# Smart AI Project Documentation

## Team Information

Team Leader: Lavanya. M

Team Members:

- Pavithra. M

- Yashwanthini. R

- Priyadharshini. K

## 1. Introduction

Project Title: Smart AI: AI-Enhanced Software Development Lifecycle

Objective:  
The Smart AI project leverages IBM Granite models (from Hugging Face) to simplify and accelerate software development. It provides developers with an AI-powered assistant capable of:  
- Understanding uploaded project documents (PDFs).  
- Generating structured requirements.  
- Turning prompts into working code.  
- Creating test cases.  
- Debugging and fixing issues.  
- Writing documentation.  
- Offering real-time AI-powered chat support.  
  
By integrating these features into a simple Gradio interface, Smart AI aims to improve developer productivity and ensure faster, more reliable software delivery.

## 2. Project Overview

Features:

1. Requirement Extraction – Analyze uploaded PDFs to generate clear requirements.  
2. Code Generation – Convert natural language prompts into functional code.  
3. Test Automation – Auto-generate test cases to validate functionality.  
4. Debugging – Detect and fix errors in uploaded or generated code.  
5. Documentation – Generate project documentation from code and requirements.  
6. AI Chat Support – Interactive assistant for queries during development.

Purpose:  
- Reduce the time needed for software development.  
- Provide intelligent automation across the SDLC.  
- Help beginners and professionals accelerate project delivery.

## 3. Architecture

Frontend (Gradio UI):  
- Upload PDFs, enter prompts, and interact with AI through a web interface.  
- Tabs for Requirements, Code Generation, Testing, Debugging, and Documentation.

Backend (Python + Transformers):  
- IBM Granite LLM model handles prompt understanding and generation.  
- PyPDF2 for extracting text from uploaded PDFs.  
- Integration with Hugging Face for model hosting.

Deployment:  
- Runs in Google Colab (with T4 GPU support).  
- Can be shared publicly via Gradio’s launch(share=True).  
- Code stored and version-controlled in GitHub.

## 4. Project Workflow

1. Exploring Smart Internz Portal – Enroll and access the guided project workspace.  
2. Choosing IBM Granite Model – Use Hugging Face to select a lightweight Granite model (granite-3.2-2b-instruct).  
3. Running in Google Colab – Install dependencies, set runtime GPU, and execute code.  
4. Uploading to GitHub – Push project code to a repository with documentation.

## 5. Setup Instructions

Prerequisites:  
- Python 3.9 or later  
- pip (package installer)  
- Gradio, Transformers, Torch, PyPDF2  
- GitHub account for version control  
- Google Colab with GPU enabled

Installation:  
  
pip install gradio torch transformers PyPDF2

Steps:  
1. Open Google Colab → Create New Notebook.  
2. Set runtime to T4 GPU.  
3. Install required libraries.  
4. Load the IBM Granite model.  
5. Run project cells to launch the Gradio app.

## 6. Folder Structure

smart\_ai\_project/  
│── app.py # Main Gradio app  
│── requirements.txt # Dependencies  
│── utils/ # Helper scripts  
│── docs/ # Generated documentation  
│── tests/ # Test cases  
│── data/ # Uploaded project PDFs

## 7. Future Enhancements

- Add authentication and user role management.  
- Convert Gradio app into REST API with FastAPI.  
- Expand support for multiple programming languages.  
- Integrate CI/CD pipelines for automated testing.  
- Provide mobile-friendly interface.