Project Design Phase-II

Technology Stack (Architecture & Stack)

Date	31 January 3035
Team ID	LTVIP2025TMID60699
Project Name	Sustainable Smart City Assistant Using IBM Granite LLM
Maximum Marks	4 Marks

Technical Architecture – Sustainable Smart City Assistant

This section outlines the technical structure of our Smart City Assistant. The system is built using modular architecture with distinct layers for user interaction, logic processing, AI-based services, and data retrieval from external APIs. The backend integrates IBM-deployed models, and the frontend is powered by Streamlit for lightweight, reactive UI rendering.

System Flow Overview

1. **User interacts with the assistant through the dashboard, chat interface, and alerts.**
2. **The frontend is built using Streamlit and JavaScript-based components.**
3. **User actions trigger backend processes — data fetching, summarization, forecasting, anomaly detection, and tips generation.**
4. **The system also calls APIs (e.g., AQI, weather) and uses a deployed IBM model for advanced ML tasks like anomaly classification and eco-behavior modeling.**

5. **The processed output is shown via dashboards, chat replies, smart alerts, and summaries.**

Technology Stack (Architecture & Stack)

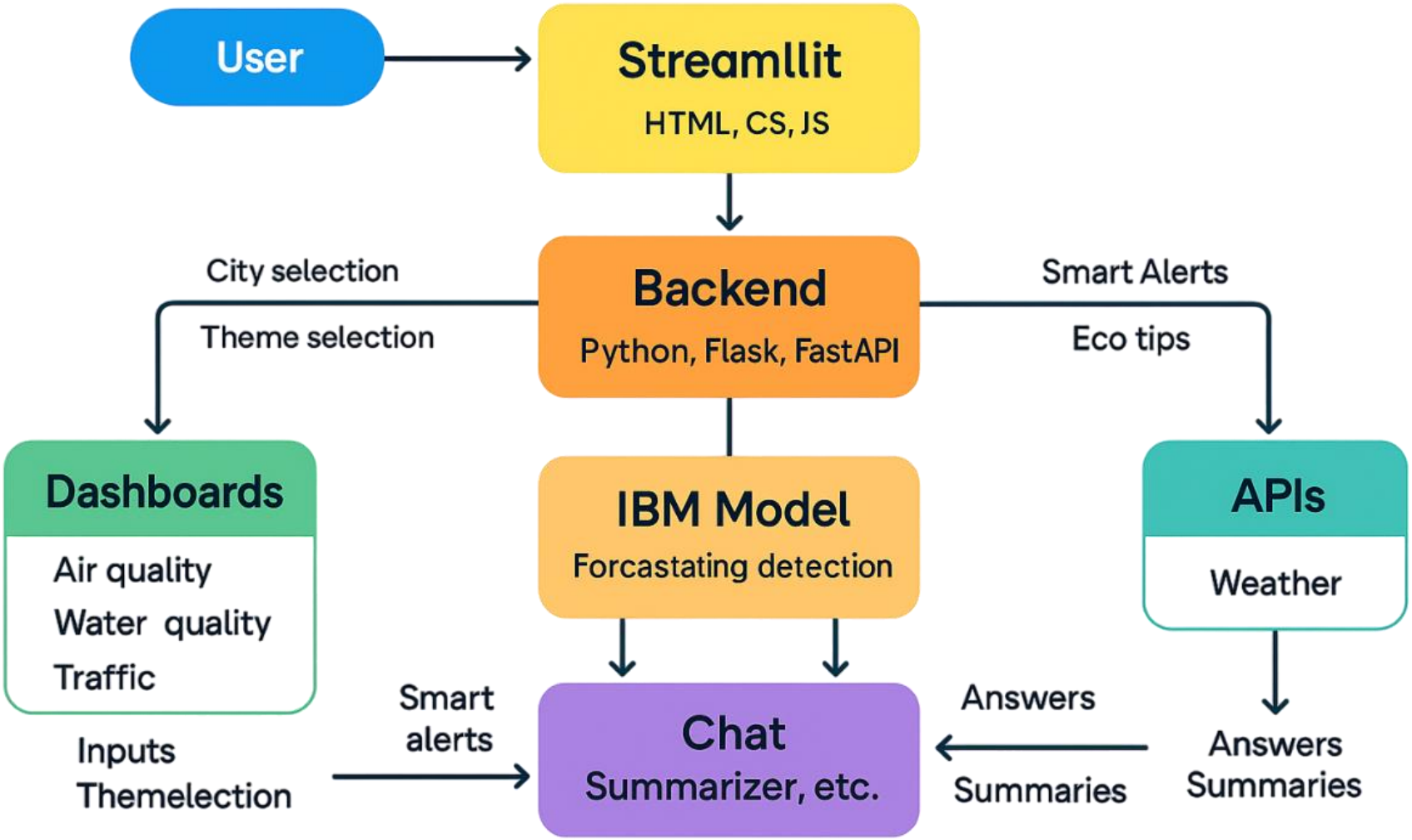

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Web-based dashboard, chat interface, alert widgets	Streamlit, HTML, CSS, JS
2	Application Logic-1	Core logic for data visualization and dashboard updates	Python (Pandas, NumPy, Plotly)
3	Application Logic-2	ML logic for forecasting and anomaly detection	IBM Cloud ML Model (deployed model endpoint)
4	Application Logic-3	Chat logic and summarization service	Python, OpenAI GPT model (or fallback rules)
5	Database	Temporary session storage (if needed)	Local JSON storage or lightweight SQLite
6	Cloud Database	For future expansion	Firebase / IBM Cloudant (optional)
7	File Storage	Eco tips, summaries, city data configs	JSON Files on local file system
8	External API-1	Real-time AQI and environmental data	AQI India API, OpenWeatherMap
9	External API-2	Weather & city coordinates	OpenWeather API / LocationIQ (optional)
10	Machine Learning Model	Forecast AQI/water/traffic + detect anomalies	IBM Deployed Time-Series ML Model
11	Infrastructure	Hosted locally for demo; cloud-ready	Localhost / Firebase Hosting / IBM Cloud

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology Used
1	Open-Source Frameworks	Tools used for rapid development and data display	Streamlit, Python, Flask, Scikit-learn
2	Security Implementations	API keys securely handled; no personal data stored; HTTPS enforced	Environment variable handling, CORS setup
3	Scalable Architecture	Modular layout: UI, logic, ML, API — easily extendable to more cities & users	Microservices-ready, Serverless architecture
4	Availability	App accessible 24x7 if hosted online, with fallback for offline city data/tips	Firebase Hosting / Streamlit Cloud
5	Performance	Pre-cached metrics; asynchronous API calls; minimal data load per request	Python caching, low-latency API use