📘 Smart City Project Documentation

1. Introduction

Project Title: Sustainable Smart City Assistant

Team Leader:

S.Subashini

Team Members:

• Member1 : G.Sangeetha

• Member2 : M.Sandhiya

• Member3 :S.Vinitha

---

2. Project Overview

Purpose:

The Sustainable Smart City Assistant empowers cities and citizens to live in a more eco-conscious, efficient, and connected urban environment. By using AI and real-time data, it helps optimize resources such as energy, water, and waste, while also promoting sustainable practices among residents. For city administrators, it acts as a decision-support system by providing policy summaries, forecasts, and citizen feedback insights.

Features:

• Conversational Interface (natural language interaction)

• Policy Summarization (simplified understanding of policies)

• Resource Forecasting (predict future usage of energy/water/waste)

• Eco-Tip Generator (personalized sustainability tips)

• Citizen Feedback Loop (community engagement and planning)

• KPI Forecasting (strategic decision-making support)

• Anomaly Detection (early warnings on unusual patterns)

• Multimodal Input Support (accepts text, PDFs, CSVs)

• User-Friendly Dashboard (Streamlit/Gradio interface)

---

3. Architecture

Frontend: Built using Streamlit/Gradio for an intuitive web interface with dashboards, file uploads, and chat.

Backend: FastAPI provides REST APIs for document analysis, eco-tips, forecasting, and chat.

LLM Integration: IBM Watsonx Granite models are used for summarization, sustainability advice, and natural conversation.

Vector Database: Pinecone stores document embeddings for semantic search.

ML Modules: Forecasting and anomaly detection powered by scikit-learn, pandas, and matplotlib.

---

4. Setup Instructions

Prerequisites:

Python 3.9+

pip, virtual environment tools

API keys for IBM Watsonx & Pinecone

Internet access

Installation Steps:

1. Clone the repository

2. Install dependencies from requirements.txt

3. Configure API keys in .env file

4. Run backend server with FastAPI

5. Launch frontend with Streamlit/Gradio

6. Upload data and interact with modules

---

5. Folder Structure

app/ → Backend logic (FastAPI routes, models, integrations)

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

ui/ → Frontend pages & components (Streamlit/Gradio)

smart\_dashboard.py → Main Streamlit dashboard launcher

granite\_llm.py → Handles IBM Granite LLM integration

document\_embedder.py → Embeds and stores documents in Pinecone

kpi\_file\_forecaster.py→ Forecasts resource usage trends

anomaly\_file\_checker.py→ Detects anomalies in KPI data

report\_generator.py → Generates AI-driven sustainability reports

---

6. Running the Application

1. Start FastAPI backend

2. Run Streamlit/Gradio frontend

3. Navigate via sidebar to dashboards, chat, eco-tips, and forecasting

4. Upload documents/CSVs for analysis

5. Receive reports, insights, and predictions in real-time

---

7. API Documentation

POST /chat/ask → AI-powered query response

POST /upload-doc → Upload and embed documents

GET /search-docs → Semantic document search

GET /get-eco-tips → Personalized eco-friendly recommendations

POST /submit-feedback → Citizen feedback submission

---

8. Authentication

Supports:

Token-based authentication (JWT/API keys)

OAuth2 with IBM Cloud credentials

Role-based access (Admin, Citizen, Researcher)

---

9. User Interface

Sidebar navigation

KPI dashboards & charts

Tabs for chat, eco-tips, forecasting

Report download (PDF)

Accessible for non-technical users

---

10. Testing

Unit Testing (prompt functions, utilities)

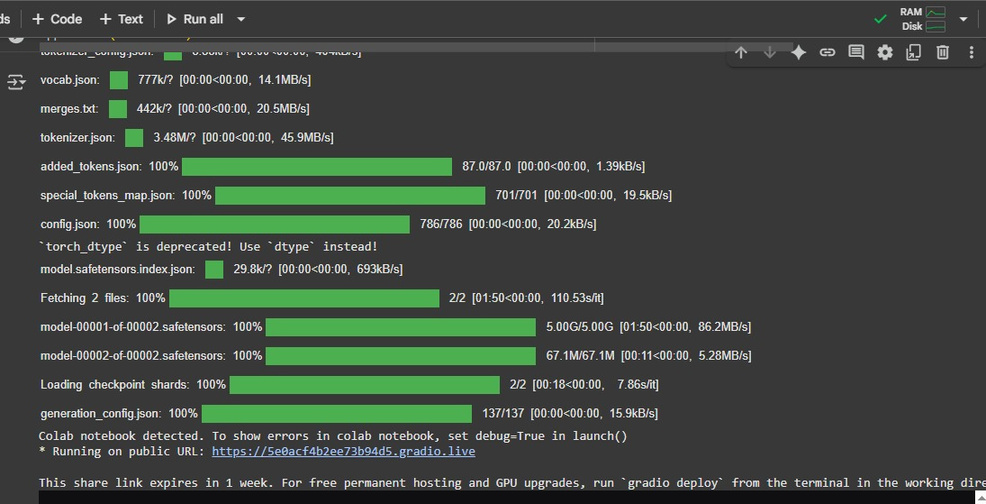
API Testing (Swagger UI, Postman)

Manual Testing (uploads, outputs, UI flows)

Edge Case Handling (invalid inputs, large files, wrong keys)

---

11. Screenshots



---

12. Known Issues

Limited scalability on free-tier GPU (Colab)

Latency with large document uploads

Dependency on cloud APIs (IBM Watsonx, Pinecone)

---

13. Future Enhancements

Mobile application integration

Multi-language support

Advanced predictive analytics for urban planning

Blockchain-based citizen data security

Integration with IoT smart city devices

---