International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

AI Powered Code Reviewer

Mr. Shaik Nagur Vali [1], M. Vikas[2], V. Smaran[3], E. Augusteen[4], N. Varun[5]

[1]Assistant Professor, Department of CSE (Data Science), ACE Engineering College, Hyderabad, India
[2,3,4,5] Student, Department of CSE (Data Science), ACE Engineering College, Hyderabad, India

ABSTRACT:

An AI-powered code reviewer project is developed using the MERN stack. The application allows users to submit code for review, identifying errors and suggesting improvements. The project includes backend and frontend integration, utilizing AI for code analysis and providing feedback. Utilizing AI in code reviews enhances efficiency, improves code quality, and ultimately leads to better software development practices. Viewers are encouraged to comment for deployment demonstrations. This project overall deals and tackle the challenges like Error Handling, Code Quality and Formatting, CORS Issues, Deployment Challenges, User Interface (UI) Presentation. In modern software development, maintaining high code quality and efficiency is essential. Traditional code review processes can be time-consuming and prone to human error. To address these challenges, this project introduces an AI-powered code reviewer built using the MERN (MongoDB, Express.js, React, Node.js) stack. The system enables users to submit their code for automated analysis, error detection, and improvement suggestions using AI-driven techniques. This project seamlessly integrates backend and frontend technologies, leveraging AI for intelligent code reviews. By automating the review process, the application enhances efficiency, accuracy, and overall software development practices. Furthermore, it tackles key development challenges such as error handling, code quality and formatting, CORS issues, deployment complexities, and UI presentation. By streamlining the code review workflow, this AI-powered system serves as a valuable tool for developers, helping them write better, more maintainable code in real-world applications.

1. INTRODUCTION

In the fast-paced world of software development, ensuring high code quality and security is crucial, yet traditional manual code reviews are often time-consuming, error-prone, and inconsistent. This project, titled **"AI-Powered Code Reviewer"**, addresses these challenges by integrating advanced AI techniques—specifically Gemini AI—with the MERN (MongoDB, Express.js, React.js, Node.js) stack to create an intelligent, automated code review system. The platform enables users to submit code for analysis, automatically detects bugs and vulnerabilities, and suggests improvements, enhancing development workflows across industries like finance, healthcare, education, and enterprise software. With applications ranging from educational feedback to CI/CD integration, this system exemplifies how AI can revolutionize the way developers ensure code quality and maintainability.

2. EXISTING SYSTEM

The current landscape of code review tools consists mainly of:

1. **Traditional Code Review Methods:**
   - Peer reviews using pull/merge requests on platforms like GitHub or GitLab.
   - These methods are **time-consuming**, **subjective**, and **depend heavily on the reviewer's expertise**.
   - Challenges include human bias, inconsistent feedback, and longer review cycles.
2. **Static Code Analysis Tools:**
   - Tools such as **SonarQube, ESLint, and Checkstyle**.
   - Focus primarily on syntactic issues and basic coding standards.
   - **Limitations**: Limited contextual understanding, high false positives, and inability to grasp code logic or intent.
3. **AI-Based Tools (like GitHub Copilot, DeepCode by Snyk):**
   - Provide suggestions using machine learning.
   - Offer better support but are not always context-aware and may produce **insecure or deprecated code**.

3. PROPOSED SYSTEM

The proposed system introduces a **Gemini AI-powered Code Reviewer** built on the **MERN stack** (MongoDB, Express.js, React.js, Node.js) with the following features:

- **Frontend (React.js)**: Intuitive interface for users to submit code.
- **Backend (Node.js + Express.js)**: Handles requests and routes code to the AI engine.
- **AI Microservice (Python + Gemini AI)**: Analyzes code for bugs, formatting, security vulnerabilities, and optimization.
- **Database (MongoDB)**: Stores user submissions, review results, and history.

ADVANTAGES OF THE PROPOSED SYSTEM

4. Real-time intelligent code review with actionable feedback.
5. Context-aware analysis using Gemini AI.
6. Improved UI and user experience.
7. Added support for GitHub integration, security-focused reviews, and review history tracking.

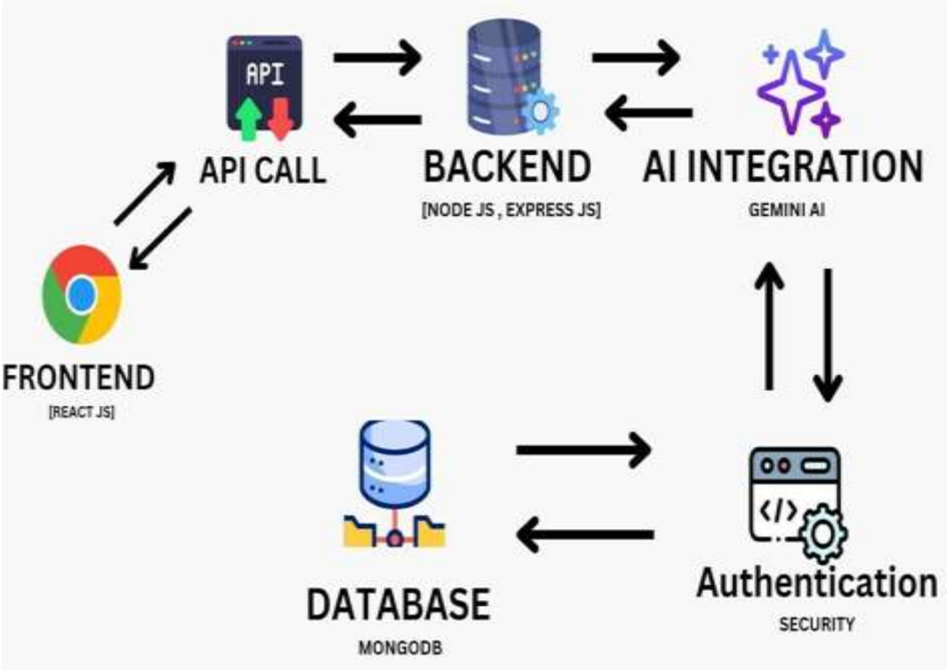
4. METHODOLOGY

The AI-Powered Code Reviewer is developed using a modular and scalable approach with the MERN stack (MongoDB, Express.js, React.js, Node.js) and Gemini AI for intelligent code analysis. The system offers real-time review with accurate AI-driven insights.

- Frontend is built with React.js to provide a user-friendly code submission interface.
- Backend uses Node.js and Express.js to handle API requests and send code to the AI service.
- AI Microservice is developed using Python and integrated with Gemini AI to process and analyze the submitted code.
- Database Layer uses MongoDB to store user data, code submissions, and review history. This structured methodology ensures the system is robust, efficient, and user-friendly**.**

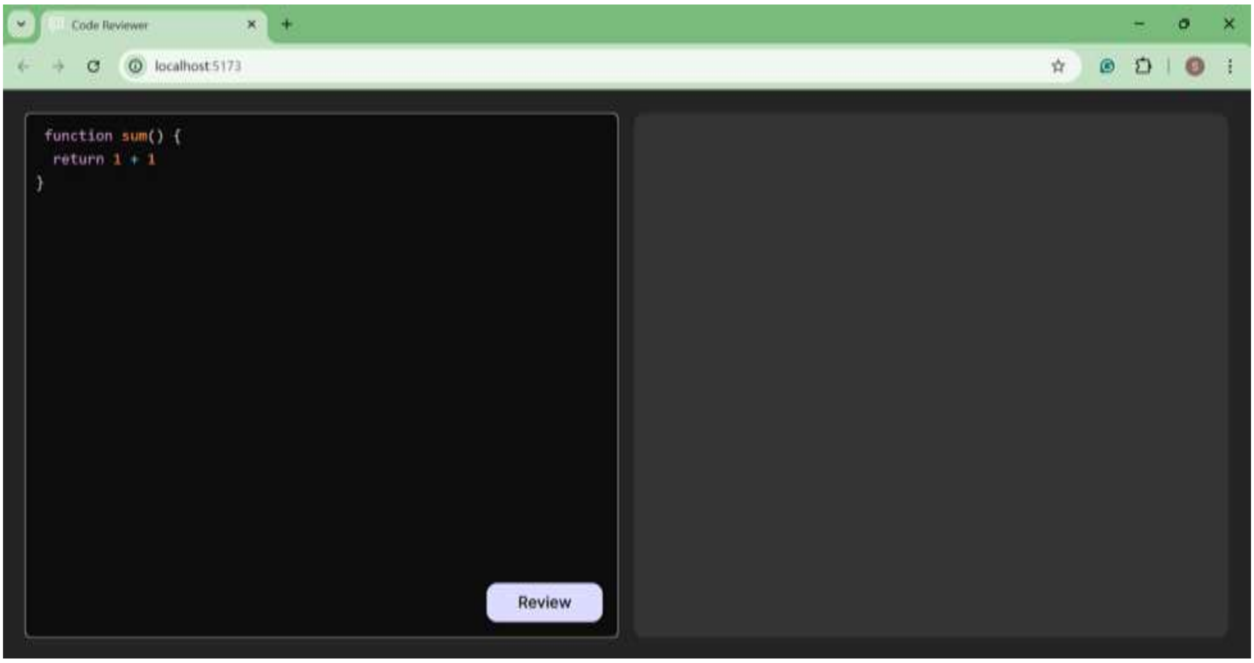
5. SYSTEM ARCHITECTURE

The system architecture of the AI-Powered Code Reviewer follows a layered structure integrating the MERN stack and Gemini AI for seamless functionality. The frontend, developed using React.js, provides an intuitive interface for users to submit code and view AI-generated feedback. The backend, built with Node.js and Express.js, handles API requests and routes code to a Python-based AI microservice. This microservice uses Gemini AI to analyze the code, detect bugs, suggest improvements, and return a detailed review report. MongoDB is used as the database to store user data, code submissions, and review history. The overall workflow ensures smooth communication between components, enabling real-time, intelligent, and secure code reviews.



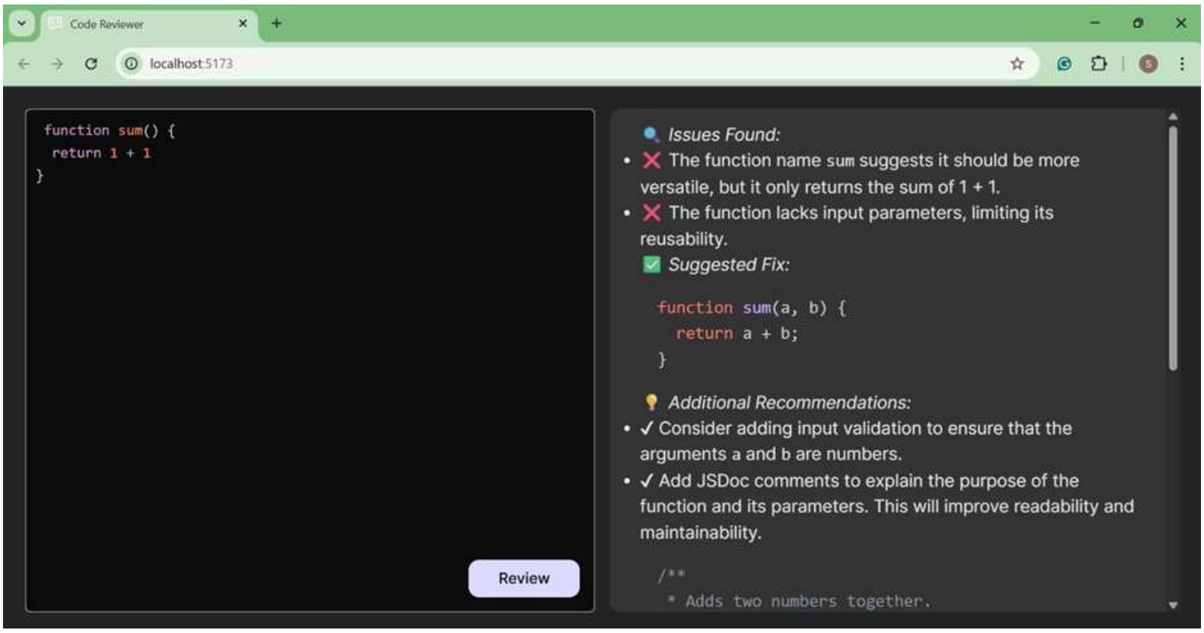
6. RESULTS AND OUTPUT

Fig: User Interface



```
function sum() {
 return 1 + 1
}
```

Fig : Output Prediction



```
function sum() {
  return 1 + 1
}
```

🔍 *Issues Found:*

- ❌ The function name sum suggests it should be more versatile, but it only returns the sum of 1 + 1.
- ❌ The function lacks input parameters, limiting its reusability.

✅ *Suggested Fix:*

```
function sum(a, b) {
  return a + b;
}
```

💡 *Additional Recommendations:*

- ✓ Consider adding input validation to ensure that the arguments a and b are numbers.
- ✓ Add JSDoc comments to explain the purpose of the function and its parameters. This will improve readability and maintainability.

```
/**
 * Adds two numbers together.
```

7. CONCLUSION

The AI-Powered Code Reviewer successfully demonstrates how Artificial Intelligence can enhance the software development process by automating code reviews. By integrating Gemini AI with the MERN stack, the system provides real-time analysis, detects bugs, suggests improvements, and ensures code quality and security. Unlike traditional review methods, this solution offers faster, more accurate, and consistent feedback, improving developer productivity and reducing human error. With features like review history, a user-friendly interface, and potential GitHub integration, the project serves as a valuable tool for developers and sets the foundation for future intelligent coding assistants..

8. FUTURE SCOPE

The AI-Powered Code Reviewer can be enhanced with several advanced features to improve usability and functionality. Future improvements include live linting for instant feedback as users type, AI explainability to show why each suggestion was made, and advanced GitHub integration for syncing reviews with pull requests. Additional features like team collaboration tools, review tracking, gamification with achievements for clean code, and a mobile-friendly interface can further expand its reach. These enhancements will make the tool more interactive, developer-friendly, and suitable for real-world, large-scale software projects.

9. REFERENCES

- Bavishi, R., et al. (2022). "AI-Powered Code Review: Enhancing Software Quality." IEEE Transactions on Software Engineering.
- Zhou, Y., et al. (2021). "Deep Learning for Automated Code Review." Springer. GitHub Copilot: https://github.com/features/copilot
- DeepCode (now part of Snyk): https://snyk.io/product/developer-security/code/
- SonarQube: https://www.sonarsource.com/products/sonarqube/
- Gemini AI Documentation (Google): https://ai.google.dev/gemini-api MERN Stack Guide:
- MongoDB Official Docs: https://www.mongodb.com/
- React Official Docs: https://react.dev/
- Express.js Official Docs: https://expressjs.com/
- Node.js Official Docs: https://nodejs.org/en/docs
- Python Flask: https://flask.palletsprojects.com/