SmartSDLC – AI-Enhanced Software Development Lifecycle

Generative AI with IBM

Project Documentation

1.Introduction

• Project title:

SmartSDLC – AI-Enhanced Software Development Lifecycle

• Team members:

- Dharani.R
- Aruli.S
- Gayathri.M
- Angel christy.G
- Dharshini.R

2.Project Overview

Purpose:

The purpose of SmartSDLC is to modernize the traditional Software Development Lifecycle by embedding Artificial Intelligence into each phase—planning, analysis, design, coding, testing, deployment, and maintenance. AI-driven insights, automation, and predictive analytics reduce errors, accelerate delivery, and ensure higher software quality. The system acts as a digital project partner—assisting developers, testers, and managers with intelligent recommendations, workflow automation, and real-time monitoring.

Features:

- Conversational AI Assistant
    - Key Point: Natural language project interaction

    - Functionality: Allows stakeholders to ask queries about project status, deadlines, bugs, and requirements in plain language.
- Automated Requirement Analysis
    - Key Point: NLP-driven requirement gathering
    - Functionality: Extracts, analyzes, and validates requirements from documents or user input.
- AI-driven Code Review
    - Key Point: Intelligent quality checks
    - Functionality: Reviews code for bugs, security flaws, and best practices.
- Test Case Generation
    - Point: Automated testing
    - Functionality: Creates and executes test cases based on project requirements and code changes.
- Effort & Risk Prediction
    - Key Point: Project planning support
    - Key Functionality: Predicts project effort, cost, timeline risks using historical and real-time project data.
- Continuous Monitoring & Feedback
    - Key Point: Post-deployment intelligence
    - Functionality: Tracks performance, user feedback, and suggests improvements.

3. Architecture

- **Frontend (Streamlit/Gradio):** Interactive dashboards, project timelines, bug reports, and AI chat assistant.

- **Backend (FastAPI):** Handles lifecycle workflows, data storage, and AI model orchestration.
- **LLM Integration (Watsonx / OpenAI):** Provides NLP support for requirement analysis, code review, and documentation.
- **Vector Search (Pinecone / FAISS):** Stores project documents and allows semantic search.
- **ML Modules:** Predictive models for project risk analysis, testing automation, and defect detection.

4. Setup Instructions

(Similar structure: Python, APIs, environment setup, etc.)

5. Folder Structure

- app/ – Backend logic (lifecycle APIs, code analysis, testing modules)
- ui/ – Frontend dashboards for monitoring and interaction
- ai_models/ – AI modules for NLP, code review, and testing
- project_forecaster.py – Predicts risks, effort, and deadlines
- bug_detector.py – Flags coding issues and vulnerabilities
- report_generator.py – Generates AI-driven project reports

6. Running the Application

- Start backend server with FastAPI
- Run dashboard with Streamlit
- Upload project documents/code for AI review
- Interact with the lifecycle assistant

7. API Documentation

Examples:

- POST /analyze-requirements – Extracts requirements from docs
- POST /review-code – Returns AI-based code review
- GET /generate-tests – Auto-creates test cases
- POST /predict-risk – Forecasts project risks

8. Authentication

Role-based access: Admin, Developer, Tester, Manager.

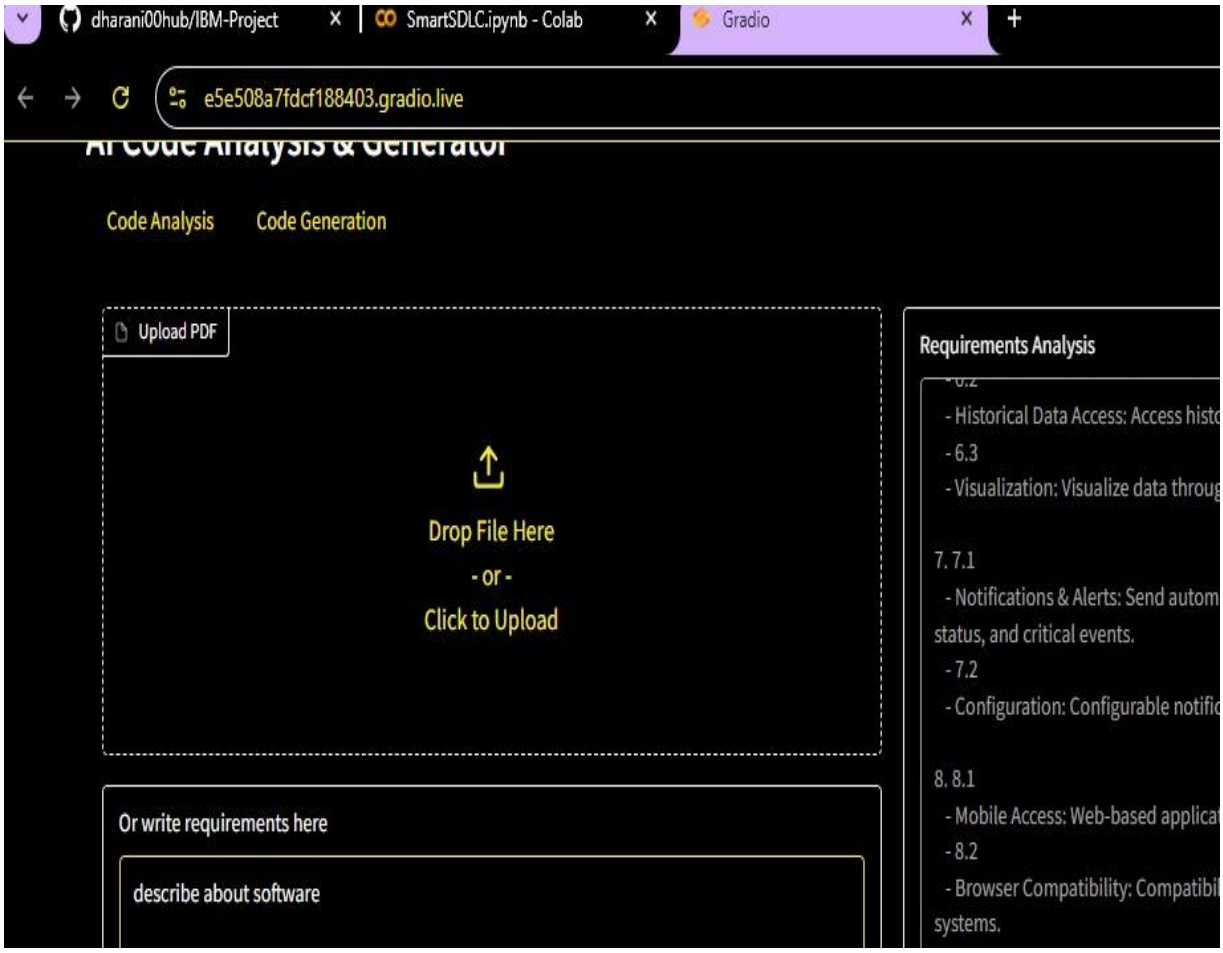
9. User Interface

- Sidebar navigation (Requirements, Code Review, Testing, Reports)
- Real-time dashboards for progress & bug tracking
- AI-powered chat assistant for project queries

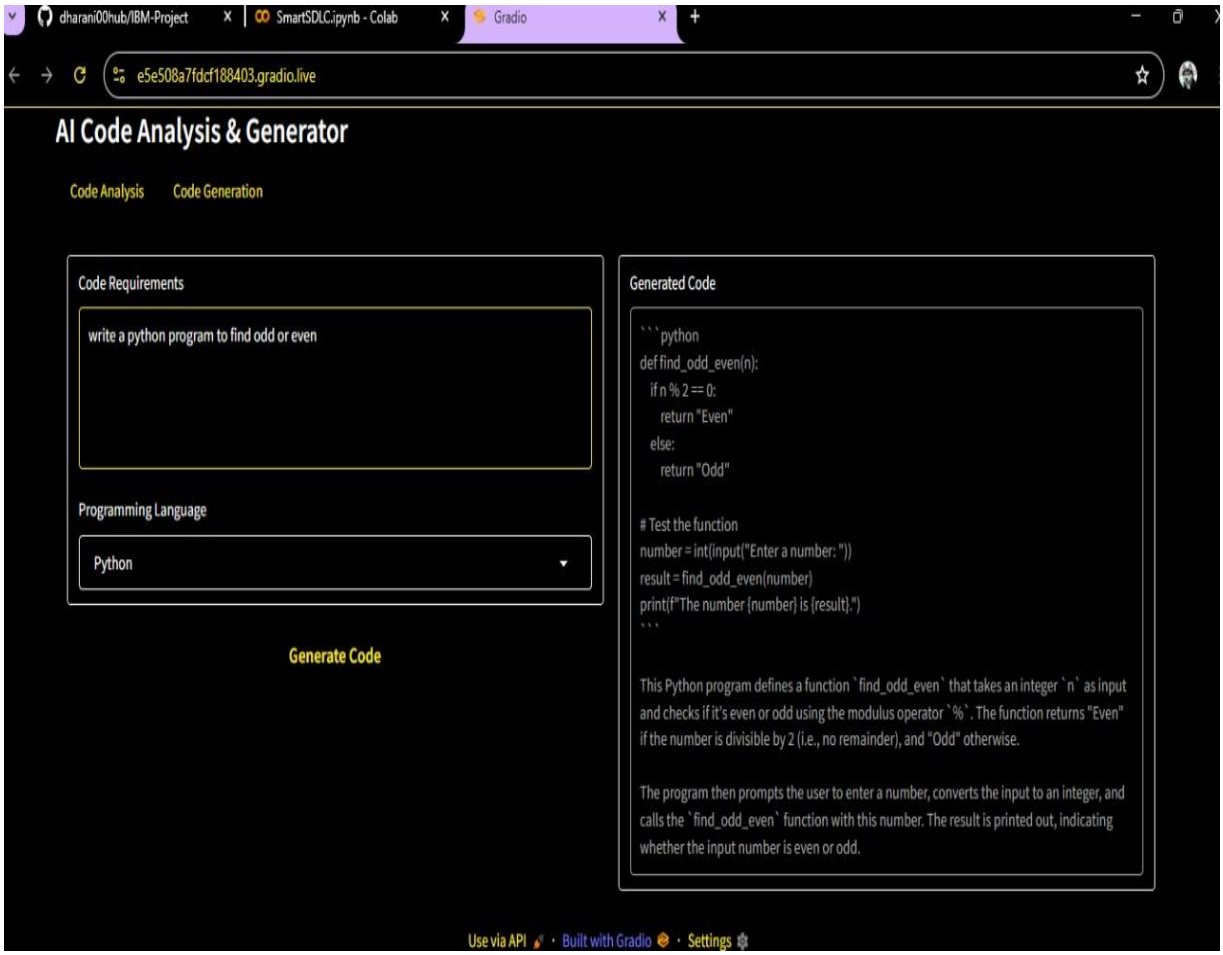
10. Testing

- Unit testing for AI models
- API testing with Postman/Swagger
- Automated regression testing for updates

11. Screenshots



Provide any requirement, and it will analyze and return the result



It will analyze the python code and generate the code

12. Known Issues

- Accuracy depends on dataset quality
- Limited explainability for AI-driven risk predictions

13. Future Enhancements

- Integration with DevOps pipelines (CI/CD)
- Support for multi-language projects
- Enhanced predictive analytics with larger datasets