ABSTRACT

CodePilot is a cloud-powered, API-driven learning platform designed to enhance how students understand and practice Data Structures and Algorithms (DSA). Traditional coding platforms often lack personalization, adaptability, and motivation, leading to disengagement and slow progress among learners. To address these issues, CodePilot integrates gamification, adaptive learning, and real-time cloud execution to create an engaging and scalable environment for mastering DSA concepts.

The platform personalizes the learning journey by dynamically adjusting problem difficulty based on user performance, delivering challenges through a structured RESTful API. It incorporates game-based elements such as experience points (XP), levels, streaks, and progress badges to keep learners motivated and consistent.

A key technical highlight is the use of Docker containers for secure, language-agnostic code execution in the cloud. This allows users to submit solutions in various programming languages (Python, C++, Java), which are executed safely in isolated environments. The backend supports high scalability and fault tolerance by leveraging cloud services like AWS Lambda or Render.

Authentication is handled via Firebase Authentication, enabling secure Google sign-in. User profiles, problem history, and XP logs are stored in a cloud-based NoSQL database like MongoDB or Firestore, supporting real-time data updates and scalability.

In addition, CodePilot features an optional interactive dashboard, displaying performance analytics including accuracy, time spent, and topic-wise proficiency. This empowers students with actionable insights for continuous improvement.

By combining modern full-stack development, cloud computing, and educational gamification, CodePilot offers a smart, scalable, and fun way to master DSA. It stands as a practical solution for self-paced learning and academic enhancement, while also demonstrating industry-grade backend architecture for real-world deployment.

PROBLEM STATEMENT

Although numerous platforms exist to help students practice Data Structures and Algorithms (DSA), many fall short in addressing core learning challenges such as lack of motivation, inconsistent practice, and insufficient personalization. Most traditional platforms follow a static, one-size-fits-all approach that fails to adapt to a learner’s pace, strengths, or weaknesses, resulting in disengagement and slow progress.

Moreover, code execution is often limited to specific programming environments or requires local setups, making access difficult across devices and limiting scalability. This can be especially problematic for learners in resource-constrained settings.

To overcome these limitations, there is a need for a dynamic, learner-focused platform that offers personalized difficulty progression, real-time feedback, and a seamless, secure cloud-based coding environment supporting multiple programming languages. Additionally, integrating gamification elements such as streaks, levels, XP, and rewards can boost engagement and consistency.

OBJECTIVES OF THE PROJECT

1.To build a gamified learning platform that adapts to each student’s skill level in Data Structures and Algorithms.

2.To implement secure cloud-based code execution using Docker containers, allowing real-time code evaluation across languages.

3.To design RESTful APIs for managing problem delivery, solution validation, and user progress tracking.
4.To integrate Firebase Authentication for secure login via Google and manage personalized user sessions.
5.To store and retrieve user history and XP using a scalable NoSQL database like MongoDB or Firestore.
6.To provide visual analytics on a user dashboard, tracking accuracy, consistency, and topic-wise progress.

LITERATURE SURVEY:

Title	Authors	Year	Methodology	Key Findings	Relevance	Limitations
OneUp: Engaging Students in a Gamified Data Structures Course	Cheng-Hsiang Liu, Yu-Ting Chung, Richard C. Holt	2019	Implemented a web-based DSA course with XP, badges, leaderboards; evaluated student engagement and retention metrics.	Showed significant increases in motivation and problem completion in gamified environment.	Supports the inclusion of XP, levels, badges, and streaks in CodePilot.	Evaluated on a single course with limited sample size.
Deep Reinforcement Learning for Adaptive Learning Systems	Xiao Li, Hanchen Xu, Jinming Zhang, Hua-hua Chang	2020	Used deep Q-learning to dynamically select learning materials based on students' latent knowledge states.	Demonstrated efficient learning policies that tailor content to individual performance.	Aligns with CodePilot's adaptive delivery engine for personalized DSA problem sequencing.	Primarily simulations; lacks real-world deployment validation.
Design and Analysis of Cloud–Based Docker Application Platforms	Byoung Soo Kim, Sang Hyeop Lee, Ye Rim Lee, Yong Hyun Park, Jongpil Jeong	2022	Analyzed Docker container orchestration on cloud platforms, focusing on deployment, scalability, and isolation.	Validated container usage for scalable, secure application hosting in cloud environments.	Informs CodePilot's secure, Docker-based multi-language code execution infrastructure.	Focused on enterprise manufacturing workloads rather than educational use.
Real-Time Dashboards for Programming Progress Visualization	Alhassan, A.; Ali, M.; Rauf, A.	2022	Built real-time dashboards using Firebase and NoSQL DB to visualize student practice time, topics completed, and	Students using dashboards self-corrected their pace and	Strongly aligns with CodePilot's interactive dashboard, showing practice	Limited customizability for visual preferences and no deep

			accuracy. User testing was done on 120+ students.	reported improved motivation and awareness of weak areas.	history, accuracy trends, language-wise stats, and adaptive feedback.	analytics beyond raw scores.
The Effectiveness of Gamification in Programming Education	David W. Anderson, Daniel A. Lane	2022	Systematic review of gamification in programming courses, analyzing student performance and engagement increases.	Found positive effects on motivation and learning but stressed that implementation nuance matters.	Offers evidence for integrating gamified elements like points, challenges, and feedback in CodePilot.	Highlights variability in effectiveness; not specific to DSA workflows.

INTRODUCTION

Data Structures and Algorithms (DSA) form the backbone of computer science and are essential for excelling in technical interviews and real-world problem-solving. Despite their importance, many students struggle to grasp DSA concepts due to the absence of personalized guidance, interactive feedback, and engaging learning methods. Traditional platforms often lack adaptability and fail to maintain learner motivation over time.

CodePilot aims to transform this experience by offering a gamified, adaptive, and cloud-based DSA learning environment. Inspired by video games, the platform rewards users with experience points (XP), unlockable levels, and dynamically tailored problem sets that evolve with the learner's progress.

To ensure a secure and flexible coding experience, all code submissions are executed within isolated Docker containers, supporting multiple programming languages. The system is integrated with Firebase Authentication for secure user management and leverages scalable cloud databases like MongoDB or Firestore to track progress in real-time. CodePilot makes DSA learning efficient, personalized, and fun.

METHODOLOGY

1. Problem Delivery via API

- RESTful API built using Node.js or Flask to serve DSA problems categorized by topic and level.
- API returns test cases, constraints, XP value, and hints.

2. Cloud Code Execution (Docker)

- Backend uses Docker containers spun up on the cloud(via AWS Lambda or Render).

- Isolated environments ensure safe and scalable code execution.
- Supports Python, C++, and Java.

3. Authentication (Firebase)

- Firebase Authentication handles Google Sign-In.
- Secure session tokens are maintained for each user.

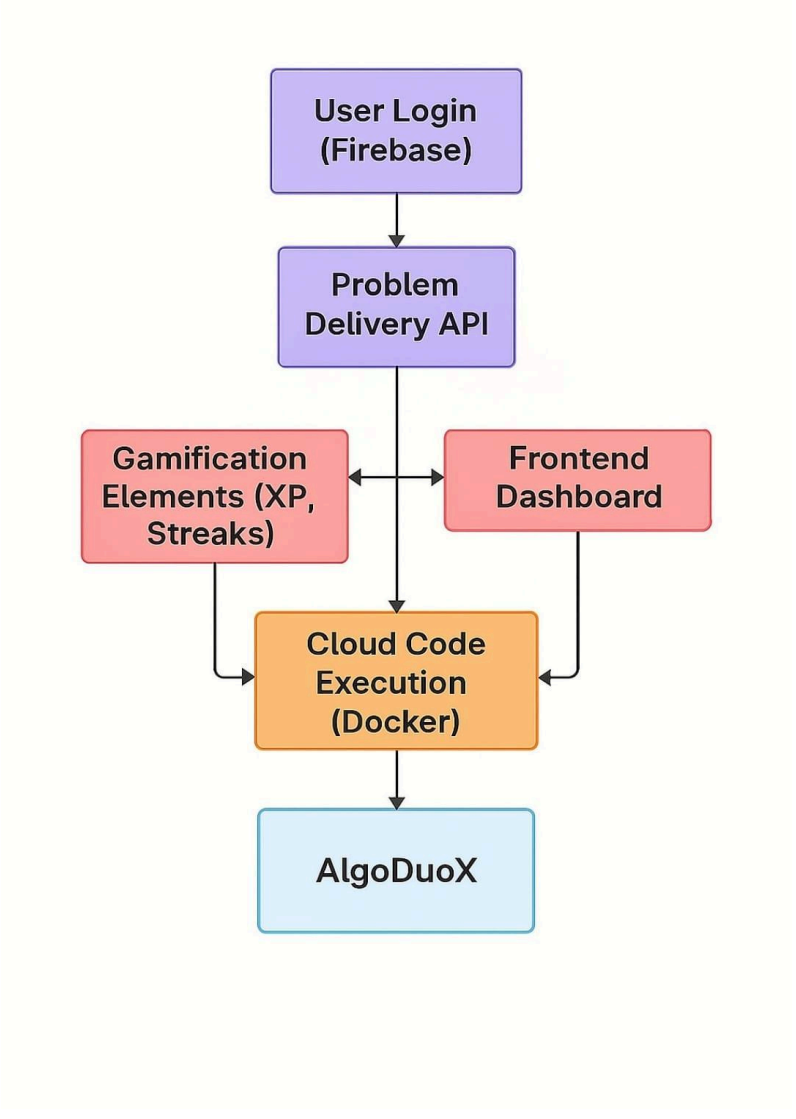
4. Database Integration

- MongoDB/Firestore used to store user profiles, past submissions, XP history, and progress logs.
- Scalable schema supports thousands of users concurrently.

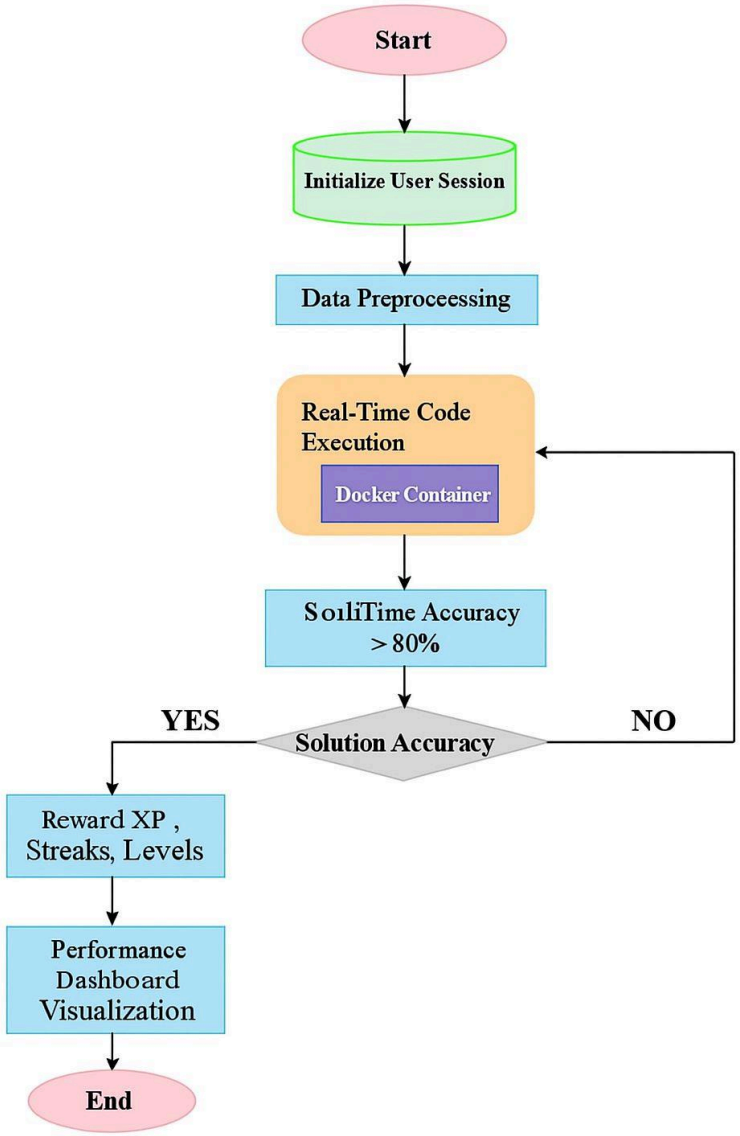
5. Frontend

- A dashboard visualizes problem stats, XP bar, graphs (via Chart.js or Power BI).
- Additional modules like streak tracking, rank boards may be added.

WORKFLOW:



ARCHITECTURE



EXPECTED RESULTS

1.Adaptive DSA Problem Delivery System
The platform adjusts problem difficulty based on the user's skill level and progress, ensuring personalized learning that matches their pace and understanding.

2.Secure and Scalable Code Execution
User code runs in cloud-hosted Docker containers, allowing safe, isolated execution of multiple programming languages without needing any local setup. The system scales easily to support many users at once.

3.XP and Level System for Motivation
Learners earn XP and unlock levels as they solve problems. Streaks and rewards keep them motivated and consistent, turning learning into a fun, game-like experience.

4.User Profile and History Tracking
Each user has a profile where their progress, solved problems, XP, and language preferences are stored—helping them track improvement over time.

5.Real-Time Progress Insights via Dashboard
A visual dashboard displays key stats like accuracy, speed, topic strengths, and daily streaks—giving learners actionable feedback and encouraging improvement.

6.Google Authentication for Login/Logout
Firebase Authentication enables secure, hassle-free login and logout using Google accounts, improving user experience and platform security.

SOFTWARE & HARDWARE REQUIREMENTS

Software Requirements:

1. **Operating System:**
    - Compatible with Windows 10/11 or Ubuntu-based Linux distributions to support cross-platform development and testing.
2. **Programming Languages:**
    - Node.js for backend REST API development.
    - Python for algorithm processing, analytics, and optional machine learning modules.
3. **Frameworks & Tools:**
    - Express.js (Node.js) or Flask (Python) for server-side application logic and API endpoints.
    - Docker for containerizing the code execution environments, ensuring secure and isolated runs for multiple languages.
    - MongoDB or Firestore for storing user profiles, XP logs, and progress history in a NoSQL, scalable format.

Libraries:

•**JWT (JSON Web Tokens)**: For secure user session management and token-based authentication.
•**Axios**: For making asynchronous API calls between frontend and backend.
•**Chart.js**: To create real-time visual dashboards of user performance and analytics.
•**TensorFlow** : For implementing future analytics or recommendation systems using machine learning.
•**IDE/Editor:** VS Code: Primary local development environment.

Hardware Requirements :
• Processor: Intel i5 or equivalent
• RAM: 8 GB
• Internet: Stable broadband connection for accessing cloud APIs and executing containerized code.
• Recommended Configuration:
• RAM: 16 GB or higher for smooth multitasking during development and testing.
• Cloud GPU Access: For performance testing or running intensive code execution and analytics workloads, especially in future machine learning modules.

CONCLUSION

CodePilot successfully showcases the potential of combining full-stack development, cloud computing, and gamification to create a modern, effective, and scalable platform for mastering Data Structures and Algorithms (DSA). Unlike traditional coding platforms, it emphasizes personalization, adaptability, and engagement—three core factors essential for sustained learning.

The platform's architecture uses secure cloud-based Docker containers for language-agnostic code execution, ensuring safety and flexibility. Firebase Authentication and NoSQL databases such as Firestore or MongoDB enable real-time data handling and scalable user management. Features like experience points (XP), levels, streaks, and visual dashboards motivate learners to stay consistent while also giving them insights into their strengths and areas of improvement.

CodePilot not only benefits students by offering a self-paced and interactive learning journey but also serves as a valuable tool for educators and institutions to monitor and guide learners. The project stands as a practical demonstration of building robust, scalable backend systems using real-world technologies—making it an ideal blend of academic innovation and industrial relevance.

REFERENCES

[1] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From Game Design Elements to Gamefulness: Defining 'Gamification'," in Proc. 15th Int. Academic MindTrek Conf., 2011, pp. 9–15.

[2] H. Xie, H.-C. Chu, G.-J. Hwang, and C.-C. Wang, "Trends and Development in Technology-Enhanced Adaptive/Personalized Learning: A Systematic Review of Journal Publications from 2007 to 2017," Computers & Education, vol. 140, pp. 103599, 2019.

[3] M. Sarikaya and C. Staley, "Gamification in Computer Science Education: A Systematic Literature Review," Comput. Appl. Eng. Educ., vol. 29, no. 2, pp. 388–408, 2021.

[4] J. R. Anderson, A. T. Corbett, K. R. Koedinger, and R. Pelletier, "Cognitive Tutors: Lessons Learned," J. Learn. Sci., vol. 4, no. 2, pp. 167–207, 1995.

[5] F. Wang and M. J. Hannafin, "Design-Based Research and Technology-Enhanced Learning Environments," Educ. Technol. Res. Dev., vol. 53, no. 4, pp. 5–23, 2005.

[6] M. González, J. C. Burguillo, and M. Llamas, "A Qualitative Evaluation of Using Gamification in a Learning Management System," Comput. Human Behav., vol. 27, no. 6, pp. 2173–2181, 2011..