SmartSDLC – AI-Enhanced Software Development Lifecycle

Project Documentation

---

1. Introduction

Project Title: SmartSDLC – AI-Enhanced Software Development Lifecycle
Team Leader: Deepika K
Team Members:

Brindha U (upragathi2006@gmail.com)

Agneeswari M (agneeswari2004@gmail.com)

Ananthi V (ananthi8270067797@gmail.com)

Bavadharani D (mdhanabal421@gmail.com)

---

2. Project Overview

Purpose

SmartSDLC is an AI-powered solution designed to optimize and automate the Software Development Lifecycle (SDLC). It leverages AI to improve planning, coding, testing, deployment, and maintenance, making software development more efficient, accurate, and adaptive.

For Developers → Provides intelligent code suggestions, bug detection, and documentation automation.

For Project Managers → Generates project timelines, risk assessments, and progress insights.

For Testers → Automates test case generation, execution, and defect prediction.

👉 SmartSDLC aims to bridge AI + Software Engineering, ensuring faster delivery, reduced costs, and high-quality software products.

Key Features

1. AI-Assisted Requirement Analysis – Converts client requirements into structured user stories.

2. Intelligent Project Planning – Generates project timelines, sprint breakdowns, and risk factors.

3. Smart Code Assistant – Provides code suggestions, debugging help, and documentation.

4. Automated Testing – Creates unit tests, integration tests, and bug predictions.

5. Continuous Integration & Deployment (CI/CD) – Suggests optimized deployment workflows.

6. Maintenance & Monitoring – Tracks performance, logs, and suggests improvements.

7. Interactive Dashboard (Gradio/Streamlit) – Single platform for developers, testers, and managers.

---

3. Architecture

Frontend (Gradio/Streamlit)

Provides a user-friendly interface for developers & managers.

Modules: Requirement input, project planner, coding assistant, testing suite, deployment monitor.

Backend (Hugging Face + FastAPI)

Powered by IBM Granite / Hugging Face LLMs for requirement analysis, code suggestions, and document generation.

FastAPI enables modular service handling.

LLM Integration

Granite 3.2.2-b Instruct model processes requirements, generates documentation, and assists in coding/testing.

Development Environment

Built and tested on Google Colab for rapid prototyping.

Key Dependencies

Transformers → AI model integration.

Torch → Model inference.

Gradio/Streamlit → Frontend.

FastAPI → Optional API layer for modularity.

---

4. Setup Instructions

Prerequisites:

Python 3.9+

pip package manager

Google account (Colab)

IBM/Hugging Face API key

Steps:

1. Open Google Colab → Create new Python file.

2. Install required libraries:

pip install transformers torch gradio fastapi

3. Configure API key in .env or inside Colab.

4. Import libraries, load Granite model, and link with Gradio.

5. Run notebook → SmartSDLC app launches.

---

5. Folder Structure

```
SmartSDLC/
├── SmartSDLC.py        # Main Colab script
├── requirements.txt    # Dependencies
├── .env                # API key
├── /modules            # Requirement, coding, testing, deployment
```

Frontend → Developer/Manager dashboard.

Backend → AI logic for requirement analysis, testing, coding.

Config → API setup & dependencies.

---

6. Running the Application

1. Open Colab → Run SmartSDLC.py.

2. Install dependencies.

3. Add API key in script.

4. Run notebook cells.

5. Use dashboard for:

Entering project requirements.

Generating project plan.

Getting code/test suggestions.

Monitoring deployment/maintenance.

---

7. API Documentation

SmartSDLC is Gradio-driven, so APIs are embedded.

Core Functions:

Requirement Analysis

Input: Raw project description.

Output: Structured requirements & user stories.

Code Assistant

Input: Module/logic request.

Output: AI-suggested code.

Test Case Generator

Input: Function/module.

Output: AI-generated unit/integration test cases.

Project Planner

Input: Requirement set.

Output: Timeline, sprint plan, risks.

---

8. User Interface

The dashboard provides:

1. Requirement Analysis Module → Converts client needs into structured format.

2. Coding Assistant → Suggests and explains code snippets.

3. Test Automation Suite → Creates and executes test cases.

4. Project Planner → Sprint & timeline generator.

5. Deployment Monitor → Suggests CI/CD workflows and tracks performance.

👉 Designed for developers, testers, and managers to ensure smooth, AI-driven software lifecycle management.