Sustainable Smart City Assistant

• Project Documentation

1. Introduction:

- Project Title: Sustainable Smart City Assistant
- Team member: SHYAM.A
- Team member: RAJESH.S
- Team member: VIJENDHRAN. P
- Team member: DHANUSH. 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: Allows 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.

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

3. Architecture:

Frontend: The frontend is built with Gradio, offering an interactive web UI with multiple pages including a chat interface and policy summarizer. The design is user-friendly and functional.

Backend: A FastAPI server serves as the backend REST framework, powering API endpoints for document processing, chat interactions, eco-tip generation, and report creation.

LLM Integration: Granite LLM models from IBM Watsonx are used for natural language understanding and generation, carefully designed to generate summaries, sustainability tips, and reports.

Vector Search: Uploaded policy documents are embedded using Sentence Transformers and stored in Pinecone. Semantic search is implemented using cosine similarity to allow users to search documents using natural language queries.

ML Modules: Lightweight ML models are used for forecasting and anomaly detection using Scikit-learn. Time-series data is parsed, modeled, and visualized using pandas and matplotlib.

4. Setup Instructions:

Prerequisites

Python 3.9 or later

pip and virtual environment tools

API keys for IBM Watsonx and Pinecone

Internet access to cloud services

Installation Process

Clone the repository.

Install dependencies from requirements.txt.

Create a .env file and configure credentials.

Run the backend server using FastAPI.

Launch the frontend via Gradio.

Upload data and interact with the modules.

5. Folder Structure:

app/: Contains all FastAPI backend logic including routers, models, and integration modules.

app/api/: Subdirectory for modular API routes like chat, feedback, report, and document vectorization.

ui/: Contains frontend components for Gradio pages, card layouts, and form UIs.

granite_llm.py: Handles all communication with the IBM Watsonx Granite model.

document_embedder.py: Converts documents to embeddings and stores them in Pinecone.

kpi_file_forecaster.py: Forecasts future energy/water trends using regression.

anomaly_file_checker.py: Flags unusual values in uploaded KPI data.

report_generator.py: Constructs AI-generated sustainability reports.

6. Running the Application:

Run the Gradio application to access the web interface.

Navigate through the pages via the tabs.

Upload documents, interact with the chat assistant, and view outputs like reports, summaries, and predictions. All interactions are real-time and use backend APIs to dynamically update the frontend.

7. API Documentation:

POST /chat/ask: Accepts a user query and responds with an AI-generated message.

POST /upload-doc: Uploads and embeds documents in Pinecone.

GET /search-docs: Returns semantically similar policies to the input query.

GET /get-eco-tips: Provides sustainability tips for selected topics.

POST /submit-feedback: Stores citizen feedback for review or analytics.

8. Authentication:

This version of the project runs in an open environment for demonstration. However, secure deployments can integrate:

OAuth2 with IBM Cloud credentials

Role-based access (admin, citizen, researcher)

Planned enhancements include user sessions and history tracking.

9. User Interface:

The interface is minimalist and functional, focusing on accessibility for non-technical users. It includes:

Tabs with navigation

KPI visualizations with summary cards

Real-time form handling

PDF report download capability

The design prioritizes clarity, speed, and user guidance with help texts and intuitive flows.

10. Testing:

Testing was done in multiple phases:

Unit Testing: For prompt engineering functions and utility scripts.

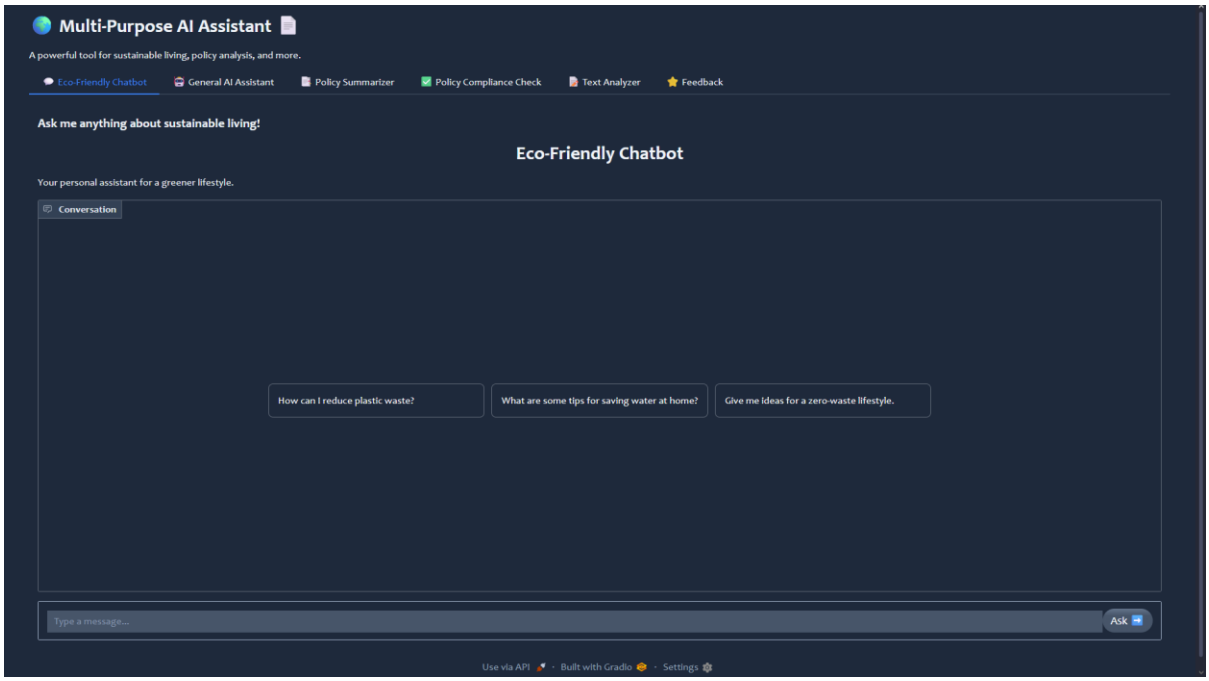
API Testing: Via Swagger UI, Postman, and test scripts.

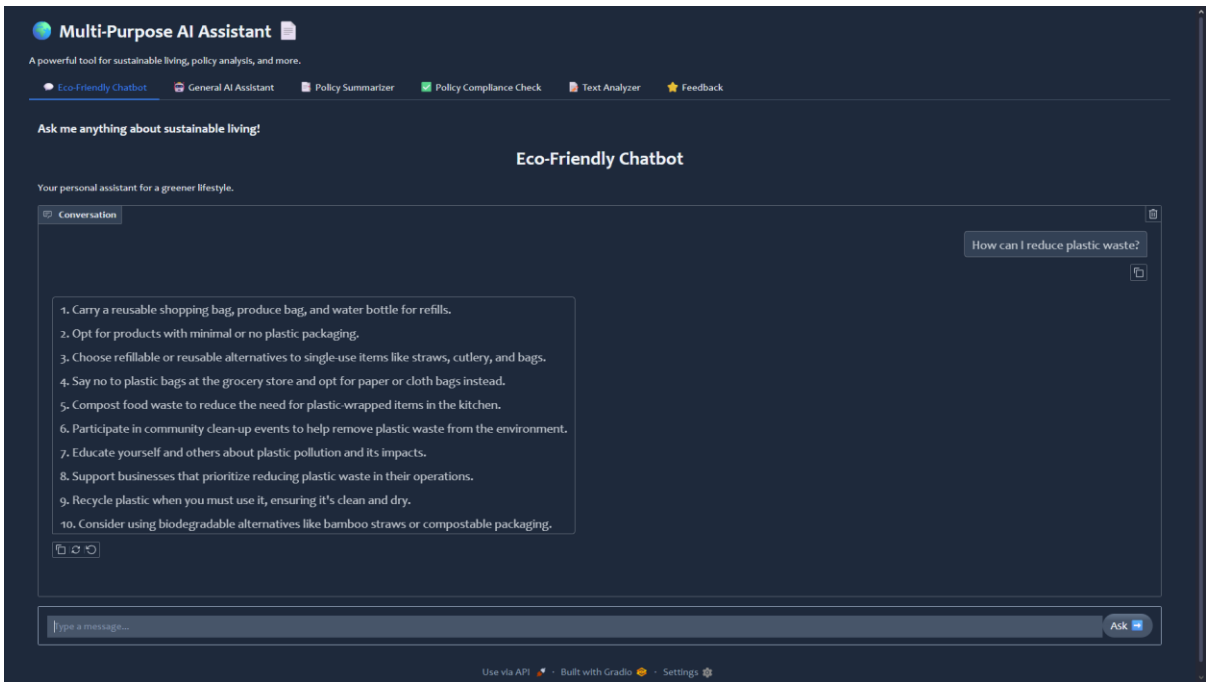
Manual Testing: For file uploads, chat responses, and output consistency.

Edge Case Handling: Malformed inputs, large files, invalid API keys.

Each function was validated to ensure reliability in both offline and API-connected modes.

11.Screenshot:





12.Future Enhancements:

- **Personalized & Targeted Services**: Use generative AI to analyze citizen data from various sources to understand needs and provide personalized services, such as tailored urban planning information or customized sustainability recommendations.
- **Proactive Assistance:** Develop chatbots and virtual assistants that not only respond to queries but also offer proactive, real-time support and information, like alerts for

public transport changes or personalized energy consumption advice.