**Project Design Phase-II**

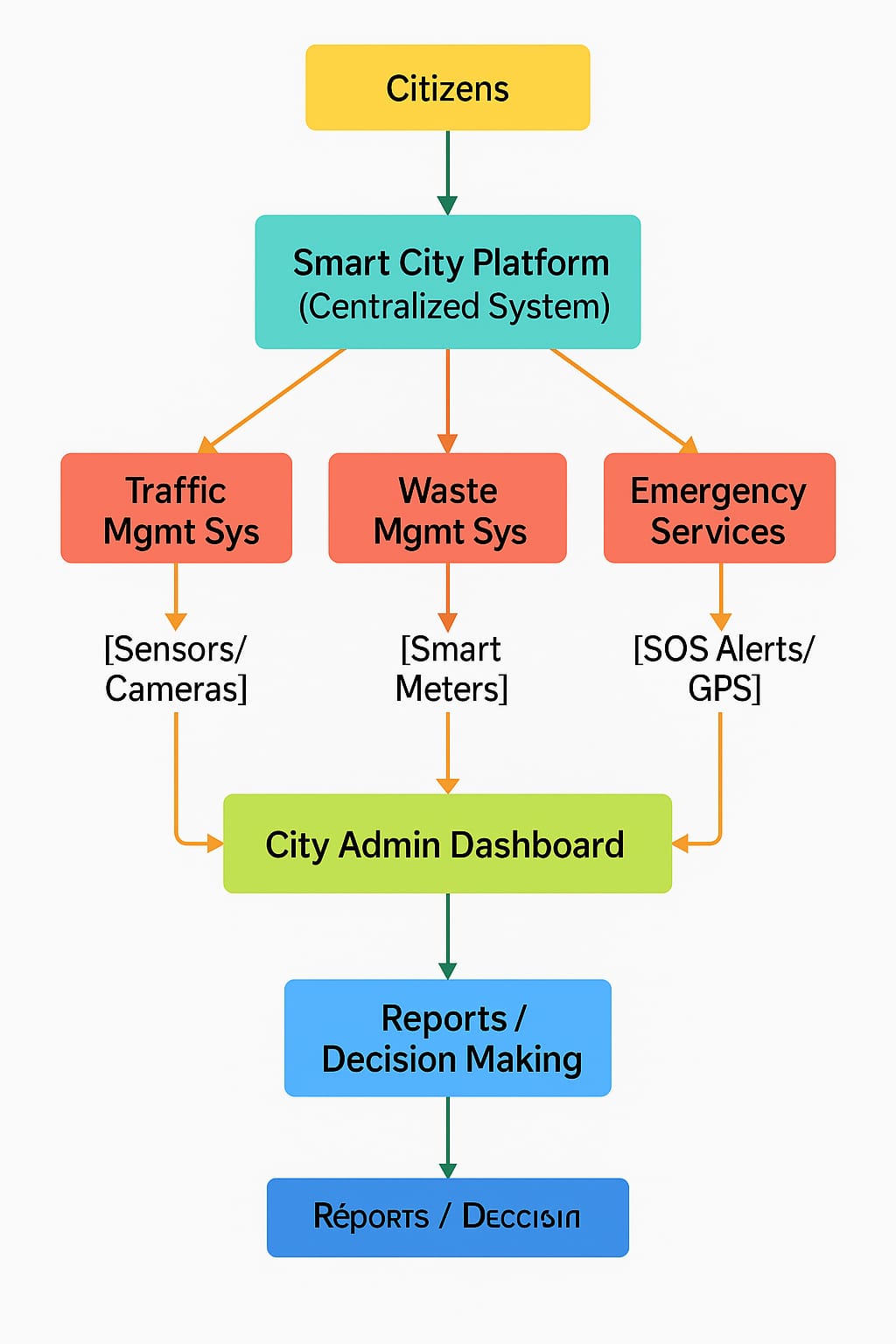
**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 26-06 2025 |
| Team ID | LTVIP2025TMID37089 |
| Project Name | Sustainable Smart City Assistant using IBM Granite LLM |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

This project utilizes **LLM-based intelligence** (IBM Granite 3.3-2b-instruct) to build a **smart city assistant** accessible via a web UI. It allows users to enter eco-queries or civic complaints in plain language, which are interpreted and responded to by the model

**Example: Real-time Civic Complaint Handling**

****

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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | Web UI interface allowing users to enter queries and complaints via text UI framework) | Gradio (Python-based UI framework) |
|  | Application Logic-1 | Handles prompt formatting, login logic, and validation | Python |
|  | Application Logic-2 | Keyword-based complaint classification and department routing | Python |
|  | Application Logic-3 | LLM-based processing of civic and eco queries | IBM Granite 3.3-2b-instruct (Hugging Face) |
|  | Database | |  | | --- | |  |  |  | | --- | | No permanent storage used, but structured prompt-response handling via in-memory Python logic | | JSON / Python Dictionaries (Temporary) |
|  | Cloud Database | Can be integrated with IBM Cloudant or Firebase for logging complaints (future enhancement) | IBM Cloudant (optional/future) |
|  | File Storage | |  | | --- | |  |  |  | | --- | | Local script and temporary memory used; PDF/image input planned in future | | Local Filesystem (currently) |
|  | External API-1 | Future plan: Integrate IBM Weather API to handle environment-based queries | IBM Weather API (future scope) |
|  | External API-2 | |  | | --- | | 2 |  |  | | --- | | Optional: Integration with Aadhaar API for identity  validation during complaint submission | | Aadhar API (optional/future) |
|  | Machine Learning Model | Uses a pre-trained LLM to understand and respond to user inputs | IBM Granite LLM (via Hugging Face) |
|  | Infrastructure (Server / Cloud) | Deployed via Google Colab / Hugging Face Spaces / Local (VS Code) | Cloud: Hugging Face / Colab / Local |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Uses open-source libraries and platforms for development and deployment | Python, Gradio, PyTorch, Hugging Face Transformers |
|  | Security Implementations | Basic login-based access control; in future: OAuth or JWT token-based authentication | SHA-256 (planned), Secure Login in Gradio |
|  | Scalable Architecture | Modular script (main.py) supports adding new modes/APIs; scalable to microservices | 2-Tier (UI + Model Logic); can evolve to 3-Tier |
|  | Availability | Hosted on cloud platforms like Hugging Face/Colab for 24x7 access | Hugging Face Spaces / Google Colab |
|  | Performance | Handles real-time responses in <3s; lightweight Gradio UI ensures high availability | Gradio + IBM Granite LLM (Optimized with PyTorch) |