**Solution Architecture – SmartSDLC**

**Date**: 25 JUNE 2025  
**Team ID**: LTVIP2025TMID31783  
**Project Name**: SmartSDLC  
**Maximum Marks**: 4

**Overview**

The solution architecture for **SmartSDLC** defines how AI is used to classify documentation into SDLC phases. The goal is to support teams, mentors, and students in understanding the software development process by mapping real project documents to specific SDLC stages.

**Goals of the Architecture**

* Automate SDLC phase detection in project documentation using Generative AI.
* Visualize document structure phase-wise for analytics and mentoring.
* Enable session-based management of data (text, classifications, feedback).
* Support real-time document feedback and analytics.
* Ensure secure API access and scalable architecture for future enhancements.

**Architectural Components**

**Frontend (UI)**

* Built using **Streamlit** with tabbed layout:
  + **Document Upload**
  + **Phase Classification**
  + **Phase-wise Dashboard**
  + **Feedback & Correction**
* Sidebar for metadata (Project Name, User Info).

**Application Logic (Python Backend)**

* **Text Extraction**: fitz (PyMuPDF) or python-docx to convert files into text.
* **Text Chunking & Classification**:
  + Segment document into paragraphs/sections.
  + Use prompt-based classification with LLM (currently Gemini; IBM Granite planned).
* **Visualization & Stats**: Use Pandas to group and aggregate classification data.

**AI Models**

* **Current Simulation**: Google Gemini (via API key) for zero-shot classification.
* **Target Integration**: IBM Granite-3.3-2B-instruct (via IBM Watson Machine Learning).

**Data Flow & Management**

* **User Inputs**: PDF upload, manual feedback, project name.
* **Session State**: Holds file text, classifications, and feedback using st.session\_state.
* **Document Processing**:
  1. Uploaded file → Text chunks
  2. Each chunk → Prompt → Gemini → Predicted SDLC Phase
* **Feedback Handling**: Stores updated phase selected by the user.
* **Logging Layer (Future)**: Save feedback and AI output as CSV or JSON.

**Analytics Engine**

* Built using **Plotly** to generate:
  + Bar chart: Distribution across SDLC phases
  + Word cloud: Top terms per phase
  + Table view: Phase-wise classified segments

**Security Layer**

* API tokens managed securely using .env and python-dotenv.
* No storage of uploaded files beyond session.
* (Future) Role-based access and user authentication planned.

**Output Services**

* **Text Display**: AI-classified chunks shown with tags.
* **Dashboard**: Charts and summaries dynamically update.
* **Correction Interface**: Dropdown per segment to relabel predicted phase.

**Data Flow Summary**

**User Interaction**:

* Uploads a file or pastes content → Sees extracted segments
* AI classifies → Phase tags shown → User gives feedback if needed
* Dashboard shows summary statistics and visualizations

**Model Processing**:

* Prompts constructed with each text chunk → AI API (Gemini now, Granite later)
* AI output → Phase label → UI + Dashboard update

**Session Management**:

* All variables (chunks, labels, corrections) stored in st.session\_state during runtime.

**Development Phases**

| **Phase** | **Description** |
| --- | --- |
| **Phase 1: UI & File Handling** | Upload interface, sidebar fields, preview viewer |
| **Phase 2: AI Integration** | Gemini-based prompt classification and response parsing |
| **Phase 3: Dashboard & Visualization** | Plotly bar/pie charts, word clouds, stats |
| **Phase 4: Feedback Mechanism** | Relabel interface with editable dropdowns |
| **Phase 5: IBM Granite Integration (Future)** | Replace Gemini with IBM Granite via Watson ML SDK |
| **Phase 6: Persistent Storage (Future)** | Integrate Firestore or SQLite to store past runs and feedback logs |

**Architecture Diagram (Described)**

**Current Flow:**

User → Streamlit UI → Python Backend → Gemini API → SDLC Phase Labels → Dashboard

**Future Flow:**

User → Streamlit UI → Python Backend → IBM Granite via IBM Watson ML API → Stored Results + Enhanced Analytics

**Conclusion**

The SmartSDLC architecture provides an intelligent, user-friendly framework to analyze software documentation. From AI-powered SDLC classification to real-time analytics and feedback, it offers a streamlined solution with a strong roadmap for future integration of IBM Watson, persistent storage, and enterprise scaling.