Team Id: LTVIP2025TMID31849

Project Title: Sustainable Smart City Assistant Using IBM Granite LLM

Abstract

Sustainable Smart City Using IBM Granite LLM is an AI-driven urban innovation project designed to tackle modern metropolitan challenges under the banner of smart, sustainable development. Conceptualized as part of a next-gen initiative, the solution harnesses the power of the IBM Granite Large Language Model (LLM) to transform traditional cities into intelligent ecosystems. By integrating real-time sensor networks, predictive analytics, and scalable cloud infrastructure, the platform enables adaptive traffic control, optimized energy and water usage, and intelligent waste management. Citizens benefit from enhanced quality of life through data-informed governance and interactive service access. The architecture emphasizes modularity, security, and citizen engagement, ensuring robust performance even at scale. With its AI core and human-centric design, the Sustainable Smart City project lays the groundwork for globally replicable smart urban environments that prioritize resilience, efficiency, and ecological balance.

INTRODUCTION

Urban environments around the world are facing an unprecedented convergence of challenges—ranging from environmental degradation and resource scarcity to inefficient infrastructure and growing population demands. Traditional governance models often fall short in addressing the dynamic needs of modern cities. In response, the Sustainable Smart City Using IBM Granite LLM initiative presents a transformative vision of how cuttingedge

artificial intelligence can be harnessed to create intelligent, resilient, and ecologically responsible urban systems.

At the heart of this project lies the IBM Granite Large Language Model (LLM), which serves as the cognitive engine behind a suite of interconnected smart city solutions. From adaptive traffic systems and predictive energy management to real-time pollution monitoring and citizen-centric service delivery, the platform is engineered to optimize urban living through automation, data-driven decisions, and responsive design.

This document outlines the end-to-end development process—from ideation and stakeholder empathy mapping to architectural design, functional modules, and implementation planning. By combining AI capabilities with cloud scalability, IoT networks, and an intuitive user interface, this project not only addresses critical urban pain points but also sets a replicable framework for future-ready cities.

The ultimate aim is to strike a balance between technological advancement and environmental stewardship, enabling cities that are not just smart—but also sustainable, inclusive, and future-proof.

MODULE DESCRIPTION

1. Project Overview Purpose:

The **Sustainable Smart City AI Assistant** is developed to address pressing urban challenges such as traffic congestion, pollution, energy inefficiency, and waste management by leveraging artificial intelligence. The core aim is to create an intelligent, data-driven platform that supports city planners, administrators, and citizens in making smarter decisions for sustainable urban living. By simulating real-time scenarios and delivering AI-generated insights, the system empowers modern cities to become more resilient, efficient, and environmentally responsible. the project integrates foundational web technologies with the potential of IBM Granite LLM and IoT-based data simulation to serve as a prototype for next-generation urban governance.

Key Features:

- **AI-Driven Decision Support**: Integrates with large language models (e.g., IBM Granite LLM) to provide predictions, recommendations, and intelligent responses based on environmental and urban data.
- **Real-Time Response System**: Uses Flask backend to simulate or respond to dynamic urban conditions such as pollution levels, traffic flow, or energy usage.
- **Citizen-Focused Interface**: Provides a lightweight frontend (via HTML) to display AI outputs and system status, which can be expanded into a full-featured dashboard.
- **Modular & Scalable Backend:** Built with Python and Flask, designed to be easily extended with additional modules like traffic control, energy management, or emergency alerts.
- **Environmentally Conscious Design:** Encourages sustainable urban planning through data insights and digital simulation, helping reduce resource waste and enhance livability.
- **Secure Configuration Management.**

2. Architecture

Frontend :

The frontend interface of the Sustainable Smart City AI Assistant is structured using a static HTML file (index.html), providing users with a simple and accessible entry point to the platform. This interface serves as the primary layer for user interaction—allowing citizens, planners, or administrators to view the system output and visual reports.

Backend:

The backend logic is implemented in app.py using Flask, a lightweight Python web framework known for rapid development and easy integration with machine learning models and APIs. The Flask server handles HTTP requests from the frontend, manages API routes, and interacts with the AI services and sensor-based logic embedded in the application.

Key backend functionalities include:

- Serving real-time responses based on AI predictions (e.g., pollution alerts, traffic insights)
- Handling data inputs and simulating sensor-based triggers • Rendering results directly to the frontend or dashboard endpoints **Environment & Virtual Setup:**

The project uses a virtual environment (myenv) to manage Python dependencies cleanly and isolate them from global packages. All required libraries and modules are

listed in requirements.txt, ensuring that the application can be replicated or deployed in different environments with consistent behavior.

Supporting Files:

- .gitignore ensures environment files and local dependencies are excluded from version control.
- README.md provides documentation and usage instructions.
- LICENSE defines the open-source usage terms.
- screenshots/ contains visual documentation or output snapshots for demo or testing purposes.

This planned integration would allow the backend to communicate with cloudhosted AI services and expand the application into a truly intelligent city assistant platform.

3. Setup Instructions

Prerequisites:

Before setting up the project, ensure the following software is installed on your system:

Python (v3.8 or higher) pip (Python package manager)

Git (for cloning the repository)

Virtual Environment tool (venv or virtualenv)

Code Editor (e.g., VS Code), Web Browser (Chrome, Firefox, etc.)

Installation Steps: Clone the Repository:

https://github.com/manasaabdas/Sustainable-Smart-City-Assistant-Using-IBM-Granite-LLM-/tree/main

Navigate to the Project Directory:

cd sustainable-smart-city-ai-assistant

Create virtual environment **python -m venv myenv**

Activate on Windows

myenv\Scripts\activate

Activate on macOS/Linux **source**

myenv/bin/activate

Install Dependencies:

pip install -r requirements.txt

Set Up Environment Variables:

Create a .env file in the root directory and add necessary configuration (e.g., API keys if using IBM Granite or other services).

Example:

FLASK_ENV=development

SECRET_KEY=your_secret_key

Run the Flask Application:

python app.py

Once the server is running, open your browser and visit:

http://localhost:5000 to interact with the AI

Assistant web interface.

4. API Documentation

This project exposes two main API endpoints through the Flask backend, integrating with Hugging Face API and OpenWeather API to deliver AI-generated insights and live environmental data.

Endpoint Overview :

Endpoint	Method	Description	Auth
/get_weather	POST	Fetches real-time weather data using OpenWeather	No
/generate_insight	POST	Generates smart city insights using Hugging Face LLM	No

5. Project Planning and Scheduling :

The development of the Sustainable Smart City AI Assistant was structured into five key phases, ensuring a smooth transition from ideation to functional deployment. The schedule followed a modular and iterative approach to prioritize core functionality and future scalability.

Phase 1: Ideation & Requirement Analysis

- Defined core objectives of a smart city simulation tool.
- Identified key APIs (Hugging Face, OpenWeather) and basic user scenarios.
- Created empathy maps and problem statements.

Phase 2: Environment Setup & Tech Stack Finalization

- Set up project directory, Git version control, and virtual environment.
- Chose Flask for backend and HTML for minimal frontend.
- Generated and tested API keys securely using .env.

Phase 3: Core Backend Development

- Integrated Hugging Face LLM for generating urban insights.
- Implemented weather data retrieval using OpenWeather API.
- Built and tested Flask routes: /get_weather, /generate_insight.

Phase 4: Frontend & API Connection

- Designed index.html form-based interface for user input.

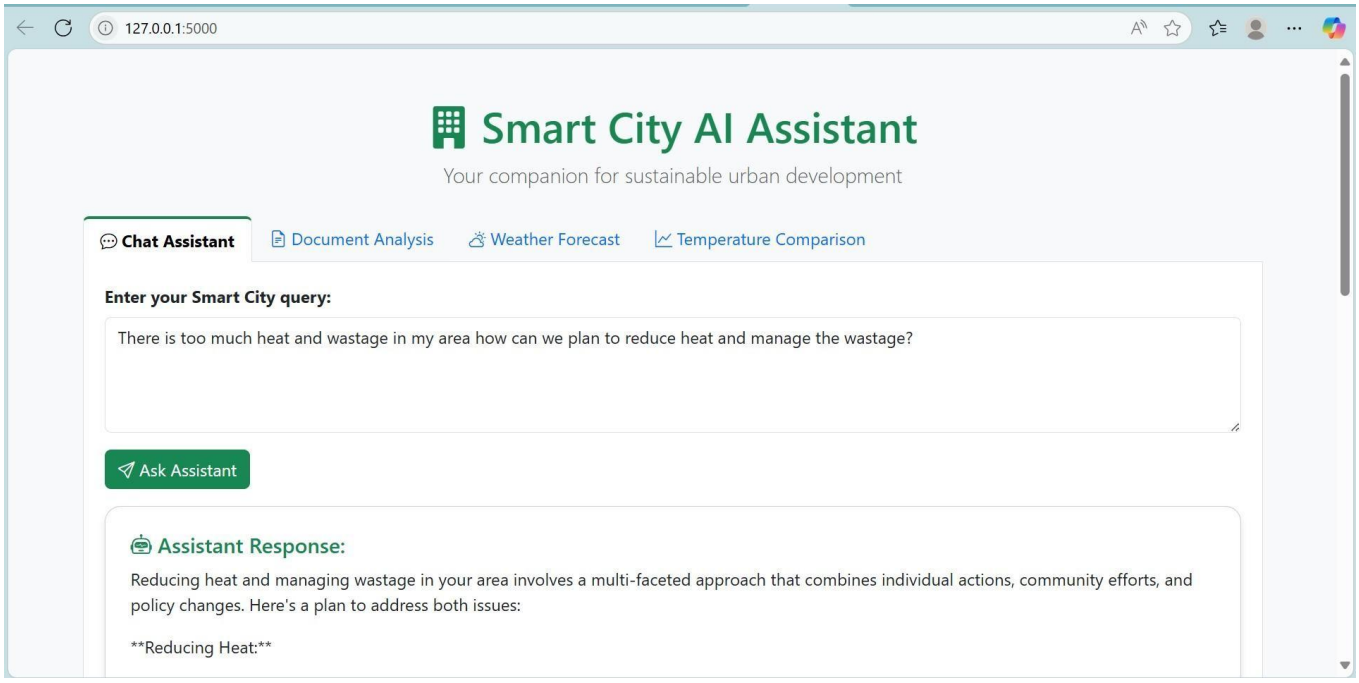
- Connected frontend to backend endpoints using form actions and POST methods.

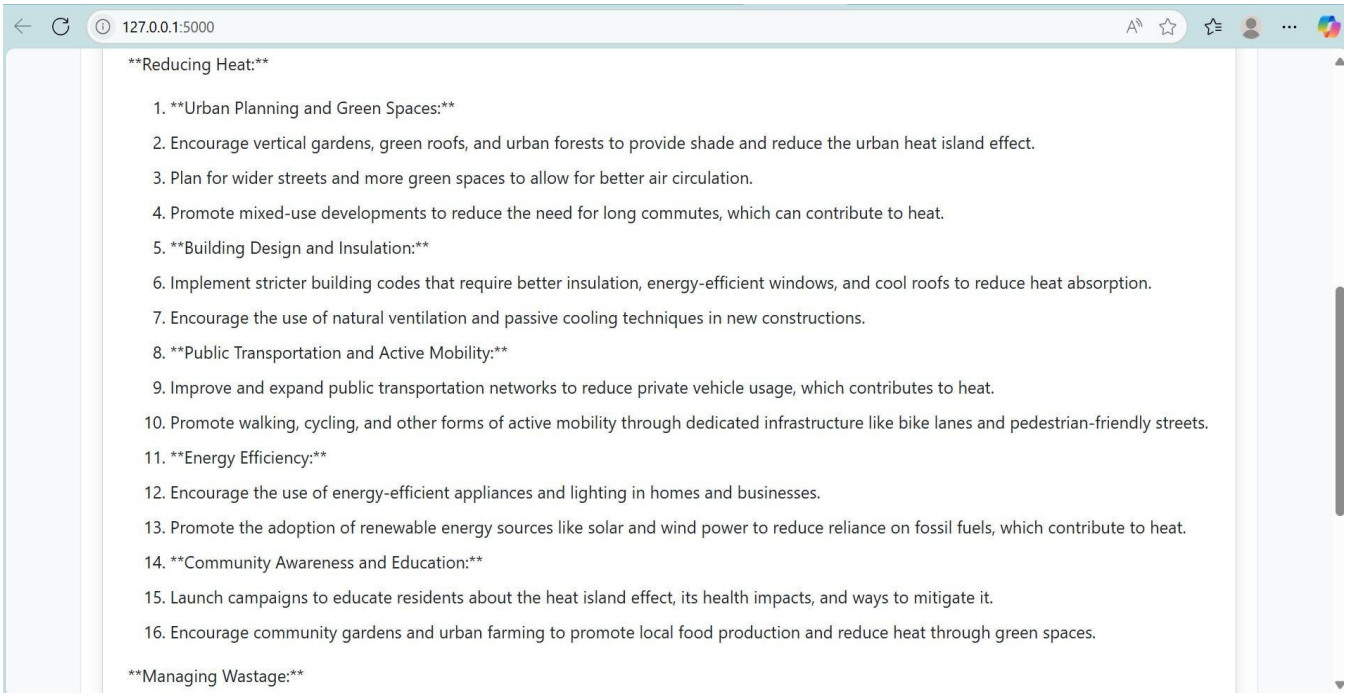
Phase 5: Testing, Documentation & Deployment Prep *(Week 6)*

- Validated API responses and user flow.
- Documented setup, architecture, and API usage.
- Prepared for local deployment and potential cloud hosting.

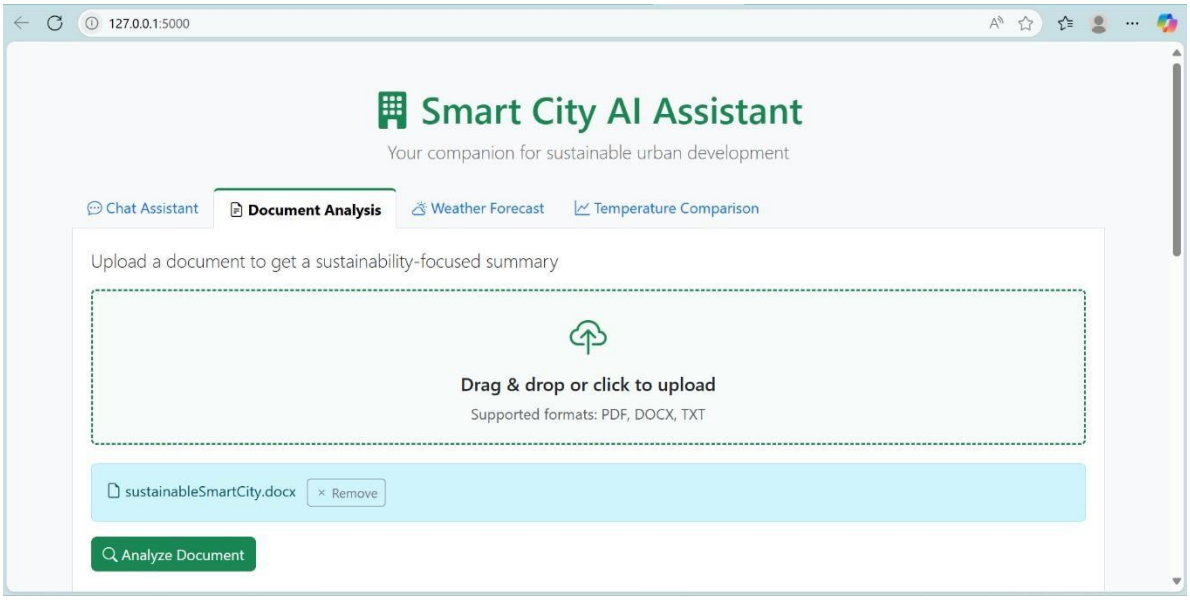
USER INTERFACE

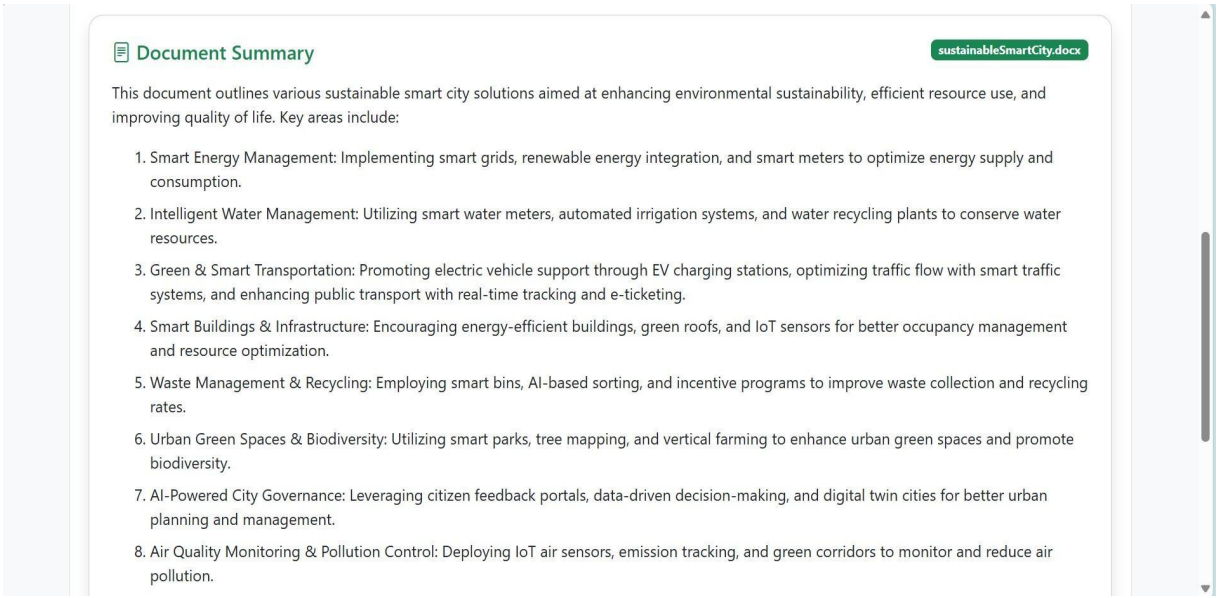
USER VIEW: -

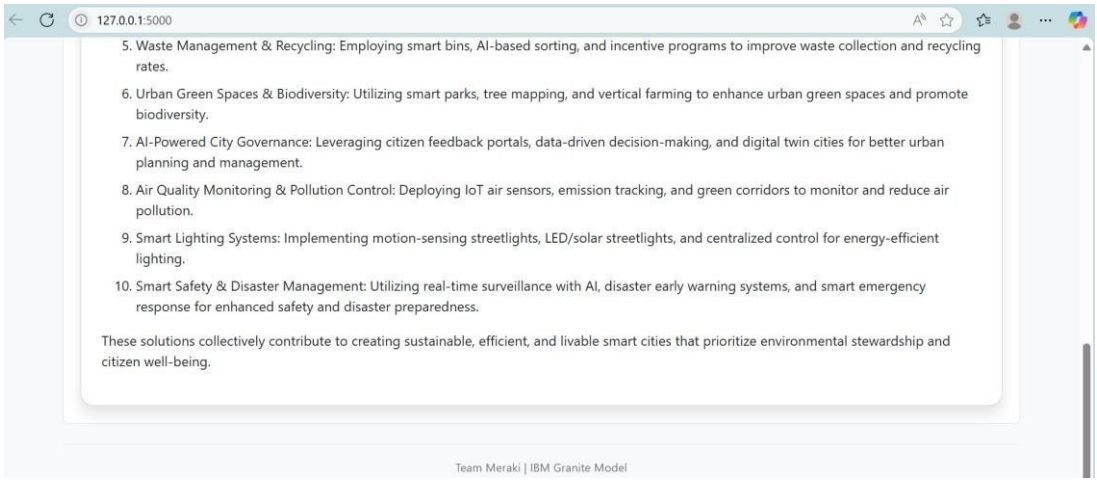




Document Analysis : -







TESTING

Testing was a critical phase in ensuring the correctness, reliability, and smooth functioning of the Sustainable Smart City AI Assistant. Manual methods were primarily used to validate API behavior, user interactions, and integration with thirdparty services. Automated testing is planned for future iterations.

Manual Testing:

- **Frontend:**
  - The index.html interface was tested by simulating user interactions such as form submissions and prompt inputs. ₒ Validated smooth data flow from frontend to backend and proper display of AI-generated insights and weather information.
- **Backend:**
  - All API endpoints (/get_weather, /generate_insight) were tested using **Postman** with both valid and invalid payloads to check:

        - Error handling
        - API latency and response structure
        - External API failures (e.g., invalid API key, city not found)
- **API Integration:**
    - Confirmed correct integration and authentication with **Hugging Face API** and **OpenWeatherAPI** using real tokens and live requests. ◦ Monitored network calls and checked logs to verify reliability and retry behavior.

Responsiveness & UX:

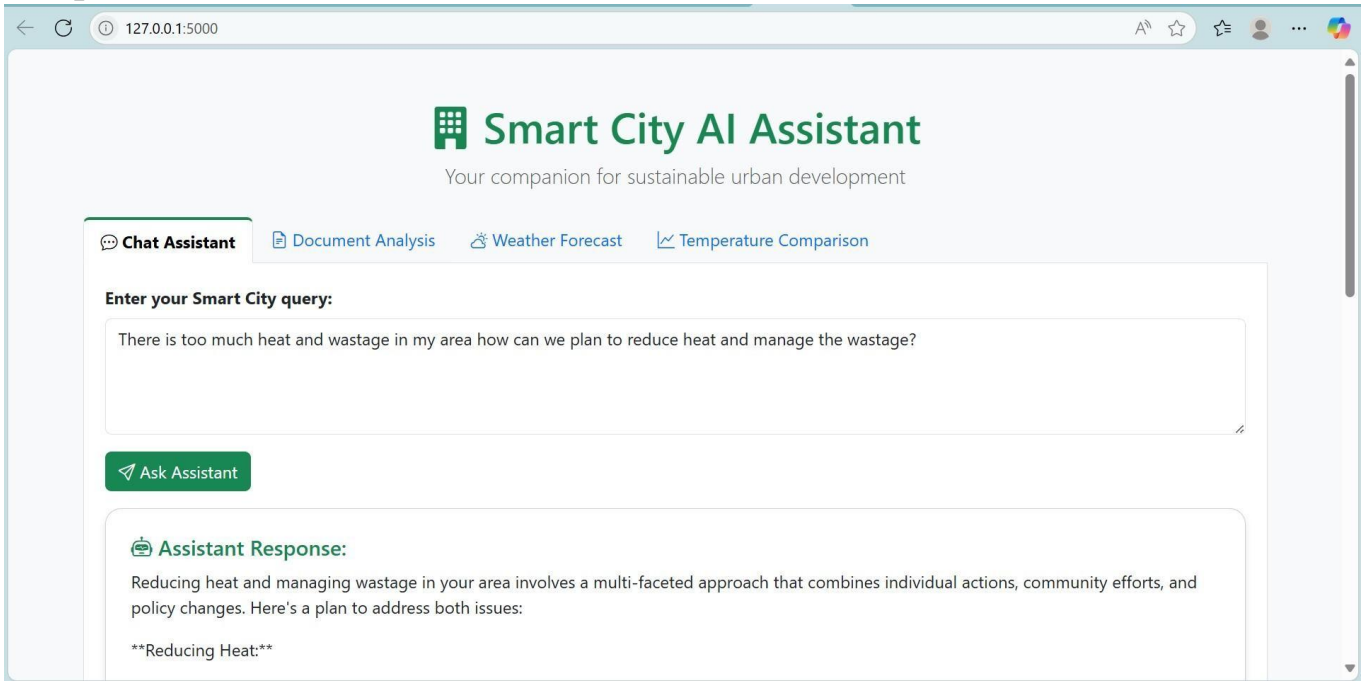
- Although the frontend is basic, rendering was manually tested across different browsers (Chrome, Firefox, Edge).
- Verified that the interface works on both desktop and mobile layouts without breaking form submission.

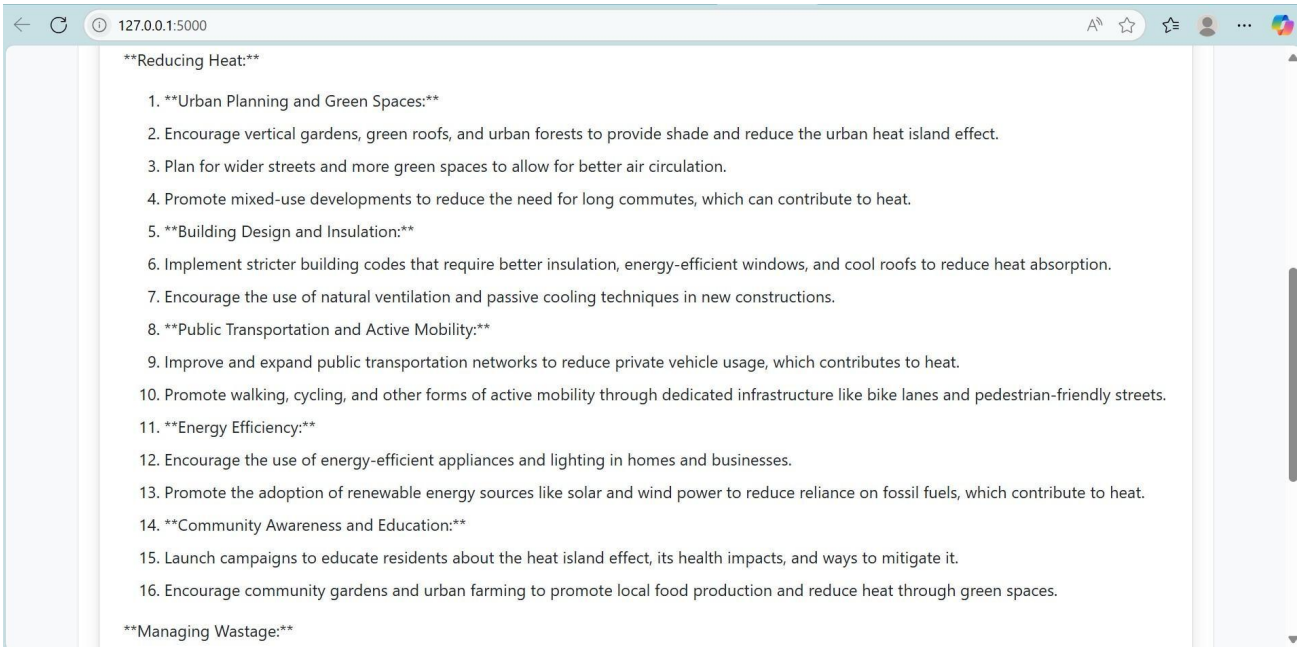
Automated Testing (Planned):

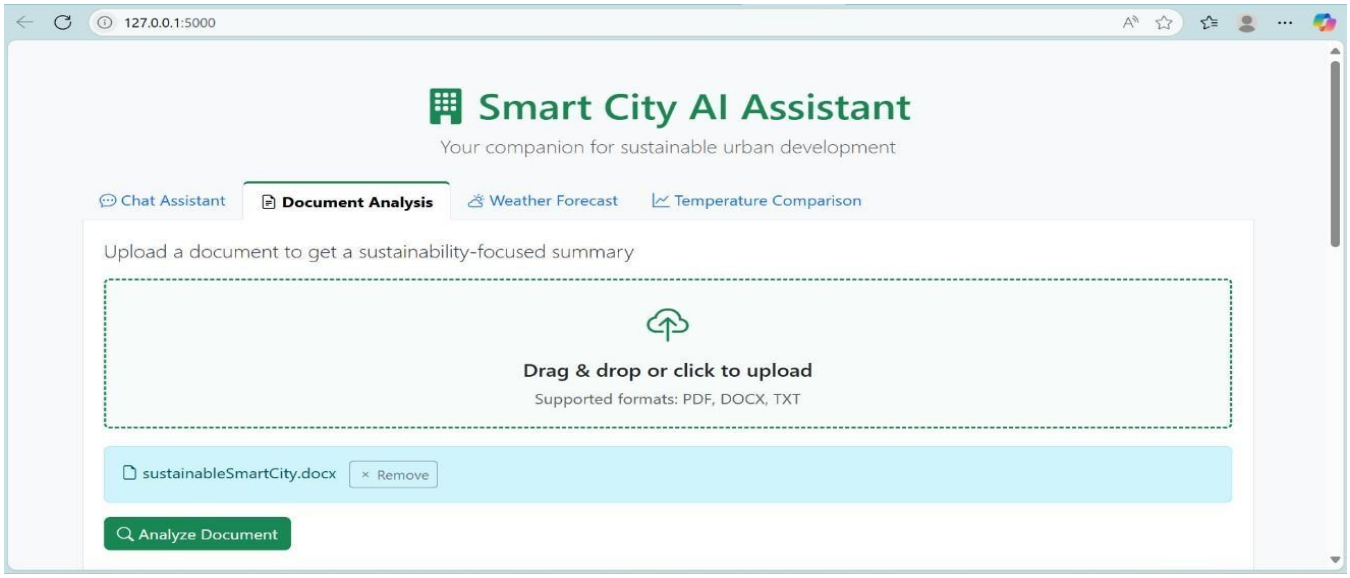
- **Backend (Flask):**
    - Planned integration of **Pytest** and **unittest** to automate endpoint testing and validate edge cases. ◦ Will include test cases for API responses, empty input handling, and API key validation.
- **Future Additions:**
    - UI testing using **Selenium** for end-to-end flow validation. ◦ Mock testing for API responses to simulate network failures and offline mode.

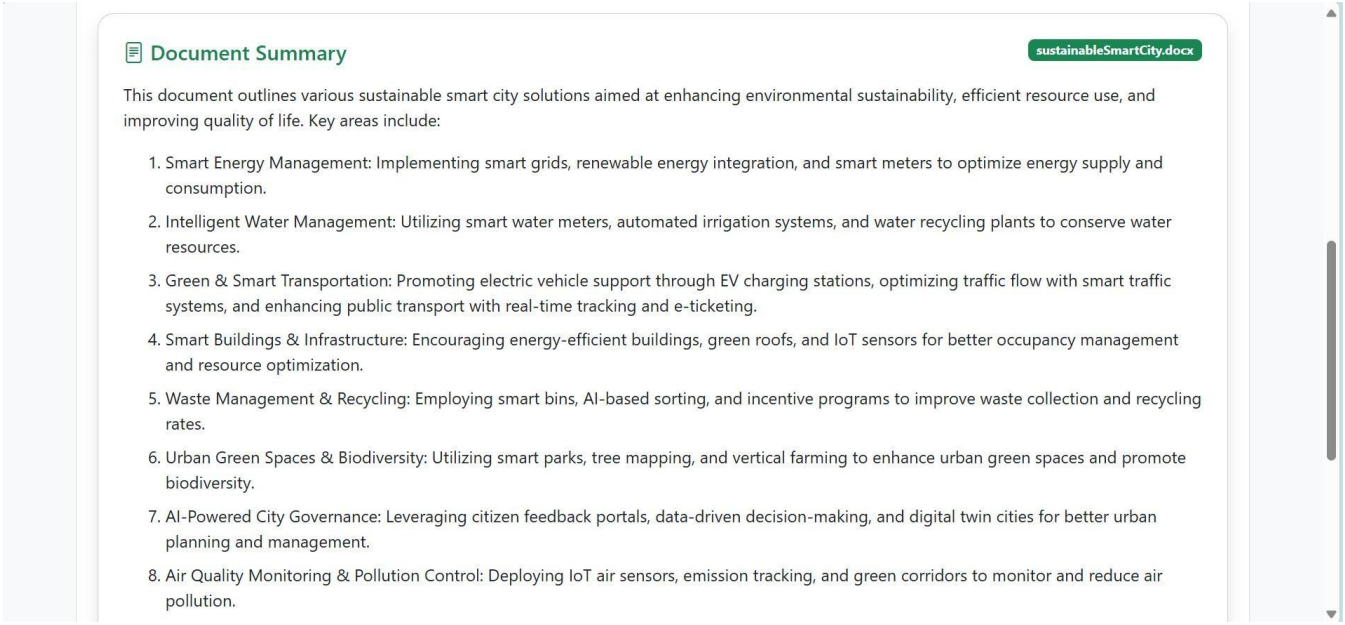
RESULT

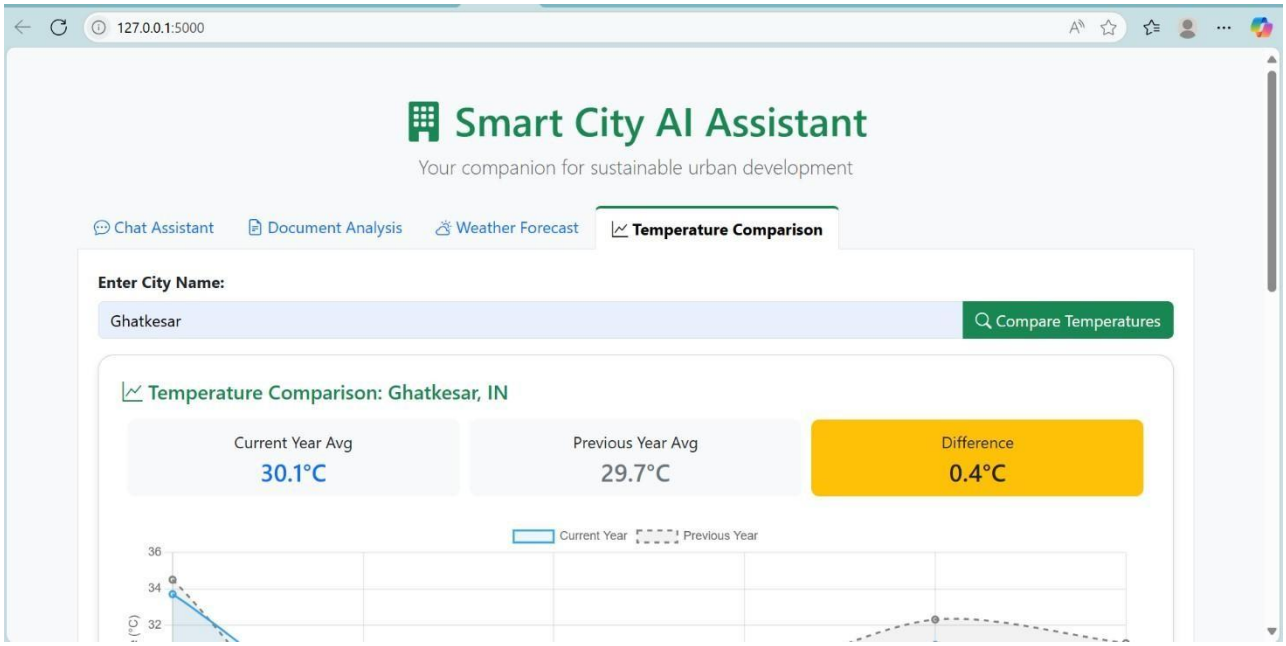
Sample Screenshots:






127.0.0.1:5000
Smart City AI Assistant
Your companion for sustainable urban development
Chat Assistant
Document Analysis
Weather Forecast
Temperature Comparison
Upload a document to get a sustainability-focused summary
Drag & drop or click to upload
Supported formats: PDF, DOCX, TXT
sustainableSmartCity.docx
Remove
Analyze Document


Document Summary
sustainableSmartCity.docx
This document outlines various sustainable smart city solutions aimed at enhancing environmental sustainability, efficient resource use, and improving quality of life. Key areas include:
1. Smart Energy Management: Implementing smart grids, renewable energy integration, and smart meters to optimize energy supply and consumption.
2. Intelligent Water Management: Utilizing smart water meters, automated irrigation systems, and water recycling plants to conserve water resources.
3. Green & Smart Transportation: Promoting electric vehicle support through EV charging stations, optimizing traffic flow with smart traffic systems, and enhancing public transport with real-time tracking and e-ticketing.
4. Smart Buildings & Infrastructure: Encouraging energy-efficient buildings, green roofs, and IoT sensors for better occupancy management and resource optimization.
5. Waste Management & Recycling: Employing smart bins, AI-based sorting, and incentive programs to improve waste collection and recycling rates.
6. Urban Green Spaces & Biodiversity: Utilizing smart parks, tree mapping, and vertical farming to enhance urban green spaces and promote biodiversity.
7. AI-Powered City Governance: Leveraging citizen feedback portals, data-driven decision-making, and digital twin cities for better urban planning and management.
8. Air Quality Monitoring & Pollution Control: Deploying IoT air sensors, emission tracking, and green corridors to monitor and reduce air pollution.



FUTURE ENHANCEMENTS

The current version of the Sustainable Smart City AI Assistant serves as a foundational prototype integrating AI and weather intelligence for urban sustainability. Several enhancements are planned to elevate its functionality, scalability, and real-world applicability.

1. Advanced UI Dashboard

- Replace the static HTML interface with a dynamic React-based dashboard.
- Add real-time charts, maps, and module-wise insights for better visualization.

2. UserAuthentication & Role Management

- Implement secure login and user roles (e.g., citizen, city planner, admin).
- Enable personalized access to AI insights and reports.

3. Multi-API Data Integration

- Extend integrations to other urban data sources (e.g., pollution sensors, traffic APIs).
- Enable multi-parameter analysis for smarter predictions.

4. Database Integration

- Add a backend database (e.g., MongoDB or PostgreSQL) to log user queries, insights, and city reports for auditing and learning.

5. Automated Alert System

- Trigger automated alerts based on weather or AI predictions (e.g., high pollution warnings, traffic redirection).

6. Granite LLM Fine-tuning (Advanced AI Layer)

- Integrate a fine-tuned version of the IBM Granite LLM (or similar) to provide cityspecific solutions and deeper analytics.

7. Deployment & Cloud Hosting

- Deploy the complete application on platforms like IBM Cloud, Heroku, or Render for public access and scalability.

CONCLUSION

The Sustainable Smart City AI Assistant demonstrates how artificial intelligence and real-time data integration can be harnessed to address critical urban challenges in a scalable, accessible, and impactful manner. By leveraging the Hugging Face LLM for intelligent insights and OpenWeather API for live environmental data, this project lays a practical foundation for future smart city applications.

With a lightweight architecture, intuitive interface, and modular design, the assistant offers a glimpse into how data-driven governance can improve decision-making, enhance sustainability, and empower citizens. Though currently a prototype, the project's structure allows for seamless expansion into areas like traffic control, energy management, and predictive analytics — pushing urban living towards a smarter, greener, and more resilient future.