Final Report for SmartSDLC

1. INTRODUCTION

1.1 Project Overview

SmartSDLC is an AI-powered platform designed to automate the Software Development Lifecycle (SDLC) using NLP and Watsonx. It enables the conversion of unstructured requirements into user stories, generates code, fixes bugs, writes test cases, summarizes code, and offers chatbot support, improving development speed, accuracy, and collaboration.

1.2 Purpose

The purpose of SmartSDLC is to automate repetitive and manual tasks in the SDLC, reducing time and effort for developers, improving collaboration among teams, and accelerating the software development process.

2. IDEATION PHASE

2.1 Problem Statement

The SDLC involves many manual tasks, such as requirement analysis, coding, bug fixing, and test case generation, which can lead to inefficiencies and errors. SmartSDLC automates these tasks using AI to improve accuracy, speed, and collaboration across all SDLC phases.

2.2 Empathy Map Canvas

Developer (Primary User): Wants to focus on coding but is bogged down by repetitive tasks like debugging and writing test cases. Feels frustrated by inefficiencies.
Project Manager: Needs faster project delivery and seamless collaboration but faces delays due to manual SDLC tasks. Feels stressed and concerned about team productivity.

2.3 Brainstorming

Ideas were brainstormed to automate various aspects of SDLC:
1. NLP for requirement analysis
2. AI for code generation and bug fixing
3. Automated test case generation
4. Real-time chatbot support

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

The user journey maps how developers and project managers interact with the SmartSDLC platform, from receiving unstructured requirements to seeing automated code and interacting with the chatbot.

3.2 Solution Requirement

Functional Requirements:
- Automate requirement analysis, code generation, bug fixing, and test case creation.
Non-functional Requirements:
- Ensure the platform is secure, reliable, scalable, and user-friendly.

3.3 Data Flow Diagram

A DFD illustrates the flow of data through the SmartSDLC system. It shows how data enters (unstructured requirements) and flows through various stages like code generation, bug fixing, and test case creation.

3.4 Technology Stack

Frontend: React.js, HTML, CSS
Backend: Python, IBM Watson NLP & AI
Database: MySQL or NoSQL
Cloud: IBM Cloud, AWS, Kubernetes
External APIs: OAuth for authentication (Google, LinkedIn)

4. PROJECT DESIGN

4.1 Problem Solution Fit

The solution addresses inefficiencies in the SDLC by automating key processes, from requirement gathering to code generation, testing, and bug fixing. This solution aligns with the need for faster, more accurate software development.

4.2 Proposed Solution

SmartSDLC uses AI and NLP to automate SDLC processes, improving productivity and reducing human error. It integrates AI-driven code generation, bug fixing, and testing, all while providing real-time chatbot assistance.

4.3 Solution Architecture

The platform uses a cloud-based architecture, with data flowing seamlessly between various components, including the user interface, backend logic, AI models, and storage. The system is scalable and can handle increasing numbers of users and tasks.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

The project is divided into two main sprints:
- Sprint 1: Requirement analysis, code generation, bug fixing, and test case generation.
- Sprint 2: Code summary generation, chatbot integration, and deployment.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Test the speed and accuracy of code generation, bug fixing, and test case generation.
Measure response time under load (multiple API calls).

7. RESULTS

7.1 Output Screenshots

Screenshots demonstrating the SmartSDLC platform in action, including the process of requirement conversion, code generation, bug fixing, and chatbot interactions.

8. ADVANTAGES & DISADVANTAGES

Advantages

Increased development speed
Reduced manual effort
Improved collaboration across teams

Disadvantages

Initial setup and learning curve
Dependency on AI accuracy

9. CONCLUSION

SmartSDLC revolutionizes the SDLC by automating tedious tasks and allowing developers and project managers to focus on higher-value activities. It improves overall productivity and collaboration.

10. FUTURE SCOPE

The future scope of SmartSDLC includes expanding its capabilities to cover more SDLC phases, such as deployment automation, advanced debugging techniques, and machine learning for predictive task allocation.

11. APPENDIX

Source Code

[GitHub Link]

Dataset Link

[Link to Datasets]

GitHub & Project Demo Link

[Demo Link]