# **Project Documentation**

## **1. Introduction**

**Project Title:** Smart City Assistant Using IBM Granite

**Team Members:**

* N. Reshma Banu
* H. Ramzan
* G. Rasika
* P. Rathi

## **2. Project Overview**

**Purpose:** The purpose of a Sustainable Smart City Assistant is to empower cities and their residents to thrive in a more eco-conscious and connected urban environment. By leveraging AI and real-time data, the assistant helps optimize essential resources like energy, water, and waste, while also guiding sustainable behaviors among citizens through personalized tips and services.

For city officials, it serves as a decision-making partner—offering clear insights, forecasting tools, and summarizations of complex policies to support strategic planning. Ultimately, this assistant bridges technology, governance, and community engagement to foster greener cities that are more efficient, inclusive, and resilient.

**Key Features:**

* **Conversational Interface:** Natural language interaction for citizens and officials.
* **Policy Summarization:** Converts lengthy government documents into concise, actionable summaries.
* **Resource Forecasting:** Predicts future energy, water, and waste usage.
* **Eco-Tip Generator:** Provides personalized sustainability advice.
* **Citizen Feedback Loop:** Collects and analyzes public input.
* **KPI Forecasting:** Helps officials track progress and plan ahead.
* **Anomaly Detection:** Flags unusual sensor or usage data.
* **Multimodal Input Support:** Accepts text, PDFs, and CSVs.
* **User-Friendly UI (Streamlit/Gradio):** Intuitive dashboards for interaction.

## **3. Architecture**

* **Frontend (Streamlit):** Interactive web UI with dashboards, file uploads, chat, feedback forms, and report viewers.
* **Backend (FastAPI):** Handles document processing, chat, eco tips, and reports.
* **LLM Integration (IBM Watsonx Granite):** Generates summaries, tips, and reports.
* **Vector Search (Pinecone):** Semantic search using embeddings.
* **ML Modules:** Forecasting and anomaly detection (Scikit-learn, Pandas, Matplotlib).

## **4. Setup Instructions**

**Prerequisites:**

* Python 3.9+
* pip & virtual environment tools
* API keys for IBM Watsonx & Pinecone
* Internet access

**Steps:**

1. Clone repository
2. Install dependencies (requirements.txt)
3. Configure .env with credentials
4. Run backend server with FastAPI
5. Launch frontend via Streamlit
6. Upload data and interact with modules

## **5. Folder Structure**

* app/ → FastAPI backend (routers, models, integrations)
* app/api/ → API routes (chat, feedback, report, vectorization)
* ui/ → Streamlit pages, layouts, and UIs
* smart\_dashboard.py → Main Streamlit dashboard launcher
* granite\_llm.py → IBM Watsonx Granite integration
* document\_embedder.py → Document embeddings with Pinecone
* kpi\_file\_forecaster.py → Forecasting module
* anomaly\_file\_checker.py → Anomaly detection module
* report\_generator.py → AI-generated sustainability reports

## **6. Running the Application**

1. Start FastAPI backend
2. Run Streamlit frontend
3. Navigate via sidebar
4. Upload documents/CSVs
5. Interact with chat assistant & view outputs

## **7. API Documentation**

* **POST /chat/ask** – AI-powered query response
* **POST /upload-doc** – Upload & embed documents
* **GET /search-docs** – Semantic policy search
* **GET /get-eco-tips** – Sustainability recommendations
* **POST /submit-feedback** – Collects citizen feedback

## **8. Authentication**

Currently open for demo. Future enhancements:

* Token-based auth (JWT/API keys)
* OAuth2 with IBM Cloud credentials
* Role-based access (Admin, Citizen, Researcher)
* User session tracking

## **9. User Interface**

* Sidebar navigation
* KPI visualizations & summary cards
* Tabs for chat, eco tips, and forecasting
* PDF report downloads
* Accessible design with help texts

## **10. Testing**

* **Unit Testing:** Prompt functions & utilities
* **API Testing:** Swagger UI & Postman
* **Manual Testing:** File uploads, chat responses, outputs
* **Edge Case Handling:** Large files, malformed inputs, invalid API keys

## **11. Screenshots**

(To be added – screenshots of dashboard, reports, chat interface

## 

