. This feedback provides valuable real-world insights that benefit other students preparing for similar opportunities. This ensures that students can easily access the material they need for targeted preparation

Overall, the Placement Management System is a practical and innovative solution to the challenges students face during placements. By offering a centralized platform for accessing interview resources, company information, placement schedules, and feedback, it makes the entire process more efficient and less stressful.

1.2 EXISTING SYSTEM

The existing system refers the manual processes, disparate tools, and scattered resources, which present several challenges for students. Cocubes is a platform that provides assessments, mock tests, and interview preparation materials. Helps students prepare for placements by offering company-specific materials and feedback reports . Prepinsta a preparation platform for providing materials for aptitude, reasoning, coding, and interview preparation. Includes company-specific question sets. TCS iON is a career edge A career preparation platform offering free and paid resources for placement readiness.

1.3 DRAWBACKS OF EXISTING SYSTEM

Following are some disadvantages of the old system

1.Lack of Centralized Resources

2.Unorganized Feedback

3.Inconsistent Quality of Resources

4.Difficulty in Tracking Progress

1.4 PROPOSED SYSTEM

The proposed system is an innovative web-based platform designed to streamline the preparation process for placement interviews. It offers a comprehensive and well-organized repository of resources, including interview questions and study materials, all categorized by company and topic to help students target their preparation effectively. With features that ensure efficient data retrieval, the platform allows users to quickly access relevant materials, saving time and effort. Additionally, it provides a personalized learning experience by tailoring recommendations based on user preferences and preparation goals, enabling focused and efficient study. Overall, this system serves as a one-stop solution for students, equipping them with the tools and knowledge needed to excel in placement interviews and secure desired roles.

1.5 ADVANTAGES OF PROPOSED SYSTEM

1.Simple and quite easy to use.

2.Improved student’s satisfaction.

3.Makes the Preparation Process More Organized

4.Helps Students Perform Better in Interviews.

5.Simplifies Feedback Collection.

6.Feedback Mechanism for Continuous Improvement

SUMMARY

This chapter gives an overview of the Placement Management System, a web-based application that aims to improve the preparations for students. The system provides a central place where students can access important resources like company-specific interview questions and study materials. This helps students prepare better for interviews. It also includes a feedback feature where students can share their experiences and suggestions. This ensures the system keeps evolving based on students needs.

The system solves many problems that existing platforms have, like disorganized resources, lack of personalized content, and complicated interfaces. The Placement Management System is designed to be user-friendly, so students can quickly find what they need. It also offers personalized content tailored to each student, making it easier to focus on what’s most relevant for their preparation. The meaning students can use the platform . This makes managing placement activities much easier and more efficient. Overall, the system helps students prepare better, improves the placement process, and boosts its overall effectiveness.

CHAPTER 2

SYSTEM ANALYSIS

2.1 IDENTIFICATION OF NEED

Students may face difficulties in accessing interview questions and company-specific preparation materials, often having to rely on multiple platforms or informal sources. Placement sharing resources, and tracking feedback, may still be handled manually, leading to inefficiencies and the risk of errors. Students need a single platform where they can access the latest interview questions, aptitude tests, and company-specific resources.

2.2 FEASABILITY STUDY

A feasibility study examines whether a proposed project is practical and achievable. For the Placement Management System, it evaluates whether the system can be developed, implemented, and maintained successfully. The study looks at important factors like the required technology, operational processes, and budget to ensure the project is possible and sustainable. It helps identify potential challenges and ensures the system will meet its goals and provide value to users

- Operational Feasibility.
- Technical Feasibility.
- Legal and Ethical Feasibility**.**
- Economic Feasibility

2.2.1 OPERATIONAL FEASIBILITY

This section assesses the organization's readiness and capacity to support the implementation and ongoing operation of the Placement Management System. The evaluation considers the organization's ability to integrate the system into its existing infrastructure and processes effectively. Once implemented, the system will serve as a centralized platform for users to access a variety of resources, such as study materials, interview preparation content, and placement-related updates.

Additionally, users will have the ability to submit feedback, providing valuable insights to refine the system further, and to track placement events seamlessly, ensuring they stay informed about schedules, deadlines, and opportunities.

From an administrative perspective, the system empowers the organization with robust control over its functionalities. Administrators can manage user roles, ensuring appropriate access levels and permissions are granted based on responsibilities. They will also oversee data management, maintaining the integrity and security of the system's information. Furthermore, administrators will have the capability to configure and customize system settings to align with the organization's unique needs and objectives. This combination of user-friendly features and administrative controls ensures that the Placement Management System becomes a reliable and efficient tool for streamlining placement-related activities and enhancing the overall user experience.

2.2.2 TECHNICAL FEASIBILITY

The system will be a web-based application, accessible via a browser on various devices (desktop, laptop, and mobile). HTML, CSS, JavaScript, and frameworks like React.js will be used for the user interface.Node.js or Django with a database like MySQL or MongoDB for data storage.

2.2.3 LEGAL AND ETHICAL FEASIBILITY

User Data such as personal details and feedback will be securely stored and processed. Feedback submissions should allow users to remain anonymous unless they choose to share their identity. Sometimes it rise some copyright compliance int the website .Ensure that interview questions, study materials, and other resources shared in the system are not copyrighted or violate intellectual property rights. Feedback submissions should allow users to remain anonymous unless they choose to share their identity.

2.2.4 ECONOMIC FEASIBILITY

The economic feasibility of the Placement Management System (PMS) with added features like student feedback, interview experience sharing, and resource management includes initial costs for developing and maintaining new modules, along with server hosting for increased data. However, these features enhance student preparation, improve placement success rates, and foster collaboration. The system creates a dynamic repository of resources, saving time and improving student outcomes. Long-term benefits include higher placement rates, increased student satisfaction, and enhanced institutional reputation, making it a cost-effective solution that provides sustained value.

2.3 SOFTWARE REQUIREMENT SPECIFICATIONS

System requirements is a statement that identifies the functionality that is needed by a system in order to satisfy the customer's requirements. System requirements are the most efficient way to address user needs while lowering implementation costs.

- Hardware Requirements
- Software Requirements

2.3.1 HARDWARE REQUIREMENTS

For your Placement Management System, which is a web-based application, the hardware requirements will depend on the specific roles (such as students, administrators, or placement coordinators) and the infrastructure used to host the system.

<u>PROCESSOR</u>

- Intel Core i5 or higher (or equivalent AMD processor)
- Minimum: 2.0 GHz or faster

<u>RAM</u>

- 8 GB of RAM

STORAGE

256 GB SSD (for fast file access, especially for large codebases and databases)

DEVELOPMENT TOOLS:

Code editors (e.g., Visual Studio Code, Sublime Text, or IntelliJ IDEA).

Browsers (e.g., Google Chrome, Mozilla Firefox) for testing.

Virtualization tools (e.g., Docker, VMWare) if using virtual environments.

Database software (e.g., MySQL, MongoDB)

2.3.2 SOFTWARE REQUIREMENTS

This section gives the details of the software that are used for the development.

- Frontend: React.js with HTML and CSS.
- Backend: Node.js.
- Database: MongoDB.
- Application Type: web application.
- Tool: Visual Studio Code

FRONT END

React.js is a popular JavaScript library used for building user interfaces, particularly for single-page applications where dynamic and responsive content is required. Developed by Facebook, React allows developers to create reusable UI components that can efficiently update and render when data changes. Its core feature, the virtual DOM, ensures optimal performance by minimizing direct manipulation of the browser's DOM. React provides a smooth, interactive experience by allowing components to update in response to user input or system events. In your Placement Management System, React.js will enable the creation of a fast, responsive, and user-friendly interface, enhancing the overall user experience for students and coordinators.

Features of REACT

1. **Component-Based Architecture** React allows developers to build applications using small, reusable components. Each component has its own logic and UI, which can be composed together to create complex user interfaces. This modularity makes development and maintenance easier.
2. **Virtual DOM** React uses a Virtual DOM to improve performance. When the state of an object changes, React first changes the object in the Virtual DOM, then compares it with the previous version, and finally updates only the parts of the real DOM that have changed. This leads to faster rendering and improved performance.

BACK-END

MongoDB is a popular open-source, NoSQL database that stores data in a flexible, JSON-like format called BSON (Binary JSON). Unlike traditional relational databases, MongoDB uses collections and documents instead of tables and rows, offering a more dynamic and flexible data model. It supports horizontal scaling through sharding, making it highly scalable to handle large datasets and increased traffic. MongoDB's flexible schema allows developers to store documents with varying fields in the same collection, which is ideal for applications where data structures may evolve. The database also features indexing to speed up queries, replication for high availability, and an aggregation framework for complex queries and data transformations. Known for its high performance, MongoDB is capable of handling large volumes of read and write operations, making it suitable for real-time applications. In the Placement Management System, MongoDB can store various data types like student details, feedback, and interview questions, offering a scalable and efficient solution for managing dynamic data.

-Features of MONGO DB

1. **Document-Oriented Storage** MongoDB stores data in flexible, JSON-like documents (BSON), which allows each document to have different fields and structures. This provides flexibility in storing data without a predefined schema.
2. **Data Consistency** MongoDB provides options for maintaining data consistency using Write Concerns (the level of acknowledgment requested from MongoDB for write operations) and Read Concerns (ensuring data consistency when reading from replicas).
3. **Real-Time processing** MongoDB is optimized for real-time data processing and is capable of handling large volumes of data in fast-paced applications. It supports high-speed reads and writes, making it suitable for real-time analytics and interactions.

NODE

In Placement Management System, Node.js is used to connect to MongoDB and handle the backend operations. Node.js is a powerful JavaScript runtime built on Chrome's V8 engine (), allowing you to run JavaScript code on the server side. It is known for its speed, scalability, and event-driven, non-blocking architecture, making it well-suited for handling concurrent requests in real-time applications.

Node.js serves as the backend technology to handle requests from the frontend (built with React.js). It connects to MongoDB using a MongoDB driver or Mongoose (a popular ODM - Object Data modeling library for MongoDB). This allows Node.js to interact with your database, perform CRUD (Create, Read, Update, Delete) operations, and manage the dynamic data like student details, interview questions, feedback, and more.

With Node.js, it create RESTful APIs to manage the flow of data between the frontend and backend. When a student or placement coordinator interacts with the Placement Management System (for example, by submitting feedback or searching for interview questions), Node.js processes the request, interacts with MongoDB to fetch or store data, and then sends the response back to the frontend. This results in a seamless and responsive user experience.

Using Node.js with MongoDB provides a full-stack JavaScript solution, allowing for easy integration between the frontend and backend, fast data retrieval, and efficient handling of the placement-related resources and feedback in real time.

Features of NODE

1. **Asynchronous and Event-Driven**

   Node.js operates on a non-blocking, event-driven model, allowing it to handle multiple operations concurrently. This means that while Node.js is waiting for an operation (like a file read or database query) to complete, it can continue to process other tasks, improving efficiency and performance.

2.NPM (Node Package Manager)

Node.js comes with NPM, a powerful package manager that allows developers to easily install and manage libraries and modules. NPM has a vast ecosystem of packages, making it simple to integrate third-party tools into Node.js applications.

2. **Real-Time Capabilities**

   Node.js is particularly well-suited for building real-time applications (e.g., chat applications, live updates) due to its non-blocking I/O and event-driven architecture. It can handle frequent I/O operations with minimal latency.

SUMMARY

The Placement Management System (PMS) is a comprehensive web-based platform developed to optimize and simplify the placement activities for both students and coordinators. The system serves as a centralized hub where students can access a wealth of resources, including interview questions, aptitude tests and feedback from past candidates. By providing these resources in one place, the PMS enhances student preparation for placement drives and fosters collaboration between students and placement coordinators.

The platform is built using React.js for the frontend, offering a dynamic and responsive user interface that ensures seamless interaction. The Node.js backend facilitates efficient handling of data, real-time updates, and smooth interaction between the frontend and backend. With MongoDB as the database, the system supports flexible, scalable data storage that accommodates various types of resources, such as interview experiences, questions categorized by company, difficulty level, and topics. Providing students with the latest resources, schedules, and feedback.

CHAPTER 3

SYSTEM DESIGN

3.1 MODULE DESCRIPTION:

The Placement Management System includes key modules to streamline the placement process. The Dashboard serves as the central hub, featuring sections like Companies, where users can access company details and interview questions, and Peer Experience, which allows students to share feedback with details like name, email, mobile number, and suggestions. These modules work together to provide a comprehensive and user-friendly platform for placement preparation and Login , feedback register.

- Companies
- Peer Experience
- logout

Register

The Registration Page in the Placement Management System allows users to create a personalized account by providing essential details such as their full name, email, password, and confirm password. Each user must register with unique information to ensure secure access and personalized experience. This module ensures that all users have individual accounts, enabling them to access company details, peer feedback, and other placement resources tailored to their needs.

Login

The Login Page in the Placement Management System allows users to securely access their profiles using their registered email and password. Each user's unique credentials ensure secure authentication and personalized access to their account. Upon successful login, users can explore placement resources, view company details, share feedback, and manage their information, ensuring a seamless and user-specific experience.

Companies

The Company Details Module in the Placement Management System provides a comprehensive repository of information about companies visited by students during placement drives. This module stores data such as interview questions, hiring patterns, and insights shared by students. Visitors to the website can access these details to better understand company expectations and prepare effectively for upcoming interviews. By offering well-organized and reusable information, this feature empowers students to enhance their preparation and boost their confidence in placement processes.

Peer Experience

The Peer Experience Module in the Placement Management System allows students to share their detailed placement journey, providing valuable insights for their peers. Students can input their information, including register number, phone number, company name, interview date, and personal interview experience. They can also share interview questions they faced, along with their name and email ID. This module serves as a collaborative platform, enabling students to contribute meaningful content and help others prepare effectively for similar placement opportunities.

Historical Trends

The Historical Trends Module in the Placement Management System provides a comprehensive overview of past placement records. This module tracks details such as the companies that visited the campus, the number of students selected during their interviews, and placement trends over the years. By analyzing these trends, students and coordinators can gain valuable insights into company hiring patterns and campus placement performance, aiding in better preparation and strategic planning for future placement activities.

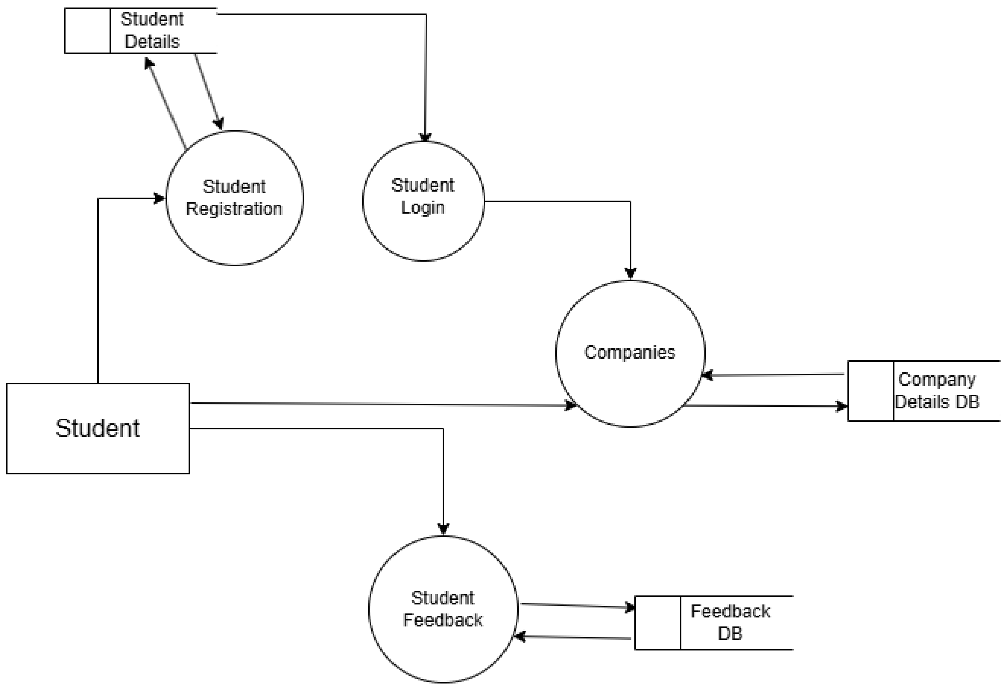
Logout

The Logout Module in the Placement Management System allows users to securely exit the platform once they have completed their session. By clicking on the logout option, users are safely logged out of their account, ensuring that their personal information and session data remain secure. This feature helps maintain privacy and security, particularly for users accessing the system from shared or public devices.

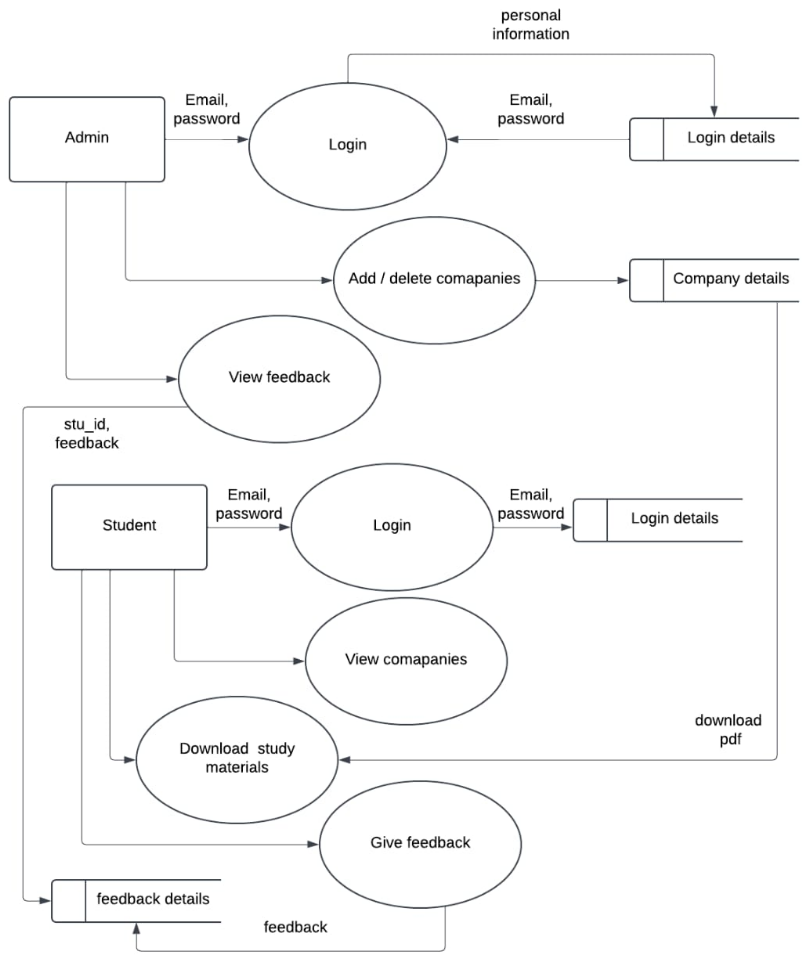
3.2 DATA FLOW DIAGRAM

The Data Flow Diagram provides information about the inputs and outputs of each entity and process itself.

LEVEL 0



LEVEL 1



3.3 DATABASE DESIGN

The database design for the Placement Management System is created to securely store and manage important data while connecting the database to the webserver for dynamic functionality. It organizes information like student details, company data, peer experiences, and historical trends. Using a backend tool like Node.js and a database like MongoDB, the system handles tasks such as user registration, login, and data retrieval in real time. Security features like encryption and access controls keep user data safe. This setup ensures students can easily register, access company details, share feedback, and view historical trends, making placement activities simple and efficient.

Student table

FIELD NAME	DATA TYPE	SIZE	CONSTRAINTS
Stu_roll	varchar	-	Primary key autoincrement
Stu_name	varchar	30	Not null
Stu_email	varchar	30	Not null
Stu_password	integer	10	Not null
Stu_phone	integer	10	Not null

Company

FIELD NAME	DATA TYPE	SIZE	CONSTRAINTS
Company_name	varchar	90	Not null
Interview_date	date	10	Not null
Interview_questions	varchar	500	Not null
feedback	varchar	100	Not null

FIELD NAME	DATA TYPE	SIZE	CONSTRAINTS
Admin_name	varchar	20	Not null
Admin_email	varchar	30	Unique
Admin_password	integer	10	Not null
Admin_add_compa	varchar	30	Not null

Admin

Database Integration

In our Placement Management System, database integration is essential for connecting various modules to ensure smooth functionality and efficient data management. Key features such as login, registration, company details, peer experience, historical trends, and logout are seamlessly integrated into the database, creating a unified platform. The database securely stores user credentials and registration details, validating login attempts to ensure authorized access. It also serves as a repository for company-related information, such as company lists, interview questions, and placement statistics, which students can access for preparation. Additionally, the peer experience module allows students to share their interview experiences, including personal details, company names, and questions, creating a valuable knowledge base for others. Historical trends, such as placement records and company visit statistics, are maintained to provide insights into past placement drives. The logout functionality ensures secure session management by preventing unauthorized access after users exit the system. This centralized database integration allows dynamic interaction between the web server and users, ensuring data security, smooth access, and an interactive user experience.

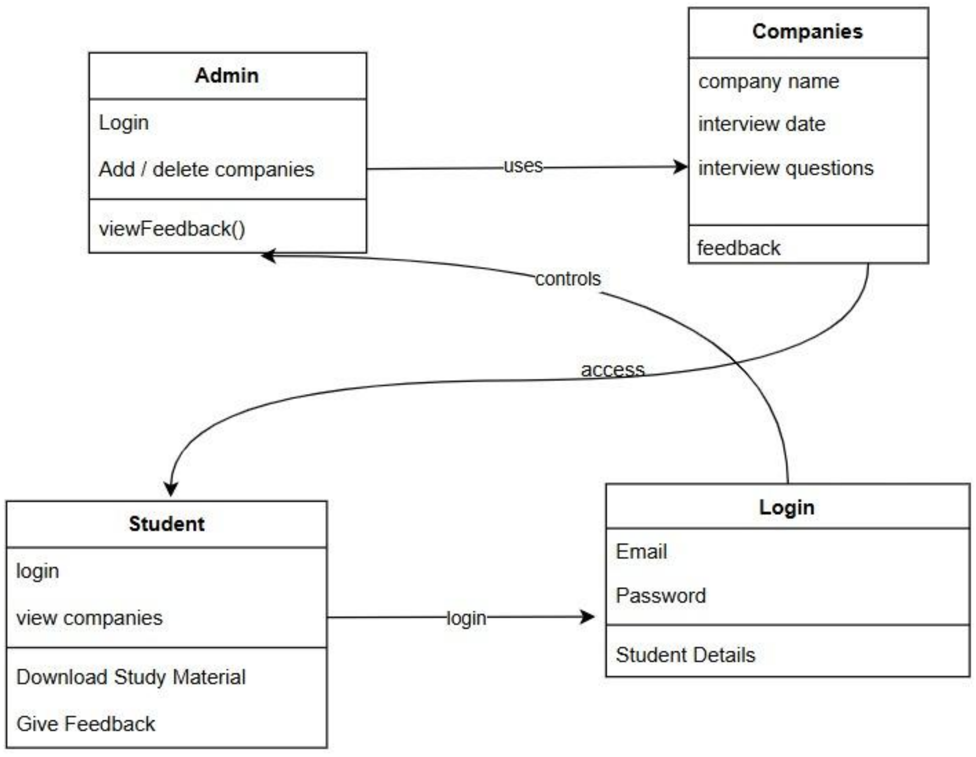
Data Independence

Data independence is an important feature of the Placement Preparation System that ensures flexibility and long-term usability. It means that changes made to how data is stored or structured do not disrupt the system's main functions. For example, physical data independence allows updates to the storage methods or infrastructure without affecting how students or administrators interact with the system. Similarly, logical data independence ensures that changes to the database design, like adding new fields for feedback or updating the schema, won’t interfere with the processes of registering, logging in, or submitting and managing feedback. This flexibility allows the system to adapt to new requirements, technologies, or better data management practices while remaining reliable and efficient for its users.

Data Security

Data security is a key priority in the Placement Management System to protect sensitive student information and maintain privacy. Personal details, feedback records, and other data stored in the system are safeguarded using strong encryption techniques. Access to the system is restricted with authentication measures to prevent unauthorized use. Regular checks and monitoring are carried out to identify and fix any potential security issues. Secure communication protocols ensure that data shared between users and the system remains safe from breaches. By focusing on data security, the system builds trust among users, ensuring that their information is handled confidentially and creating a secure platform for managing placement-related activities.

Class Based Diagram



3.6 INPUT DESIGN

The input design for the Placement Management System focuses on creating user-friendly and structured forms for various functionalities. On the Registration and Login Page, users will provide their full name, email, password, and confirm password with validation for email format and password strength. The Company Details Page will allow students to input interview questions, interview process details, and feedback for specific companies. In the Peer Experience Page, students can share their interview experiences, including their register number, phone number, company name, interview date, and suggestions.The Search and Filter functionality will let users search and filter placement data by company name, topic, question type. The Logout button will allow users to securely exit the platform. Each form will have real-time validation, success/error messages, and clear labels to ensure an easy and efficient user experience.

Login Form

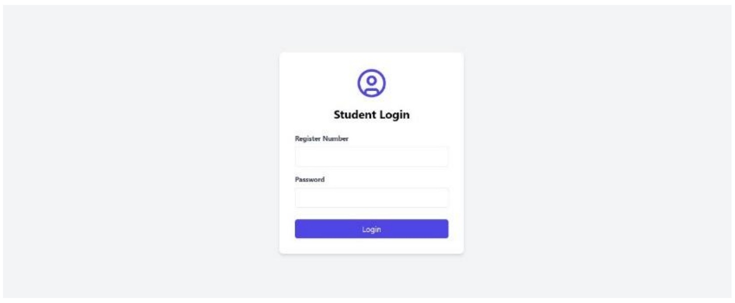


Figure 3.5 In this Placement Management System project, the login page is a key feature designed to ensure secure access for users. It requires each user to enter a unique username, which serves as their identifier in the system, and a password, which is confidential and must be correctly entered to gain access. This login process not only helps in authenticating users but also ensures that the system's resources and features are accessed only by authorized individuals. The implementation of unique usernames prevents duplication and confusion, while the password requirement adds

a layer of security to protect user data and maintain the system's integrity. This design enhances user accountability and safeguards the system against unauthorized access.

3.7 OUTPUT DESIGN

The output design for the Placement Management System focuses on delivering a user-friendly and visually intuitive interface. Students will have access to a comprehensive dashboard displaying their progress across technical, aptitude, and logical rounds using visual elements such as progress bars or pie charts. Performance analytics will provide detailed insights into scores, time management, and the accuracy of responses. Additionally, a feedback section will highlight strengths and areas for improvement, offering actionable suggestions. Students can review questions with detailed explanations and bookmark challenging ones for future reference. Personalized insights, such as identifying strong and weak areas, will guide students effectively in their preparation journey.

CHAPTER 4

IMPLEMENTATION

4.1 CODE DESCRIPTION

The development of the placement preparation and management system as they explain the purpose and logic behind the code. Clear and meaningful comments help developers understand the intent of the code, making it easier to maintain, update, or debug the system. These comments also facilitate generating external documentation and integrating with other tools or systems.

Front-end Built using HTML, CSS, and React to provide a responsive and user-friendly interface for students, administrators, and employers.

Back-end Powered by MongoDB, ensuring efficient management of data such as student profiles, preparation resources, placement records, and employer details.

4.2 STANDARDIZATION OF THE CODING

Coding standards establish a consistent style for writing code. These standards are not about deciding what is universally right or wrong but rather providing a structured approach to organizing and formatting source code. They also help in maintaining uniformity within or across development teams. Well-written code is easy to debug, modify, and maintain.

In this project, all coding standards have been carefully implemented. Starting with clean and organized code from the beginning simplifies the development process and avoids the hassle of fixing messy code later. Each programmer tends to have a distinct way of naming variables, structuring functions, and adding comments. By standardizing these practices, the team ensures a smoother workflow, making collaboration easier and the final product more reliable and efficient over time.

4.3 ERROR HANDLING

Exception handling is an essential method for managing unexpected issues in the code and ensuring the placement management system continues to run smoothly. It helps maintain control over the program's flow when abnormal situations arise. Various handlers are employed to process these exceptions and execute the code accordingly. A common approach involves using if-else blocks to manage errors by checking predefined conditions. However, if-else statements are limited to the scenarios explicitly defined by the developer, and they cannot address unexpected edge cases dynamically.

Errors, on the other hand, represent serious issues that can cause the system to crash. For example, an error in the placement management system might disrupt student registrations or job postings. Errors often require monitoring and debugging tools to detect them in production. Developers then address these issues to restore the application to normal operation.

In contrast, exceptions represent unusual but manageable conditions that the system can handle without crashing. For example, if a student attempts to upload an unsupported file format for their resume, the system could use try-catch statements to handle the exception, alerting the user and providing guidance without interrupting the workflow.

However, when the system encounters an exception with no predefined solution, it becomes an unhandled exception—essentially equivalent to an error.

By implementing robust exception handling practices, the placement management system ensures a smoother user experience, minimizes disruptions, and maintains reliability

CHAPTER 5

TESTING AND RESULTS

5.1 TESTING

Once the source code has been finalized and documented, including associated statistical structures, the next phase involves testing and validation. During this stage, there is a dedicated focus on thoroughly testing the program, actively seeking errors and discrepancies.

The developer approaches testing with caution, conducting detailed design and execution tests to showcase the program's functionality and identify any potential issues. While the aim is to demonstrate the program's effectiveness, it is acknowledged that errors may still be present. If the developer fails to uncover these errors, it is likely that end-users will discover them.

Responsibility for testing extends to individual program components or modules, with the developer consistently conducting tests to ensure their proper functioning. Additionally, in many cases, the developer undertakes integration testing, a crucial step in validating the overall application structure.

5.1.1 Unit Testing

In the context of unit testing, the examination focuses on individual components that constitute the system. Unit testing is sometimes referred to as program testing because it involves evaluating the software within a unit, which comprises modules and routines assembled to perform specific functions. Unit testing is primarily directed at assessing the modules independently of each other, aiming to identify errors.

This approach is instrumental in detecting coding and logic errors confined within the module itself. The testing process occurs during the programming stage, allowing for early identification and resolution of issues.

Example Test Case 1

Module: User Login

Test Type: Loading of appropriate form for administrator

Input: Username and Password

Output: Display user pages

Sample Test Output: Redirect to the main page and display the dashboard pages.

Analysis In this scenario, the test ensures that the username and password are in the correct format. If there is a mismatch between the entered credentials and the data stored in the database, it may lead to a data mismatch error. This highlights the importance of thorough testing to catch such issues early in the development process.

5.1.2 Integration Testing

Integration testing is conducted to verify whether individual modules seamlessly operate as a cohesive unit. During integration testing, the specific modules earmarked for integration are subjected to examination. In this phase, manual test data previously employed to assess interfaces is replaced with data automatically generated from diverse modules.

This process serves the purpose of assessing how a module would interact within the proposed system. The integration and subsequent testing of modules aim to expose any issues related to interfaces within the system. The focus is on identifying and addressing potential problems that may arise when modules interact with each other.

5.1.3 Validation Testing

Verification and validation testing are essential assessments conducted prior to delivering the product to the customer. This ensures the early commencement of the software testing life cycle. The primary objective of both verification and validation is to confirm that the product aligns with the customer's requirements and effectively fulfils its intended purpose.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The Placement Management System is a centralized platform designed to simplify and enhance placement preparation for students. It provides essential features such as a feedback module where students can share their experiences and a database of interview questions. By organizing and storing these valuable resources, the system makes it easier for students to access the information they need in one place. This project aims to streamline the placement process, helping students prepare effectively and confidently as they work toward achieving their career goals.

FUTURE ENHANCEMENT

The Placement Management System has the potential for significant enhancements in the future, making it an indispensable tool for students preparing for placements. As it evolves, the system could integrate advanced features like artificial intelligence to provide personalized recommendations based on individual student profiles, such as tailored study materials, practice tests, and interview tips.

Additionally, real-time analytics and dashboards could be introduced to help students track their preparation progress and identify areas of improvement. The system could also expand to include live webinars, mentorship programs, and collaboration tools to connect students with industry professionals and alumni for guidance.

For students, this project ensures a centralized, secure repository of placement-related resources, saving them time and effort in finding reliable materials. By providing structured and easily accessible information, the system empowers students to prepare more effectively, enhancing their confidence and readiness for interviews.

In the long term, it not only supports individual career growth but also strengthens the institution's placement outcomes, benefiting future batches by creating a comprehensive knowledge base that grows with every placement cycle.

APPENDICES

A. SAMPLE CODING

```
import { useState } from 'react';


const AdminDashboard = () => {
  const [newCompany, setNewCompany] = useState({
    name: '',
    description: '',
  });


  const handleSubmit = async (e) => {
    e.preventDefault();
    // TODO: Implement company creation logic
  };


  return (
    <div className="container mx-auto px-4 py-8">
    <h2 className="text-3xl font-bold mb-8">Admin Dashboard</h2>

    <div className="bg-white rounded-lg shadow-md p-6 mb-8">
     <h3 className="text-xl font-semibold mb-4">Add New Company</h3>
     <form onSubmit={handleSubmit} className="space-y-4">
      <div>
       <label className="block text-sm font-medium text-gray-700">Company
Name</label>
       <input
        type="text"
        className="mt-1 block w-full px-3 py-2 border rounded-md"
        value={newCompany.name}
        onChange={(e) => setNewCompany({ ...newCompany, name: e.target.value
})}
       />
      </div>
      <div>
       <label className="block text-sm font-medium text-gray-
700">Description</label>
       <textarea
        className="mt-1 block w-full px-3 py-2 border rounded-md"
        rows="4"
        value={newCompany.description}
        onChange={(e) => setNewCompany({ ...newCompany, description:
e.target.value })}
```

```
          />
        </div>
        <button
          type="submit"
          className="bg-blue-600 text-white px-4 py-2 rounded-md hover:bg-blue-700"
        >
          Add Company
        </button>
      </form>
    </div>

    {/* Add more admin features here */}
  </div>
 );
};

export default AdminDashboard;
import { useState } from 'react';
import { useAuth } from '../../context/AuthContext';
import { login as loginApi } from '../../utils/api';

const Login = () => {
  const [credentials, setCredentials] = useState({ email: '', password: '' });
const [error, setError] = useState('');
const [loading, setLoading] = useState(false);
const { login } = useAuth();

const handleSubmit = async (e) => {
  e.preventDefault();
  setError('');
  setLoading(true);

  try {
    const { user } = await loginApi(credentials);
    login(user);
  } catch (err) {
    setError(err.response?.data?.message || 'Failed to login');
  } finally {
    setLoading(false);
  }
};

return (
  <div className="min-h-screen flex items-center justify-center bg-gray-50">
    <div className="max-w-md w-full space-y-8 p-8 bg-white rounded-lg shadow-md">
      <h2 className="text-3xl font-bold text-center text-gray-900">Login</h2>
```

```
{error && (
  <div className="bg-red-100 border border-red-400 text-red-700 px-4 py-3 rounded">
    {error}
  </div>
)}
<form className="mt-8 space-y-6" onSubmit={handleSubmit}>
  <input
    type="email"
    placeholder="Email"
    className="w-full px-3 py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500"
    value={credentials.email}
    onChange={(e) => setCredentials({ ...credentials, email: e.target.value })}
    required
  />
  <input
    type="password"
    placeholder="Password"
    className="w-full px-3 py-2 border rounded-md focus:outline-none focus:ring-2 focus:ring-blue-500"
    value={credentials.password}
    onChange={(e) => setCredentials({ ...credentials, password: e.target.value })}
    required
  />
```

```
          <button
            type="submit"
              className="w-full py-2 px-4 bg-blue-600 text-white rounded-md hover:bg-blue-700 focus:outline-none focus:ring-2 focus:ring-blue-500 focus:ring-offset-2 disabled:opacity-50"
            disabled={loading}
          >
            {loading ? 'Logging in...' : 'Login'}
          </button>
        </form>
      </div>
    </div>
  );
};


export default Login;


import React, { useState } from 'react'; import { Send } from 'lucide-react'; import { Feedback } from '../types';


interface FeedbackFormProps
{
onSubmit: (feedback: Feedback) => void;
}
```

```
const FeedbackForm = ({ onSubmit }: FeedbackFormProps) => { const [feedback,

setFeedback] = useState<Feedback>({ registerNo: '',name: '',phone: '',email: '',

companyName:  '',  interviewDate:  '',  experience:  '',  questions:  '',  suggestions:  ''});

const handleSubmit = (e: React.FormEvent) => {

e.preventDefault();onSubmit(feedback);};

return ()

<div className="bg-white rounded-lg shadow-md p-6">

<h2 className="text-2xl font-bold mb-6 text-gray-800">Interview Feedback</h2>

<form onSubmit={handleSubmit} className="space-y-4">

<div className="grid grid-cols-1 md:grid-cols-2 gap-4">

<div>

<label    className="block    text-sm    font-medium    text-gray-700">Register
Number</label>

<input type="text"

value={feedback.registerNo}

onChange={(e)  =>  setFeedback({...feedback,  registerNo:e.target.value})}

className="mt-1 block w-full rounded-md border-gray-300 shadow- sm

focus:border-indigo-500 focus:ring-indigo-500"required/>

</div>


<div>

<label className="block text-sm font-medium text-gray- 700">Name</label>

<input type="text"
```

```
value={feedback.name}
onChange={(e) => setFeedback({...feedback, name: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow-
sm focus:border-indigo-500 focus:ring-indigo-500"required/>
</div>
<div>
<label className="block text-sm font-medium text-gray- 700">Phone</label>
<input type="tel"
value={feedback.phone}
onChange={(e) => setFeedback({...feedback, phone: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow-
sm focus:border-indigo-500 focus:ring-indigo-500" required/>


</div>


<div>
<label className="block text-sm font-medium text-gray- 700">Email</label>

<input type="email"
value={feedback.email}
onChange={(e) => setFeedback({...feedback, email: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow-
sm focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>
```

```
</div>
<div className="grid grid-cols-1 md:grid-cols-2 gap-4">
<div>
<label className="block text-sm font-medium text-gray-700">Company
Name</label>
<input type="text"
value={feedback.companyName}

onChange={(e) => setFeedback({...feedback, companyName: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow- sm
focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>

<div>

<label className="block text-sm font-medium text-gray-700">Interview
Date</label>
<input type="date"
value={feedback.interviewDate}
onChange={(e) => setFeedback({...feedback, interviewDate: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow- sm
focus:border-indigo-500 focus:ring-indigo-500"
required/>
```

```
</div>

</div>
<div>
<label className="block text-sm font-medium text-gray-700">Interview
Experience</label>
<textarea value={feedback.experience}
onChange={(e)           =>       setFeedback({...feedback,      experience:
e.target.value})}
rows={4}
className="mt-1 block w-full rounded-md border-gray-300 shadow-sm
focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>


<div>
<label className="block text-sm font-medium text-gray-700">Interview
Questions</label>
<textarea value={feedback.questions}
onChange={(e) => setFeedback({...feedback, questions: e.target.value})}
rows={4}
className="mt-1 block w-full rounded-md border-gray-300 shadow-sm
focus:border-indigo-500 focus:ring-indigo-500"
required/>
```

```
</div>

<div>
<label  className="block      text-smfont-medium   text-gray
700">Suggestions</label>
<textarea value={feedback.suggestions}
onChange={(e)           =>       setFeedback({...feedback,      suggestions:
e.target.value})}
rows={3}
className="mt-1 block w-full rounded-md border-gray-300 shadow-sm
focus:border-indigo-500 focus:ring-indigo-500"/>
</div>

<button type="submit"
className="flex items-center justify-center w-full bg-indigo-600 text- white py-2
px-4 rounded-md hover:bg-indigo-700 transition-colors">
<Send className="h-5 w-5 mr-2" /> Submit Feedback
</button>
</form>
</div>
);
};
export default FeedbackForm;
```

NAV BAR

```
import React from 'react';
import { Menu, LogOut } from 'lucide-react';
const Navbar = ({ onLogout }: { onLogout: () => void }) => { return (
<nav className="bg-indigo-600 text-white p-4">
<div className="container mx-auto flex justify-between items-center">

<div className="flex items-center space-x-2">
<Menu className="h-6 w-6" />
<span className="text-xl font-bold">Placement Portal</span>
</div>
<button onClick={onLogout}
className="flex items-center space-x-2 hover:text-indigo-200"
>
<LogOut className="h-5 w-5" />
<span>Logout</span>
</button>
</div>
</nav>
);
}
export default Navbar;
```

FEEDBACK FORM

```
import React, { useState } from 'react'; import { Send } from 'lucide-react'; import

{ Feedback } from '../types';

interface

FeedbackFormProps

{
onSubmit: (feedback: Feedback) => void;

}

const FeedbackForm = ({ onSubmit }: FeedbackFormProps) => { const [feedback,

setFeedback] = useState<Feedback>({ registerNo: '',

name: '',

phone: '',

email: '', companyName: '', interviewDate: '', experience: '', questions: '',

suggestions: ''});

const handleSubmit = (e: React.FormEvent) => { e.preventDefault();

onSubmit(feedback);

};

return ()

<div          className="bg-white          rounded-lg          shadow-md          p-6">

<h2     className="text-2xl  font-bold       mb-6   text-gray-800">Interview

Feedback</h2>

<form onSubmit={handleSubmit} className="space-y-4">
```

```
<div className="grid grid-cols-1 md:grid-cols-2 gap-4">
<div>
<label className="block text-sm font-medium text-gray-700">Register
Number</label>
<input type="text"value={feedback.registerNo}

onChange={(e)           =>       setFeedback({...feedback,      registerNo:
e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow- sm
focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>

<div>
<label  className="block      text-smfont-medium   text-gray-
700">Name</label>
<input type="text"
value={feedback.name}
onChange={(e) => setFeedback({...feedback, name: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow-
sm focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>
```

```
<div>

<label  className="block      text-smfont-medium   text-gray-
700">Phone</label>
<input type="tel"
value={feedback.phone}
onChange={(e) => setFeedback({...feedback, phone: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow-
sm focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>

<div>
<label  className="block      text-smfont-medium   text-gray-
700">Email</label>
<input type="email"
value={feedback.email}
onChange={(e) => setFeedback({...feedback, email: e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow-
sm focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>

</div>
```

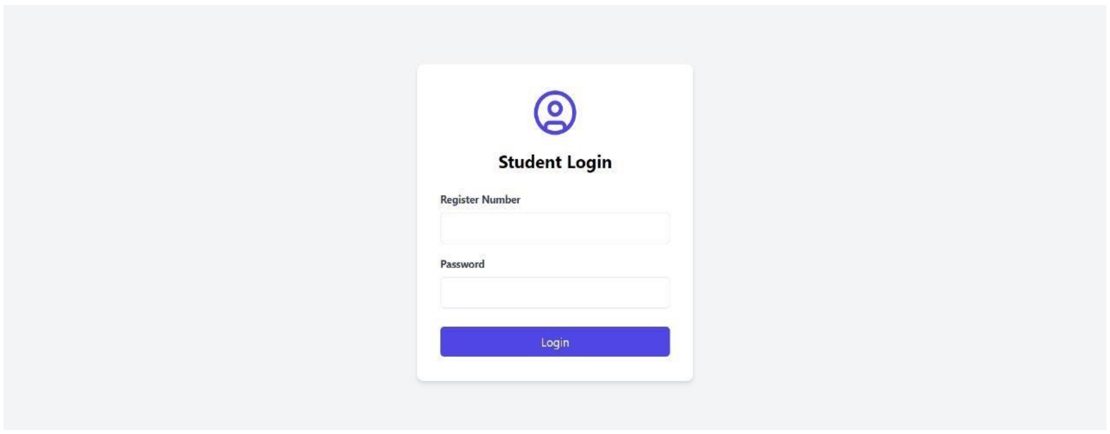
```
<div className="grid grid-cols-1 md:grid-cols-2 gap-4">
<div>
<label  className="block      text-smfont-medium   text-gray- 700">Company
Name</label>
<input type="text"
value={feedback.companyName}
onChange={(e)         =>       setFeedback({...feedback,      companyName:
e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow- sm
focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>
<div>
<label  className="block      text-smfont-medium   text-gray- 700">Interview
Date</label>
<input type="date"
value={feedback.interviewDate}
onChange={(e)         =>       setFeedback({...feedback,      interviewDate:
e.target.value})}
className="mt-1 block w-full rounded-md border-gray-300 shadow- sm
focus:border-indigo-500 focus:ring-indigo-500"
required/>
</div>
```

```
</div>



<div>
<label className="block text-sm font-medium text-gray-700">Interview
Experience</label>
<textarea value={feedback.experience}
onChange={(e)        =>     setFeedback({...feedback,     experience:
e.target.value})}
rows={4}
className="mt-1 block w-full rounded-md border-gray-300 shadow-sm
focus:border-indigo-500 focus:ring-indigo-500"
required
/>
</div>
<div>
<label className="block text-sm font-medium text-gray-700">Interview
Questions</label>
<textarea value={feedback.questions}
onChange={(e)        =>     setFeedback({...feedback,     questions:
e.target.value})}
rows={4}
className="mt-1 block w-full rounded-md border-gray-300 shadow-sm
focus:border-indigo-500 focus:ring-indigo-500"
```

```
required

/>

</div>

<div>

<label  className="block      text-smfont-medium   text-gray-

700">Suggestions</label>

<textarea value={feedback.suggestions}

onChange={(e)          =>       setFeedback({...feedback,      suggestions:

e.target.value})}

rows={3}

className="mt-1 block w-full rounded-md border-gray-300 shadow-sm

focus:border-indigo-500 focus:ring-indigo-500"

/>

</div>
<button type="submit"

className="flex items-center justify-center w-full bg-indigo-600 text- white py-2 px-
4 rounded-md hover:bg-indigo-700 transition-colors">

<Send className="h-5 w-5 mr-2"/>

SubmitFeedback

</button>

</form>

</div>

);

};

export default FeedbackForm;
```

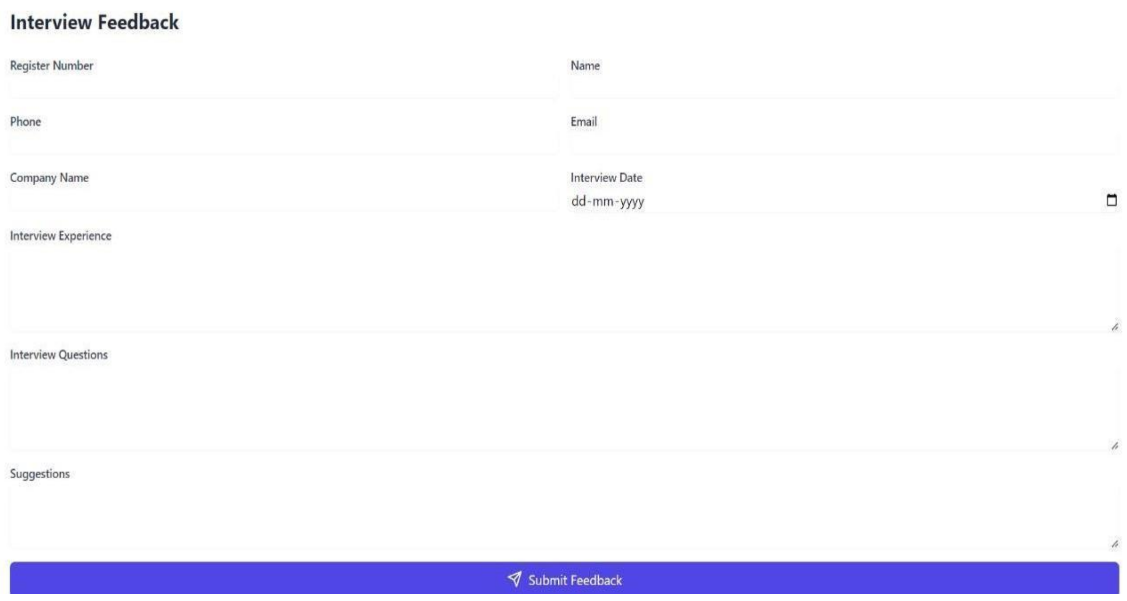
B.SCREENSHOTS

HOST LOGIN



This page is the Student Login Page for the Placement Management System. It serves as the entry point for students to access the platform. Students are required to provide their Register Number and Password to log in securely. The page is designed for simplicity and efficiency, ensuring quick authentication and access to the system

FEEDBACK FORM


Interview Feedback

Register Number	Name
Phone	Email
Company Name	Interview Date dd-mm-yyyy
Interview Experience	
Interview Questions	
Suggestions	

This page represents the Interview Feedback Form in the Placement Management System, designed to allow students to share their interview experiences and provide feedback about the placement process. It includes fields for entering the student's register number, name, phone number, and email for identification and verification purposes. Students can specify the company name and interview date, ensuring the feedback is relevant and detailed. Once all fields are completed, the Submit Feedback button allows students to save their responses securely in the database. This form promotes collaborative learning by creating a repository of valuable insights that can help other students prepare effectively for their placements.

REFERENCES:

- https://www.w3schools.com
- https://www.hackerrank.com
- https://www.codechef.com
- https://www.glassdoor.com