Sustainable Smart City Assistant Using IBM Granite LLM

Project Documentation

1.Introduction

- Project title : sustainable smart city assistant using IBM granite LLM
- Team member : Dharun kumar S
- Team member : Ruthra chandran V
- Team member : Sanjai p
- Team member : Sathya A

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 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**
  Key Point: Natural language interaction
  Functionality: Allows citizens and officials to ask questions, get updates, and receive guidance in plain language.
- **Policy Summarization**
  Key Point: Simplified policy understanding
  Functionality: Converts lengthy government documents into concise, actionable summaries.

- **Resource Forecasting**
  Key Point: Predictive analytics
  Functionality: Estimates future energy, water, and waste usage using historical and real-time data.
- **Eco-Tip Generator**
  Key Point: Personalized sustainability advice
  Functionality: Recommends daily actions to reduce environmental impact based on user behavior.
- **Citizen Feedback Loop**
  Key Point: Community engagement
  Functionality: Collects and analyzes public input to inform city planning and service improvements.
- **KPI Forecasting**
  Key Point: Strategic planning support
  Functionality: Projects key performance indicators to help officials track progress and plan ahead.
- **Anomaly Detection**
  Key Point: Early warning system
  Functionality: Identifies unusual patterns in sensor or usage data to flag potential issues.
- **Multimodal Input Support**
  Key Point: Flexible data handling
  Functionality: Accepts text, PDFs, and CSVs for document analysis and forecasting.
- **Streamlit or Gradio UI**
  Key Point: User-friendly interface
  Functionality: Provides an intuitive dashboard for both citizens and city officials to interact with the assistant.

3. Architecture

Frontend(Streamlit):

Built with Streamlit, offering an interactive web UI with dashboards, file uploads, chat interfaces, feedback forms, and report viewers. Navigation is handled through the streamlit-option-menu library. Each page is modularized for scalability.

Backend(FastAPI):

FastAPI serves as the REST framework powering document processing, chat interactions, eco tip generation, report creation, and vector embedding. Optimized for asynchronous performance and Swagger integration.

LLMIntegration(IBMWatsonxGranite):

Granite LLM models from IBM Watsonx are used for natural language understanding and generation. Carefully designed prompts generate summaries, sustainability tips, and reports.

VectorSearch(Pinecone):

Uploaded policy documents are embedded using Sentence Transformers and stored in Pinecone. Semantic search is implemented using cosine similarity for querying.

MLModules(Forecasting and Anomaly Detection):

Lightweight ML models built with Scikit-learn parse and visualize time-series data 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

Installation Process:

1. Clone the repository.
2. Install dependencies from requirements.txt.
3. Create a .env file and configure credentials.
4. Run the backend server using FastAPI.
5. Launch the frontend via Streamlit**.**

5. Folder Structure

app/ # FastAPI backend logic

app/api/ # Modular API routes

ui/ # Frontend Streamlit components

smart_dashboard.py # Streamlit entry point

granite_llm.py # IBM Watsonx Granite communication

document_embedder.py # Document embeddings with Pinecone

kpi_file_forecaster.py # KPI forecasting

anomaly_file_checker.py # Anomaly detection

report_generator.py # Sustainability reports generation

6.Running the Application

1. Launch the FastAPI backend server.
2. Run the Streamlit frontend.
3. Navigate through the sidebar.
4. Upload documents, CSVs, and interact with chat interfaces.
5. View reports, summaries, and predictions in real-time.

7. API Documentation

Available endpoints:

- POST /chat/ask – Accepts queries and generates responses.
- POST /upload-doc – Uploads documents for embedding.
- GET /search-docs – Semantic document search.
- GET /get-eco-tips – Provides sustainability tips.
- POST /submit-feedback – Stores citizen feedback.

Endpoints are tested using Swagger UI.

8. Authentication

For demonstration, the project runs in an open environment. For secure deployment, you can integrate:

- Token-based authentication (JWT or API keys)
- OAuth2 with IBM Cloud credentials
- Role-based access control (admin, citizen, researcher)

Planned features: session management and history tracking.

9. User Interface

Key UI elements:

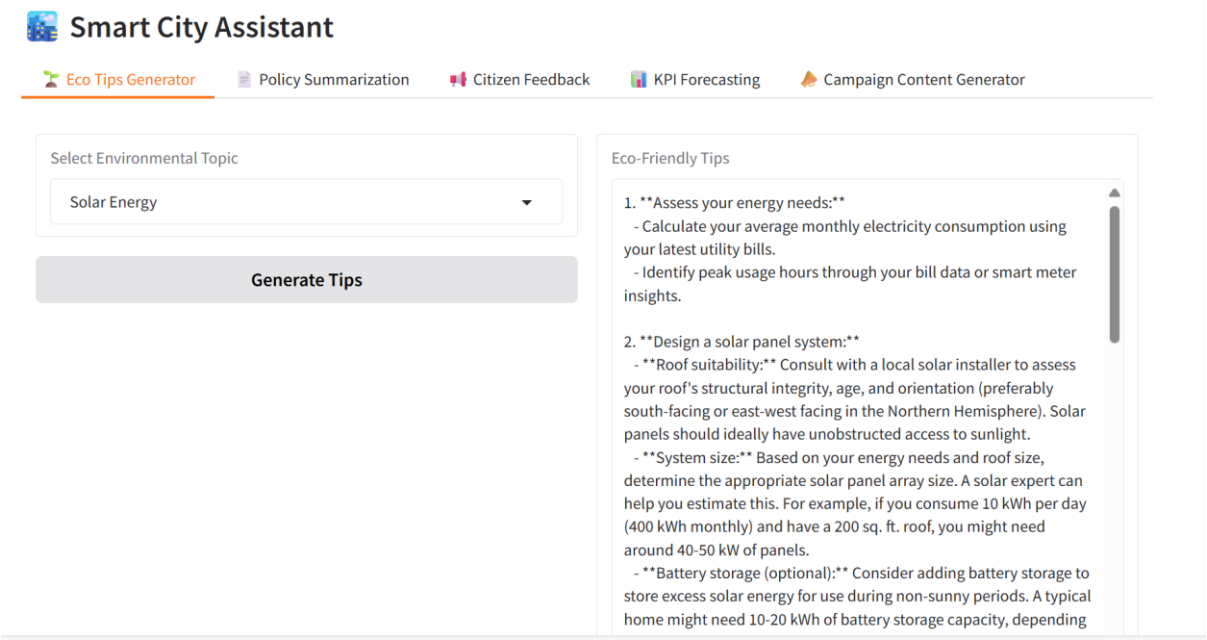
- Sidebar navigation
- KPI visualizations
- Tabbed layouts for chats, eco tips, forecasting
- Real-time form handling
- PDF report downloads

Designed for accessibility, clarity, and speed**.**

10. Testing

- Unit testing of prompt functions
- API testing using Swagger and Postman
- Manual testing of uploads and outputs
- Edge case handling for invalid inputs

11.screen shots



12.Known Issues

- KPI forecasting is oversimplified (fixed 5% growth).
- Limited error handling for wrong inputs.
- Citizen feedback and campaign options are restricted to predefined choices.
- Temporary deployment (`share=True`), not production-ready.
- No user authentication or data privacy measures.

13.Future enhancement

- Improve model accuracy with larger or fine-tuned LLMs.
- Add OCR support for scanned PDFs and better text extraction.
- Develop advanced KPI forecasting using ML time-series models.
- Allow custom citizen feedback input (not just dropdowns).
- Implement user authentication and data privacy measures.

- Enable multi-user support and scalable deployment.
- Add multilingual support for wider accessibility.