**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 27 June 2025 |
| Team ID | LTVIP2025TMID34066 |
| Project Name | SmartSDLC – AI-Enhanced Software Development Lifecycle |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

A Smart SDLC Developer Assistant uses a modular architecture powered by IBM's Granite LLM to support software development tasks. Inputs like code, requirements, or bugs are submitted through a Streamlit interface. These are processed by FastAPI-based backend services. Each core feature—bug fixing, code generation, test creation, and doubt-solving is handled by a separate function module. The system uses a lightweight API structure for fast responses. It’s hosted on scalable cloud infrastructure. The architecture ensures simple maintenance and quick updates. Security layers ensure safe input handling and output validation. The assistant helps developers save time, reduce errors, and learn faster. It supports real-time, AI-driven development guidance.

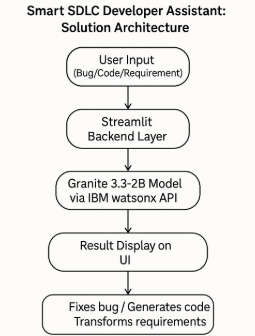
Guidelines:

 Be developer-first: Design an intuitive, clean interface to assist both beginners and experienced developers.

 Support learning: Provide clear explanations, suggestions, and coding help to boost developer skills.

 Ensure privacy: Handle code and inputs securely, maintaining user data confidentiality.

 Stay modular: Use scalable, function-based backend logic for easy updates and feature expansion.

****

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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | Simple web-based or Streamlit app UI for developers to interact with tools. | Python + Streamlit, HTML, CSS |
|  | IoT Sensors | Accepts developer queries, code snippets, or requirement descriptions | Python functions, Streamlit forms |
|  | Data Infrastructure | Logic for a process in the application To build a SmartSDLC – AI-Enhanced Software Development Lifecycle, you need a blend of key components and enabling technologies that work together to support eco-friendly urban living | IBM Watson STT service |
|  | AI Engine | For forecasting, anomaly detection, and natural language understanding. | IBM Watson Assistant |
|  | APIs & Integration Layer | To connect with city systems and third-party services. | MySQL, NoSQL, etc. |
|  | Cloud Database | Database Service on Cloud | IBM DB2, IBM Cloudant etc. |
|  | File Storage | **Cloud storage** for documents, logs, and uploads. **Edge/local storage** for quick, temporary access near IoT devices. **Databases** for structured data like user info and sensor logs.**Vector databases** for storing AI-friendly data like document embeddings. | Services like AWS S3, Azure Blob Storage, and Google Cloud Storage are ideal for storing unstructured data like PDFs, images, and logs. They offer scalability, durability, and easy integration with AI pipelines. |
|  | External API-1 | **IBM Watsonx Granite LLM**: Powers natural language understanding and summarization. **FastAPI**: Handles backend logic and exposes RESTful endpoints. | 5G/6G Networks: Enable ultra-fast, low-latency communication between IoT devices and cloud systems. . |
|  | External API-2 | **Pinecone**: Stores and retrieves AI embeddings for semantic search.**Streamlit**: Provides the interactive user interface. | **Internet of Things (IoT)**: External sensors and devices that monitor air quality, traffic, energy, and more.**Artificial Intelligence (AI)**: External LLMs like IBM Watsonx Granite for summarization, forecasting, and citizen interaction. |
|  | Machine Learning Model | **Core Purpose**: Help cities become greener, smarter, and more citizen-friendly using AI and real-time data.**Key Components**: IoT sensors, cloud databases, AI models (like IBM Watsonx Granite), user interfaces, and secure APIs.**Technologies Used**: FastAPI, Streamlit, Pinecone, cloud storage (AWS S3, GCP), vector databases, and NLP tools.**Features**: Forecasting, anomaly detection, policy summarization, eco-tip generation, and citizen feedback.**Guidelines**: Focus on inclusivity, sustainability, data ethics, modular design, and real-time responsiveness.**Machine Learning Models**: Regression, classification, clustering, time series (LSTM), and reinforcement learning. | **Big Data Analytics**: Tools like Apache Spark or Google BigQuery to analyze massive urban datasets. |
|  | Infrastructure (Server / Cloud) | Application Deployment on Local System / Cloud  Local Server Configuration:**Edge Servers** Deployed near IoT devices for low-latency processing .  Cloud Server Configuration : AWS, Azure, or Google Cloud provide scalable compute, storage, and AI services. |  |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Hugging Face Transformers For integrating AI models like IBM Watsonx Granite | Hugging Face Transformers |
|  | Security Implementations | Role-Based Access Control (RBAC) Ensures only authorized users access sensitive data. | Role-Based Access Control |
|  | Scalable Architecture | Edge + Cloud Hybrid Real-time processing at the edge, heavy lifting in the cloud | Edge + Cloud Hybrid |
|  | Availability | Redundancy Backup systems for critical services like citizen feedback and anomaly alerts. | Redundancy |
|  | Performance | Optimized AI Models Use quantized or distilled models for faster inference. | Redundancy |