Bug Detection And Fixing Project Report by Pooja D,Mohammed Aasheeq S,Bhuvaneswari T

Self Overview:

Hello all, we are POOJA D, MOHAMEED AASHEEQ S and BHUVANESHWARI T, Information Technology (IT) students from Sri Sairam Institute of Technology. As part of our academic and research interests, we undertook an impactful project focused on AI-powered Bug Detection and Fixing in software code.

This project aimed to build an end-to-end automated pipeline that leverages deep learning and generative AI models to identify functional bugs in source code and provide intelligent fix recommendations. Through this initiative, we explored and applied cutting-edge AI techniques such as transformer-based models (e.g., CodeLlama, CodeGemma, CodeBERT) and integrated these solutions into developer workflows for real-time assistance.

Working on this project gave us hands-on experience in areas such as AI model integration, data pipeline automation, system design, real-time code analysis, and the development of a user-friendly interface for seamless IDE and CI/CD pipeline integration. This endeavor enhanced our technical expertise in both AI and software engineering while contributing toward the vision of intelligent and efficient software development.

Introduction:

Debugging is a crucial phase of the software development lifecycle. Traditional debugging methods are manual, which leads to inefficiency. With advancements in AI, automated bug detection and fixing are now achievable. This project focuses on building a complete bug detection system utilizing state-of-the-art AI models for analyzing and correcting code.

Problem Statement:

Software bugs can lead to system failures, security vulnerabilities, and increased development costs. Detecting and fixing bugs manually is labor-intensive and time-consuming. There is a need for an automated system that not only detects bugs but also recommends accurate fixes.

Objectives:

1.To develop an AI system that can detect bugs in source code.

2.To provide automatic suggestions or fixes for the detected bugs.

3.To integrate a web-based user interface using FastAPI.

4.To support various programming languages and code formats.

5.To log all analysis and results into a database for further review.

Literature Review:

Recent research in program analysis and AI shows that models like CodeBERT and CodeLlama perform well in understanding programming syntax and semantics. Transformer-based models have revolutionized the way source code is represented and processed. Techniques such as static analysis, pattern recognition, and NLP help in identifying syntactic and semantic errors.

Methodology:

1.Data Collection: Use datasets like CodeNet, BigCloneBench, and open-source repositories.

2.Preprocessing: Tokenize and normalize code.

3.Model Integration: Employ transformer-based models like CodeBERT and DeepSeek-Coder-V2.

4.Bug Detection: Use model inference to identify suspicious code patterns.

5.Bug Fixing: Generate patches using generative models like CodeLlama or Qwen2.5-Coder.

System Architecture:

Input Code -> Preprocessing -> Bug Detection Model -> Bug Fix Model -> Response Generator -> FastAPI UI -> Database Logging

Interface: Implement FastAPI frontend for user interaction.

Logging: Store results in a structured database.

Technologies Used:

1.Python

2.FastAPI

3.HuggingFace Transformers

4. CodeBERT, CodeLlama

5.Docker

Implementation:

Modules:

bug\_detector: Uses AI model to detect bugs.

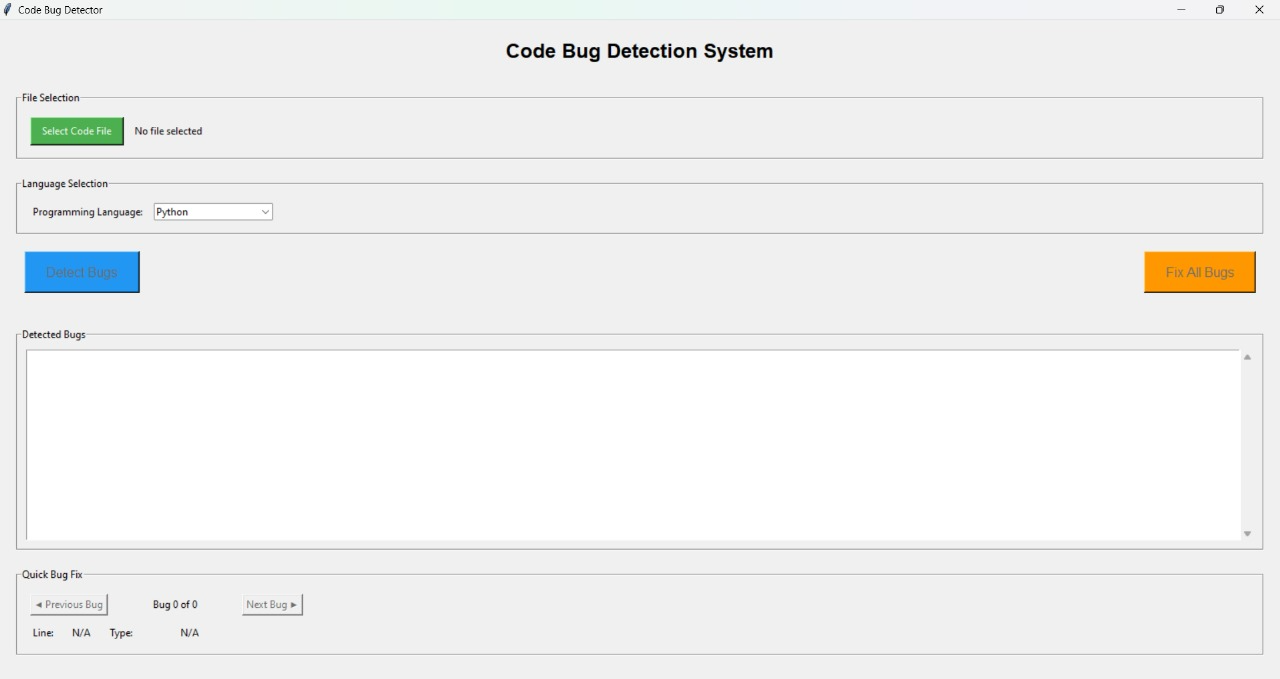
bug\_fixer: Generates fix suggestions.

api\_server: FastAPI-based backend service.

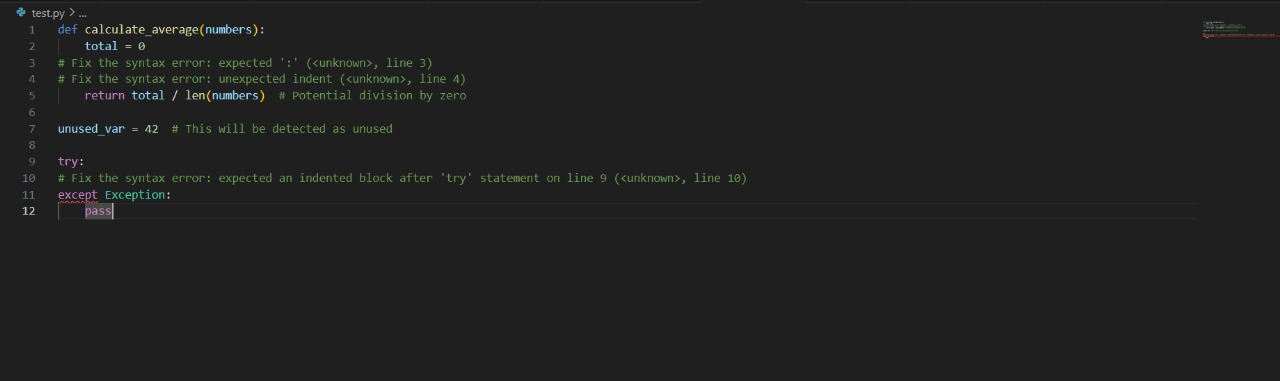
db\_logger: Logs inputs and results.

* The system is containerized for portability and ease of deployment.
* Extensive testing was done using sample buggy codes.

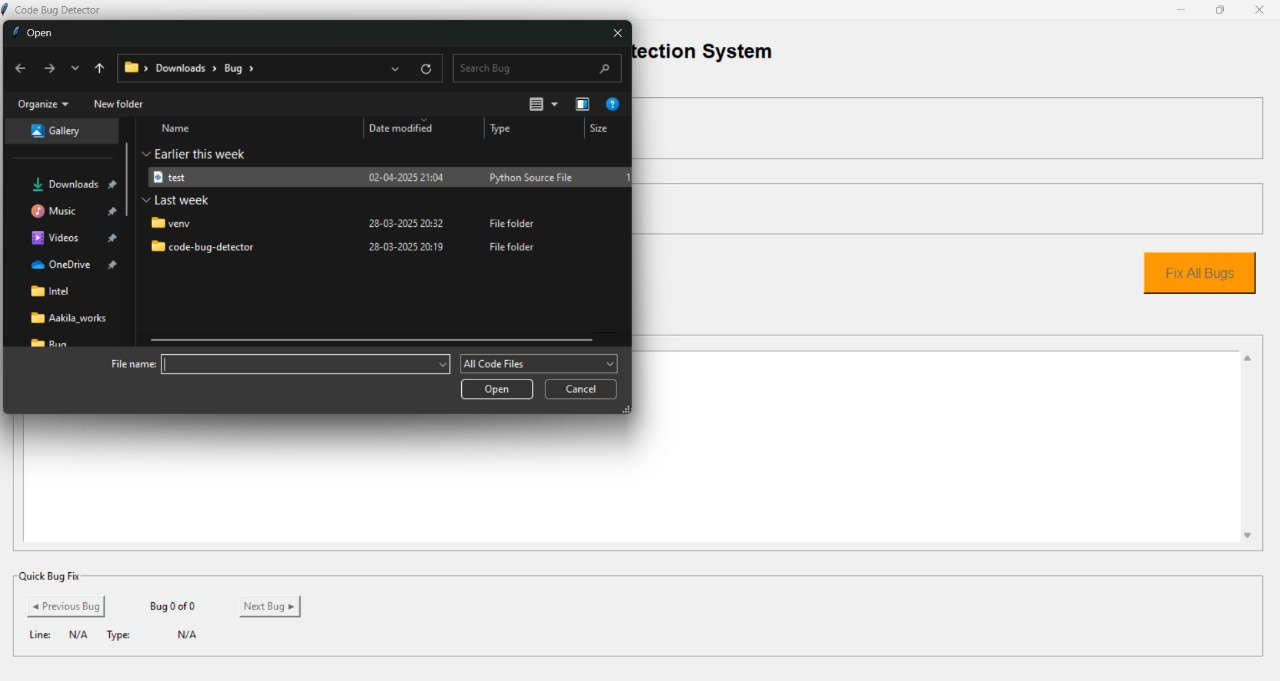
OUTPUT IMAGES:



Sample code:

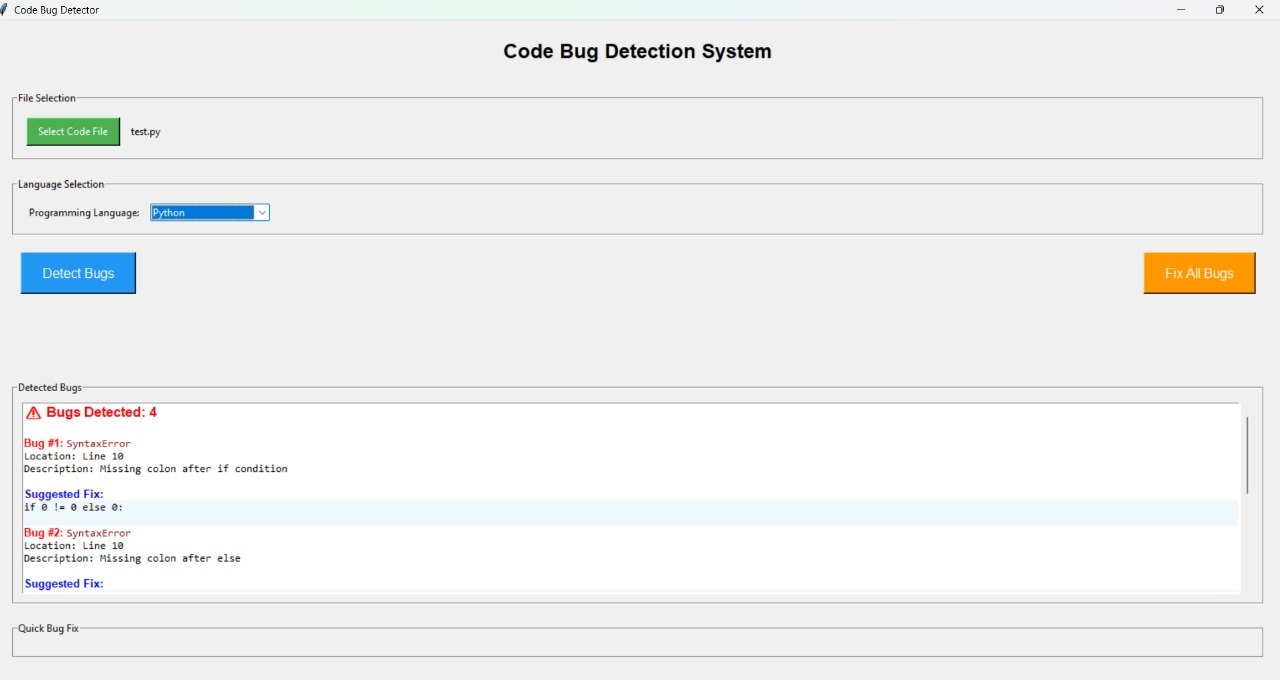


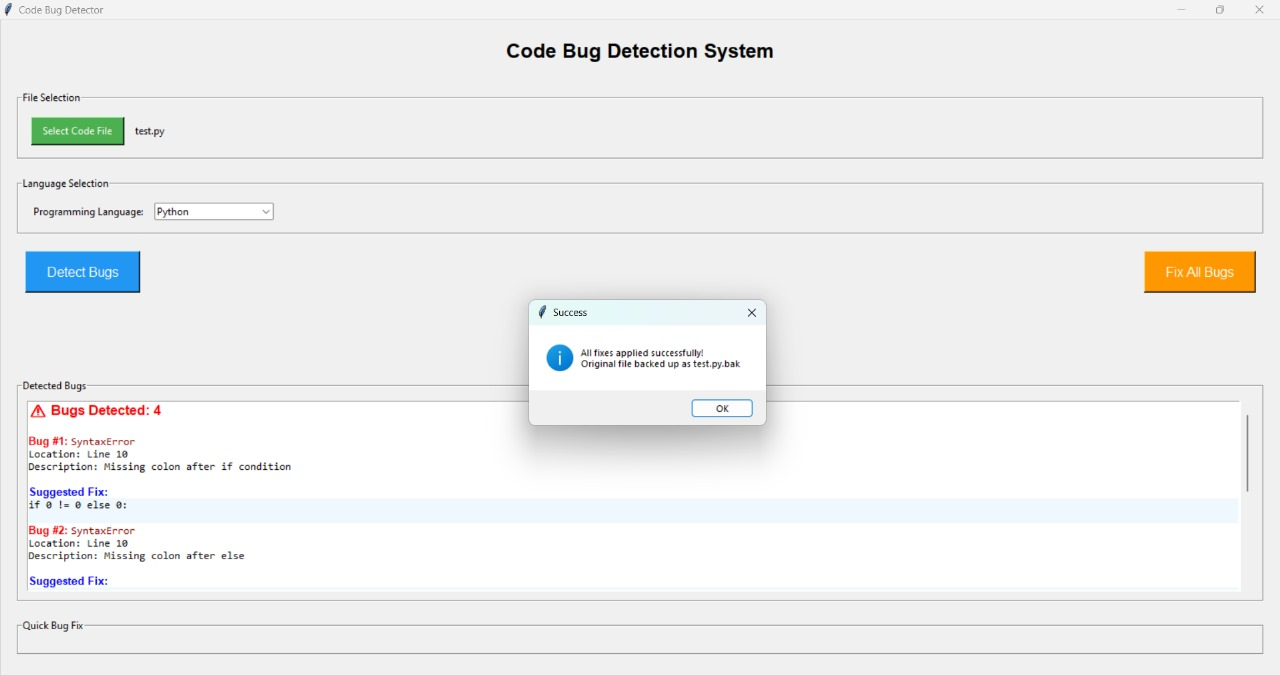
Uploading the sample file:



Testing the sample code:

Detecting Bugs:



Fixing the Bug:

Conclusion:

The project successfully developed an AI-powered system for detecting and fixing software bugs. It enhances developer productivity and code reliability using cutting-edge transformer models. The system’s modular design ensures easy maintenance and scalability.

Future Work:

1.Support for more languages like JavaScript, C++.

2.Real-time integration into popular IDEs.

3.Enhanced natural language explanations of bugs and fixes.

4.Fine-tuning with domain-specific datasets.

References:

1.Microsoft CodeBERT Research Paper

2.CodeXGLUE Benchmark

3.HuggingFace Transformers Documentation

4.GitHub Repositories of CodeLlama and DeepSeek-Coder

5.FastAPI Documentation

Project Link:

https://github.com/Aasheeqcode/Bug-detection