SmartSDLC -AI-Enhanced Software Development Lifecycle

Generative AI with IBM

---

1. Introduction

- **Project Title:** SmartSDLC – AI-Enhanced Software Development Lifecycle
- **Team Members:**
  - S. Iniyaval
  - R. Deva Dharshini
  - R. Atchaya
  - U. Aafrin

---

2. Project Overview

Purpose:
SmartSDLC aims to integrate AI across the entire software development lifecycle to optimize productivity, improve software quality, and enhance decision-making. By leveraging **Generative AI with IBM Watsonx Granite LLMs**, the framework supports developers, managers, and stakeholders with real-time insights, policy summarization, anomaly detection, and predictive analytics.

Features:

- Conversational Interface (natural language queries)
- Policy & Document Summarization
- Resource & KPI Forecasting
- Eco-Tip Generator (sustainability insights)
- Feedback Loop for stakeholders

- Anomaly Detection (early warnings)
- Multimodal Input Support (text, PDFs, CSVs)
- Streamlit/Gradio UI for interaction

---

3. Architecture

- **Frontend (Streamlit):** Interactive dashboards, chat, reports, feedback forms
- **Backend (FastAPI):** Handles APIs for chat, summarization, embedding, forecasting
- **LLM Integration (IBM Watsonx Granite):** Provides natural language understanding, summarization, and recommendations
- **Vector Search (Pinecone):** Embedding + semantic search of documents
- **ML Modules:** Forecasting & anomaly detection (Scikit-learn, Pandas, Matplotlib)

---

4. Setup Instructions

Prerequisites:

- Python 3.9+
- pip & venv tools
- IBM Watsonx API keys
- Pinecone API key
- Internet connection

Installation Steps:

1. Clone repository

2. Install dependencies (requirements.txt)
3. Create .env file for credentials
4. Run backend with FastAPI
5. Launch frontend with Streamlit
6. Upload documents and interact

---

5. Folder Structure

- app/ – FastAPI backend logic
- app/api/ – Modular API routes (chat, feedback, report, embeddings)
- ui/ – Streamlit pages and layouts
- smart_dashboard.py – Main dashboard entry point
- granite_llm.py – Watsonx LLM integration
- document_embedder.py – Embedding & Pinecone storage
- kpi_file_forecaster.py – KPI forecasting module
- anomaly_file_checker.py – Anomaly detection module
- report_generator.py – AI-generated reports

---

6. Running the Application

1. Start FastAPI server
2. Run Streamlit dashboard
3. Navigate via sidebar
4. Upload docs/CSVs
5. Interact with assistant (chat, reports, predictions)

6. Get outputs in real-time

---

7. API Documentation

- POST /chat/ask – AI-generated Q&A
- POST /upload-doc – Document embedding
- GET /search-docs – Semantic document search
- GET /get-eco-tips – AI-driven sustainability tips
- POST /submit-feedback – Feedback storage

---

8. Authentication

- Token-based (JWT / API keys)
- OAuth2 (IBM Cloud)
- Role-based access (admin, citizen, researcher)
- Planned: sessions + history tracking

---

9. User Interface

- Sidebar navigation
- KPI visualizations with summary cards
- Tabbed chat, eco-tips, forecasting views
- Real-time forms
- Downloadable PDF reports

---

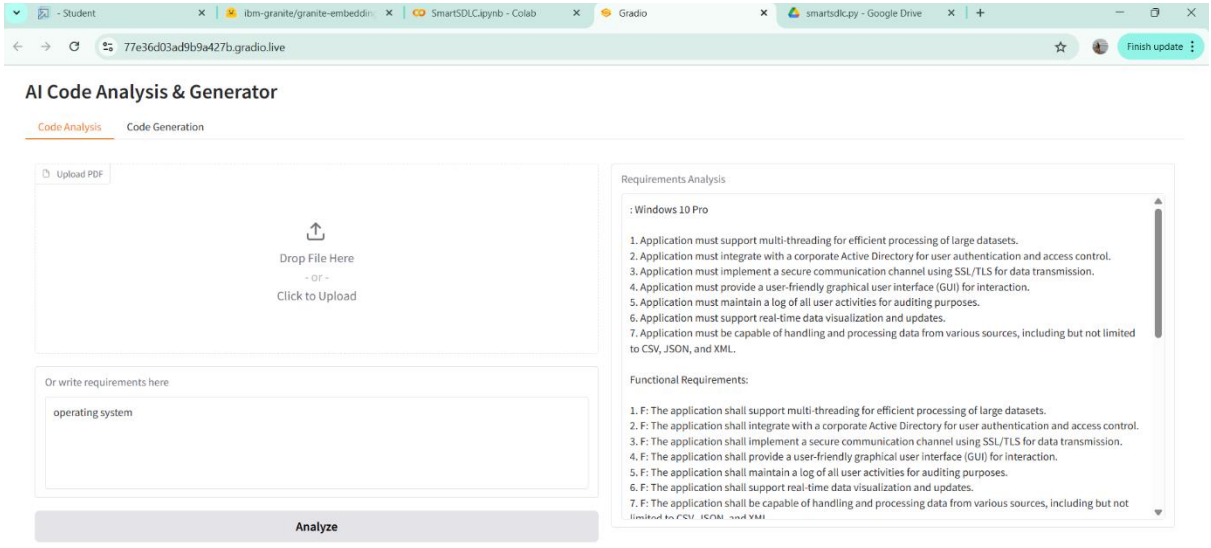
10. Testing

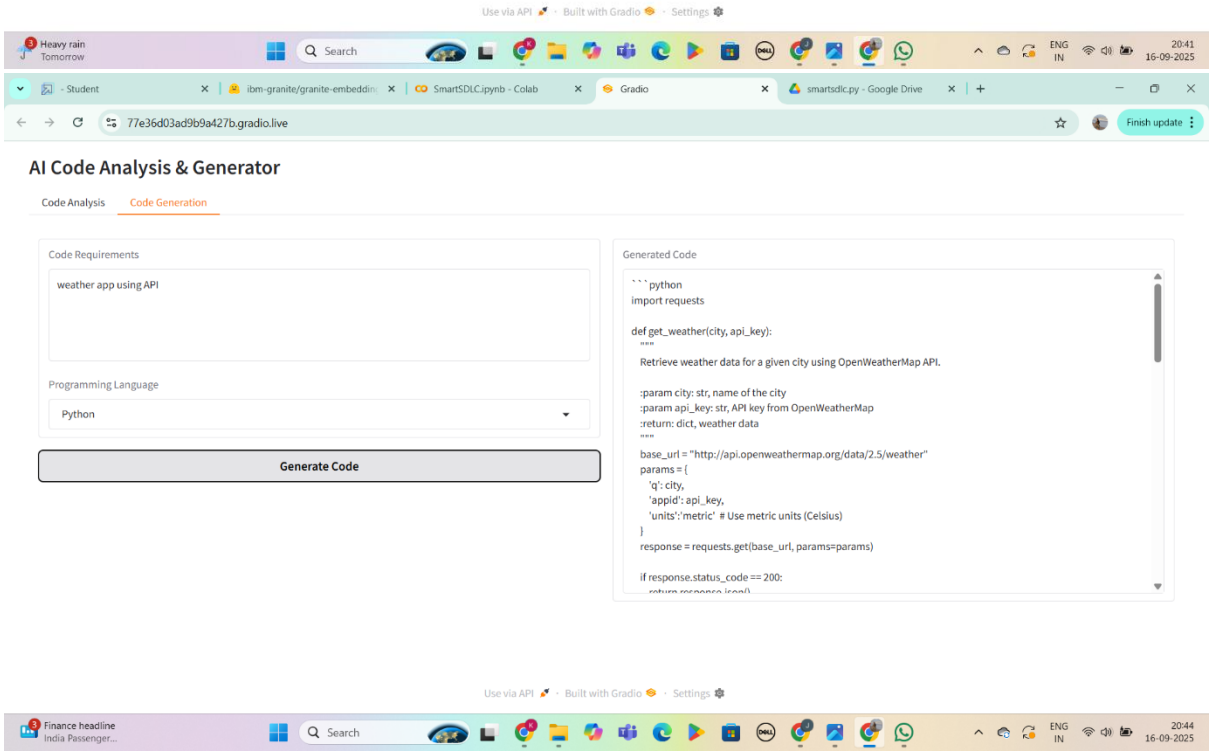
- **Unit Testing:** prompt functions & utilities

- **API Testing:** Swagger UI & Postman
- **Manual Testing:** file uploads, chat, predictions
- **Edge Cases:** large files, malformed inputs, invalid keys

---

11. Screenshots





---

12. Known Issues

- Limited scalability for very large datasets
- Latency in high-volume document embedding
- No offline mode (requires IBM Watsonx API access)

---

13. Future Enhancements

- Advanced security & role management
- Session management & user history
- Deeper Watsonx model fine-tuning
- Expanded forecasting models
- Enterprise/government deployment