Competitive Programming Contest Listing Platform For Students And Developers

Prof. Sonu Khapekar[1], Vaibhav Pangare[2], Mahesh Pohekara[3], Pratik Potdar[4]
Computer Engineering Department[1,2,3,4]
Nutan Maharashtra Institute of Engineering and Technology, Pune, Maharashtra[1,2,3,4]

ABSTRACT:
The Contest Lister project aims to revolutionize the way coding enthusiasts engage with coding contests, hackathons, and hiring challenges across various online platforms. With the exponential growth of coding competitions, enthusiasts face the daunting task of keeping track of events, often leading to missed opportunities and disorganized participation[8]. In response to this challenge, the Contest Lister project presents a comprehensive solution that aggregates contest information from prominent platforms like CodeChef, HackerRank, and LeetCode into a centralized platform. By leveraging modern web technologies and APIs, the system provides users with real-time updates on upcoming contests, intuitive filtering options, and a user-friendly interface for seamless navigation. This paper outlines the architecture, methodology, results, and future prospects of the Contest Lister project, offering insights into its potential to streamline contest discovery and enhance user engagement in the coding community.

Keywords—Coding contests, Competitive programming, Contest Lister, System architecture, API integration, User interface.

INTRODUCTION

The introduction of the Contest Lister project contextualizes the significance of coding contests, hackathons, and hiring challenges in the realm of software development and competitive programming. It begins by acknowledging the growing popularity of these events as platforms for skill development, networking, and recruitment within the programming community[10]. Furthermore, it highlights the diverse range of coding platforms hosting such events, including CodeChef, HackerRank, LeetCode, and others[2].

The introduction also emphasizes the challenges faced by coding enthusiasts in keeping track of the multitude of contests and opportunities available across different platforms. This includes issues such as scattered information, inconsistent event schedules, and the difficulty of finding contests that align with individual preferences and skill levels.

Moreover, the introduction outlines the motivation behind the Contest Lister project, which is to address these challenges by providing a centralized platform for contest discovery and engagement. By aggregating contest information from various platforms into a single interface, the project aims to streamline the process of finding and participating in coding contests, hackathons, and hiring challenges.

I. OBJECTIVE

The primary objective of the Contest Lister project is to develop a centralized platform that facilitates the discovery and participation of coding contests, hackathons, and hiring challenges for coding enthusiasts. The project objectives are as follows:

Aggregation of Contest Information: To aggregate contest data from various coding platforms such as CodeChef, HackerRank, and LeetCode in real-time.

User-Friendly Interface: To design an intuitive and user-friendly interface that allows users to explore, filter, and engage with contest listings effortlessly[1].

Real-Time Updates: To provide users with real-time updates on upcoming contests, hackathons, and hiring challenges across multiple platforms.

Customizable Filters: To implement customizable filters that enable users to refine their contest search based on criteria such as platform, duration, difficulty level, and contest type.

Integration of External APIs: To integrate APIs from coding platforms to fetch contest data efficiently and ensure data accuracy and reliability.

Evaluation and Testing: To evaluate the system's performance and usability through rigorous testing and user feedback, ensuring a seamless and satisfying user experience.

II. SYSTEM ARCHITECTURE

The system architecture of the Contest Lister project encompasses both the frontend and backend components, designed to seamlessly interact to provide users with a comprehensive contest discovery experience. Here's an expansion of the system architecture:

Frontend Architecture:
The frontend architecture of the Contest Lister project is designed to deliver an intuitive and user-friendly interface for users to explore coding contests, hackathons, and hiring challenges. It is built using modern web technologies such as HTML, CSS, and

Keywords—Coding contests, Competitive programming, Contest Lister, System architecture, API integration, User interface.
JavaScript, ensuring compatibility across various devices and browsers. The frontend interface comprises multiple tabs, including contests, hackathons, and hiring, allowing users to navigate between different types of events effortlessly. Each tab is designed to display relevant contest information sourced from various platforms in a structured and visually appealing manner. Additionally, the frontend includes features such as customizable filters, search functionality, and interactive elements to enhance the user experience.

Backend Architecture:
The backend architecture of the Contest Lister project is responsible for data retrieval, processing, and storage. Node.js, a runtime environment for server-side JavaScript code execution, is used to implement it. The backend interacts with external APIs provided by coding platforms such as CodeChef, HackerRank, and LeetCode to fetch contest information in real-time. These APIs serve as the primary source of data for the system, providing details such as contest name, platform, start time, duration, and other relevant metadata. The backend processes the retrieved data, performs any necessary transformations or aggregations, and stores it in a relational database management system (RDBMS) for efficient retrieval and manipulation. The database schema is designed to accommodate various types of contest data and optimize query performance. Additionally, the backend includes modules for handling user authentication, session management, and other essential functionalities to ensure the security and integrity of the system.

![Fig. System Architecture](image1)

Integration and Communication:
The frontend and backend components of the Contest Lister project communicate with each other via HTTP requests and responses. The frontend sends requests to the backend to retrieve contest data based on user interactions such as tab selection, filter application, or search queries. After processing these requests and retrieving the necessary information from the database or external APIs, the backend replies to the frontend. This communication flow is orchestrated using RESTful APIs, with well-defined endpoints and data formats to facilitate interoperability and maintainability[12]. Additionally, the system may implement caching mechanisms at both the frontend and backend layers to improve performance and reduce latency, especially for frequently accessed data.

Scalability and Reliability:
The system architecture of the Contest Lister project is designed to be scalable and reliable, capable of handling a large volume of concurrent users and data requests. It
employs best practices such as load balancing, horizontal scaling, and fault tolerance to ensure high availability and performance under varying traffic conditions. The use of asynchronous programming techniques in the backend allows for non-blocking I/O operations, enabling efficient resource utilization and responsiveness. Furthermore, the system is deployed on cloud-based infrastructure providers such as AWS, Google Cloud, or Microsoft Azure, leveraging their robust and scalable services to achieve operational excellence and cost efficiency.

### III. THODOLOGIES

Our methodology for developing the Contest Lister involves a systematic approach aimed at achieving the project objectives efficiently. The methodology encompasses several key stages, each designed to address specific aspects of the project and ensure its successful implementation.

**Requirements Collection:**
The beginning involves administering a detailed reasoning of the necessities gathered from partners, including consumers and domain specialists. This analysis helps us identify the essential features and functionalities expected from the Contest Lister.

**Design Planning:**
We continue to construct the Contest Lister's architecture and user interface in accordance with the requirements analysis. At this stage, the application's structure and layout are visualised through the creation of wireframes, mockups, and prototypes.

**Development:**
After the design is finalised, the Contest Lister is actually coded during the development stage. We utilize appropriate programming languages, frameworks, and technologies, such as React, Node.js, and PostgreSQL, to implement the required features.

**Testing:**
Quality assurance is paramount in ensuring the reliability and functionality of the Contest Lister. We conduct exact experiment processes, holding whole experiment, unification experiment, and services agreement experiment, to label and advance some bugs or issues.

### IV. ALGORITHM

1. **Initialization:**
   Begin the process by setting up the necessary environment and dependencies.

2. **Data Schema Setup:**
   Utilize Sequelize ORM to define the schema for the tables in a PostgreSQL database, ensuring efficient data management and integrity.

3. **Automated Data Retrieval:**
   Implement a CRON job scheduler to execute a data retrieval task at regular intervals, such as every 6 hours.

Fetch data from the provided API endpoint (e.g., https://kontestlist.com/) to acquire updated information in JSON format.

4. **Data Processing and Storage:**
   Process the retrieved JSON response, formatting it according to the predefined schema. Store the formatted data securely in the PostgreSQL database, segregating it into relevant tables for efficient querying.[3].

5. **User Interface Interaction:**
   Develop a React-based frontend interface to interact with the stored data. Enable users to access contest, hackathon, and hiring information through the interface.

6. **Dynamic Data Presentation:**
   Display the fetched data dynamically on the user interface, providing real-time updates based on the stored information.

7. **Filtering Mechanism:**
   Implement filtering options based on user preferences, allowing users to customize the displayed data according to specific criteria.

8. **Completion:**
   Conclude the algorithm execution after fulfilling user requests and providing relevant data.

### V. FUTURE SCOPE

Looking ahead, there are several avenues for future enhancement and expansion of the Contest Lister project.

- **Enhanced Coverage:** Expand the coverage of coding platforms to include a wider range of contests and events, thereby providing users with access to a more comprehensive set of opportunities.

- **Advanced Filtering:** Introduce advanced filtering options to allow users to tailor their contest discovery experience based on criteria such as programming language, difficulty level, and contest type.

- **User Feedback Integration:** Incorporate user feedback mechanisms to gather insights and suggestions for improving the system's functionality, usability, and feature set.

- **Community Engagement:** Foster a vibrant community around the Contest Lister platform by enabling users to interact, collaborate, and share insights and experiences related to coding contests and hackathons.

- **Scalability Optimization:** Implement scalability optimizations to ensure that the system can accommodate a growing user base without compromising performance or reliability.

Mobile Application: Develop a dedicated mobile application for Contest Lister to provide users with seamless access to contest information on the go, enhancing accessibility and convenience.

### VI. ADVANTAGES

1. **Centralized platform for contest discovery:** The Contest Lister project provides users with a single, centralized platform where they can conveniently discover upcoming coding contests, hackathons, and hiring challenges from various coding platforms.[4]. This eliminates the need for users to visit multiple websites or platforms separately,
streamlining the contest discovery process.

2. Real-time updates from multiple platforms: By integrating APIs from popular coding platforms like CodeChef, HackerRank, and LeetCode, the Contest Lister system ensures that users receive real-time updates on contest listings[7]. This means that users have access to the latest and most accurate information about upcoming contests, enabling them to plan and participate effectively.

3. User-friendly interface with customizable filters: The user interface of the Contest Lister platform is designed to be intuitive and easy to navigate, ensuring a seamless user experience. Additionally, the system offers customizable filters that allow users to refine their contest search based on criteria such as platform, duration, difficulty level, and more. This empowers users to tailor their contest discovery experience according to their preferences and requirements[15].

In addition to assessing the technical performance of the system, user engagement metrics were also analyzed to gauge the platform's effectiveness in meeting user needs and preferences. Metrics such as user interactions, time spent on the platform, and feedback received from users were considered to evaluate the platform's usability and appeal.

To provide a comprehensive overview of the system's performance, the Result and Discussion section includes visual representations such as screenshots of result tables and graphs. These visual aids help illustrate key findings and trends observed during the evaluation process, making it easier for readers to interpret and understand the results[14].

Overall, the discussion focuses on interpreting the results obtained from the evaluation and highlighting the strengths and weaknesses of the Contest Lister platform. Any significant findings or insights gleaned from the evaluation process are discussed in detail, along with recommendations for future enhancements or refinements to the system.

In conclusion, the Result and Discussion section serves to present a thorough analysis of the Contest Lister system's performance and user engagement, providing valuable insights into its effectiveness and potential areas for improvement.
VIII. CONCLUSION

In conclusion, the Contest Lister project presents a comprehensive solution to the challenges encountered by coding enthusiasts in discovering and participating in coding contests, hackathons, and hiring challenges across various platforms. Through the development and implementation of this project, several key insights and accomplishments have been achieved.

Firstly, the project successfully addresses the need for a centralized platform for contest discovery, providing users with a convenient and efficient way to explore upcoming events. By aggregating contest information from multiple platforms and presenting it in a unified interface, the Contest Lister system simplifies the process of finding and participating in coding competitions.

Secondly, the user-friendly interface of the system, coupled with customizable filters and real-time updates, enhances the overall contest discovery experience[6]. Users can easily navigate through different tabs, apply filters based on their preferences, and stay informed about the latest events in the coding community.

Additionally, the integration of external APIs from platforms like CodeChef, HackerRank, and LeetCode ensures that users have access to up-to-date contest information[7]. This seamless integration enhances the reliability and accuracy of the data presented to users, contributing to a more engaging and informative user experience[5].

Looking ahead, there are several opportunities for further enhancement and refinement of the Contest Lister project. Future iterations of the system could focus on expanding coverage to include more coding platforms, improving scalability to accommodate a growing user base, and incorporating feedback from users to address any usability issues or feature requests[9].

IX. ACKNOWLEDGEMENT

We wish to thank our parents and associates for their valuable support and encouragement throughout the development of the project work and we would also like to thank our guide Prof. Sonu Khapekar for guiding us throughout the project work.

X. REFERENCES

[12] Study on REST API Test Model Supporting Web Service Integration Hu Wenhui Liu, Xu Chen 2017 ieee 3rd international conference on big data security on cloud (big data security), iee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids).
[13] Design Patterns and Extensibility of REST API for Networking Applications Li Li ; Wu Chou; Wei Zhou; Min Luo IEEE Transactions on Network and Service Management Year: 2016 | Volume: 13, Issue: 1 | Journal Article | Publisher: IEEE
Responsive web design commitment by the web developers in Palestine Fawwaz Yousef Alnawaj'ha; Mohammed Saeed Abutaha 2018 4th International Conference on Computer and Technology Applications (ICCTA) Year: 2018 | Conference Paper | Publisher: IEEE.