Project Design Phase
Solution Architecture

Date	27 June 2025
Team ID	LTVIP2025TMID33968
Project Name	SmartSDLC-AI-Enhanced Software Development Lifecycle
Maximum Marks	4 Marks

Solution Architecture:

Solution Architecture – Smart SDLC with IBM Watsonx GenAI

1. Overview

The Smart SDLC solution architecture integrates IBM Watsonx GenAI with popular developer tools to automate key phases of the Software Development Life Cycle (SDLC). It streamlines documentation, sprint planning, and test case generation while providing real-time updates to project tracking system

2. Key Components

- **Developer**
  - **The end user who interacts with GitHub, JIRA, and the Web App.**
  - **Triggers actions such as generating documentation, creating tasks, or initiating AI-based analysis.**

- **Dev Tools**
  - **Integration with GitHub, JIRA, and Slack for real-time code, issue, and communication tracking.**
  - **These tools serve as inputs and outputs for the AI workflows.**

- **Web Application (Core Layer)**

Handles the following operations:

- **Documentation Generator: Auto-generates PR summaries, README files, and inline docs.**
- **Sprint Planner: Estimates task duration and workload distribution using past sprint data.**

- **Test Case Generator: Converts user stories or feature descriptions into test cases.**
- **Automation Engine: Sends updates to tracking tools or triggers alerts on Slack/JIRA.**

🔹 **IBM Watsonx GenAI (AI Layer)**

- **Processes developer inputs and queries via natural language.**
- **Uses prompt templates to generate content (e.g., docs, test cases).**
- **Sends AI-generated outputs back to the Web App.**

🔹 **Project Management Module**

- **Receives AI-generated content and updates sprint dashboards.**
- **Tracks issues, bug reports, team velocity, and delivery health metrics.**

🔹 **Database**

- **Stores project metadata, user prompts, test results, sprint history, and generated artifacts.**
- **Ensures traceability and historical learning for future recommendations.**

3. Workflow Summary

1. **Developer triggers an action via GitHub/JIRA or the Web App.**
2. **Input is sent to Watsonx GenAI using structured prompts.**
3. **AI processes the request and returns formatted content.**
4. **Content is displayed or pushed to project tracking tools.**
5. **All outputs and interactions are logged in the backend database.**

4. Technology Stack

- **Frontend: React.js (or HTML/CSS/JS)**
- **Backend: Node.js with Express**

- **AI Layer: IBM Watsonx GenAI API**
- **Integration APIs: GitHub, JIRA, Slack Webhooks**
- **Database: MongoDB or Firebase**

5. Security & Access

- **API access controlled using tokens and role-based permissions.**
- **Sensitive content (e.g., code) is never stored in raw form; only summaries or references are logged.**

- Solution Architecture Diagram: