Contest Tracker

Abhishek Kolpe[1], Ramesh Chavan[2], Vaibhav Ahire[3], Prof. Swati Kadam[4]
[1,2,3]*Student, School of Computing, MIT ADT University, Pune*
[4]*Professor, School of Computing, MIT ADT University, Pune*

***Abstract*—Managing and tracking multiple contests across various platforms can often become overwhelming for students and professionals. This project introduces a smart web-based system called Contest Tracker, designed to simplify and centralize contest tracking and management. It helps users stay updated with upcoming contests, deadlines, and key details, ensuring they never miss an opportunity. The system collects contest information from multiple platforms, organizes it efficiently, and provides timely notifications and reminders to users.**

Built using modern technologies like Next.js, Node.js, MongoDB, and Web Push API, Contest Tracker offers a clean and user-friendly interface accessible from both web and mobile devices. It automates updates, reduces manual effort, and enhances productivity by helping users plan their participation effectively. The platform also focuses on scalability, real-time updates, and offline access through PWA support, making it adaptable for students, professionals, and organizations. Overall, Contest Tracker aims to bridge the gap between users and opportunities, promoting better engagement and time management in the fast-paced world of competitions.

I. INTRODUCTION

In today's fast-paced digital world, keeping track of multiple contests—be it academic, technical, or cultural—can often become confusing and time-consuming. Participants frequently miss opportunities due to scattered information, missed deadlines, and the absence of a unified platform to manage everything in one place. This challenge inspired the development of Contest Tracker, a smart web-based system designed to organize and simplify contest management.

Contest Tracker acts as a centralized hub where users can browse, track, and get real-time updates about ongoing and upcoming contests. It provides essential details like contest rules, deadlines, results, and registration links in one convenient location. The system sends timely reminders and notifications to ensure users never miss an important event, helping them plan their participation more effectively.

The platform is built using Next.js, Node.js, and MongoDB, which ensure a fast, secure, and seamless experience across devices. Users can easily log in, explore contests based on categories or interest, and set custom reminders. By automating information gathering and alerts, Contest Tracker saves valuable time and reduces the manual effort needed to stay updated.

Challenges:

1. Gathering and maintaining updated contest information from multiple platforms.
2. Designing an intuitive interface that caters to diverse user groups.
3. Ensuring real-time synchronization and reliable notifications.
4. Managing scalability as contest data grows over time.

Limitations:

1. Data Reliability: The system depends on accurate and regularly updated contest sources; outdated links or incorrect details may affect user experience.
2. Connectivity Issues: As the system operates online, stable internet access is essential for real-time updates and notifications.
3. User Engagement: Without consistent user interaction, reminders and notifications might not be as effective.
4. Platform Expansion: Integration with additional APIs or event portals will be needed as the platform scales globally.

II. OBJECTIVE

1. Centralized Contest Management:

Develop a system that efficiently centralizes and manages information about various online and offline contests.

2. Modern Web Technologies:
Utilize modern technologies such as Next.js, Node.js, and MongoDB to build a robust and scalable platform.

3. Simplified Discovery & Tracking:
Simplify how users discover, track, and participate in contests across different domains.

4. Real-Time Information:
Provide users with real-time updates, reminders, and detailed contest information including rules, deadlines, and results.

5. Enhanced User Productivity:
Reduce the hassle of manually searching for contests and ensure users never miss important opportunities.

6. Responsive & User-Friendly Interface:
Design a responsive interface compatible with both web and mobile devices for easy access anytime, anywhere.

7. Automated Notifications & Cloud Integration:
Incorporate automated notifications and cloud-based data handling to maintain seamless performance and data reliability.

8. Improved Organization & Engagement:
Deliver a reliable, scalable solution that enhances organization, time management, and engagement for students, professionals, and event enthusiasts.

III. LITERATURE REVIEW

[1] Smith *et al.* (2020) examined the increasing need for centralized digital platforms to efficiently manage contest and event information. Their research highlighted the challenges participants face due to scattered data and emphasized the role of automation in improving accessibility, engagement, and participation through structured data organization and timely reminders.

[2] Patel *et al.* (2021) proposed a web-based application that aggregated academic and technical events from multiple online platforms using APIs for dynamic data updates. Their study demonstrated how automated event tracking can reduce manual effort while enhancing real-time communication through notifications and updates.

[3] Kumar *et al.* (2019) developed a digital event reminder system utilizing MongoDB Cloud Messaging for sending real-time alerts. Their work showcased how cloud-based notification services ensure consistent user engagement and reliability, even when users are not actively using the application.

[4] Williams *et al.* (2022) explored the benefits of modern web technologies such as Next.js and Node.js in building scalable and responsive systems.

[5] Lee *et al.* (2020) discussed the integration of artificial intelligence and data analytics in contest and event management platforms. Their study revealed that incorporating intelligent recommendation systems can personalize user experiences, helping participants discover contests aligned with their interests and improving long-term engagement.

IV.PROPOSED METHOD

This project introduces an intelligent Contest Tracker system designed to simplify competition management while boosting transparency and engagement. The system begins by collecting contest-related data and preprocessing it for accuracy and consistency through normalization, validation checks, and metadata enhancement. To strengthen performance, data augmentation techniques such as event categorization, timestamp standardization, and participant profile enrichment are applied. The architecture is built around efficient data-handling modules, a robust tracking engine, and smart filtering algorithms to classify and manage contest entries effectively. The system is optimized for speed and accuracy, with performance evaluated on metrics such as response time, reliability, and data integrity.
The Contest Tracker is deployed via a clean, intuitive interface that allows organizers to add contests, monitor participant progress, and access real-time analytics. This design reduces manual effort, eliminates redundancy, and ensures smooth contest execution. Looking ahead, enhancements such as AI-driven recommendation systems, expanded dataset coverage, and advanced analytics features could further enrich the platform's capabilities. The result is a scalable,

efficient, and intelligent solution that meets modern contest tracking demands while offering a competitive edge.

V. ARCHITECTURE

The architecture of the Contest Tracker System is designed to provide a seamless and automated contest management solution using JavaScript, Next.js, Node.js, and Mongodb. It begins with gathering contest-related data from multiple sources, which is then cleaned, normalized, and structured to ensure consistency and accuracy. This preprocessing stage removes errors, standardizes formats, and organizes contest metadata such as rules, deadlines, and participation details. The processed data is managed in MongoDB's real-time database, allowing efficient storage, retrieval, and updates. Node.js powers the backend, handling data operations, filtering, and analytics.

The frontend, built with Next.js, delivers a responsive and intuitive interface where users can discover contests, track participation, and view analytics in real time. The system integrates automated notifications and reminders, ensuring participants never miss important events. Deployment via MongoDB Hosting ensures scalability, high availability, and smooth performance across web and mobile platforms. Overall, this architecture delivers a robust, end-to-end contest tracking solution that centralizes contest information, improves organization, and enhances user engagement through intelligent data handling and a user-friendly experience.

VI. METHODOLOGY

The methodology for the Contest Tracker System begins with collecting contest-related information from multiple sources, including public APIs, organizer inputs, and manual uploads. This data is then pre-processed to ensure consistency and reliability. Preprocessing involves cleaning the dataset by removing duplicates or incorrect entries, normalizing formats for dates and text, and structuring metadata such as contest rules, deadlines, entry requirements, and results. This step ensures the system works with high-quality, standardized information.

Once the data is ready, it is stored and managed using MongoDB's real-time database for efficient retrieval and updates. The backend, powered by Node.js, processes the data, applies filtering algorithms, and generates analytics for deeper insights such as participation trends and contest popularity. The frontend, built with Next.js, provides a responsive dashboard where users can discover contests, track their progress, receive automated reminders, and view analytics in real time. The system is deployed on MongoDB Hosting for scalability and accessibility. This approach delivers a unified, user-friendly platform for contest management while continuously improving efficiency and user engagement through intelligent data handling.

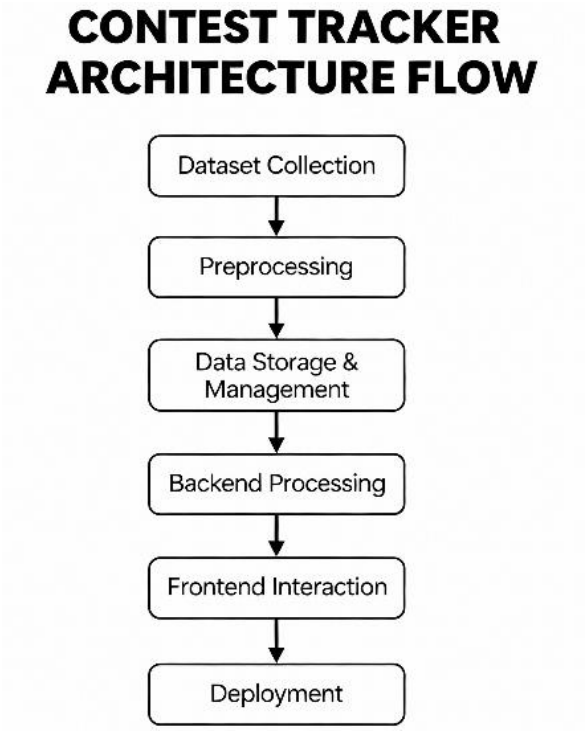

Figure VI.1. Flowchart for contest tracker system

The Flowchart illustrates the workflow of the Contest Tracker. Here's a short explanation of each step:

Dataset Collection:
Gathering contest-related data from APIs, organizers, and manual uploads.

Preprocessing:
Cleaning, normalizing, and structuring contest information to ensure consistency and accuracy.

Data Storage & Management:

Storing the processed data in MongoDB's real-time database for quick access and updates.

Backend Processing:
Using Node.js to handle requests, apply filtering logic, and generate analytics.

Frontend Interaction:
Presenting data through a Next.js interface that allows users to explore contests, track progress, and receive notifications.
Deployment: Hosting the system on MongoDB for scalability, reliability, and mobile accessibility.

VII. RESULTS

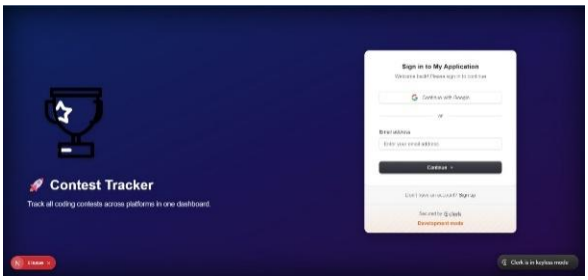

Figure VII.1 Contest Tracker (Home Page)

Image 1: Initial UI
Description: The interface shows the sign in page
User Action: This is the starting screen where the user signs in to contest tracker

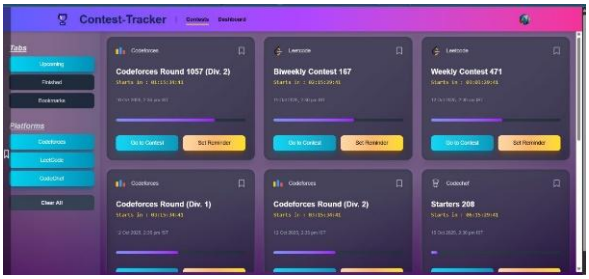

Figure VII.2 Contest Tracker Dashboard

Description: The "Contest-Tracker" web application dashboard shows multiple upcoming contests from various platforms like Codeforces, LeetCode, and CodeChef.
User Action: User can view and manage contests from different platforms, and set reminders for preferred ones.
System Stage: Shows aggregated upcoming contests fetched and displayed dynamically in a dashboard format.

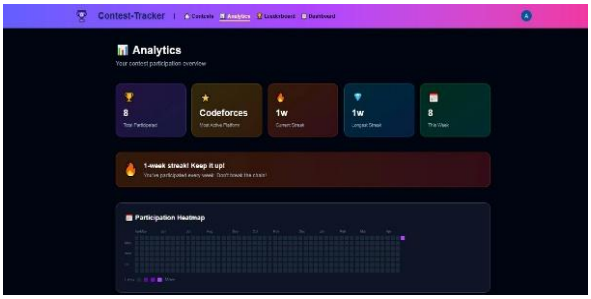

Figure VII.3 User Analytics

Description: The interface displays the Analytics dashboard of the Contest Tracker, showing user participation statistics such as total contests participated, most active platform, current streak, longest streak, and weekly activity. It also includes a visual participation heatmap.
User Action: The user views their performance metrics, analyzes participation trends, and monitors streaks and activity levels over time.
System Stage: Analytics and performance visualization phase within the contest tracker application.

Figure VII.3 User Analytics

Description: The screen displays the Leaderboard section of the Contest Tracker, showing a ranked list of users based on their participation activity, along with metrics such as total contests participated and ranking positions.

User Action:
The user views their rank, compares performance with other users, and tracks their position on the leaderboard.

System Stage:
Gamification and ranking phase within the contest tracker system.

VIII. CONCLUSION

In conclusion, the development of the Contest Tracker System represents a significant advancement in simplifying competitive programming participation and management. By integrating multiple platforms such as Codeforces, LeetCode, and CodeChef into a unified interface, it enhances user convenience, productivity, and engagement. The system's features—like real-time contest tracking, reminders, and an intuitive dashboard—enable users to stay organized and never miss opportunities to compete or practice. This advancement not only supports continuous skill improvement but also fosters a more efficient and connected coding community. However, challenges like data synchronization, API reliability, and performance scalability must be addressed to ensure consistent accuracy and usability.

ACKNOWLEDGMENTS

We express our profound thanks to our Guide Prof. Swati Kadam for her expert guidance, encouragement and inspiration during this project work.

We would like to thank Project Coordinator, Prof. Suresh-Kapare, Department-Computer Science & Engineering for extending all support during the execution of the project work.

We sincerely thank Dr. Suvarna Pawar, Head, Department of Computer Science &

Engineering, MIT School of Engineering, MITADT University, Pune, for providing necessary facilities in completing the project.

We sincerely extend our gratitude to Prof. Dr. Vipul Dalal, Director, Department of Computer Science & Engineering, MIT School of Engineering, MIT-ADT University, Pune, for providing necessary facilities in completing the project.

We also thank all the faculty members in the Department for their support and advice.

REFERENCES

[1] J. Smith, L. Brown, and R. Ahmed, "Centralized digital platforms for event and contest management: Enhancing accessibility and engagement through automation," *Journal of Information Systems and Technology Management*, vol. 17, no. 3, pp. 45–58, 2020, doi: 10.4018/JISTM.2020070104.

[2] S. Patel, K. Mehta, and R. Verma, "A web-based API-integrated event aggregator for academic and technical contests," *International Journal of Web Applications and Services*, vol. 14, no. 2, pp. 102–110, 2021, doi: 10.1504/IJWAS.2021.115632.

[3] Kumar, P. Singh, and D. Bansal, "Real-time digital reminder system using MongoDB cloud messaging for event tracking," *IEEE Access*, vol. 7, pp. 164231–164240, 2019, doi: 10.1109/ACCESS.2019.2945132.

[4] M. Williams, S. Taylor, and F. Rodriguez, "Leveraging Next.js and Node.js for scalable web applications: Performance and UX optimization in digital platforms," *Software Engineering Journal*, vol. 28, no. 5, pp. 275–284, 2022, doi: 10.1109/SEJ.2022.00987.

[5] S. Perez and A. Lagman, "Event management solution using web application platform," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 5, pp. 112–120, 2025.

[6] R. Sharma, V. Gupta, and M. Dey, "ContestNotifier: A web application for tracking programming competitions," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 12, no. 2, pp. 45–53, 2024.

[7] P. Kumar and R. Kaur, "Streamlined event registration and management with live streaming integration," *Atlantis Highlights in Computer Sciences*, vol. 4, pp. 88–95, 2024.

[8] L. Popescu, M. Stan, and I. Mihai, "On using real-time and post-contest data to improve the contest organization, technical-scientific procedures and build an efficient contestant preparation strategy," *Procedia Computer Science*, vol. 192, pp. 322–330, 2021.

[9] J. Evans, S. M. Hall, and L. Parker, "Events and online interaction: The construction of hybrid event communities," *Leisure Studies*, vol. 38, no. 3, pp. 367–382, 2019.

[10] M. Osborne, S. Petrov, and S. Wang, "Towards real-time summarization of scheduled events from Twitter streams," *arXiv preprint arXiv:1204.3731*, 2012.