CAMPUSCOACH - AI POWERED PLACEMENT AND INTERVIEW GUIDE

A Project Report Submitted In partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology
In
Information Technology

By

K. SINDHU	**22N31A1292**
K.SAKETH	**22N31A1296**
M.AJAY	**22N31A12C4**

Under the esteemed guidance of

MS .G. LIKITHA REDDY

Assistance Professor

Department of Information Technology

Malla Reddy College of Engineering & Technology

(Autonomous Institution- UGC, Govt. of India)
(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA &NAAC with ‘A’Grade)
Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100

website: www.mrcet.ac.in

Malla Reddy College of Engineering and Technology

(Autonomous Institution-UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA&NAAC with „A‟ Grade)
Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100
Website: www.mrcet.ac.in

CERTIFICATE

This is to certify that this is the bonafide record of the Project Phase- 1 entitled **"CAMPUSCOACH – AI POWERED PLACEMNET AND INTERVIEW GUIDE"** submitted by **K SINDHU LAHARI (22N31A1292), K SAKETH (22N31A1296) and M AJAY KUMAR (22N31A12C4)** of B-Tech in the partial fulfillment of the requirements for the degree of Bachelor of Technology in Information Technology during the year 2025-2026. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide
MS. G. LIKITHA REDDY
Assistant Professor

Head of the Department
Dr.G.SHARADA
Professor

External Examiner

DECLARATION

We hereby declare that the Project titled "**CAMPUSCOACH – AI POWERED PLACEMNET AND INTERVIEW GUIDE**" submitted to Malla Reddy College of Engineering and Technology (UGC Autonomous), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH)for the award of the degree of Bachelor of Technology in Information Technology is a result of original work carried-out in this project .It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

K.SINDHU	**22N31A1292**
K.SAKETH	**22N31A1296**
M.AJAY	**22N31A12C4**

ACKNOWLEDGEMENT

We feel honored to place our warm salutation to our college Malla Reddy College of Engineering and Technology (UGC-Autonomous) for giving us an opportunity to do this Project as part of our B.Tech Program. We are ever grateful to our Principal Dr.VAKA MURALI MOHAN who enabled us to have experience in engineering and gain profound technical knowledge.

We express our heartiest thanks to our HOD, Dr. G. Sharada for encouraging us in every aspect of our course and helping us realize our full potential.

We would like to thank our Project Guide MS.G.Likitha Reddy for her regular guidance, suggestions and constant encouragement. We are extremely grateful to our Project Coordinator Ms. MS. G.Likitha Reddy for her continuous monitoring and unflinching co operation throughout project work.

We would like to thank our Class Incharge Ms .G .Likitha Reddy who in spite of being busy with her academic duties took time to guide and keep us on the correct path.

We would also like to thank all the faculty members and supporting staff of the Department of IT and all other departments who have been helpful directly or indirectly in making our project a success.

We are extremely grateful to our parents for their blessings and prayers for the completion of our project that gave us strength to do our project.

With regards and gratitude

K.SINDHU	**22N31A1292**
K.SAKETH	**22N31A1296**
M.AJAY	**22N31A12C4**

ABSTRACT

The growing challenge in campus placements today lies in the lack of a centralized and structured source of information for students. Most students find it difficult to access clear details about ongoing placement drives, company-specific eligibility criteria, recruitment rounds, and preparation materials. This information gap often results in confusion, missed opportunities, and uneven access to resources. To overcome this problem, CampusCoach was conceptualized and developed as an AI-powered placement and interview guidance platform. It acts as a one-stop solution where students can view real-time updates on campus drives, understand company-wise recruitment patterns, and access curated preparation materials such as aptitude papers, coding challenges, resume-building tips, and behavioral interview guides. The objective was to create a system that not only bridges the information divide but also provides students with personalized, intelligent assistance to prepare for interviews effectively.

From a technical perspective, CampusCoach is a full-stack MERN web application that combines modern web development with artificial intelligence integration. It uses MongoDB for database management, Express.js and Node.js for backend logic and RESTful API handling, and React.js for building a dynamic, responsive frontend interface. The application integrates Google's Gemini API to power an AI chatbot that provides real-time query support related to placements, interview processes, and preparation guidance. The backend ensures secure user authentication and data storage through JWT tokens and encrypted API communication. The platform was developed and tested using Visual Studio Code and hosted using Render for the backend and Vercel for the frontend deployment, ensuring smooth accessibility and scalability. By merging intelligent automation with robust web architecture, CampusCoach delivers a smart, interactive, and scalable solution that simplifies campus placement preparation and empowers students with equal access to the right information at the right time.

TABLE OF CONTENTS

1.INTRODUCTION

CampusCoach is an AI-powered placement and interview preparation platform developed to address the lack of a centralized and structured system for campus placement information. Many students struggle to find reliable details about eligibility criteria, recruitment rounds, and preparation materials, often relying on scattered sources or word-of-mouth updates. CampusCoach solves this problem by providing a single, interactive platform that consolidates real-time placement drive updates, company-specific information, downloadable study resources, and interview preparation guidance. It integrates an AI chatbot using Google's Gemini API, which allows students to ask placement-related queries and receive instant, intelligent responses, thereby enhancing accessibility and personalized learning.

Built using the MERN stack (MongoDB, Express.js, React.js, and Node.js), CampusCoach offers a complete full-stack architecture ensuring smooth data handling, authentication, and a dynamic user interface. MongoDB manages all placement data and resources, while Node.js and Express.js handle backend logic and REST APIs. The frontend, powered by React.js, provides a visually appealing and responsive interface with sections for company recruitment details, aptitude materials, and interview experiences. The application is securely authenticated using JWT tokens and hosted on Render and Vercel for backend and frontend deployment respectively. Overall, CampusCoach stands as a scalable and intelligent solution that simplifies placement preparation, promotes equal opportunity, and empowers students to approach interviews with confidence and clarity.

1.1 PROBLEM DEFINATION

In most colleges, placement-related information such as company eligibility criteria, recruitment rounds, and interview updates is scattered across multiple sources like notice boards, WhatsApp groups, or informal student discussions. This lack of a centralized and reliable platform often leads to confusion, missed opportunities, and uneven access to important details. Students also face challenges in finding authentic preparation materials and previous interview experiences, which are essential for effective placement readiness.CampusCoach aims to solve these issues by providing a unified digital platform where all placement-related information is organized and easily accessible. It centralizes company-specific details, round-wise processes.

1.2 EXISTING SYSTEM & PROPOSED SYSTEM

EXISTING SYSTEM

Many colleges currently rely on traditional methods to manage and share placement-related information, such as notice boards, emails, classroom announcements, or messaging groups. Students often receive company eligibility details, interview schedules, and preparation resources in an unorganized manner, leading to confusion and missed opportunities. There are also online platforms and forums that provide placement materials, but they are generic and not customized for a specific college or set of companies visiting the campus. Additionally, students have to visit multiple websites for aptitude tests, coding practice, and company insights, making the preparation process time-consuming and scattered. Some institutions use spreadsheets or Google Forms to track placements, but these methods lack real-time updates, interactivity, and integration with personalized learning support.

Limitation of Existing System

- Information about company drives and eligibility criteria is scattered across different channels with no centralized access point.These URLs were extracted only from E-mail and did not include other attack vectors.
- Students do not receive real-time updates about placement activities or schedule changes.
- No personalized guidance or AI-driven support is available to help students prepare effectively for interviews.
- Existing systems lack integration of study materials, previous year interview experiences, and video-based learning resources in one place.

PROPOSED SYSTEM

To overcome the drawbacks of the existing unstructured and manual placement information system, the proposed project **CampusCoach** introduces a centralized, AI-powered web application designed to streamline and enhance the entire campus placement process. This system integrates all placement-related data—such as company eligibility criteria, recruitment rounds, interview experiences, and preparation materials—into one dynamic, interactive platform. It leverages the power of artificial intelligence through the Google Gemini API, allowing students to interact with an intelligent chatbot that provides instant, relevant answers to their queries about placements, interviews, and preparation strategies. The platform is developed using the MERN stack (MongoDB, Express.js, React.js, and Node.js) to ensure a smooth user experience, scalability, and efficient data handling. Through this proposed approach, both students and placement officers can benefit from automation, real-time updates, and a unified interface for placement management and preparation.

Advantages:

- Provides a single centralized platform for all placement-related updates, resources, and company information.
- Integrates an AI chatbot powered by Google Gemini API for real-time, intelligent placement guidance.The use of a hybrid machine learning model that combines logistic regression, support vector machine, and decision tree algorithms to improve the accuracy and efficiency of phishing detection.
- Ensures secure and authenticated access using JWT tokens and RESTful API architecture.
- Improves user engagement with a responsive and visually appealing interface built using React.js.
- Enhances accessibility and student preparedness by providing 24/7 AI-driven support and organized learning content.

2. REQUIREMENTS SPECIFICATION

2.1 FUNCTIONAL AND NON FUNCTIONAL REQUIREMENTS

1. Functional Requirements

These define the specific behaviors and functions the system must provide:

- Students can view detailed information about companies visiting the campus for placements, including eligibility criteria and recruitment rounds.
- The platform includes an AI-powered chatbot that assists students with placement-related queries and provides real-time guidance.
- The website functions as a Single Page Application (SPA) to ensure smooth and fast navigation without page reloads.

2. Non-Functional Requirements:

- The user interface should be simple, intuitive, and easy for students to navigate.
- Security is ensured by restricting signup and login to valid college domain email addresses only.
- The system must maintain high reliability and uptime, especially during peak placement seasons.

APPLICATION DESCRIPTION

2.2 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS

Processor	**Intel i5**
RAM	**4GB and above**
SSD	**At least 50GB**

SOFTWARE REQUIREMENTS

Operating System	**Windows 10 and Higher**
Front-End	**Html, CSS, React.js**
Backend	**Node.js, Express.js,MongoDB**
Others	**Google Gemini API, VS Code**

3. SYSTEM DESIGN

3.1 ARCHIECTURE DIAGRAM

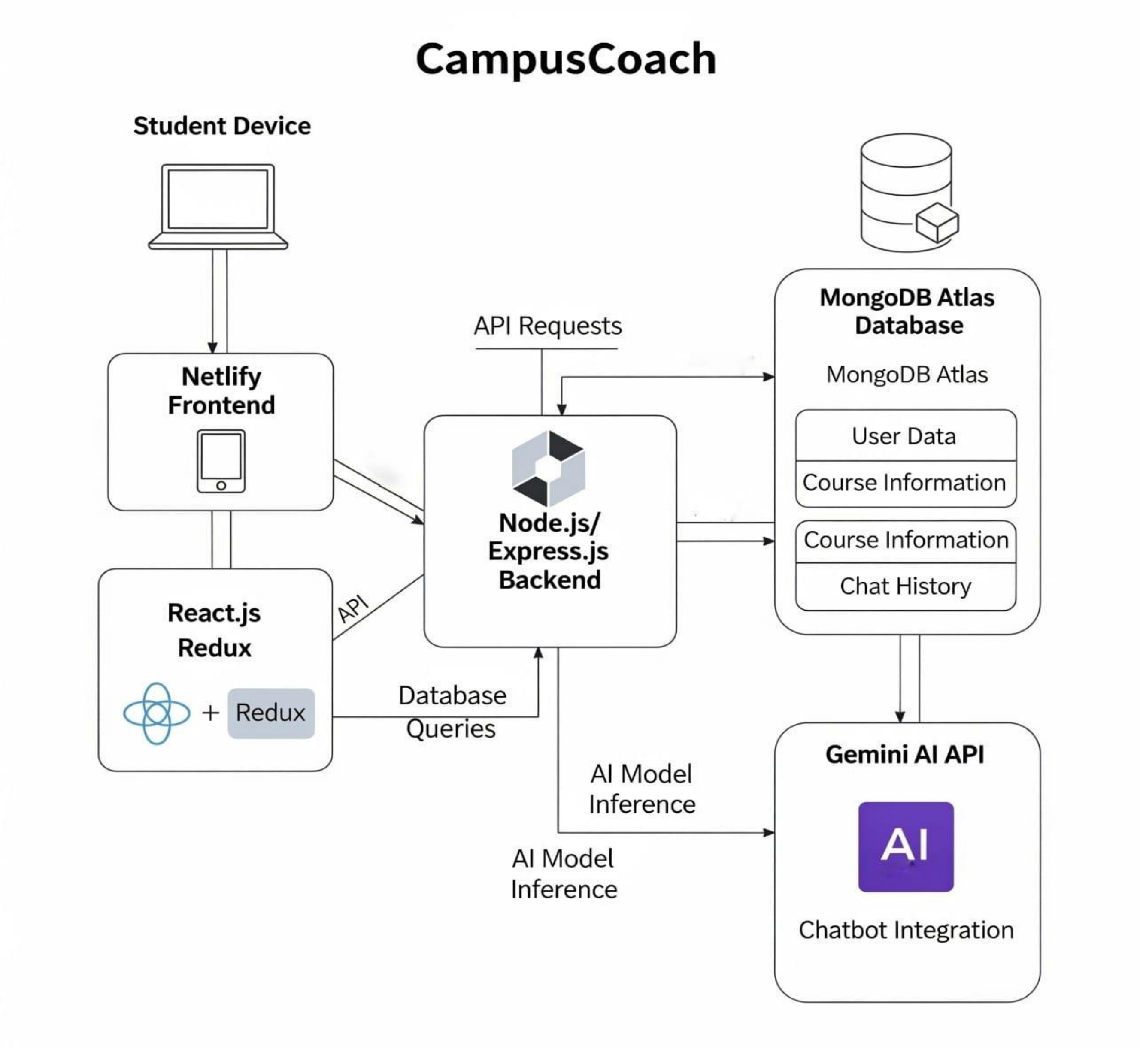


Fig:3.1-Architecture Diagram

3.2 UML DIAGRAMS

Usecase Diagram

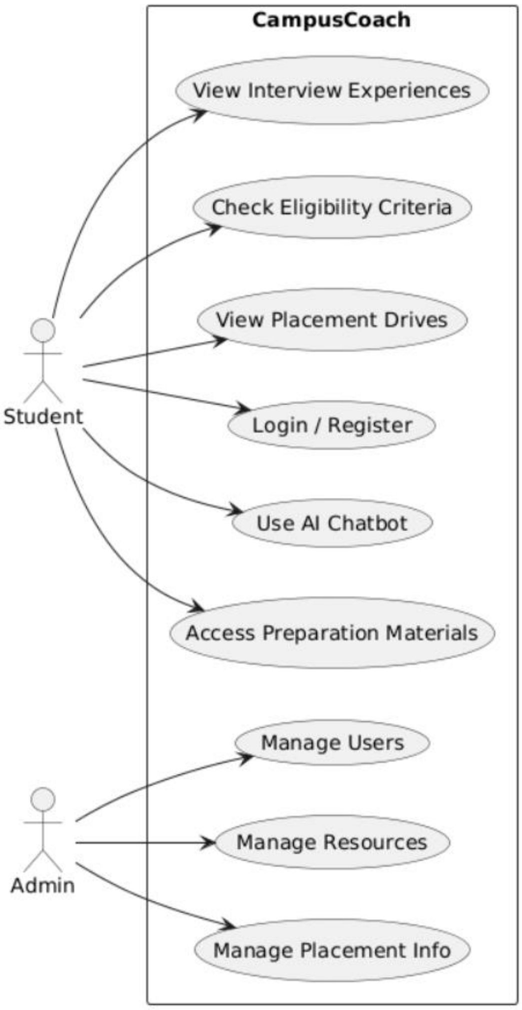


Fig:3.2-Usecase Diagram

Sequence Diagram

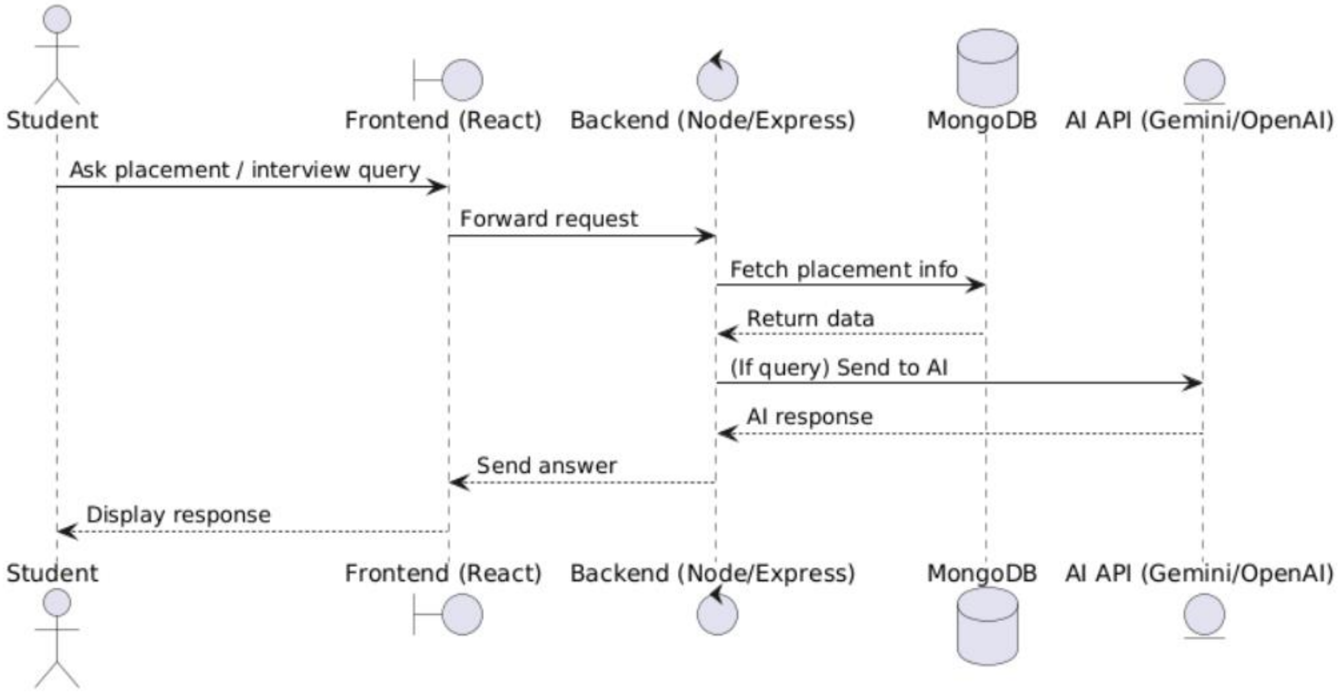


Fig:3.3-Sequence Diagram

4. METHODOLOGY

4.1 Technologies Used

The proposed system, *CampusCoach*, is designed to bridge the communication and information gap in campus placements through an AI-powered full-stack web platform. The methodology for developing this project involves multiple phases — requirement analysis, system design, development, integration, and deployment — all aligned with modern web development and AI integration standards. Each stage ensures the platform's scalability, performance, and usability for both students and placement coordinators.

The development process follows a **modular and iterative approach**, where each component of the system is built, tested, and integrated incrementally. The architecture of the project follows the **MERN stack** model, which includes MongoDB for database management, Express.js and Node.js for backend logic and API creation, and React.js for building an interactive and responsive frontend interface. The integration of the **Google Gemini API** introduces AI capabilities into the system, enabling intelligent responses through a chatbot embedded in the frontend.

1. **System Design and Architecture**
   The system follows a client-server architecture. The **frontend** (React.js) handles user interactions and requests, while the **backend** (Node.js + Express.js) processes these requests and communicates with the **MongoDB** database. Data such as company information, placement drives, and study materials are stored in MongoDB, allowing for efficient CRUD operations. The chatbot communicates with the Gemini API through secure endpoints to generate contextual responses for student queries.
2. **Frontend Development**
   The frontend is built using **React.js**, ensuring a fast, smooth, and dynamic user experience through a **Single Page Application (SPA)** model. React components are used to create separate modules for features such as company listings, interview preparation materials, downloadable resources, and chatbot interaction. CSS and responsive design principles ensure the platform functions effectively on both desktop and mobile devices. The interface emphasizes ease of navigation, clear content organization, and visual appeal, aligning with the needs of students preparing for placements.

3. **Backend Development**

   The backend is developed using **Node.js** and **Express.js**, which manage the server-side logic, routing, and communication with the database. RESTful APIs are created to handle all data exchanges between the client and server, including fetching company details, managing study materials, and processing chatbot queries. The backend also includes middleware for input validation, authentication, and error handling. JWT (JSON Web Token) authentication secures user sessions, ensuring that only registered students and authorized placement officers can access restricted sections of the platform.

4. **Database Management**

   MongoDB serves as the primary database for the project, providing a NoSQL structure that is flexible and efficient for handling large and dynamic data sets. It stores information such as student profiles, placement drives, company details, and uploaded resources.

5. **AI Chatbot Integration**

   The system integrates an AI chatbot using the **Google Gemini API**, which allows students to interact with the system conversationally. The chatbot provides responses related to company eligibility, interview processes, and preparation materials. The backend handles API calls to the Gemini service, processes the responses, and displays them in real-time on the frontend chat interface.

6. **Maintenance and Future Enhancements**

   Post-deployment, the system can be easily updated due to its modular architecture. Future enhancements may include integrating predictive analytics to suggest companies and preparation strategies based on student profiles, expanding AI capabilities for personalized learning paths, and adding real-time dashboards for placement analytics. Additionally, the platform can be extended into a mobile application using React Native for greater accessibility.

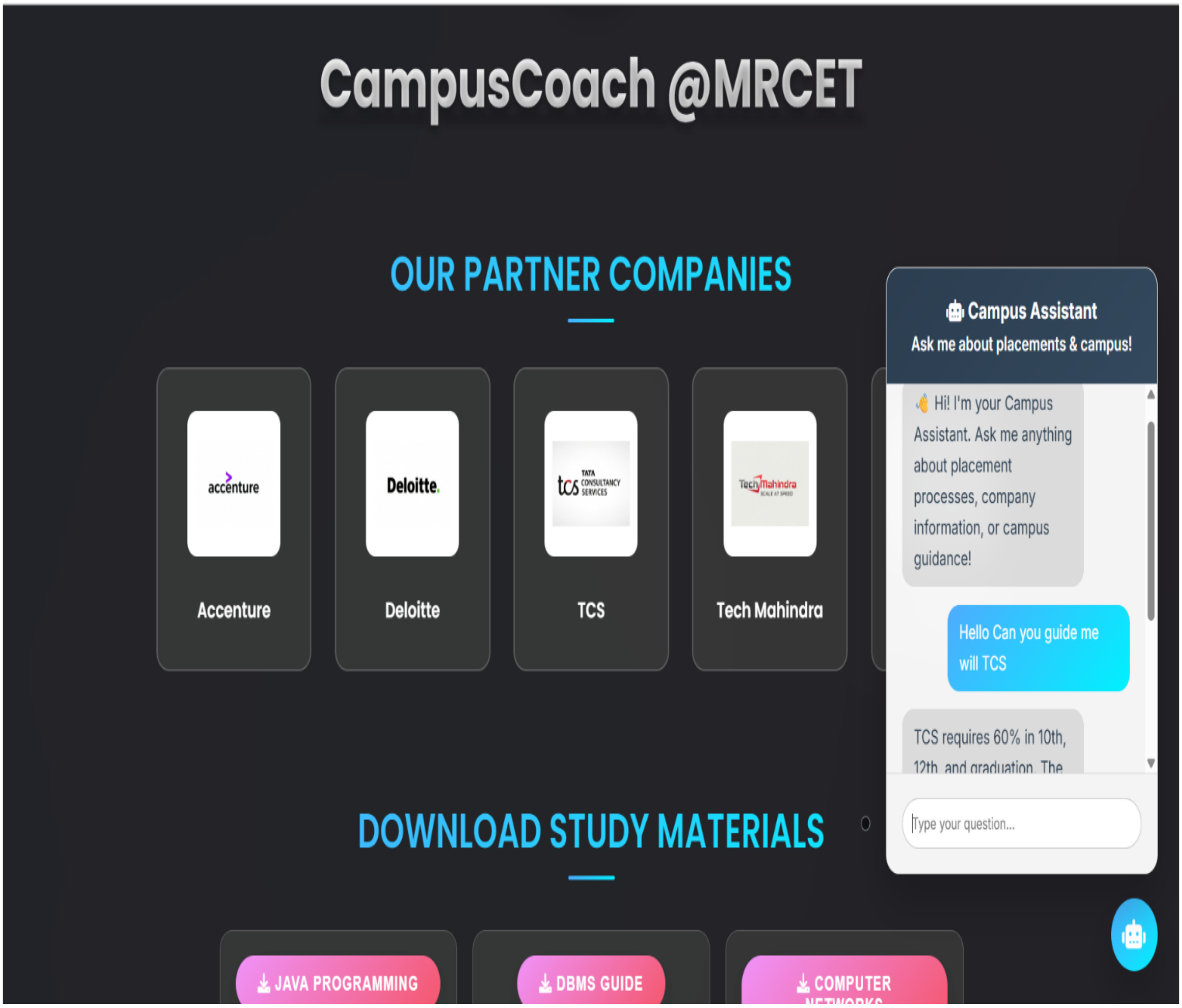
5. IMPLEMENTATION

SOURCE CODE

```
// server/index.js
const express = require('express');
const path = require('path');
const cors = require('cors');

const app = express();
app.use(cors());
app.use(express.json());

// --- In-memory sample data (replace with DB in production) ---
const companies = [
  { id: 'accenture', name: 'Accenture', eligibility: '65% / No active backlogs', rounds: ['Cognitive', 'Coding', 'Communication', 'HR'] },
  { id: 'tcs', name: 'TCS', eligibility: '60% in 10th/12th/Grad', rounds: ['Online Test', 'Coding', 'Technical', 'HR'] },
  { id: 'deloitte', name: 'Deloitte', eligibility: '60% throughout', rounds: ['Online Assessment', 'GD', 'Technical', 'HR'] },
  { id: 'techmahindra', name: 'Tech Mahindra', eligibility: '60% / No backlogs', rounds: ['Written Test', 'GD', 'Technical', 'HR'] },
  { id: 'cognizant', name: 'Cognizant', eligibility: '65% or 6.5 CGPA', rounds: ['Online Assessment', 'Coding', 'Technical', 'HR'] }
];

const materials = [
  { id: 'java', title: 'Java Programming Guide', file: '/assets/Java_Programming_Guide.pdf' },
  { id: 'dbms', title: 'DBMS Complete Guide', file: '/assets/DBMS_Complete_Guide.pdf' },
  { id: 'python', title: 'Python Programming Guide', file: '/assets/Python_Programming_Guide.pdf' }
];

// --- API routes ---
```

```
app.get('/api/companies', (req, res) => {
  res.json(companies);
});

app.get('/api/companies/:id', (req, res) => {
  const company = companies.find(c => c.id === req.params.id);
  if (!company) return res.status(404).json({ error: 'Company not found' });
  res.json(company);
});

app.get('/api/materials', (req, res) => {
  res.json(materials);
});

// Simple chatbot endpoint - keyword matching. Replace with real NLP/AI call.
app.post('/api/chat', (req, res) => {
  const { message } = req.body;
  if (!message) return res.status(400).json({ error: 'No message provided' });

  const m = message.toLowerCase();
  let reply = "I can help with company eligibility, selection rounds, and preparation tips. Ask about a specific company or topic.";

  if (m.includes('accenture')) {
    reply = 'Accenture: 65% / No active backlogs. Rounds: Cognitive, Coding, Communication, HR.';
  } else if (m.includes('tcs')) {
    reply = 'TCS: 60% in 10th/12th/Graduation. Rounds: Online Test, Coding, Technical, HR.';
  } else if (m.includes('deloitte')) {
    reply = 'Deloitte: 60% throughout. Expect group discussions and case studies along with technical rounds.';
  } else if (m.includes('prepare') || m.includes('preparation') || m.includes('study')) {
    reply = 'Study plan: Quantitative, Logical reasoning, Verbal ability, Core CS subjects, and coding practice (LeetCode/GFG).';
  } else if (m.includes('eligibility')) {
    reply = 'Eligibility varies by company (typically 60-65%). Check specific company page for details.';
  }

  res.json({ reply });
});

// --- Serve client build (if exists) ---
const clientBuildPath = path.join(__dirname, '..', 'client', 'build');
if (require('fs').existsSync(clientBuildPath)) {
  app.use(express.static(clientBuildPath));
  app.get('*', (req, res) => {
    res.sendFile(path.join(clientBuildPath, 'index.html'));
  });
}

// --- Start server ---
const PORT = process.env.PORT || 5000;
app.listen(PORT, () => {
  console.log(`Server running on http://localhost:${PORT}`);
});
```

6. OUTPUT SCREENS

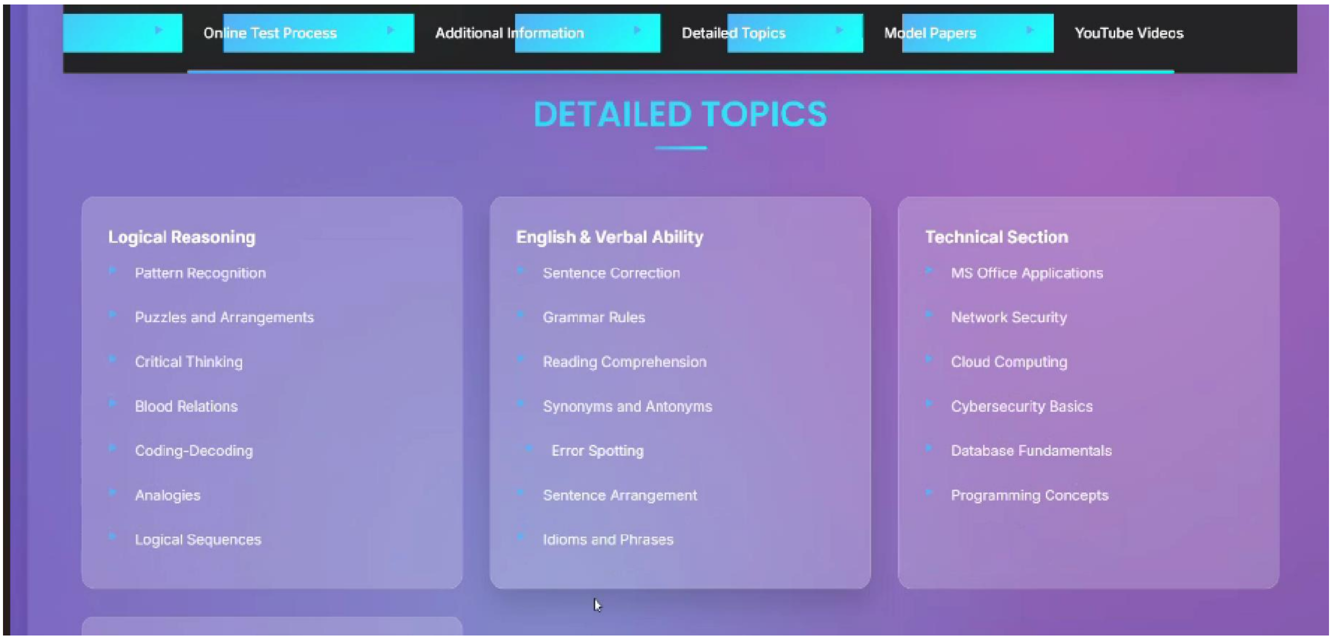


👋 Hi! I'm your Campus Assistant. Ask me anything about placement processes, company information, or campus guidance!

Hello Can you guide me will TCS

TCS requires 60% in 10th, 12th, and graduation. The

Fig:6.1



DETAILED TOPICS

Logical Reasoning

- Pattern Recognition
- Puzzles and Arrangements
- Critical Thinking
- Blood Relations
- Coding-Decoding
- Analogies
- Logical Sequences

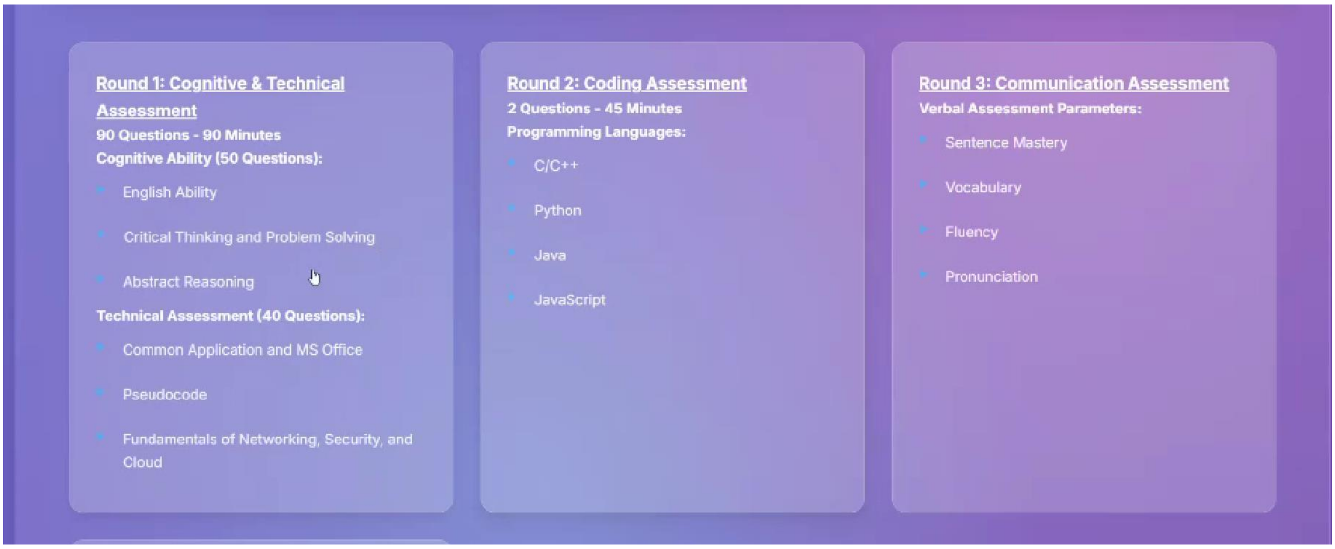
English & Verbal Ability

- Sentence Correction
- Grammar Rules
- Reading Comprehension
- Synonyms and Antonyms
- Error Spotting
- Sentence Arrangement
- Idioms and Phrases

Technical Section

- MS Office Applications
- Network Security
- Cloud Computing
- Cybersecurity Basics
- Database Fundamentals
- Programming Concepts

Fig:6.2



Round 1: Cognitive & Technical Assessment

90 Questions - 90 Minutes

Cognitive Ability (50 Questions):

- English Ability
- Critical Thinking and Problem Solving
- Abstract Reasoning

Technical Assessment (40 Questions):

- Common Application and MS Office
- Pseudocode
- Fundamentals of Networking, Security, and Cloud

Round 2: Coding Assessment

2 Questions - 45 Minutes

Programming Languages:

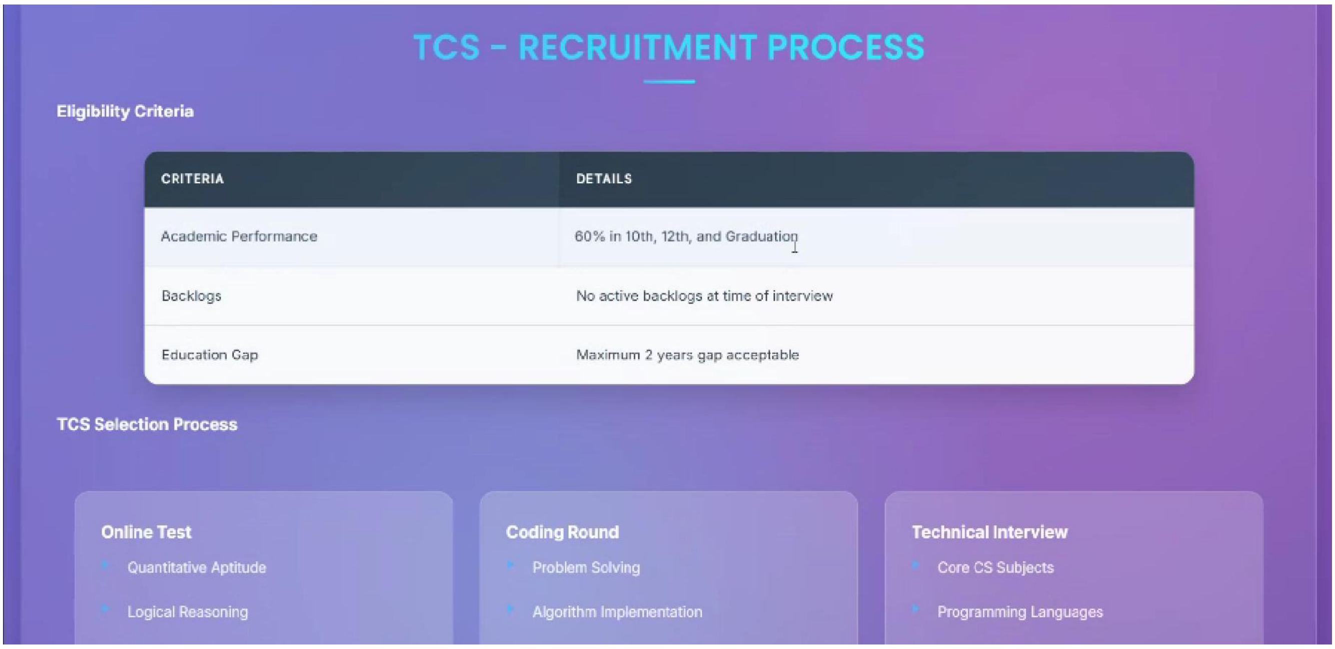
- C/C++
- Python
- Java
- JavaScript

Round 3: Communication Assessment

Verbal Assessment Parameters:

- Sentence Mastery
- Vocabulary
- Fluency
- Pronunciation

Fig:6.3



TCS - RECRUITMENT PROCESS

Eligibility Criteria

CRITERIA	DETAILS
Academic Performance	60% in 10th, 12th, and Graduation
Backlogs	No active backlogs at time of interview
Education Gap	Maximum 2 years gap acceptable

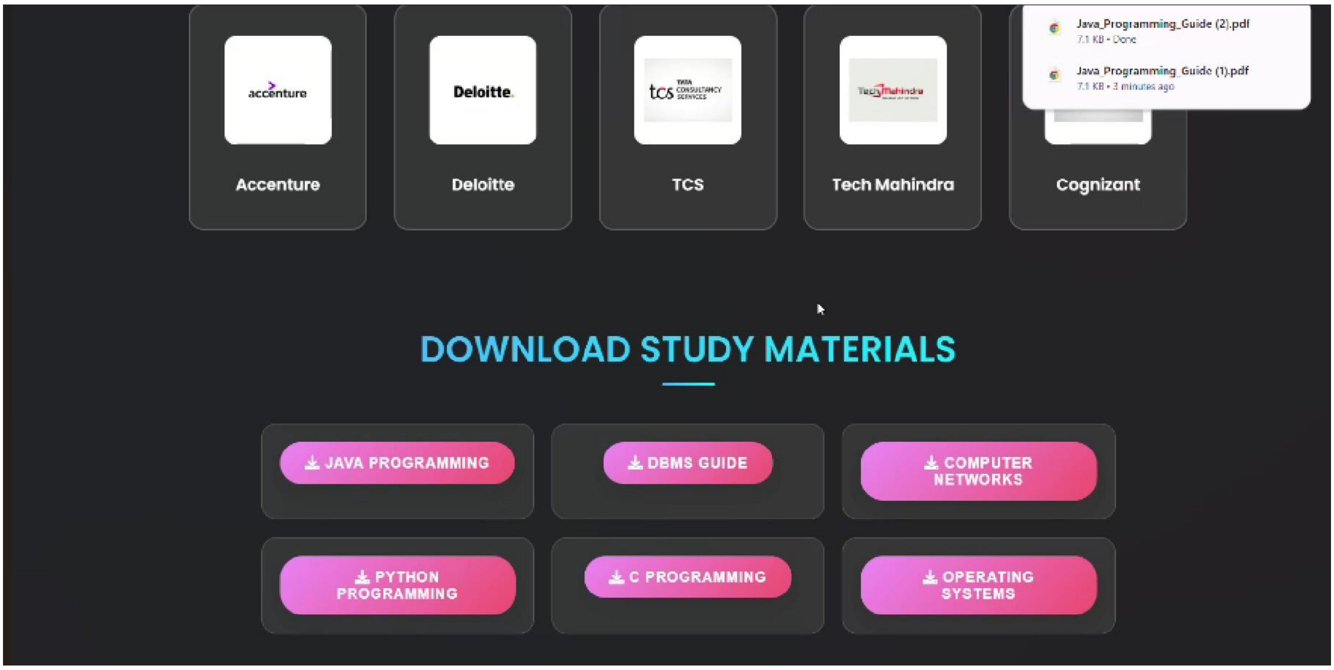
TCS Selection Process

Online Test
- Quantitative Aptitude
- Logical Reasoning

Coding Round
- Problem Solving
- Algorithm Implementation

Technical Interview
- Core CS Subjects
- Programming Languages

Fig:6.4

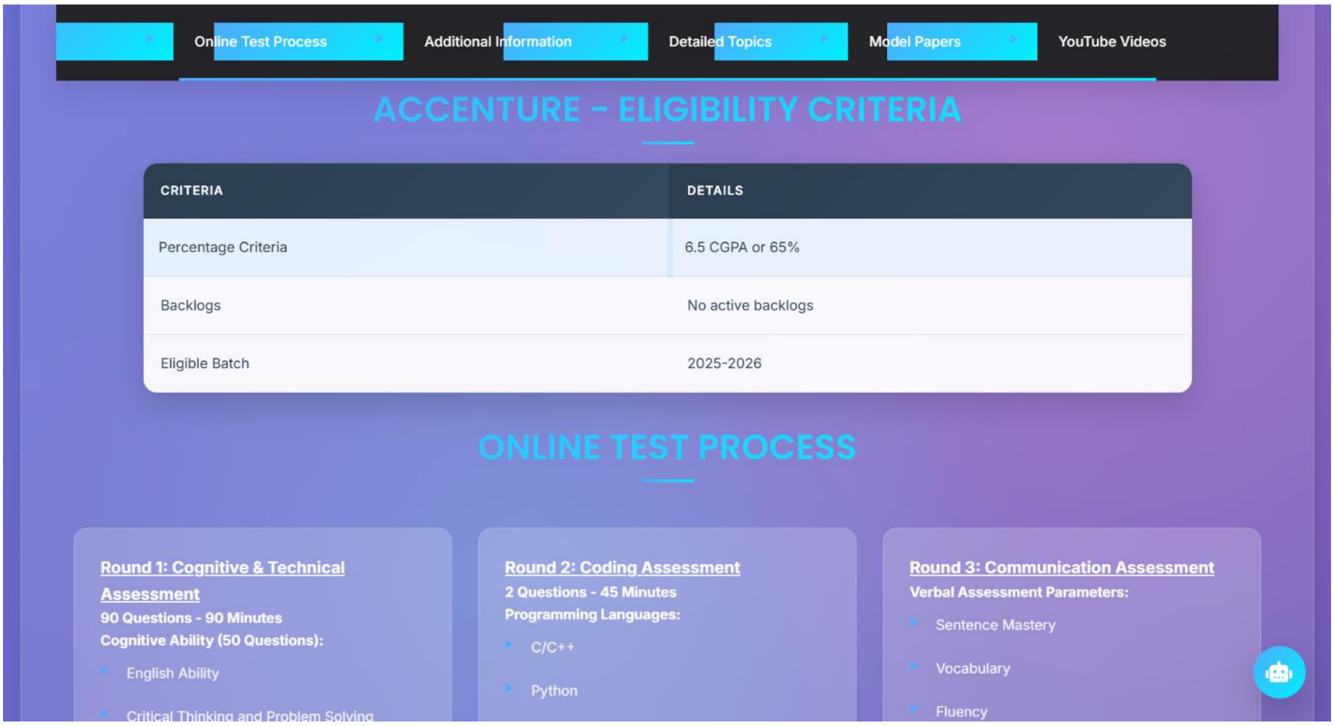


Java_Programming_Guide (2).pdf
7.1 KB • Done

Java_Programming_Guide (1).pdf
7.1 KB • 3 minutes ago

DOWNLOAD STUDY MATERIALS

Fig:6.5



Online Test Process | Additional Information | Detailed Topics | Model Papers | YouTube Videos

ACCENTURE – ELIGIBILITY CRITERIA

CRITERIA	DETAILS
Percentage Criteria	6.5 CGPA or 65%
Backlogs	No active backlogs
Eligible Batch	2025-2026

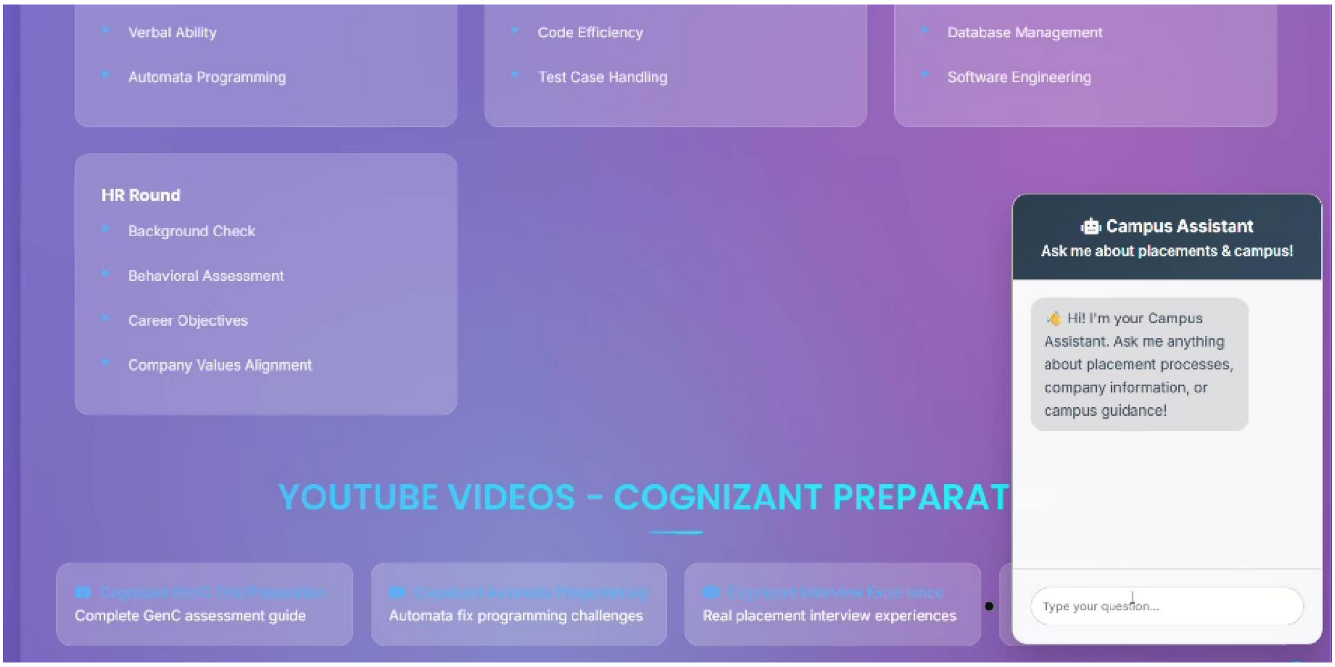
ONLINE TEST PROCESS

Round 1: Cognitive & Technical Assessment
90 Questions - 90 Minutes
Cognitive Ability (50 Questions):
- English Ability
- Critical Thinking and Problem Solving

Round 2: Coding Assessment
2 Questions - 45 Minutes
Programming Languages:
- C/C++
- Python

Round 3: Communication Assessment
Verbal Assessment Parameters:
- Sentence Mastery
- Vocabulary
- Fluency

Fig:6.6



- Verbal Ability
- Automata Programming

- Code Efficiency
- Test Case Handling

- Database Management
- Software Engineering

HR Round
- Background Check
- Behavioral Assessment
- Career Objectives
- Company Values Alignment

YOUTUBE VIDEOS – COGNIZANT PREPARAT

Cognizant GenC Test Preparation
Complete GenC assessment guide

Cognizant Automata Programming
Automata fix programming challenges

Cognizant Interview Experience
Real placement interview experiences

Campus Assistant
Ask me about placements & campus!

👋 Hi! I'm your Campus Assistant. Ask me anything about placement processes, company information, or campus guidance!

Type your question...

Fig:6.6

7.CONCLUSION

The CampusCoach project represents a comprehensive full-stack web-based platform developed to streamline and enhance the campus placement preparation process for students. It serves as a centralized digital ecosystem where learners can access company-specific recruitment details, download essential study materials, and interact with an intelligent AI chatbot for personalized guidance. By combining user-friendly design with robust technical architecture, the system bridges the gap between institutional training and real-world placement readiness.

The system effectively bridges the gap between students and placement information by providing dynamic company-specific details, downloadable study materials, and an AI-powered chatbot assistant for instant query resolution. The chatbot, powered by pattern-matching logic and extensible to machine learning APIs, simulates intelligent human interaction and provides students with quick, relevant, and contextual information regarding eligibility criteria, interview processes, and preparation strategies.

Through its modular architecture, the application demonstrates strong scalability and maintainability. The front-end React framework ensures responsive and dynamic interaction, while the Node.js backend serves structured company and material data through RESTful APIs. This architecture enables real-time data updates, seamless user experience, and potential integration with a database (MongoDB/MySQL) for user authentication and content management.

The CampusCoach system successfully achieves its goal of delivering an all-in-one placement preparation platform that combines automation, interactivity, and educational resources. By leveraging component-based design, responsive UI, and simulated AI intelligence, the project demonstrates how modern full-stack development can enhance campus training and student engagement.

Overall, the project provides a robust and adaptable framework for future enhancement—such as integrating real-time chatbot AI, user login systems, and admin dashboards for placement coordinators. With its intuitive design, modular logic, and extendable architecture, CampusCoach @MRCET stands as a reliable, efficient, and future-ready digital assistant for student placement success.

8. BIBLIOGRAPHY

1. **MDN Web Docs – HTML, CSS, and JavaScript Documentation**
   Mozilla Developer Network (MDN), 2025.
   Available: https://developer.mozilla.org

   ➤ Core reference for front-end development concepts, DOM manipulation, and responsive web design.
2. **Node.js & Express.js Official Guide**
   OpenJS Foundation, 2025.
   Available: https://expressjs.com

   ➤ Describes server-side routing, REST API creation, and middleware integration applied in the backend of the system.
3. **"Design and Implementation of a Web-Based Placement Management System"**
   International Journal of Computer Applications (IJCA), Vol. 181, No. 19, 2018.
   Available: https://www.ijcaonline.org

   ➤ Discusses architecture and challenges similar to the CampusCoach platform for managing placement data and automating workflows.
4. **R. Kumar, S. Sharma, "AI-Powered Chatbot for Student Assistance"**
   International Journal of Innovative Research in Computer Science and Technology (IJIRCST), 2022.

   ➤ Explores chatbot implementation for educational institutions, focusing on student interaction and guidance.
5. **S. Gupta, A. Jha, "Full Stack Web Application Development using MERN Stack"**
   International Research Journal of Engineering and Technology (IRJET), Vol. 9, Issue 5, 2022.

   ➤ Highlights best practices in developing scalable and modular full-stack systems similar to CampusCoach.

`