SmartSDLC – AI-Enhanced Software Development Lifecycle

Team Leader: YASMIN BEEVI S
Team Members: SHREEMATHI G, SHPINO J, SRIVITHYA A, SHAHANA S

1. Introduction

• Project title: SmartSDLC – AI-Enhanced Software Development Lifecycle
• Team Leader: YASMIN BEEVI S
• Team Members: SHREEMATHI G, SHPINO J, SRIVITHYA A, SHAHANA S

2. Project Overview

Purpose:
The purpose of SmartSDLC is to optimize the software development lifecycle (SDLC) using AI-driven enhancements. This system integrates intelligent tools for requirement gathering, code generation, testing automation, and deployment monitoring. By leveraging machine learning models, it ensures faster, more efficient, and error-minimized development cycles.

Features:
• AI-Powered Requirement Analysis – Automatically analyzes and refines project requirements.
• Smart Code Suggestions – Provides intelligent code recommendations and bug detection.
• Automated Testing – Generates and executes test cases with high accuracy.
• Continuous Monitoring – Tracks deployment health and performance issues.
• Predictive Analytics – Forecasts potential delays and risks.
• Knowledge Summarization – Summarizes project documents for quick understanding.

3. Architecture

Frontend: Built using ReactJS with TailwindCSS for UI, providing dashboards, project timelines, and analytics.
Backend: Powered by FastAPI for scalable APIs handling SDLC processes.
AI Integration: Uses GPT-based LLM models for requirement refinement, code suggestions, and summarization.
Database: MongoDB for storing project data, user stories, and test cases.
Monitoring Tools: Integrated logging and anomaly detection for deployment monitoring.

4. Setup Instructions

Prerequisites:
• Python 3.9+
• Node.js and npm
• MongoDB
• API keys for AI models

Installation Process:
• Clone the repository
• Install backend dependencies using pip
• Install frontend dependencies using npm

• Configure environment variables (.env file)
• Run FastAPI backend
• Start React frontend

5. Folder Structure

• backend/ – FastAPI APIs for requirements, testing, and monitoring
• frontend/ – ReactJS UI components and dashboards
• ai_modules/ – AI models for code suggestion, testing, and summarization
• database/ – MongoDB schemas and utilities
• utils/ – Helper functions and scripts

6. Running the Application

• Start backend server with FastAPI
• Start frontend React dashboard
• Access via browser to manage SDLC process
• Upload requirements, generate test cases, monitor deployments

7. API Documentation

• POST /requirements/analyze – AI-powered requirement analysis
• POST /code/suggest – Suggests optimized code
• POST /testing/auto – Auto-generates test cases
• GET /monitor/status – Retrieves deployment monitoring data
• POST /summarize/docs – Summarizes project documentation

8. Authentication

• JWT-based authentication
• Role-based access: Developer, Tester, Manager
• Secure API key management

9. User Interface

• Dashboard with project progress visualization
• Tabs for requirements, code, testing, and monitoring
• Real-time notifications for risks and delays
• Downloadable reports in PDF

10. Testing

• Unit Testing – For AI modules and backend APIs
• Integration Testing – For combined workflow validation
• Manual Testing – For UI and user experience
• Edge Case Handling – Invalid inputs, large data

11. Screenshots

To be added after UI completion.

12. Known Issues

Minor UI glitches during large-scale data uploads.

13. Future Enhancements

Integration with DevOps pipelines, advanced predictive analytics, and AI-driven project management assistance.