PROJECT REPORT ON SmartSDLC – AI-Enhanced Software Development Lifecycle

https://huggingface.co/spaces/23ucs589/AIsmartsdlc

TEAM MEMBERS :

S.NO	NAME	REG.NO
01	S.Sathish	23053161802111089
02	S.Syamsanthosh	23053161802111098
03	S.Syedfahad	23053161802111099
03	R.Tamilarasan	23053161802111100

ANNAI VAILANKANNI ARTS AND SCIENCE COLLEGE

OWNED AND MANAGED BY"DIOCESE OF TANJORE SOCIETY"(REGD.NO.S.8 OF 1958)

(AFFILIATED TO BHARATHIDASAN UNIVERSITY,TIRUCHIRAPPALLI-24)

(AFFILIATED TO BHARATHIDASAN UNIVERSITY,TIRUCHIRAPPALLI-24)

BISHOP SUNDARAM CAMPUS,PUDUKOTTAI ROAD,THANJAVUR–613007.

index

1.Introduction

The Software Development Lifecycle (SDLC) provides a structured framework for planning, creating, testing, and maintaining software systems. Traditionally, this process relies heavily on human expertise, which, while invaluable, can introduce challenges such as inefficiency, inconsistency, and delays. With the rapid evolution of Artificial Intelligence (AI), organizations now have the opportunity to transform the way software is designed, developed, and managed.

An AI-Enhanced SDLC integrates advanced AI technologies—such as machine learning, natural language processing, and intelligent automation—into each phase of the lifecycle. This integration enables smarter decision-making, predictive insights, automated workflows, and higher quality outcomes. From requirement gathering and system design to testing, deployment, and maintenance, AI augments human capabilities by reducing manual effort, identifying risks early, and ensuring continuous optimization.

By embedding AI into the SDLC, businesses can achieve:

- **Accelerated development cycles** with automated coding, testing, and deployment.
- **Improved software quality** through predictive defect detection and intelligent monitoring.
- **Enhanced adaptability** to evolving user requirements and market conditions.
- **Optimized resource utilization** with data-driven project management.

As software becomes the backbone of digital transformation, adopting an AI-Enhanced SDLC not only improves efficiency but also ensures resilience and innovation in a competitive landscape.

Project Overview

The AI-Enhanced Software Development Lifecycle (SDLC) project aims to explore how Artificial Intelligence can be integrated into every stage of the software development process to improve efficiency, quality, and adaptability. Traditional SDLC models often face challenges such as unclear requirements, lengthy testing cycles, human error, and high maintenance costs. By embedding AI-driven tools and techniques into requirements gathering, design, coding, testing, deployment, and maintenance, this project demonstrates how intelligent automation can reduce risks, optimize resources, and deliver better software outcomes.

2.This project highlights the role of AI in:

Requirements Analysis: Using natural language processing (NLP) to interpret user needs and detect ambiguities.

Design & Development: Leveraging AI pair programming and automated architecture recommendations.

Testing: Employing predictive defect detection and intelligent test case generation.

Deployment & Maintenance: Utilizing anomaly detection, self-healing systems, and predictive maintenance.

Project Management: Supporting decision-making with AI-powered planning, workload optimization, and risk assessment.

3.Architecture of AI-Enhanced SDLC

The architecture of the AI-Enhanced Software Development Lifecycle integrates artificial intelligence components into the traditional stages of software engineering. It is designed as a layered and iterative framework where AI modules support automation, prediction, and optimization at every stage.

◆ 1. Input Layer

Stakeholder Data & Requirements: User stories, business goals, and feedback.

Historical Project Data: Past defects, test cases, and performance logs.

Development Environment Inputs: Code repositories, system logs, and project management data.

◆ 2. AI-Driven Core Components

1.Requirements & Analysis

- Natural Language Processing (NLP) for extracting requirements.
- AI-powered consistency and conflict detection.

2.Design & Development

- Generative AI for architecture and design suggestions.
- AI-based code assistants (pair programming, auto-completion).
- Automated code optimization engines.

3.Testing & Quality Assurance

- Predictive defect detection models.
- AI-driven test case generation and prioritization.
- Visual/UI testing with image recognition models.

4.Deployment & Maintenance

- AI-based anomaly detection in CI/CD pipelines.
- Self-healing mechanisms for error recovery.
- Predictive maintenance using machine learning models.

5.Project Management & Collaboration

- AI-powered resource allocation and risk assessment.
- Intelligent dashboards for progress tracking.
- Sentiment analysis for team performance insights.

◆ 3. Data Layer

- Knowledge Base: Stores patterns, reusable components, and best practices.
- Feedback Loop: Continuous learning from defects, user behavior, and system performance.
- Data Lakes & Repositories: Centralized storage for structured and unstructured project data.

◆ 4. Output Layer

- Optimized Software Deliverables: Higher quality, fewer defects, and faster delivery.
- Predictive Insights: Risk forecasts, performance trends, and workload predictions.
- Automation Reports: AI-generated documentation, testing coverage, and compliance reports.

1. Prerequisites

Before starting, ensure the following are installed:

Programming Languages:

Python (≥ 3.9) for AI/ML modules

JavaScript/TypeScript (Node.js ≥ 18) for frontend & backend services

AI/ML Libraries: TensorFlow / PyTorch, scikit-learn, spaCy, HuggingFace Transformers

DevOps Tools: Docker, Kubernetes (optional), Git, Jenkins or GitHub Actions

Database: PostgreSQL or MongoDB

Other Tools: VS Code, Postman, Jupyter Notebook

◆ **2. Clone the Repository**

git clone https://23ucs589-creator.github.io/AIlifecycle/

◆ **3. Environment Setup**

Create a virtual environment (Python for AI modules):

```
python -m venv venv
source venv/bin/activate      # Linux/Mac
venv\Scripts\activate         # Windows
```

Install dependencies:

```
docker-compose up --build
```

5.AI-Enhanced SDLC – Folder Structure

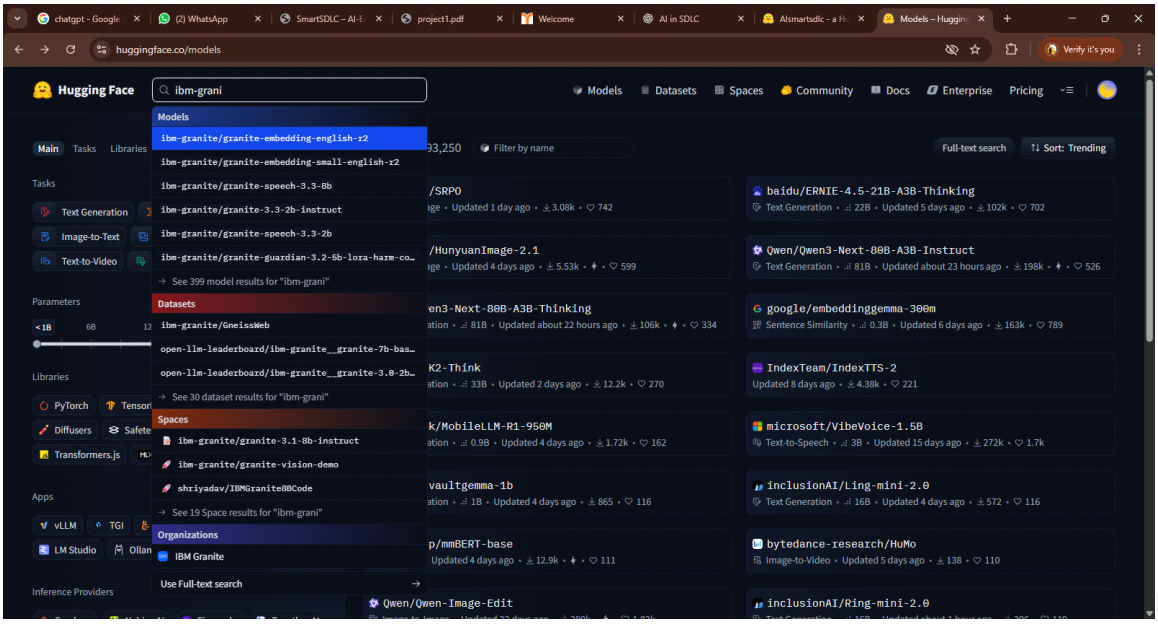
```
AI-Enhanced-SDLC/
|
├── 🗁 requirements/
|     ├── business_requirements.md
|     ├── functional_requirements.md
|     ├── nonfunctional_requirements.md
|     └── ai_requirements_extraction/      # NLP models, requirement parsing
|
├── 🗁 design/
|     ├── architecture_diagrams/
|     ├── uml_models/
|     ├── ai_design_recommendations/       # Generative AI design outputs
|     └── prototypes/
|
├── 🗁 development/
|     ├── src/                             # Source code
|     |     ├── backend/
|     |     ├── frontend/
|     |     └── ai_modules/                # AI assistants, ML models
|     ├── tests/                           # Unit & integration tests
```

```
|   ├── code_quality/                  # Static analysis, AI bug detection
|   └── documentation/
|
├── 🗁 testing/
|   ├── test_cases/                    # AI-generated and manual test cases
|   ├── regression_tests/
|   ├── performance_tests/
|   ├── ai_test_automation/            # ML models for defect prediction
|   └── reports/
|
├── 🗁 deployment/
|   ├── ci_cd_pipelines/               # AI-enhanced CI/CD scripts
|   ├── docker_configs/
|   ├── cloud_infrastructure/
|   └── monitoring/                    # AI anomaly detection, logs
|
├── 🗁 maintenance/
|   ├── issue_tracking/                # AI triage & bug reports
|   ├── logs_analysis/                 # AI-based log monitoring
|   ├── predictive_maintenance/        # ML models for failure prediction
|   └── updates/
|
├── 🗁 project_management/
```

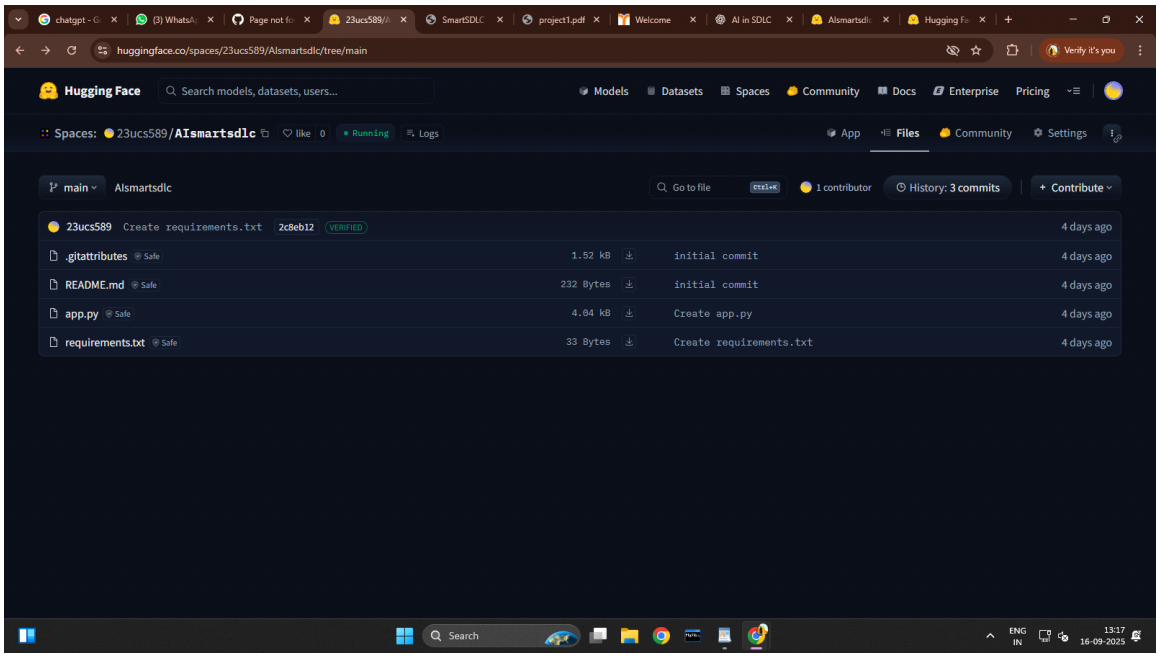
```
|   ├── schedules/
|   ├── workload_distribution/
|   ├── risk_analysis/                    # AI-powered risk predictions
|   └── dashboards/
|
└── README.md                             # Project overview & guidelines
```

6.Screenshots

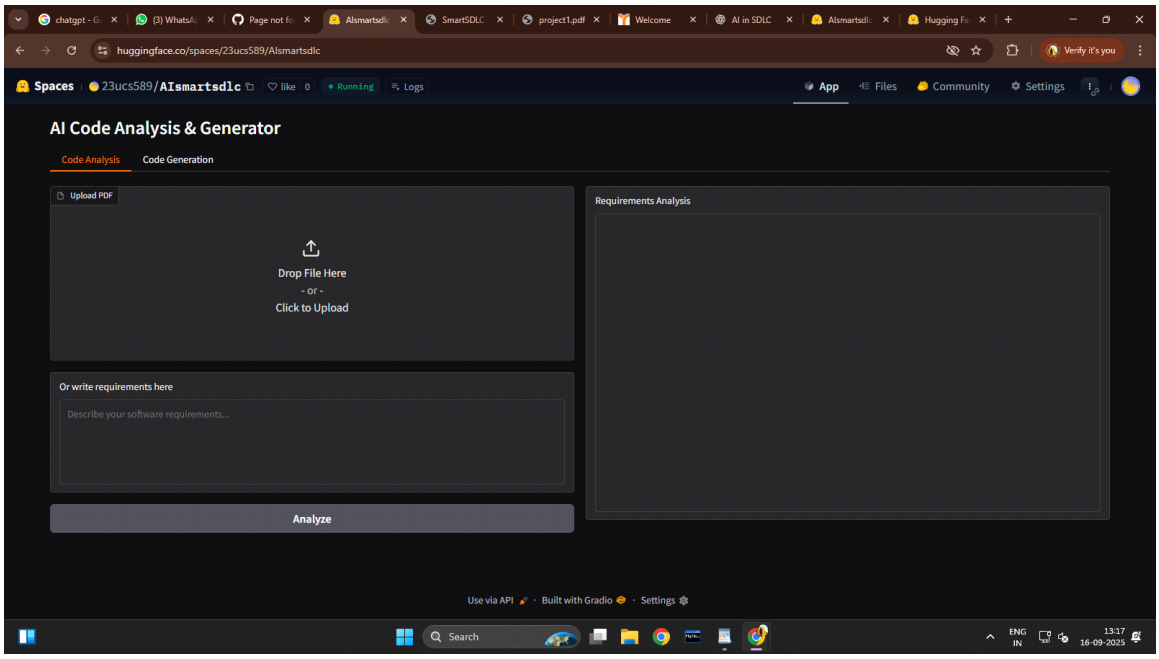
Selecting model:



Uploading code for app.py ,recquirement.txt file to new space:



Testing code in hugging face and the result will be displayed:



7.uture Enhancements in AI-Enhanced SDLC

While the current AI-Enhanced SDLC model introduces automation, predictive analytics, and intelligent decision-making across the lifecycle, there are several opportunities for further improvement and innovation:

1. Advanced Requirement Engineering

Integration of conversational AI agents to capture requirements directly from stakeholders in natural language.

Real-time requirement validation using AI to align evolving user needs with business goals.

2. Generative AI in Design & Development

Use of large language models (LLMs) for full-feature code generation beyond boilerplate.