**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

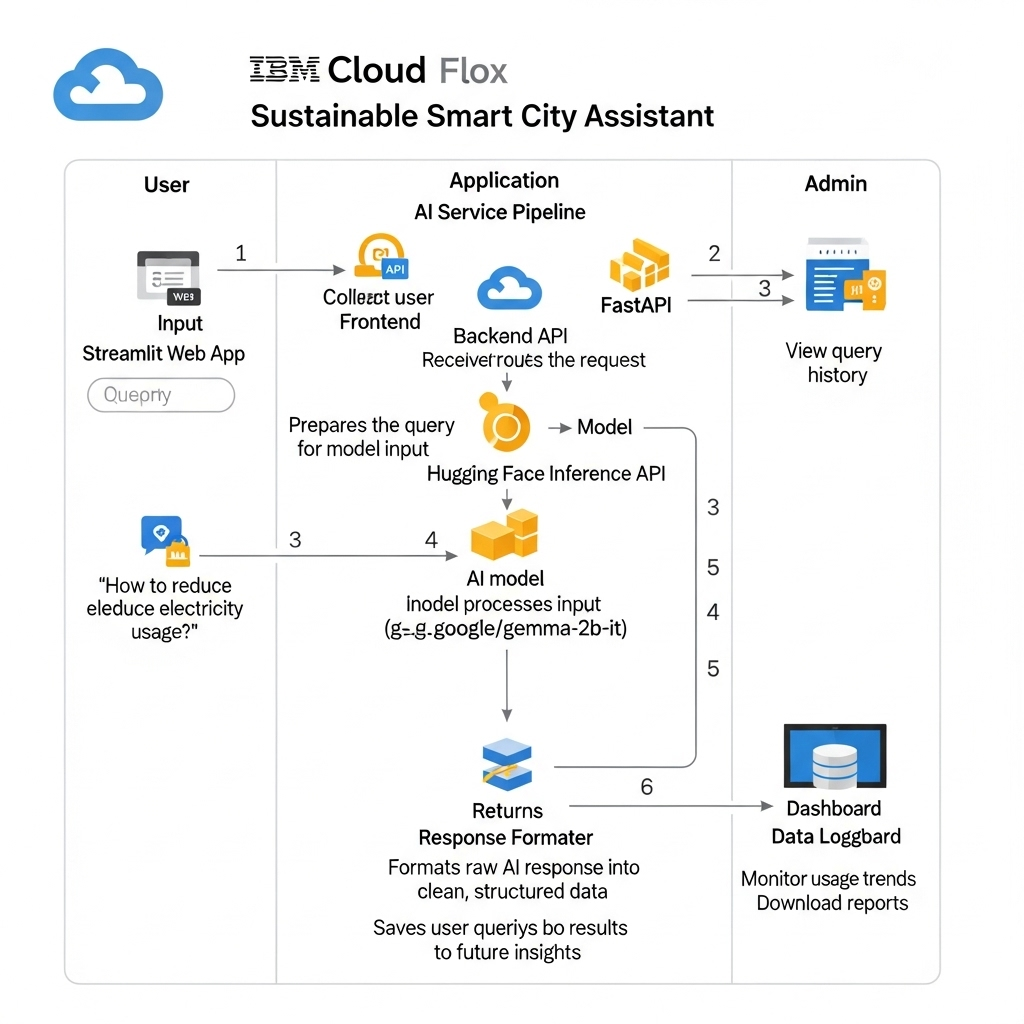
|  |  |
| --- | --- |
| Date | 24 june 3035 |
| Team ID | LTVIP2025TMID32074 |
| Project Name | Sustainable smart city using IBM Granite |
| Maximum Marks | 4 Marks |

**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-based dashboard for citizen interaction | | Streamlit (Python-based web UI) |
|  | Application Logic-1 | |  | | --- | |  |  |  | | --- | | Handles user registration, login, and dashboard routing | | Python + FastAPI |
|  | Application Logic-2 | Manages AI requests and routing | FastAPI (API framework) |
|  | Application Logic-3 | |  | | --- | |  |  |  | | --- | | Formats and handles model response | | Python backend (response formatter) |
|  | Database | |  | | --- | |  |  |  | | --- | | Store logs, feedback, user queries | | SQLite or MongoDB (lightweight NoSQL) |
|  | Cloud Database | |  | | --- | |  |  |  | | --- | | Optional if hosted on cloud, for scalability | | Firebase or MongoDB Atlas |
|  | File Storage | |  | | --- | |  |  |  | | --- | | Store uploaded reports or result summaries | | Local Filesystem / Google Drive API |
|  | External API-1 | |  | | --- | |  |  |  | | --- | | Access smart suggestions via LLM | | Hugging Face Inference API |
|  | External API-2 | |  | | --- | |  |  |  | | --- | | Future use for location/weather integration | | OpenWeatherMap API |
|  | Machine Learning Model | Understands and responds to smart city queries | |  | | --- | |  |  |  | | --- | | ibm-granite/granite-3.3-8b-instruct from Hugging Face | |
|  | Infrastructure (Server / Cloud) | |  | | --- | |  |  |  | | --- | | Application hosted on cloud/local for development | | Vs code + Localhost / Streamlit Cloud |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | FastAPI, Streamlit, Hugging Face Transformers | Python Ecosystem |
|  | Security Implementations | |  | | --- | |  |  |  | | --- | | Authentication, API key protection, HTTPS, OAuth (planned) | | JWT / HTTPS / Streamlit Auth (planned) |
|  | Scalable Architecture | Modular FastAPI + Independent Streamlit UI with scalable APIs | Microservices-ready structure |
|  | Availability | Hosted on distributed dev platforms | Load-balancing ready, modular endpoints |
|  | Performance | Fast inference via Hugging Face; caching and API reuse | Hugging Face API, Uvicorn, Python cache |

**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)