Team Members

- Team Leader /ID: marimuthu A /NM2025TMID07058
- Team member :Murugesan N
- Team member : CHANDHIRAN S
- Team member :Thirumoorthi T

Introduction

The AI Code Analysis & Generator project is designed to simplify software development by automating the requirement analysis and code generation process. It is an academic and practical project showcasing the power of Generative AI in software engineering.

Project Description

This project is a Generative AI-powered tool that can analyze software requirement documents and generate relevant code snippets in different programming languages. It leverages the IBM Granite model to perform requirement analysis and automatic code generation. The tool also integrates with Gradio for an interactive web interface.

Features

- Requirement Analysis: Extracts and organizes functional, non-functional, and technical requirements from PDF or text input.
- Code Generation: Generates code in various programming languages including Python, JavaScript, Java, C++, C#, PHP, Go, and Rust.
- Interactive UI: Built with Gradio, allowing easy interaction for uploading documents, analyzing requirements, and generating code.
- Model Integration: Uses Hugging Face Transformers with IBM Granite for advanced natural language understanding and generation.
- PDF Support: Reads and extracts text from uploaded PDFs using PyPDF2.

Technology Stack

- Python
- Gradio
- Hugging Face Transformers
- IBM Granite 3.2 2B Instruct
- PyTorch
- PyPDF2
- ReportLab (for documentation)

System Architecture

The system follows a modular architecture consisting of the following layers:- Input Layer: Accepts requirements via text or PDF uploads.- Processing Layer: Extracts data, analyzes requirements, and processes text using IBM Granite.- Code Generation Layer: Produces code snippets in the requested programming language.- Presentation Layer: Provides an interactive UI through Gradio for user interaction.

Workflow

1. User uploads a requirements document (PDF) or enters text manually.2. The system extracts and analyzes the requirements into functional, non-functional, and technical categories.3. Based on user input, the model generates code in the chosen programming language.4. Results are displayed interactively in the Gradio interface.

Use Cases

- Academic Projects: Helping students understand requirement analysis and automated coding.- Software Development: Speeding up requirement documentation and code prototyping.- Research: Demonstrating the integration of AI in software engineering tasks.- Training & Learning: Aiding beginners to learn how requirements translate into code.

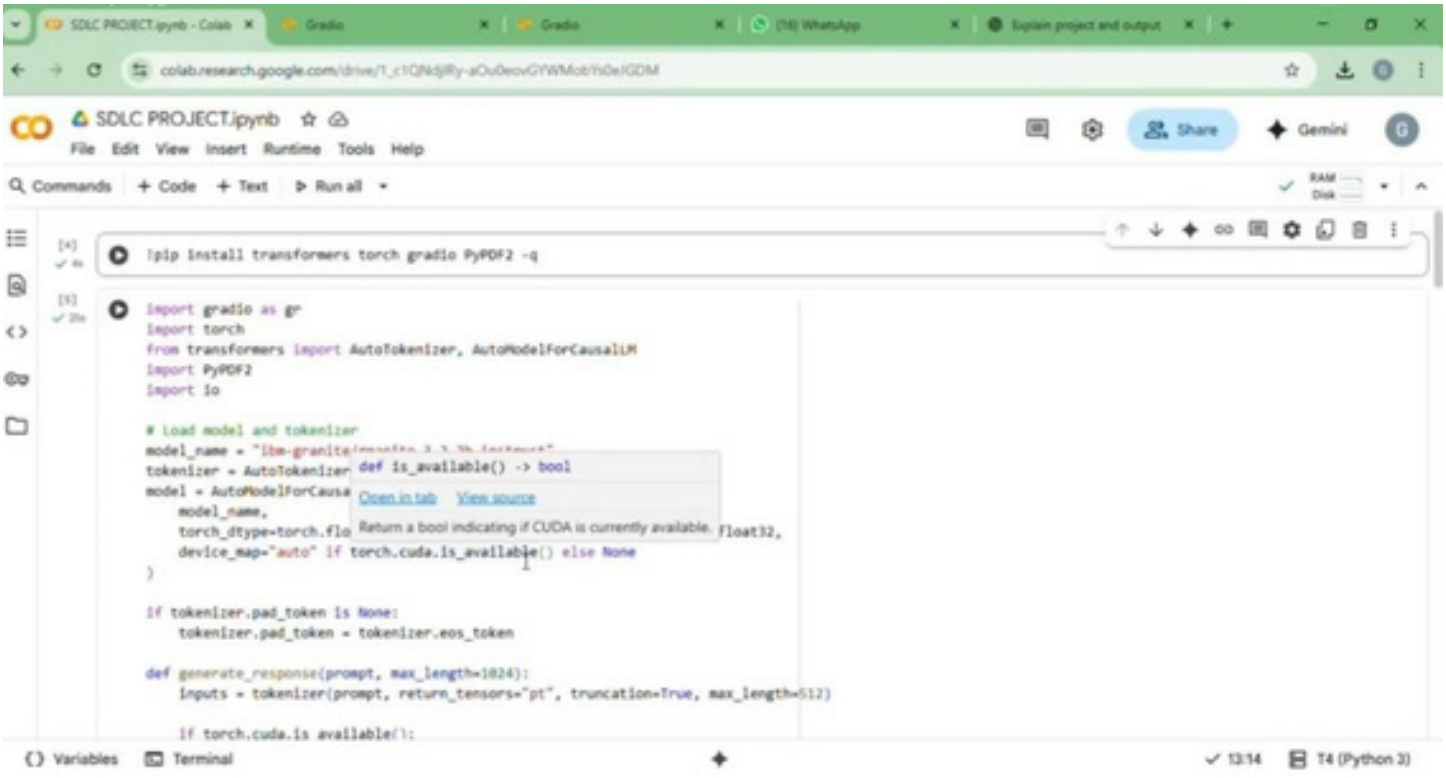
Limitations

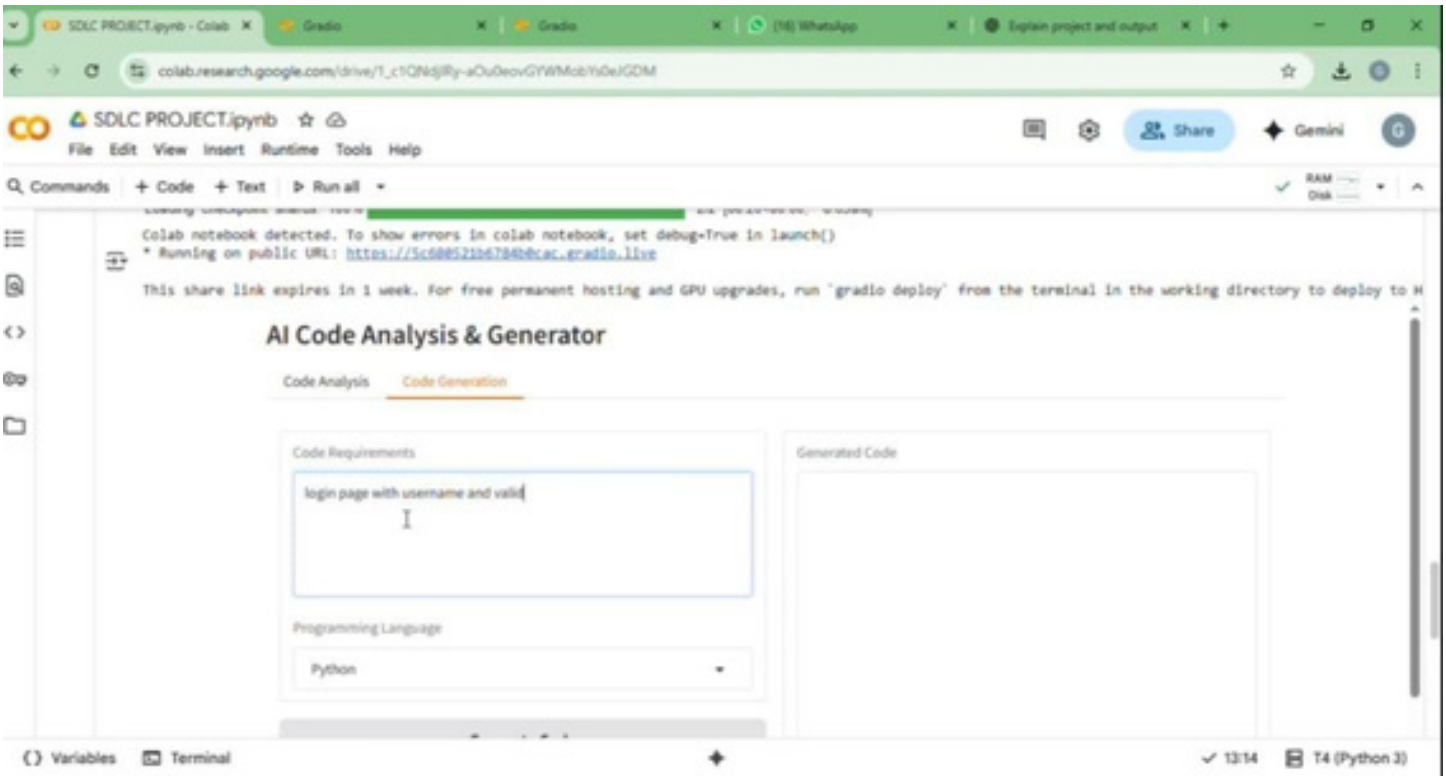
- Limited context understanding in large and complex documents.- Generated code may need refinement and optimization.- Dependency on internet for model downloads and updates.- Currently supports a predefined set of programming languages.

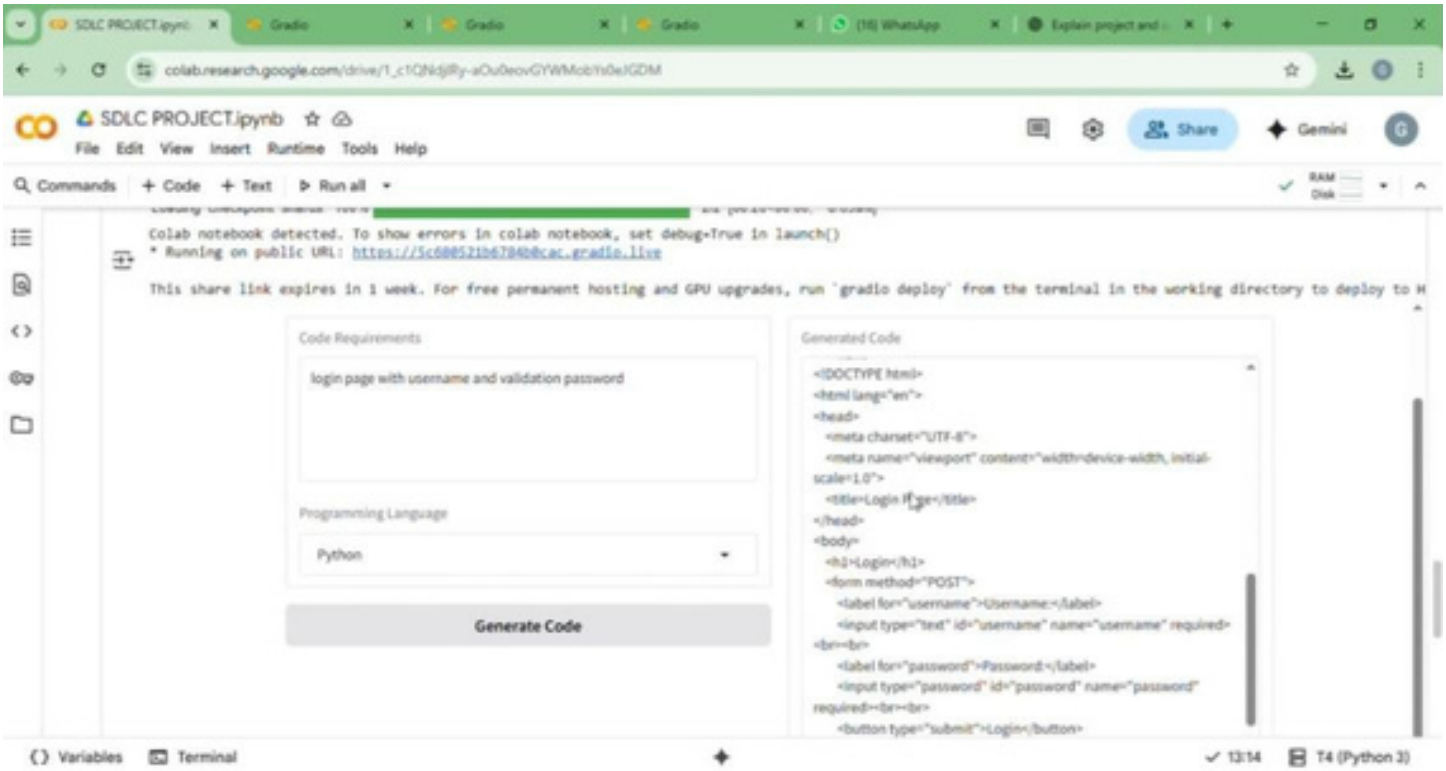
Future Enhancements

- Add support for more programming languages.- Improve contextual understanding for better requirement-to-code mapping.- Integrate with IDEs like VS Code for seamless development.- Enable real-time collaboration features.- Expand to mobile-friendly applications.

Screenshots







Conclusion

The AI Code Analysis & Generator project highlights the importance of Generative AI in modern software engineering. By streamlining requirement analysis and automating code generation, it reduces development time,

improves accuracy, and enhances productivity. This tool lays the foundation for future innovations in AI-driven software development workflows.