Project Title

Sustainable Smart City Assistant Using IBM Granite LLM

Project Documentation

1. Introduction

• Project title : Sustainable Smart City Assistant Using IBM Granite LLM

• Team member : THANGAPUSHPA M

• Team member : Roshini.S

• Team member : A. Anitha

• Team member : HARISH.V.M

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.

• Features:

- Conversational Interface: Natural language interaction to allow citizens and officials to ask questions, get updates, and receive guidance in plain language.

- Policy Summarization: Converts lengthy government documents into concise, actionable summaries.

- Resource Forecasting: Estimates future energy, water, and waste usage using historical and real-time data.

- Eco-Tip Generator: Recommends daily actions to reduce environmental impact based on user behavior.

- Citizen Feedback Loop: Collects and analyzes public input to inform city planning and service improvements.

- KPI Forecasting: Projects key performance indicators to help officials track progress and plan ahead.

- Anomaly Detection: Identifies unusual patterns in sensor or usage data to flag potential issues.

- Multimodal Input Support: Accepts text, PDFs, and CSVs for document analysis and forecasting.

- Streamlit or Gradio UI: Provides an intuitive dashboard for both citizens and city officials to interact with the assistant.

3. Architecture

Frontend (Streamlit): Provides an interactive web UI with dashboards, chat, feedback forms, and report viewers. Modularized for scalability.

Backend (FastAPI): Powers API endpoints for document processing, chat, eco tip generation, and vector embedding with asynchronous performance.

LLM Integration (IBM Watsonx Granite): Enables natural language understanding, summarization, and sustainability tips.

Vector Search (Pinecone): Stores and retrieves policy embeddings for semantic search.

ML Modules (Forecasting & Anomaly Detection): Uses lightweight models to forecast trends and detect anomalies.

4. Setup Instructions

• Prerequisites: Python 3.9+, pip, API keys for IBM Watsonx and Pinecone, internet access.

• Installation: Clone repository, install requirements, configure credentials, run backend (FastAPI), and launch frontend (Streamlit).

5. Folder Structure

app/ – FastAPI backend logic

app/api/ – Modular API routes (chat, feedback, reports, embeddings)

ui/ – Streamlit frontend components

smart_dashboard.py – Launches dashboard

granite_llm.py – Handles IBM Granite LLM communication

document_embedder.py – Manages embeddings with Pinecone

kpi_file_forecaster.py – Forecasts KPI trends anomaly_file_checker.py – Detects anomalies in KPI data

report_generator.py – Builds AI-generated sustainability reports

6. Running the Application

Run FastAPI backend, launch Streamlit UI, upload documents or data, interact with assistant, view reports and insights.

7. API Documentation

POST /chat/ask – AI chat responses

POST /upload-doc – Upload and embed documents

GET /search-docs – Semantic search for policies

GET /get-eco-tips – Provides eco-friendly suggestions

POST /submit-feedback – Records citizen feedback

8. Authentication

This project currently runs in an open environment for demonstration purposes.

For secure deployments:

- Use JWT or API key authentication.

- Implement OAuth2 with IBM Cloud credentials.

- Introduce role-based access (admin, citizen, researcher).

9. User Interface

The user interface is designed to be minimal, clean, and highly functional. It includes:

- Sidebar navigation for quick access.

- KPI visualization with charts and summary cards.

- Tab-based layouts for eco-tips, forecasting, and reports.

- Real-time feedback forms.

- PDF export option for reports.

10. Testing

Testing is performed in multiple stages:

- Unit Testing for individual functions.

- API Testing using Postman and Swagger UI.

- Manual Testing with sample datasets.

- Edge Case Testing for invalid inputs and large files.

11. Known Issues

1. Integration challenges when handling real-time IoT data.

2. Occasional latency in semantic search queries.

3. Dependency version conflicts during upgrades.

12. Future Enhancements

1. Expansion of multi-language support for inclusivity.

2. Integration with IoT sensors for live data collection.

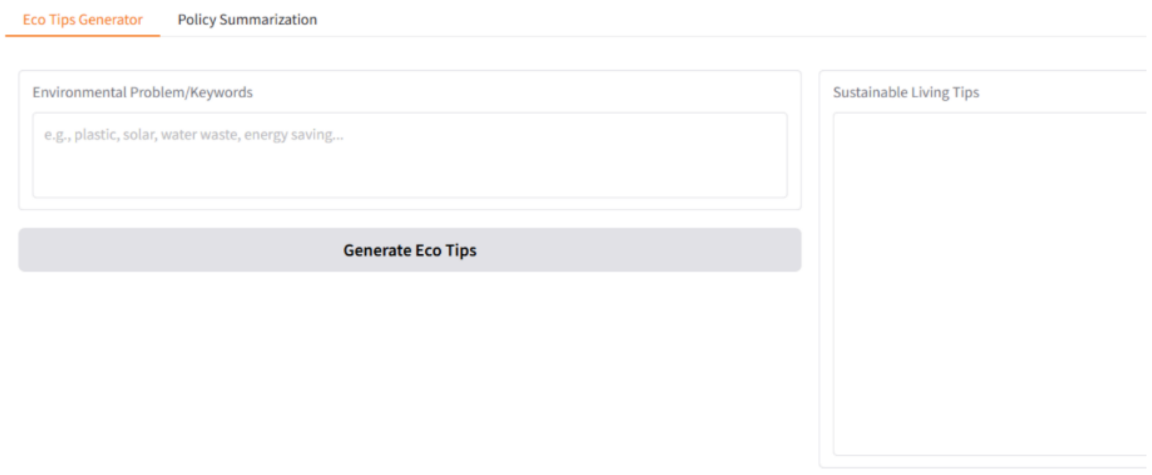
3. Enhanced data visualization with AI-driven insights.

4. Mobile app version for broader accessibility.

5. Integration with blockchain for secure policy tracking.

13. Screen Shots

OUTPUT:


Eco Tips Generator
Policy Summarization
Environmental Problem/Keywords
e.g., plastic, solar, water waste, energy saving...
Generate Eco Tips
Sustainable Living Tips