SmartSDLC-AI-Enhanced Software Development Lifecycle

Project Documentation

1. Introduction

- **Project Title:** SmartSDLC – AI-Enhanced Software Development Lifecycle
- **Team Member: Kokila.I**
- **Team Member: Saffena Nasrieen.B**
- **Team Member: Vijaya SRI.R**
- **Team Member: Nishanthini.M**

---

2. Project Overview

Purpose:
The purpose of SmartSDLC is to automate and streamline the **Software Development Lifecycle (SDLC)** by leveraging **AI, LLMs, and real-time analysis**. It enhances efficiency for developers, analysts, and project managers by providing automated requirement analysis, intelligent code generation, bug detection, and policy summarization. SmartSDLC acts as an intelligent assistant to improve productivity, reduce errors, and optimize the overall software development process.

Features:

- **Requirement Analysis**
  - *Key Point:* AI-driven requirements extraction
  - *Functionality:* Analyzes uploaded documents (PDFs, text) and organizes requirements into **functional, non-functional, and technical** categories.
- **Code Generation**
  - *Key Point:* Multi-language code synthesis
  - *Functionality:* Generates code in multiple programming languages (Python, Java, C++, etc.) based on natural language requirements.
- **Bug Fixing & Suggestions**
  - *Key Point:* Automated debugging assistance
  - *Functionality:* Identifies potential issues in given code snippets and suggests optimized fixes.
- **Policy Summarization**
  - *Key Point:* Simplified document understanding
  - *Functionality:* Converts lengthy software specifications or compliance documents into clear, actionable summaries.
- **Test Case Creation**
  - *Key Point:* Automated QA support
  - *Functionality:* Generates unit test cases from functional requirements to ensure coverage.
- **Eco Tips for Developers (Optional Extension)**

    - *Key Point:* Sustainable coding practices
    - *Functionality:* Provides tips on writing energy-efficient, optimized, and maintainable code.
- **Gradio/Streamlit UI**
    - *Key Point:* User-friendly AI interface
    - *Functionality:* Provides an intuitive dashboard with **code analysis, requirement upload, and generation features**.

---

3. Architecture

- **Frontend (Gradio / Streamlit):**
  Provides an interactive web interface with tabs for **requirement analysis, code generation, and bug fixing**. Users can upload PDFs, type requirements, and view AI outputs in real-time.
- **Backend (FastAPI):**
  Serves as the REST framework that handles **API requests for analysis, generation, summarization, and testing**.
- **LLM Integration (IBM Watsonx Granite):**
  Uses **Granite LLM models** for natural language understanding, code synthesis, and summarization.
- **Document Processing (PyPDF2):**
  Extracts text from uploaded requirement PDFs for analysis.
- **ML Modules (Bug Detection & Optimization):**
  Machine learning models analyze and detect