
An Innovative Online Judge for UIU's Competitive Programming

By

Abir Hossain Amee (011193032)
Md Iftexhar Hossain (011193034)
Sara Ferdous Khan (011193030)
Sanjida Jannat Anannaya (011201440)
Upama Roy (011201183)
Arpita Mazumder (011193014)

Submitted in partial fulfilment of the requirements
of the degree of Bachelor of Science in Computer Science and Engineering

November 4, 2024



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNITED INTERNATIONAL UNIVERSITY

Abstract

Competitive programming has emerged an essential skills for students, in computer science and technological landscape. Numerous online judging systems were developed to automatically evaluate computer code. Open-source online judge systems, such as Codeforces and Hackerrank, are also progressively evolving in tandem with the development of online judge systems. Writing algorithmic solutions for computer programming involves a set of intellectual abilities that can be acquired readily with practice along with requirement of feedback on given exercises and learn from mistakes. This study introduces an online judge platform: CodeKoro, especially designed for UIU students as a way to sharpen the competency in programming process and promote a culture of competitive programming inside our university. CodeKoro aims to offer a place to compete, practice, and improve their problem-solving expertise in a methodical and effective way empowering students with a dedicated platform defining the objectives to create a succinct online judge software application. This online judge platform is equipped with optimal infrastructure, an assortment of problem sets, real-time feedback, and performance analytics to anticipate a paradigm shift raising the academic and programming skills of students and to support a collaborative and competitive learning environment.

Acknowledgements

This work would have not been possible without the input and support of many people over the last two trimesters. We would like to express my gratitude to everyone who contributed to it in some way or other.

Firstly, our sincere gratitude goes to our supervisor Dr. Suman Ahmmed Sir for his guidance, expertise and counsel. We would also like to thank Dr. A.K.M. Muzahidul Islam Sir, our course teacher, who mentored us with all the necessary guidelines and prepared us for the final presentation.

Last but not the least, We owe to our family including our parents for their unconditional love and immense emotional support.

Table of Contents

Table of Contents	iv
List of Figures	v
List of Tables	vi
1 Introduction	1
1.1 Project Overview	1
1.2 Motivation	2
1.3 Objectives	2
1.4 Methodology	3
1.5 Organization of the Report	5
2 Background	6
2.1 Preliminaries	6
2.2 Literature Review	7
2.2.1 Similar Applications	7
2.2.2 Related Research	9
2.3 Gap Analysis	12
2.4 Summary	12
3 Project Design	14
3.1 Requirement Analysis	14
3.1.1 Functional and Nonfunctional Requirements	14
3.1.2 Context Diagram	16
3.1.3 ER Diagram	17
3.1.4 Data Flow Diagram Level 1	18
3.1.5 UI Design	19
3.2 Detailed Methodology and Design	24
3.3 Project Plan	25
3.4 Task Allocation	26
3.5 Summary	28

4	Implementation and Results	29
4.1	Environment Setup	29
4.2	Testing and Evaluation	31
4.3	Results and Discussion	33
4.4	Summary	36
5	Standards and Design Constraints	37
5.1	Compliance with the Standards	37
5.1.1	Software Standards	37
5.1.2	Hardware Standards	38
5.1.3	Communication Standards	38
5.2	Design Constraints	39
5.2.1	Economic Constraint	39
5.2.2	Environmental Constraint	39
5.2.3	Ethical Constraint	40
5.2.4	Social Constraint	40
5.2.5	Political Constraint	41
5.2.6	Sustainability	41
5.3	Cost Analysis	41
5.4	Complex Engineering Problem	43
5.4.1	Complex Problem Solving	43
5.4.2	Engineering Activities	44
5.5	Summary	45
6	Conclusion	46
6.1	Summary	46
6.2	Limitation	46
6.3	Future Work	47
	References	49

List of Figures

1.1	Project Overview	1
1.2	Methodology diagram	4
3.1	Context Diagram of the system	16
3.2	ER Diagram of the system	17
3.3	Data Flow Diagram of the system	18
3.4	Signup Page	19
3.5	Login Page	20
3.6	Home Page	20
3.7	Problem Set	21
3.8	Ranking Page	21
3.9	Contest Problems Page	22
3.10	Details of a Problem	22
3.11	Problem Submit page	23
3.12	My Submission Page	23
3.13	Standing Page	24
3.14	Flowchart of the Detailed Methodology	25
3.15	Gantt Chart for the Project Plan	26
4.1	Signup	33
4.2	Ranking	34
4.3	Create Contest	34
4.4	Add Problems to Contest	35
4.5	Contest Problems	35
4.6	Contest Problem Submit	36

List of Tables

2.1	Mapping with complex problem-solving.	9
3.1	WBS Package 1	27
3.2	WBS Package 2	28
4.1	UI/End-to-End test Selection	32
5.1	Development Costs	42
5.2	Infrastructure Costs	42
5.3	Operational Costs	43
5.4	Other Costs	43
5.5	Mapping with complex problem solving.	44
5.6	Mapping with complex engineering activities.	44

Chapter 1

Introduction

This chapter provides the background of our project, including motivation, objectives, the problem statement, a brief methodology, the project outcome, and the organization of the report. Section 1.1 gives an overview of the project and its scope. The reasons for choosing this project are explained in Section 1.2, while Section 1.3 outlines the project's objectives. The methodology is detailed in Section 1.4, followed by the project's outcome in Section ?? and the organization of the report in Section 1.5.

1.1 Project Overview

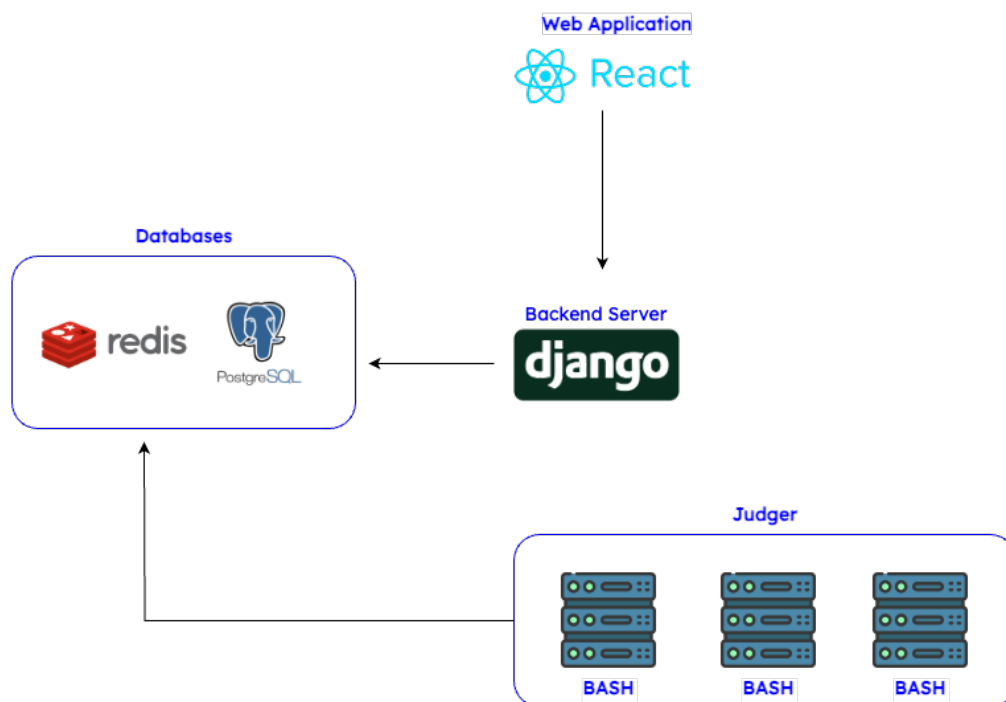


Figure 1.1: Project Overview

Our goal is to develop an online platform exclusively for UIU students that will assist them in competitive programming. We noticed students aren't as interested or doing well in coding competitions, so we're creating CodeKoro to fix that. It's like a special place where students can practice coding, compete with each other, and get better together. This will offer a focused environment for students to practice coding, solve algorithmic challenges, track their progress, compare performance with peers, form coding teams, and cultivate a vibrant coding community within UIU.

1.2 Motivation

Setting is a valuable discipline in computer science, as far as problem-solving skills and algorithmic reasoning, which are always in great demand by employers. But over the years, the students of United International University have been facing quite a lot of challenge in this area that can be detrimental to their future careers. To address this issue, we propose the development of a platform, named CodeKoro, tailored to the specific needs of the student of United International University.

The students of United International University are not efficient enough in competitive programming from their performance. This may be because practicing is scarce, and there is no organized forum in which they can improve their skills. Therefore, these students may not easily be placed to compete for those internships or jobs that may demand good programming skills. The aim of this project is to build an effective online learning tool called CodeKoro that shall facilitate practice, appraisal and progress check of the students in programming questions. On this platform we seek to improve competitiveness of United International University students in programming and in turn help boost their employability.

1.3 Objectives

This section highlights the objectives we have for this project:

- **Inspecting the performance in competitive programming:**

Investigate the key circumstance underlying the UIU student's deteriorating active performance in this field. Apprehend the barriers and challenges impeding students from enthusiastically participating in CP (competitive programming) in corresponding activities.

- **Advancing and Reinforcing the CODEKORO platform:**

In order this serve the need of the UIU student, it is important to acquire comprehensive understanding of the requirements to establish a competitive programming platform. Upgrade CodeKoro, an online judging platform that is easily accessible and easily available to UIU student only.

- **To Elevate student’s active participation:**

Provide CodeKoro we-gimmick/tools that should help the students self-assess and chart their progress in the methodical approach to coding. Also, consider provision of appliances that are used to track development of each employee and provide encouragement as a way of raising interest and disciplined self-motivation. Letting students benefit from the extensive opportunities for developing their skills alongside their counterparts by using CodeKoro’s leaderboard and ranking. Students should be allowed to communicate with other students who might be part of their programming contest team through functions so that the spirit of cooperation and teamwork noticed in group affair be adopted in programming contests.

- **To Appraise CodeKoro’s Effectiveness:** Assess how codeKoro affects students’ involvement and output in competitive programming activities by analyzing input from instructors and students to compress user happiness and highlight areas that require work. Integrate Knowledge attained from developing and software of CodeKoro to advance the field of instructional technology. To decide upon the most implemented procedures and acquired knowledge that can be used within other educational settings and come up with the same systems.

- **To Examine Sustainability and Long-Term Advantages:**

Analyse the potential positive impacts of CodeKoro for building competency and organizing UIU’s competitive programming event. Reflect on the idea of promoting the overall, long-run continuous improvement of CodeKoro as to the needs of UIU students as they adapt.

1.4 Methodology

The initial phase of our study methodology proposes employing the online judging platform “CodeKoro”, only for UIU students to discover and boost the immense potential and competencies in the context of UIU students. In the problem statement session, we demonstrate detailed insights about the lack of activity among UIU students in competitive programming and the consequences of talent identification for CP.

This guides us in determining the oversight problem of underperformance in competitive programming contests of UIU students as well as the benefits and advancing the potential for online judge practices intervention to address the issues of student’s lack of enthusiasm and prejudice besides discouragement about taking part in competitive programming contests like Hackerrank and HackerEarth, as well as ICPC and GCJ (Google Code Jam). Performing a thorough “Gap Analysis” to pinpoint the precise issues that our research seeks to solve and highlight the specific limitations that “CodeKoro” aims to address along with identifying any gaps in the current infrastructure and processes, entails researching the literature, investigating and comprehending comparable platforms,

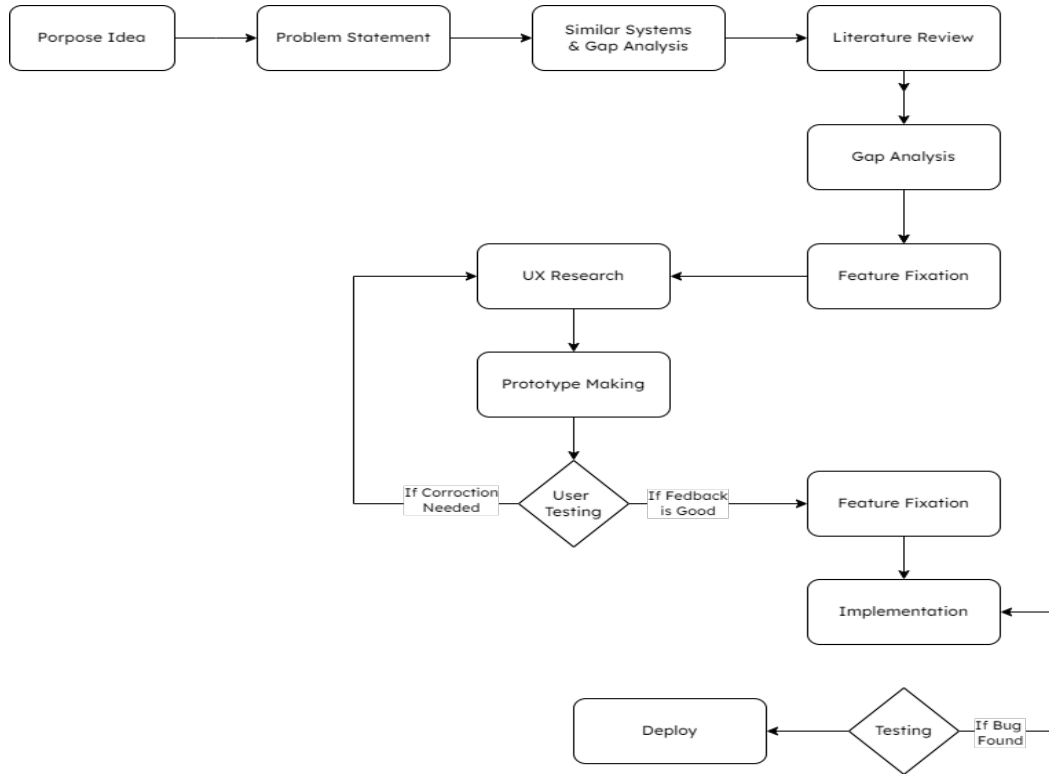


Figure 1.2: Methodology diagram

examining consequential methods, inspecting their features, user base and effectiveness in promoting competitive programming. The literature review step involves deep examination studies on the impact of online judge platforms on student learning outcomes and performances. To optimize the online judging platform and make it more interactive and user-friendly for novices in this sector, it is necessary to gather more significant insights and data sources. This entails the use of surveys questionnaires and other related items. For the development and sustenance of CodeKoro online judging platform, the adaptable deployment patterns for the purposes of improvement in the future are required. This leads to the “Feature List Finalization” where features for a comprehensive list for “CodeKoro” are compiled from gaps and requirements collected from users further prioritizing features that increase engagement, skill tracking, and competition. For our UX Research section, we ought to conduct interviews in our team to understand the preferences and problematic points of UIU students regarding online judge platforms and utilize feedback to refine the user interface improving overall user experience. We will develop a prototype of “CodeKoro” using Figma to visualize the platform and incorporate additional feedback from stakeholders. In the forward step, we implement “CodeKoro” integrating features such as access control, problem set management, contest dashboard, live contest hosting and compiler functionality. Also, we develop the backend logic for user authentication, data management and contest administration. In the Test phase, we perform unit tests, integration tests and user acceptance tests to identify and rectify any

issue ensuring functionality, usability, availability and security through choosing optimal multi-architecture. Additionally, in the deployment and documentation phase, we deploy “CodeKoro” on a suitable hosting platform, ensuring scalability and reliability and provide comprehensive documentation and support for users and administrators to facilitate smooth adoption and operation of the platform.

1.5 Organization of the Report

In section 1 the introduction part is highlighted. Here we talked about the competitive programming platform for the students of UIU. Then section 1.1 we talked about the project overview. Here we highlighted a simple diagram of our project. Section 1.2, Motivation, explains all the factors that provide illuminate the reasoning behind our research. Moving on, in Section 1.3, Objective, we distinctly stated the goals that we aim to fulfil through this project. We discussed our study design in more detail in Section 1.4, Methodology. The organization of the report is described in Section 1.5, which additionally provides an outline of the next chapters. We explored the fundamental elements of our research in Chapter 2, Background. Here, in Section 2.1, Preliminaries, we provided the terminology required to read into our research. The literature review in Section 2.2 covers Related Research in Section 2.2.1 and Similar Applications in Section 2.2.2, respectively, providing a comprehensive overview of the body of knowledge. Subsequently, we carried out a thorough Gap Analysis in Section 2.3 to determine the research gaps that still need to be filled. In Section 2.4, we concluded by summarizing the most significant lessons learned and findings from our research. The planning and design phases of our study were covered in the next section Chapter 3, Project Design. Section 3.1 offers a comprehensive evaluation of both functional and non-functional requirements. It includes an ER diagram, a context diagram, and data flow diagrams, in Sections 3.1.1, 3.1.2, 3.1.3 respectively. The user interface for our project is represented in Section 3.1.4, UI Design. The user interface for our project is represented in Section 3.1.5, UI Design. In Section 3.2, the Detailed Methodology and Design, we further examined at the technical aspects of the design and strategy we adopted. Section 3.3 presents our project plan, while Section 3.4 provides the task allocation of our team. Finally, Section 3.5 shows the overall summary of our project. In chapter 5 the standards and design constraints. In chapter 5.3 the cost analysis part is highlighted. Then In chapter 5.4 complex engineering problem is highlighted. And finally chapter 5.5 the summary part is highlighted.

Chapter 2

Background

In this chapter, the literature on "Competitive Programming Online Judge Platform" is thoroughly analyzed, along with previous studies and implementations. Section 2.1 contains the background information needed to comprehend our project. The assessment of the literature is included in Section 2.2 and it is broken down into two subsections: Subsection 2.2.1 offers a thorough analysis of comparable applications, and Subsection 2.2.2 covers related research in this field. After that, Section 2.3 discusses the gaps in relevant research publications, and at the end with the summary of the chapter in section 2.4 .

2.1 Preliminaries

Competitive Programming has developed into a sport code among programmers globally, in which people compete on algorithmic content. Competitive programming online judge platforms are part and parcel of programming education and have been adopted as performance benchmarking tools for novices and experienced programmers as well as recreational programmers who host coded competition events where they can compete with other participants. At the heart of this practice are Competitive Programming Online Judge Platforms, virtual environments where programmers can submit solutions to coding problems, run these solutions against the test cases provided and compare them to those of several other competing programmers either simultaneously or at different times.

1. Competitive Programming Landscape: Competitive Programming is solved algorithmic problems which are provided to the competitor within some span of time and the competitor must come up with efficient problem solving techniques across various problems. Programmable has become popular as a game with festival and competition being celebrated both globally and on the internet.

2. Evolution of Online Judge Platforms: Online Judge Platforms provide the environment and system to hold the contests and practice session of competitive programming. They represent a vast source of problems in many categories including data structures, algorithms, mathematics and so much more. these more recent platforms have embraced

enhancements that include the occasion real-time feedback, users' polls ranking facility, and educational or training contents to help address a growing demographic.

3. Significance of Competitive Programming: Competitive programming is not just about solving problems as it improves someone's logical thinking skills, algorithms, and code structuring and writing skills. In addition, knowledge and skills in competitive programming is appreciated in the technology market, because it proves problem-solving skills and computer literacy skills.

4. Purpose of the Report: This paper will seek to capture different reviews of existing literature on Competitive Programming Online Judge Platforms. For one, it will discuss earlier scholarships and application of CBR approaches, evaluating the merits and demerits of similar systems, and reviewing the literature for research gaps. This way, we try to make a relevant contribution to the discussions around these platforms and have glimpses of how our insights can influence further developments.

Section 2.1: Main subdivisions are the following: Background Information section which aims to introduce the key facts to get acquainted with if one wants to grasp the importance and the scope of the project. Section 2.2: Literature Review: As this project does not have a comprehensive literature review, but rather organizes the literature into sections that examine similar applications and other research. Section 2.3: Contribution to Management Knowledge: Outlines directions in which prior research may have omitted, underestimated, or down-played significant phenomena, and hence provides potential research topics for aspiring authors. Section 2.4: Chapter Conclusion: Presented a consolidation of ideas on the pertinent themes from the literature analysis before presenting the subsequent discussions and suggestions.

2.2 Literature Review

Numerous research has been conducted in the context of the Competitive Programming Online Judge platform, and as technology advances daily, so do the methods for efficiently sharpening the skills with user-friendly and more effective platforms. For computer programming learners, attaining the utmost competence in a competitive programming context has been a persistent issue worldwide. Student's perceptions of competitive programming acquisition are always been lacking behind the current requirement due to ignorance and inexperience with online judge platforms and challenges in this very crucial field. Therefore, there is a limitless amount of study in this field. The platforms and studies that are now available that deal with online judge platform are described in the literature review.

2.2.1 Similar Applications

We have analyzed five similar applications that are comparable to our project scope. To better understand each system, we went through the following applications meticulously

by creating accounts and scrutinizing the whole system. Based on our inquiry, we then carried out a benchmark analysis to have an overview of the similarities and differences of these five applications to ours.

- Codeforces [1]: It is an online competitive programming platform of users who encounter diverse features and components to enhance their algorithmic and coding abilities. Some of features include Contests, Problem Sets, Rating Systems, Virtual Contests, Community Interaction, API and Tools.
- LightOJ [2]: LightOJ short for Light Online Judge is an online judge system that aims to allow Competitive programmers and other interested in solving algorithmic problems to practice online. It is a Web Judge System that provides a number of programming problems of different difficulty levels. Websites are available where problems involving data structure can be solved by users, and the site gives feedback to the user on correctness and optimized running time of the users function.
- CODECHEF [3]: CodeChef is a competitive programming platform offering a variety of features for programmers of all skill levels. Some of its key features are Contests, Practice Sections, Discussion Sections, Editorials, a Rating System, User Profiles, Multi language support. From novices to seasoned professionals, participants engage in various competitions such as long challenges, cook-offs, and lunchtime contests. Every event is carefully designed in terms of choosing problems that cover varying skill levels and numerous algorithms for every contestant to find the task exciting.
- Toph [4]: Dear developers who are residing in Bangladesh, we are here to call you! Toph is actually your one-stop shop for competitive programming. Develop the skill by solving practice problems from their large problem database or try out problems in an online or live competition. Toph scales to handle thousands of participants so if you want to race against the best this is for you. Toph's platform is very easy to use, so the organisers can easily create coding challenges. Submissions, announcements, plagiarism check, and many more can all be done from the comfort of your dashboard. Cons One is that the scalable infrastructure makes everything appear very seamless to everyone who is or is not involved. That is to say, Toph is the place where bangladesh programming community gathers to learn, compete and grow.
- Virtual Judge [5]: Virtual Judge is not a real online judge. It can grab problems from other regular online judges, such as Codeforces, Toph, Codechef, etc., and simulate submissions to other online judges. It acts as a platform to host competitive programming contests with a unique twist. vJudge can pull problems from various online judges. vJudge can pull problems from various online judges. vJudge uses pre-downloaded test cases from the original judge to evaluate your code locally.

Table 2.1: Mapping with complex problem-solving.

Platform	Live Con- test	Only Ac- cess For UIU Stu- dents	Own Problem Set	Free Contest Arrange	See Sub- mission Code
LightOJ	✓	X	✓	✓	✓
Virtual Judge	✓	X	X	✓	✓
CodeMarshal	✓	X	✓	X	X
CodeChef	✓	✓	X	X	✓
Codeforces	✓	X	✓	✓	✓
CodeKoro	✓	✓	✓	✓	✓

2.2.2 Related Research

Aldrich et al. [6] describes that feedback is required in introductory programming seminars to help students improve their knowledge and problem-solving skills. Traditional textbooks frequently lack enough exercise and feedback methods. The transition to e-learning has necessitated the use of automated feedback systems such as online judges such as Codeforces, UVa Online Judge, and CSES. These platforms augment traditional teaching approaches by offering immediate problem-solving and feedback. Scalability and instructor burden are still challenges when adding online judges into the programming curriculum. Future studies should focus on creating more advanced feedback mechanisms.

Petit et al.[7] proposed a platform named Jutge.org which is an online programming judge integrates features from both traditional online judges and learning management systems like Moodle. It offers instructors the ability to manage courses, add problems, monitor student progress and interact withg students. In this platform, problems are organized by topics and graded by difficulty enabling systematic progress irrespective of the programming language chosen.

Wasiket al.[8] address the function of online judge systems in programming competitions, with a focus on automated solution evaluation. They talk about design and implementation considerations, security, execution precision, and server architecture. They also investigate crowdsourcing’s potential in industrial optimization challenges, using platforms such as Kaggle.

Watanobe et al.[9] mentioned that Online Judge Systems (OJSs) are critical in both academics and industry for determining software correctness. However, they have difficulties in executing, assessing, and controlling solution codes. This paper describes the internal design and implementation of the Aizu Online Judge (AOJ), a popular OJS with more than 100,000 registered users and six million assessed program codes. The article analyzes functional and non-functional needs, discusses AOJ’s design, and presents insights and problems.

Brito et al. presented the development of a web-based platform called Codeflex for com-

petitive programming, aiming to enhance user experience skills through problem-solving and participation in programming tournaments. The model is built on a web services architecture with security measures like JSON Web Tokens(JWT) to manage access [10]. The backend, developed in Java with Spring framework, handles business logic and data manipulation, while a separate module compiles and executes code submissions in a sandbox environment. The platform supports multiple programming languages providing real-time feedback on submissions. Performance tests demonstrate a significant improvement in execution time with parallel processing.

Jannatul Ferdows et al. introduced a system for plagiarism detection in programming assignments, consisting of five main phases: tokenization with lexical analysis and parsing, k-gram hash generation using the Rabin-Karp algorithm, fingerprint generation with the winnowing method, similarity calculation and percentage estimation based on similarity and line matches [11]. The system is implemented using tools like Flex, bison and C++ and integrated with a contest management framework CUET online judge and experimental results demonstrate the system's time efficiency, correctness and effectiveness in comparison to the widely used MOSS plagiarism detection system. While the system provides efficient local server-based detection, there are limitations such as fixed parameters for W and K, fixed threshold values and potential vulnerabilities to certain types of plagiarism.

Pham et al. proposed an enhanced the existing DOMjudge online judge system focusing on adding functionalities tailored towards uplifting user experience, fostering discussion, and implementing a plagiarism detection mechanism. The authors modify DOMjudge to allow individual or team registration for contests, enhance user profiles with features like personal information management and team building and make adjustments to the administration site [12]. Moreover, they introduce a plagiarism detection system integrated into the administration page, utilizing preprocessing techniques, k-Grams, Hashing and the suffix Array algorithm aiming to detect code plagiarism.

Wenju et al. presented a broad analysis of the online judge system that proposes a new framework for OJ system and addresses the identified shortcomings of existing OJ systems lack of personalized feedback, neglect of code quality, code plagiarism and limited support [13]. The new framework incorporates these issues and leverages ML algorithms for multi-method grading, integrates code quality assessment tools and allows teachers to set their plagiarism boundaries highlighting parameters such as efficiency, correctness rate, repeatability and consistency.

Hui et al. [14] introduces an online judge system (YOJ) designed to support the teaching of programming courses, particularly aimed at novice programmers. The system facilitates independent programming practice for students and assists teachers in assigning homework on key programming concepts. Initially developed for an introductory programming methodology course (CSt), YOJ is flexible and can be adapted to support other programming courses as well.

Wilson et al.[15] indicate the issue of overwhelming choice on competitive programming

platforms like CodeChef. With so many coding challenges available, students struggle to pick the right ones for their skill level. This frustration can discourage them from learning. The study proposes using recommender systems, like those used by online stores, to suggest suitable challenges to students. They built a cloud-based system and tested different recommendation techniques using real data from these platforms. Their analysis showed that Singular Value Decomposition (SVD) performed the best. This method offered the most accurate suggestions (measured by RMSE) and worked fastest with large datasets. Overall, the research suggests that recommender systems, especially SVD, can be a valuable tool for competitive programming platforms. By guiding students towards appropriate challenges, they can improve the user experience and potentially help them learn more effectively.

Carrillo et al.[16] focuses on the development and implementation of a framework to formalize the working methodology of a competitive programming study group at Universidad Francisco de Paula Santander. Additionally, a web-based training platform was created to monitor the group's activities and progress. The platform aims to provide students with comprehensive tools for training, studying, practising, competing, and improving their programming and algorithmic skills, particularly in the context of competitive programming.

Wu et al.[17] provided a thorough solution that improves students' programming skills and engagement, and it successfully describes the construction and successful deployment of an Online Judge system in computer science education. The system addresses fundamental issues in experimental teaching. Innovative strategies for enhancing computer science teaching are proposed, including incorporating the ACM contest mode into the curriculum and establishing a friendly, competitive learning environment. These techniques show promise for both student outcomes and educational change.

Bez et al.[18] describe URI Online Judge Academic is a web system for programming classes. Professors can create classes and assignments with problems from a categorized online judge (URI Online Judge). This online system helps professors manage assignments and students practice coding with immediate feedback, making learning more engaging.

Wang et al.[19] mentioned Online Judge (OJ) systems are widely used in computer education to automatically judge students' programming solutions. These systems typically involve compiling code, running it with test data, and comparing the results to expected outputs. This paper introduces MetaOJ, a system designed to create a distributed and scalable online judge system from an existing one. MetaOJ achieves this by adding several interfaces to the original system and creating multiple copies of it. The authors mention that modifying an existing system like TUOJ only requires adding a small amount of code (around 3) and results in minimal performance loss (around 12) on a single instance. MetaOJ can also be integrated with cloud infrastructure to automate deployment. This system is particularly useful for OJ systems designed for large programming contests that experience performance bottlenecks due to a high number of contestants.

Francisco et al.[20] describe Students learning computer programming need to practice

solving problems by writing code. Instructors traditionally review this code, but this takes a lot of time. PROBOCA is a project that is developing a new online judge system specifically designed to help teach computer programming. This system is based on an existing system called BOCA, but PROBOCA adds features to make it easier for teachers to use. One feature is a problem database that lets teachers organize problems by difficulty and topic. Another feature automatically estimates how difficult a problem is based on how many students have submitted solutions and how many students found it difficult. This information can help teachers create problem lists that are appropriate for their students. PROBOCA also creates reports that show how students are using the system, which can help teachers identify students who are struggling. Other systems have been developed to help with teaching programming, such as systems that give students feedback on their code or allow students to work together on code. PROBACA looks like a promising approach to help teach computer programming because it combines automatic assessment with features that help teachers and students.

2.3 Gap Analysis

Although there are many online judge platforms available, there is no platform designed for UIU students due to which there are major gaps in its availability and relevancy for this group of users. Every leading platform widely known including Codeforces, CodeChef and LightOJ provides competitive programming instruments but does not have a special section for UIU students which makes it rather problematic to control learning outcomes and compare them inside the university environment. Conversely, all the platforms support live contests, however, such events can be arranged only by purchasing subscriptions, or having a premium account, which can be expensive to institutions such as UIU. Moreover, there is not a single social network that contains the UIU's students' exclusive section where a well-developed and organized learning environment to build up their tones and motivation are equally important for personal and professional development. To overcome these issues, CodeKoro provides a UIU specific, affordable, advanced, and well-suitable web service for UIU students so that their specific performance, skills, and certain interactions with others can be effectively measured uniquely in the university.

2.4 Summary

There will always be research on the development of online judge platforms to practice competitive programming skills as, the programming language is the main factor of today's and future generations' advanced life-leading root, also known to be the primary language for the digital world. From the existing literature, it is evident that standalone software applications exist but there is no optimal implementation of these technologies under one umbrella which is only dedicated to UIU students. Even though numerous of these similar applications cover the basis of language skills, students of UIU are not confident and

delightedly encouraged enough to participate in competitive programming contests as per the current requisition to keep up with our university's mission. To achieve our goal, it is dictated to build the online judge platform, "CodeKoro" and utilize it to acquire the best outcome of it in this field.

Chapter 3

Project Design

This chapter aims to provide the overall project view of our project, CodeKoro. Section 3.1 presents the Requirements Analysis which is subdivided into five subsections 3.1.1 consisting of the Functional and Nonfunctional Requirements, 3.1.2 representing the Context Diagram, 3.1.4 representing the Data Flow Diagram, 3.1.5 presenting the UI Design of our platform. Section 3.2, 3.3, 3.4 and 3.5 demonstrates the Detailed Methodology and Design, Project Plan and Task Allocation respectively. Finally, the chapter concludes with a summary.

3.1 Requirement Analysis

Requirement analysis is of utmost importance when it comes to developing a new project. The process entails multiple sessions with stakeholders where their needs and expectations are gathered and analyzed. The major objective of requirement analysis is to understand what users might expect from the project and how these expectations can be delivered from the developer's end. This entire procedure needs to be documented in a distinct and comprehensible way.

3.1.1 Functional and Nonfunctional Requirements

A system has two types of requirements: i) Functional Requirements and ii) Nonfunctional Requirements. This section contains short representations of the functional and non-functional requirements of our project.

Functional Requirements

Functional Requirements determine how the system can be utilized by the users. The features that are employed in the system and how each user role can interact with the features are described here. The course of action for data fetching and processing is also explained for each of the features. After analyzing the requirements, the features are taken into account based on what is feasible for the system.

- **Access Control:** To use the platform, users must be able to securely register and log in. Certainly, administrators, teachers, and students should all have varying levels of access permissions. Furthermore, depending on our user responsibilities, access to particular features and data needs to be controlled.
- **Problem Set:** The platform must facilitate an intelligible and comprehensive problem set covering various difficulty levels and topics in competitive programming as well as users should be empowered to view their current position in the platform's problem set, including completed and pending problems.
- **Contest Dashboard:** Instructors need to be able to view a dashboard with the status of every student taking part in competitions. Updates on students' progress, submissions, and performance during contests should be shown in real-time on the dashboard.
- **Live Contests:** The "CodeKoro" online judging platform must facilitate the planning and running of live programming competitions. Furthermore, it should be possible for students to submit their ideas, engage in real-time contests, and get prompt feedback on their work.
- **Compiler:** A compiler must be integrated into the platform to evaluate and execute users' code submissions and provide accurate feedback on whether the submitted code solves the given problem and meets the specified requirements.
- **Rank List:** A ranking list for contests, highlighting participant performance based on many parameters such as the number of issues solved, time consumed, and penalties incurred, must be generated and shown by the CodeKoro online judge platform.

Nonfunctional Requirements

The non-functional requirement of the CodeKoro ensures that it can provide the standard service that it requires to the competitive programming community of UIU. These requirements concentrate on quality characteristics that define user experience as well as enable product sustainability.

- **Security:** The application necessitates that CodeKoro guards user information, such as username and password and other details. They are protection of data access with user authentications, protecting the actual data by encryption and performing security assessment at regular intervals.
- **Performance:** The performance of CodeKoro should be fast, mean low latency especially during the occasion such as live contests. The platform should be able to accommodate concurrent users who will be interacting using this platform without compromising on the interaction.

- **Scalability:** The system architecture has to be designed such that could be horizontally scalable given that it is inherent that the number of users would increase with time. This covers areas of large databases and cloud based platforms to support end users during a rise in traffic.
- **Usability:** The platform web interface must be simple and easy for people with no or low technical skills in using technological interfaces. Essentials of navigation should therefore be easily understandable hence the need to incorporate guidance and/or indications.
- **Reliability:** This implies that CodeKoro should be very available with little to no time being experienced on the system. Data loss is one of the credible risks that must be availed to respond adequately; hence, reliable backup and recovery solutions.
- **Accessibility:** It means that CodeKoro should follow accessibility for people with the disabilities to provide correct build for the screen reader and other options such as keyboard navigation and text descriptions for the images.

3.1.2 Context Diagram

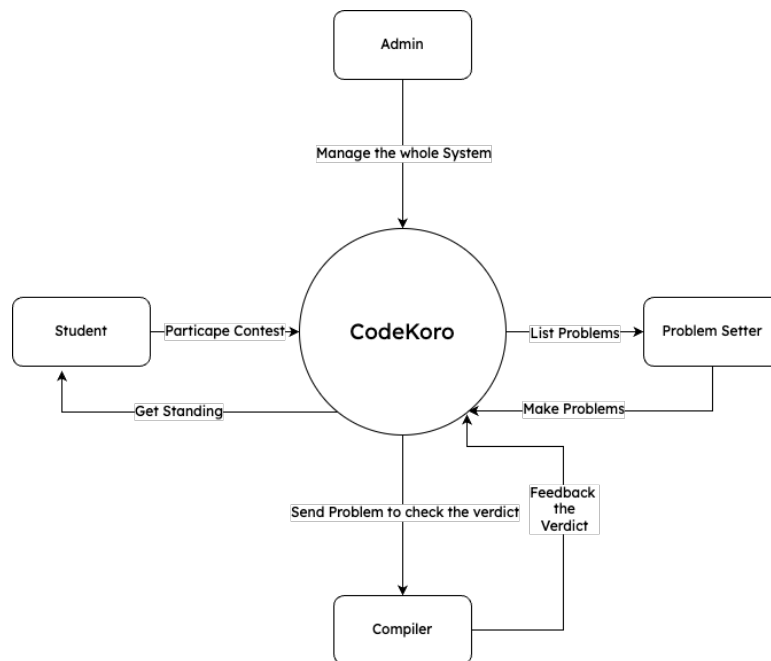


Figure 3.1: Context Diagram of the system

The context diagram for CodeKoro illustrates interactions between the system and its external entities: Program Admin, Student Participants, Problem Setter, and Compiler. Admin is in charge of user control, contests, and problem maintenance making the platform to run efficiently and safely. The target users include students who take part in coding contests, submit the solutions and are able to view rankings with real-time feedback

to enhance their experience. The Problem Setter is responsible for problem formulation and definition, statements, levels of difficulty, and test cases. These are basically available for practice and competitions. The Compiler also acts as the backend assessor to check a student's submissions against pre-determined invocations or tests and provide an outcome for display. Combined, all the above interactions form a kind of integrated and communicating platform thereby nurturing a devoted atmosphere for the essence of competitive programming at UIU.

3.1.3 ER Diagram

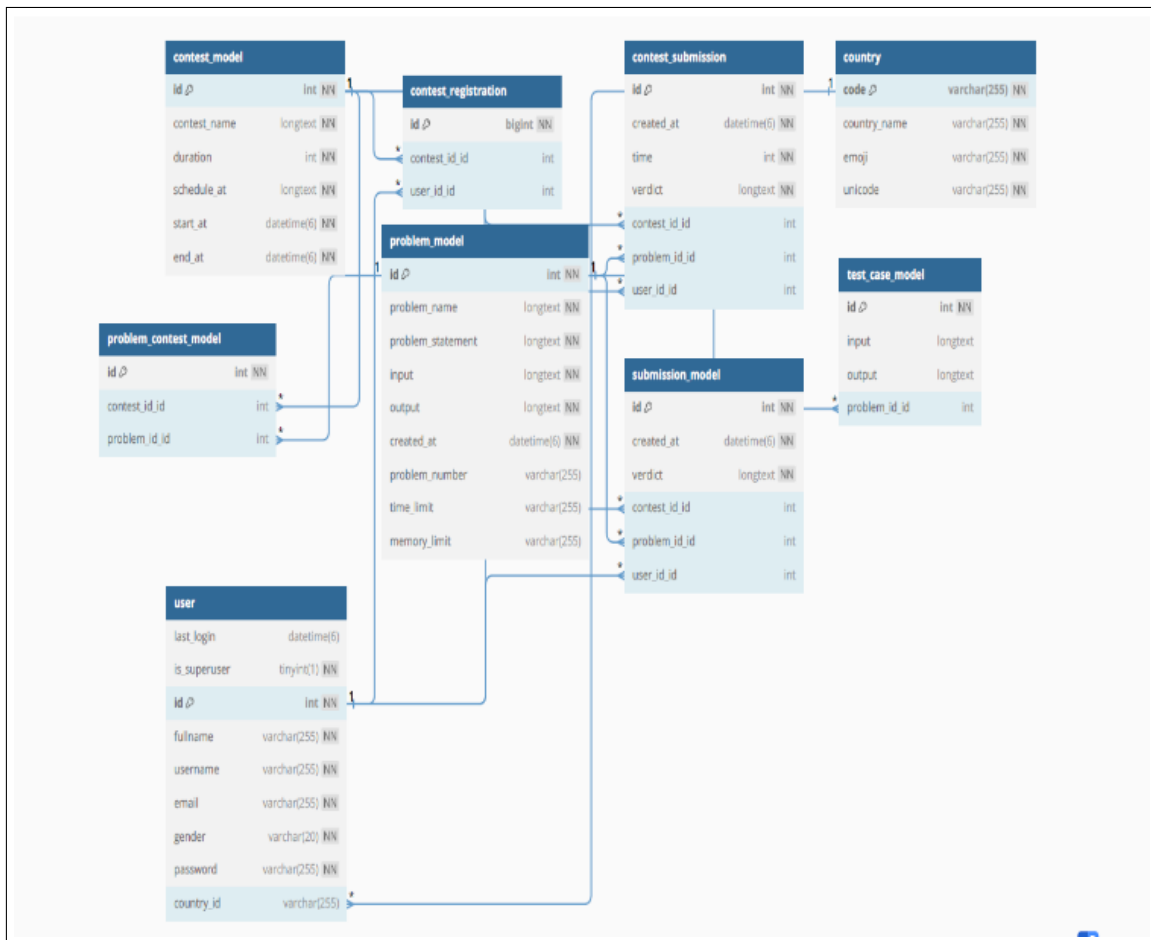


Figure 3.2: ER Diagram of the system

From the above explanation, the entities available in the CodeKoro ER diagram include User, Contest, Problem, Submission, TestCase, and Leaderboard where each of them offer the foundation for a competitive programming system. The User entity stores information about all users of the platform, be it students, admins, or problem setters, and only provides such attributes as user ID, name, email, and role. Contest specifies individual contests, which have the fields like contest id, name, duration and linked problem. The specifics of each coding problem are stored in The Problem entity; title, difficulty

level, and description are stored; Relations with TestCase are established for defining the input-output scenarios for precise submission assessment. Submission records each try made by students. The attributes of this table comprise of Sub ID, UserID, Problem ID, Time stamp, Status (example: Accepted, Rejected). Last but not least, the data from submissions are compiled on the leader board to categorize users as per the performance in the contest and create healthy competitiveness backed by organization. With these entities and relationships in place, the identified entities and relationships also help manage users, solve issues, and measure performance on CodeKoro.

3.1.4 Data Flow Diagram Level 1

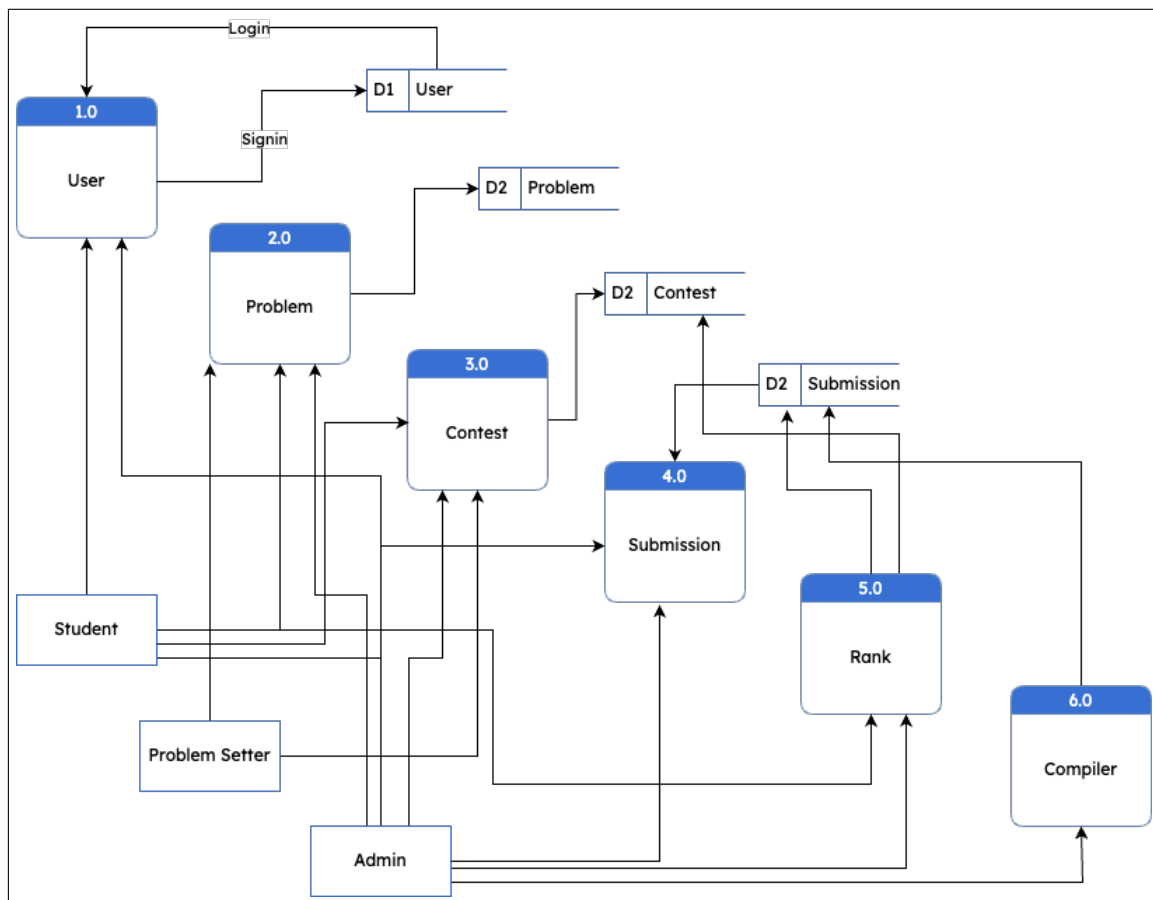


Figure 3.3: Data Flow Diagram of the system

The data flow diagram of CodeKoro maps the flow of the data in its fundamental activities such as user registration, submission of problems, handling of contests, and ranking list. Three types of users are students, admins, and problem setters; they register with logging in by providing valid information stored in the User Database. The users contribute problems to the problem setters update the problems in the Problem Database and make the problems available for the contests. Using information from the Contest Database admins create contests that are visible to students and they can join.

In contests, students send solutions in code, and these flow to the Compiler where submissions are analyzed against tests cases from the Test Case Database, then move to the Submission Database. The assessment criteria for the submission comprises success rate, time penalties, as well as the scores where students can attain the real-time Leaderboard ranking they desire. Further, the system logs the usage information allowing the admins to manage and monitor engagement, and in turn perfect the usage of CodeKoro's features.

3.1.5 UI Design

The UI Diagram for CodeKoro represents the main components of the application and how all these components are related, giving it a simple design in order to let the UI to flow with the most sense it will provide users with a simple user interface experience. The main layout is nice and free of excessive icons and the major objectives are to provide quick access to all the resources and info to students, admin and problem setters.

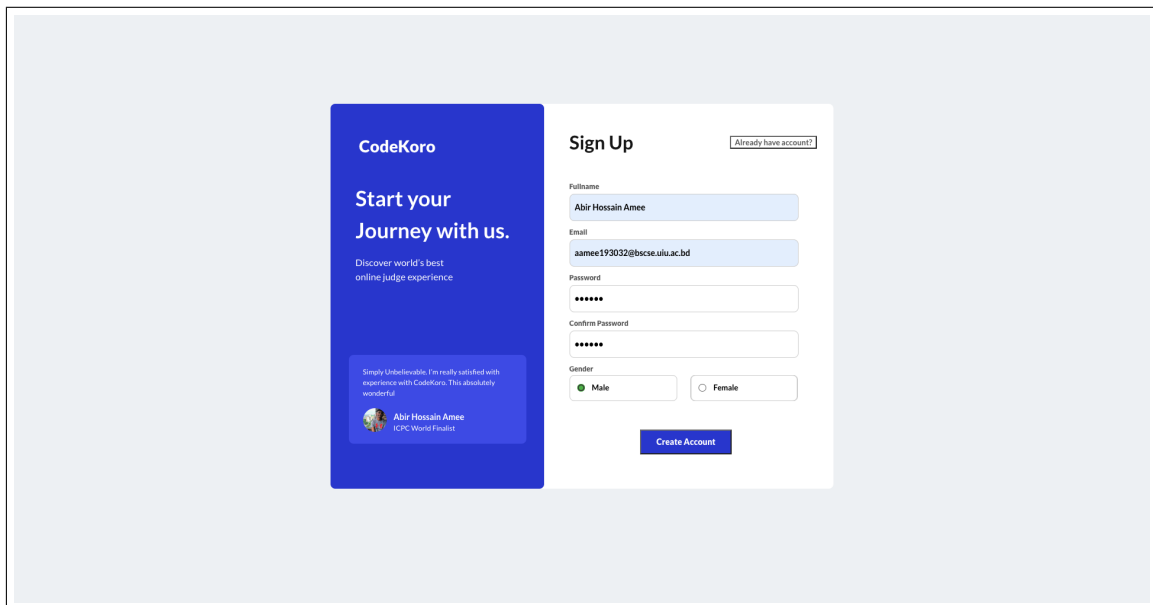


Figure 3.4: Signup Page

Figure 3.4 shows the Sign Up page, It leads a new starting point where the user can either sign up as a student. When their account is identified or after sign up; is redirected to the main account page, based on the privileges given to them.

Figure 3.5 shows the Login page, where users can securely authenticate to access CodeKoro platform.

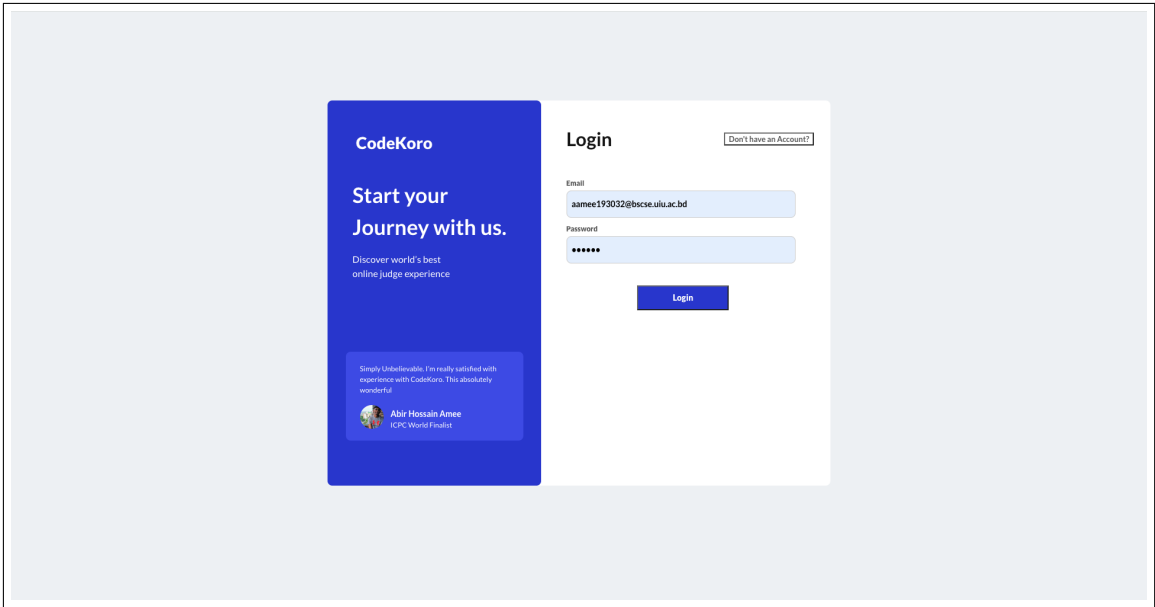


Figure 3.5: Login Page

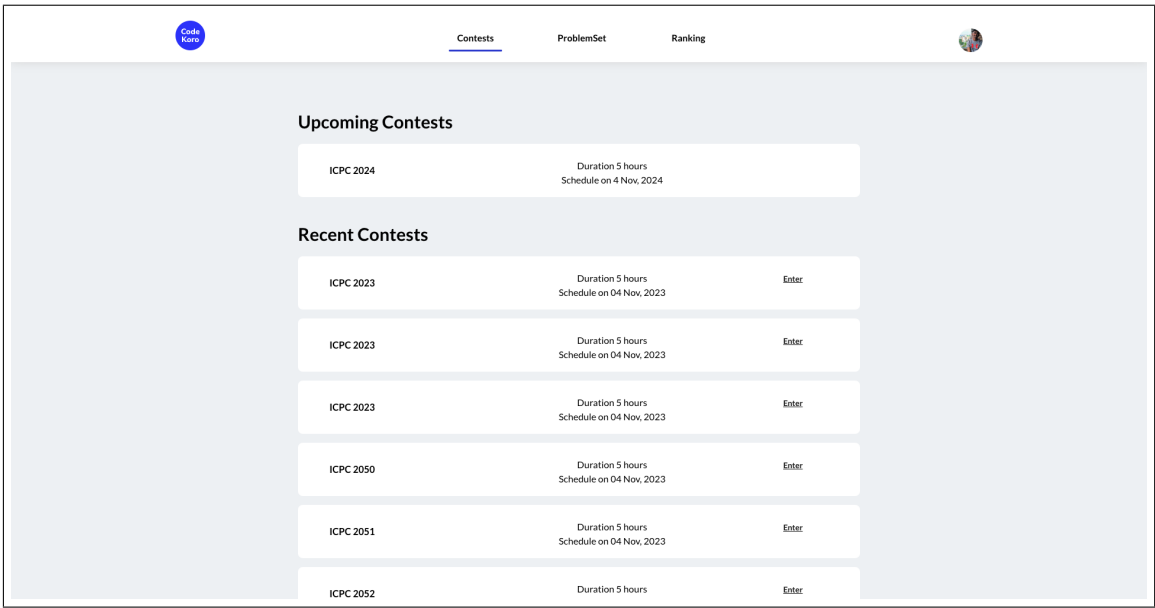


Figure 3.6: Home Page

Dashboard is one more common page for every user which is used for becoming familiar with the key areas of the site including the contests, problems, rankings and personal results. Students can see this as a list of events: every user also get options to interact with contests and problems including admins and problem setters.

Figure 3.6 displays the Home page, This page provides an overview of upcoming contests, running contests and user can register for new contest.

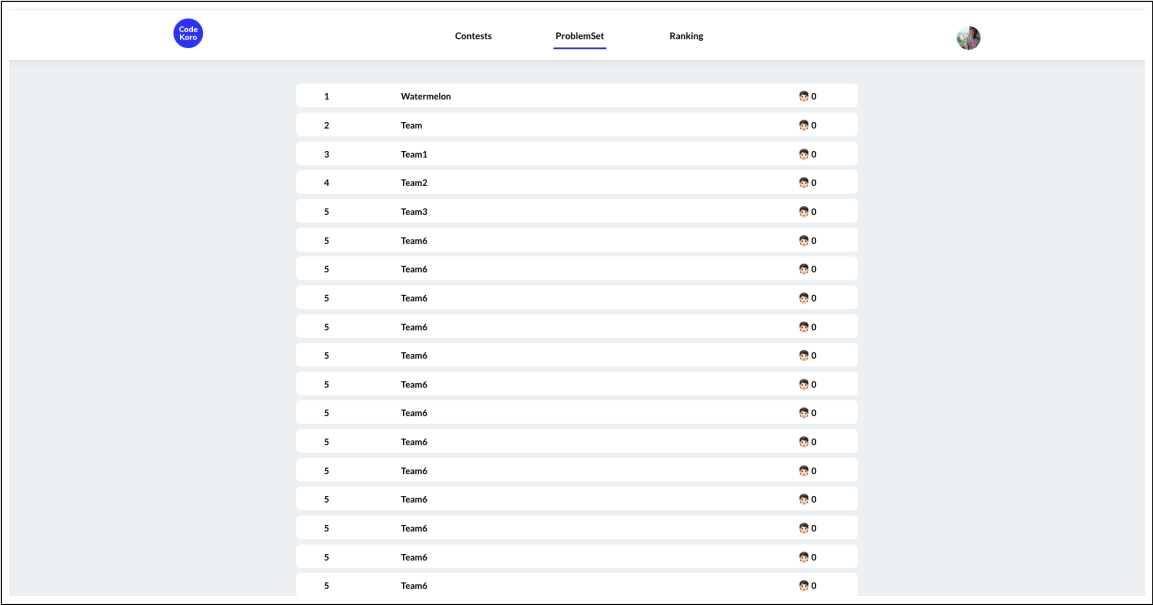


Figure 3.7: Problem Set

Figure 3.7 displays the Problem Set page, which lists all the available programming problems categorized by topics, difficulty levels, and other relevant filters. This organized view allows users to easily navigate and select problems aligned with their interests and skill levels.

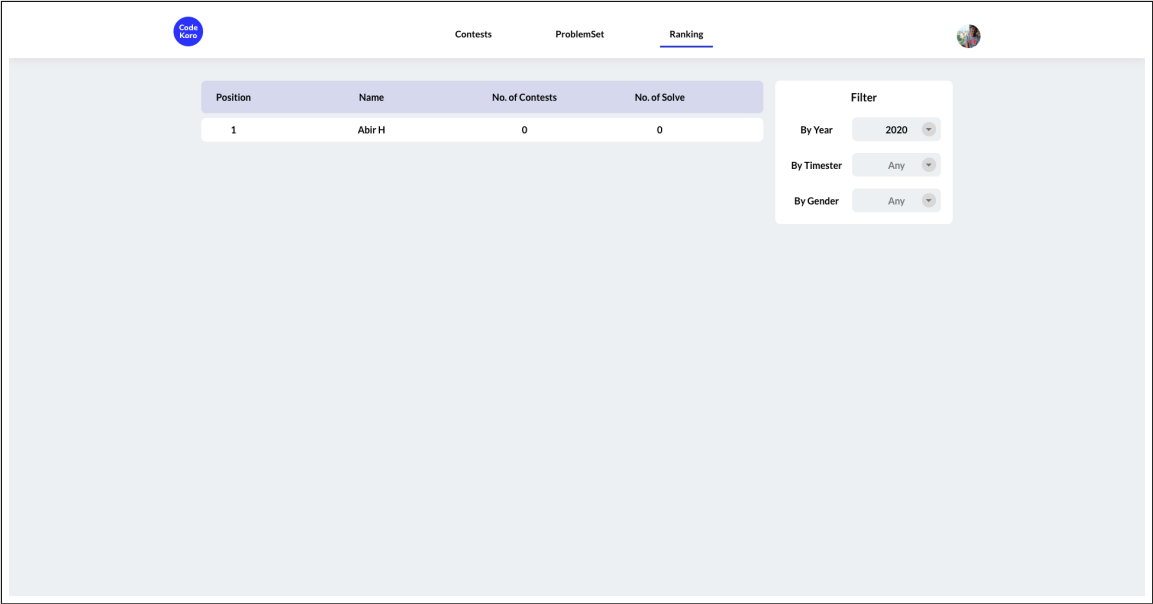


Figure 3.8: Ranking Page

Figure 3.8 illustrates the Ranking page, which offers a more detailed view of user rankings and performance metrics. This page allows students to analyze their strengths and weaknesses, identify areas for improvement, and track their overall progress within the platform.

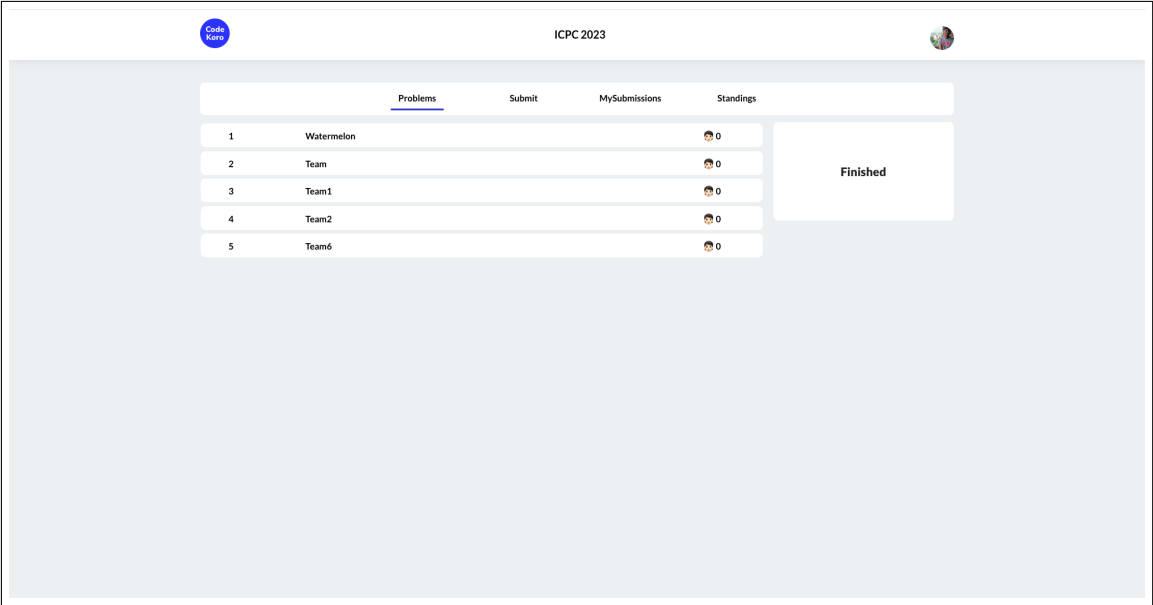


Figure 3.9: Contest Problems Page

Figure 3.9 showcases the Contest Problems page, this page provides information on the issues that are with regard to the problem level and field of study. Selected problems can be solved or attempted during a competition for practitioners/solvers while possibilities for creating new problems or modifying existing ones is available for the problem creators/editors.

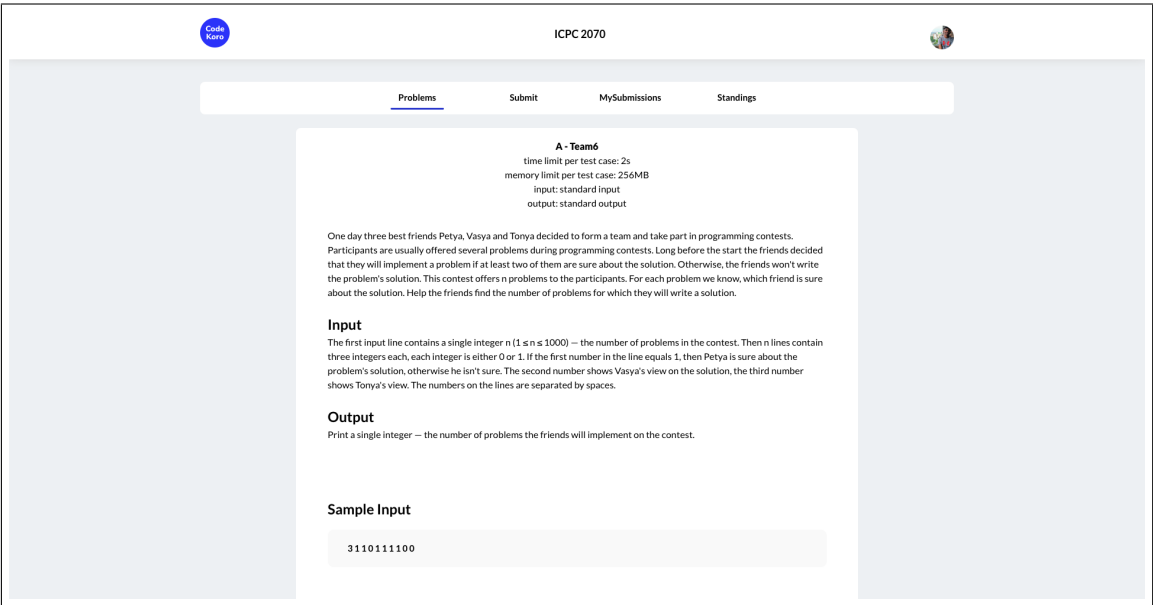


Figure 3.10: Details of a Problem

Figure 3.10 provides a glimpse into the details of a specific problem, including the problem statement, input and output formats, constraints, and sample test cases. This comprehensive information aids users in understanding the problem requirements and

developing efficient solutions.

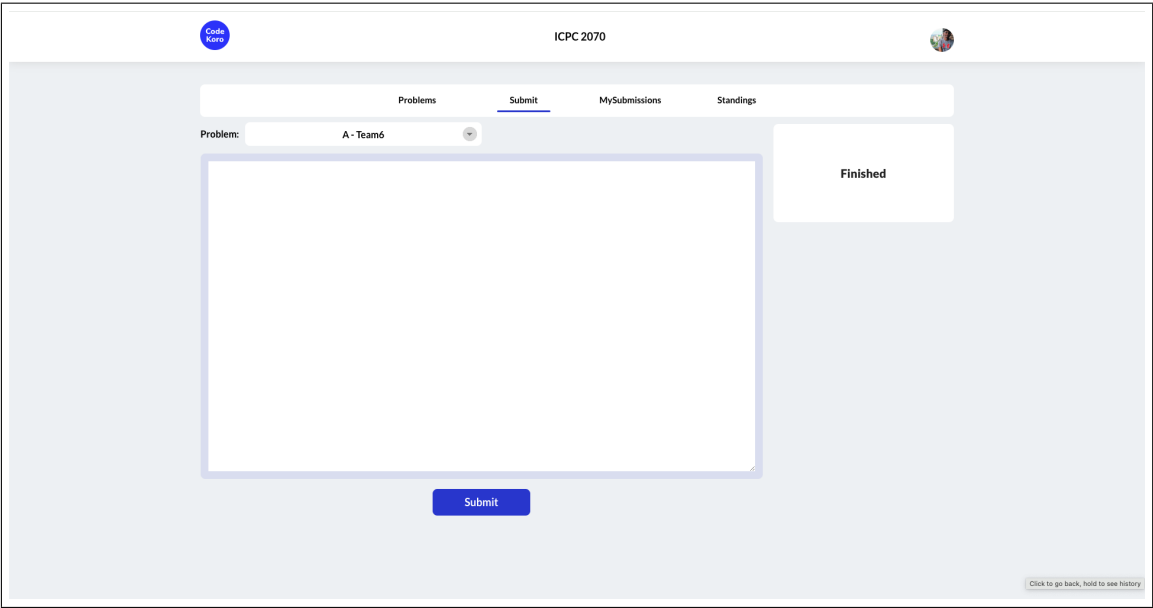


Figure 3.11: Problem Submit page

Figure 3.11 showcases the Problem Submit page, Here, students write in their solutions and then submit their code. There is a code editor and a Submit button for the student or the user to actually obtain the checking result based on the Test Case.

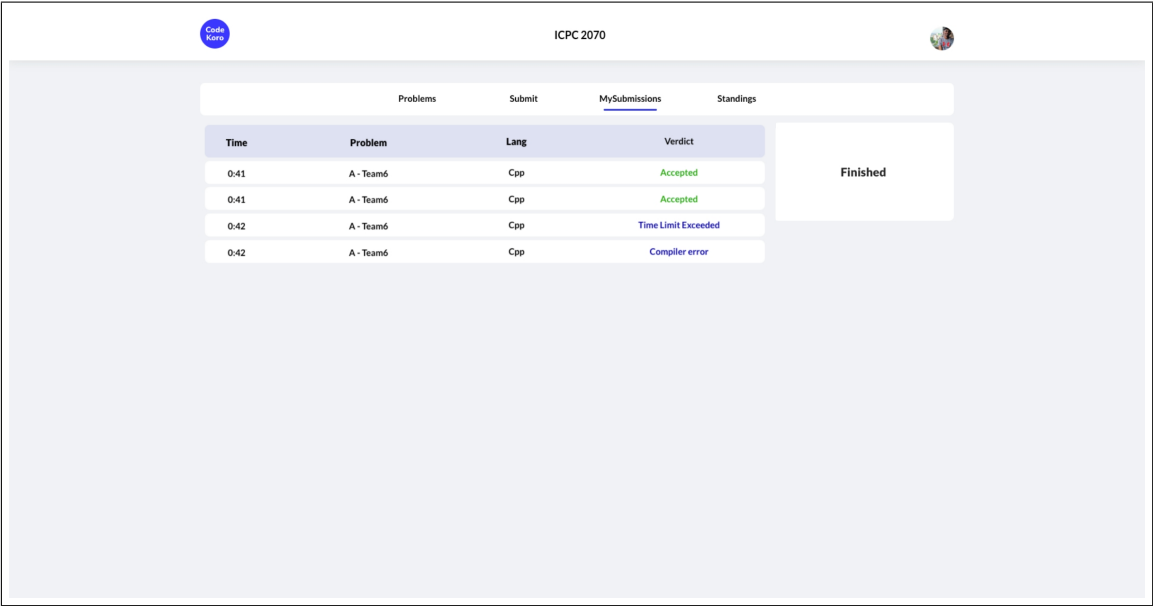


Figure 3.12: My Submission Page

Figure 3.12 displays the My Submission page, which provides users with a comprehensive overview of their submitted solutions. Users can track their progress, view problem statuses (such as pending or accepted), and access additional details about each submission.

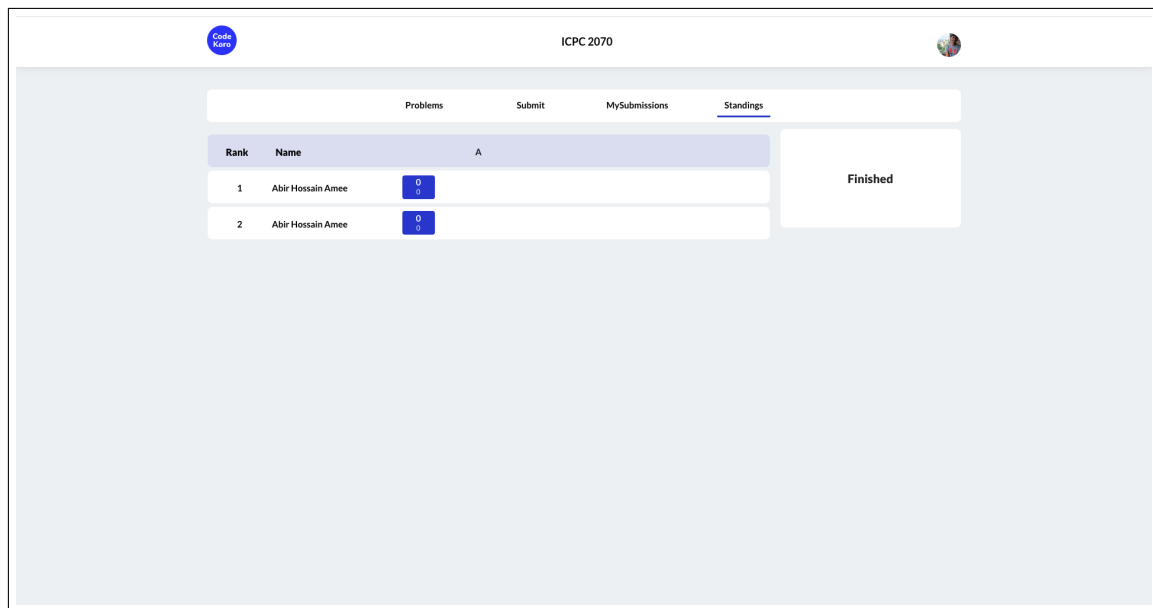


Figure 3.13: Standing Page

Figure 3.13 presents the Standing page, Outcomes of the contest concerning the problems solved and the time taken as well as scores will determine the student rankings showed on the leader board. This feature finds ways of developing competition and pushes all the students to be as best as they can be.

By offering a visually appealing and intuitive user interface, CodeKoro aims to foster a vibrant competitive programming culture at UIU, encouraging students to actively participate, collaborate, and excel in this domain while providing a seamless experience throughout their coding journey.

3.2 Detailed Methodology and Design

The flowchart below shows the detailed steps of how the overall processes work in our proposed system. The system is divided into three main sections: Contest Showing Page, Problemset Page, and Ranking. Contest Showing Page is the first page that a user sees. Here the user can sign up for a new contest or even participate in a current contest. If the user does not own an account, they will create one here. In our case, if the user enters an already existing contest, they will be lead to the Contest Page. This page shows the details of the contest, including the problems that need to be solved. The user can also submit their code for the problems on this page. The system judges submissions against test cases, marking them accepted or rejected. Users can track their submissions' status on My submission page and see their rankings on standing page. The Ranking Page shows the user's progress in the contest. The user can see their ranking by trimester, year, and gender.

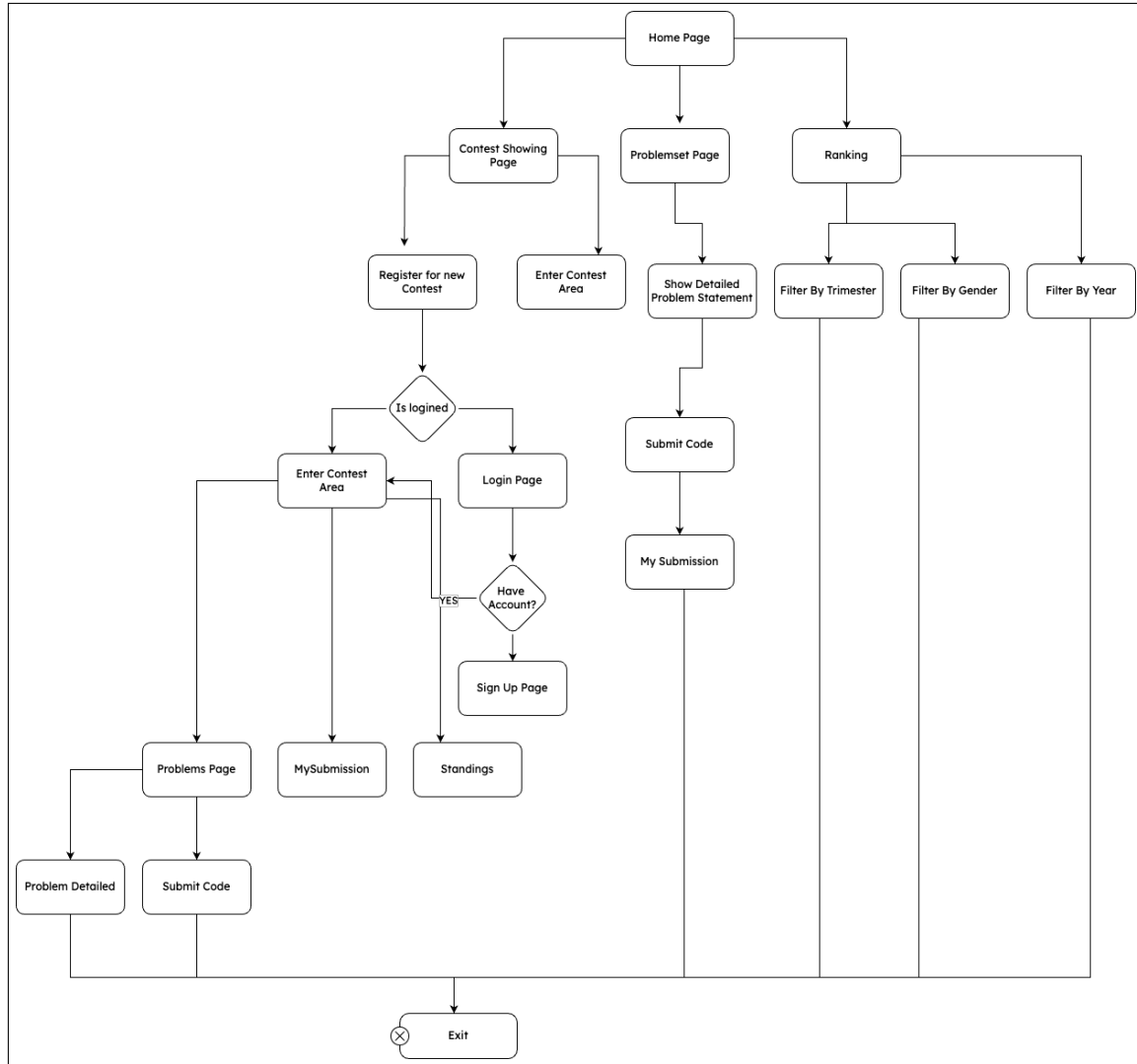


Figure 3.14: Flowchart of the Detailed Methodology

3.3 Project Plan

The tentative plan we developed to finish the first half of our project is described in this part. We met with our supervisor to generate ideas and finally decide on the project concept. Every member of the group reviewed the current applications and associated papers in brief for the background research. To investigate their working process, we next assessed comparable applications. These evaluations led us to investigate online judge system-related literature. Once the literature study was finished, we conducted a thorough benchmark study of related applications. Considering that elementary students are our target demographic, we developed the UI design appropriately. As each assignment was finished, we concurrently composed our report.

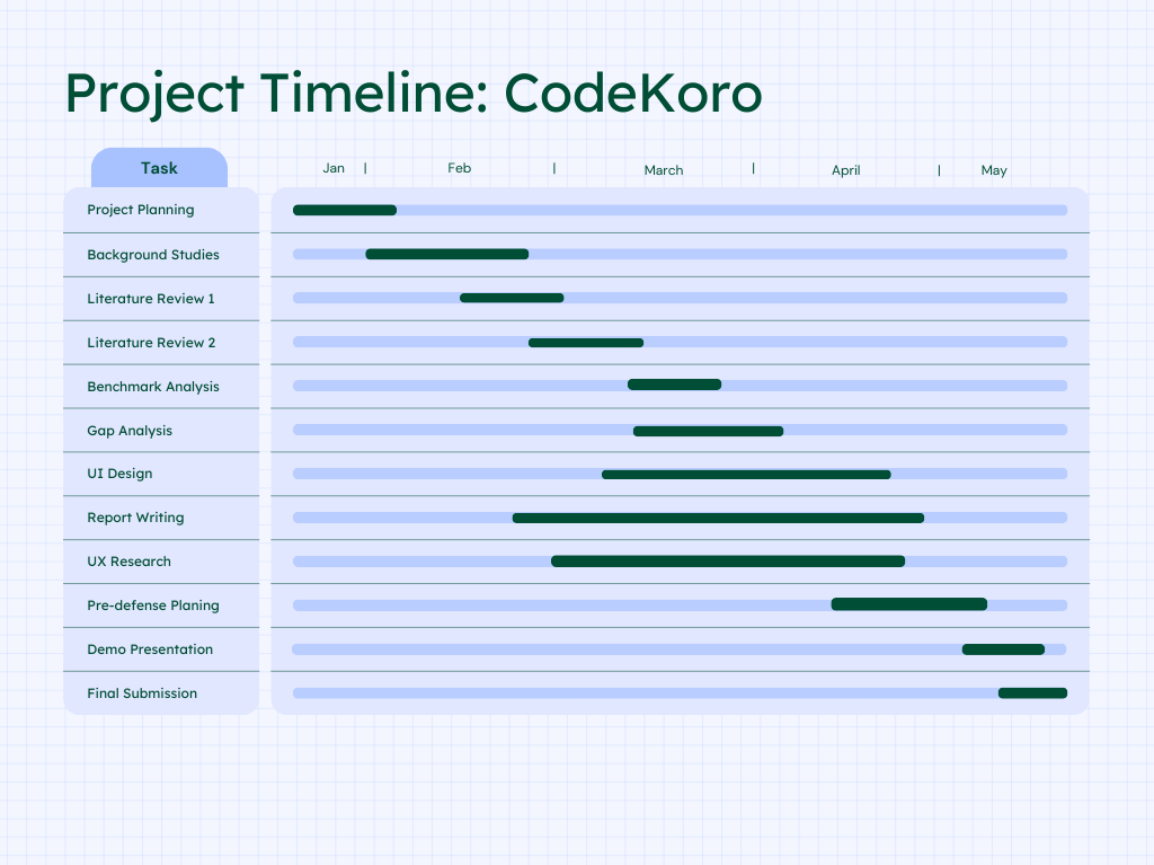


Figure 3.15: Gantt Chart for the Project Plan

3.4 Task Allocation

We organized our task allocation using the Work Breakdown Structure (WBS) format. Using this format helped us to outline all the work needed to be done and the time required for each task to be completed. Task allocation was based on a timeline spanning four months, from January 2024 to April 2024.

Table 3.1: WBS Package 1

WBS	Task	Allocated to	Start Time	End Time	Duration (In days)	%Done
1	Background Study					
1.1	Project Idea	Abir Hossain Amee, Md Iftekhar Hos- sain	28th January 2024	30th January 2024	3	100
1.2	Similar Project Analysis	All members	31st January 2024	3rd February 2024	4	100
1.3	Literature Review	All members	4th February 2024	5th March 2024	31	100
1.4	Gap Analysis	Upoma Roy, Sanjida Jan- nat Anan- naya, Arpita Mazumder	16th March 2024	18th March 2024	3	100

Table 3.2: WBS Package 2

WBS	Task	Allocated to	Start Time	End Time	Duration (In days)	%Done
2	Project Definition and Planning					
1.1	System Diagram	Abir Hos-sain Amee, Sara Ferdous Khan	19th March 2024	21st march 2024	3	100
1.2	Detailed Methodology	All members	22nd March 2024	23rd March 2024	2	100
1.3	UI Design	Md Iftekhar Hossain,Sara Ferdous Khan,Sanjida Jannat Anannaya	24th March 2024	31st March 2024	8	100
1.4	Complex Engineering Problems	Upoma Roy, Arpita Mazumder	18th April 2024	20th April 2024	2	100

3.5 Summary

In this chapter, we describe the rigorous design effort that went into creating CodeKoro, an online judging platform specifically for UIU competitive programming. Separating functional from nonfunctional requirements is the first step in the process of requirement analysis. Next, using the demo prototype designed in figma of important context, the chapter highlights the UI design, focusing on engagement and simplicity along with entity relationship and data flow diagrams. It also encompasses the work breakdown structure (WBS) for efficient distribution of tasks, adherence to standard and because it is methodical. The project outlined below is the Fourth Section of the Annual Social Media calendar, for the cultural plan, which runs from January to April, 2024. record and solemnize ensure an orderly and systematic approach to project design and execution.

Chapter 4

Implementation and Results

The objective of this chapter is to provide a synopsis of the implementation and result details of our project. Section 4.1 discusses the environmental setup for our project, Section 4.2 displays the testing and evaluation procedures that are used in our project, the results are further discussed in Section 4.3 and finally Section 4.4 ends with the summary of the chapter.

4.1 Environment Setup

This section includes the processes required to set up all the tools, software, libraries and configurations that were necessary for the project development. This setup process had to be kept consistent among all the developers to ensure that the versions of each tool were the same to avoid variances. The basic configurations were:

- **Operating System:** Windows
- **Code Editor:** Visual Studio Code
- **Node.js installation:**

Step-1: Download the Windows Installer (.msi) from the website: <https://nodejs.org/en/download/>

Step-2: Run the downloaded Node.js installer.

Step-3: After the installation is complete, go to the command prompt and type `node -v` to check if Node.js has been installed properly. If the version is shown, it means that Node.js has been installed properly.

- **npm installation:**

Prerequisite: Node.js must be installed.

Step-1: Go to the command prompt and type in the following command to download the latest version of npm: `npm install -g npm`

Step-2: After installation is complete, check if npm has been installed properly by checking the version using the command: `npm -v`

- **React.js installation:**

React.js does not require any installation, only one command is required to add React.js to the project.

Prerequisite: Node.js and npm must be installed.

Step-1: Go to command prompt and type the following command: `npx create-react-app my-app`

Step-2: Navigate to the project directory using the command: `cd my-app`

Step-3: Then start the server from the project opened in VSCode using the command `npm start` in terminal

- **Python installation:**

Step 1: Download the latest Python installer from the official website: <https://www.python.org/download>

Step 2: Run the downloaded installer. During installation, make sure to select the option “Add Python to PATH” to ensure Python is accessible from the command line.

Step 3: After installation, open the command prompt.

Step 4: Type `python --version` to verify the installation. If the Python version is displayed, the installation was successful.

Step 5: To check that pip, Python’s package manager, is also installed,

- **Django installation:**

Step 1: Open the command prompt and ensure your virtual environment is activated (if applicable).

Step 2: Install Django using pip by typing

Step 3: After installation, verify it by checking the Django version.

If the version number appears, Django has been installed successfully.

Step 4: To start a new Django project, type `Django projectname` with your desired project name.

- **Redis installation:**

Step 1: Download and install Redis. For Linux, use the command: `bash sudo apt-get install redis-server` download a compatible installer from a trusted source, as Redis is natively supported on Linux and macOS.

Step 2: Start the Redis server by running

Step 3: To verify that Redis is running, open a new terminal window

Step 4: To integrate Redis with Django, install the `django-redis` package in your virtual environment .

4.2 Testing and Evaluation

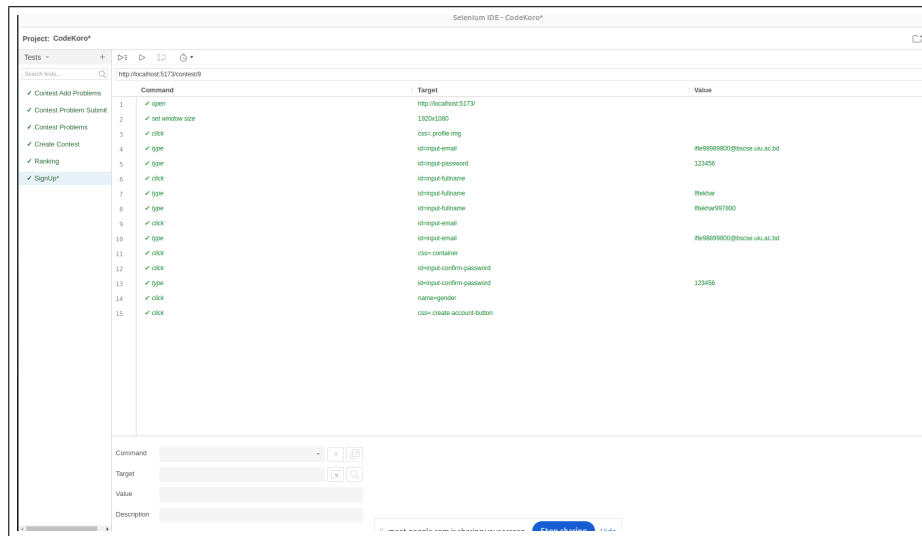
In this section, we have conducted a thorough UI/End-to-End testing using Selenium IDE on key functionalities of our system to simulate real user interactions and authenticate the system's functionality. Since the platform is to be used by children of ages 7-12, this testing stands out to be important for our platform as it ensures whether the platform's interface and user interactions function seamlessly. The testing evaluation results are showcased in Table 4.2, outlining the testing steps, the motive behind each selection, and our final verdict.

Table 4.1: UI/End-to-End test Selection

Test case ID	Feature	Test Steps	Selection Purpose	Expected Result	Verdict
TC01	SignUp	Landing page > Student SignUp > SignUp	Checking whether the signup working properly	Signup Successfully	PASS
TC02	Ranking List	Landing page > Ranking > Filter(By Year, By Trimester, By Gender)	Checking whether the ranklist working properly	Ranking List Submitted Successfully	PASS
TC03	Create Contest	Admin page > Create Contest > Contest Name > Duration, Start At > End At > Schedule At > Submit	Checking whether the contest creating properly	Contest Created Successfully	PASS
TC04	Contest Add Problem	Admin page > Add Problems to Contest > Contest ID > Problem ID > Submit	Checking whether the contest problem adding properly	Problem Added Successfully	PASS
TC05	Contest Problems	Landing page > NCPC 2024 > A	Checking whether the contest problem working properly	Contest Problem Created Successfully	PASS
TC06	Contest Problem Submit	Landing page > NCPC 2024 > Submit > Problem Select(B) > Code Submit > Submit	Checking whether the contest problem submission working properly	Contest Problem Submitted Successfully	PASS

4.3 Results and Discussion

The testing and evaluation phase of CodeKoro provided considerable evidence regarding the software's ability to support efficient competitive programming. Done with Selenium IDE for UI, and end to end testing, these tests ensured me that critical flows including sign Up, ranklist, contest creation and problem submitting sites worked smoothly as was expected. Thus test case TC01 confirmed that the SignUp and Login operation worked correctly, and offered a controlled environment for different types of user. TC02 also agreed that the ranking list provided the correct order dependent on filters such as the trimester, year or gender, which creates motivation and performance measure. TC03 TC04 proved that through contest management features, admin could create and manage contests to boost organization and offer time constraints. TC05 reaffirmed that the GUI is fully functional for navigating and retrieving selective contest problems and, similarly, TC06 corroborated the prompt feedback of the compiler by offering instantaneous feedback in terms of submissions, which supports learning and mastery in the process. User interface was understandable and thus the interaction with the system was easy in areas such as the contest dashboard, the problem set and the ranking.



Command	Target	Value
open	http://localhost:5173/	
set window size	1920x1080	
click	css=profile-link	
type	id=input-email	Ph98899800@hacore.ulu.ac.id
type	id=input-password	123456
click		
type	id=input-fullname	Ph4har
type	id=input-fullname	Ph4har987800
click	id=input-email	
type	id=input-email	Ph98899800@hacore.ulu.ac.id
click	css=container	
click	id=input-confirm-password	
type	id=input-confirm-password	123456
click	name=gender	
click	css=create-account-button	

Figure 4.1: Signup

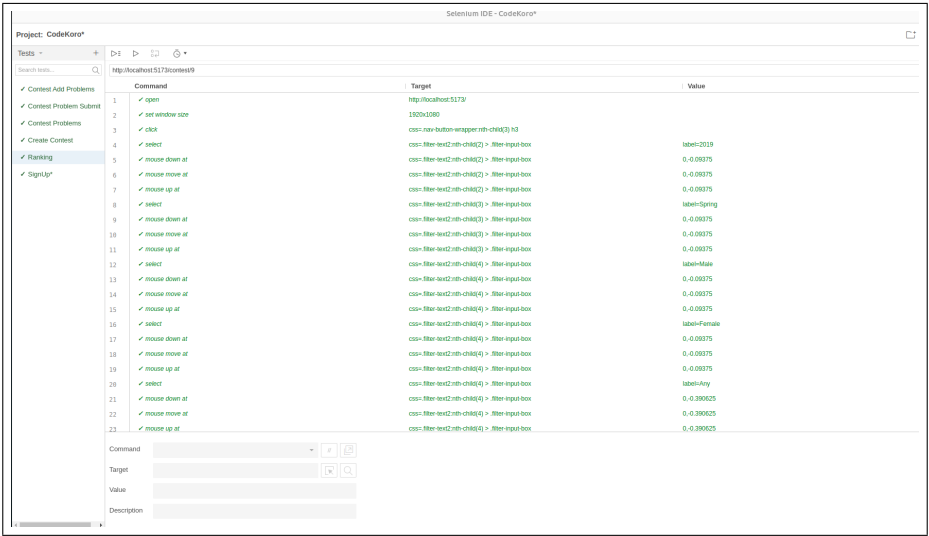


Figure 4.2: Ranking

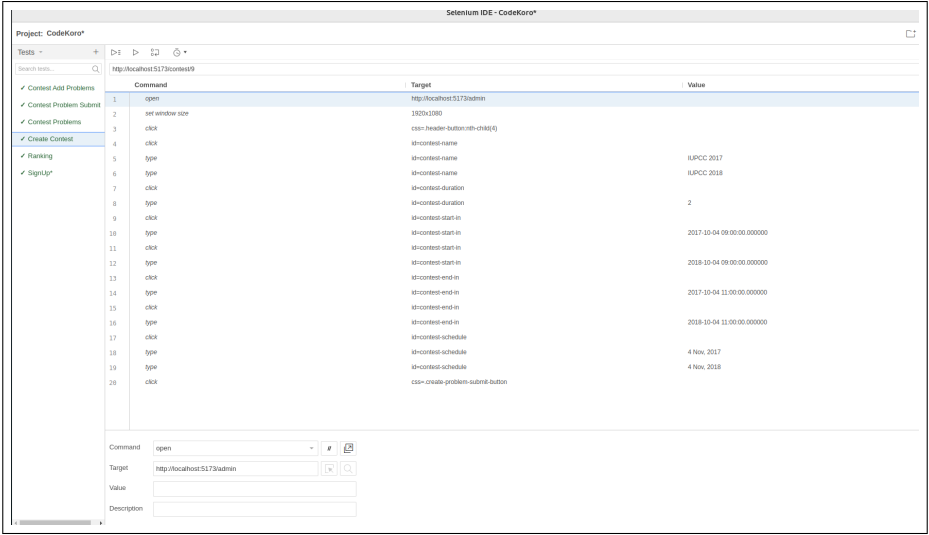


Figure 4.3: Create Contest

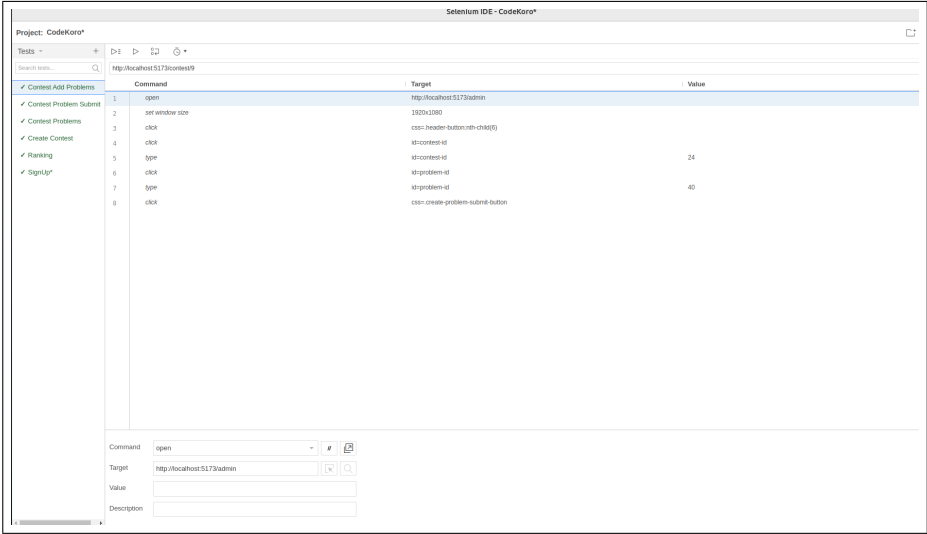


Figure 4.4: Add Problems to Contest

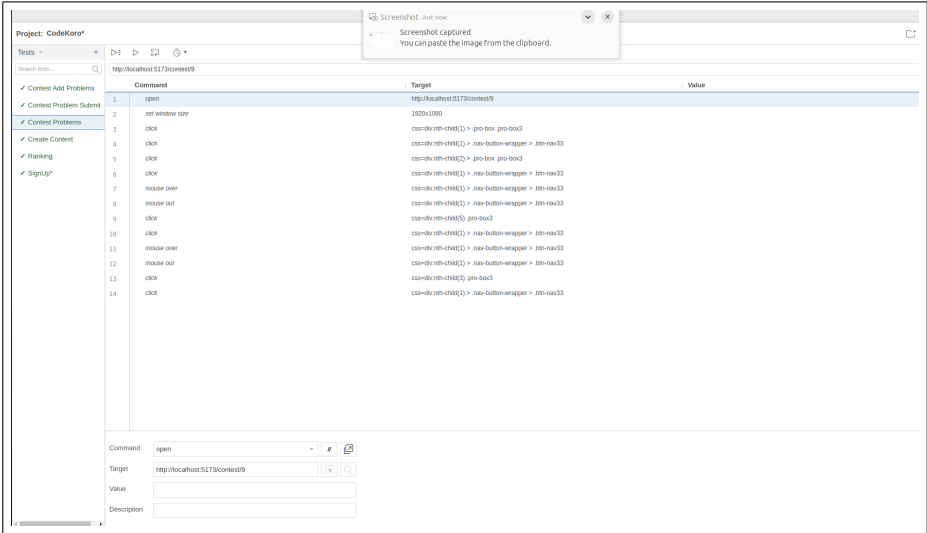


Figure 4.5: Contest Problems

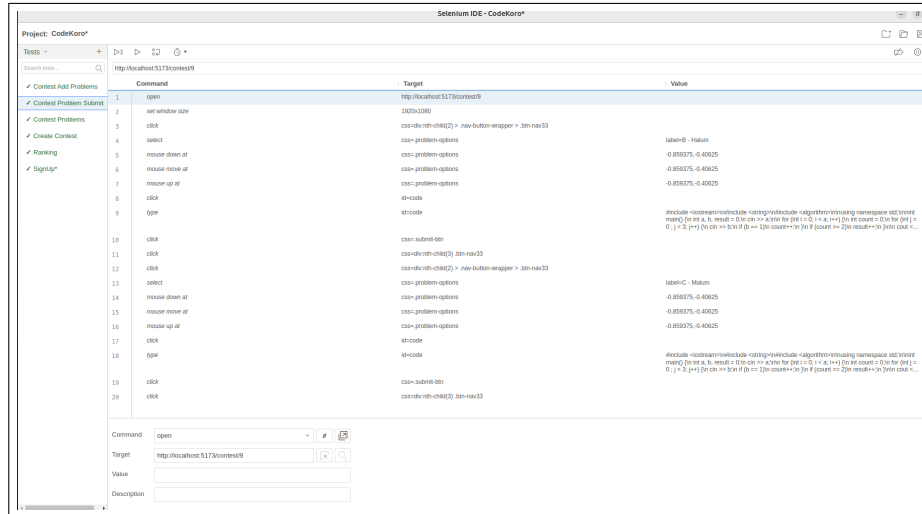


Figure 4.6: Contest Problem Submit

4.4 Summary

In this chapter, we explained how the development environment was set up and the tools used when developing CodeKoro to achieve cohesiveness. Selenium IDE was used to perform intensive UI and end-to-end that enabled us to confirm the effectiveness of the engagements between users and the platform. The results revolved around integrity sub operations such as secure user registration, rank display, contesting, and real-time feedback, all supported by test cases TC01 to TC06. Evaluation of these outcomes confirmed that the effectiveness of main functionalities of CodeKoro did not encounter any major problems, and hence establishing reliability and effectiveness of the proposed tool to complement the training of UIU's competitive programmers.

Chapter 5

Standards and Design Constraints

In this chapter, we described the tools and parameters that have been used to build up our project, along with the environment setup. We conducted UI testing using Selenium IDE to make sure there was thorough end-to-end interaction between the user and the system by replicating actual user scenarios to verify the system's functionality. The outcomes of these tests are analysed, concentrating on key functionalities like data input validation, lesson approval, and profile stats update. From this we can conclude that the main features performed as expected, with no major issues detected, thus confirming our system's robustness and flexibility.

5.1 Compliance with the Standards

Only mention the standards that are related to your project. This list is not complete. For each of the standards discuss the alternates with pros and cons and rationale of selection.

Ensuring compliance with the standards in our project demands adhering to specific guide- lines, regulations or standards established globally. The key points to maintain compliance include Documentation, Version Control, Coding Practices and Quality Assurance. For the software development life cycle, we used agile methodology to make our development process smooth and adaptable for all developers. In our project, we have given the afore- mentioned aspects great importance to develop an interactive and user-friendly interface so that it can be accessible to anyone connected with the internet.

5.1.1 Software Standards

To make CodeKoro more reliable and easily scalable, we maintained the software standards while in the development process. Based on Node.js for the backend and React.js for the frontend, while the main framework that will be followed is Django, CodeKoro has a clean, well-structured structure to achieve code readability and easier maintenance. Other coding policies that are observed in the program include use of appropriate and consistent naming standard and design of the code in the form of modules to enhance ease of future changes. It was easy to control the changes through Git versioning to provide the needed version

of the system to the team; also, proper message formatting in commit messages. The transaction data is secured using Bcrypt while the data exchange between the client and the server is protected by JSON Web Tokens (JWTs). The decision to follow the RESTful API design facilitates unambiguous and expected interaction amongst the various pieces. npm and pip for dependency management resolved installation procedure and ensured versioning. README files and comments in the code were included to facilitate easy onboarding as well as code maintainability to ensure that CodeKoro is secure and scalable to meet the competitive programming needs of UIU.

5.1.2 Hardware Standards

To ensure that CodeKoro runs as need and so that any of its users can have easy access, we set up some of the following hardware standards for development and deployment. The creation of the platform was focused on resource management, so compatibility with different devices, even those which have average performing hardware. For development workstations setup, it was recommended that each of them has a RAM of Not less than 4.0 GB, Processor being 1-2core or its equal and above and Storage not below 20.0 GB free space. Each of these specifications makes it possible to run multiple applications concurrently, such as local servers, databases to work with, and code editors, all without significant performance concerns. The recommended specifications regarding computer server deployment are at least 4 GB RAM and multi-core processors (4 or more), as well as cloud capabilities of going up as higher numbers are needed particularly during the live contests. This way, the system can handle multiple user interactions simultaneously, and the CodeKoro app will be operational, with desirable performance even as more and more people use it.

5.1.3 Communication Standards

When designing CodeKoro, having concise and unambiguous medium that was also easy to read and interpret was important to establish the right learning environment. Any written information provided for users was made simple and straightforward in order to let students easily use the platform. The language manipulation is clear and affirmative with an intent of assuring users that, without an acute misunderstanding, they ought to consider using the offered features. The icons used and the organization of the layout patterns were made standard throughout the sections to achieve usability but not too different from previous sections people have used. In order to increase interest and motivation contributors to CodeKoro, additional features like badges and progress bars, and leader boards are included. Thus internal communication standards that is applicable to the development team are the use of consistent documentation practices where code comments, documentation files and version control messages are well defined and all uses the same mediums. These practices are useful to keep the development processes aligned and facilitate aboard and work in general to benefit the long-term sustainable of the platform.

5.2 Design Constraints

The main design constraints, as it may be recalled, include making the CodeKoro platform more accessible to its users, as well as promoting its sustainability and ethical use. These are the same as the general limitations of the system and include the factors such as affordability since students and edu-corporations can barely afford to spend a lot of money. The platform supports use on low-cost devices making the platform accessible to as many people as possible. Organizational-Business factors support digital materials over conventional paper based documents, thus environmental considerations. Any issue of ethics is handled by only collecting basic details which are student name and ID number and institution issued email to avoid violating any privacy of the students. On a social level, this platform is designed to promote interactive, effective learning of the teaching concepts presented. In political terms, CodeKoro is not affiliated to any political thought or program, being strictly educational in its orientations. Finally, the growth is made permanent as new contents are developed to replace obsolete ones and by including customer feedback into the mix CodeKoro is made to sustain UIU's competitive spirits in programming.

5.2.1 Economic Constraint

The following are some of the economic impediments have been solved by the CodeKoro platform; As for the UIU students and faculty, the platform is cost efficient. Although the platform addresses real-life scenarios in which user budgets are tightly constrained, it has been developed in order to work on inexpensive hardware and is compatible with execution on a range of different hardware platforms. The broad compatibility also means that students from all different economic brackets can use it. Moreover, CodeKoro has an online platform which means that there is no need for additional software application which in turn does not require significant investment on infrastructure by the university. Unlike dealing directly with users, the platform effectively helps the company avoid individual charges, which are important for students since the platform is free for them and does not pose a high risk to UIU since it balances its operations costs for offering the service to the learners effectively. This supports economical sustainability through the utilization of easily scalable, cost efficient open source technologies such as Node.js and MongoDB which do not warrant long-term high maintenance costs. Altogether, these economic aspects make CodeKoro as an inexpensive tool to improve competitive programming abilities in the entire university.

5.2.2 Environmental Constraint

The CodeKoro platform incorporates constraints related to the use of funds according to sustainable development to limit and manage the platform's impact on the environment. Being an electronic product, CodeKoro goes for digital over printed product distribution

cutting on expenditure as well as an environmental concern. The design of the platform helps to minimize energy consumption by the servers using horizontal scalable and cloud-based architecture that controls the amount of resources depending on their load. Furthermore, the problem afloat the social platform eliminates the usage of physical resources or transportation for face-to-face events. Compatibility with low power machines including tablets and mobile phones makes CodeKoro uphold sustainability since its users do not need to engage in high power consumption gadgets to use the platform. On this basis of thought, therefore, CodeKoro can be well regarded as an eco-friendly tool to develop competitive programming talents, consistent with sustainable teaching-learning strategies.

5.2.3 Ethical Constraint

Ethical limitation that CodeKoro observes optimistically provides a secure and courteous experience to the UIU students. Data protection is the main consideration; the platform gathers only identification details, including student names and identification numbers, as well as university-issued email addresses only, avoiding unnecessary collecting of data in compliance with data minimization standards to safeguard the customer's data. CodeKoro is only open to UIU students and faculty thus keeps the environment befitting competition and learning. Furthermore, there is a Content Policy that bans any proximal content which is either unsuitable or potentially useful, making sure all the programming challenges as well as community interactions among the members obey the rules of ethics and effective learning. Another aspect that has contributed to the design of CodeKoro is respect of diversity and inclusion they have included features that can be understood by students of different classes and ability. To achieve these ethical considerations, the platform ensures students' rights of privacy, respect and equal opportunity within their given structure in safe environment learner environment.

5.2.4 Social Constraint

The social constraints of code Koro are to promote UIU student togetherness, cooperation, and support. Since it is an environment that allows multiple students enrolled in CodeKoro to work alongside one another, the program has to be creative enough to cater for all the levels of competency among the students while, at the same time offering the competitive edge for the advanced learners without any hint of imposition. The categorizing and the leader board that the platform incorporates are designed in a way to encourage the progress rather than pressure for higher ranking, and cooperation. There are components of using competition as a way of working with students, for example, contest with divisions or forums, that foster group cooperation and synergy. In like manner, CodeKoro does not allow unsavory behavior that border on disrespect by practicing and endorsing community norms that would ban the Partisan spirit of disrupting others, bullying, or negative competitiveness. CodeKoro enables constructive interaction and fosters a safe space that embraces participants within a community of learners who work together

improving individual and shared skills within the framework of competitive programming.

5.2.5 Political Constraint

On political limitation, CodeKoro is friendly, politically sensitive-free organization with a clear vision and goals, aimed at delivering educational experiences. Since the platform is aimed at improving the competitive programming skills of learners in UIU it does not allow content that has political messages, challenges that incite political affiliations, or community guidelines that are inclined to any political beliefs or partisanship. CodeKoro shields students of different origin from any problem that may contain politically or culturally sensitive issues; their programming problems and resources are selected with controls to prevent any sensitive issues which are likely to cause division among students. Further, access and participation portfolios remain organized by conforming to academic affiliation with UIU, thus excluding any form of bias attributed to political or social differentiation. It also makes it a point to keep CodeKoro politically neutral meaning this channel encourages learners to explore academic related goals with no interferences from current political systems worldwide making it a perfect source of information for students.

5.2.6 Sustainability

This way, CodeKoro is intended to build the university's sustainable source of educational information that lasts for as long as UIU exists. Implemented using largely simple, accessible software that does not require excessive investment in long-term maintenance, the platform means that with growing user numbers the system can expand while remaining similarly affordable. From the feedback obtained from the users and the recent trends in programming, content in CodeKoro is updated frequently to meet the needs of future students. They can also easily develop new characteristics enhancing the platform permanency and uses as well in the process. Environmentally, CodeKoro excludes anything printed, CD's, DVD's, etc and uses digital and cloud based platform which adapts resources according to usage. Also, CodeKoro also encourages ecological learning management in such a way that participants can join from other places while eliminating the need for more travels and resources. These sustainability efforts assist in establishing a strong and sustainable environment whose support keeps on addressing UIU's educational requirements in competitive programming.

5.3 Cost Analysis

The section expands the total cost associated with our project. The first four tables show the budget analysis on a monthly basis. The first table 5.1 shows the development costs required for the implementation of the project. Table 5.2 presents the infrastructure costs which are needed to make the platform available to the users. Table 5.3 explains the operational costs required to smoothly run the system and Table 5.4 shows the costs required

for miscellaneous purposes. Lastly, Table 5.5 presents the revenue model budgeting of our platform on a yearly basis.

Table 5.1: Development Costs

Item	Description	Monthly Estimated Cost (BDT)
Developer Salaries	Payment for developers (5 x 75,000)	375,000
UI/UX Designer	Designing user interface (2 x 60,000)	120,000
Project Manager	Overseeing project	300,000
QA/Testers	Ensuring software quality	200,000
Tools & Software Licences	IDEs, Design Tools, etc.	100,000
Third-party Services	Media Storage, etc.	500,000
	Total	1595,000

Table 5.2: Infrastructure Costs

Item	Description	Monthly Estimated Cost (BDT)
Hosting/Cloud Services	For servers and storage	75,000
Database	SQL	120,000
CDN	To deliver high quality media files	150,000
SSL Certificates	Establish secure connection	150,000
	Total	495,000

Table 5.3: Operational Costs

Item	Description	Monthly Estimated Cost (BDT)
Marketings	Advertising, SEO	100,000
Customer Support	Support staff and tools	60,000
Maintenance	Regular updates and bug fixes	250,000
	Total	410,000

Table 5.4: Other Costs

Item	Description	Monthly Estimated Cost (BDT)
Legal and Compliance	Ensuring data protection	100,000
Office Space	Rent for office space	75,000
Miscellaneous	Unexpected costs	150,000
	Total	325,000

5.4 Complex Engineering Problem

In this section, we intend to go into further detail in this part as to why our project represents a complex engineering problem. The problem-solving categories and their thorough descriptions are provided in Section 5.4.1 along with the engineering activities map in Section 5.4.2.

5.4.1 Complex Problem Solving

Seven characteristics are necessary to classify a problem as complex in order to solve it. The requirements are as follows:

- P1 Depth of Knowledge
- P2 Range of Conflicting Requirements
- P3 Depth of Analysis
- P4 Familiarity of Issues
- P5 Extent of Applicable Codes
- P6 Extent of Stakeholder Involvement

- P7 Interdependence

Each of the criteria is extensively explained in the subsections that follow, along with the extent to which they apply to our project. The characteristics that are in line with our project are shown in Table 5.5

Table 5.5: Mapping with complex problem solving.

P1 Depth of Knowl- edge	P2 Range of Con- flicting Require- ments	P3 Depth of Analysis	P4 Familiarity of Issues	P5 Extent of Applicable Codes	P6 Extent of Stake- holder Involve- ment	P7 Inter- dependence
✓	X	✓	✓	✓	✓	✓

5.4.2 Engineering Activities

Engineering tasks have five characteristics. The challenging engineering tasks within the framework of our project are shown in Table 5.6. The following lists the characteristics of engineering activities along with a project map including these activities.

- A1 Range of resources
- A2 Level of Interaction
- A3 Innovation
- A4 Consequences for society and environment
- A5 Familiarity

Table 5.6: Mapping with complex engineering activities.

A1 Range of re- sources	A2 Level of Interac- tion	A3 Innovation	A4 Consequences for society and environment	A5 Familiarity
✓	x	✓	✓	✓

5.5 Summary

In the this chapter, the standards and design constraints followed in the development of CodeKoro were discussed. Measures like standard coding practices like how to write secure code, using version control, and modular design principles were observed making the platform as reliable, scalable and maintainable as possible. The hardware standards were established to provide the stability and the adaptability that would facilitate development for both development computers and the user/operating environment, as well as the capacity to handle multiple user simultaneously. Communications standards guaranteed straightforward accessibility so that the interface would not be complicated by confusing design; the guidelines were clear and easy to comprehend for the users, including students and staff members. Also, ethical, social, and environmental standard adherence was maintained for data protection, representation of members, and global, environment preservation during the project. s help CodeKoro to run effectively as a competitive programming platform for UIU students and faculty with technical and ethical standards.

Chapter 6

Conclusion

This chapter briefly revisits the undertaking of the CodeKoro project to highlight its major milestones, as well as discuss its implications, limitations and possible directions for improvement. CodeKoro was designed initially in an attempt of making it a dedicated platform for more competitive programming expectation in the students of United International University (UIU). This chapter explores the ways the formulated project objectives were achieved, identifies the limitations of the platform, and provides recommendation for enhancing the educational value and functionality of CodeKoro.

6.1 Summary

Through this assessment, the CodeKoro platform achieves the primary ontological goal of establishing a competitive programming culture at the UIU through practice. Ensuring that the program is accessible and environmentally friendly, CodeKoro offers the students the platform to try different levels of programming problems, join live competitions, monitor results, and discuss their projects with other students. Not only does CodeKoro help student academic development but it also fosters both teamwork and problem-solving which are relevant in the computer science field at UIU. Such aspects as feedback, performance evaluation or even interface and navigation are also available on the platform, making the LeetCode platform as a whole a competitive programming platform for students. CodeKoro has thus achieved its intended purpose: to enhance students coding skills, employability and self efficiency in programming through a designated site.

6.2 Limitation

However, in helping UIU establish a competitive programming team, the following are some of the drawbacks that currently hinder the maximum functionality of CodeKoro. Initially, the ability of the platform's compiler is not vast enough in embracing numerous programming languages, and the management may reduce the addressing of numerous and complex testing cases that can be a challenge for students using less common programming

languages or solving tasks with higher levels of difficulty. Moreover, there are several downsides to this platform; it has poor scalability as of now and performance might decrease if the number of users or activities in contests rise up considerably. Another is that the feedback system in most of the learning apps is quite crude and often it only either confirms that you are right or wrong without giving any clue or spending time explaining about the erroneous part. This could minimize the chances and capacity of the students in adapting from the mistakes they make as well as their consideration in relation to problem solving approaches. As of now, CodeKoro also has no mobile support, which prevents people who might want to participate using phones or tablets only. Last but not the least; CodeKoro has integrated result reporting for each contributor but lacks detailed analysis of several sub-abilities that students and instructors can use to track their progress. Such shortcomings reveal directions for further development to establish an additional, improved, more customer-oriented platform that would be more relevant for more engaged students of the UIU competitive programming team.

6.3 Future Work

The future plans for code Koro will be to enhance the identified features and create a better environment for UIU's clued up competitive programming population. The first will be expanding the general capacity of the platform to thousands of participants and increasing the complexity of live contests. Enhancing the compiler skills for recognising the multiplicity of program languages and even rigorous testing forms is vital for expanding the platform use. Moreover, they could also algorithm expert suggestions for problems based on users' skill and learning capabilities so learners get more efficient means of self improvement for the platform. To encourage learning even more, the CodeKoro can suggest the hint button which will provide users with solutions, explanations, and suggestions on how to solve a few problems. This would give the students a more detailed guided learning instead a feedback of either right or wrong. Current analytics could include new options for students and teachers; the results, performance trends, and midpoints and weaks of performance could be explored more in details. Based on positive education learning and community participation, CodeKoro might be featured as team challenges, peer feedback and forums. Of course, the last addition of mobile compatibility will enhance the availability of the site, enabling students to use it via multiple platforms. These enhancements would foster changes in code koro towards becoming a more useful, useful, and interactive resource for UIU students.

References

- [1] Codeforces. Most popular website for competitive programming, 2024. Accessed: 2024-01-29.
- [2] LightOJ. aims to allow competitive programmers and other interested in solving algorithmic problems to practice online., 2024. Accessed: 2024-01-29.
- [3] Codechef. offering a variety of features for programmers, 2024. Accessed: 2024-01-29.
- [4] Toph. one-stop shop for competitive programming, 2024. Accessed: 2023-02-01.
- [5] Virtual Judge. Best for arrange a contest, 2024. Accessed: 2024-02-01.
- [6] Aldrich Ellis ASUNCION, Brian Christopher GUADALUPE, and Gerard Francis ORTEGA. The abc workbook: Adapting online judge systems for introductory programming classes.
- [7] Jordi Petit, Omer Giménez, and Salvador Roura. Judge. org: an educational programming judge. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, pages 445–450, 2012.
- [8] Szymon Wasik, Maciej Antczak, Jan Badura, Artur Laskowski, and Tomasz Sternal. A survey on online judge systems and their applications. *ACM Computing Surveys (CSUR)*, 51(1):1–34, 2018.
- [9] Yutaka Watanobe, Md Mostafizer Rahman, Taku Matsumoto, Uday Kiran Rage, and Penugonda Ravikumar. Online judge system: Requirements, architecture, and experiences. *International Journal of Software Engineering and Knowledge Engineering*, 32(06):917–946, 2022.
- [10] Miguel Brito and Celestino Goncalves. Codeflex: A web-based platform for competitive programming. In *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE, 2019.
- [11] Jannatul Ferdows, Sumi Khatun, Miftahul Jannat Mokarrama, and Mohammad Shamsul Arefin. A framework for checking plagiarized contents in programs submitted at online judging systems. In *International Conference on Image Processing and Capsule Networks*, pages 546–560. Springer, 2020.

- [12] Minh Tuan Pham and Tan Bao Nguyen. The domjudge based online judge system with plagiarism detection. In *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 1–6. IEEE, 2019.
- [13] Wenju Zhou, Yigong Pan, Yinghua Zhou, and Guangzhong Sun. The framework of a new online judge system for programming education. In *Proceedings of ACM turing celebration conference-China*, pages 9–14, 2018.
- [14] Hui Sun, Bofang Li, and Min Jiao. Yoj: An online judge system designed for programming courses. In *2014 9th International Conference on Computer Science Education*, pages 812–816, 2014.
- [15] Wilson Julca-Mejia and Herminio Paucar-Curasma. A cloud based recommender system for competitive programming platforms with machine and deep learning. In *Anais do VIII Congresso sobre Tecnologias na Educação*, pages 11–20. SBC, 2023.
- [16] GY Lázaro Carrillo, AM Delgado León, MJ Vera Contreras, and FH Vera-Rivera. Development of a technological platform for teaching and training in competitive programming contests. In *Journal of Physics: Conference Series*, volume 1386, page 012145. IOP Publishing, 2019.
- [17] Jianhua Wu, Shuangping Chen, and Rongrong Yang. Development and application of online judge system. In *2012 international symposium on information technologies in medicine and education*, volume 1, pages 83–86. IEEE, 2012.
- [18] Jean Luca Bez, Neilor A Tonin, and Paulo R Rodegheri. Uri online judge academic: A tool for algorithms and programming classes. In *2014 9th International Conference on Computer Science & Education*, pages 149–152. IEEE, 2014.
- [19] Miao Wang, Wentao Han, and Wenguang Chen. Metaoj: A massive distributed online judge system. *Tsinghua Science and Technology*, 26(4):548–557, 2021.
- [20] Rodrigo Elias Francisco and Ana Paula Ambrosio. Mining an online judge system to support introductory computer programming teaching. In *EDM (Workshops)*. Citeseer, 2015.