

# **Placement Preparation Module Focusing on DSA**

*Project report submitted to  
Dr. Babasaheb Ambedkar Technological University, Maharashtra  
in partial fulfilment of the requirements for the award of  
the degree*

## **Bachelor of Technology In Computer Engineering**

*By*

**Suyash Patalbansi (2046491245052)  
Shreya Raut (2046491245047)  
Shreya Chinchmalatpure (2046491245048)  
Susmita Kujur (2046491245051)**

under the guidance of

**Prof. Sheetal Kale  
Professor and Head of the Department**



**Department of Computer Engineering  
Bajaj Institute of Technology  
Wardha-442 001 (India)  
2023-24**

# **BAJAJ INSTITUTE OF TECHNOLOGY, WARDHA**

## **DEPARTMENT OF COMPUTER ENGINEERING**

### **CERTIFICATE**



*This is to certify that the Project report titled*

**PLACEMENT PREPARATION MODULE FOCUSING ON DSA**

*has been successfully completed*

*by*

*Suyash Patalbansi, (2046491245052)*

*Shreya Raut, (2046491245047)*

*Shreya Chinchmalatpure, (2046491245048)*

*Susmita Kujur, (2046491245051)*

*in partial fulfilment of the requirements for the award of  
the degree in Bachelor of Technology, Computer Engineering*

**Prof. Sheetal Kale  
Guide**

Department of Computer Engineering  
Bajaj Institute of Technology, Wardha

**Prof. Sheetal Kale  
Head of the Department**  
Department of Computer Engineering  
Bajaj Institute of Technology, Wardha

**Dr. N. M. Kanhe  
Principal**  
Bajaj Institute of Technology,  
Wardha

**Date: 28/11/2023**

**Place: Wardha**

## **DECLARATION**

We, hereby declare that the project report titled “Placement Preparation Module Focusing On DSA” submitted by us to the Bajaj Institute of Technology, Wardha, in partial fulfilment of the requirement for the award of Degree of B. Tech. in Computer Engineering discipline is a record of bonafide project work carried out by us under the guidance of Prof. Sheetal Kale Professor and Head of the Department.

We, further declare that this submission by the undersigned represents our original work and We have quoted the references where others ideas/words have been included. We understand any violation of the above will levy a disciplinary action on us.

We, further declare that the work reported in this project report has not been submitted either in-part or in-full for the award of any other degree in any other Institute or University.

Date:- 28/11/2023

Place:- Wardha

<b>S. No.</b>	<b>Name of Student</b>	<b>PRN Number</b>	<b>Signature</b>
1.	Suyash Patalbansi	2046491245052	
2.	Shreya Raut	2046491245047	
3.	Shreya Chinchmalatpure	2046491245048	
4.	Susmita Kujur	2046491245051	

## **ACKNOWLEDGEMENT**

We wish to extend my sincere gratitude to project guide to Prof. Sheetal Kale, Professor and Head of the Department for having made available her valuable time, support and guiding us throughout this project with timely inputs and suggestions.

We are extremely grateful to Dr Narendra Kanhe, the Principal, Bajaj Institute of Technology, Wardha for providing required resources for the successful completion of our project.

Date: 28/11/2023

Place: Wardha

<b>S. No.</b>	<b>Name of Student</b>	<b>PRN Number</b>	<b>Signature</b>
1.	Suyash Patalbansi	2046491245052	
2.	Shreya Raut	2046491245047	
3.	Shreya Chinchmalatpure	2046491245048	
4.	Susmita Kujur	2046491245051	

## **ABSTRACT**

Data Structure and Algorithms is an important basic course for computer science and other information related majors. It carries a tremendous responsibility for cultivating students' computer program design capabilities and computational thinking skills. Data Structures is a course that combines theory with practice. Students are not only required to possess a solid foundation of higher mathematics and data structures, but also a strong ability for abstract thinking and computer programming. The traditional teaching of data structures pays most attention to mathematical reasoning, proving correctness and the complexity analysis of algorithms. Students need regular practice and programming skills to improve in their professional aspects.

To address this issue, we have developed a web application engineered on the MERN stack—MongoDB, Express.js, React.js, and Node.js. that provide students with DSA content and quizzes and related programming questions to improve user's skills and knowledge in DSA. The web application helps user prepare for placements and improve their knowledge and skills so that they perform better in exams. User Authentication utilizes secure authentication mechanisms for user account creation and login. Offers structured lessons on DSA concepts and provides interactive quizzes based on difficulty levels.

Website have integrated coding practice section featuring a versatile code editor supporting multiple programming languages and also implemented Q&A system for users to engage in discussions related to DSA topics. Implemented visual representations of performance metrics for enhanced user engagement. Chatbot feature is incorporated which utilizes ChatGPT API for user assistance that enhances user interactions through real-time responses and guidance. There is notification system which provide users reminders and Cheatsheets are provided for summarizing and quick revision of key time and space complexities for common algorithms and data structures.

**Keywords – DSA, MERN, Code Editor, Quizzes, Programming, Placements.**

## TABLE OF CONTENTS

<b>Chapter No.</b>	<b>Title</b>	<b>Page No.</b>
	Certificates	i
	Declaration	ii
	Acknowledgement	iii
	Abstract	iv
	Table of Contents	v
	Abbreviations	vii
<b>1</b>	<b>Introduction</b>	<b>1</b>
	1.1 Introduction	
	1.2 Motivation	
	1.3 Objectives	
	1.4 Scope of Work	
	1.5 Organization of report	
<b>2</b>	<b>Literature Survey</b>	<b>6</b>
	2.1 Literature Review	
	2.2 Summary of Literature Review	
<b>3</b>	<b>Methodology &amp; System Requirements and Specifications</b>	<b>13</b>
	3.1 Web Development	
	3.2 MERN Stack	
	3.3 Methodology	
	3.4 Data Structures and Algorithms	
	3.5 Methods/Algorithm	
	3.6 System Design	
	3.7 Software Requirement	
	3.8 Functional Requirements	

3.9 Non-Functional Requirements	
3.10 UML Diagram	
3.10.1 Use Case Diagram	
3.10.2 Sequence Diagram	
<b>4      Implementation of Project</b>	<b>32</b>
<b>5      Results and Discussions</b>	<b>38</b>
5.1 Home Page	
5.2 Roadmap Page	
5.3 Prepare Page	
5.4 Problems Page	
5.5 Cheatsheet Page	
5.6 Discuss Page	
5.7 Login Page	
5.8 User Dashboard	
5.9 Admin Dashboard	
<b>6      Conclusions and Future Scope</b>	<b>52</b>
5.1 Conclusions	
5.2 Future Scope	
<b>References</b>	<b>55</b>

---

## **ABBREVIATIONS**

<i>DSA</i>	Data Structures and Algorithms
<i>CS</i>	Computer Science
<i>IT</i>	Information Technology
<i>LMS</i>	Learning Management Systems
<i>API</i>	Application Programming Interface
<i>Q&amp;A</i>	Question and Answer
<i>HTTP</i>	Hypertext Transfer Protocol
<i>HTML</i>	Hypertext Markup Language
<i>CSS</i>	Cascading Style Sheets

# **CHAPTER 1**

## **INTRODUCTION**

---

### **1.1 Introduction**

Data structures and Algorithms are really important courses for every concept used in computer science and software engineering. There are many concepts involved in the subjects of Data structures and Algorithms. There will be many questions for educators like how to learn Algorithms and Data structures, as there are many concepts involved without getting confused during attending the course.

Learning problems in the data structures course have always been the topic of discussion among lecturers in computer science [1] because this course has a high number of failures. The lecturers have used various approaches, methods, strategies, patterns, and technologies, but there has been no appropriate solution to be applied. Different levels of acceptance, understanding or learning capacity for each student are a challenge for lecturers. The failure to learn the course data structure for students occurs because this course requires abstract thinking. Many theories require research and deep understanding. Also, the density of teaching materials and its contents and methods are different from one subject to another. This requires high thinking skills in order to understand and learn the data structures course [9].

Data Structures is a course that combines theory with practice. Students are not only required to possess a solid foundation of higher mathematics and data structures, but also a strong ability for abstract thinking and computer programming. The traditional teaching of data structures pays most attention to mathematical reasoning, proving correctness and the complexity analysis of algorithms. Developing computing skills at higher learning levels, including the ability to apply the learned concepts in practice, requires using active learning and motivational strategies. Active learning is especially important in introductory Data Structures courses

In order to improve the educational process, it is necessary to motivate and engage the student and make them practice regularly. The placement preparation DSA module is a dynamic website for students to build a strong foundation in data structures and algorithms, providing them with structured learning resources, visualized learning

materials, practice quizzes and challenging programming questions. The placement preparation DSA module website provides a comprehensive and supportive learning environment to the students to prepare better for the placements, and ultimately helping students master the essentials DSA concepts.

## 1.2 Motivation

Traditionally, instructors in lecture-based courses in introductory computer science have an expectation that the material presented in class is reinforced via textbook reading, lab exercises, assigned homeworks or quizzes. Given the rapid increases in the CS population over the past few years [7], these courses are large, and monitoring the performance of all students and reaching out to at-risk students is a challenge. While students may be given practice problems to reinforce material learned in class, the time-consuming nature of many of these assignments, both for students and for graders, mean they can only cover a small amount of the material that students are expected to understand. Students require feedback on their performance to either verify their comprehension or identify where they made mistakes. Identifying and addressing student's misconceptions early is vital to ensuring students have a strong foundation on which to build future knowledge and will help reduce their frustration. Another limitation is the finite set of examples and practice problems available to students under this model. Some students rely heavily on examples to learn material and the limited number presented in class or in the textbook may not be sufficient for students to gain comprehension, especially when instructors use the same textbook examples in their lectures. Likewise, whether homework problems come from the instructor or a textbook, the limited number of problems covering a particular topic in turn limits the amount a student is able to practice.

This has resulted in new ways of teaching such courses, broadly classified as active learning techniques, that can include any combination of lab-based instruction, flipped classroom settings, gamification, peer-learning, or use of multimedia content all of which attempt to better engage students. Our focus in this work is towards building new interactive learning modules and analyzing student performance in data structures and algorithms courses. These modules can help reinforce lecture content outside of class.

The nature of computing and the generation of students have changed drastically in the past years. Still, most higher education computing courses are taught in traditional ways that may not be adequate to keep up with the changing educational reality and may not support the necessary learning. Developing computing skills at higher learning levels, including the ability to apply the learned concepts in practice, requires using active learning and motivational strategies. Active learning is especially important in introductory Data Structures courses that teach fundamental conceptual and programming constructs [8]. Improving the learning of these concepts is crucial to enhance the formation of professionals in the area [3].

### **1.3 Objectives**

- Create a robust module for Data Structures and Algorithms (DSA) to aid in placement preparation.
- To implement topic-wise and level-wise quizzes along with challenging coding questions to enhance problem-solving skills.
- To generate detailed graphical reports to provide users with insights into their performance and learning activities.
- To provide users with concise study material covering the entire DSA syllabus for effective learning.
- To facilitate a user-driven question and answer system to encourage interaction and resolve doubts among users.
- Include a cheatsheet feature for quick and summarized revision of the DSA syllabus.
- To design a structured DSA learning roadmap to guide users in their preparation journey effectively.
- To allow users to view content without login but require authentication for quizzes, coding questions, discussions, and dashboard access. Implement access control for administrators.
- Provide administrators with tools to control and manage content, quizzes, and coding questions for effective platform administration.
- To integrate a code editor with support for multiple programming languages to cater to diverse user preferences.

- To implement a notification system to keep users informed about updates, reminders, and relevant activities on the platform.

## 1.4 Scope of Work

The implementation of a web-based platform to aid in placement preparation by providing a space for student to improve their knowledge and skills in Data Structures and Algorithms. Our focus in this work is towards building new interactive learning modules and analyzing student performance in data structures and algorithms courses by providing them with structured learning resources, visualized learning materials, practice quizzes and challenging programming questions. These modules can help reinforce lecture content outside of class.

By leveraging the capabilities of Web and Internet, the web application can be accessed through internet at anytime and anywhere. We have provided comprehensive study material covering the entire DSA syllabus. Also developed challenging programming questions and quizzes for different DSA topics and levels to reinforce and practice the knowledge gained through contents. programming questions will help users to develop programming skills and improve problem solving ability. The platform's design focuses on an intuitive and user-friendly interface for seamless navigation. Set up a secure user authentication system to implement access controls to restrict certain features to authenticated users. These will help in tracking user's progress. We have developed a user-driven question and answer system to facilitate doubt resolution. Include a cheatsheet feature for users to quickly revise and summarize key concepts. Platform include a structured DSA learning roadmap to guide users through their preparation. Platform have integrated code editor with support for multiple programming languages. Users informed about updates, reminders, and activities through notification system to keep them engaged and motivated. Users are provided graphical reports with insights into their performance and learning activities. Administrators are given tools and features for to manage and control content on the platform.

## **1.5 Organization of Report**

The report is organized in six chapters

**CHAPTER 1:** Introduction. The chapter gives a brief introduction to Data Structures and Algorithms and its importance in computer science. The chapter also discusses the motivation and objective behind the project it also contains the chapter organization.

**CHAPTER 2:** Literature Review. The chapter gives an overview on some of the prior research done which are directly related to the project. The chapter also contains a summary of literature review.

**CHAPTER 3:** Methodology/System requirements and Specification The chapter explains the working and the methodology of developing web application for learning DSA. It also contains the technology, frameworks and tools used in project. And also, the hardware and software requirements of the project.

**CHAPTER 4:** Implementation of Project. The chapter contains implementation of project work. The chapter gives overview of implementation of software, tools and framework used in this project.

**CHAPTER 4:** Results and Discussions. The chapter contains results obtained by the project. The chapter gives explanation of results obtained through our web application.

**CHAPTER 6:** Conclusion and Future Scope. The chapter gives the conclusion to the project work done in this report. The chapter discusses the scope of The Placement Preparation DSA Module. The potential of Placement Preparation DSA Module is also mentioned in this chapter.

# CHAPTER 2

## LITERATURE SURVEY

---

### 2.1 Literature Review

#### 2.1.1 Learning Management System for Data Structures and Algorithm

**Year of Publication:** 2021

**Author:** Marco Paulo J. Burgos

**Objective:** To develop of Learning Management System (LMS) for Data Structures and Algorithms with the main feature that allows students to take modular online learning.

**Contribution:** The study addressed the adjustments of academic institutions to online class and modular learning caused by the Covid-19 pandemic. The research and development approach includes (i) stages of system development using the Waterfall Method, (ii) level of acceptability of the developed system based on the ISO 25010 standard, (iii) difference in the evaluation of the three groups of respondents, (iv) challenges encountered while using the system, and (v) implementation plan. The respondents chosen through convenience sampling were 60 students, 15 faculty members, and 15 Information Technology (IT) experts. The checklist-format questionnaire was based on the ISO 25010 which determined acceptability using functional suitability, performance efficiency, usability, and reliability criteria. An interview was also conducted to evaluate respondents' experiences using the system. Based on the respondents' evaluation, the developed system was 'acceptable' as reflected by the obtained weighted means. Further results showed no significant difference in the evaluation of the three groups of respondents in terms of functional suitability, performance efficiency, usability, and reliability.

**Conclusion:** The Waterfall Model was used in the development of LMS for Data Structures and Algorithms. This method encompassed several methods which are requirements analysis, system design, implementation, testing, deployment, and maintenance phases. The overall assessment indicated that the system is 'acceptable' as supported by the weighted means of 4.49 for the students, 4.43 for the faculty members, and 4.38 for the IT experts. Furthermore, the weighted means for each component are 4.51 for functionality, 4.38 for performance, 4.45 for usability, and 4.39 for reliability. This numerical evidence proves that the LMS for Data Structures and Algorithms serves its purpose and greatly benefits the end users especially in terms of functionality, performance, usability, and reliability in accordance with the standards of ISO 25010. An implementation plan was also developed for the use of the LMS. It shows comprehensive procedures that elucidate the distinct functionalities of the system. A Gantt chart was created to have an idea of the timeline for the implementation. The methods in the user manual have been evaluated and tested to facilitate the understanding of the detailed usage of the LMS.

## **2.1.2 Some ways of increasing the efficiency of teaching data structures**

**Year of Publication:** 2022

**Author:** Zarema S. Seidametova.

**Objective:** To present some ways of increasing the efficiency of teaching data structures (hashing, trees) during the Algorithms and data structures course.

**Contribution:** Nowadays computer modelling and simulation are widely used in computer science education. Many concepts of computer science and software engineering are difficult for students to understand. Algorithms and Data structures are one of the most important subjects in Computer Science, which is required by all programmers. Programmers need to know how to handle the data in huge quantities, how the data should be stored in the memory and how the memory should be used efficiently. There are several ways of increasing the efficiency of the educational process. One of such ways is visualization, the other way is flipped classroom concept. They conducted two experiments with four study groups of two subjects of the Algorithms and Data Structures – (1) Hashing, (2) Trees (BST, RBT, AVL), – taught second-year bachelor students. In the first experiment, study groups were prepared on the subjects (1) and (2) by the same level of teaching technique with or without visualization tools. Unlike the first experiment, in the second experiment, they used flipped classroom concepts for one study group and a standard way of teaching for another group.

**Conclusion:** There are a lot of approaches to the implementation of some ways of increasing the efficiency of teaching data structures (hashing, trees). They conducted two experiments with four study groups of two subjects of the Algorithms and Data Structures – (1) Hashing, (2) Trees (BST, RBT, AVL), – taught second-year bachelor students. In the first experiment, study groups were prepared on the subjects (1) and (2) by the same level of teaching technique with or without visualization tools. Unlike the first experiment, in the second experiment, they used flipped classroom concept for one study group and a standard way of teaching for another group. Flipped classroom and visualization as pedagogical tools can be used to teach students the necessary skills and competencies in algorithms and data structures.

## **2.1.3 Interactive Data Structure Learning Platform**

**Year of Publication:** 2014

**Author:** Estevan B. Costa1, Armando M. Toda, Marcell A. A. Mesquita,

Fabio T. Matsunaga, Jacques D. Brancher

**Objective:** To develop and implement an interactive web learning environment, called DSLEP (Data Structure Learning Platform), to support students in higher education IT courses.

**Contribution:** The advent in technology in the past few years allowed an improvement in the educational area, as the increasing in the development of educational system. One of the techniques that emerged in this lapse is called Gamification, defined as the utilization of video game mechanics outside its bounds. Researchers in this area found

positive results in the application of these concepts in several areas from marketing to education. Among higher education, focusing on IT courses, Data Structures can be considered an important subject to be taught, as they are base for many systems. The system includes basic concepts taught on this discipline as stacks, queues, lists, arrays, trees. The system is also implemented with gamification concepts, as points, levels, and leader boards, to motivate students in the learning process and stimulate self-learning.

**Conclusion:** This work presents the development and implementation of an Interactive Learning Environment for Data Structures which was also proposed by other authors. To improve the gamification core of the system, 33 achievements and 9 leader boards were made to be used within the system and its 7 activities. The objective of this work was achieved since the platform is finished and already being tested with students from local colleges. Also, it is implemented with the gamification concepts to engage and stimulate their motivation. To secure and guarantee the information stored within the system, we used a tool (Clay.IO) to manage the gamification concept. This also allowed the user to exchange personal information with others, as a social network system. As future works, new activities and new concepts will be implemented. Also, it is in development a profile analyser to work within the system so the teachers may create a 'class' to monitor the student's development and progress within the lessons. Those analysis also may determine which activities may be improved or balanced.

#### **2.1.4 Research on Data Structure Course Teaching Based on Open Teaching System Model**

**Year of Publication:** 2019

**Author:** Lasheng Yu, Xiaopeng Zheng and Ye Biao

**Objective:** To develop students' learning initiative, thinking innovation and independent thinking ability in the process of data structure teaching which may be useful to the other instructors in the universities.

**Contribution:** Data Structure is an important basic course for computer science and other information related majors. It carries a tremendous responsibility for cultivating students' computer program design capabilities and computational thinking skills. Data Structure is the core course in the computer science curriculum system, and corresponds to the theories, abstractions, and design methodology of problem solving in the discipline. This article mainly discusses how to develop students' learning initiative, thinking innovation and independent thinking ability in the process of data structure teaching which may be useful to the other instructors in the universities.

**Conclusion:** This paper takes the data structure as an example to illustrate the application of open teaching in the data structure curriculum teaching system, from the curriculum syllabus, teaching content arrangement, teaching process principles and methods, coursework settings, course assessment mechanism and other teaching process. Starting from the requirements of bringing student's subjective initiative into cultivating students' innovative awareness and ability to innovate, we will grasp the main problems of students' ability to solve complex problems, and do a good job of continuously improving the quality of course teaching and the mastery of students'

ability, and promote the achievement of graduation requirements, which in turn supports the realization of training goals. Through the reform of the data structure course teaching model, significant results have been achieved, teaching quality and teaching levels have been improved, and the teaching reform has been demonstrated as a model, and students' ability in innovation and entrepreneurship has been improved and local economic development has been served.

### **2.1.5 Studied Questions in Data Structures and Algorithms Assessments**

**Year of Publication:** 2023

**Author:** Iris Gaber, Amir Kirsh and David Sttater

**Objective:** To argue that a good exam should contain questions that students have seen during the semester, and that the grading of those questions should be strict.

**Contribution:** Designing a proper exam that accurately evaluates students' knowledge and skills is one of the important tasks of every teacher. The format of the exams affects the way students learn throughout the course, and a well-designed exam can enhance meaningful learning. We describe a case study which, over three semesters, supports the claim that answering these questions require the "Understand" level of Bloom's taxonomy, and that this strategy fosters more meaningful learning and better assesses students' knowledge.

**Conclusion:** They showed that having "studied questions" in the exam does not impair the assessment process. Solving studied questions is not trivial, and, in our case study, still produced variance within the grades, so these questions can still differentiate between students who understood the material and those who did not. Studied questions are also perceived as fair. Students do not consider something they saw as tricky or difficult. It also promotes their self-efficacy and reduces their anxiety, as presented in the questionnaire that we distributed. There are several ways to present "Studied Questions" in an exam. One of the ways is to create a pool of questions that would be shared with the students, then have the exam draw questions from that pool. Our experience is that when a large percentage of the exam (between 50% and 70%) is based on studied questions, it enhances meaningful learning, improves the fairness of the exam and reduces students' anxiety.

### **2.1.6 Learning Management System (LMS) among University Students: Does It Work?**

**Year of Publication:** 2013

**Author:** Nor Azura Adzharuddin and Lee Hwei Ling

**Objective:** To provide several insights of the LMS phenomenon.

**Contribution:** The Learning Management System (LMS) has been established in a number of universities worldwide to help connect students and lecturers without the confines of the traditional classroom. It is an environment with digital software which is designed to manage user learning interventions as well as deliver learning content and resources to students. Since the LMS system has already been implemented and it has also been made compulsory for the lecturers to apply in their daily lectures, it is

vital to identify feedback of students as users of LMS. Previous studies have shown various findings in relation to the impact of using LMS in the higher learning environment in various universities worldwide.

**Conclusion:** In this modern world where information is disseminated quickly via the internet, the LMS is an essential tool for university students as not they can keep updated with their coursework, but get instant notifications pertaining to their daily assignments. In turn, lecturers have an easier time reaching out to their students out of class hours and can instantly update them over the LMS about issues regarding their coursework. Although those using the LMS might encounter some problems, it's all part and parcel of learning and using a whole new system altogether. Universities should provide proper training and guidance for students and lecturers using the LMS, as well as have a team which is on-call at all times to solve any problems that may arise. Nevertheless, most university students have access to their university's LMS or similar systems that help to enhance their learning process. Many have also expressed positive views about LMS, therefore proving that LMS is a necessary implementation in all universities worldwide.

### **2.1.7 Visualization, Assessment and Analytics in Data Structures Learning Modules**

**Year of Publication:** 2018

**Author:** Matthew Mcquaigue, David Burlinson, Kalpathi Subramanian, Erik Saule, Jamie Payton

**Objective:** To build interactive teaching modules for data structures and algorithms courses.

**Contribution:** In recent years, interactive textbooks have gained prominence in an effort to overcome student reluctance to routinely read textbooks, complete assigned homeworks, and to better engage students to keep up with lecture content. Interactive textbooks are more structured, contain smaller amounts of textual material, and integrate media and assessment content. They build interactive teaching modules for data structures and algorithms courses with the following characteristics, (1) the modules are highly visual and interactive, (2) training and assessment are tightly integrated within the same module, with sufficient variability in the exercises to make it next to impossible to violate academic integrity, (3) a data logging and analytic system that provides instantaneous student feedback and assessment, and (4) an interactive visual analytic system for the instructor to see students' performance at the individual, sub-group or class level, allowing timely intervention and support for selected students.

**Conclusion:** Their current implementation demonstrates a prototype of the parts of the system. They have presented new learning modules that can be used as a part of an interactive textbook in the data structures and algorithms courses. Their module are built to work with the OpenDSA system, using JSav library for generating the module visualizations that have been customized to provide a highly interactive, visual and engaging experience for students. Much work remains to be done in connecting the various components and integrating the data into the visualization modules, prior to

routine use as part of the LMS. This will be followed by formal user studies in data structures and algorithms courses. Finally, more work is needed in developing the visual analytics system to better understand the needs of the instructors as well as identification of at-risk students for timely intervention and support.

### **2.1.8 Active Learning through Game Play in a Data Structures Course**

**Year of Publication:** 2018

**Author:** Darina Dicheva, Austin Hodge.

**Objective:** To present an educational game intended to explicate several features hindering student's understanding of the data structure Stack on conceptual and practical level.

**Contribution:** Data Structures is a fundamental Computer Science discipline, challenging students' abstract thinking, problem solving and programming skills. In this paper, they present an educational game intended to explicate several features hindering students' understanding of the data structure Stack on conceptual and practical level. The game targets all three aspects of teaching data structures: conceptualization, application and implementation. These aspects are embodied as three parts of the game tied together through a meaningful storyline. The application part targets the use of stacks to solve problems, such as converting arithmetic expressions from infix to postfix notation and evaluating postfix and infix expressions. The implementation part involves solving Parson's problems and writing Java code for implementing the methods of the Stack class. The results of the conducted evaluation of the game show statistically significant learning gains for the students and a strong positive attitude towards this type of active learning.

**Conclusion:** Due to the difficulties that many students face in learning data structures, it is often problematic to impart working knowledge of their creation and operation by using only traditional teaching methods. As a response to these challenges, we designed the Stack Game. The objective was to enable students to acquire a correct cognitive model of the stack concept and behaviour through gameplay. The Stack Game was incorporated as part of the learning activities in a regular Data Structures course, to promote deeper and more active learning. Our experience with designing and integrating the game into the classroom experience shows the importance of embedding sound instructional design into games in order to create effective learning tools. In particular, offering a game that includes all aspects of the targeted topic helps students to reflect on the learned concepts in a holistic way. The manner in which learning experiences from games are incorporated into the classroom is another important factor to its effectiveness. Instructors can further facilitate the transfer of skills by discussions connecting the game experience with relevant concepts students are learning in class.

## **2.2 Summary of Literature Review**

Above mentioned research papers and official documentation explains the architecture of Learning Management System for Data Structures and Algorithm and its implementation. Also, there is some documentation that discusses its implementation through visualizations and game-playing. It highlighted the importance of active learning in learning data structures and algorithms. It also provided various possible solutions using various methods of learning data structures and algorithms. These papers discuss the various techniques to improve programming skills of the user. Some documentation discussed about assessment and analytics of users and its importance in improving data structures and algorithms knowledge in user. There is also problem with quality of exam questions by which study and retention of data structure and algorithms. The nature of computing and the generation of students have changed drastically in the past years. Still, most higher education computing courses are taught in traditional ways that may not be adequate to keep up with the changing educational reality and may not support the necessary learning. Developing computing skills at higher learning levels, including the ability to apply the learned concepts in practice, requires using active learning and motivational strategies. Improving the learning of these concepts is crucial to enhance the formation of professionals in the area

# CHAPTER 3

## METHODOLOGY / SYSTEM REQUIREMENTS AND SPECIFICATIONS

---

### **3.1 Web Development**

Web development refers to the creating, building, and maintaining of websites. It includes aspects such as web design, web publishing, web programming, and database management. It is the creation of an application that works over the internet i.e. websites. Web development is closely related to the job of designing the features and functionality of apps (web design). The term development is usually reserved for the actual construction of these things (that is to say, the programming of sites).

The basic tools involved in web development are programming languages called HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), and JavaScript. There are, however, a number of other programs used to “manage” or facilitate the construction of sites that would otherwise have to be done “from scratch” by writing code. A number of content management systems (CMS) fall into this category, including WordPress, Joomla!, Drupal, TYPO3, and Adobe Experience Manager, among others.

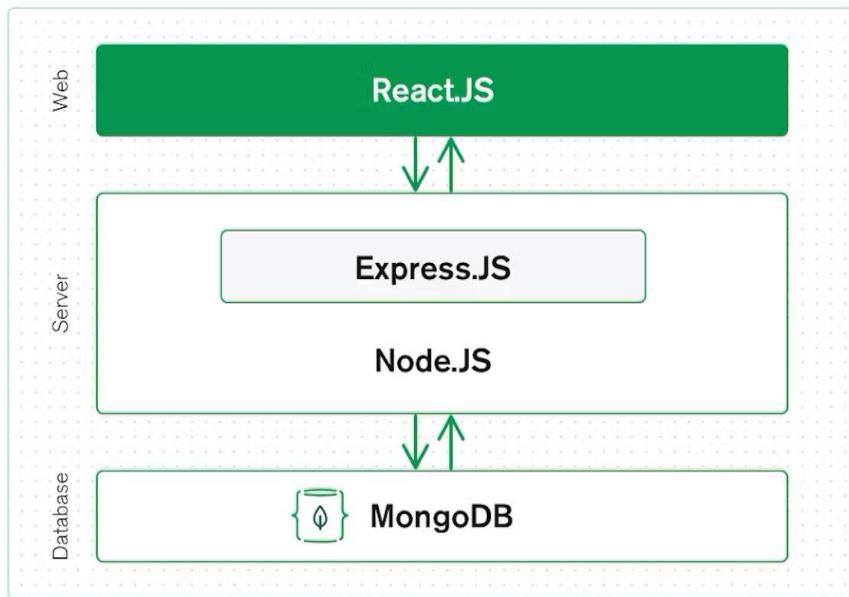
There are two main types of web development: front-end development, back-end development. Front-end development involves the “client-facing” side of web development. That is to say usually, front-end web development refers to the portion of the site, app, or digital product that users will see and interact with. A Back-End Developer creates the basic framework of a website before maintaining it and ensuring it performs the way it should, including database interactions, user authentication, server, network and hosting configuration, and business logic.

### **3.2 MERN Stack**

The MERN stack is a web development framework made up of the stack of MongoDB, Express.js, React.js, and Nodejs. It is one of the several variants of the MEAN stack. MERN Stack is a JavaScript Stack that is used for easier and faster deployment of full-stack web applications. MERN Stack comprises of 4 technologies namely: MongoDB,

Express, React and Node.js. It is designed to make the development process smoother and easier.

- MongoDB: Non-Relational Database
- Express: Node.js web server
- React: JavaScript Frontend Framework
- Node: JavaScript Web Server



**Fig. 3.1** Architecture of MERN Stack

When you use the MERN stack, you work with React to implement the presentation layer, Express and Node to make up the middle or application layer, and MongoDB to create the database layer. We will utilize these four technologies to develop a basic application.

### 3.2.1 MongoDB: Cross-platform Document-Oriented Database

MongoDB is a NoSQL database where each record is a document comprising of key-value pairs that are similar to JSON (JavaScript Object Notation) objects. MongoDB is flexible and allows its users to create schema, databases, tables, etc. Documents that are identifiable by a primary key make up the basic unit of MongoDB. Once MongoDB is installed, users can make use of Mongo shell as well. Mongo shell

provides a JavaScript interface through which the users can interact and carry out operations (Eg: querying, updating records, deleting records).

Reasons to use MongoDB:

- Fast – Being a document-oriented database, easy to index documents.  
Therefore, a faster response.
- Scalability – Large data can be handled by dividing it into several machines.
- Use of JavaScript – MongoDB uses JavaScript which is the biggest advantage.
- Schema Less – Any type of data in a separate document.
- Data stored in the form of JSON –
- Objects, Object Members, Arrays, Values, and Strings
- JSON syntax is very easy to use.
- JSON has a wide range of browser compatibility.
- Sharing Data: Data of any size and type (video, audio) can be shared easily.
- Simple Environment Setup – It's really simple to set up MongoDB.
- Flexible Document Model – MongoDB supports document-model (tables, schemas, columns & SQL) which is faster and easier.

### **3.2.2 Express: Back-End Framework**

Express is a Node.js framework. Rather than writing the code using Node.js and creating loads of Node modules, Express makes it simpler and easier to write the back-end code. Express helps in designing great web applications and APIs. Express supports many middlewares which makes the code shorter and easier to write.

Reasons to use Express:

- Asynchronous and Single-threaded.
- Efficient, fast & scalable
- Has the biggest community for Node.js
- Express promotes code reusability with its built-in router.
- Robust API

### **3.2.3 React: Front-End Library**

React is a JavaScript library that is used for building user interfaces. React is used for the development of single-page applications and mobile applications because of its

ability to handle rapidly changing data. React allows users to code in JavaScript and create UI components.

Reasons to use React:

- Virtual DOM – A virtual DOM object is a representation of a DOM object. Virtual DOM is actually a copy of the original DOM. Any modification in the web application causes the entire UI to re-render the virtual DOM. Then the difference between the original DOM and this virtual DOM is compared and the changes are made accordingly to the original DOM.
- JSX – Stands for JavaScript XML. It is an HTML/XML JavaScript Extension which is used in React. Makes it easier and simpler to write React components.
- Components – ReactJS supports Components. Components are the building blocks of UI wherein each component has a logic and contributes to the overall UI. These components also promote code reusability and make the overall web application easier to understand.
- High Performance – Features like Virtual DOM, JSX and Components makes it much faster than the rest of the frameworks out there.
- Developing Android/Ios Apps – With React Native you can easily code Android-based or IOS-Based apps with just the knowledge of JavaScript and ReactJS.

### **3.2.4 Node.js: JS Runtime Environment**

Node.js provides a JavaScript Environment which allows the user to run their code on the server (outside the browser). Node pack manager i.e. npm allows the user to choose from thousands of free packages (node modules) to download.

Reason to use Node.JS:

- Open-source JavaScript Runtime Environment
- Single threading – Follows a single-threaded model.
- Data Streaming
- Fast – Built on Google Chrome's JavaScript Engine, Node.js has a fast code execution.
- Highly Scalable

### **3.3 Methodology**

#### **3.3.1. Project Inception**

- Conduct an extensive market analysis to identify the need for a DSA preparation module.
- Define project goals, objectives, and success criteria based on user requirements.
- Formulate a detailed project charter outlining scope, stakeholders, and deliverables.

#### **3.3.2. Requirement Elicitation and Analysis**

- Engage with stakeholders, including potential users and subject matter experts, to gather comprehensive requirements.
- Prioritize requirements based on their impact and relevance to the project goals.
- Develop user personas and use cases to guide the design process.

#### **3.3.3. System Architecture Design**

- Define the overall system architecture, including backend and frontend components.
- Choose appropriate technologies and frameworks based on scalability, security, and performance requirements.
- Design database schema and relationships to efficiently store and retrieve data.

#### **3.3.4. Content Development**

- Collaborate with subject matter experts to curate high-quality content covering the entire DSA syllabus.
- Ensure content is concise, clear, and aligned with the learning goals.

#### **3.3.5. Quiz and Question Design**

- Develop a systematic approach to create topic-wise and level-wise quizzes.
- Design challenging programming questions to assess and enhance problem-solving skills.

#### **3.3.6. Performance Tracking System**

- Implement a robust system for tracking user performance.

- Generate graphical reports to provide users with detailed insights into their progress.

### **3.3.7. Community Interaction Features**

- Integrate a user-friendly Q&A platform to facilitate collaborative learning.
- Encourage user participation in asking and answering questions to build a supportive community.

### **3.3.8. Learning Resources**

- Provide concise study materials for each DSA topic to supplement learning.
- Develop cheatsheets for quick revision of key concepts.

### **3.3.9. Learning Roadmap Development**

- Create a structured learning roadmap to guide users through the DSA syllabus.
- Break down the syllabus into milestones for effective learning progression.

### **3.3.10. User Authentication and Accessibility**

- Allow users to access content without login for convenience.
- Implement a secure login system for quizzes, coding questions, discussions, and dashboard interaction.

### **3.3.11. Admin Control Panel**

- Design an admin interface for content management, ensuring admins can update quizzes, manage coding questions, and monitor platform activity.
- Design a comprehensive admin control panel for content management, user analytics, and system configuration.
- Ensure real-time monitoring of platform performance and user engagement metrics.

### **3.3.12. Code Editor Integration**

- Integrate a versatile code editor that support multiple programming languages and frameworks to accommodate diverse user preferences.
- Ensure seamless integration with learning resources and quizzes.

- Ensure a seamless coding experience for users with features like syntax highlighting and code completion.

### **3.3.13. Notification System**

- Develop a notification system to keep users informed about updates, activities, and upcoming events.
- Implement reminders for quizzes and other important milestones.

### **3.3.14. Testing and Quality Assurance:**

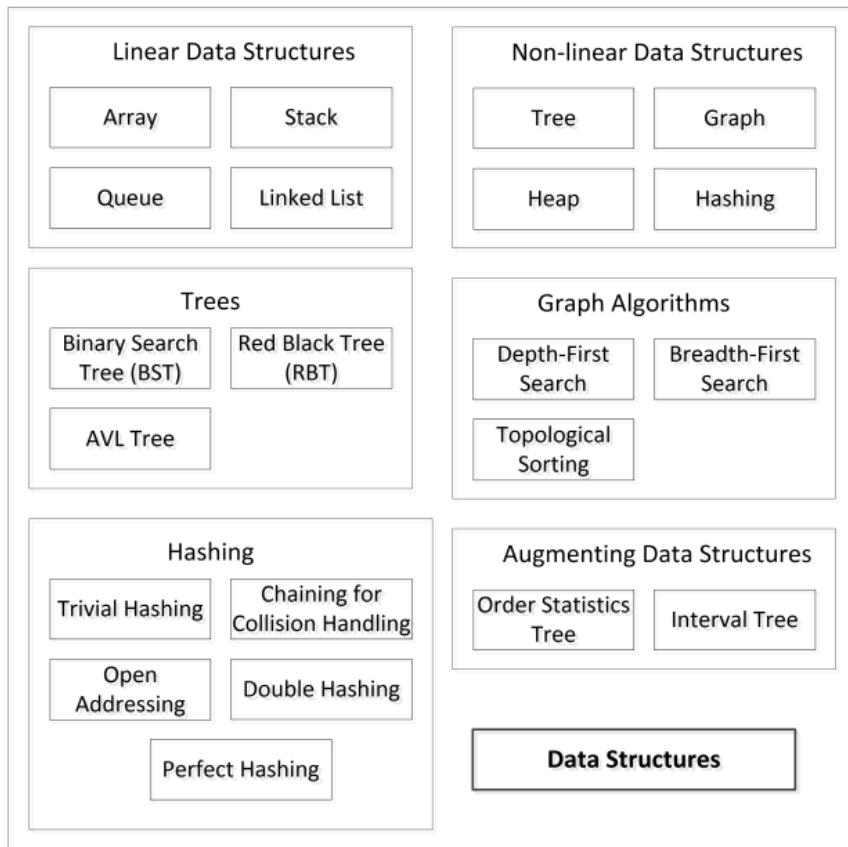
- Conduct rigorous testing to identify and address any bugs or usability issues.
- Ensure the platform is user-friendly, responsive, and functions seamlessly across devices.

## **3.4 Data Structures and Algorithms**

Data structures and Algorithms are really important courses for every concept used in computer science and software engineering. There are many concepts involved in the subjects of Algorithms and Data structures. There will be many questions for educators like how to learn Algorithms and Data structures, as there are many concepts involved without getting confused during attending the course. Data Structures is a course that combines theory with practice. Students are not only required to possess a solid foundation of higher mathematics and data structures, but also a strong ability for abstract thinking and computer programming.

Data structures topics:

- Linear data structures – array, stack, queue, linked list;
- Non-linear data structures – hashing (direct hash tables, collisions, open addressing, double hashing, perfect hashing), trees (BST, RBT, AVL), graphs, heaps;
- Augmented data structures.



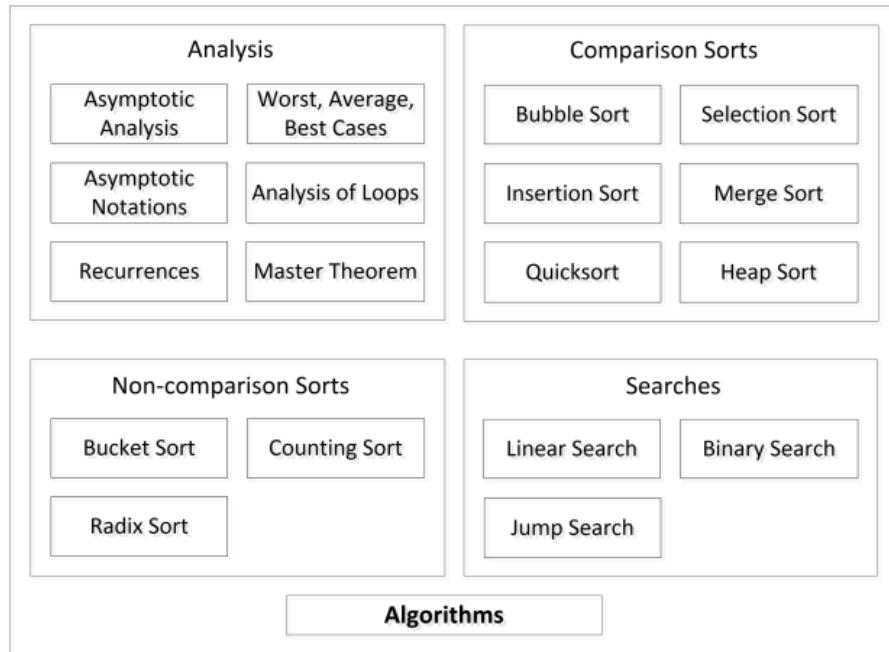
**Fig. 3.2** Data structures

Algorithms and Data structures are used to design an efficient algorithm, organize data access, and economize the execution time of programs. Data structures are widely used in operating systems, database management systems, statistical soft-ware, compiler design, numerical analysis, and artificial intelligence simulations. Algorithms and Data structures play a significant role in the computer science and soft-ware engineering domain. The course of algorithms and data structure introduces the basic data type, such as byte and character string, and presents collective data types, linked lists, stacks, arrays, queues, trees, heaps, and priority queues and complex data structures and, and focuses on programming methodologies, problem-solving strategies, and algorithm development.

Algorithms includes such topics as:

- Algorithms analysis technics – asymptotic analysis and notations, Big-Oh, complexities of algorithms, worst, average, best cases of algorithm performing, recurrences, master theorem for solving some types of recurrence equations;

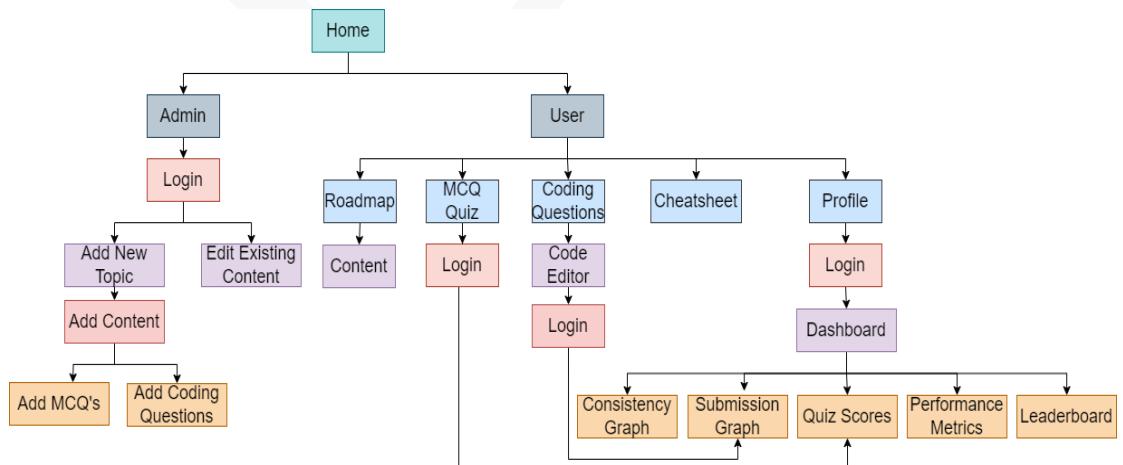
- Sorting algorithms with comparisons – bubble sort, selection sort, insertion sort, merge sort, heap sort, quicksort;
- Sorting algorithms without comparisons (in linear-time) – bucket sort, counting sort, radix sort;
- Searcher algorithms – linear search, binary search, jump search.



**Fig. 3.3** Algorithms

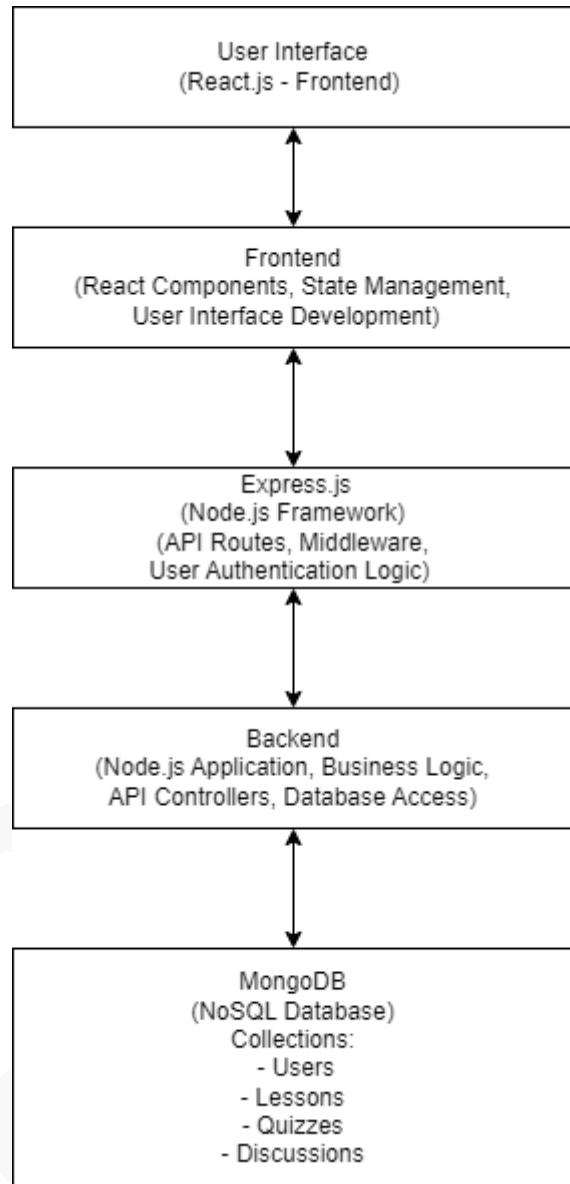
## 3.5 Methods/Algorithm

### 3.5.1 System Flow Chart



**Fig. 3.4** Flow Chart for Placement Preparation Module Focusing on DSA

### 3.6 System Design



**Fig. 3.5** System Architecture for Placement Preparation Module Focusing on DSA

### 3.7 Software Requirement

#### 3.7.1 VS Code

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust, and Julia. It is based on the Electron framework, which is used to develop Node.js web applications that run on the Blink layout engine. Visual Studio Code

employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called "Visual Studio Online" and "Visual Studio Team Services").

Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace.

Visual Studio Code can be extended via extensions available through a central repository. This includes additions to the editor and language support. A notable feature is the ability to create extensions that add support for new languages, themes, debuggers, time travel debuggers, perform static code analysis, and add code linters using the Language Server Protocol.

Source control is a built-in feature of Visual Studio Code. It has a dedicated tab inside of the menu bar where users can access version control settings and view changes made to the current project. To use the feature, Visual Studio Code must be linked to any supported version control system (Git, Apache Subversion, Perforce, etc.). This allows users to create repositories as well as to make push and pull requests directly from the Visual Studio Code program.

### **3.7.2 Postman**

Postman is one of the most popular software testing tools which is used for API testing. With the help of this tool, developers can easily create, test, share, and document APIs. Postman is a standalone software testing API (Application Programming Interface) platform to build, test, design, modify, and document APIs. It is a simple Graphic User Interface for sending and viewing HTTP requests and responses.

While using Postman, for testing purposes, one doesn't need to write any HTTP client network code. Instead, we build test suites called collections and let Postman interact with the API. In this tool, nearly any functionality that any developer may need is embedded. This tool has the ability to make various types of HTTP requests like GET,

POST, PUT, PATCH, and convert the API to code for languages like JavaScript and Python.

Postman is based on a wide range of extremely user-friendly power tools. For more than 8 million users, Postman has become a tool of convenience. Following are the reasons why Postman is used:

- Accessibility- One can use it anywhere after installing Postman into the device by simply logging in to the account.
- Use Collections-Postman allows users to build collections for their API-calls. Every set can create multiple requests and subfolders. It will help to organize the test suites.
- Test development- To test checkpoints, verification of successful HTTP response status shall be added to every API- calls.
- Automation Testing-Tests can be performed in several repetitions or iterations by using the Collection Runner or Newman, which saves time for repeated tests.
- Creating Environments- The design of multiple environments results in less replication of tests as one can use the same collection but for a different setting.
- Debugging- To effectively debug the tests, the postman console helps to track what data is being retrieved.
- Collaboration- You can import or export collections and environments to enhance the sharing of files. You may also use a direct connection to share the collections.
- Continuous integration-It can support continuous integration.

### **3.7.3 Web Browser (Chrome, Firefox, Internet Explorer, Safari)**

A web browser is an application for accessing websites and the Internet.[1] When a user requests a web page from a particular website, the browser retrieves its files from a web server and then displays the page on the user's screen. Google Chrome is a cross-platform web browser developed by Google. It was first released in 2008 for Microsoft Windows, built with free software components from Apple WebKit and Mozilla Firefox. Versions were later released for Linux, macOS, iOS, and also for Android, where it is the default browser. The browser is also the main component of ChromeOS, where it serves as the platform for web applications.

### **3.7.4 APIs**

#### **ChatGPT**

We have used this API for chatbot feature in our project. OpenAI's text generation models (often referred to as generative pre-trained transformers or "GPT" models for short), like GPT-4 and GPT-3.5, have been trained to understand natural and formal language. Models like GPT-4 allows text outputs in response to their inputs. Assistants refer to entities, which in the case of the OpenAI API are powered by large language models like GPT-4, that are capable of performing tasks for users. These assistants operate based on the instructions embedded within the context window of the model.

#### **Judge0**

Judge0 is a robust, scalable, and open-source online code execution system. You can use it to build a wide range of applications that need online code execution features.

- Rich and verbose API documentation
- Scalable architecture
- Sandboxed compilation and execution
- Support for 60+ languages
- Compilation and execution of multi-file programs
- Support for additional files alongside the user's program
- Support for custom user-defined compiler options, command-line arguments, and time and memory limits
- Detailed execution results
- Webhooks (HTTP callbacks)

## **3.8 Functional Requirements**

Functional requirements for an air quality monitoring system using IoT include the following:

### **3.9.1. User Authentication**

- Users can create accounts with unique usernames and passwords.

- The system securely stores user credentials.
- Users can log in using their credentials to access personalized features.

### **3.9.1. Learning Resources**

- The platform provides structured lessons on various Data Structures and Algorithms (DSA) concepts.
- Learning resources include textual content, visual aids, and examples.
- Questions are categorized based on difficulty levels (easy, medium, hard) and topics (arrays, linked lists, sorting algorithms, etc.).

### **3.9.1. Quizzes and Practice**

- Users can access quizzes covering different DSA topics and difficulty levels.
- Quizzes are interactive with multiple-choice questions and coding challenges.
- A coding practice section allows users to solve algorithmic problems.

### **3.9.1. Discussion Forum**

- The platform features a Q&A section where users can ask and answer questions.
- Discussion boards are organized around DSA topics for community engagement.

### **3.9.1. Roadmap**

- A personalized learning roadmap is designed for each user.
- The roadmap tracks completed lessons, quiz scores, and coding practice achievements.

### **3.9.1. Content Access**

- Users can browse and view website content, including lessons and non-interactive resources without logging in.
- Login is required for interactive activities such as quizzes, coding practice, and discussions.

### **3.9.1. Cheatsheet**

- Cheatsheets are provided for summarizing key time and space complexities for common algorithms and data structures.

### **3.9.1. Profile Management**

- Users can create, edit, and manage their profiles.
- Profile information includes personal details, academic history, and areas of interest.

### **3.9.1. Visual Progress Tracking**

- Visual representations, such as charts or graphs, depict users' performance and progress over time.
- Progress is visually displayed for completed lessons, quiz scores, and coding practice achievements.

### **3.9.1. Code Editor**

- The platform integrates a code editor that supports multiple programming languages.
- Users can write, test, and submit code directly within the platform.

### **3.9.1. Accessibility**

- The user interface is designed to be accessible and user-friendly for users with diverse needs.
- Navigation elements are intuitive, and the platform adheres to accessibility standards.

### **3.9.1. Feedback Mechanism**

- Users can provide feedback on lessons, quizzes, and overall platform experience.
- A mechanism for reporting issues or suggesting improvements is implemented.

### **3.9.1. Responsive Design**

- The platform is responsive, ensuring optimal user experience across various devices and screen sizes.

### **3.9.1. Security Measures**

- Secure protocols, such as encryption, are implemented to protect user data and privacy.
- Measures are in place to prevent unauthorized access and ensure data integrity.

### **3.9.1. Chatbot**

- The platform includes a chatbot feature to assist users with queries related to DSA concepts, quizzes, and platform navigation.
- The chatbot is accessible through a designated interface, providing users with a conversational way to seek information.

## **3.9 Non-Functional Requirements**

### **3.10.1. Performance**

- Response Time: The system should respond to user interactions within 2 seconds for all major functionalities.
- Scalability: The platform should handle a simultaneous user load of at least 1000 users without significant degradation in performance.

### **3.10.1. Reliability**

- The system should have an uptime of at least 99.5% for continuous availability.
- In the event of a server failure, the system should recover within minutes.

### **3.10.1. Availability:**

- The platform should be accessible 24/7, with scheduled maintenance communicated in advance.

### **3.10.1. Scalability**

- The system should scale horizontally to accommodate an increasing number of users and data.

### **3.10.1. Security**

- User data, including personal information and performance metrics, should be encrypted during transmission and storage.
- The platform should implement secure user authentication and authorization mechanisms.
- Regular security audits and vulnerability assessments should be conducted.

### **3.10.1. Usability**

- The user interface should be intuitive and require minimal training for new users.

### **3.10.1. Maintainability**

- Codebase should be well-documented for ease of maintenance and future enhancements.
- System updates and patches should be deployable without significant downtime.

### **3.10.1. Compatibility**

- The platform should be compatible with major web browsers, including Chrome, Firefox, Safari, and Edge.
- Mobile responsiveness should be maintained across various devices and operating systems.

### **3.10.1. Portability**

- The system should be deployable on cloud platforms like AWS, Azure, or Google Cloud.
- Code should be modular and portable for easy migration between different environments.

### **3.10.1. Interoperability**

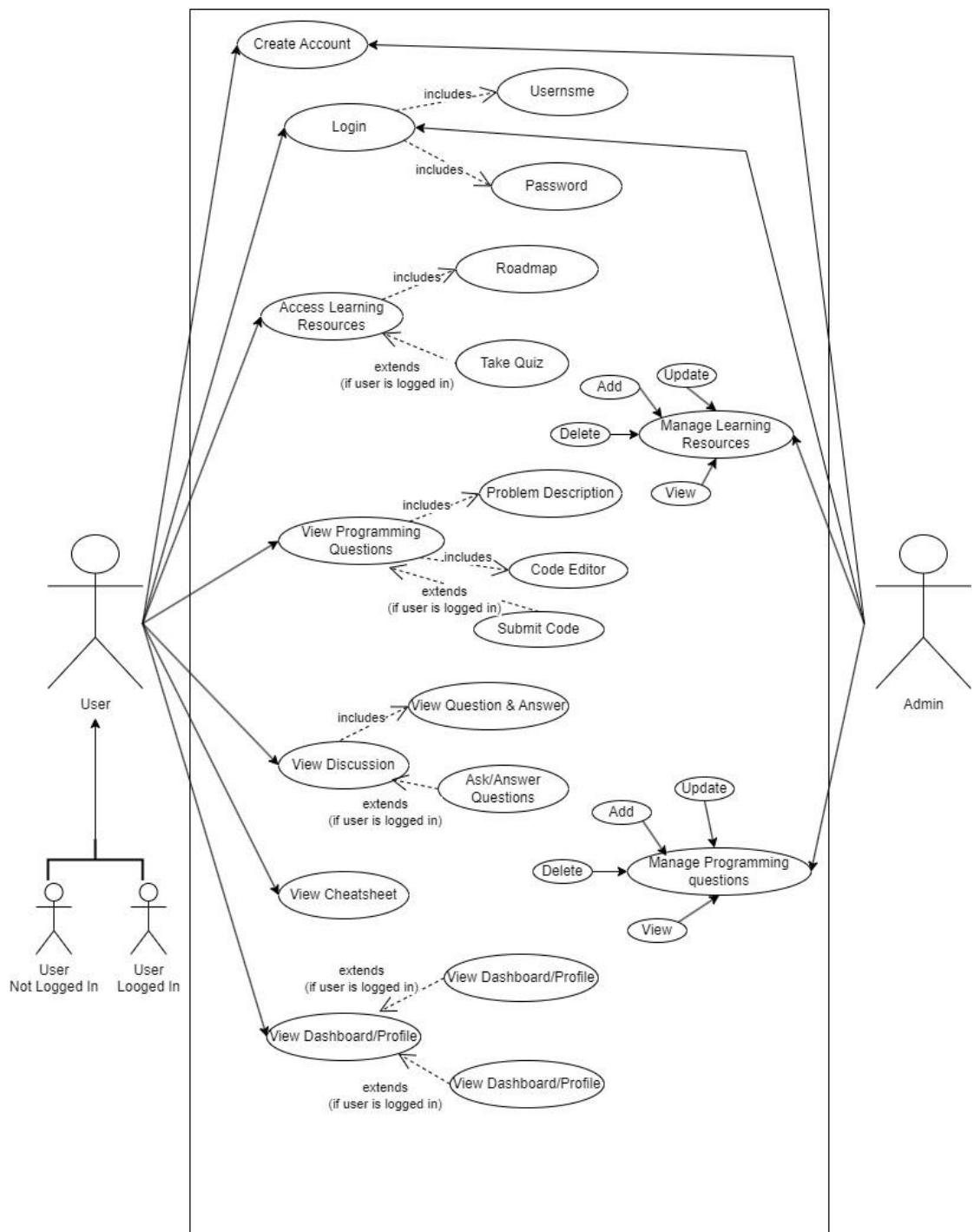
- The platform should integrate seamlessly with external services such as email and SMS for notifications.
- APIs should be available for potential future integrations with other educational platforms.

### **3.10.1. Capacity**

- Database capacity should accommodate a growing volume of lessons, quizzes, and user data.

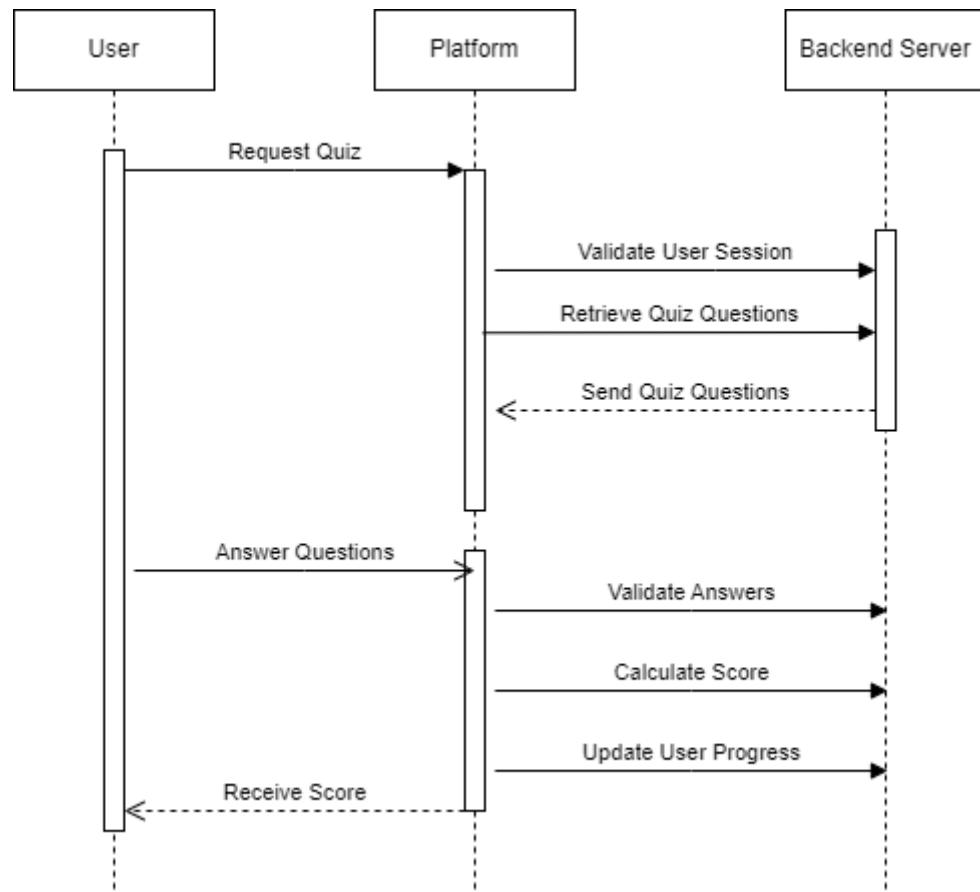
## 3.10 UML Diagram

### 3.10.2 Use Case Diagram



**Fig. 3.6** Use Case Diagram for Placement Preparation Module Focusing on DSA

### 3.10.1 Sequence Diagram



**Fig. 3.7** Sequence Diagram for Placement Preparation Module Focusing on DSA.

## **CHAPTER 4**

### **IMPLEMENTATION OF PROJECT**

---

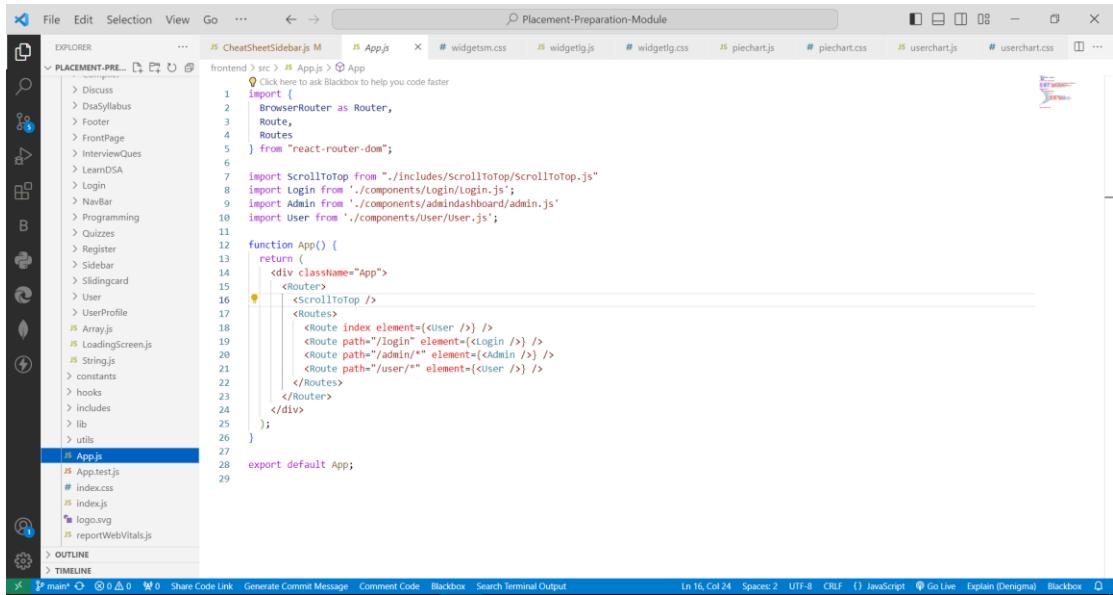
This chapter contains implementation of project. In this project we have implemented a Placement Preparation Module for DSA. web application engineered on the MERN stack—MongoDB, Express.js, React.js, and Node.js. We set up the development environment with Node.js, MongoDB, and a code editor. Initialize a React.js project for the frontend and a Node.js/Express.js project for the backend.

Implement user registration and login functionalities. Created React components for displaying lessons on DSA concepts. Develop interactive quizzes with React components, handling real-time user responses. Implemented a coding practice section with a custom code editor supporting multiple programming languages using API Judge0 CE that has logic to execute and evaluate user-submitted code. Created API endpoints for posting and fetching discussion threads and Implemented React components for rendering discussion threads and facilitating user interactions.

Designed React components to visualize the user's learning roadmap. Implemented visual progress tracking using charts or graphs based on user performance metrics. Integrate a chatbot using ChatGPT API. We made responsive design using CSS frameworks and tested the platform on various devices to ensure compatibility.

Designed the MongoDB database schema to accommodate user data, lessons, quizzes, and study material. Established relationships between collections for efficient data retrieval. Conducted unit testing for individual components and performed end-to-end testing to ensure the seamless integration of features.

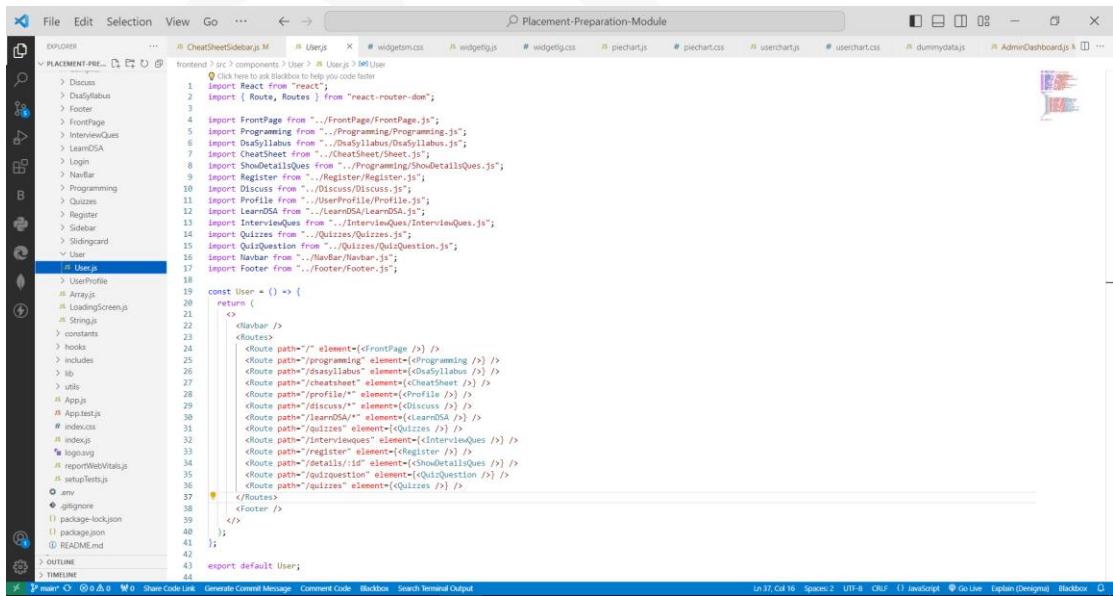
## 4.1 App.js



```
File Edit Selection View Go ... < > Placement-Preparation-Module
EXPLORER frontend > src > App.js
CheatSheetSidebar.js M App.js # widgetsm.css # widgetlg.js # widgetlg.css # piechart.js # piechart.css # userchart.js # userchart.css ...
frontend > src > components > User > User.js
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";
import ScrollToTop from './includes/ScrollToTop/ScrollToTop.js';
import Login from './components/login/Login.js';
import Admin from './components/adminDashboard/admin.js';
import User from './components/User/User.js';
function App() {
  return (
    <div className="App">
      <Router>
        <ScrollToTop />
        <Routes>
          <Route index element={<User />} />
          <Route path="/login" element={<Login />} />
          <Route path="/admin/*" element={<Admin />} />
          <Route path="/user/*" element={<User />} />
        </Routes>
      </Router>
    </div>
  );
}
export default App;
```

Fig. 4.1 App.js file of Placement Preparation Module Focusing on DSA.

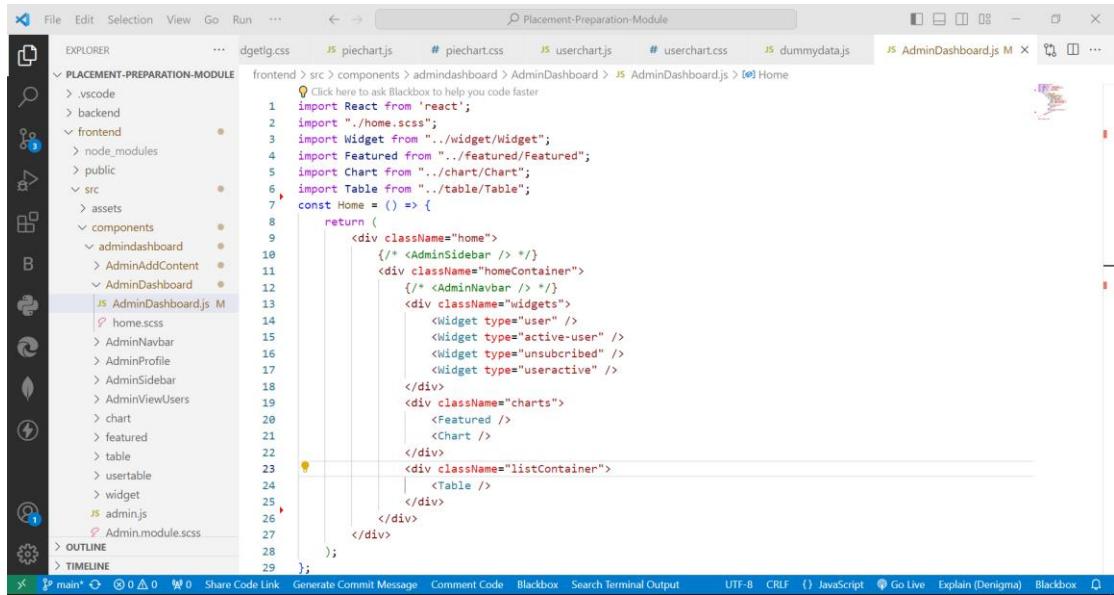
## 4.2 User.js



```
File Edit Selection View Go ... < > Placement-Preparation-Module
EXPLORER frontend > src > components > User > User.js
CheatSheetSidebar.js M User.js # widgetsm.css # widgetlg.js # widgetlg.css # piechart.js # piechart.css # userchart.js # userchart.css ...
frontend > src > components > User > User.js
import { Route, Routes } from "react-router-dom";
import FrontPage from "./FrontPage/FrontPage.js";
import Programming from "./Programming/Programming.js";
import DisSyllabus from "./DisSyllabus/DisSyllabus.js";
import CheatSheet from "./CheatSheet/CheatSheet.js";
import ShowDetailsQues from "./Programming>ShowDetailsQues.js";
import Register from "./Register/Register.js";
import Discuss from "./Discuss/Discuss.js";
import DisSyllabus from "./DisSyllabus/DisSyllabus.js";
import LearnDSA from "./LearnDSA/LearnDSA.js";
import InterviewQues from "./InterviewQues/InterviewQues.js";
import Quizzes from "./Quizzes/Quizzes.js";
import QuizQuestion from "./Quizzes/QuizQuestion.js";
import NavBar from "./NavBar/NavBar.js";
import Footer from "./Footer/Footer.js";
const User = () => {
  return (
    <div>
      <UserBar />
      <Routes>
        <Route path="/" element={<FrontPage />} />
        <Route path="/programming" element={<Programming />} />
        <Route path="/disyllabus" element={<DisSyllabus />} />
        <Route path="/details" element={<ShowDetailsQues />} />
        <Route path="/profile" element={<Profile />} />
        <Route path="/discuss" element={<Discuss />} />
        <Route path="/learndsa" element={<LearnDSA />} />
        <Route path="/quizzes" element={<Quizzes />} />
        <Route path="/details/:id" element={<ShowDetailsQues />} />
        <Route path="/quizquestion" element={<QuizQuestion />} />
        <Route path="/quizzes" element={<Quizzes />} />
      </Routes>
    </div>
  );
}
export default User;
```

Fig. 4.2 User.js file of Placement Preparation Module Focusing on DSA.

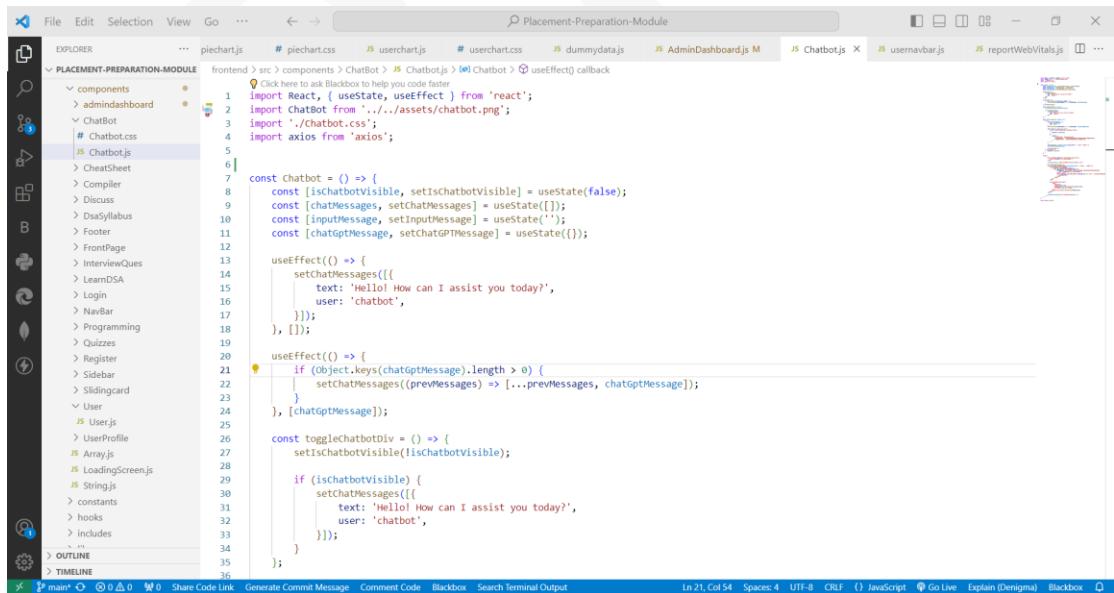
## 4.3 Admin Dashboard



```
Click here to ask Blackbox to help you code faster
1 import React from 'react';
2 import './home.scss';
3 import Widget from '../widget/widget';
4 import Featured from '../featured/Featured';
5 import Chart from '../chart/Chart';
6 import Table from '../table/Table';
7 const Home = () => {
8   return (
9     <div className="home">
10       <AdminSidebar />
11       <div className="homeContainer">
12         <AdminNavbar />
13         <div className="widgets">
14           <Widget type="user" />
15           <Widget type="active-user" />
16           <Widget type="unsubscribed" />
17           <Widget type="useractive" />
18         </div>
19         <div className="charts">
20           <Featured />
21           <Chart />
22         </div>
23         <div className="listContainer">
24           <Table />
25         </div>
26       </div>
27     );
28   };
29 };
```

Fig. 4.3 AdminDashboard.js file of Placement Preparation Module Focusing on DSA.

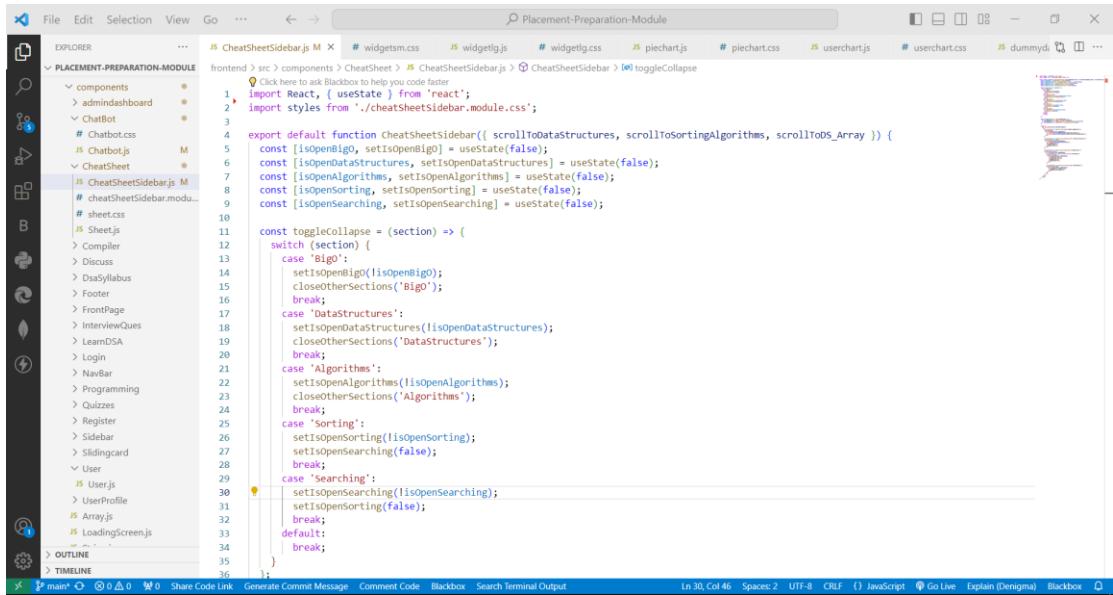
## 4.4 Chatbot



```
Click here to ask Blackbox to help you code faster
1 import React, { useState, useEffect } from 'react';
2 import Chatbot from './assets/chatbot.png';
3 import './Chatbot.css';
4 import axios from 'axios';
5
6 const Chatbot = () => {
7   const [isChatbotVisible, setIsChatbotVisible] = useState(false);
8   const [chatMessages, setChatMessages] = useState([]);
9   const [inputMessage, setInputMessage] = useState('');
10  const [chatGptMessage, setChatGptMessage] = useState({});
11
12  useEffect(() => {
13    setChatMessages([
14      {
15        text: 'Hello! How can I assist you today?',
16        user: 'chatbot',
17      },
18    ]);
19  }, []);
20
21  useEffect(() => {
22    if (Object.keys(chatGptMessage).length > 0) {
23      setChatMessages([...prevMessages, chatGptMessage]);
24    }
25  }, [chatGptMessage]);
26
27  const toggleChatbotDiv = () => {
28    setIsChatbotVisible(!isChatbotVisible);
29
30    if (!isChatbotVisible) {
31      setChatMessages([
32        {
33          text: 'Hello! How can I assist you today?',
34          user: 'chatbot',
35        },
36      ]);
37    }
38  };
39
40  const handleInput = (e) => {
41    setInputMessage(e.target.value);
42  };
43
44  const handleSubmit = (e) => {
45    e.preventDefault();
46    const messageData = {
47      text: inputMessage,
48      user: 'user',
49    };
50
51    axios
52      .post('https://api.chatgpt.com/generate', messageData)
53      .then((response) => {
54        const gptMessage = response.data.message;
55
56        setChatGptMessage(gptMessage);
57        setInputMessage('');
58      })
59      .catch((error) => {
60        console.error(error);
61      });
62  };
63
64  return (
65    <div>
66      <div>
67        <img alt="Chatbot icon" />
68      </div>
69      <div>
70        <h3>Ask me anything!</h3>
71        <form>
72          <input type="text" value={inputMessage} onChange={handleInput} />
73          <button type="submit" onClick={handleSubmit}>Ask</button>
74        </form>
75      </div>
76    </div>
77  );
78}
```

Fig. 4.4 Chatbot.js file of Placement Preparation Module Focusing on DSA.

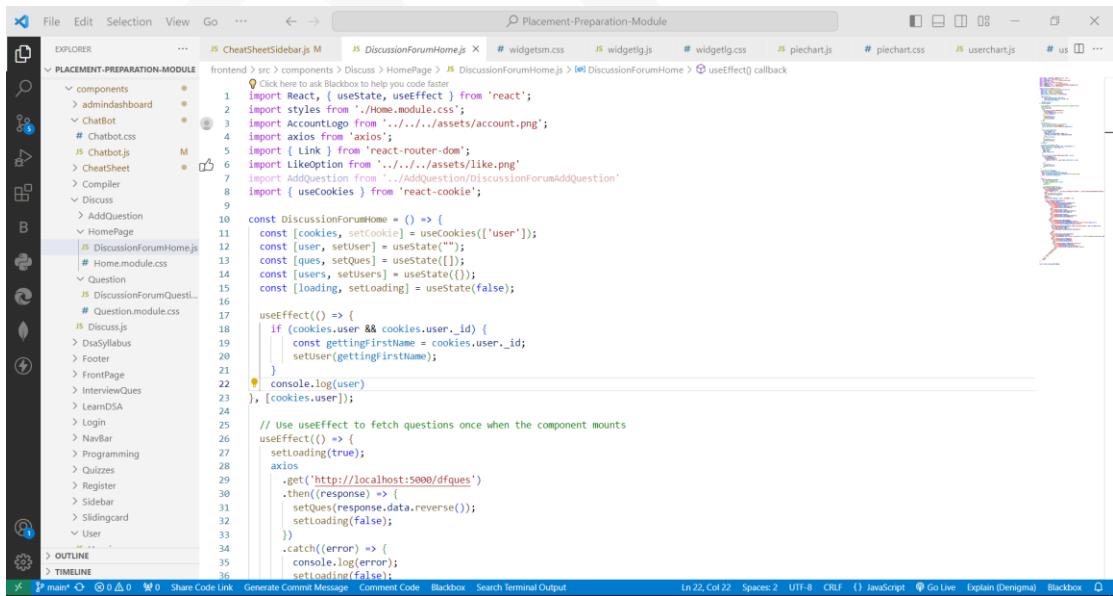
## 4.5 Cheatsheet



```
1 > import React, { useState } from 'react';
2 > import styles from './cheatSheetsSidebar.module.css';
3
4 > export default function CheatSheetsSidebar({ scrollToDataStructures, scrollToSortingAlgorithms, scrollToDS_Array }) {
5   const [isopenBigO, setIsopenBigO] = useState(false);
6   const [isOpenDataStructures, setIsopenDataStructures] = useState(false);
7   const [isOpenAlgorithms, setIsopenAlgorithms] = useState(false);
8   const [isOpenSorting, setIsopenSorting] = useState(false);
9   const [isOpenSearching, setIsopenSearching] = useState(false);
10
11   const toggleCollapse = (section) => {
12     switch (section) {
13       case 'BigO':
14         setIsopenBigO(!isopenBigO);
15         closeOtherSections('BigO');
16         break;
17       case 'DataStructures':
18         setIsopenDataStructures(!isOpenDataStructures);
19         closeOtherSections('DataStructures');
20         break;
21       case 'Algorithms':
22         setIsopenAlgorithms(!isOpenAlgorithms);
23         closeOtherSections('Algorithms');
24         break;
25       case 'Sorting':
26         setIsopenSorting(!isOpenSorting);
27         setIsopenSearching(false);
28         break;
29       case 'Searching':
30         setIsopenSearching(!isOpenSearching);
31         setIsopenSorting(false);
32         | break;
33         default:
34         | break;
35     }
36   };
37 }
```

Fig. 4.5 Cheatsheet.js file of Placement Preparation Module Focusing on DSA.

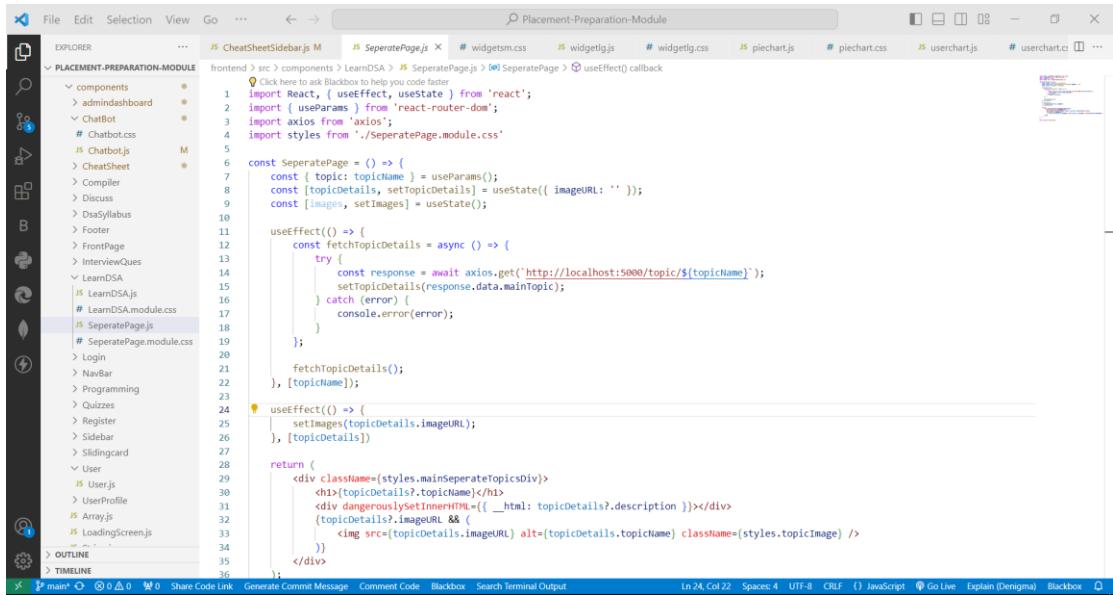
## 4.6 Discussion Forum



```
1 > import React, { useEffect } from 'react';
2 > import styles from './Home.module.css';
3 > import AccountLogo from '../../../../../assets/account.png';
4 > import axios from 'axios';
5 > import { Link } from 'react-router-dom';
6 > import LikeQues from '../../../../../assets/like.png';
7 > import AddQuestion from '../../../../../AddQuestion/DiscussionForum/AddQuestion';
8 > import { useCookies } from 'react-cookie';
9
10 const DiscussionForumHome = () => {
11   const [cookies, setCookie] = useCookies(['user']);
12   const [user, setUser] = useState("");
13   const [ques, setQues] = useState([]);
14   const [users, setUsers] = useState([]);
15   const [loading, setLoading] = useState(false);
16
17   useEffect(() => {
18     if (cookies.user && cookies.user._id) {
19       const gettingFirstName = cookies.user._id;
20       setUser(gettingFirstName);
21     }
22   }, [cookies.user]);
23
24   // Use useEffect to fetch questions once when the component mounts
25   useEffect(() => {
26     setLoading(true);
27     axios
28       .get('http://localhost:5000/dfques')
29       .then((response) => {
30         setQues(response.data.reverse());
31         setLoading(false);
32       })
33       .catch((error) => {
34         console.log(error);
35         setLoading(false);
36       });
37 }
```

Fig. 4.6 DiscussionForum.js file of Placement Preparation Module Focusing on DSA.

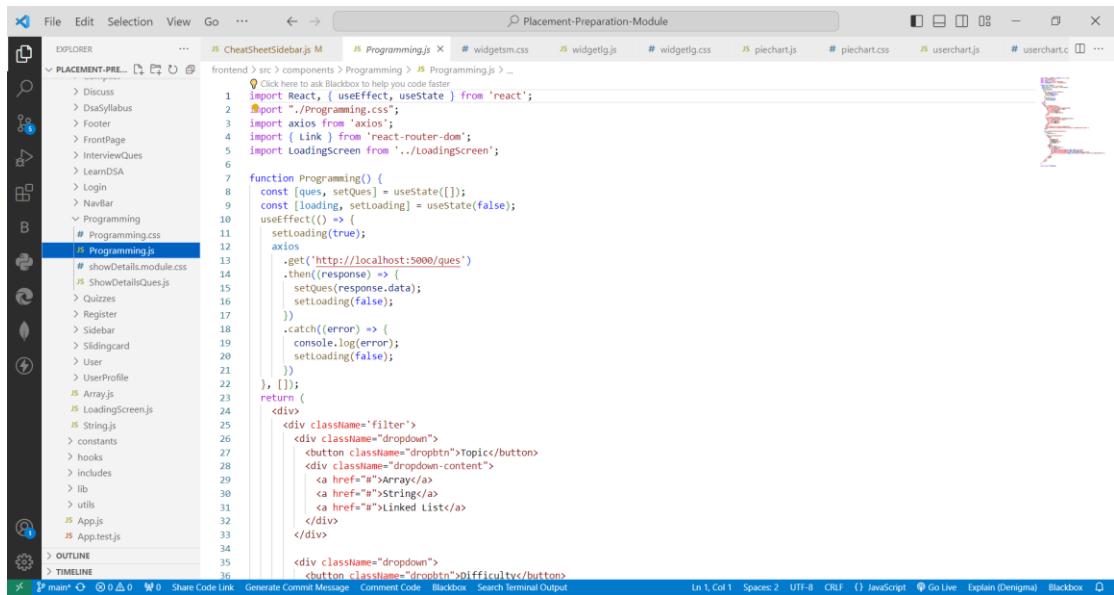
## 4.7 Learn DSA



```

File Edit Selection View Go ... < > Placement-Preparation-Module
EXPLORER JS CheatSheetSidebar.js M JS SeparatePage.js X # widgetsm.css JS widgetlg.js # widgetlg.css JS piechart.js JS piechart.css JS userchart.js # userchart.css ...
PLACEMENT-PREPARATION-MODULE frontend > ... > components > LearnDSA > JS SeparatePage.js > (0) SeparatePage > (1) useEffect() callback
1 import React, { useEffect, useState } from 'react';
2 import { useParams } from 'react-router-dom';
3 import axios from 'axios';
4 import styles from './SeparatePage.module.css';
5
6 const SeparatePage = () => {
7   const [topicName] = useParams();
8   const [topicDetails, setTopicDetails] = useState({ imageURL: '' });
9   const [images, setImages] = useState();
10
11   useEffect(() => {
12     const fetchTopicDetails = async () => {
13       try {
14         const response = await axios.get(`http://localhost:5000/topic/${topicName}`);
15         setTopicDetails(response.data.mainTopic);
16       } catch (error) {
17         console.error(error);
18       }
19     };
20
21     fetchTopicDetails();
22   }, [topicName]);
23
24   useEffect(() => {
25     setImages(topicDetails.imageURL);
26   }, [topicDetails]);
27
28   return (
29     <div className={styles.mainSeparateTopicsDiv}>
30       <h1>{topicDetails?.topicName}</h1>
31       <div dangerouslySetInnerHTML={__html: topicDetails?.description}></div>
32       {topicDetails?.imageURL && (
33         <img src={topicDetails.imageURL} alt={topicDetails.topicName} className={styles.topicImage} />
34       )}
35     </div>
36   );
37 }
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
948
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1377
1378
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1427
1428
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1437
1438
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1447
1448
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1457
1458
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1467
1468
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1477
1478
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1517
1518
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1527
1528
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1537
1538
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1547
1548
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1557
1558
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1567
1568
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1577
1578
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1617
1618
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1627
1628
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1637
1638
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1647
1648
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1657
1658
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1667
1668
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1677
1678
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1717
1718
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1727
1728
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1737
1738
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1747
1748
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1767
1768
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1777
1778
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1817
1818
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1827
1828
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1837
1838
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1847
1848
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1867
1868
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1917
1918
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1927
1928
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1937
1938
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1947
1948
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1967
1968
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1977
1978
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1987
1988
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
```

## 4.9 Programming



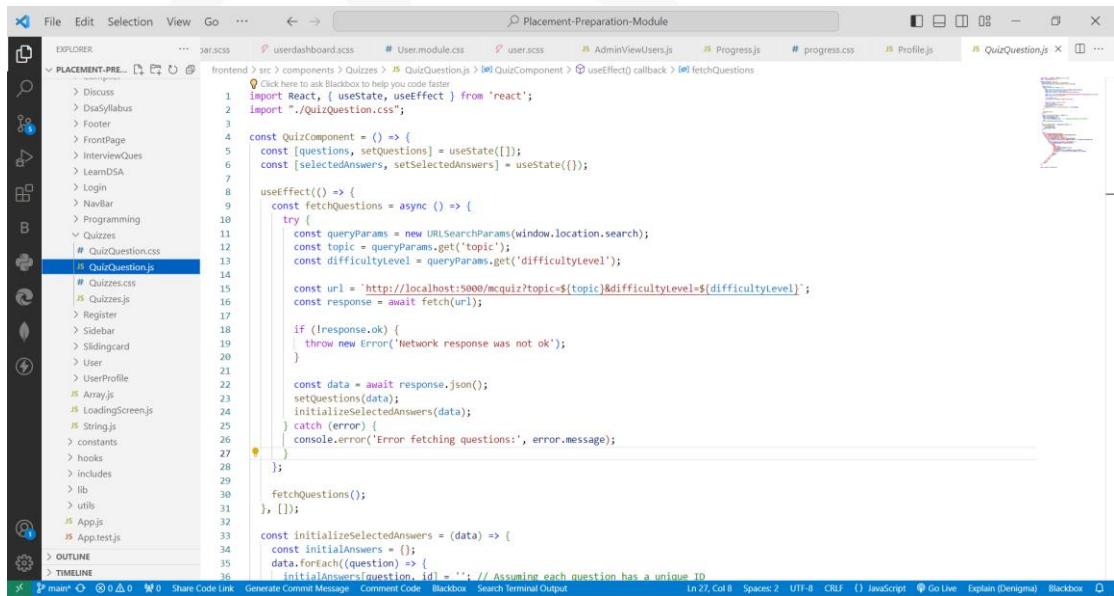
```
File Edit Selection View Go ... < > Placement-Preparation-Module
EXPLORER PLACEMENT-PREP... FRONTEND
  > Discuss
  > DisSyllabus
  > Footer
  > FrontPage
  > InterviewQues
  > LearnDSA
  > Login
  > NavBar
  > Programming
    # Programming.css
    # Programming.js
    # showDetails.module.css
    # ShowDetailsQues.js
    Quizzes
    Sidebar
    Slidingcard
    User
    UserProfile
    Array.js
    LoadingScreen.js
    String.js
    constants
    hooks
    includes
    lib
    utils
    App.js
    App.test.js
  > OUTLINE
  > TIMELINE
  > main.js Share Code Link Generate Commit Message Comment Code Blackbox Search Terminal Output
  Click here to ask Blackbox to help you code faster
  import React, { useState, useEffect } from 'react';
  import './Programming.css';
  import axios from 'axios';
  import { Link } from 'react-router-dom';
  import LoadingScreen from '../LoadingScreen';

  function Programming() {
    const [ques, setQues] = useState([]);
    const [loading, setLoading] = useState(false);
    useEffect(() => {
      setLoading(true);
      axios
        .get('http://localhost:5000/ques')
        .then((response) => {
          setQues(response.data);
          setLoading(false);
        })
        .catch((error) => {
          console.log(error);
          setLoading(false);
        })
    }, []);
    return (
      <div>
        <div className='filter'>
          <div className='dropdown'>
            <button className='dropdownbtn'>Topic</button>
            <div className='dropdown-content'>
              <a href="#">Array</a>
              <a href="#">String</a>
              <a href="#">Linked List</a>
            </div>
          </div>
        <div className='dropdown'>
          <button className='dropdownbtn'>Difficulty</button>
        </div>
      </div>
    );
  }

```

Fig. 4.9 Programming.js file of Placement Preparation Module Focusing on DSA.

## 4.10 Quiz Questions



```
File Edit Selection View Go ... < > Placement-Preparation-Module
EXPLORER PLACEMENT-PREP... FRONTEND
  > Discuss
  > DisSyllabus
  > Footer
  > FrontPage
  > InterviewQues
  > LearnDSA
  > Login
  > NavBar
  > Programming
    # Programming.css
    # Programming.js
    # QuizQuestion.css
    # QuizQuestion.js
    # Quizzes.css
    # Quizzes.js
    Register
    Sidebar
    Slidingcard
    User
    UserProfile
    Array.js
    LoadingScreen.js
    String.js
    constants
    hooks
    includes
    lib
    utils
    App.js
    App.test.js
  > OUTLINE
  > TIMELINE
  > main.js Share Code Link Generate Commit Message Comment Code Blackbox Search Terminal Output
  Click here to ask Blackbox to help you code faster
  import React, { useState, useEffect } from 'react';
  import './QuizQuestion.css';
  const QuizComponent = () => {
    const [questions, setQuestions] = useState([]);
    const [selectedAnswers, setSelectedAnswers] = useState({});
    useEffect(() => {
      const fetchQuestions = async () => {
        try {
          const queryParams = new URLSearchParams(window.location.search);
          const topic = queryParams.get('topic');
          const difficultyLevel = queryParams.get('difficultyLevel');

          const url = `http://localhost:5000/mquiz?topic=${topic}&difficultyLevel=${difficultyLevel}`;
          const response = await fetch(url);

          if (!response.ok) {
            throw new Error('Network response was not ok');
          }

          const data = await response.json();
          setQuestions(data);
          initializeSelectedAnswers(data);
        } catch (error) {
          console.error('Error fetching questions:', error.message);
        }
      };
      fetchQuestions();
    }, []);
    const initializeSelectedAnswers = (data) => {
      const initialAnswers = {};
      data.forEach((question) => {
        initialAnswers[question.id] = '';
      });
    };
  };

```

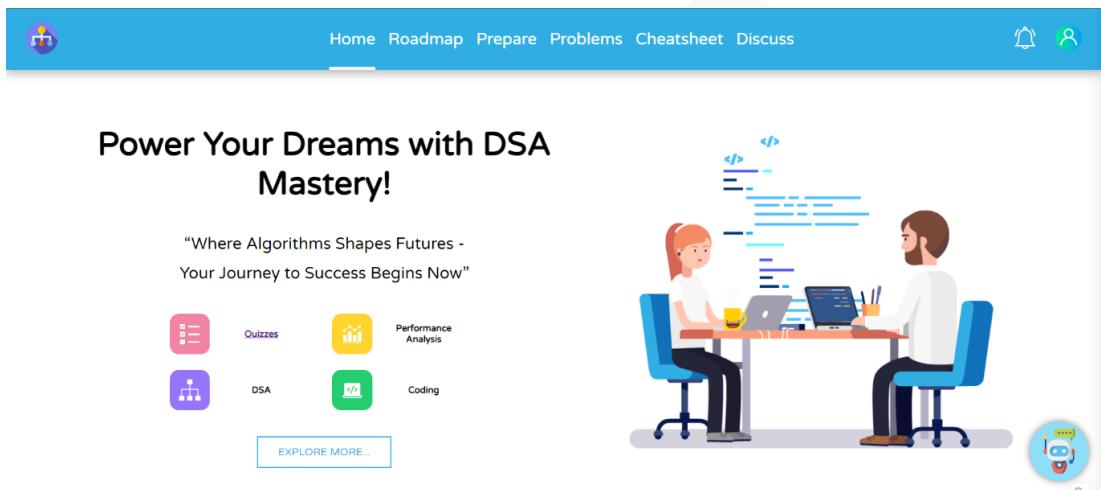
Fig. 4.10 QuizQuestions.js file of Placement Preparation Module Focusing on DSA.

# CHAPTER 5

## RESULTS AND DISCUSSIONS

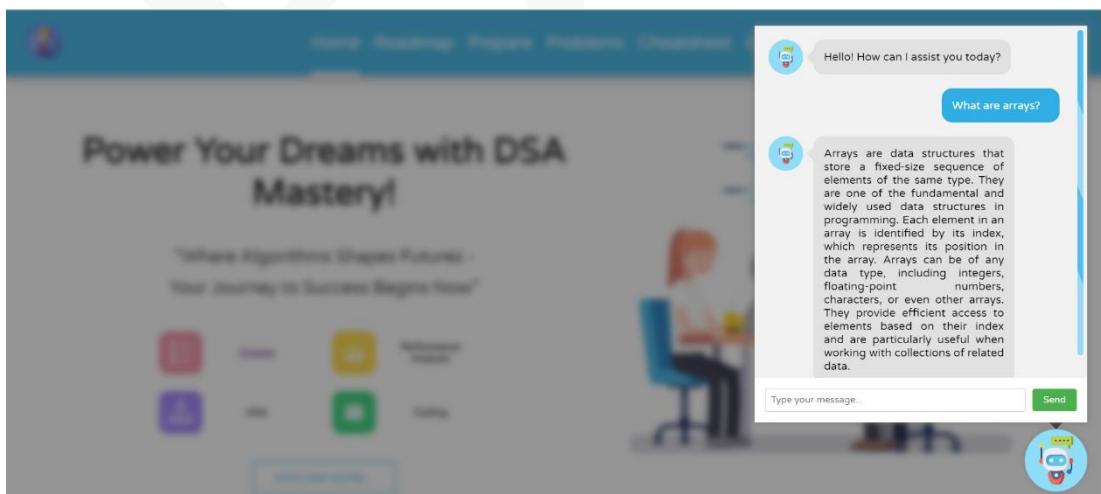
### 5.1 Home Page

The homepage interface enables user to navigate through the concept and purpose of learning DSA and learn about features of our website. There is also contact us sections with option of newsletters.



**Fig. 5.1** Home Page of Placement Preparation Module Focusing on DSA.

#### 5.1.1 Chatbot



**Fig. 5.2** Chatbot of Placement Preparation Module Focusing on DSA.

**Power Your Dreams with DSA Mastery!**

"Where Algorithms Shapes Futures - Your Journey to Success Begins Now"

Quizzes, Performance Analysis, DSA, Coding

EXPLORE MORE...

**Practice Coding**

**Learn Algorithm**

**WHY PLACEMENT PREPARATION MODULE**

- 1 Structured Learning Path**  
Roadmap covering the important topic related to data structure and algorithm
- 2 Interactive Coding Challenges**  
Real time coding competitions on the platform for students to practice
- 3 Community and Support**  
Discussion forum where students can post their doubts and chat with their peers.
- 4 Regular Updates**  
Regular updates and notification for students to practice daily.

**Data Structure** [Read More](#)

**Algorithm** [Read More](#)

**Coding Problems** [Read More](#)

**Frequently Asked Questions**

What about language support?

Do I need to be technical to use this?

What is the difficulty level of questions?

**STILL IN DOUBT?**

**Welcome**  
Let's create your account!

Enter First Name Here  
Enter Email Here  
Enter Your Query Here

submit

**INFORMATION**

- Home
- About Us
- FAQs
- Developers
- Contact Us

**FOLLOW US ON**

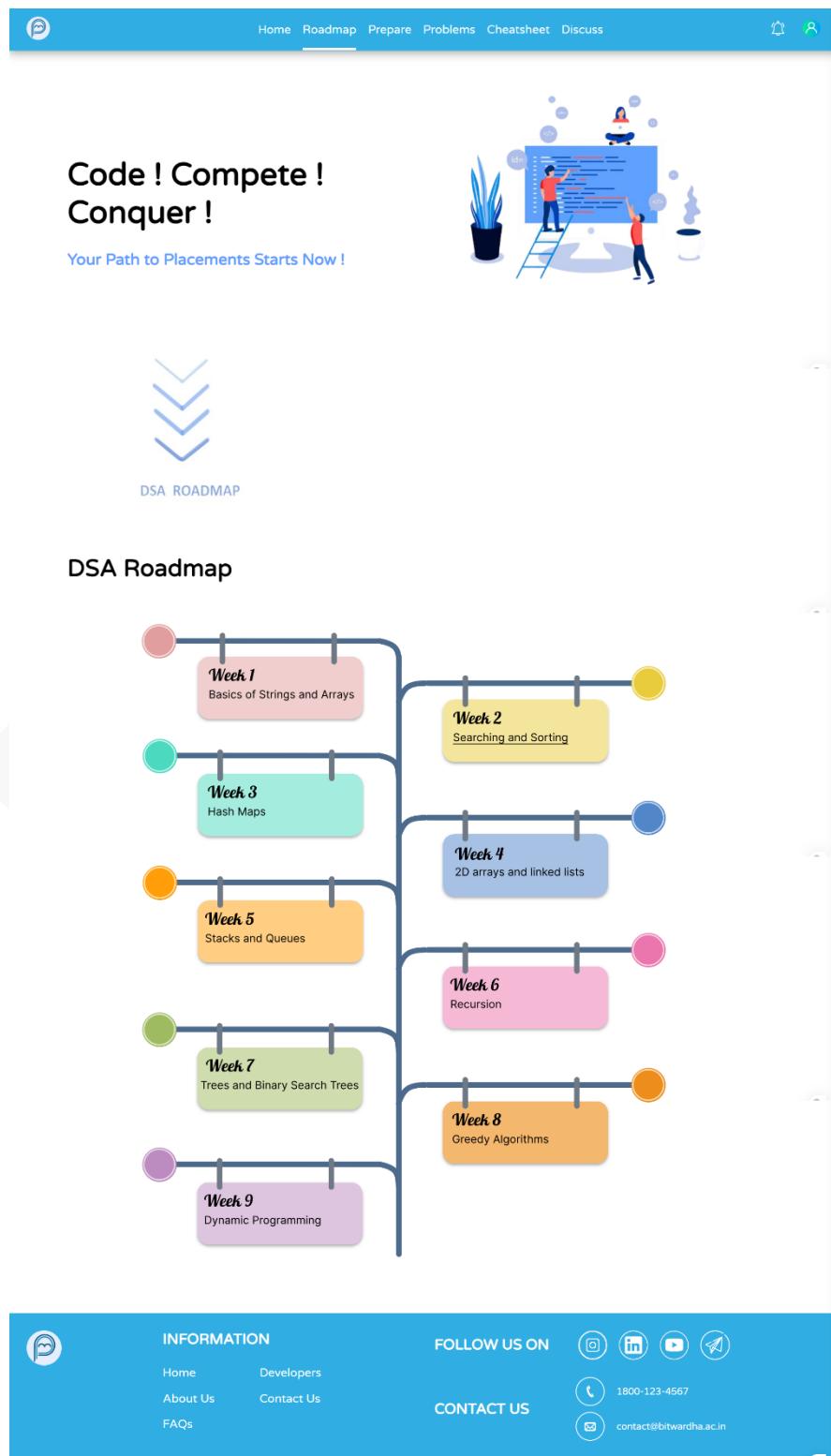
**CONTACT US**

1800-123-4567  
contact@bitwardha.ac.in

**Fig. 5.3** Home Page of Placement Preparation Module Focusing on DSA.

## 5.2 Roadmap Page

The homepage interface enables user to navigate through the concept and purpose of learning DSA and learn about features of our website. There is also contact us sections with option of newsletters.



**Fig. 5.4** Roadmap Page of Placement Preparation Module Focusing on DSA.

## 5.3 Prepare Page

There is a side navigation bar which contains topics of DSA and the page contains roadmap which displays users learning path. It contains topic wise short contents with short and precise information. Also interview questions are provided.

### 5.3.2 Quizzes

The screenshot shows a web interface for a placement preparation module. At the top, there is a blue header bar with a logo on the left, followed by navigation links: Home, Roadmap, Prepare, Problems, Cheatsheet, and Discuss. On the far right of the header are icons for a bell (notifications) and a user profile. Below the header, there is a grid of nine cards, each representing a different data structure or algorithm topic. Each card has a title, a difficulty level indicator, and a link to more details. The topics and their difficulty levels are:

Topic	Difficulty
Array	Easy Medium Hard
Linked List	Easy Medium Hard
String	Easy Medium Hard
Dynamic Programming	Easy Medium Hard
Array-List	Easy Medium Hard
Sorting Algorithm	Easy Medium Hard
Stack and Queue	Easy Medium Hard
Recursion	Easy Medium Hard
Hash Maps	Easy Medium Hard

At the bottom of the page, there is a footer section with a blue background. It includes a logo, links to 'INFORMATION' (Home, Developers, About Us, Contact Us, FAQs), social media links for 'FOLLOW US ON' (Instagram, LinkedIn, YouTube, Telegram), and contact information for 'CONTACT US' (phone number 1800-123-4567 and email contact@bitwardha.ac.in).

**Fig. 5.5** Quizzes Page of Placement Preparation Module Focusing on DSA.

The screenshot shows a quiz interface with a blue header bar containing navigation links: Home, Roadmap, Prepare, Problems, Cheatsheet, and Discuss. On the right side of the header are icons for a bell and a user profile.

### Quiz Questions

**Question 1: Which of these best describes an array?**

- A data structure that shows a hierarchical behavior
- Container of objects of similar types
- Arrays are immutable once initialised
- Array is not a data structure

**Question 2: How do you instantiate an array in Java?**

- int arr[] = new int[3];
- int arr[];
- int arr[];
- int arr() = new int[3];

**Question 3: With the help of which operator array elements can be accessed?**

- Parenthesis ()
- Braces {}
- Subscript Operator []
- None of these

**Question 4: Which of the following statements correctly declares a two-dimensional integer array in C/C++?**

- arr[5 \* 4]
- int arr[5][4];
- arr[2][2]
- All of these

**Question 5: What is the famous mathematical example of 2-d array?**

- cube
- Dice
- Matrix
- None of these

[Submit](#)

**INFORMATION**

<a href="#">Home</a>	<a href="#">Developers</a>
<a href="#">About Us</a>	<a href="#">Contact Us</a>
<a href="#">FAQs</a>	

**FOLLOW US ON**

<a href="#"></a>	<a href="#"></a>	<a href="#"></a>	<a href="#"></a>
------------------	------------------	------------------	------------------

**CONTACT US**

<a href="#"></a>	1800-123-4567
<a href="#"></a>	contact@bitwardha.ac.in

**Fig. 5.6** Quizzes Form of Placement Preparation Module Focusing on DSA.

### 5.3.1 Learn DSA

Home Roadmap Prepare Problems Cheatsheet Discuss

Syllabus Stack

Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO(Last In First Out) or FILO(First In Last Out).  
The reason why Stack is considered a complex data structure is that it uses other data structures for implementation, such as Arrays, Linked lists, etc. based on the characteristics and features of Stack data structure.

**Basic Operations on Stack**

In order to make manipulations in a stack, there are certain operations provided to us.

- push() to insert an element into the stack
- pop() to remove an element from the stack
- top() Returns the top element of the stack.
- isEmpty() returns true if stack is empty else false.
- size() returns the size of stack.

**Push:**

Adds an item to the stack. If the stack is full, then it is said to be an Overflow condition.

Algorithm for push:

```
begin
if stack is full
    return
endif
else
    increment top
    stack[top] assign value
end else
end procedure
```

**Pop:**

Removes an item from the stack. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an Underflow condition.

Algorithm for pop:

```
begin
if stack is empty
    return
endif
else
    store value of stack[top]
    decrement top
    return value
end else
end procedure
```

**Top:**

Returns the top element of the stack.

Algorithm for Top:

```
begin
return stack[top]
end procedure
```

**isEmpty:**

Returns true if the stack is empty, else false.

Algorithm for isEmpty:

```
begin
if top < 1
    return true
else
    return false
end procedure
```

**Stack Data Structure**

**INFORMATION**

Home Developers  
About Us Contact Us  
FAQs

**FOLLOW US ON**

**CONTACT US**

1800-123-4567  
 contact@bitwardha.ac.in

Fig. 5.7 DSA Content Page of Placement Preparation Module Focusing on DSA.

### 5.3.3 Interview Questions

The screenshot shows a web page titled "Interview Questions". The top navigation bar includes links for Home, Roadmap, Prepare, Problems, Cheatsheet, and Discuss. There are also icons for a bell and user profile.

**What is a framework in programming?**

**Answer:** A framework is a platform or software developed and used by developers to build software applications. It can be used to process inputs, manage hardware, and interact with system software. It provides the basis on which web developers can build programs for a specific platform. For example, a framework can have predetermined classes as well as functions.

**What is a wrapper class in Java?**

**Answer:** Wrapper class is used in Java to access the primitive data type as an object. When we create an object to a wrapper class, it contains a field, and in this field, we can store primitive data types. In other words, we can wrap a primitive value into a wrapper class object.

**List out components of a computer system**

**Answer:** The components of a computer system are: 1.CPU (Central Processing Unit) including control unit and arithmetic logic unit 2. Memory like primary and secondary 3.Input and output devices like keyboard mouse, printer scanner, etc.

**What is SDLC?**

**Answer:** SDLC stands for Software Development Life Cycle is a process that produces quality software products in less time. The stages involve by SDLC are: 1) planning, 2) design, 4) construction, 5) testing, and 6) deployment.

**Explain the framework**

**Answer:** The framework is a platform for making software applications. It provides the basis on which developers can build programs for a specific platform. For example, a framework may include predetermined classes as well as functions. It can be used to process inputs, manage hardware, and interact with system software.

**What do you understand by a class and a superclass?**

**Answer:** Class and superclass are two important terms used mainly in Object-Oriented computer programming. In Object-Oriented programming languages such as Java and C++, a class is used to define the characteristics of an object. It specifies how they will respond to a message and what type of message the object will respond to. In other words, we can say that a class that we have derived from another class is called a subclass. It is also called a derived class, extended class, or child class. The class from which the subclass is derived is called a superclass. It is also called a base class or a parent class. A superclass is the basis of the class being considered.

**What is a constructor in an object-oriented programming language?**

**Answer:** In a class-based object-oriented programming language, a constructor is a special method in the class that is automatically called when the object of that class is created. The constructors have the same name as the class, and they usually initialize the data members of the new object. A constructor is very similar to an instance method, but the difference between the constructor and the method is that the constructor has no explicit return type, it is not implicitly inherited, and it usually has different rules for scope modifiers. If you do not write a constructor in your program, the first thing the object does when it is created is that it looks for a constructor. Java automatically creates a default constructor and calls it if it doesn't find the constructor. Here, we are talking about Java (A popular object-oriented programming language). Let's take an example to understand it well. There are two rules to write a constructor: 1. Class name and Constructor name should be the same. 2. It should not have any return type.

**<h3>What Is A Super-class?</h3>**

**Answer:** <p><span style="background-color: #28a745; color: white; padding: 2px 5px; border-radius: 10px; font-weight: bold; font-size: 0.9em; margin-right: 5px; ">A super class is the basis of all the classes. The object of the rest of the class has all the characteristics related to the superclass.</span></p>

**INFORMATION**

Home      Developers  
About Us      Contact Us  
FAQs

**FOLLOW US ON**

Instagram      LinkedIn      YouTube      Telegram

**CONTACT US**

1800-123-4567      contact@bitwardha.ac.in

**Fig. 5.8** Interview Questions Page of Placement Preparation Module for DSA.

## 5.4 Problems Page

This interface contains all the coding questions will be displayed based on topic wise and difficulty level wise. The coding interface will have Problem Description on left half side and code editor will be on right side.

SR NO.	TITLE	DIFFICULTY	ACTION
1	Reverse the string	Easy	<button>Solve</button>
2	Two Sum	Easy	<button>Solve</button>
3	Container With Most Water	Medium	<button>Solve</button>
4	Sudoku Solver	Hard	<button>Solve</button>
5	Count Primes	Easy	<button>Solve</button>
6	Remove Element	Easy	<button>Solve</button>

**Fig. 5.9** Problems Page of Placement Preparation Module for DSA.

**Container With Most Water** Medium

Array

You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the i<sup>th</sup> line are (i, 0) and (i, height[i]). Find two lines that together with the x-axis form a container, such that the container contains the most water. Return the maximum amount of water a container can store.

**Sample Input:**  
height = [1,8,6,2,5,4,8,3,7]

**Sample Output:**  
49

**Custom Input**

Custom input

Compile and Execute

**Output**

**Fig. 5.10** Code Editor of Placement Preparation Module for DSA.

## 5.5 Cheatsheet Page

The interface contains the summary of important DSA contents such as time and space complexity of various data structures and algorithms and short revision material.

**Data Structures**

Data Structures	Time Complexity							Space Complexity	
	Average			Worst					
	Indexing	Search	Insertion	Deletion	Indexing	Search	Insertion		Deletion
Basic Array	O(1)	O(n)	-	-	O(1)	O(n)	-	-	O(n)
Dynamic Array	O(1)	O(n)	O(n)	O(n)	O(1)	O(n)	O(n)	O(n)	O(n)
Singly-Linked List	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Doubly-Linked List	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Hash Table	-	O(1)	O(1)	O(1)	-	O(n)	O(n)	O(n)	O(n)
Binary Search Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)
AVL Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)

**Array Sorting Algorithms**

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	
Quick Sort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n^2)$	$O(\log(n))$
Merge Sort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n \log(n))$	$O(n)$
Heap Sort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n \log(n))$	$O(1)$
Bubble Sort	$O(n)$	$\Theta(n^2)$	$O(n^2)$	$O(1)$
Insertion Sort	$O(n)$	$\Theta(n^2)$	$O(n^2)$	$O(1)$
Selection Sort	$\Omega(n^2)$	$\Theta(n^2)$	$O(n^2)$	$O(1)$
Radix Sort	$\Omega(nk)$	$\Theta(nk)$	$O(nk)$	$O(n+k)$

**Data Structures**

**Array**

**Definition**

- Stores data elements based on a sequential, most commonly 0-based index.
- Based on tuples from set theory.
- One of the oldest and most commonly used data structures.

**What you need to know**

- Optimal for indexing; bad at searching, inserting, and deleting (except at the end).
- Linear arrays or one-dimensional arrays are the most basic.
  - Static in size, declared with a fixed size.
  - Dynamic arrays are similar to one-dimensional arrays but have reserved space for additional elements.
    - If a dynamic array is full, it copies its contents to a larger array.
  - Multi-dimensional arrays are nested arrays that allow multiple dimensions, such as an array of arrays providing a 2-dimensional spatial representation via x, y coordinates.

**Time Complexity**

- Indexing: Linear array: O(1), Dynamic array: O(1)
- Search: Linear array: O(n), Dynamic array: O(n)
- Optimized Search: Linear array: O(log n), Dynamic array: O(log n)
- Insertion: Linear array: n/a, Dynamic array: O(n)

**Linked List**

**Definition**

- Stores data with nodes that point to other nodes.
  - Nodes, at its most basic it has one datum and one reference (another node).
  - A linked list chains nodes together by pointing one node's reference towards another node.

**Fig. 5.11** Cheatsheet Page of Placement Preparation Module for DSA.

## 5.6 Discuss Page

This interface has question and answer section and user will add question through filling a form with question and submit it and also user can add answers to the questions.

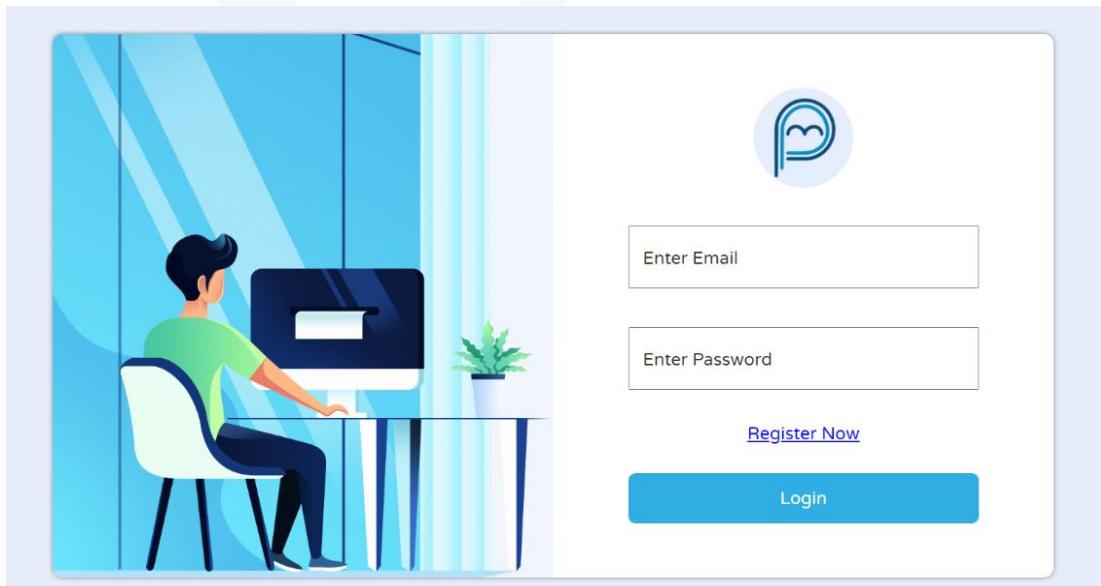
The screenshot shows a discussion forum interface with three posts listed:

- Post 1:** Question: "What are Linked Lists?", Status: "I want to know something about Linked Lists.", Tags: Data Structures, Linked List, DSA, Likes: 0, Views: 10, Answers: 0, Submitted by Suyash Patalbansi on 11/26/2023, 12:35:04 PM.
- Post 2:** Question: "What are methods called as?", Status: "Explain methods", Tags: methods, Likes: 0, Views: 4, Answers: 0, Submitted by Shreya Raut on 11/21/2023, 11:31:17 AM.
- Post 3:** Question: "What are Arrays and ArrayList?", Status: "I want to know about the main difference in Arrays and ArrayList", Tags: DSA, Data Structures, Likes: 3, Views: 1, Answers: 0, Submitted by Suyash Patalbansi on 11/21/2023, 10:29:04 AM.

**Fig. 5.12** Discussion Forum of Placement Preparation Module for DSA.

## 5.7 Login Page

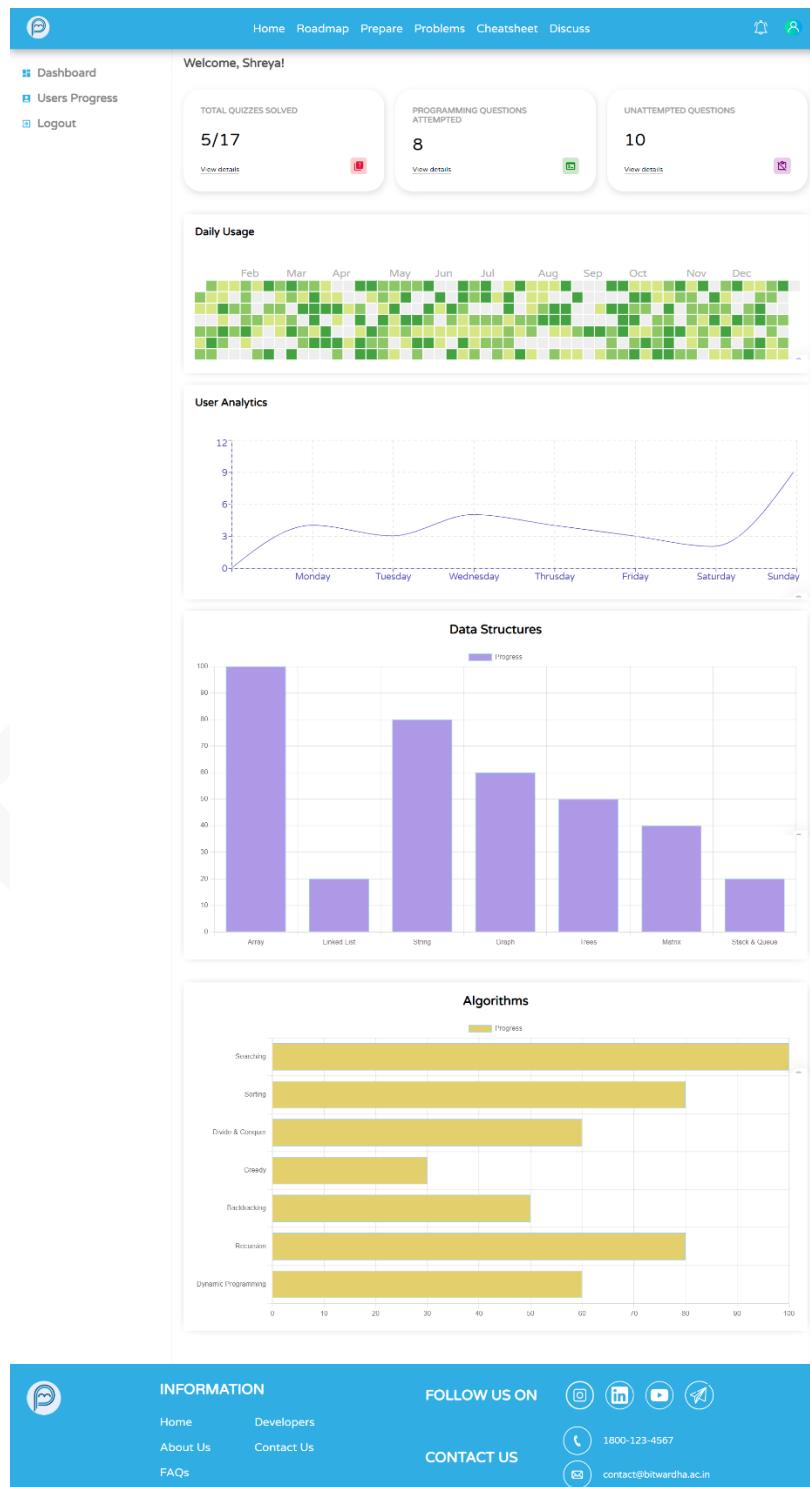
The login interface enables you to integrate user login with the content of our website. Login as a user or admin.



**Fig. 5.13** Login Page of Placement Preparation Module for DSA.

## 5.8 User Dashboard

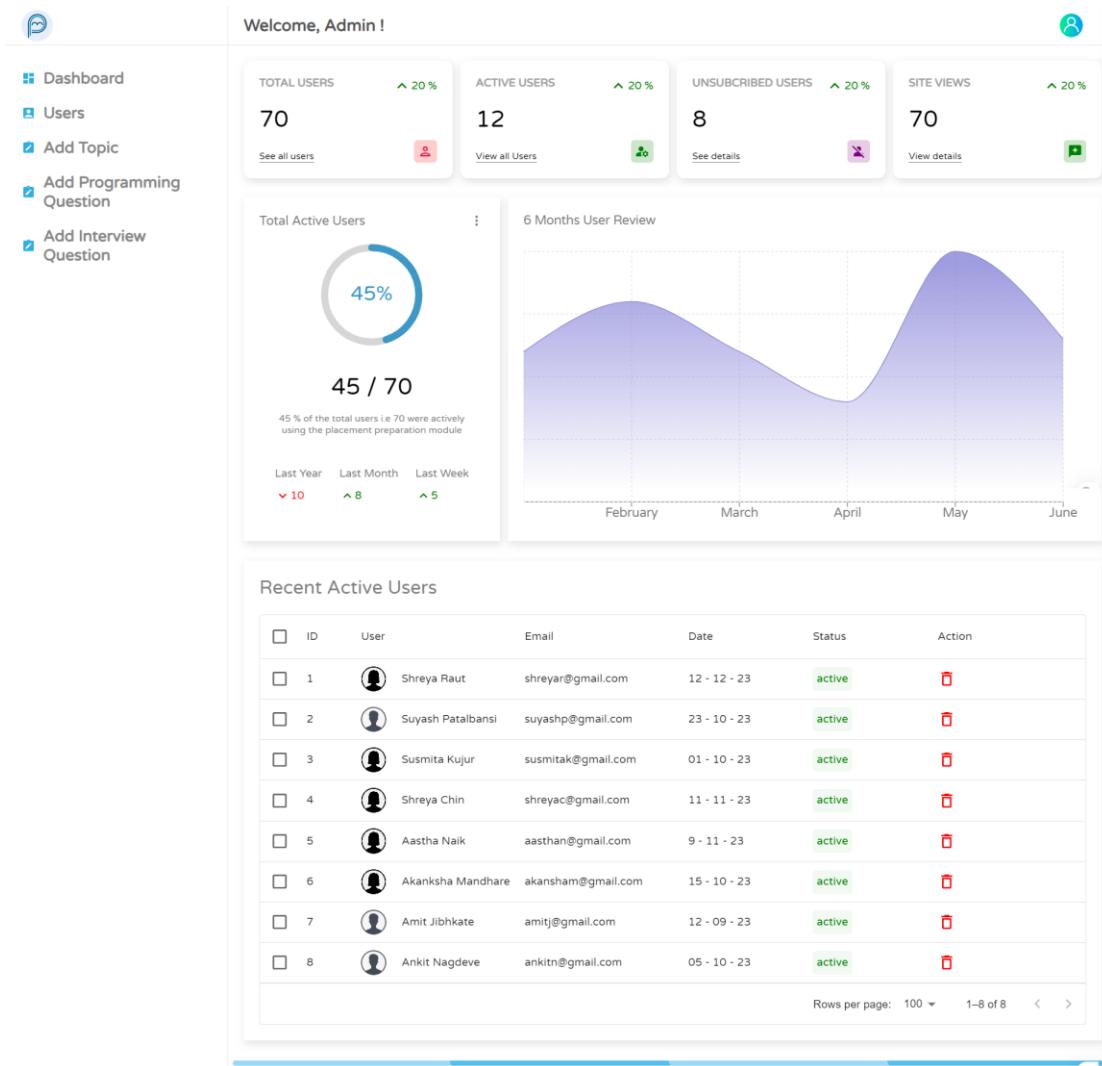
The homepage interface enables user to navigate through the concept and purpose of learning DSA and learn about features of our website. There is also contact us sections with option of newsletters.



**Fig. 5.14** User Dashboard of Placement Preparation Module for DSA.

## 5.9 Admin Dashboard

In this interface Admin will add new content or edit the previous content. The admin dashboard has user statistics and page statistics.



**Fig. 5.15** Admin Dashboard of Placement Preparation Module for DSA.

Welcome, Admin !

**Add Topic:**

Topic Name:

Description:  

Normal **B** **I** **U** **%** **≡** **≡** **T<sub>x</sub>**

Image URL:  
(Upload your image on [https://drive.google.com/drive/u/0/folders/1aG2YKVETv-5ohP\\_X6XrkVeryRjTe6Bga](https://drive.google.com/drive/u/0/folders/1aG2YKVETv-5ohP_X6XrkVeryRjTe6Bga) and provide its url)

**Advantages:**  

Normal **B** **I** **U** **%** **≡** **≡** **T<sub>x</sub>**

**Disadvantages:**  

Normal **B** **I** **U** **%** **≡** **≡** **T<sub>x</sub>**

**Applications:**  

Normal **B** **I** **U** **%** **≡** **≡** **T<sub>x</sub>**

**Add Sub Topics:**

Name:  
 \*

Description:  
 \*

Available Program Links:

C:

C++:

Java:

**Add Topic**

**Fig. 5.16 Add Content in Admin Dashboard.**

## 5.10 Backend Database

The screenshot shows the MongoDB Compass interface with the database 'placementprep...' selected. The left sidebar lists databases (admin, local, toDoList, users) and collections (answers, interviewquestions, mcquizzes, programmingques, users). The main area displays five collections: 'answers', 'interviewquestions', 'mcquizzes', 'programmingques', and 'users'. Each collection card provides storage size, document count, average document size, index count, and total index size.

Collection	Storage size	Documents	Avg. document size	Indexes	Total index size
answers	20.48 kB	6	188.00 B	1	36.86 kB
interviewquestions	20.48 kB	8	510.00 B	1	36.86 kB
mcquizzes	24.56 kB	25	364.00 B	1	36.86 kB
programmingques	20.48 kB	6	597.00 B	1	36.86 kB
users					

**Fig. 5.17** MongoDB Database of Placement Preparation Module for DSA.

The screenshot shows the MongoDB Compass interface with the 'mcquizzes' collection selected under the 'users' database. The left sidebar lists databases (admin, local, toDoList, users) and collections (answers, feeds, interviewquestions, mcquizzes, programmingques, questions, subtopics, topics, users). The main area shows the 'mcquizzes' collection with 25 documents and 1 index. It includes a 'Documents' tab showing three array objects and a 'More Options' button.

```

_id: ObjectId('655c76324b5a673e2e9666a2')
topic: "Array"
difficultyLevel: "Easy"
question: "Which of these best describes an array?"
  > options: Array
    correctAnswer: "b) Container of objects of similar types"
    ...V: 9

_id: ObjectId('655c76324b5a673e2e9666a2')
topic: "Array"
difficultyLevel: "Easy"
question: "How do you instantiate an array in Java?"
  > options: Array
    correctAnswer: "c) int arr() = new int(3);"
    ...V: 9

_id: ObjectId('655c76324b5a673e2e9666a2')
topic: "Array"
difficultyLevel: "Easy"
question: "With the help of which operator array elements can be accessed?"
  > options: Array
    correctAnswer: "c) Subscript Operator [ ]"
    ...V: 9

```

**Fig. 5.18** MongoDB Database (quiz) of Placement Preparation Module for DSA.

# **CHAPTER 6**

## **CONCLUSIONS AND FUTURE SCOPE**

---

### **6.1 Conclusion**

In conclusion, the placement preparation DSA module is designed to provide a comprehensive and user-friendly platform for individuals aiming to excel in technical interviews and competitive coding assessments. The project objectives, outlined in the initial scope, were successfully translated into a functional and feature-rich application, aligning with the identified requirements. The architecture, based on the MERN stack, ensures a seamless flow of data and interactions between the frontend and backend components. The incorporation of a chatbot feature enhances user engagement, providing real-time assistance and fostering a more interactive learning experience.

The functional requirements address key aspects such as user authentication, learning resources, quizzes, discussions, progress tracking, and more. The chatbot, as a new addition, adds a layer of user support, allowing for natural language interactions and guidance. Non-functional requirements underscore the importance of performance, reliability, security, and usability. Ensuring scalability, maintainability, and compliance with industry standards further solidify the robustness of the platform.

The project's responsiveness to user needs, demonstrated by features like visual progress tracking, cheatsheets, and a user-driven Q&A system, aligns with the goal of creating a dynamic and adaptive learning environment. The platform's compatibility with various devices and its commitment to accessibility enhance inclusivity and user satisfaction. Moving forward, the project is poised for scalability and future enhancements. The establishment of a feedback mechanism encourages ongoing user engagement and ensures that the platform remains responsive to evolving requirements. The integration of a code editor, multi-language support, and external API compatibility positions the platform for versatility and broader integration possibilities.

In summary, the placement preparation DSA module stands as a comprehensive solution, leveraging technology to empower users in their journey towards mastering Data Structures and Algorithms, and ultimately excelling in their placement pursuits.

The collaboration of technical prowess, thoughtful design, and user-centric features solidify its potential impact in the realm of educational technology.

## 6.2 Future Scope

Although our placement preparation DSA module has successfully achieved its objectives, there are several potential avenues for future development and enhancement:

### 1. Advanced Learning Paths:

- Introduce advanced learning paths for specialized topics within Data Structures and Algorithms.
- Include modules for advanced algorithms, optimization techniques, and real-world application scenarios.

### 2. Collaborative Coding Environment:

- Implement a collaborative coding feature, allowing users to work together on coding challenges and projects.
- Support real-time code sharing and collaborative debugging.

### 3. Integration with Industry Platforms:

- Explore partnerships and integrations with industry platforms for real-world application scenarios and exposure to industry-standard coding practices.

### 4. Machine Learning and Personalization:

- Implement machine learning algorithms to analyze user performance data and provide personalized recommendations for improvement.
- Tailor learning paths based on individual strengths and weaknesses.

### 5. Gamification and Challenges:

- Introduce gamification elements such as badges, leaderboards, and virtual challenges to enhance user engagement and motivation.
- Organize coding competitions and challenges to simulate real-world scenarios.

## 8. Enhanced Chatbot Capabilities:

- Improve chatbot capabilities through continuous integration of natural language processing advancements.
- Enable the chatbot to provide more personalized and context-aware assistance.

## 9. Interactive Coding Challenges:

- Develop interactive coding challenges that simulate real-world scenarios and require problem-solving skills.
- Include scenarios from different domains like artificial intelligence, machine learning, and cybersecurity.

## 10. Advanced Analytics and Reporting:

- Enhance visual progress tracking with advanced analytics tools.
- Provide in-depth performance reports with insights into strengths, weaknesses, and areas for improvement.

## 11. Mobile Application Development:

- Develop a dedicated mobile application to facilitate on-the-go learning and practice.
- Ensure a seamless transition and synchronization between the web and mobile platforms.

## 13. Community Features:

- Expand community features by introducing forums, interest groups, and mentorship programs.
- Foster a sense of community and collaborative learning among users.

The future scope of the project lies in the pursuit of continuous innovation, adaptability to evolving educational trends, and a steadfast commitment to providing users with a cutting-edge and personalized learning experience in the realm of Data Structures and Algorithms.

## REFERENCES

---

- [1] M. P. Burgos, “Learning management system for data structures and algorithm,” *International Journal of Science, Technology, Engineering and Mathematics*, vol. 1, no. 1, pp. 1–26, 2021. doi:10.53378/352852
- [2] Z. S. Seidametova, “Some methods for improving data structure teaching efficiency,” *Educational Dimension*, vol. 6, pp. 164–175, 2022. doi:10.31812/educdim.4509
- [3] E. B. Costa, A. M. Toda, M. A. Mesquita, F. T. Matsunaga, and J. D. Brancher, “Interactive Data Structure Learning Platform,” *Computational Science and Its Applications – ICCSA 2014*, pp. 186–196, 2014. doi:10.1007/978-3-319-09153-2\_14
- [4] L. Yu, X. Zheng, and Y. Biao, “Research on data structure course teaching system based on open teaching model,” *Proceedings of the 2nd Symposium on Health and Education 2019 (SOHE 2019)*, 2019. doi:10.2991/sohe-19.2019.73
- [5] I. Gaber, A. Kirsh, and D. Statter, “Studied questions in data structures and algorithms assessments,” *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1*, 2023. doi:10.1145/3587102.3588843
- [6] N. Adzharuddin, “Learning management system (LMS) among university students: Does it work?,” *International Journal of e-Education, e-Business, e-Management and e-Learning*, 2013. doi:10.7763/ijeeee.2013.v3.233
- [7] M. Mcquaigue, D. Burlinson, K. Subramanian, E. Saule, and J. Payton, “Visualization, assessment and analytics in data structures learning modules,” *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 2018. doi:10.1145/3159450.3159460
- [8] D. Dicheva and A. Hodge, “Active learning through game play in a Data Structures course,” *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 2018. doi:10.1145/3159450.3159605
- [9] E. Budiman, H. Haeruddin, U. Hairah, and F. Alameka, “Mobile learning: Visualizing contents media of data structures course in mobile networks,” *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, <https://jtec.utm.edu.my/jtec/article/view/3877> (accessed Oct. 25 2023).
- [10] D. B. Silva, R. de Aguiar, D. S. Dvconlo, and C. N. Silla, “Recent studies about teaching algorithms (CS1) and data structures (CS2) for computer science students,” *2019 IEEE Frontiers in Education Conference (FIE)*, 2019. doi:10.1109/fie43999.2019.9028702

- [11] A. Jagadeesha, P. Rayavaram, J. Marwad, S. Narain, and C. S. Lee, “Integrating data structures and algorithms in K-12 education using block-based programming,” *2023 IEEE Global Engineering Education Conference (EDUCON)*, 2023. doi:10.1109/educon54358.2023.10125165
- [12] C. Segura, I. Pita, R. del Vado Vírseda, A. I. Saiz, and P. Soler, “Interactive learning of data structures and algorithmic schemes,” *Computational Science – ICCS 2008*, pp. 800–809, 2008. doi:10.1007/978-3-540-69384-0\_85
- [13] P. Hubwieser, M. Berges, M. Striewe, and M. Goedicke, “Towards competency based testing and feedback: Competency definition and measurement in the field of Algorithms & Data Structures,” *2017 IEEE Global Engineering Education Conference (EDUCON)*, 2017. doi:10.1109/educon.2017.7942896
- [14] E. Fouh, S. Hamouda, M. F. Farghally, and C. A. Shaffer, “Automating learner feedback in an eTextbook for data structures and algorithms courses,” *Formative Assessment, Learning Data Analytics and Gamification*, pp. 135–165, 2016. doi:10.1016/b978-0-12-803637-2.00008-7
- [15] V. Karavirta and C. A. Shaffer, “Creating engaging online learning material with the JSAV JavaScript algorithm visualization library,” *IEEE Transactions on Learning Technologies*, vol. 9, no. 2, pp. 171–183, 2016. doi:10.1109/tlt.2015.2490673
- [16] E. Fouh *et al.*, “Design and architecture of an interactive etextbook – the opendsa system,” *Science of Computer Programming*, vol. 88, pp. 22–40, 2014. doi:10.1016/j.scico.2013.11.040