A

Project Synopsis

on

CodeZoo : A Gamified Platform for Computer Science Education

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2025-26

in

Computer Science & Engineering

By

Shiv Pratap Singh (2300290100231) (Team Leader)

Prasiddha Pal (2400290109014)

Shubham Gupta (2300290100247)

Vikash Kushwaha(2400290109026)

Under the supervision of

Mr. Vaibhav Kori

KIET Group of Institutions, Ghaziabad

Affiliated to

Dr. A.P.J. Abdul Kalam Technical University, Lucknow

(Formerly UPTU)

Academic Year 2023-27

Abstract

Computer Science education is often perceived as difficult, abstract, and intimidating for beginners, leading to high dropout rates in early programming courses. Traditional learning methods—such as textbooks or static video lectures—often lack the engagement and immediate feedback required to maintain student motivation. To address this, this project proposes **CodeZoo**, a gamified learning platform inspired by the success of language-learning apps like Duolingo, but tailored specifically for programming and computer science concepts.

The proposed solution utilizes modern web technologies, specifically **React.js** for a dynamic frontend and **Node.js with MySQL** for a robust backend. The system transforms learning into a game where users earn Experience Points (XP), maintain daily streaks, unlock badges, and compete on leaderboards. A key innovation of CodeZoo is the integration of **Artificial Intelligence (Google Gemini API)**, which provides real-time, personalized explanations when a user answers a question incorrectly, acting as a virtual tutor.

Additionally, the platform includes a **Visual Coding Lab** to introduce logic without syntax errors and a **Practice Lab** powered by the **Skulpt** browser-based compiler, allowing users to write and execute Python code instantly. By combining gamification, AI-driven feedback, and practical coding environments, CodeZoo aims to make computer science education accessible, engaging, and effective for learners of all ages.

Introduction

In the digital age, programming literacy is becoming as fundamental as reading and writing. However, the learning curve for Computer Science (CS) is steep. Students often struggle with syntax errors, abstract logic, and a lack of motivation. Traditional educational platforms often fail to retain user attention due to passive learning models. This creates a significant need for an **Intelligent, Gamified Learning System** that adapts to the user and makes the learning process addictive and rewarding.

The purpose of this project is to design and implement **CodeZoo**, a web application that revolutionizes how CS concepts are taught. Unlike standard IDEs or video courses, CodeZoo breaks down complex topics (like loops, variables, and data structures) into bite-sized, interactive lessons. The system tracks user progress using a sophisticated backend database, rewarding consistency through "Streaks" and "Levels."

A major gap in existing free solutions is the lack of personalized feedback. When a student makes a mistake, they are often just told "Incorrect." CodeZoo bridges this gap by leveraging **Generative AI (Google Gemini Model)**. When a user selects a wrong answer, the system analyzes the specific mistake and generates a beginner-friendly explanation in real-time, helping the user learn from their errors immediately.

The key contribution of this project is the unification of three distinct learning modalities into one platform:

1. **Conceptual Learning:** Quiz-based lessons with gamified rewards.
2. **Visual Learning:** A drag-and-drop coding interface for understanding algorithmic logic.
3. **Practical Application:** An in-browser code editor/compiler for writing actual Python code.

This holistic approach ensures that users not only memorize syntax but also understand the underlying logic of computer science, preparing them for more advanced software development roles.

The system is built using the **MERN Stack principles** (using MySQL instead of Mongo), ensuring scalability and performance. The frontend is developed in **React 19** with **Tailwind CSS** for a responsive, modern, and accessible user interface that supports Dark Mode. The backend utilizes **Express.js** and **MySQL** to handle user authentication, progress tracking, and admin management.

Several technical terms are essential to understanding this project:

- **Gamification:** The application of game-design elements (XP, Leaderboards, Badges) to non-game contexts to improve engagement.
- **LLM (Large Language Model):** Refers to the Google Gemini AI used to generate human-like explanations for coding questions.
- **Client-Side Compilation:** Using libraries like **Skulpt** to run Python code directly in the browser without sending it to a server, ensuring speed and security.
- **REST API:** The architectural style used for communication between the React frontend and the Node.js backend.

In conclusion, the Introduction highlights the urgency of modernizing CS education. By integrating AI, gamification, and practical coding tools, CodeZoo provides a comprehensive solution that addresses the engagement crisis in e-learning.

Literature Survey

The field of e-learning and gamification has evolved significantly over the last decade. Research indicates that gamified elements significantly increase student retention and motivation.

Early approaches to online coding education, such as **Codecademy** or **W3Schools**, focused on "read and repeat" methodologies. While effective for syntax, these platforms often lack the "hook" to keep users returning daily. Research by *Deterding et al. (2011)* defined gamification as using game design elements in non-game contexts, proving its efficacy in education.

Duolingo revolutionized language learning by introducing streaks, leagues, and bite-sized lessons. However, its application has been primarily linguistic. While Duolingo has recently attempted math and music, there is a lack of a dedicated, robust platform that applies this exact model to the depth of Computer Science engineering concepts (Data Structures, Algorithms, OOPs).

Scratch (MIT) introduced visual block-based coding, which is excellent for children but lacks a bridge to professional text-based programming. **CodeZoo** addresses this by offering both a Visual Coding environment (similar to Scratch) and a Practice Lab for real text-based Python programming, creating a bridge for learners to advance.

Furthermore, the integration of **AI in Education (AIEd)** is a nascent field. Most platforms offer static feedback. Recent studies on Large Language Models (LLMs) show their potential as personal tutors. By integrating **Google Gemini**, CodeZoo implements what is known as "Intelligent Tutoring Systems" (ITS), providing dynamic feedback that was previously only possible with a human instructor.

The reviewed literature establishes that while gamification and coding platforms exist separately, there is a significant opportunity to combine **Gamification, Client-side Compilers, and Generative AI** into a single, cohesive web application. CodeZoo builds upon these technologies to create a next-generation learning platform.

Methodology/Planning of work

The development of CodeZoo follows the **Agile Development Lifecycle**, allowing for iterative improvements and testing.

1. Requirement Analysis
Identify the core curriculum (Python Basics, Logic, Data Structures). Define the gamification mechanics (Points, Streaks, Ranks). Select the technology stack: React (Vite) for frontend, Node.js for backend, and MySQL for the database.

2. System Architecture Design
Design the database schema in MySQL to store Users, Courses, Skills, Lessons, and User_Progress. Design the API endpoints for user authentication (Login/Register) and data synchronization.

3. Frontend Development (React & Tailwind)
Develop the User Interface components:

- **Dashboard:** Visualizing the "Skill Tree" nodes and progress.
- **Lesson View:** Interactive component for Multiple Choice, Fill-in-the-blank, and True/False questions.
- **Visual Coding Lab:** A drag-and-drop interface using HTML5 Drag and Drop API.
- **Practice Lab:** Integration of the Skulpt library to compile Python code in the browser.

4. Backend Development (Node.js & MySQL)
Implement the REST API server. Create secure endpoints for saving user progress (XP, Levels). Implement a hashing algorithm for password security. Create a seeding script to populate the database with initial courses and an Admin user.

5. AI Integration
Integrate the **Google GenAI SDK**. Create a service that constructs a prompt based on the user's incorrect answer and the question context, sending it to the Gemini 2.5 Flash model to retrieve a concise explanation.

6. Gamification Logic
Implement logic for calculating streaks (checking last login dates), assigning badges based on milestones, and generating dynamic leaderboards based on total XP.

7. Testing & Deployment

Perform unit testing on API endpoints. Test the "Offline" capabilities and responsiveness on different screen sizes. Deploy the database locally and verify data persistence.

Facilities Required for Proposed Work

The successful development of CodeZoo requires specific software and hardware facilities to support full-stack web development, database management, and AI integration.

Software Facilities

- **Operating System:** Windows 10/11 or Linux.
- **Programming Languages:** TypeScript, JavaScript (ES6+), SQL.
- **Frontend Framework:** React.js (v18/19), Vite.
- **Styling:** Tailwind CSS.
- **Backend Runtime:** Node.js.
- **Database:** MySQL (Community Server).
- **AI Models:** Google Gemini API (gemini-2.5-flash).
- **Compiler Library:** Skulpt (for client-side Python execution).
- **IDE / Tools:** VS Code, MySQL Workbench / phpMyAdmin, Postman (for API testing).

Hardware Facilities

- **Computer System:** Minimum 8GB RAM, Intel i5 processor or equivalent (to run the local server and React build process simultaneously).
- **Internet Connection:** Required for fetching data from the Google Gemini API and downloading npm packages.
- **Storage:** Sufficient storage for the codebase and local database.

Refrences

[1] Google AI for Developers, "Gemini API Documentation," *ai.google.dev*, 2024. [Online]. Available: https://ai.google.dev/docs.

[2] React Documentation, "React: The Library for Web and Native User Interfaces," *react.dev*, 2024.

[3] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining gamification," *Proceedings of the 15th International Academic MindTrek Conference*, 2011.

[4] Skulpt, "Skulpt: Python in the Browser," *skulpt.org*. [Online]. Used for client-side Python compilation.

[5] MySQL Documentation, "MySQL 8.0 Reference Manual," *dev.mysql.com*.

[6] Duolingo Research, "How Duolingo teaches: The science behind the app," *blog.duolingo.com*.

[7] MDN Web Docs, "Web Audio API," *developer.mozilla.org*. (Used for sound effects implementation).