■ Project Documentation

Project Name: AI Code Analysis & Generator

Version: 1.0

Date: 2025-09-16

Team member : Karthikeyan

Team member : Prem kumar

Team member : vasanth

Team member : Gowtham

1. Introduction

1.1 Purpose

The purpose of this project is to provide an AI-powered tool that can analyze software requirements and generate corresponding code snippets. The system also supports requirement extraction from uploaded PDF documents, organizing them into functional requirements, non-functional requirements, and technical specifications, and allows exporting the results as a project document (PDF).

1.2 Scope

Input requirements manually or via PDF documents.

Perform automated requirement analysis (functional, non-functional, technical).

Generate source code in multiple programming languages (Python, JavaScript, Java, C++, C#, PHP, Go, Rust).

Create and export structured project documentation (PDF).

Provide a user-friendly Gradio-based interface.

1.3 Intended Users

Software Engineers – to accelerate prototyping.

Business Analysts – to extract structured requirements.

Students / Researchers – to learn requirement analysis and quick code generation.

1.4 System Features Overview

Requirement Analysis (PDF or text input).

Code Generation (multi-language).

Project Documentation PDF Export.

2. Functional Requirements

Requirement Input

The system shall allow uploading of PDF files.

The system shall allow manual entry of requirements.

Requirement Analysis

The system shall analyze requirements and classify them as:

- Functional requirements
- Non-functional requirements
- Technical specifications

Code Generation

The system shall generate code snippets based on user requirements.

The system shall support multiple programming languages.

Documentation Export

The system shall create project documentation in PDF format.

The system shall allow users to download the generated PDF.

3. Non-Functional Requirements

Performance

The system shall generate results within 10 seconds for typical requirements.

Scalability

The system should support large PDF inputs (up to 20MB).

Usability

The system should have a clean, simple UI with tabs for each feature.

Reliability

The system must handle invalid or corrupted PDF files gracefully.

Portability

The system should run on both CPU and GPU environments.

4. Technical Specifications

Frontend: Gradio (Python UI framework).

Backend AI Model: IBM Granite 3.2 (2B Instruct).

Frameworks: Hugging Face Transformers, PyTorch.

Libraries:

- gradio – UI

- torch – Model inference
- transformers – Model & tokenizer
- PyPDF2 – PDF text extraction
- reportlab – PDF generation

Programming Language: Python 3.10+

Deployment: Local execution with optional GPU acceleration.

5. System Design

5.1 Architecture

User Interface (UI): Gradio app with 3 tabs:

- Code Analysis
- Code Generation
- Project Documentation (PDF Export)

Backend Processing:

- Requirement analysis with IBM Granite model.
- Code generation using prompt engineering.
- PDF export using ReportLab.

5.2 Workflow

User uploads PDF or types requirements.

AI analyzes and outputs structured requirements.

User selects programming language for code generation.

AI generates code snippet.

User compiles results into a project document and downloads PDF.

6. Future Enhancements

Add support for diagram generation (UML, ERD).

Integration with Jira/Trello for requirement tracking.

Export to Word (.docx) in addition to PDF.

Allow collaborative editing of requirements before finalizing.