SmartSDLC – AI-Enhanced Software Development Lifecycle

1. Introduction

Project Title: **SmartSDLC** – AI-Enhanced Software Development Lifecycle

Team Members:

Afra Juwairiya

Dharshini S

Varshini D V

Varshini J

Rizwana G

2. Project Overview

Purpose:

The goal of SmartSDLC is to streamline the Software Development Lifecycle (SDLC) using AI technologies such as IBM Watsonx Granite LLM, FastAPI, LangChain, and Streamlit.

The platform automates requirement classification, code generation, debugging, and documentation, reducing manual effort and improving productivity.

Key Features:

Requirement Upload & Classification – Upload unstructured requirements in PDF, classify them into SDLC phases, and convert into structured user stories.

AI Code Generator – Convert natural language prompts into production-ready code.

Bug Fixer – Identify and fix syntax & logic issues in code automatically.

AI-Powered Chatbot – Real-time project assistance.

Modular & Secure Backend – Built using FastAPI, scalable and API-documented with Swagger UI.

3. SDLC Phases with Examples

Phase 1: Requirement Analysis

Users upload PDFs with unstructured requirements (e.g., "System should allow citizens to check city safety").

AI (Watsonx Granite) classifies into Requirements, Design, Development, Testing, Deployment.

Structured User Story Example:

As a citizen, I want to check the safety index of my city so that I can plan my travel safely.

Phase 2: System Design

Backend: FastAPI for API management, LangChain for workflow integration.

Frontend: Streamlit dashboard for interactive visualizations.

Authentication: JWT-based role authentication (future enhancement).

Example Design Output:

Entity Diagram for “Citizen Query → AI Model → Response Generator”.

API Design: POST /analyze_city, POST /generate_code, POST /fix_bug.

Phase 3: Development

AI-powered code generator creates functional code from natural prompts.

Example Prompt:

“Generate a Python function to calculate crime rate per 1000 citizens using input statistics.”

Example AI-Generated Output:

```
def crime_rate(population, crimes):
    if population <= 0:
        return "Invalid input"
    return (crimes / population) * 1000
```

Phase 4: Testing

Unit Testing: Each generated code snippet is tested automatically.

Example Test Case:

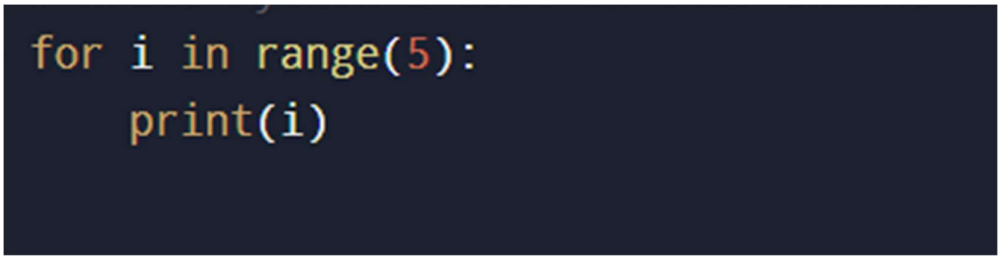
```
def test_crime_rate():
    assert crime_rate(1000, 20) == 20
    assert crime_rate(5000, 50) == 10
    assert crime_rate(0, 5) == "Invalid input"
```

Bug Fixer Example:

Input Code:

```
for i in range(5)
    print(i)
```

AI-Fixed Output:



Phase 5: Deployment & Maintenance

Deployed with Uvicorn + FastAPI for backend and Streamlit for frontend.

Swagger UI provides API documentation.

Future Enhancements:

Real-time database integration (crime stats, weather).

Role-based authentication (citizen, admin).

Multi-language support.

4. Example Workflow (End-to-End)

Upload City Safety Requirements PDF → AI classifies text into SDLC phases.

Select “Code Generator” tab → Enter prompt “Generate accident report analyzer in Python” → AI returns working code.

Paste buggy code snippet in Bug Fixer tab → AI corrects errors and displays both versions.

All results visible in Streamlit dashboard, exportable to PDF.

5. Conclusion

SmartSDLC shows how AI can automate the entire SDLC – from requirement analysis to deployment – making development faster, accurate, and more efficient. This reduces manual effort, improves software quality, and helps teams focus on innovation.