**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

| Date | 25 JUNE 2025 |
| --- | --- |
| Team ID | LTVIP2025TMID21159 |
| Project Name | SmartSDLC – AI-Powered Software Development Lifecycle Optimization |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

**Example: Order processing during pandemics for offline mode**

**Reference:** [**https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/**](https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/)

****

****

**Table-1 : Components & Technologies:**

| **S.No** | **Component** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | User Interface | |  | | --- | | Web UI for interacting with all features |   . | |  | | --- | | Streamlit, streamlit-lottie | |
|  | Application Logic-1 | Code generation from natural language requirements | Python, transformers |
|  | Application Logic-2 | Bug fixing and test case generation logic | Python, Hugging Face Models |
|  | Application Logic-3 | Chatbot for SDLC-related queries | Python, LLM prompting |
|  | Database | Runtime session handling (temporary) | In-memory or file-based via Streamlit |
|  | Cloud Database | NA (no persistent cloud DB used currently) | NA |
|  | File Storage | PDF uploads for requirement analysis | Local filesystem / Streamlit uploader |
|  | External API-1 | Fallback API for model inference | Groq API |
|  | External API-2 | Authentication for model endpoints | Hugging Face API |
|  | Machine Learning Model | Model used for classification, generation, summarization | IBM Granite 3.3, transformers |
|  | Infrastructure (Server / Cloud) | Deployment on local/server with GPU support and fallback | FastAPI, Uvicorn, asyncio, torch.cuda |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Frameworks and libraries used | Streamlit, FastAPI, Transformers, PyMuPDF |
|  | Security Implementations | Token management via .env; external access restricted via backend | dotenv, API key encryption |
|  | Scalable Architecture | Modular microservices with fallback support and async endpoints | FastAPI, asyncio, Groq fallback |
|  | Availability | High availability ensured via fallback APIs and model retry logic | Groq API, retry mechanisms |
|  | Performance | |  | | --- | | GPU detection, concurrent processing, and async requests |  |  | | --- | |  | | torch.cuda.is\_available, asyncio.create\_task() |

### Architecture Summary

* **Frontend:** Streamlit
* **Backend:** FastAPI
* **Core AI Engine:** Hugging Face LLMs (e.g., IBM Granite 3.3)
* **Fallback Engine:** Groq API
* **Deployment:** Local GPU server, optionally cloud deployable
* **File Handling:** Local file system (for uploaded PDFs)

**References:**

[**https://c4model.com/**](https://c4model.com/)

[**https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/**](https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/)

[**https://www.ibm.com/cloud/architecture**](https://www.ibm.com/cloud/architecture)

[**https://aws.amazon.com/architecture**](https://aws.amazon.com/architecture)

[**https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d**](https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d)