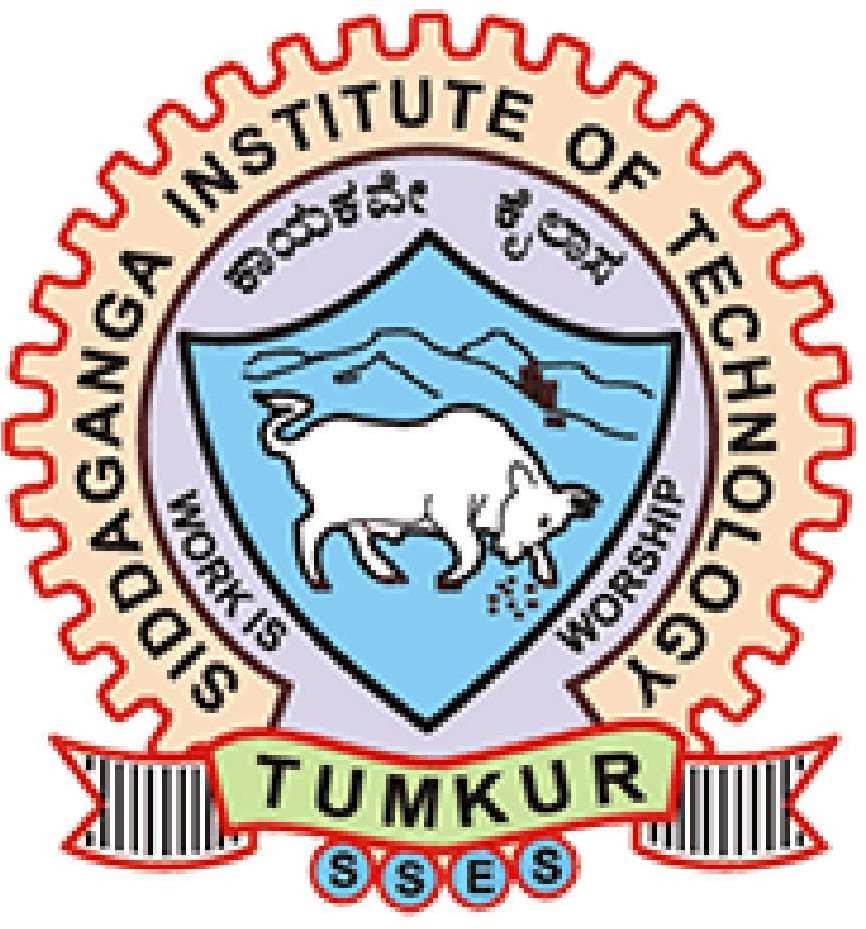
VISVESVARAYA TECHNOLOGICAL UNIVERSITY JNANA SANGAMA, BELAGAVI - 590014



Major Project Report on

## “A Web-Based Coding Practice Platform to Empower MCA Students”

Submitted in the partial fulfillment of the requirements for the degree of

Master of Computer Applications

Submitted By

H S Nagendra Hebbar

(USN- lSI23MC012)

Under the Guidance of

Dr. C Bhanuprakash Dept. of MCA

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

(Program Accredited by NBA, New Delhi)

SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMAKURU-572 103

(An autonomous institution affiliated to VTU, Belagavi, Approved by AICTE, New Delhi, Accredited by NAAC with 'A++’ Grade & ISO 9001-2015 Certified)

2024-2025

SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMAKURU-572 103

(An **autonomous institution affiliated to** VTU, **Belagavi, Approved by** AICTE, **New Delhi,**

Accredited by NAAC with ‘A++’ Grade & ISO 9001-2015 Certified)

**DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS**

(Accredited by NBA, New Delhi)

**Certificate**

Certified that the project work entitled “A Web-Based Coding Practice Platform to Empower MCA Students” is a bonafide work carried out by Mr. H S Nagendra Hebbar, USN 1SI23MC012 in partial fulfillment of 4th Semester Major Project (N4MCMP) for the award of the degree Master of Computer Applications from Siddaganga Institute of Technology, an autonomous institute under Visvesvaraya Technological University, Belagavi during the academic year 2024-25.

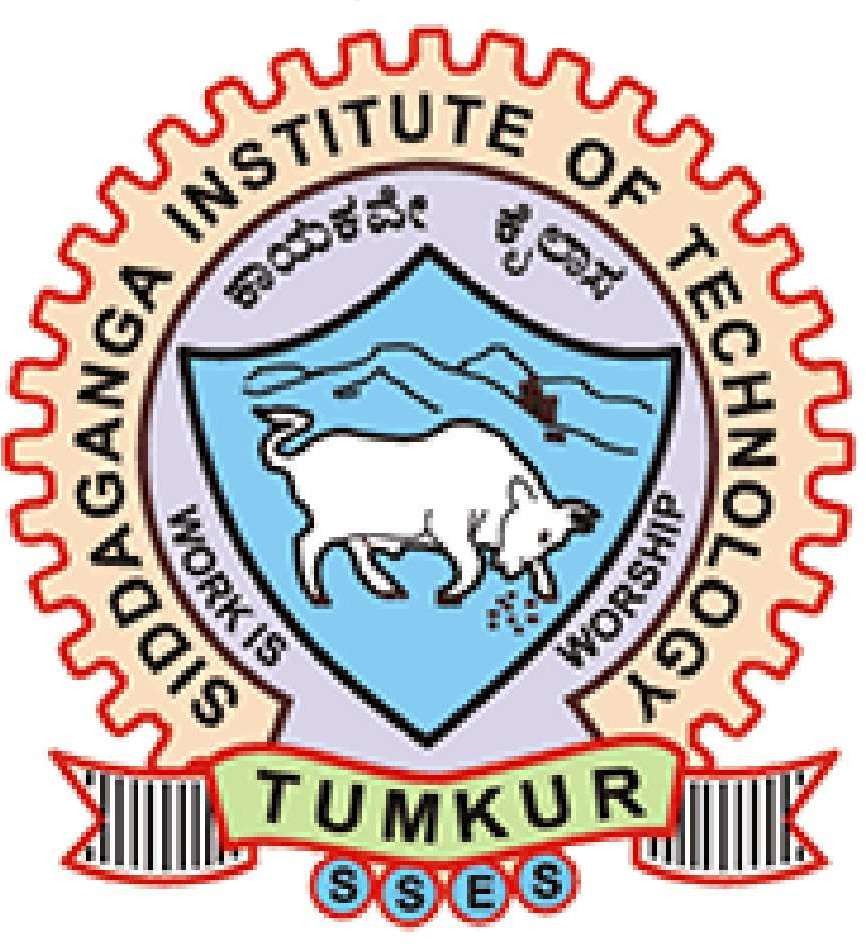
The Major Project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the Master of Computer Applications.

Dr. C Bhanuprakash

Assistant Professor Internal Guide

Prof. Premasudha B G Professor and Head

Prof. S V Dinesh Principal



Internal Examiner External Examiner

**DECLARATION**

I, *Mr. H S Nagendra Hebbar,* student of final semester Master of Computer Applications , Siddaganga Institute of Technology, Tumakuru, hereby declare that the dissertation work entitled “**A Web-Based Coding Practice Platform to Empower MCA Students”** has been carried out under the supervision of *Dr. C Bhanuprakash, Assistant Professor,* Department of Master of Computer Applications, Siddaganga Institute of Technology, Tumakuru, for the partial fulfillment of the requirements for the award of the degree of Master of computer Applications by Visvesvaraya Technological University, Belagavi during the academic year 2024-2025.

I also declare that this Dissertation entitled “**A Web-Based Coding Practice Platform to Empower MCA Students”** has not been submitted by anybody for any of the universities so far for the award of any degree.

Date:



Place: Tumakuru

H S Nagendra Hebbar (USN: 1SI23MC012)



## SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMAKURU-572 103

(An autonomous institution affiliated to VTU, Belagavi, Approved by AICTE, New Delhi, Accredited by NAAC with ‘A++’ Grade & ISO 9001-2015 Certified)

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

(Accredited by NBA, New Delhi)

VISION STATEMENT

*“To effectively mould quality and responsible Computer Professionals, with a mind of service and spirituality for nurturing the technological competence”*

MISSION STATEMENT

*“Imparting quality education to students to make them professionals in their domain replete with IT and computational skills par excellence”*

PROGRAM EDUCATIONAL OBJECTIVES

PEO-1: - Pursue career in computer applications domain by developing abilities that are in synchrony with changing needs of Industry or academia

PEO-2: - Demonstrate professionalism when working with teams and align with ethical principles.

PEO-3: - Engage in lifelong learning to upgrade the professional skills

## SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMAKURU-572103

(An autonomous institution affiliated to VTU, Belagavi, Approved by AICTE, New Delhi, Accredited by NAAC with ’A++’ Grade & ISO 9001-2015 Certified)

DEPARTMENT OF MASTER **OF COMPUTER APPLICATIONS**

(Accredited by DBA, New Delhi)

**Course Outcomes (COs)**

CO1: Identify the relevant domain and apply the computing knowledge for the problem.

CO2: Analyse the defined problem and identify the corresponding requirements / objectives by conducting a literature survey.

CO3: Design and develop a model / procedure / algorithm/ incorporating project relevance to the global goals

CO4: Implement the developed model using appropriate software tools / technologies.

CO5: Prepare a report and demonstrate the project work.

CO6: Apply the management principles for managing projects.

CO7: Adhere to professional ethics and norms of computing practices.

CO8: Effectively engage in lifelong learning.

## SIDDAGANGA INSTITUTE OF TECHNOLOGY, TUMAKURU-572103

**(An autonomous institution affiliated to VTU, Belagavi, Approved by AICTE, New Delhi, Accredited by NAAC with ‘A++’ Grade & ISO 9001-2015 Certified)**

### DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

**(Accredited by NBA, New Delhi)**

**Program Outcomes (POs)**

PO1: Foundation Knowledge: Apply knowledge of mathematics, programming logic and coding fundamentals for solution architecture and problem solving.

PO2: Problem Analysis: Identify, review, formulate and analyse problems primarily focusing on customer requirements using critical thinking frameworks.

PO3: Development of Solutions: Design, develop and investigate problems with as an innovative approach for solutions incorporating ESG/SDG goals.

PO4: Modern Tool Usage: Select, adapt and apply modern computational tools such as development of algorithms with an understanding of the limitations including human biases.

PO5: Individual and Teamwork: Function and communicate effectively as an individual or a team leader in diverse and multidisciplinary groups. Use methodologies such as agile.

PO6: Project Management and Finance: Use the principles of project management such as scheduling, work breakdown structure and be conversant with the principles of Finance for profitable project management.

PO7: Ethics: Commit to professional ethics in managing software projects with financial aspects. Learn to use new technologies for cyber security and insulate customers from malware

PO8: Life-long learning: Change management skills and the ability to learn, keep up with contemporary technologies and ways of working.

**Table of Contents**

**Chapter Numbers**

**Contents Page**

**Number**

Acknowledgement i

Abstract ii

List of figures iii

List of tables iv

[Chapter 1: Introduction](#_TOC_250018)

* 1. [Introduction about the project 1](#_TOC_250017)
  2. [Problem Definition 2](#_TOC_250016)
  3. [Purpose and Scope of the Project 3](#_TOC_250015)
  4. [Objectives 3](#_TOC_250014)
  5. [Target Audience 3](#_TOC_250013)
  6. [Technologies Required to Build the Project 3](#_TOC_250012)

[Chapter 2: Literature Review 5](#_TOC_250011)

[Chapter 3: Methodology](#_TOC_250010)

* 1. [Methodology of the project 9](#_TOC_250009)
  2. System development life cycle (SDLC) 10
     1. Project Planning 11
        1. [Stakeholders Identification & Roles 11](#_TOC_250008)
        2. Data collection 12
        3. [Workflow Diagram 14](#_TOC_250007)
        4. Work breakdown structure & Project Timeline 16
        5. [Hardware requirement 17](#_TOC_250006)
        6. [Software requirement 17](#_TOC_250005)
        7. Functional & Non-functional Requirements 18
     2. System Design 21
        1. ER diagram 21
        2. [Use Case Diagram 23](#_TOC_250004)
        3. [Activity Diagram 24](#_TOC_250003)
        4. [Sequence Diagram 26](#_TOC_250002)
     3. [Implementation, maintenance, and support 27](#_TOC_250001)
     4. [Testing and Validation 28](#_TOC_250000)

|  |  |  |
| --- | --- | --- |
| Chapter 4: | Results and Discussion | 31 |
|  | 4.1 Snapshots | 32 |
| Chapter 5: | Conclusion and Future Enhancements | 47 |
|  | 5.1 Conclusion | 47 |
|  | 5.2 Future Enhancements | 48 |
| Chapter 6: | Teamwork Details |  |
|  | Development Team | 49 |
|  | Backend Developers |  |
|  | Stakeholders |  |
|  | Collaboration and Expertise |  |
|  | References | 51 |
|  | Project Self-assessment for Program Outcomes |  |
|  | Plagiarism Certificate |  |

# ACKNOWLEDGEMENT

At the outset, I submit my reverential pranamas to the divine soul of his Holiness Dr. Sree Sree Sivakumara Swamigalu, Founder President, and his Holiness Sree Sree Siddalinga Swamigalu, President, Sree Siddaganga Education Society, Tumakuru, for their eternal blessings.

I express my sincere gratitude to Dr. Shivakumaraiah, CEO & Joint Secretary SSES and Dr. S. V. Dinesh, Principal, Siddaganga Institute of Technology, Tumakuru for providing an excellent academic environment.

I am grateful to Dr. Premasudha B. G., Professor and Head, Department of MCA, for her guidance, motivation, and support during this project work.

It is with profound appreciation that I thank my internal guide, Dr. C Bhanuprakash, Assistant Professor, Department of Master of Computer Applications, SIT, Tumakuru, for his valuable guidance, insightful feedback, and constant encouragement provided at every stage of the project.

My sincere thanks to Dr. Prashanth G. K., Project Coordinator, Department of MCA, for his consistent guidance and valuable input during the project.

I would also like to acknowledge the support and cooperation of all those who have directly or indirectly contributed to the successful completion of my project.

Finally, I extend my heartfelt thanks to my family and friends for their unwavering support, patience, and encouragement throughout the project duration.

H S Nagendra Hebbar 1SI23MC012

# ABSTRACT

The existing online coding platforms allow users to practice problems and participate in contests but often lack flexibility in custom problem creation, seamless contest management, and role-based access. Many systems are either too complex for small organizations or too limited for academic use. The proposed project introduces an integrated coding and contest platform that streamlines learning and assessment, while addressing these limitations through automation, scalability, and user-friendly design.

This proposed work describes the development of A web-based coding and assessment system. The platform comprises several modules: The User Module enables learners to register, log in, attempt coding problems, and participate in contests. The Admin Module provides administrators with tools to create, update, and delete problems and contests, manage participants, and track statistics. The Problem and Submission Module supports structured problem statements, test cases, and automated evaluation using Judge0. The Contest Module facilitates time-bound competitions and records participant performance, ensuring fair assessments. Together, these modules create a cohesive environment for practice and competitive coding.

The proposed work is implemented using React for the front-end, Node.js with Express for the backend, MongoDB for data storage, and Dockerized Judge0 as the execution engine. Security is ensured with JWT-based authentication and bcrypt password hashing, while containerization through Docker guarantees portability and consistency.

The final system underwent validation to ensure accuracy in user registration, login, contest scheduling, submission evaluation, and admin CRUD operations. All tests confirmed that the platform performs reliably and provides a smooth user experience across different roles.

The project successfully delivers a complete coding and contest management solution, enhancing usability for both learners and administrators. Future enhancements may include AI-driven problem recommendations, advanced analytics dashboards, collaborative coding features, and mobile support to extend accessibility and engagement.

### List of Figures

#### Figure Number

**Name of the Figure**

**Page Number**

* 1. visual representation of entire SDLC process 11
  2. Coding Problems from Kaggle dataset 13
  3. This Figure illustrates the contests document from the

14

codenest collection

* 1. Workflow diagram for user 15
  2. Workflow diagram for admin 15
  3. Gantt Chart Depicting the Project Timeline and Task

17

Sequence for the project

* 1. Entity Relationship Diagram 21
  2. Use Case Diagram 23
  3. Activity Diagram 25
  4. Sequence Diagram 26
  5. Login page for Admin 32
  6. Admin Dashboard 33
  7. Contest Management Page 34
  8. Problem Management Page 35
  9. Submission Page of Admin 36
  10. Admin Profile Page 37
  11. User Sign-Up Screen 38
  12. User Login Page 39
  13. User Dashboard Page 40
  14. User Contest Page 41
  15. Contest Registration Page 42
  16. Problem Solving Page 43
  17. Submission Page of User 44
  18. Leader Board Page 45
  19. User Profile Page 46

### List of Tables

**Table Number**

**Name of Table**

**Page Number**

2.1

Summary of Related Works in A Web-Based Coding

7

Practice Platform to Empower MCA Students

* 1. Detailed Planning of Project 16
  2. Test Cases 29

# CHAPTER 1 INTRODUCTION

#### Introduction about the project

The rapid growth of online education has transformed how students learn and practice programming. Traditional teaching approaches often depend on lectures, written assignments, and manual evaluation, which delay feedback and limit opportunities for hands-on problem-solving. Since programming is a skill that improves through practice and immediate correction, the absence of timely feedback slows learning progress and reduces motivation. Moreover, instructors face challenges in scaling assessments for larger groups, which makes the process less effective and time-consuming.

Online coding platforms have emerged as a solution to these challenges by providing interactive environments where students can attempt problems and receive instant evaluation. These platforms reduce instructor workload through automated grading while helping learners stay engaged through problem libraries and contests. The combination of guided practice and competition enhances learning outcomes, builds confidence, and prepares students for real-world scenarios such as coding interviews and programming competitions.

This project introduces project, a web-based coding platform that integrates problem- solving, contests, and automated evaluation into a single application. The system allows students to register, log in, and attempt problems categorized by difficulty. Each problem includes a description, constraints, sample inputs and outputs, and hidden test cases that test deeper understanding. Contest features enable students to participate in timed challenges, simulate real assessments, and improve speed and accuracy. On the administrative side, project allows instructors to create, update, and delete problems or contests, ensuring that content stays relevant and engaging.

Technically, the system is built using a modern, scalable stack. The frontend is developed with React, delivering an intuitive and interactive interface. The backend uses Node.js with Express to handle authentication, problem logic, and API interactions. MongoDB

manages data storage for users, problems, and submissions. For automated evaluation, project integrates Judge0, a secure and reliable code execution engine that supports multiple programming languages. Docker containerization ensures the system runs smoothly across different environments, making deployment efficient and consistent.

The architecture of the project is modular, with separate components for problem management, contests, submissions, and user accounts. This modularity makes the system easier to maintain and extend with future features, such as leaderboards, gamification with badges, and advanced analytics for instructors. Testing confirmed the system’s reliability by simulating common scenarios like incorrect submissions, retries, and simultaneous contest participation. Results showed that project delivers accurate feedback and maintains performance under different use cases.

In essence, project enhances programming education by addressing the limitations of traditional methods. Its ability to provide instant feedback, manage contests, and reduce instructor workload creates a balanced platform that benefits both learners and educators. With its scalable design and technical foundation, the system can evolve into a comprehensive learning solution by introducing leaderboards, gamification, and mobile support in future versions.

The report is structured into distinct chapters to present the work in a clear and systematic manner. Chapter 2 provides the Literature Survey, which reviews existing research and related studies in coding platforms, online learning systems, and automated code evaluation. Chapter 3 focuses on the Project Methodology, explaining the development approach, system design, technologies employed, and the overall workflow followed during implementation. Chapter 4 presents the Results and Discussion, highlighting the core features of the system, outcomes of testing, and an analysis of its performance. Chapter 5 concludes the report with a summary of the key findings and outlines directions for future enhancements to improve the system further. Finally, Chapter 6 details Teamwork Contributions, documenting collaboration, role distribution, and the collective effort behind the successful development of the project.

### Problem Definition

Current online coding education platforms, while offering problem sets and practice environments, often lack the ability to deliver instant evaluation and meaningful feedback. Manual or delayed assessment reduces learning effectiveness, while limited features such as contest management, leaderboards, and adaptive problem generation fail to keep learners engaged. Additionally, the absence of an integrated system for both students and administrators creates inefficiencies in managing problems, monitoring progress, and updating content. There is a clear need for a unified and automated coding platform that ensures real-time evaluation, effective problem management, and an engaging learning experience.

### Purpose and Scope of the Project

The purpose of this project is to develop the project, an online coding platform that allows users to solve programming problems, participate in contests, and receive instant feedback on submissions. The system aims to improve the learning experience through features such as leaderboards, personalized dashboards, and real-time evaluation. Its scope includes managing users, problems, contests, and submissions while also providing an admin panel for efficient content management. Designed with scalability in mind, the platform can be extended with advanced features like adaptive problem generation and analytics to support future growth.

### Objectives

The main objectives of the project are:

* + - To build an online platform for coding practice tailored for MCA students.
    - To provide role-based access for Students and Admins.
    - To support multi-language coding: Python, C, Java, etc.

### Target Audience

The target audience for this project includes students and learners who want to enhance their programming skills through practice problems and contests, as well as educators and

institutions seeking a platform to organize coding challenges and evaluate student performance. The system also serves administrators who manage problems, contests, and leaderboards, ensuring smooth operations. Additionally, the platform will be valuable for Faculty interested in exploring online coding evaluation, automated feedback mechanisms, and the integration of scalable educational technologies.

### Technologies Required to Build the Project

The development of project integrates a combination of modern technologies to provide a seamless coding practice and contest management platform.

The frontend is built with React and Vite, ensuring a fast, interactive, and responsive user interface that supports smooth navigation for both learners and administrators.

On the backend, Node.js with Express is used to handle server-side operations and manage requests efficiently. MongoDB serves as the database, offering scalability and flexibility in storing user details, problems, contests, and submissions.

To support real-time code execution, the system integrates with the Judge0 API, enabling users to compile and run programs directly within the platform. Authentication and security are strengthened using JWT tokens and bcrypt for password encryption, ensuring safe user access. Docker is used to containerize the application, simplifying deployment and improving scalability.

Together, these technologies create a robust architecture that supports problem-solving, contest management, and user engagement in an efficient and secure manner.

# CHAPTER 2 LITERATURE SURVEY

The evolution of programming education has been significantly shaped by the use of artificial intelligence in task generation. Personalized programming tasks, which adapt to learners’ skill levels, have been shown to improve engagement and retention when integrated into online platforms [1], particularly because they reduce the repetitive nature of static exercises and encourage incremental learning. Similarly, the use of large language model-driven Parsons puzzles has demonstrated how cognitive load can be reduced during problem-solving by requiring learners to organize pre-written code fragments [2], thereby fostering logical reasoning instead of overwhelming students with syntax errors at the outset.

Effective coding platforms also emphasize the role of automated testing and structured practice. By incorporating both visible and hidden test cases, learners are exposed to more authentic problem-solving conditions [3], which enhances debugging skills and teaches the importance of edge-case handling. Platforms designed at scale have validated that thousands of simultaneous submissions can be processed without compromising reliability [4], making such systems accessible to a wider audience. Furthermore, blended learning environments that combine online practice with classroom instruction have been found to improve long-term outcomes [5], indicating that technology should complement, not replace, traditional teaching methods.

Collaboration and motivation play critical roles in sustaining learner interest. Studies confirm that group-based coding activities increase not only technical proficiency but also social engagement among students [6], creating stronger peer-learning networks. Automated hint generation frameworks provide another level of support by delivering context-sensitive guidance at scale [7], which eases the burden on instructors. Meanwhile, gamification features such as leaderboards and badges have proven highly effective in sustaining motivation [8], as they transform learning into a competitive but enjoyable experience. Unlike superficial implementations, systems that integrate gamification

directly into the core workflow are more successful in maintaining continuous engagement.

Automation in evaluation has also been central to modern programming education. Domain-specific systems that focus on assessing data structure concepts allow instructors to deliver targeted feedback [9], while advances in automated program repair demonstrate that incorrect submissions can be restructured into working solutions [10], reducing student frustration and improving persistence. However, a notable drawback of these systems is their lack of support for multiple programming languages, which limits inclusivity. Platforms that leverage universal execution engines such as Judge0 overcome this issue by enabling multi-language practice and competition within a single environment.

Accessibility and scalability are also major themes in recent research. Mobile-first coding environments have broadened access by allowing learners to practice directly on smartphones [12], an important step for students in under-resourced regions. Scalability, on the other hand, has been achieved through microservice-based architectures and real- time data processing frameworks [13], which allow coding platforms to handle large workloads efficiently. Low-latency inventory and feedback systems, when integrated into academic coding tools, further enhance the student experience by minimizing delays during compilation and submission cycles [14], ensuring that learners remain in active problem-solving flow.

Finally, interactive coding environments have long been identified as a cornerstone of effective programming education. Immediate feedback, delivered during the problem- solving process, has been proven to enhance understanding far more effectively than delayed feedback [15], highlighting the value of real-time evaluation. These insights directly influence platforms like the project, which aim to combine real-time execution, automated feedback, contests, and gamification into one cohesive system. Unlike earlier tools that addressed only individual aspects such as automated grading or contest hosting, project offers an integrated ecosystem that supports both self-learning and competitive practice, while remaining scalable and accessible to diverse user groups.

Table 2.1 Summary of Related Works in A Web-Based Coding Practice Platform to Empower MCA Students

|  |  |
| --- | --- |
| Authors | Key findings |
| [1] M. Jacobs, et al (2025) | Introduced AI-driven personalized coding tasks to improve student learning outcomes in programming education. |
| [2] Z. Hou, et al (2024) | Proposed the use of large language models to automatically generate Parsons puzzles for beginner programmers. |
| [3] A. Montoya-Dato, et al (2023) | Highlighted how coding practice platforms help students develop strong problem-solving  and test case design skills. |
| [4] Elice Team,  et al (2023) | Showcased how coding education platforms can scale effectively to support thousands of learners globally. |
| [5] D. Djenic, et al (2021) | Emphasized how blended learning models improve student engagement and coding  proficiency. |
| [6] M. Ivanović, et al (2019) | Demonstrated the value of collaborative coding tools in improving teamwork and peer learning among students. |
| [7] A. McBroom, et al (2019) | Reviewed techniques for automated hint generation in coding platforms to support struggling learners. |
| [8] R. Topalli, et al (2018) | Showed that gamification elements increase motivation and retention in programming  education. |
| [9] A. Barczak, et al (2018) | Presented an automated grading and feedback system for data structure assignments. |
| [10] S. Gulwani, et al (2016) | Proposed automated debugging and repair tools  to help beginners fix code errors effectively. |
| [11] A. Sharma, et al (2021) | Developed an academic coding platform with RESTful APIs to manage coding problems and submissions. |
| [12] M. Rajan, et al (2022) | Introduced mobile-first coding platforms to improve accessibility and flexibility in coding  education. |
| [13] A. Vashisht, et al (2025) | Discussed the benefits of microservices for scalability and real-time processing in coding platforms. |
| [14] S. Jain, et al (2022) | Demonstrated approaches to reduce latency in Spring Boot systems, useful for fast code evaluation engines. |
| [15] A. Pears, et al (2007) | Outlined the foundations of interactive coding  environments for teaching computer science effectively. |

Although prior research has advanced programming education through innovations such as AI-driven task generation, problem-solving aids, test case practice, scalable infrastructures, gamification, and automated feedback systems, these contributions often remain fragmented, focusing on isolated features within limited contexts. Existing systems rarely integrate multi-language support, real-time code execution, contests, leaderboards, and community-driven practice into a single cohesive platform. Project addresses this gap by offering a unified ecosystem that combines automated feedback, gamified learning, and scalable infrastructure, enabling learners and educators to experience a comprehensive coding environment rather than relying on multiple disconnected tools.

The next chapter introduces the Methodology followed in building the web-based coding practice platform. It explains how the project was planned and developed step by step, beginning with identifying requirements and designing the system structure. The chapter then describes how the System Development Life Cycle (SDLC) was applied to guide each stage, from planning and analysis to implementation and testing. Details will be provided on how the backend, frontend, and database were developed and connected, and how the Judge0 engine was integrated to evaluate code submissions automatically. The use of tools and technologies such as Node.js, React.js, and MongoDB will also be highlighted, showing how they were applied to make the platform scalable, reliable, and easy to use. In addition, the chapter outlines the testing and deployment process, ensuring the system meets its objectives and delivers a smooth experience for both students and administrators.

# CHAPTER 3 METHODOLGY

### Methodology of the project

This chapter presents the methodology followed in the development of project, an online coding and contest management platform. The process began with designing the core system architecture, focusing on modularity, scalability, and real-time responsiveness. The backend was built with Node.js and Express.js to handle server-side operations, while MongoDB was used for efficient data storage and management of user information, problems, submissions, and contest data.

The front-end was implemented using React and Vite, providing a dynamic and responsive user interface that allows seamless navigation between modules such as problems, contests, leaderboards, and user dashboards. To ensure real-time code execution, the platform integrates with Judge0, a robust open-source API that enables multi-language support for evaluating programming solutions.

For user authentication and security, JSON Web Tokens (JWT) were employed alongside password hashing using bcrypt, ensuring that sensitive user credentials are protected. Additionally, role-based access control was implemented to distinguish between student and admin functionalities, allowing administrators to create, update, and manage contests and problems while users focus on solving challenges.

The system also incorporates gamified features such as leaderboards and performance tracking to enhance engagement and motivation among learners. During development, extensive testing was conducted, including unit tests, integration tests, and stress testing, to ensure stability, accuracy of evaluation, and system scalability under heavy usage.

The final implementation of project delivers a comprehensive coding practice and contest platform that combines learning, evaluation, and competition in one ecosystem, offering an effective and user-friendly environment for both learners and administrators.

### Software Development Life Cycle

The System Development Life Cycle (SDLC) is the structured methodology adopted for the development of project, an online coding and contest management platform. This approach ensures that each stage of the project, from initial planning to long-term maintenance, is systematically executed to meet both functional requirements and user expectations. The SDLC framework, as illustrated in Figure 3.1, includes the phases of planning, analysis, design, implementation, testing, deployment, and maintenance.

In the planning phase, the scope, objectives, and expected outcomes of project were defined. This step established the foundation for the platform, ensuring alignment with the needs of learners, administrators, and educators. The analysis phase focused on identifying detailed functional and non-functional requirements, such as secure authentication, problem-solving features, real-time code execution, leaderboard management, and role-based access for both users and administrators.

The design phase translated these requirements into a technical model. The chosen technologies included React with Vite for the front-end, Node.js with Express.js for the back-end, and MongoDB for database management, while Judge0 was integrated for real- time code compilation and execution. Special attention was given to system scalability, security, and modularity to support future enhancements.

The implementation phase involved coding and integration of system modules, including user registration and login, problem and contest management, submission handling, and leaderboard functionality. This phase also covered the role-based access controls, ensuring that administrators could create and manage content while users engaged with coding challenges.

In the testing phase, the system underwent rigorous validation through unit testing, integration testing, and user acceptance testing. These tests ensured accuracy in code evaluation, reliability of authentication, and robustness under high user loads. Following successful validation, the deployment phase made accessible to end users.

Finally, the maintenance phase ensures the platform remains reliable and up-to-date, with regular updates, bug fixes, and the addition of new features to enhance usability and learning outcomes.

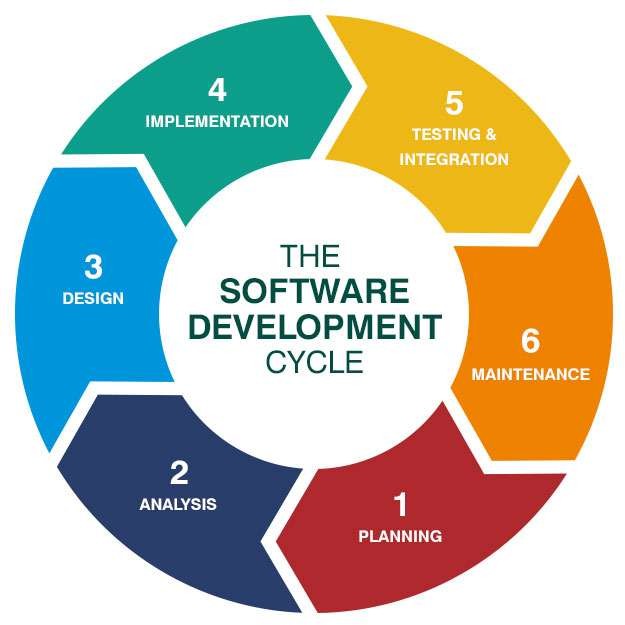


Figure 3.1 visual representation of entire SDLC process

### Project Planning

In the planning phase of the project, the main objective is to clearly define the scope, goals, and requirements of the system. This includes identifying the key functionalities such as user authentication, problem management, contest creation, submission handling, leaderboard tracking, and an intuitive user interface for both learners and administrators. Resource planning, such as setting up the development environment, choosing technologies (React/Next.js, Node.js, MongoDB, Docker, Judge0), and allocating timeframes for each module, is also carried out. This phase provides a roadmap that ensures every aspect of development is aligned with the project’s vision of delivering a reliable and user-friendly coding practice and contest platform.

#### Stakeholders Identification and Roles

Identifying stakeholders and understanding their roles is essential for the success of the project. The primary stakeholders are the end users, including students, coding enthusiasts, and competitive programmers, who depend on the platform to practice coding problems, participate in contests, and track their progress through leaderboards. Their

engagement ensures the platform fulfils its educational and skill-building purpose. Administrators and educators are also key stakeholders, as they manage problems, create contests, evaluate submissions, and maintain the overall functionality of the platform. Their role directly impacts the quality of learning resources and contest experiences provided to users.

Additionally, researchers and developers benefit from the platform by analysing submission data, identifying learning patterns, and improving the system’s algorithms for better problem recommendation and evaluation. Finally, the Judge0 API and external integration tools serve as technical stakeholders, providing the execution environment for code submissions and ensuring reliable, real-time evaluation. Together, these stakeholders contribute to building a scalable, effective, and user-centred coding platform.

#### Data Collection

The data collection for project focuses on gathering and managing essential information to support its core functionalities of coding practice, contests, and performance tracking. User data, including name, email, Password, and roles, is collected during registration for authentication and authorization. Problem data consists of titles, descriptions, difficulty levels, formats, sample inputs/outputs, constraints, and test cases, created either by admins or seeded into the system. Contest data includes details such as title, description, timing, and associated problems, enabling smooth scheduling and participation. Submission data records problem attempts, code, language, time, and execution results, which are further used to calculate scores and update leaderboards which is depicted in Figure 3.2. Performance data derived from submissions supports ranking and progress tracking, fostering competition among users. Additionally, external data is collected via the Judge0 API, which processes user code and returns execution details like results, runtime, and memory usage. Together, these datasets ensure that platform delivers secure authentication, real-time coding contests, automated evaluation, and meaningful user insights.

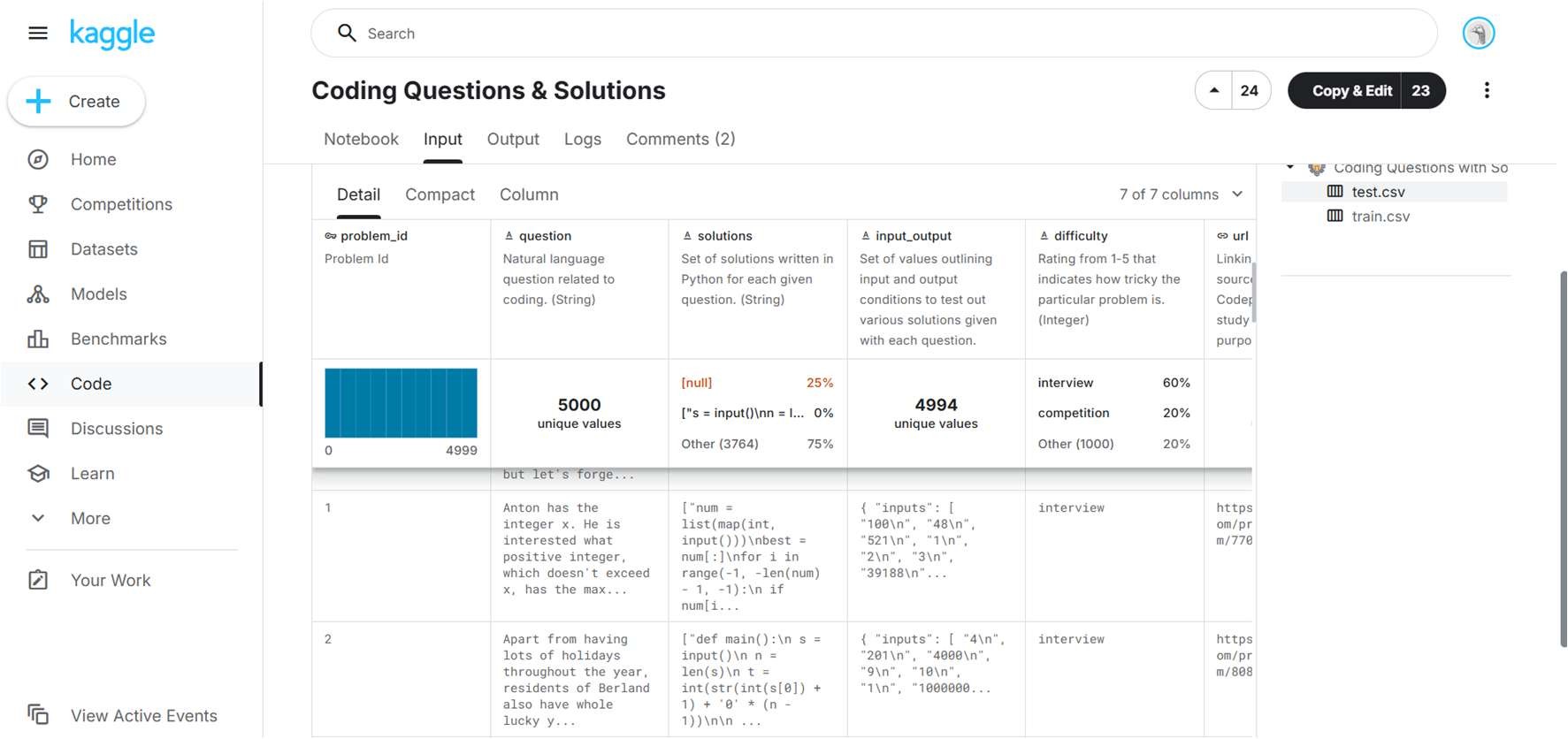


Figure 3.2 Coding Problems from Kaggle dataset

#### Data Storage and Retrieval

In the project, data storage and retrieval are handled using MongoDB Atlas, a cloud-based NoSQL database. The database named codenest is organized into multiple collections such as users, problems, contests, and submissions, each designed to store relevant information as flexible JSON-like BSON documents. For instance, the contests collection maintains information about coding contests, including the contest title, description, start and end times, and an embedded array of problems. Each problem entry contains attributes like its unique ID, title, and difficulty level as shown in Figure 3.3. This schema- less design allows hierarchical and related data to be stored without the constraints of rigid table structures. Data retrieval is performed through queries such as find, which efficiently fetch documents based on defined conditions. For example, a query on the contests collection can return the details of the CodeNest September Challenge along with all its associated problems. By using MongoDB’s document-oriented model, project Gachieves fast and efficient storage and retrieval of both parent and nested data in a single query, removing the need for complex joins and ensuring seamless access to data across the application.

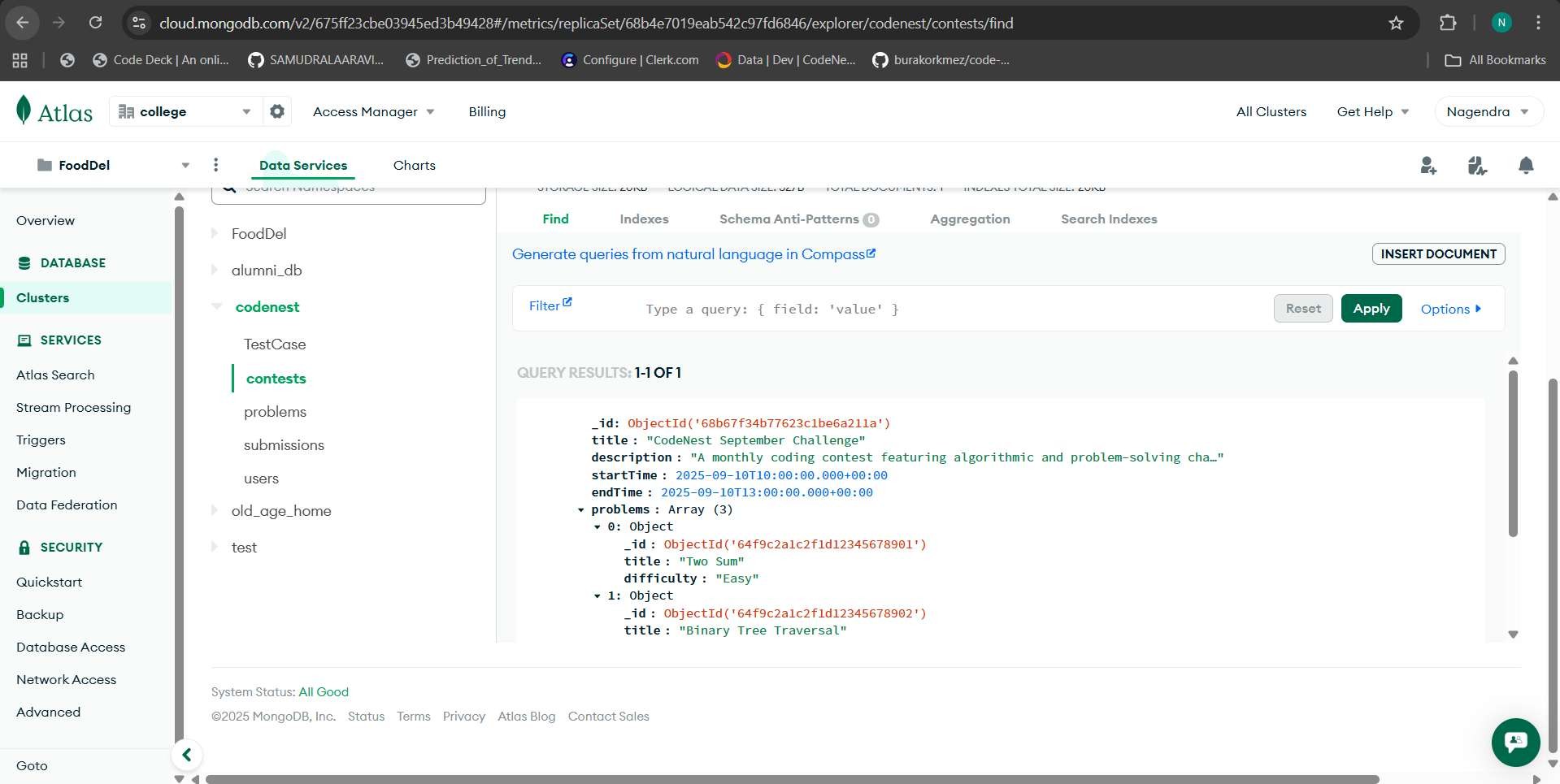


Figure 3.3 This Figure illustrates the contests document from the codenest collection

#### Workflow Diagram

The user workflow in the project, illustrated in Figure 3.4, begins with the process of registration and login, after which the user gains access to the appropriate dashboard (either User or Admin). From the dashboard, a user can explore and take part in coding contests, attempt individual problems, and submit their code solutions. Every submission is automatically routed to the Judge0 evaluation engine, which compiles and executes the code before returning the results. These outcomes are then displayed in the results and leaderboard section, enabling users to monitor their performance and compare it with others. Finally, the session concludes when the user logs out of the system.

The admin workflow in the system, shown in Figure 3.4, begins with the admin logging in and accessing the Admin Dashboard, which acts as the central hub for all management activities. From this dashboard, the admin can create, update, or remove contests, manage problem statements by adding or editing challenges, and review submissions made by users. The leaderboard is also accessible here, allowing the admin to track overall performance and monitor user progress. Once administrative tasks are completed, the workflow ends with the admin logging out, marking the close of the session.

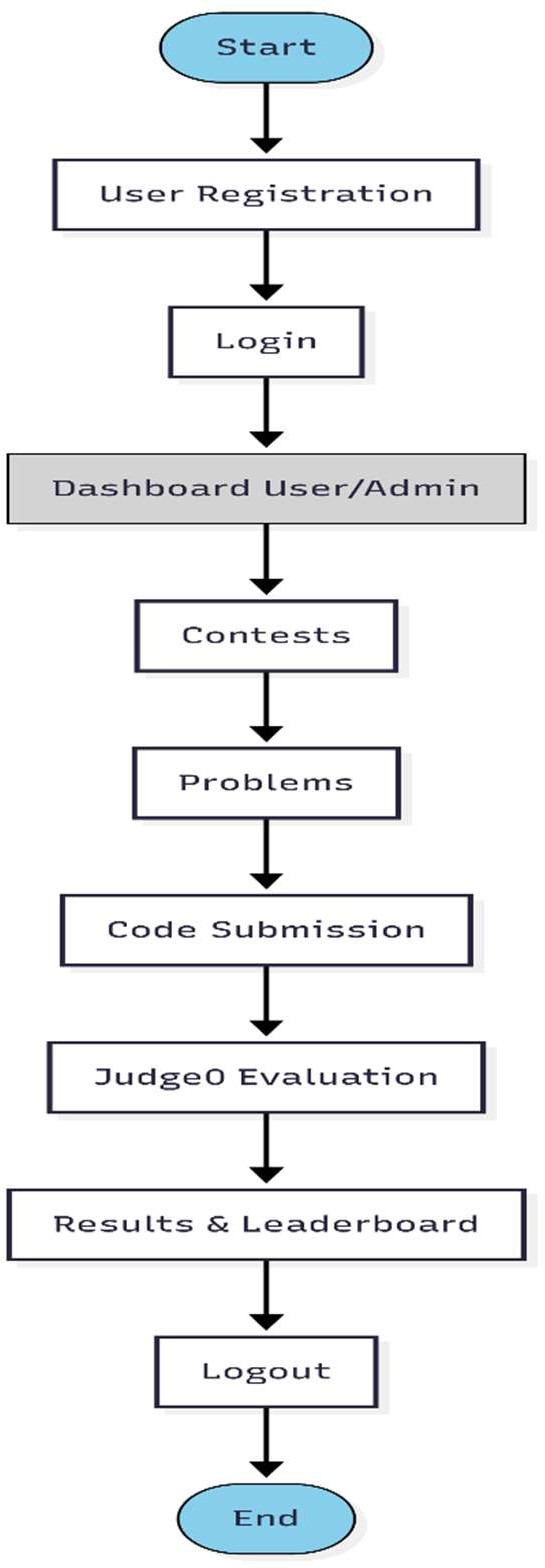


Figure 3.4 Workflow diagram for user

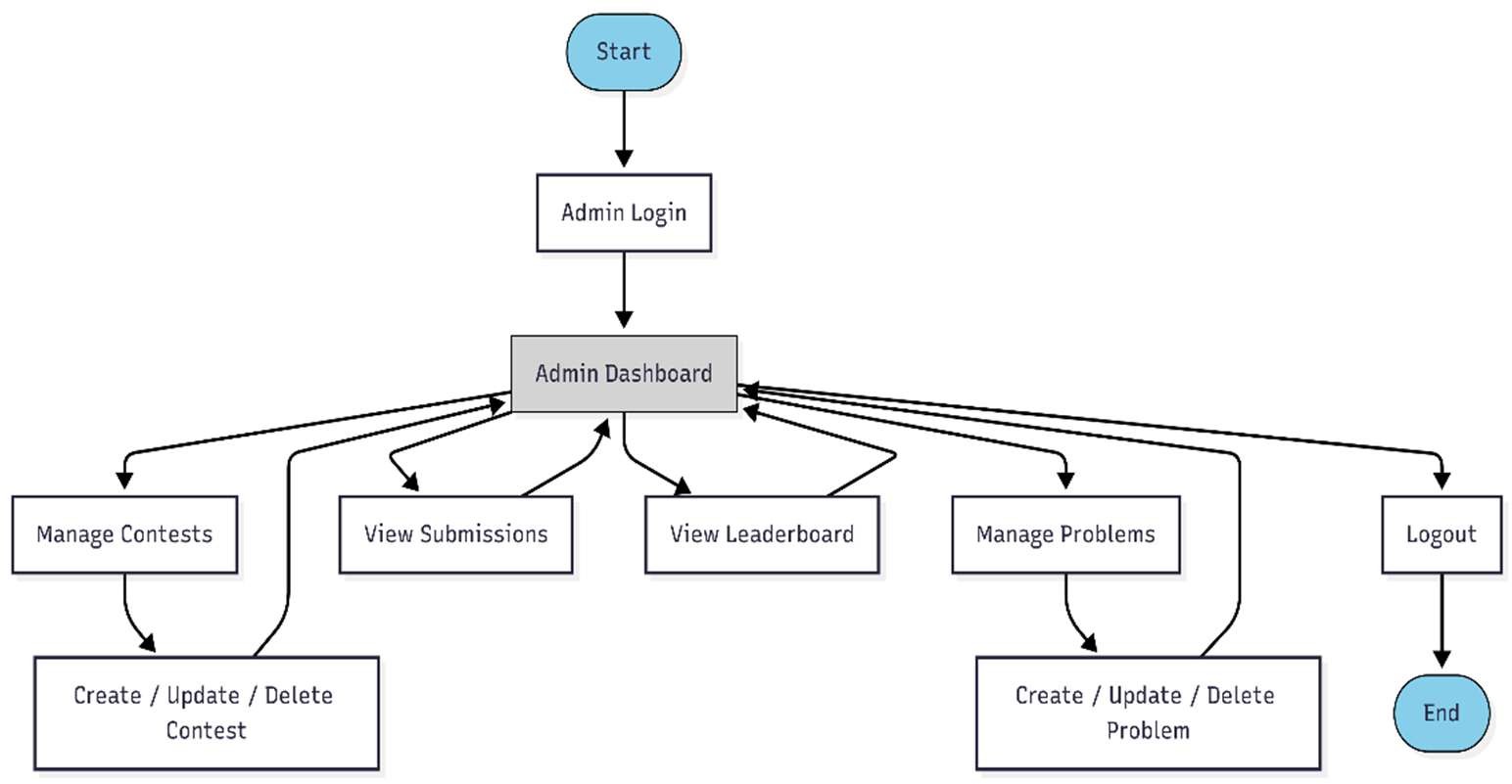


Figure 3.5 Workflow diagram for admin

#### Work Break Down Structure and Project Timeline

The work breakdown structure and project timeline for the project is meticulously organized and presented through a comprehensive table and an accompanying Gantt chart, as depicted in Table 3.1 and Figure 3.6. These tools are instrumental in mapping out the sequence of tasks, the duration of each phase and the interdependencies among various activities. The Gantt chart offers a visual overview of the project’s progress, ensuring that all tasks are on schedule and that resources are optimally allocated. Meanwhile, the table breaks down the project into manageable tasks, facilitating easier tracking and efficient management**.**

Table 3.1 Detailed Planning of Project

|  |  |  |
| --- | --- | --- |
| **Task Id** | **Tasks** | **Duration** |
| 1 | Requirement Gathering & Analysis | 20days |
| 2 | Literature Review & Background Study | 15days |
| 3 | System Design & Architecture Planning | 10days |
| 4 | Database & Data Source Identification | 9days |
| 5 | Backend Module Development (APIs) | 9days |
| 6 | Database Schema Design & Implementation | 5days |
| 7 | Judge0 Integration | 6days |
| 8 | User Interface Design (User/Admin Dashboards) | 4days |
| 9 | Module Integration (Frontend + Backend) | 5days |
| 10 | Testing & Quality Assurance | 5days |
| 11 | Deployment & Launch Preparation | 5days |
| 12 | Final Review & Documentation | 25days |
| 13 | Project Release | 1day |

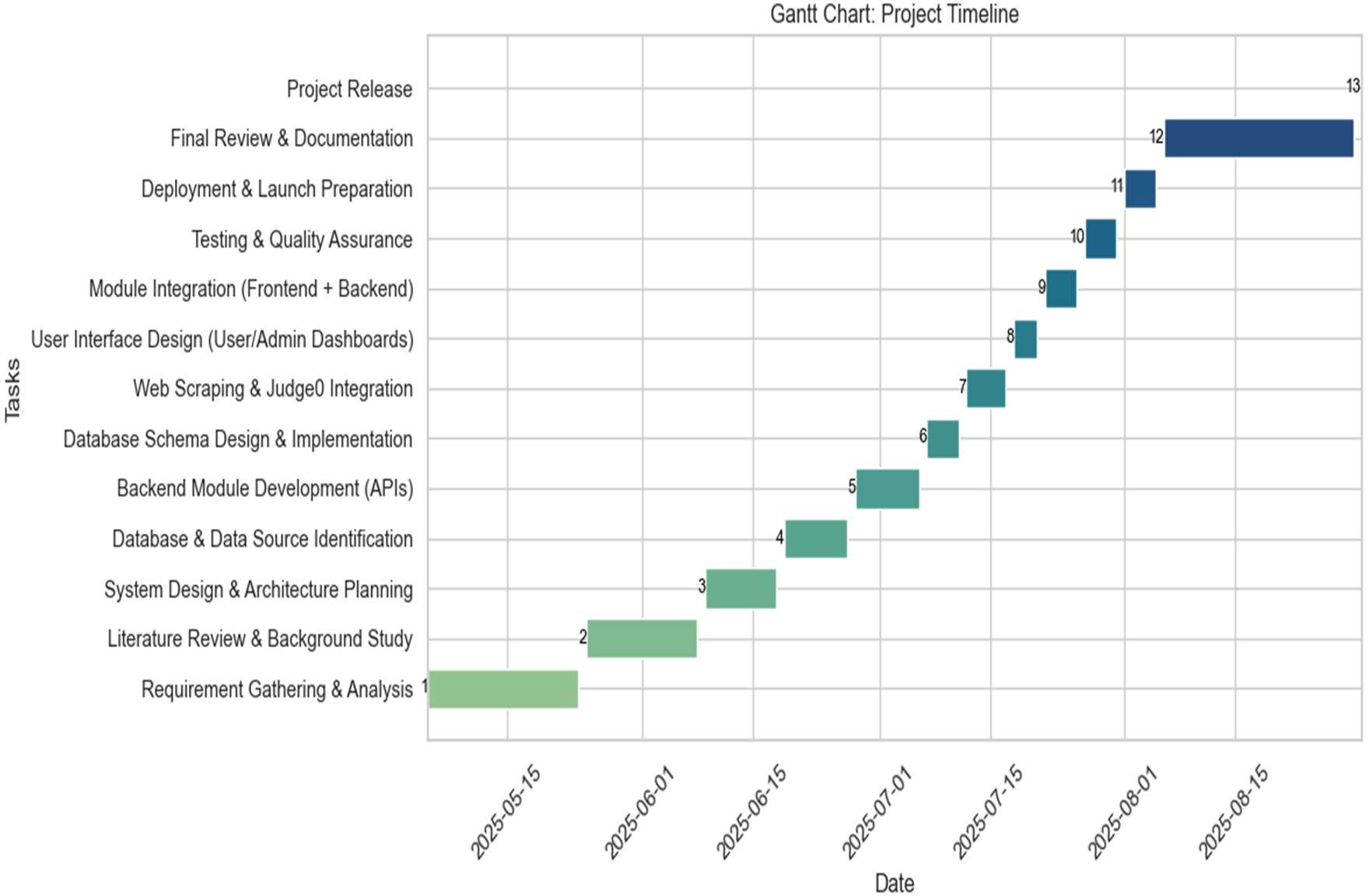


Figure 3.6 Gantt Chart Depicting the Project Timeline and Task Sequence for the project

#### Hardware Requirements

* + - * 1. Processor: Intel(R) Core (TM) i5-10500H CPU @ 2.50GHz (2.50 GHz) or higher
        2. RAM: Minimum 8 GB (7.45 GB usable), ideally 16 GB or more to handle Docker containers and multiple services smoothly
        3. Storage: 256 GB SSD for development, preferably 512 GB SSD or higher for deployment to ensure faster data access and scalability
        4. GPU: A dedicated GPU with at least 4 GB VRAM is recommended for improved performance during heavy operations and future AI/ML integration
        5. Network: Stable broadband internet connection with a minimum speed of 10 Mbps

#### Software Requirements

* + - * 1. **Node.js & Express.js**: Used as the backend framework to build a robust and scalable server-side application. It handles RESTful APIs, authentication, and communication between the frontend and the database efficiently.
        2. **MongoDB Atlas**: Serves as the NoSQL cloud-based database for storing users, contests, problems, and submissions in a flexible schema. Its JSON-like structure allows easy storage and retrieval of hierarchical and nested data.
        3. **React.js with JavaScript**: Powers the frontend user interface, offering a responsive and dynamic experience. React.js enables seamless rendering of pages such as login, dashboard, problems, contests, and leaderboard for both users and admins.
        4. **JWT (JSON Web Token) & Bcrypt**: Ensures secure authentication and authorization within the system. JWT manages user sessions, while Bcrypt hashes Passwords to protect sensitive user credentials.
        5. **Postman**: Utilized for testing API endpoints during development, verifying that routes for user authentication, contest management, and problem submissions work correctly and securely.
        6. **MongoDB Compass**: Provides a graphical interface to manage and visualize the MongoDB database, enabling schema design, query execution, and real-time monitoring of collections such as users, contests, and problems.
        7. **Development Environments**: Visual Studio Code is the primary IDE used for writing, debugging, and managing both frontend and backend codebases. Docker Desktop is used for containerization, ensuring smooth deployment of all services.

#### Functional & Non-Functional Requirements

1. **Functional Requirements**

The project is composed of several key modules, each designed to provide a seamless and secure experience for both users and administrators while managing coding activities.

#### User Management:

* + 1. User Registration and Login: Secure registration and authentication for users, including Admin and general user roles.
    2. Profile Management: Allows users to update personal details, while admins can manage user accounts and assign roles.

#### Problem Management:

* + 1. Admins can create, edit, and delete coding problems, including details such as title, description, difficulty level, input/output format, and test cases.
    2. Problems are dynamically displayed to users in the dashboard for practice and contests.

#### Contest Management:

* + 1. Admins can create, update, and delete contests, specifying contest titles, descriptions, start and end times, and assigning problems.
    2. Users can view upcoming and active contests, participate in them, and attempt the associated problems.

#### Submission & Evaluation Module:

* + 1. Users can submit code solutions which are automatically sent to the Judge0 API for compilation and execution.
    2. The system evaluates submissions against visible and hidden test cases, returning results such as Accepted, Wrong Answer, or Runtime Error.

#### Leaderboard and Performance Tracking:

* + 1. A leaderboard displays rankings based on contest participation and problem-solving performance.
    2. Users can track their progress through submission history and results.

#### Role-Based Access Control:

* + 1. Differentiates access levels so that Admins can manage problems, contests, and users comprehensively, while general users have access only to solving problems and participating in contests.
    2. Ensures secure and restricted access to sensitive administrative functions.

#### Non-Functional Requirements

Several qualities are emphasized in project to ensure high performance, user satisfaction, and long-term sustainability of the platform.

#### User Interface:

* + 1. The interface is designed to be responsive and user-friendly, ensuring smooth navigation on desktops, laptops, tablets, and mobile devices.
    2. A clean and intuitive layout allows both beginners and experienced coders to access problems, contests, and results with minimal effort.

#### Performance:

* + 1. Code submissions and evaluations are processed in real time through Judge0, providing immediate feedback to users.
    2. The platform is built with scalability in mind, so it can handle growing numbers of users, contests, and submissions without slowing down.

#### Security and Privacy:

* + 1. Sensitive data such as passwords and user details are encrypted, while secure session management protects users during login and usage.
    2. Strong authentication mechanisms and role-based access control prevent unauthorized access to admin features and ensure user data privacy.

#### System Reliability:

* + 1. The system ensures high availability with minimal downtime, allowing uninterrupted participation in contests and problem-solving.
    2. Robust error handling and logging mechanisms are implemented to quickly detect, trace, and resolve issues, keeping the system dependable.

#### Maintainability:

* + 1. The project uses a modular architecture and well-documented codebase, making future maintenance, debugging, and upgrades easier.
    2. Its flexible design supports the addition of new features such as discussion forums, advanced analytics, or mobile app integration without major rework.

#### System Design

* + - 1. **ER Diagram**

This diagram illustrates the entity relationships within the project, as shown in Figure 3.7. The main entities in the system include User, Testcase, Submission, Problem, Contest, ContestProblem.

The database design for the project has been structured to ensure smooth handling of user participation, contests, problems, and submissions while maintaining scalability and efficiency. At the core of the system is the User entity, which represents every participant or administrator on the platform. Each user has attributes such as name, email, password, role, and timestamps to distinguish between general users and administrators. From a relational perspective, a single User can enroll in multiple contests, forming a 1:N relationship between Users and Contests.

The Contest entity captures information about coding competitions, including the contest title, description, start and end times, and a collection of linked problems. A contest can include many problems, while a single problem can also be reused across multiple contests, thereby forming an M:N relationship between Contests and Problems. This flexible design ensures that problems do not have to be recreated for each contest, reducing redundancy.

The Problem entity is central to the coding practice environment. It includes attributes such as the problem title, detailed description, difficulty level, input and output formats, constraints, and test cases. Each problem can be attempted by many users through their submissions, which leads to a 1:N relationship between Problems and Submissions.

The Submission entity records each attempt made by users. It stores details like the associated problem ID, user ID, the submitted code, chosen programming language, results from the Judge0 evaluation engine, and timestamps. Every Submission belongs to exactly one User and one Problem, establishing N:1 relationship with both Users and

Problems. This ensures that submissions are always traceable back to the user who made them and the problem they attempted.

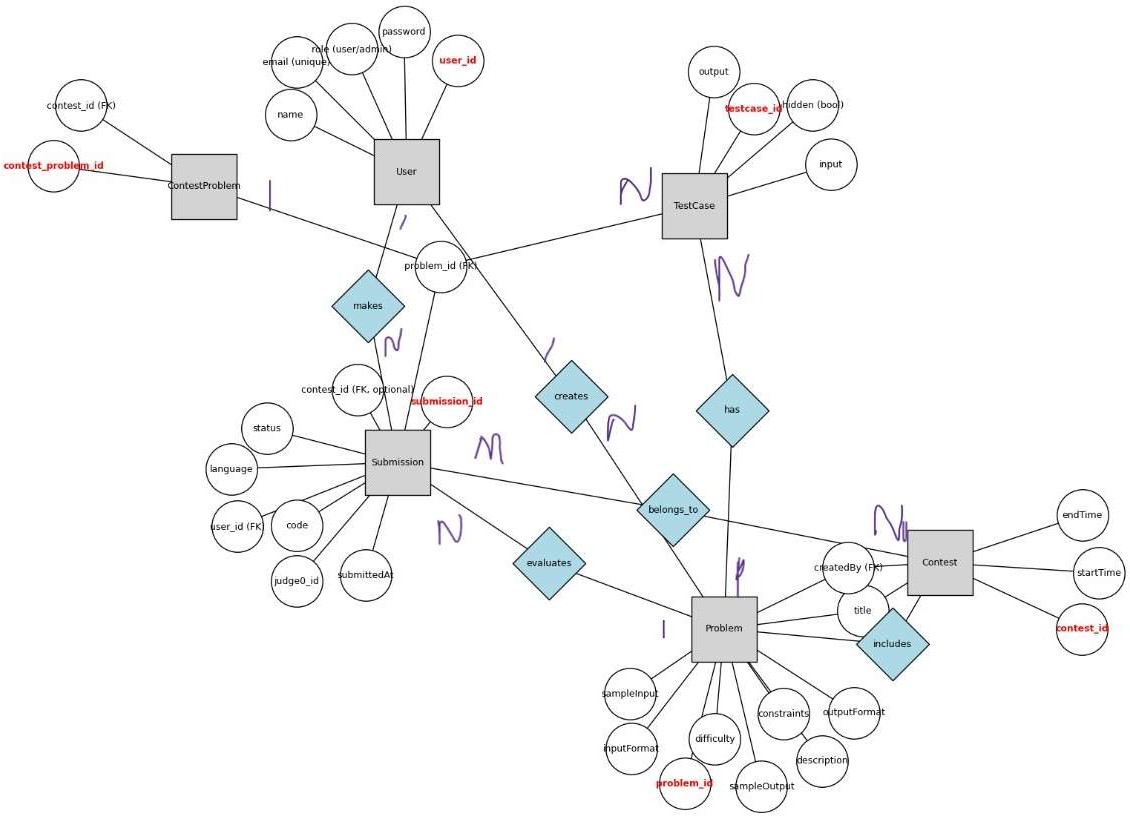


Figure 3.7 Entity Relationship Diagram

Overall, this design guarantees consistency across the system: Users participate in Contests, Contests contain Problems, and Submissions connect Users and Problems. With

cardinality ratios explicitly defined, the structure provides an efficient way to manage relationships while supporting scalability for future features such as leaderboards and advanced analytics.

#### Use Case Diagram

The following diagram for the project illustrates the interactions between the primary users (Admin and Client) and the system’s core functionalities, as shown in Figure 3.8.

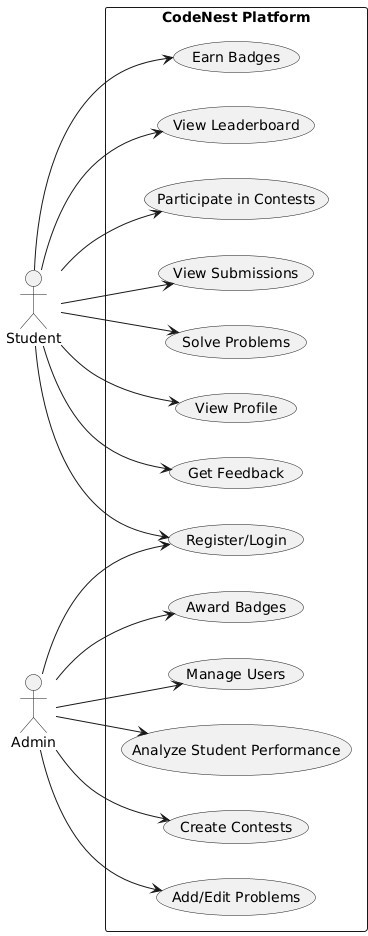


Figure 3.8 Use Case Diagram

The project brings together two types of users students and administrators each with their own set of responsibilities that keep the system balanced and effective. For students, the platform is a space to learn, practice, and compete. They begin by registering or logging in, after which they can participate in coding contests, attempt different problems, and track their progress through submissions. The platform also provides leaderboards to encourage healthy competition, badges to reward achievements, and personalized profiles where students can view their growth. Additionally, feedback features help learners identify areas for improvement, making the system more engaging and supportive of continuous learning.

Administrators, on the other hand, focus on managing and maintaining the platform’s operations. Their role includes creating contests, adding or editing problems, and monitoring user accounts to ensure smooth participation. They also evaluate student performance by analyzing submissions and results, while awarding badges as a way to motivate learners. By overseeing these activities, admins not only maintain order but also nurture a fair and enriching learning environment.

Together, this division of responsibilities ensures that students concentrate on skill development and competition, while administrators handle the behind-the-scenes tasks of content management and performance tracking. The result is a well-structured system where learning, assessment, and recognition are seamlessly integrated, fostering both growth and engagement within the coding community.

#### Activity Diagram

The visual representation below illustrates the typical workflow for users interacting with the coding practice platform, as depicted in Figure 3.9. The process begins with either the admin or user, logging into the system, followed by validation of their login credentials.

When an admin logs into the system successfully, they are directed to the dashboard, which acts as the central hub for all management activities. From here, the admin can create and update product categories, add new items to the inventory, and manage user accounts. They also have the ability to monitor orders, view order history, and ensure that the overall store operations remain organized and efficient. If the login credentials are

incorrect, the system immediately alerts the user and prevents access until valid details are provided.

For clients, a successful login opens their personal dashboard, where they can explore available products, place new orders, and check their past purchases. The steps are designed to be simple and intuitive, beginning with login, followed by the main activities, and ending with logout to close the session securely.

This workflow ensures that both admins and clients interact with the system in a way that suits their roles. Admins focus on managing the backend operations, while clients concentrate on browsing and ordering products. Together, this role-based structure creates a smooth, secure, and reliable billing process from start to finish.

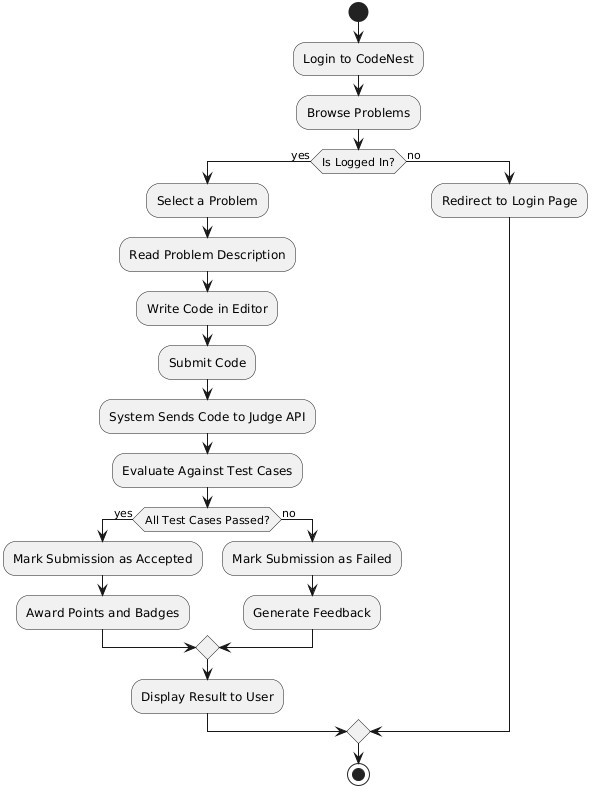


Figure 3.9 Activity Diagram

#### Sequence Diagram

This sequence diagram shows the interactions between User, Admin, Frontend (React), Backend (Node/Express), Database (MongoDB), and Judge0 as depicted in Figure 3.10. It includes the main flows: user registration/login, solving problems, submitting code, and admin managing contests/problems.

The sequence diagram begins with user registration and login. A new user provides their details, which are validated by the backend and stored in the database. Existing users log in with their credentials, and upon successful authentication, a session token is issued. This process ensures secure access to the system for both users and admins.

After login, users can view problems or contests through the frontend. The frontend communicates with the backend, which fetches the relevant data from the database. When a user submits code for a problem, the backend sends it to the Judge0 API for execution. The results are returned, stored in the database, and displayed back to the user along with leaderboard updates.

Admins follow a similar login process but are given elevated privileges. Once authenticated, they access the admin dashboard where they can manage problems and contests. Each create, update, or delete operation is validated by the backend and reflected in the database, ensuring data integrity. Admins can also review submissions and monitor leaderboards.

The diagram also highlights the interaction between different system components. The frontend handles user interaction, the backend manages business logic, the database ensures persistent storage, and Judge0 provides external code execution services. This collaboration ensures real-time feedback and smooth system performance.

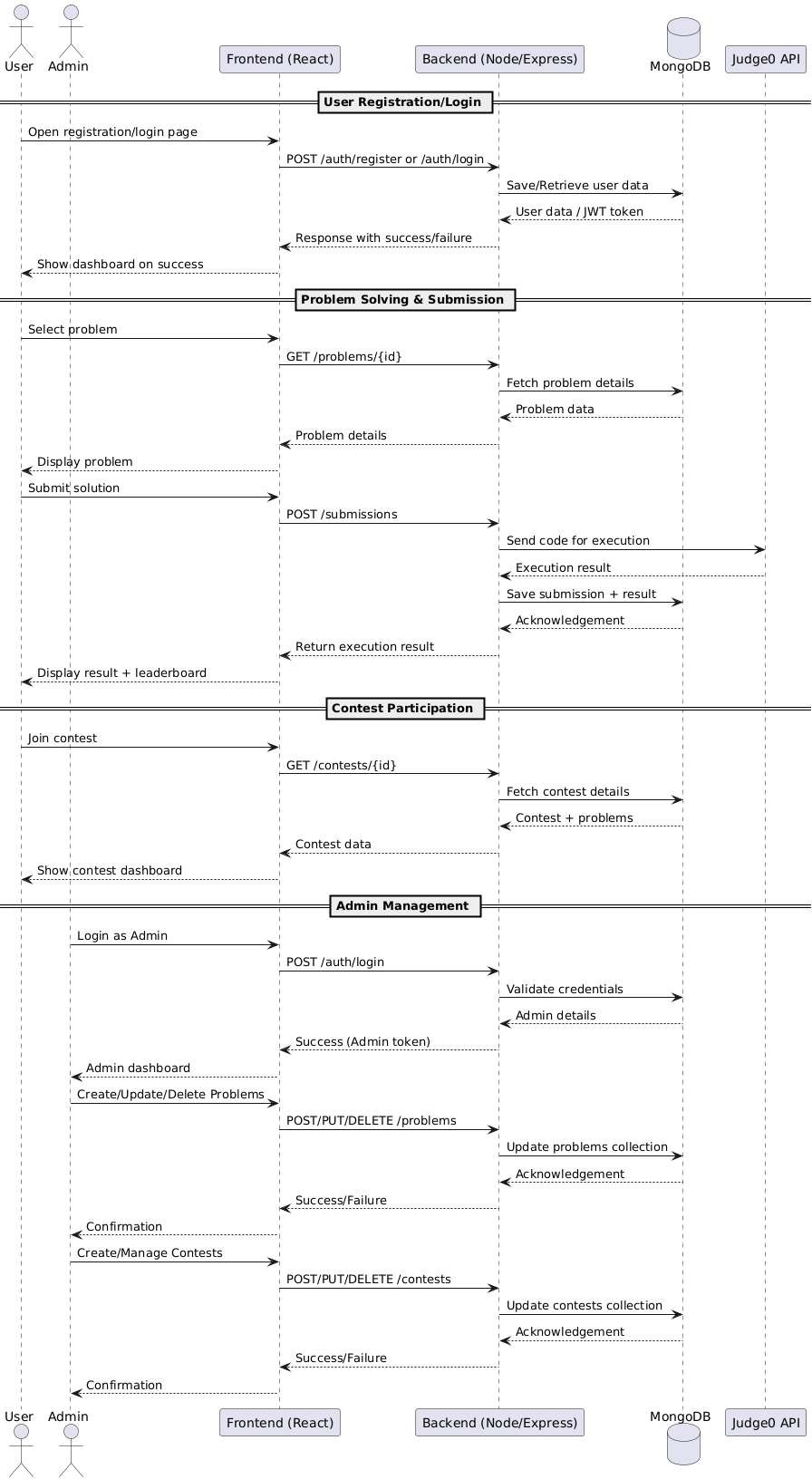


Figure 3.10 Sequence Diagram

Finally, both users and admins can securely log out, ending their session by clearing tokens and closing access to protected routes. This completes the flow, showing how the system maintains secure, role-based operations while supporting coding, contests, and administrative management.

#### Implementation, maintenance, and support

The implementation phase of this project marked the transition from design to a fully operational coding and contest management platform. At this stage, the system’s modules were coded, integrated, and tested to create a smooth and functional workflow for both users and administrators. The primary objective was to bring together user registration, problem-solving, contest participation, and code evaluation into one seamless application.

The system was developed using React (with Vite) for the frontend and Node.js with Express for the backend. React ensured a dynamic and responsive user interface, enabling users to navigate easily between dashboards, contests, problems, and leaderboards. On the server side, Express handled all business logic, API requests, authentication, and secure communication between different modules.

A central part of the implementation was integrating Judge0, which served as the code execution engine. User submissions were processed in real-time by Judge0, and results such as passed test cases, errors, or timeouts were sent back to the system. The backend stored all these results in MongoDB, which acted as the main database, storing users, problems, contests, and submission records. The schema was defined and managed through Mongoose, making data handling structured and reliable.

For security and authentication, JWT (JSON Web Tokens) and bcrypt were used. This ensured that login sessions were protected and passwords were securely stored in the database. An admin panel was implemented to allow administrators to create, update, and delete problems and contests. Meanwhile, the notification and feedback system was designed to keep users updated about contest schedules and submission results.

Deployment was managed using Docker and Docker Compose, which allowed all services (frontend, backend, database, and Judge0) to run in containers for easy scalability and cross-platform compatibility. Regular monitoring and testing were carried out to identify potential issues early and to keep the system stable. Maintenance plans were established to keep the system updated with bug fixes, security patches, and feature improvements.

This phase ensured the system moved from blueprint to reality, providing an accessible, reliable, and secure platform where users could practice coding, participate in contests, and receive instant feedback on their work.

#### Testing and Validation

The testing and validation phase of the project was crucial to ensure that the system performed as expected and met the defined requirements. A manual testing approach was adopted, where testers interacted with the application directly to simulate real user behavior. This method allowed for a thorough evaluation of both the functional and non- functional aspects of the system, including usability, responsiveness, and reliability.

Test cases were carefully designed to cover all the major components of the project. These included verifying user registration and login, validating contest and problem management, checking code submissions through the Judge0 integration, and ensuring leaderboard updates were accurate. The process also assessed database consistency, system security features, and the smooth navigation across the user and admin dashboards.

By executing these scenarios, testers were able to uncover and resolve defects early, validate data flow between modules, and confirm that performance remained stable under varying conditions. This phase ensured the system provided a reliable and user-friendly experience before deployment.

Table 3.2 Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Web Page** | **Task** | **Expected Result** | **Status** |
| TC\_01 | Login Page | Admin logs in with valid credentials | Admin is successfully authenticated and redirected to the Admin Dashboard. | Pass |
| User logs in with valid credentials | User is authenticated and redirected to the User Dashboard. | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TC\_02 | Registration Page | New user registers with valid details | User account is created successfully and can log in afterwards. | Pass |
| TC\_03 | Manage Users (Admin) | Admin adds, updates, or deletes a user | Users are managed successfully, and changes reflect immediately. | Pass |
| TC\_04 | Manage Contests (Admin) | Admin creates, updates, or deletes contests | Contests are correctly created/updated/deleted and visible to users. | Pass |
| TC\_05 | Manage Problems (Admin) | Admin adds, updates, or deletes coding problems | Problems are saved successfully and associated with contests if linked. | Pass |
| TC\_06 | Contest Page (User) | User views contest details | Contest information (title, description, start & end times, list of problems) is displayed correctly. | Pass |
| TC\_07 | Problem Page (User) | User attempts a coding problem | Problem statement, constraints, and sample input/output are displayed correctly. | Pass |
| TC\_08 | Code Submission | User submits a solution to a problem | Code is sent to Judge0 for evaluation and results (Accepted, Wrong Answer, Error) are shown. | Pass |
| TC\_09 | Leaderboard | User views leaderboard | Leaderboard displays scores/rankings based on submissions and contest rules. | Pass |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TC\_10 | Dashboard (User/Admin) | User/Admin views dashboard | Dashboard shows relevant data: contests, problems, submissions, and performance summaries. | Pass |
| TC\_11 | Logout | User/Admin logs out | User/Admin is securely logged out and redirected to the login page. | Pass |

# CHAPTER 4 RESULTS AND DISCUSSIONS

The Web-Based Coding Practice Platform has been developed using Node.js and Express for the backend and React.js with Vite for the frontend, while MongoDB serves as the primary database for managing data. The system integrates with Judge0 API to provide real-time code execution and evaluation. It consists of several core modules: User Management, Problem & Contest Management, Submissions & Evaluation, and Dashboard with Leaderboard.

In the Problem & Contest Management module, administrators can create, update, and delete coding problems and organize them into contests. Each problem includes detailed descriptions, input/output formats, constraints, and test cases, enabling students to practice and compete in a structured environment. Contests allow timed challenges, promoting competitive programming and problem-solving skills.

The Submissions & Evaluation module enables students to write and submit code directly on the platform. Submissions are automatically compiled and executed via Judge0, and results such as Accepted, Wrong Answer, or Compilation Error are instantly displayed. This feature ensures immediate feedback, helping students learn from their mistakes and improve their coding abilities

The User Management module allows for secure registration and login with role-based access. Students can register, log in, and attempt problems, while admins manage the platform by overseeing users, contests, and submissions. This ensures a controlled and organized coding environment.

The Dashboard and Leaderboard module provides students with an overview of their coding progress, including recent submissions, contest participation, and performance tracking. The leaderboard ranks students based on problem-solving accuracy and contest scores, encouraging healthy competition and continuous improvement.

This integrated system streamlines the process of coding practice by combining learning, evaluation, and competition into a single platform. By offering real-time feedback,

structured contests, and performance insights, the solution empowers MCA students to strengthen their programming skills, preparing them for both academic success and professional challenges in the software industry.

### 4.1 Snapshots

#### Admin Side i. Admin Login

The Admin login page serves as the entry point for admins. Admins are required to enter their email and password to securely access the system. Upon successful authentication, the interface directs admin to the dashboard, ensuring system security and role-based feature access.

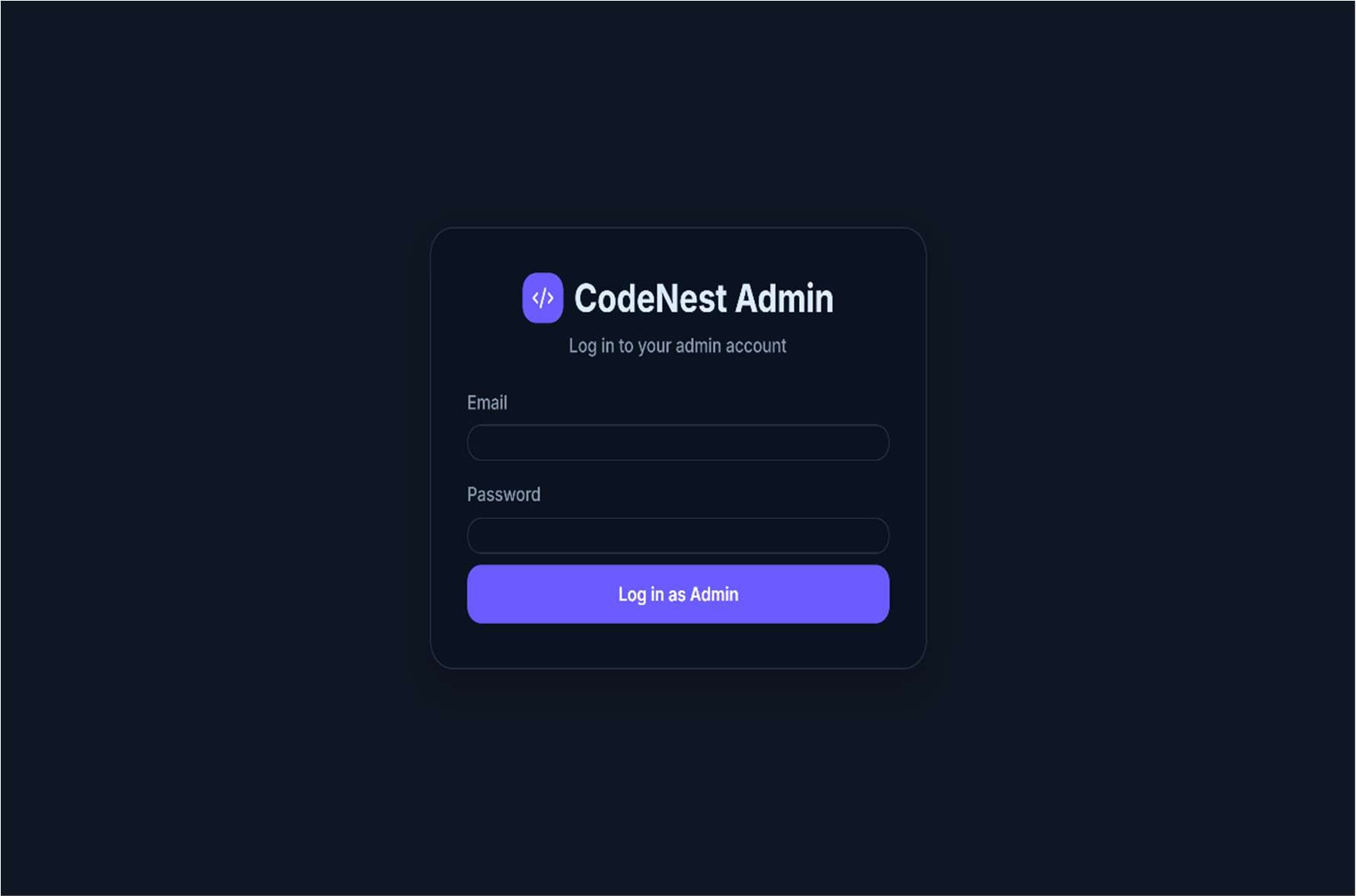


Figure 4.1 Login page for Admin

In this admin login page, as shown in Figure 4.1, admins are prompted to enter their email address and password to securely access the system. After successful authentication, Admin can proceed to utilize the features of admin like managing the problems, contests and users.

#### Admin Dashboard

This is the Admin Dashboard of the platform. It gives a quick overview of the system with key stats like total students, active contests, and projects. The panel on the left allows the admin to navigate to contests, leaderboard, users, and settings. In the center, the dashboard highlights recent activities such as new contests, student registrations, project submissions, and feedback updates. It also shows when each activity took place, keeping the admin up to date. Overall, it serves as the control hub for monitoring and managing the platform.

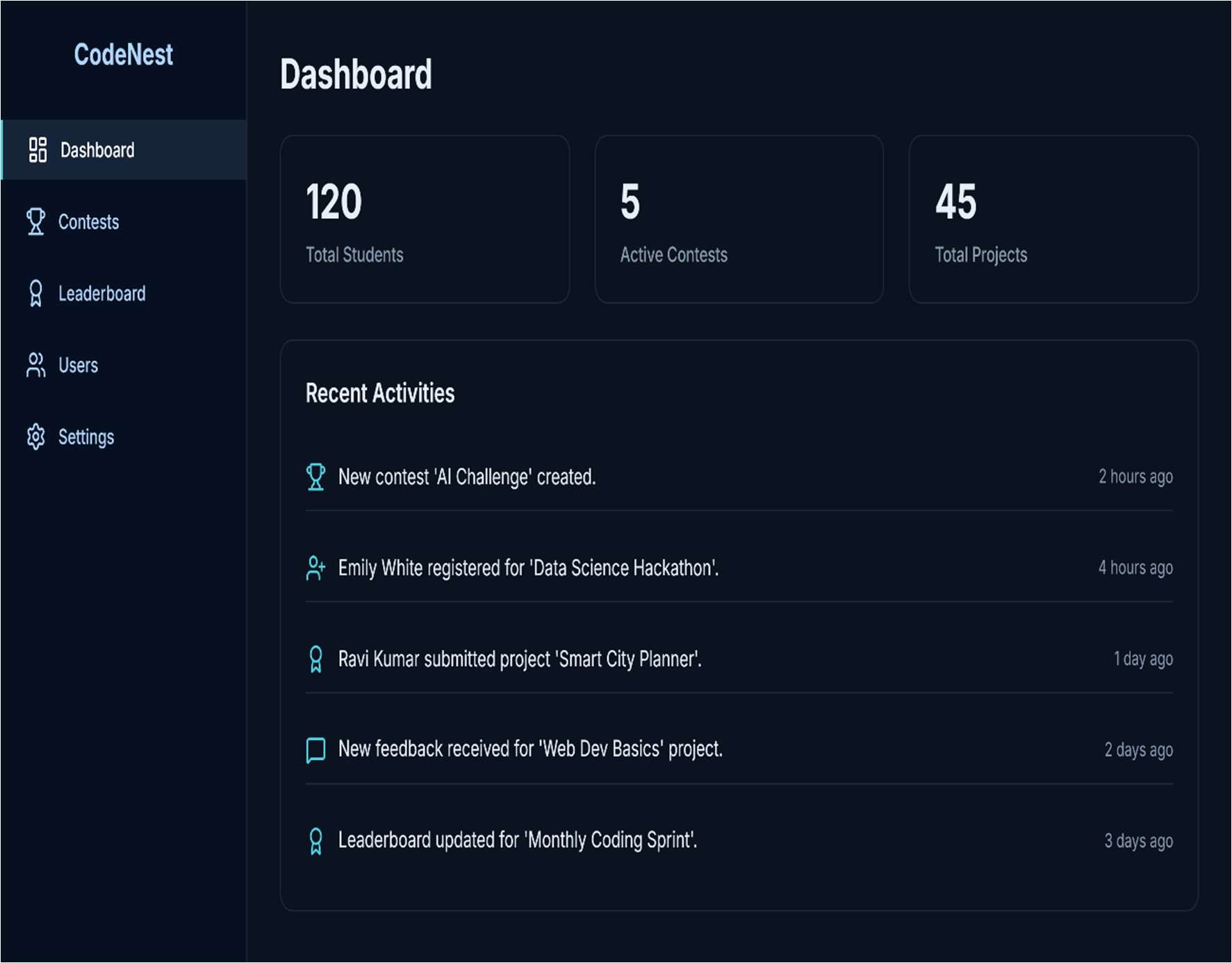


Figure 4.2 Admin Dashboard

In this admin dashboard page, as shown in Figure 4.2, the admin is provided with an overview of the platform through key statistics such as the total number of students, active contests, and total projects. The dashboard also highlights recent activities, including new contest creation, student registrations, problem submissions, and feedback updates. From here, the admin can easily navigate to different sections like contests, leaderboard, users, and settings to efficiently manage and monitor the platform.

#### Contest Management Page

This is the Contest Management page of the platform. Here, the admin can see all contests along with their status, such as active, completed, or upcoming. Each contest shows its start and end dates for better tracking. The admin also has options to edit or delete contests directly from this page. On the top right, there is a button to create a new contest. Overall, it helps the admin easily manage and organize coding competitions.

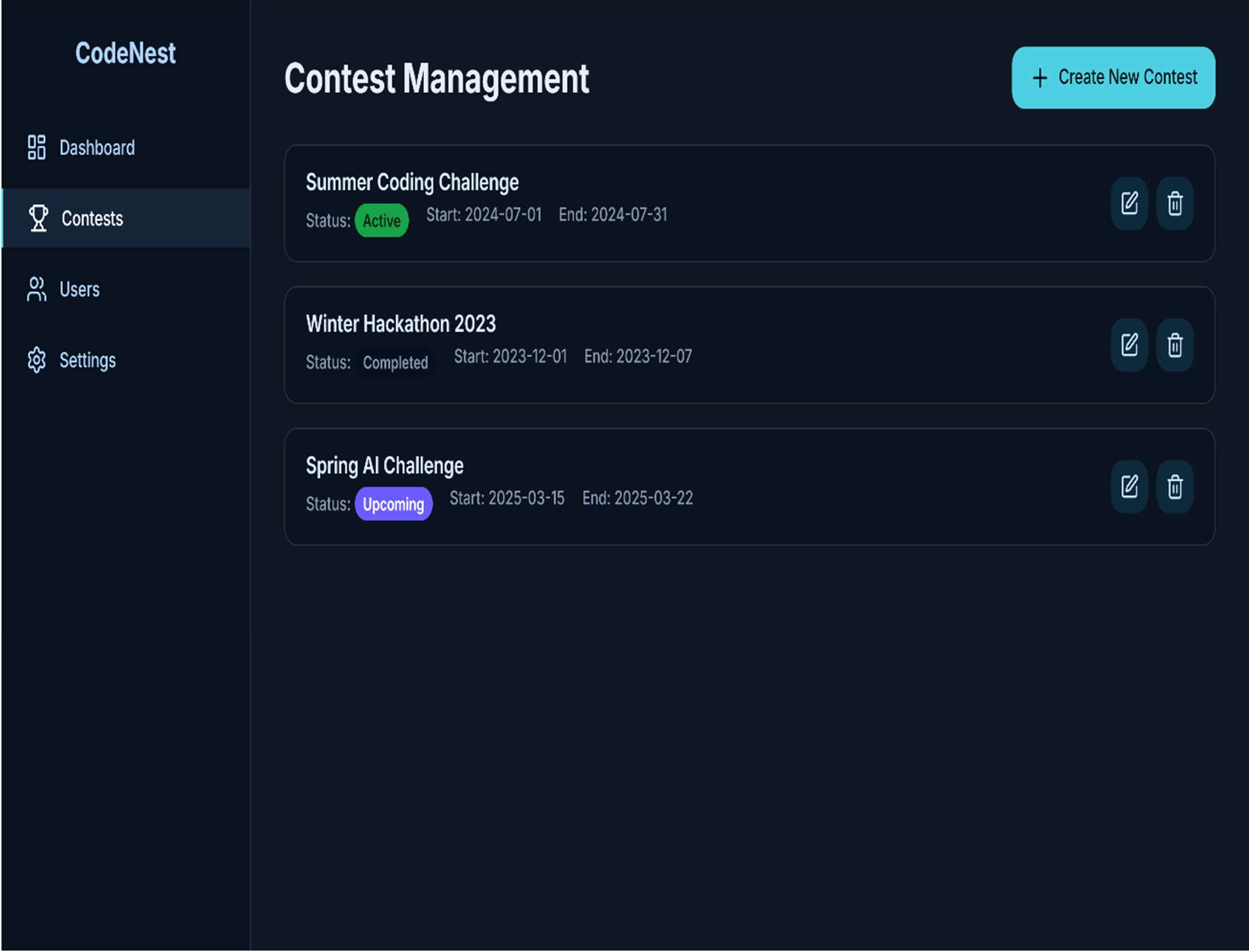


Figure 4.3 Contest Management Page

In this contest management page, as shown in Figure 4.3, the admin is able to view and manage all coding contests on the platform. The page displays contests with their status, such as active, completed, or upcoming, along with their start and end dates. Admins are provided with options to edit or delete contests for better control and updates. Additionally, a Create New Contest button is available, enabling admins to quickly set up new competitions. This page ensures smooth organization and efficient handling of contests within the platform.

#### Problem Management Page

This is the Problem Management page of the platform. It allows the admin to view, edit, or delete coding problems listed with details like ID, title, difficulty, tags, status, and last updated date. The search and filter options make it easier to find specific problems quickly. On the right side, there is a section to create or edit problems by adding titles, statements, examples, tags, and difficulty levels. Admins can also choose to save problems as drafts or publish them directly. This page helps in efficiently managing the problem database for contests and practice.

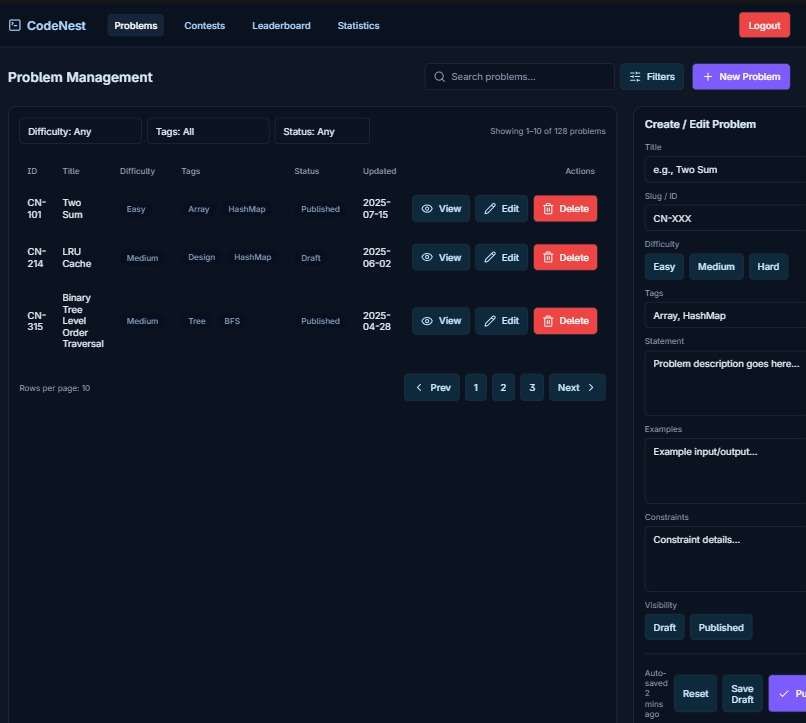


Figure 4.4 Problem Management Page

In this problem management page, as shown in Figure 4.4, the admin is provided with a complete view of all coding problems along with details such as problem ID, title, difficulty, tags, status, and last updated date. The page allows the admin to perform key actions like viewing, editing, or deleting problems directly from the list. On the right side, there is a section to create or edit problems by entering statements, examples, constraints, and tags, while also setting the difficulty level. The admin can save problems as drafts or publish them, ensuring smooth organization and management of the problem database within the platform.

#### Submission Page

This is the Submissions page of the platform. It displays all student submissions with details like submission ID, user name, problem attempted, contest, programming language, status, score, and execution time. The results are shown with clear labels such as Accepted, Wrong Answer, or Time Limit Exceeded. Admins can use filters to search by contest or problem, making it easy to track performance. Each submission also has a view option for deeper inspection. This page helps in monitoring and evaluating student solutions effectively.

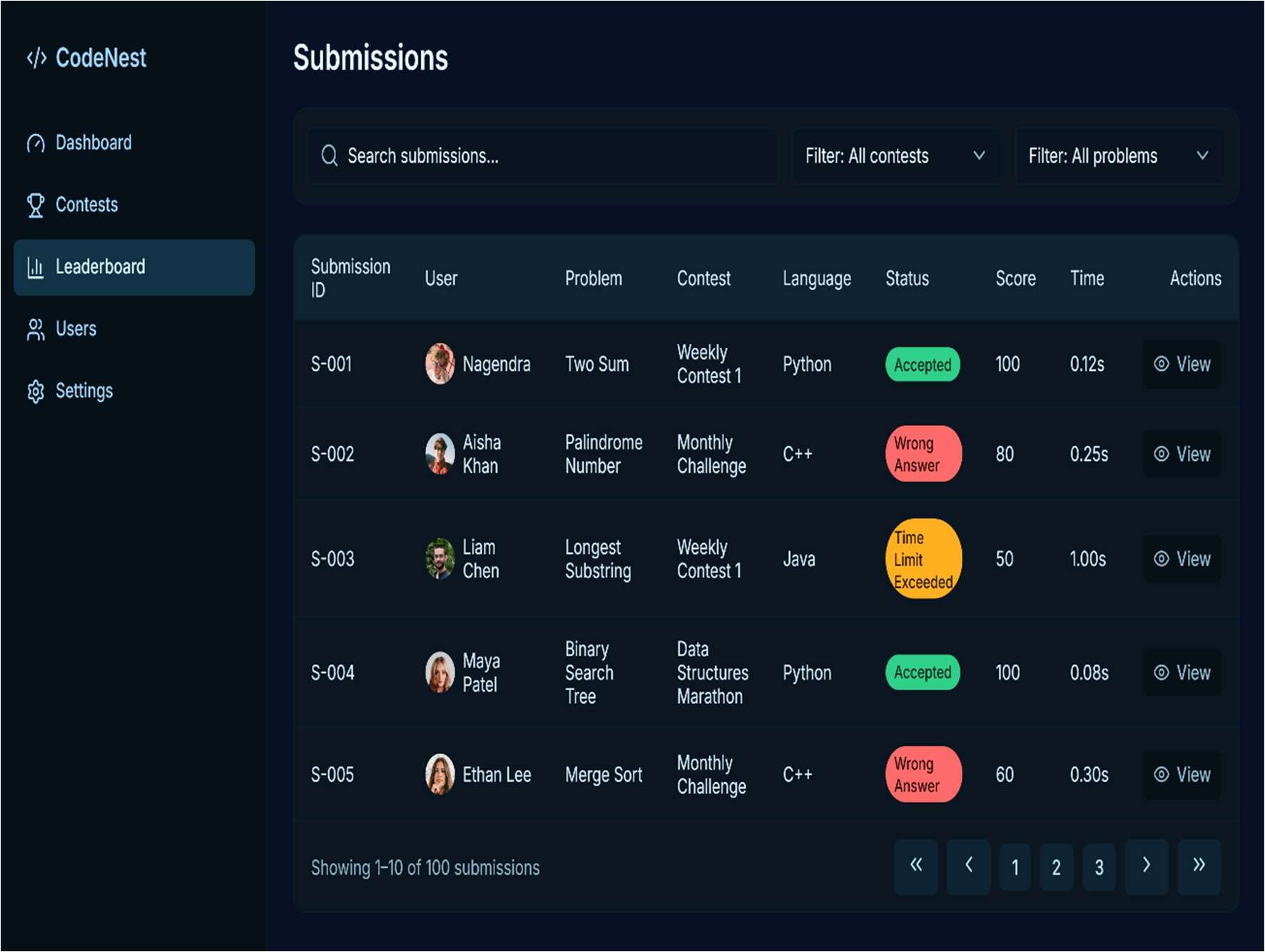


Figure 4.5 Submission Page of Admin

In this submissions page, as shown in Figure 4.5, the admin is provided with a detailed view of all student submissions across different contests. The page displays key information such as submission ID, user details, problem name, contest, programming language, status, score, and execution time. Each submission is clearly marked with results like Accepted, Wrong Answer, or Time Limit Exceeded for quick evaluation. The admin can also apply filters to search by contest or problem, and use the view option to inspect individual submissions. This ensures efficient tracking and monitoring of student performance on the platform.

#### Admin Profile Page

This is the Admin Profile page of the platform. It displays the admin’s details such as name, email, role, and a unique admin ID. The page also confirms that the admin has full privileges to manage users, problems, and platform settings. At the bottom, there are quick access buttons to manage users or manage problems. This page provides a simple way for admins to view their information and access key management tools.

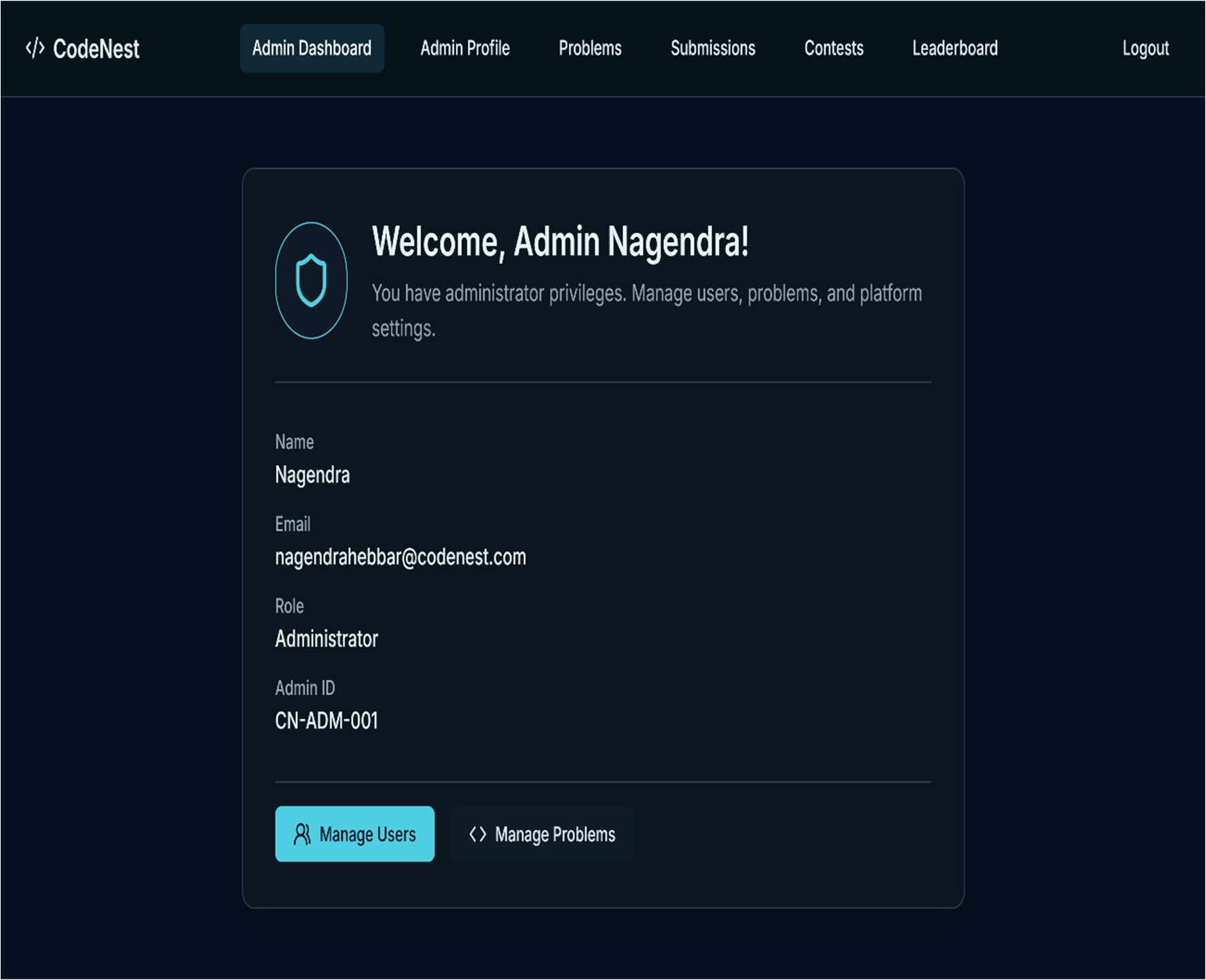


Figure 4.6 Admin Profile Page

In this admin profile page, as shown in Figure 4.6, the admin is presented with their personal and account details, including name, email, role, and a unique admin ID. The page also highlights the administrator’s privileges, allowing them to manage users, problems, and platform settings. Additionally, quick action buttons are provided to directly access user management and problem management features. This page ensures that the admin can easily review their information while seamlessly navigating to essential management tasks.

#### User Side

* 1. **Sign-Up Screen Page**

This is the Sign-Up page of the platform. It allows new users to create an account by entering their full name, email, password, and confirming the password. Once the details are filled in, users can click the Sign-Up button to register. There is also an option to log in if the user already has an account. This page ensures a simple and secure way for users to get started with the platform.

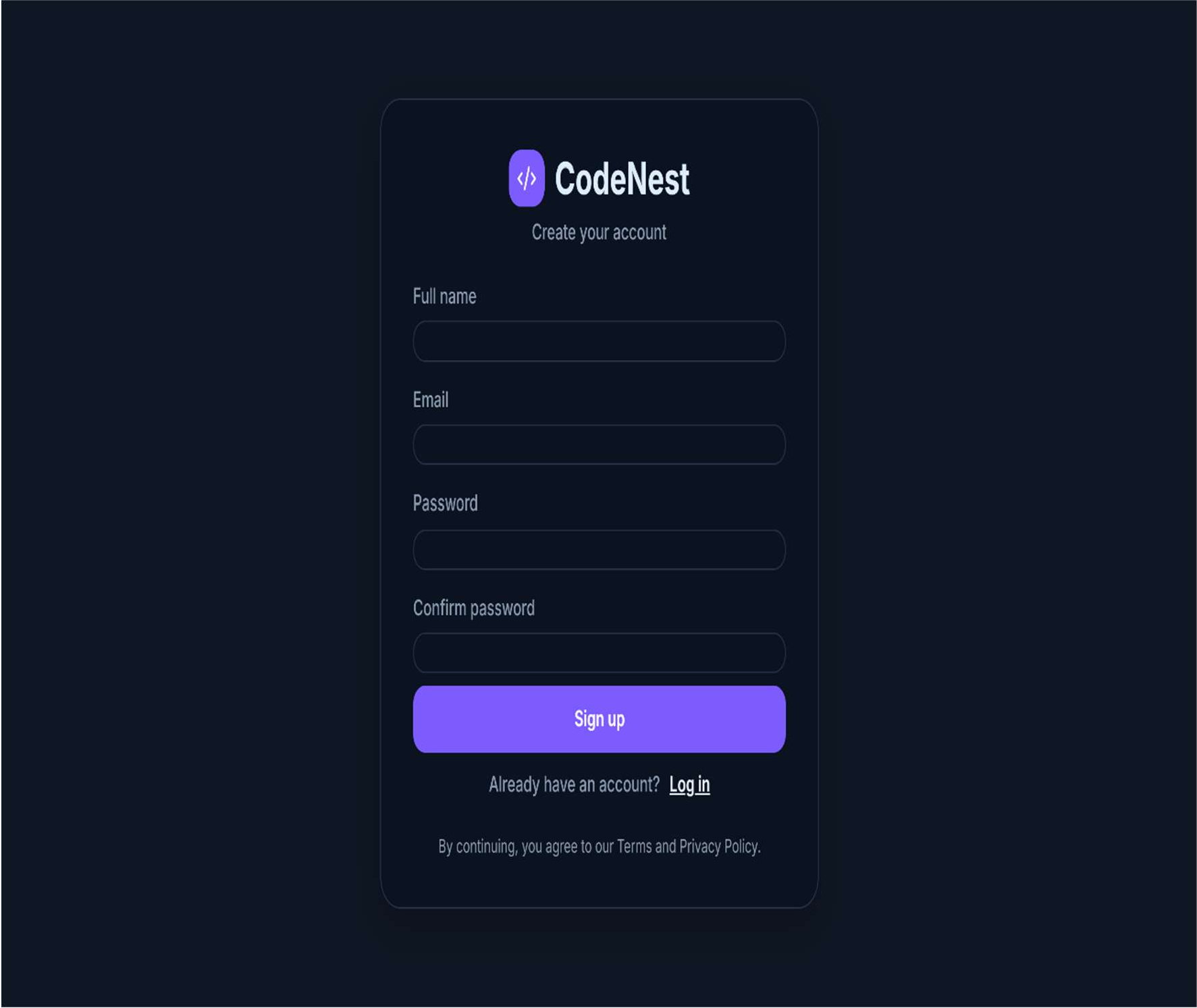


Figure 4.7 User Sign-Up Screen

In this sign-up page, as shown in Figure 4.7, new users are provided with a simple form to create their account by entering details such as full name, email, and password, along with confirming the password for security. Once the information is entered, users can click the Sign-Up button to register successfully. For those who already have an account, a direct Log in link is also available. This page ensures a smooth and secure onboarding process for users to access the platform.

#### User Login Page

This is the Login page of the platform. It allows existing users to access their accounts by entering their email and password. After filling in the details, they can click the Log in button to proceed. For new users who don’t have an account yet, a Sign-up option is provided. This page ensures secure and easy access to the platform.

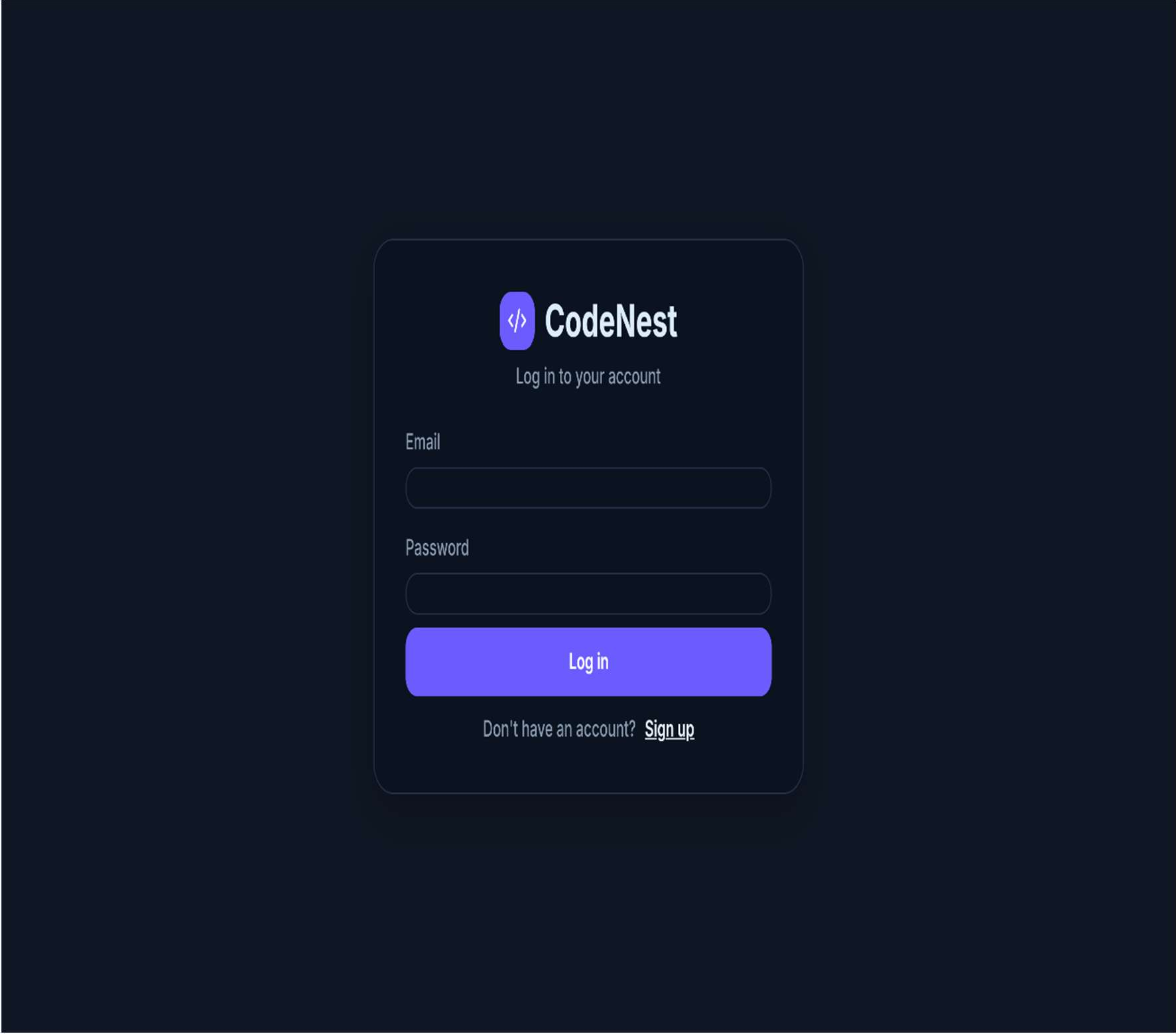


Figure 4.8 User Login Page

In this login page, as shown in Figure 4.8, users are prompted to enter their registered email and password to securely access the platform. After successful authentication, they can proceed to participate in contests, solve problems, and track their progress. For those who do not yet have an account, a direct Sign-Up link is provided, allowing new users to register. This page ensures a simple and secure entry point for accessing the platform’s features.

#### User Dashboard Page

This is the homepage of the platform. It introduces platform as a space for competitive programming, skill development, and community learning. Users can explore contests, practice coding problems, and track their progress through leaderboards. The page highlights features like thrilling contests, curated problem sets, and opportunities to join a coding community. Clear call-to-action buttons guide users to start coding, sign up, or explore challenges. Overall, it serves as the welcoming entry point to the platform



Figure 4.9 User Dashboard Page

In this homepage, as shown in Figure 4.9, users are introduced to the platform with key features such as contests, curated problems, and a vibrant coding community. Clear options guide them to start coding, explore challenges, or sign up to join the platform.

#### User Contest Page

This is the Contests page of the platform. It shows a list of ongoing, upcoming, and completed contests with their titles, descriptions, and dates. Each contest card also displays its status, such as Active, Upcoming, or Ended. Users can view details, register for upcoming contests, or check results of finished ones. This page helps participants stay updated and easily join coding challenges.



Figure 4.10 User Contest Page

In this contests page, as shown in Figure 4.10, users are presented with a list of coding challenges along with their status, descriptions, and timelines. The page allows participants to register for upcoming contests, view details of active ones, or check results of completed challenges, making it easy to stay engaged and track opportunities.

#### Contest Registration Page

This is the Contest Registration page for the Web Dev Hackathon. It allows users to join the contest by optionally entering a team name and agreeing to the contest rules. Once confirmed, they can click the Register button to complete their entry. There is also an option to cancel if they decide not to proceed. This page ensures a simple and clear process for participants to register for contests.

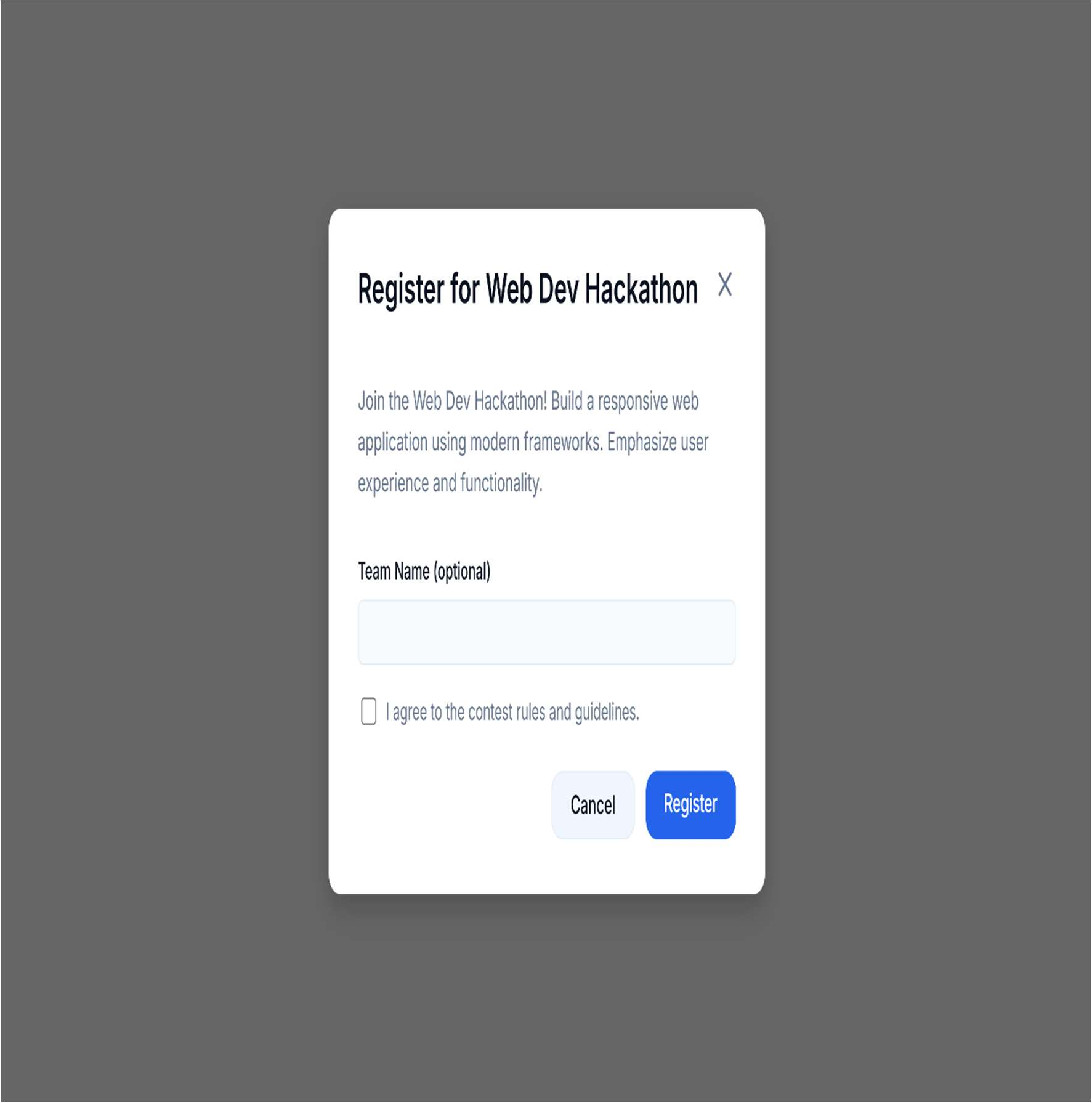


Figure 4.11 Contest Registration Page

In this contest registration page, as shown in Figure 4.11, users can join a contest by providing an optional team name and agreeing to the contest rules and guidelines. The page offers clear options to either complete the registration or cancel, ensuring a simple and user-friendly process for participation.

#### Problem Solving Page

This is the Problem Solving page of the platform. It shows a coding problem with its description, examples, and constraints on the left side. On the right, users can write their solution in the provided code editor, choosing from different programming languages. They can run their code to test it and then submit it for evaluation. This page provides a simple and interactive way for learners to practice and solve coding challenges.

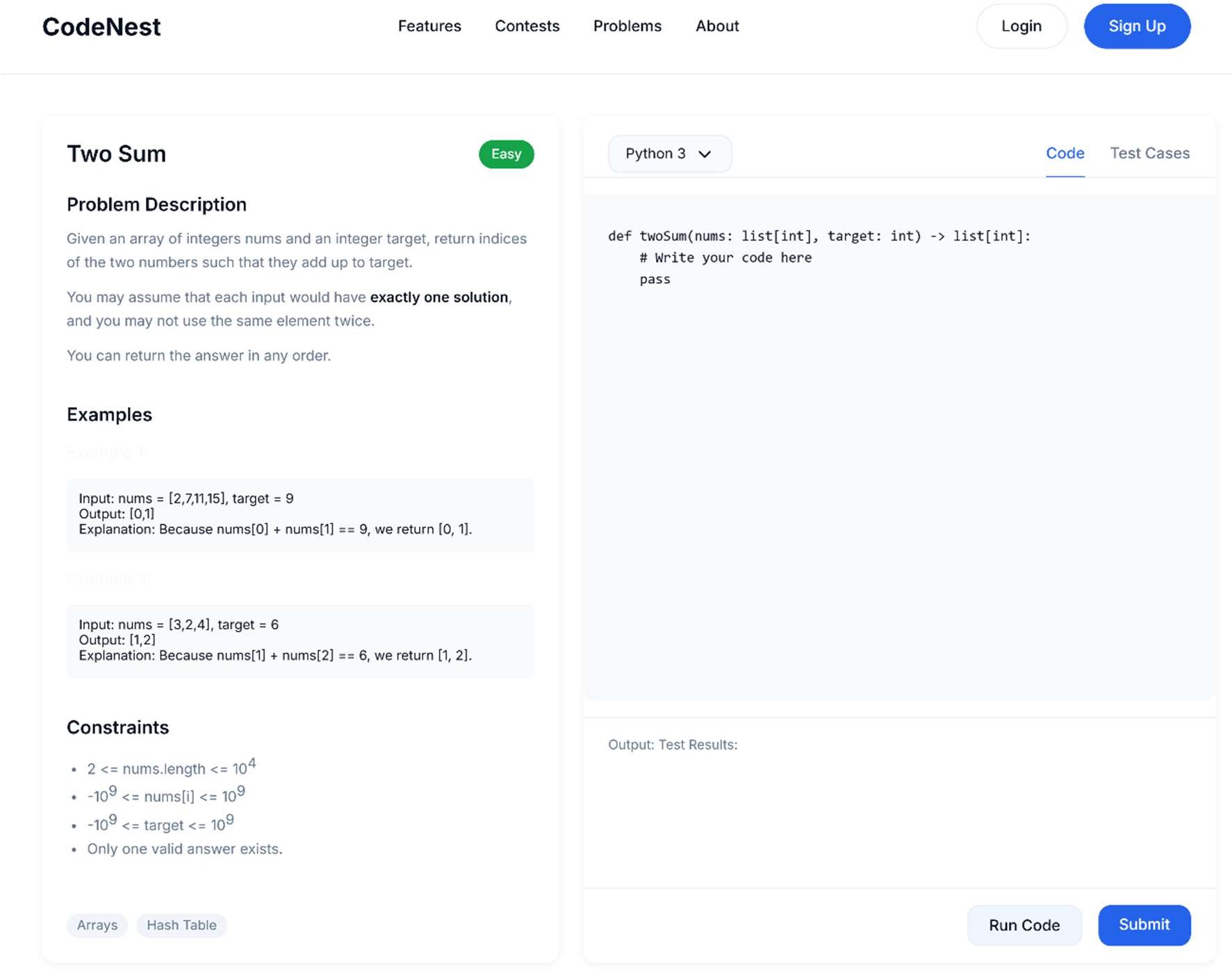


Figure 4.12 Problem Solving Page

In this problem solving page, as shown in Figure 4.12, users are presented with a coding challenge that includes a detailed problem description, sample inputs and outputs, and specific constraints to guide their solution. On the left side, all the necessary information about the problem is displayed, while on the right, users can write their code directly in the built-in editor. The platform allows them to select their preferred programming language, run the code to test against sample cases, and finally submit their solution for evaluation. This page provides an interactive environment that helps learners practice, test, and improve their coding skills in real time, making it both engaging and educational.

#### Submission Page of User

This is the My Submissions page of the platform. It shows a user’s past submissions with details like problem name, contest, programming language, score, and execution time. Each submission also has a status, such as Accepted, Wrong Answer, or Time Limit Exceeded. Users can search or filter submissions by contest or problem for quick access. The View option allows them to review their code and results. This page helps learners track and analyse their performance over time.

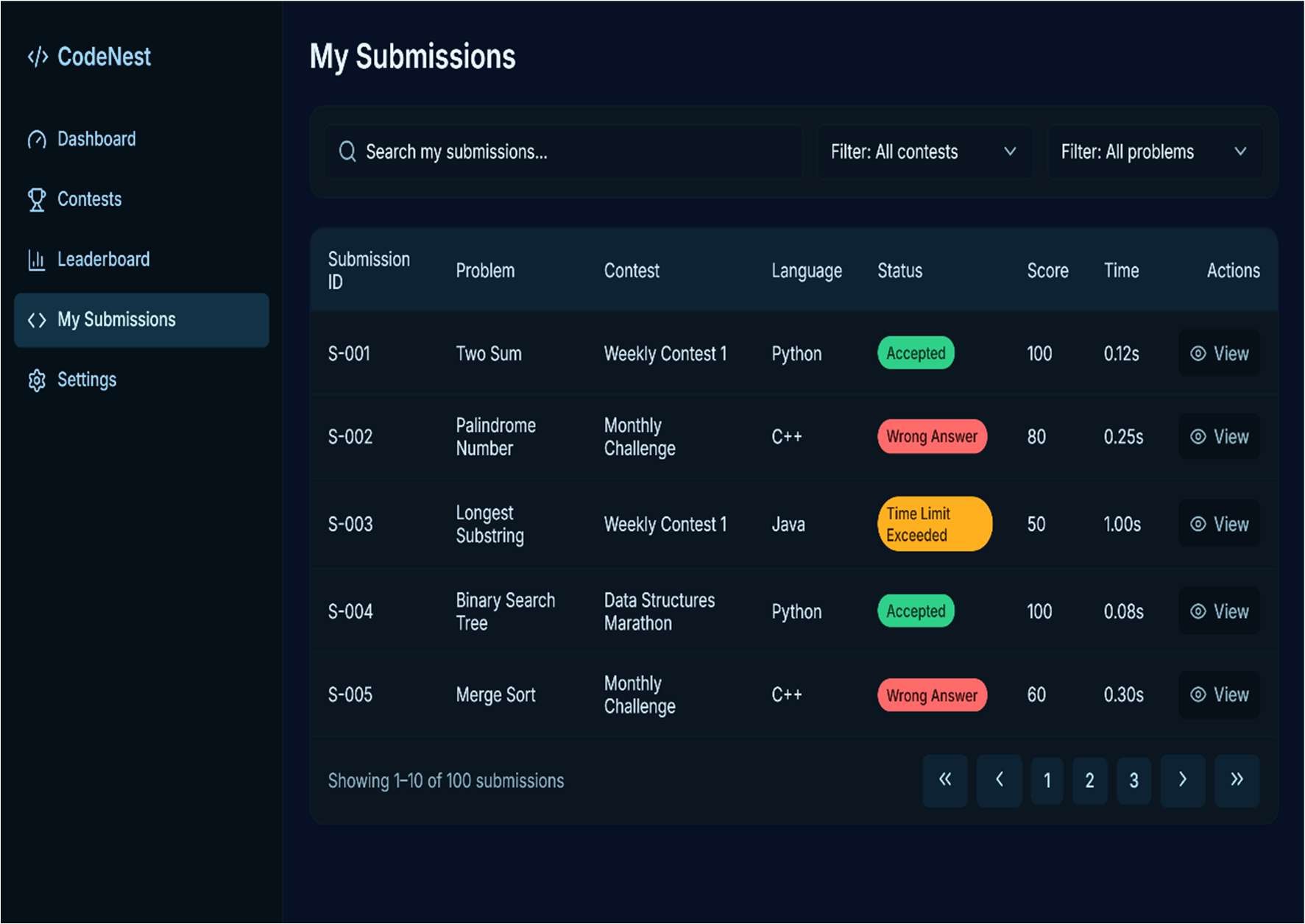


Figure 4.13 Submission Page of User

In this my submissions page, as shown in Figure 4.13, users are provided with a complete record of all their coding submissions across different contests and challenges. Each entry displays important details such as the submission ID, problem name, contest title, programming language used, status, score, and execution time. The results are clearly highlighted with labels like Accepted, Wrong Answer, or Time Limit Exceeded to help users quickly understand their performance. Additionally, filters and search options are available, making it easier to find specific submissions. The page also includes a View option, allowing users to review their code and outcomes in detail. This feature helps learners track progress, identify mistakes, and improve their problem-solving skills over time.

#### Leader Board Page

This is the Leaderboard page of the platform. It displays the ranking of students based on their scores from contests and problem-solving activities. Each student’s name, profile picture, and total points are shown along with their rank. The leaderboard highlights top performers and encourages healthy competition among participants. It helps users track their progress and stay motivated to improve their skills.

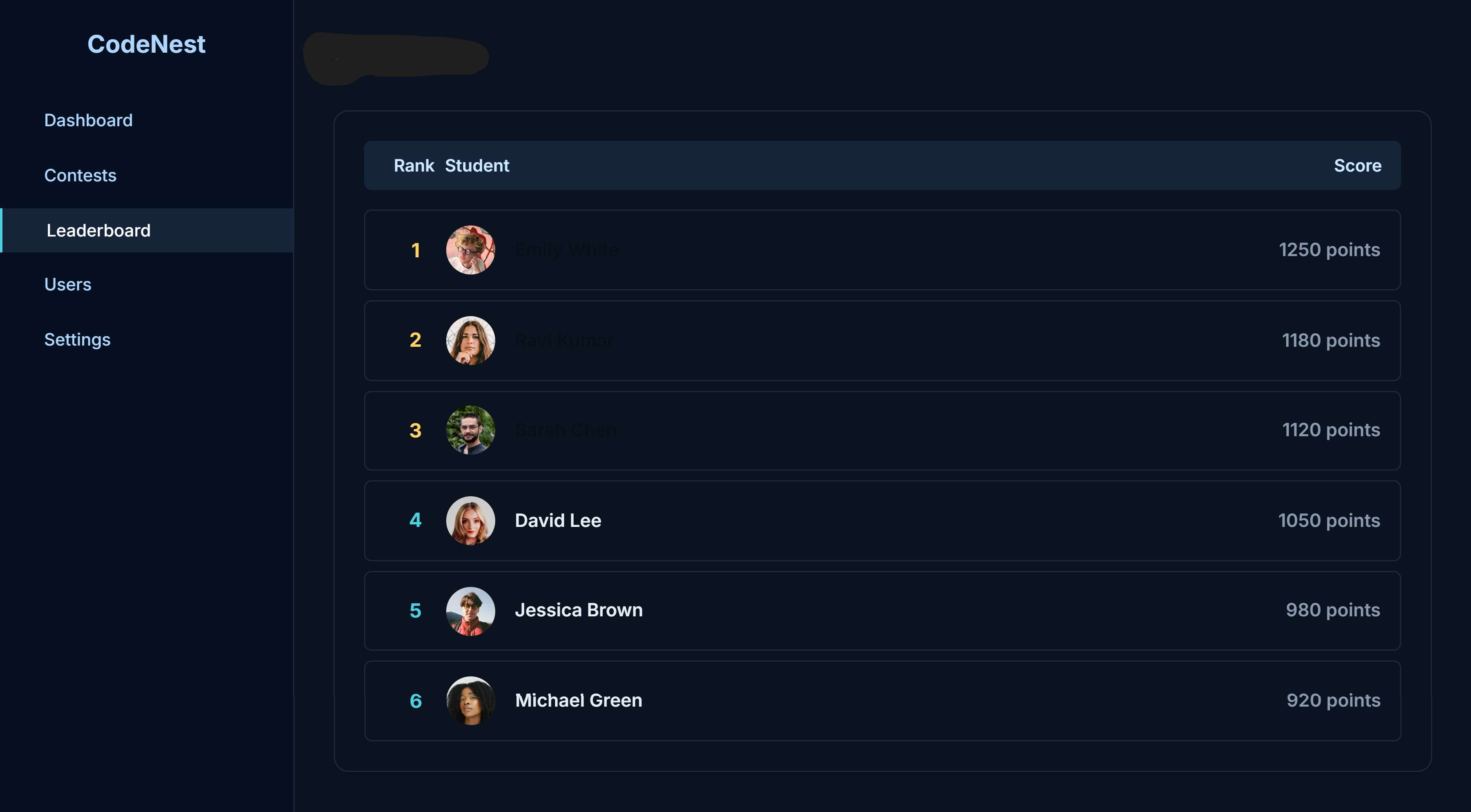


Figure 4.14 Leader Board Page

In this leaderboard page, as shown in Figure 4.14, students are ranked based on their overall performance and points earned through contests and problem-solving activities. The page displays the student’s name, profile image, rank, and total score, making it easy to identify top performers. This feature encourages healthy competition by motivating learners to improve their skills and climb higher on the leaderboard. It also provides transparency by showcasing achievements publicly within the platform. From here, students can track their progress, compare with peers, and stay inspired to participate more actively in challenges.

#### User Profile Page

This is the User Profile page of the platform. It welcomes the user by showing their name and email along with a short motivational message. The page provides quick options to either view past submissions or start solving new problems. It helps users easily access their progress and continue practicing without hassle. This page keeps the profile simple while focusing on learning and engagement.

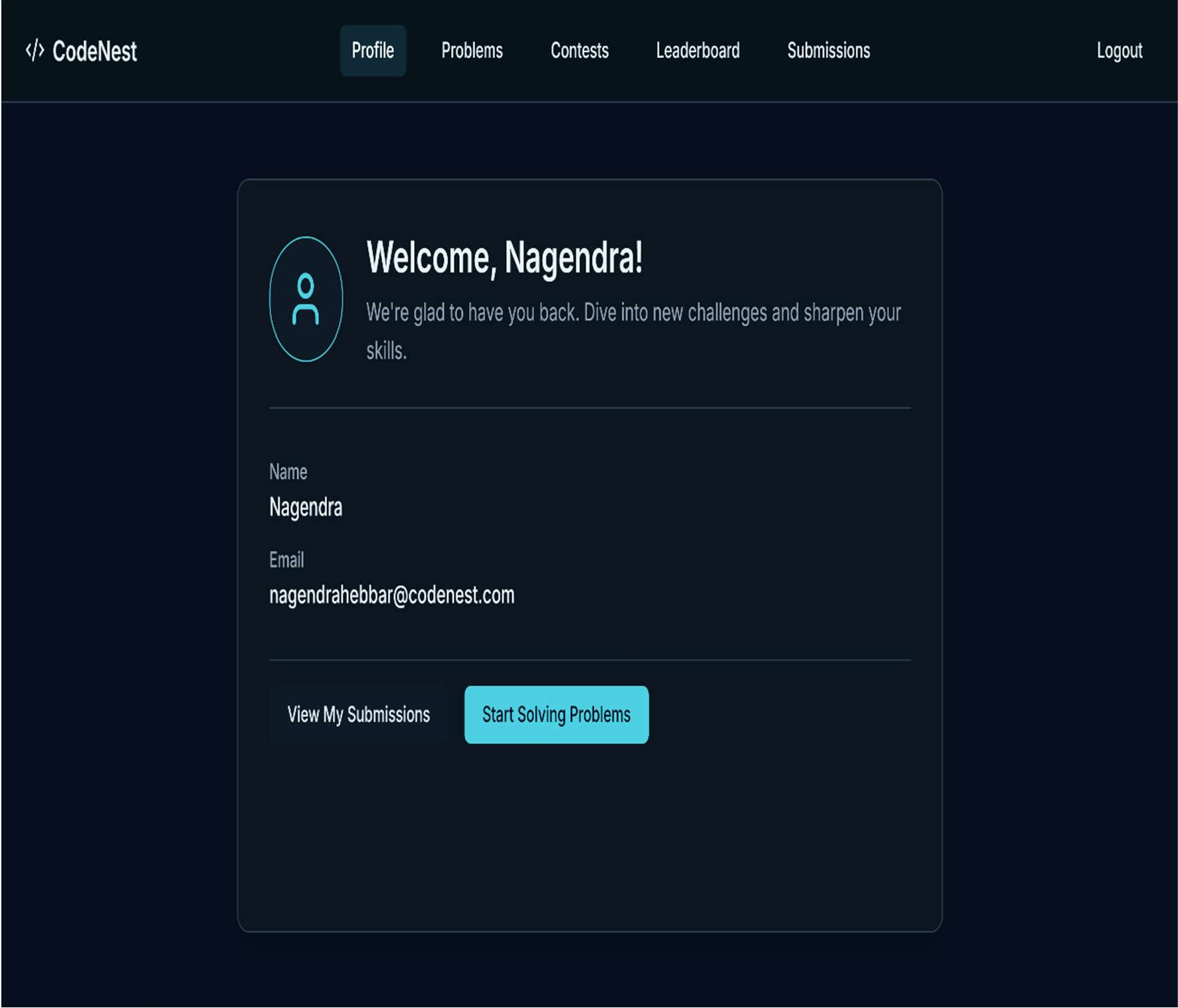


Figure 4.15 User Profile Page

In this user profile page, as shown in Figure 4.15, the platform displays the user’s personal details such as name and email, along with a warm welcome message to encourage engagement. The page provides quick access buttons to either view past submissions or start solving new problems, making it easy for users to continue their learning journey. It acts as a central hub for students to review their activity and track their progress. By offering simple navigation and essential information in one place, this page ensures a smooth and user-friendly experience. It highlights the balance between personalization and functionality, keeping users motivated to participate actively.

# CHAPTER 5

**CONCLUSION AND FUTURE ENHANCEMENT**

#### Conclusion

The Web-Based Coding Practice Platform to Empower MCA Students highlights how modern web technologies and secure engineering practices can transform programming education. By leveraging Node.js with Express for the backend and React.js with Vite for the frontend, supported by a MongoDB database and integrated with Judge0 API for real-time code execution, the platform provides a seamless, interactive, and reliable environment for coding practice and competitive problem-solving. Its modular architecture, RESTful APIs, and responsive dashboards ensure scalability, fast performance, and an intuitive experience for both students and administrators.

Throughout development, agile practices such as iterative design, continuous testing, and secure role-based access control were followed. This engineering discipline produced well-structured modules for problem management, contest creation, submission handling, and leaderboard tracking, all designed with usability in mind. The inclusion of real-time evaluation and automated feedback enables students to learn from mistakes instantly, fostering continuous improvement in coding skills.

The platform directly addresses limitations of traditional learning methods by automating code evaluation, reducing instructor workload, and providing immediate feedback, thus creating a more engaging and effective learning environment. Its flexible and maintainable design and gamification features like badges and rewards to further boost student engagement.

In conclusion, this project demonstrates how innovative web technologies and structured software design can create a scalable, secure, and user-friendly platform that supports skill development, practice, and competition for MCA students. By offering real-time problem-solving opportunities, progress tracking, and collaborative features, the platform equips learners with the digital tools necessary to succeed in academic and professional programming challenges, laying a solid foundation for future innovations in coding education.

#### Future Enhancements

Building on the current platform’s foundation and its web-based deployment, future enhancements could focus on expanding cloud integration to support larger student cohorts and multiple institutions, enabling scalable and collaborative coding environments. Implementing AI-powered features, such as automated hint generation, personalized problem recommendations, and plagiarism detection, would provide intelligent support to learners and help instructors monitor academic integrity. Developing a mobile application would increase accessibility, allowing students to practice coding and participate in contests anytime, anywhere. Additional enhancements like gamification elements (badges, leaderboards, and rewards) could boost engagement and motivation among learners. Strengthening user interface accessibility, along with advanced security features such as two-factor authentication and real-time monitoring, would make the platform more inclusive, resilient, and secure. These improvements will evolve the system into a comprehensive, intelligent, and scalable learning tool, empowering MCA students to master coding through continuous practice, competition, and innovation.

# CHAPTER 6 TEAM WORK DETAILS

#### Project Team:

1. Development Team:

The project “A Web-Based Coding Practice Platform to Empower MCA Students” was developed by Mr. H S Nagendra Hebbar, who designed and implemented both the frontend and backend. He integrated core modules such as User Management, Contest Management, Problem Repository, Submission and Evaluation Engine, and the Admin Dashboard. His work covered system architecture, database design, RESTful API development, and frontend UI creation, ensuring the platform operates as a cohesive and efficient coding practice environment.

1. Backend Developers:

Although the project was developed by Mr. H S Nagendra Hebbar, the overall development process benefited from valuable guidance and technical support from mentors experienced in Node.js, MongoDB, and web application development. Their inputs were especially helpful in shaping the backend to be robust, scalable, and capable of securely handling authentication, contest workflows, problem management, and data storage.

1. Stakeholders:
   1. Admin users (faculty/mentors): They oversee the platform by managing coding contests, problems, student accounts, and monitoring performance reports.
   2. Students (primary users): They register, log in, attempt coding problems, participate in contests, and review their submissions and results.
   3. Academic stakeholders (faculty coordinator): They benefit from real-time dashboards, leaderboards, and analytics that track student progress and help streamline the learning and evaluation process.
2. Collaboration and Expertise:

The project was completed through collaboration between Mr. H S Nagendra Hebbar and academic supervisors including Dr. C Bhanuprakash, who provided continual guidance and ensured that the project met academic and industry standards. The

development process leveraged combined expertise in backend development, frontend design, and secure system integration, resulting in a scalable and user-friendly retail billing platform tailored for small and medium businesses.

# References

1. M. Jacobs, R. Andrews, and T. Liang, “Personalized Programming Task Generation Using AI,” Springer AI Education Journal, vol. 15, no. 2, pp. 120–134, 2025.
2. Z. Hou, L. Zheng, and J. Gao, “LLM-Based Parsons Puzzles for Introductory Programming,” arXiv preprint, arXiv:2402.04123, 2024.
3. A. Montoya-Dato, S. de la Torre, and M. Hernández, “Building Test Case Skills Through Online Coding Practice,” J. Softw. Educ., vol. 9, no. 1, pp. 88–102, 2023.
4. Elice Team, “Scaling Online Coding Education: A Case Study,” Elice Reports, vol. 7, no. 3, pp. 55–63, 2023.
5. D. Djenic and D. Mitic, “Blended Learning for Programming Courses: A Systematic Review,” Educ. Technol. Rev., vol. 19, no. 4, pp. 211–225, 2021.
6. M. Ivanović, D. Budimac, and Z. Putnik, “Collaborative Coding in Higher Education,” J. Educ. Comput. Res., vol. 57, no. 6, pp. 1432–1450, 2019.
7. A. McBroom, I. Koprinska, and K. Yacef, “A Survey of Automated Programming Hint Generation – The HINTS Framework,” IEEE Trans. Learn. Technol., vol. 12, no. 4, pp. 413–429, 2019.
8. R. Topalli and K. Cagiltay, “Gamification in Programming Education: A Systematic Review,” Comput. Educ., vol. 127, pp. 21–36, 2018.
9. A. Barczak, K. Singh, and L. Chen, “Automated Assessment System for Teaching Data Structures,” Int. J. Educ. Technol., vol. 6, no. 1, pp. 77–85, 2018.
10. S. Gulwani, I. Radicek, and T. Zuleger, “Automated Program Repair for Introductory Programming Assignments,” ACM Trans. Comput. Educ., vol. 16, no. 4, pp. 1–20, 2016.
11. A. Sharma and M. Patel, “Spring Boot Based Academic Coding Application with REST APIs,” Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 11, no. 5, pp. 98–104, 2021.
12. M. Rajan, R. Das, and P. Shetty, “Mobile Coding Platforms for Learning,” Int. Res. J. Eng. Technol. (IRJET), vol. 9, no. 2, pp. 1045–1050, 2022.
13. A. Vashisht and B. S. Rekha, “Microservices and Real-Time Processing in Educational Platforms: A Review,” Academia.edu, pp. 1–6, 2025.
14. S. Jain and K. Bhatnagar, “Low-Latency Inventory Systems in Java Spring Boot,” in Proc. Int. Conf. Softw. Syst. Eng., pp. 115–121, 2022.
15. A. Pears, J. Seidman, and D. Malmi, “Foundations of Interactive Coding Tools in CS Education,” ACM SIGCSE Bull., vol. 39, no. 4, pp. 96–114, 2007.

**Project Self-assessment for Program Outcomes**

|  |  |  |  |
| --- | --- | --- | --- |
| **PO’s**  **No** | **Program Outcomes** | **Contribution from the project** | **Level** |
| 1. | Foundation Knowledge: Apply knowledge of mathematics, programming logic and coding fundamentals for solution architecture and problem solving. | Applied programming fundamentals and database design principles to build a coding practice platform supporting problem-solving, contests, and  submissions. | 4 |
| 2. | Problem Analysis: Identify, review, formulate and analyze problems for primarily focusing on customer requirements using critical thinking  frameworks | Analyzed gaps in existing coding platforms, identified the need for MCA- focused practice, and designed solutions to improve coding engagement and  evaluation. | 3 |
| 3. | Development of Solutions: Design, develop and investigate problems with as an innovative approach for solutions incorporating ESG/SDG goals. | Developed an interactive web-based platform with modules for problem repositories, contests, submissions, and automated evaluation to foster hands-on  learning. | 4 |
| 4. | (Modern Tool Usage): Select, adapt and apply modern computational tools such as development of algorithms with an understanding of the limitations including  human biases. | Implemented Node.js, React.js, and MongoDB with Judge0 API for real-time code execution and assessment, validating performance through testing  and results. | 4 |
| 5. | (Individual and Teamwork): Function and communicate effectively as an individual or a team leader in diverse and multidisciplinary groups. Use  methodologies such as agile. | Followed agile methodology, collaborating with peers and mentors to integrate frontend, backend, and evaluation modules into a cohesive  platform. | 4 |
| 6. | (Project Management and Finance): Use the principles of project management such as scheduling, work breakdown structure  and be conversant with the principles of | Structured project timeline with clear milestones for design, development, testing, and deployment, ensuring timely  completion within available resources. | 3 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Finance for profitable project  management. |  |  |
| 7. | (Ethics): Commit to professional ethics in managing software projects with financial aspects. Learn to use new technologies for cyber security and insulate customers from  malware | Ensured fair evaluation, plagiarism-free submissions, and secure user data handling while promoting academic integrity in coding practices. | 4 |
| 8. | (Life-long learning): Change management skills and the ability to learn, keep up with contemporary technologies and ways of working. | Explored and applied modern web frameworks and coding evaluation engines, adapting to new tools to improve functionality and scalability of the  system. | 4 |

|  |  |
| --- | --- |
| Level Grade | Level Grade |
| Poor | 1 |
| Average | 2 |
| Good | 3 |
| Very Good | 4 |
| Excellent | 5 |

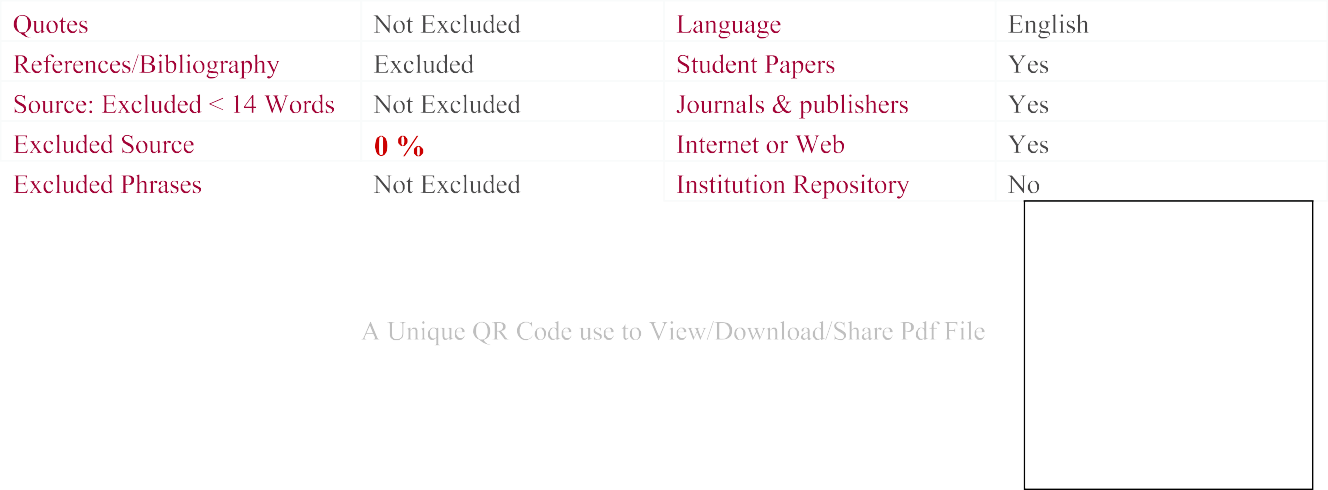
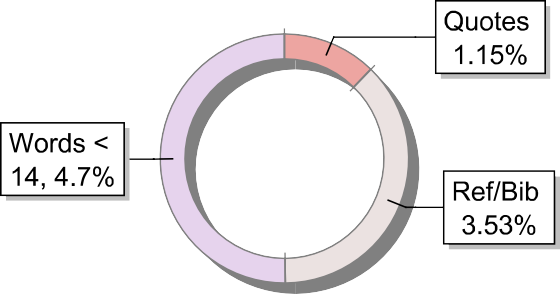
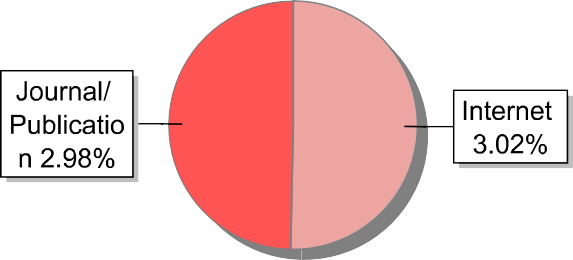


****

****

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

****



**VITAE**



Name: H S Nagendra Hebbar USN: 1SI23MC012

DOB: 30/06/2002

Permanent Address: #3/428, Moodumane, Sanjay Gandhi Road, Hangalore, Kundapura, Udupi,

Pin code: 576217

Phone No: 8616319835

Email: [nagendrahebbar06@gmail.com](mailto:nagendrahebbar06@gmail.com) CGPA: 8.74 (upto 3rd sem)

Placed: TCS(Ninja) CTC: 3.5 LPA