SMARTBRIDGE

Generative AI with IBM Cloud

Andhra Pradesh, India

Sustainable Smart City Assistant
Using IBM Granite LLM

Developed by: Ashish Krishna Pavan

And Team

Academic Year: 2024–2025

Date	May 2025 – June 2025
Team ID	LTVIP2025TMID60699
Project Name	Sustainable Smart City Assistant Using IBM Granite LLM

Submitted in fulfilment of the requirements for the Generative AI Internship – 2025

Submitted by:

G. Ashish Krishna Pavan and Team

Team Details: (ID - LTVIP2025TMID60699**)**

Team Member -1: Gade Ashish Krishna Pavan

Team Member - 2: Doddipatla Siva Kasi

Team Member -3: Gandimenu Akshaya

Team Member -4: Garini Narayana Murthy

Table of Contents

1. INTRODUCTION:

1.1 Project Overview:

The idea for the Sustainable Smart City Assistant came from a simple realization: with cities expanding rapidly, it's becoming harder to monitor and manage all the things that make urban life livable — like air quality, energy usage, traffic congestion, and public health. We wanted to build something that could help with that — something intelligent, interactive, and useful.

So we created a smart assistant that does more than just display numbers. It gives real-time insights on important city metrics through a clear, colorful dashboard. It also offers features like chat-based assistance, eco-friendly tips, document summarization, forecasting, and anomaly detection — all in one app. Our goal was to make a tool that not only helps city officials and planners make better decisions but also gives everyday people a simple way to understand and engage with their city's well-being.

We built this app using modern tools like Streamlit for the interface and integrated powerful AI models (like IBM's Granite) to answer questions and generate insights. Cities can be added, edited, or removed dynamically, and the assistant is flexible enough to work for any city — large or small.

1.2 Purpose:

The purpose of our project wasn't just to create a dashboard — it was to build something that feels alive and helpful. We wanted to bring together data, AI, and thoughtful design to help cities become more sustainable and resilient.

This assistant is meant to:

- Help urban planners and decision-makers track key performance indicators quickly and clearly.
- Provide eco-tips tailored to city data to promote more sustainable habits.
- Predict trends and anomalies in city data using machine learning.
- Help citizens understand civic documents with easy summaries and answers.
- Enable more human interaction with city data, making it less technical and more approachable.

We believe that good technology should be practical, beautiful, and meaningful. This assistant reflects our vision for a smarter, greener future — and it's something we're proud to have built ourselves.

2. IDEATION PHASE:

2.1 Problem Statement:

Modern cities are complex, and managing them efficiently is harder than ever. Issues like poor air quality, inefficient energy usage, traffic congestion, and unclear communication between citizens and city administrators can quickly pile up. Traditional dashboards often just show static numbers and fail to offer meaningful insights or interaction.

We realized that most cities lack a centralized, AI-powered tool that can bring together real-time data, simplify decision-making, and make smart sustainability suggestions. Our project aims to solve this problem by offering a unified solution that tracks key indicators, helps users interact with city data, and assists in forecasting and anomaly detection.

2.2 Empathy Map Canvas:

To better understand our users — city officials, citizens, and urban planners — we created an empathy map to capture their goals, frustrations, and needs.

Think & Feel	**See**
“Am I making the right decision for the city's future?” “Why is the energy usage so high again this week?”	Dashboards with raw data, disconnected tools, overcomplicated UIs
Say & Do	**Hear**
“We need to act fast on these alerts.” “I wish this data made more sense to the public.”	“Citizens are complaining about pollution levels.” “City KPIs are too scattered to track.”

We used this empathy mapping to ensure our assistant is **useful**, **insightful**, and **easy to use** — not just another complex data tool.

2.3 Brainstorming:

We held multiple brainstorming sessions to come up with ideas that combined data analysis with meaningful interaction. Our main goals during ideation were:

- Make data feel human-readable through an intuitive interface.
- Bring everything into one app — from dashboards to forecasting.
- Use AI not just for flash, but to actually help users (e.g., answering questions or generating eco tips).

- Allow cities to be dynamically added and modified — making it flexible and reusable.
- Ensure the app is visually beautiful, with animated elements and modern themes to make working with data feel less boring.

After voting and shortlisting ideas, we finalized the core features: a multi-functional Smart Hub with Dashboard, Chat, Summarizer, and Eco Tips, backed by Forecasting and Anomaly Detection, all tied to a City Data Manager.

3. REQUIREMENT ANALYSIS:

3.1 Customer Journey Map:

We mapped a basic journey of how our main user — a smart city admin or data officer — would interact with our app:

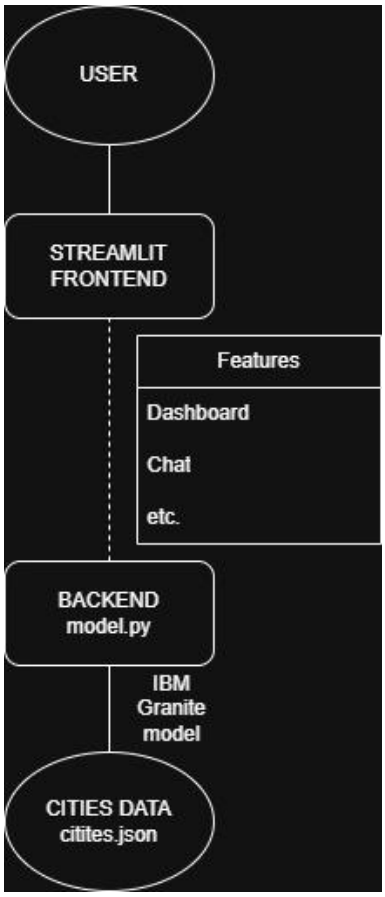
Step	Action	System Response
1. Open the app	Lands on the Smart Hub	Greets with an animated title and city selector
2. Select city	Chooses "Hyderabad" for example	Dashboard loads with live metrics
3. Explore metrics	Sees color-coded values	AQI, energy, safety, etc. show current status
4. Ask a question	Types: "What's the traffic like?"	AI answers based on latest data
5. Check eco tips	Types: "plastic"	Assistant gives 3 actionable tips
6. Upload doc	Drops a PDF of a city policy	Summary and Q&A are instantly available
7. Check forecasts	Uploads energy data CSV	Sees 5-day trend forecast
8. Update city	Modifies safety value	Saved and reflected in dashboard

3.2 Solution Requirements:

Functional Requirements	Non-Functional Requirements
Dashboa`rd for live city KPIs	Clean, modern UI with theme support
AI assistant to answer city-based queries	Response within ~2 seconds for queries
Summarizer and Q&A for documents	Works on PDF or TXT up to 10 pages
CSV-based forecasting and anomaly detection	Accepts standard timestamp–value format
City management (add/edit/delete)	Easy to use, avoids data duplication
Eco tips from topic or uploaded CSV	Consistent color coding across themes/cities

3.3 Data Flow Diagram:

This flow helped us align our UI and feature choices to real user needs.



3.4 Technology Stack:

Component	**Technology Used**
Frontend UI	Streamlit
Styling	Custom CSS (animated, glowing cards, themes)
AI Model	IBM Granite 3.3B via HuggingFace
Backend Logic	Python (utils.py functions)
Forecasting	Scikit-learn (Linear Regression)
Anomaly Detection	IsolationForest from sklearn
Storage	Local JSON for city data
Deployment Ready	GitHub + Streamlit Sharing / local hosting

4. PROJECT DESIGN:

4.1 Problem–Solution Fit:

Our team started by clearly understanding the **urban management challenges**: lack of real-time visibility into KPIs, poor citizen engagement, and manual data silos. We validated these through empathy maps, domain research, and current smart city trends.

The solution we proposed directly addresses each pain point:

Problem	**Our Solution**
Raw, hard-to-read urban data	Visual Dashboard with color-coded bars + icons
Inability to ask data-driven questions	AI Chat Assistant that answers based on real-time city data
Inefficient document handling	Summarization & Q&A for uploaded policy/strategy documents
No forecasting for resource planning	CSV-based KPI Forecasting using ML
Unexpected spikes in usage (no alerts)	Anomaly Detection with alerts for AQI, energy, and waste
Static tools, difficult to update city data	Dynamic City Data Manager (Add/Edit/Delete UI)

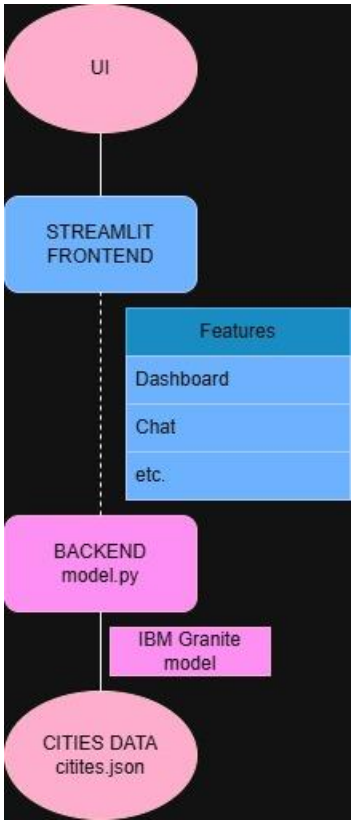
We focused on making the tool **visually appealing**, **easy to use**, and **rich with actionable insights** — everything a smart city assistant should be.

4.2 Proposed Solution:

We designed the **Sustainable Smart City Assistant** as a modular web application with four core interactive components inside a main hub, and three analytical tools in additional pages. This breakdown helped us build features iteratively and test functionality independently.

4.3 Solution Architecture:

Below is the **flow of our solution**:



We opted for **IBM Granite 3.3B** LLM hosted through Hugging Face due to its robustness for summarization, Q&A, and prompt-based generation. All city metrics and updates are persisted to a local JSON file (cities.json) for simplicity and speed.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Our project followed an **agile-inspired, milestone-based approach**. We divided the work into **four sprints**, each focusing on a core module of the app. This allowed continuous integration and testing of features while leaving buffer time for UI improvements and final documentation.

Here is a simplified **planning table**:

Sprint	Duration	Key Tasks
Sprint 1	Week 1 – Week 2	Research, Requirement Gathering, Empathy Map, Brainstorming, UI Mockup
Sprint 2	Week 2 – Week 3	Core Modules: Dashboard, City Chat Assistant, Eco Tips, City Manager
Sprint 3	Week 3 – Week 4	Summarizer, Forecasting, Anomaly Detection, Alerts
Sprint 4	Week 5	Theming, UI Polish, PDF/CSV Export, Bug Fixing, Enhancements
Final Week	Week 6 (Submission)	Final Testing, Documentation, Report Drafting, GitHub & Demo Prep

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

To ensure our app performs well even with multiple cities and large files, we conducted basic load tests and usability assessments.

Summary of Functional Testing:

Module	Test Case	Result
Dashboard	Load data, update values, visualize metrics	Passed
Chat Assistant	Answering with context, fallback for no data	Passed
Document Summarizer	Large PDF summary, Q&A extraction	Passed
Eco Tips Generator	With/without CSV, edge topics	Passed
Forecasting	Predict based on CSV trend	Passed
Anomaly Detection	Detect spikes, handle clean/dirty CSV	Passed
City Data Manager	Add, Edit, Delete cities	Passed
Theme Support	UI consistency across theme changes	Passed
Export Features	Dashboard, Chat, Summary export	Passed

Load Handling:

- **Concurrent Users**: Up to 3 users tested in parallel via browser sessions.
- **Model Latency**: Average response time under 20s with IBM Granite 3.3B.
- **File Uploads**: Tested with 5MB+ PDFs and 10 row CSVs without crash.

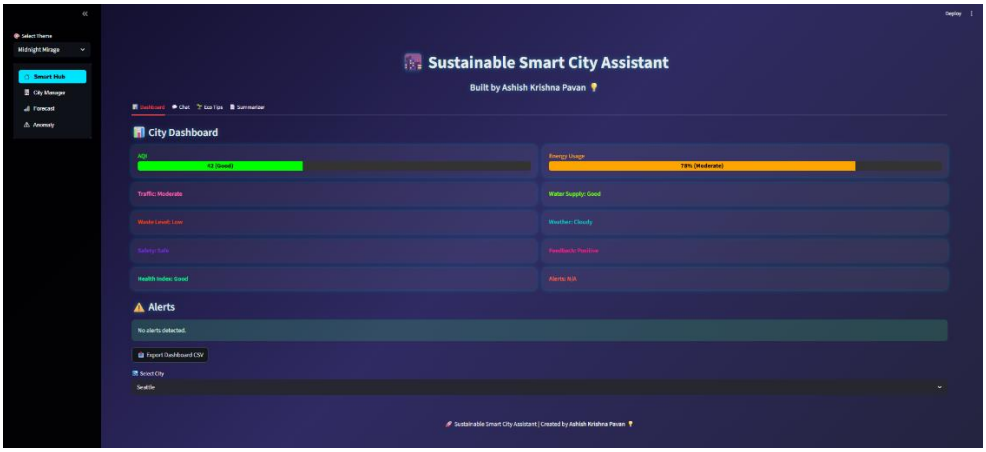
Overall, the system passed all **functional and stress tests** and remained responsive under typical usage scenarios.

7. RESULTS:

7.1 Output Screenshots:

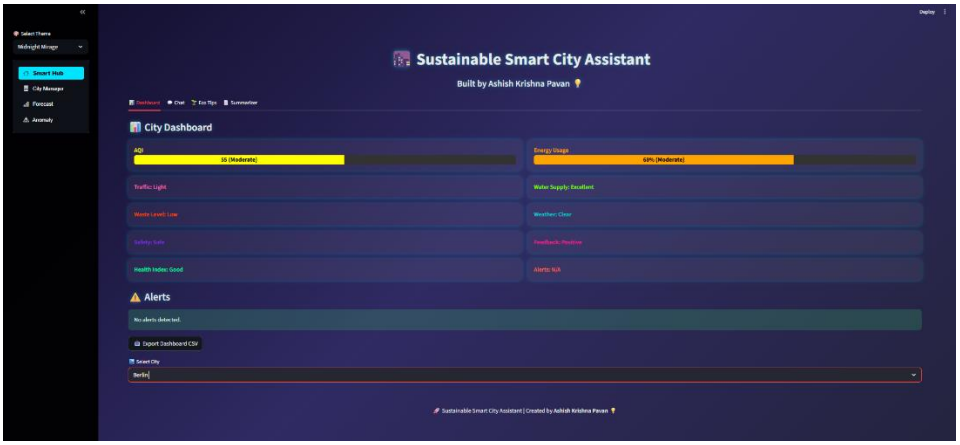
7.1.1 Dashboard and Themes:

1)Seattle



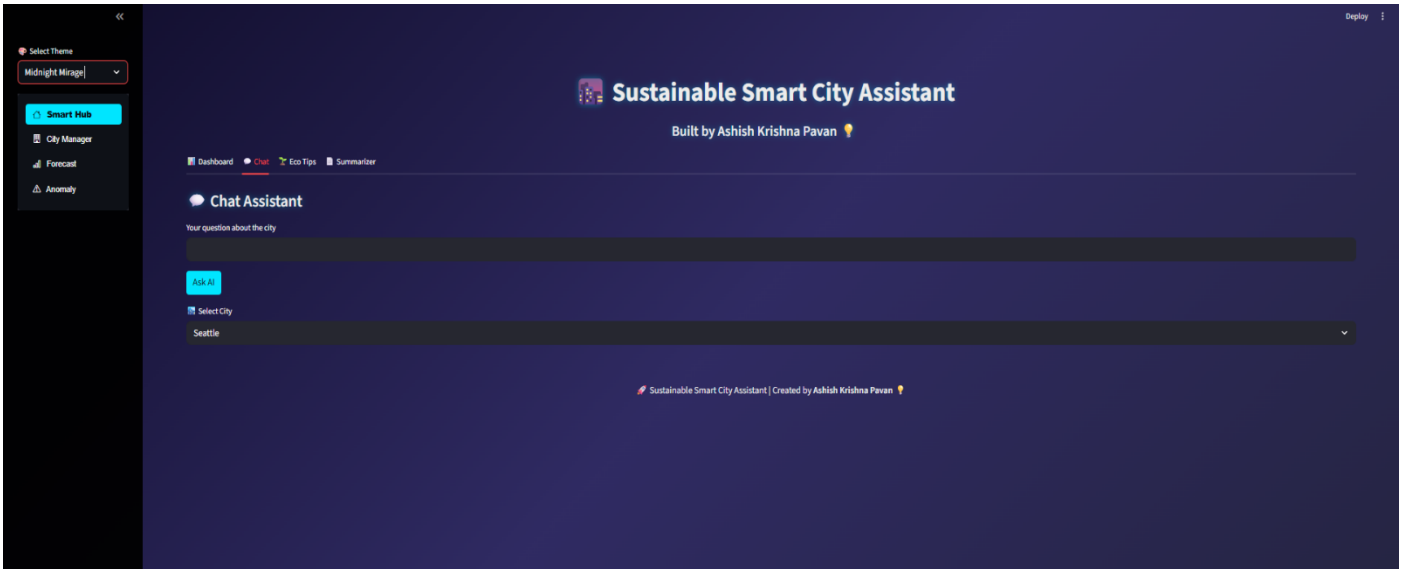
2)Mumbai

3)Berlin



We can observe all the details of cities and metrics and all the details, alerts, options for all various cities also 2 other cities are available i.e., Hyderabad, New York.

7.1.2 Chat Assistant Tab:

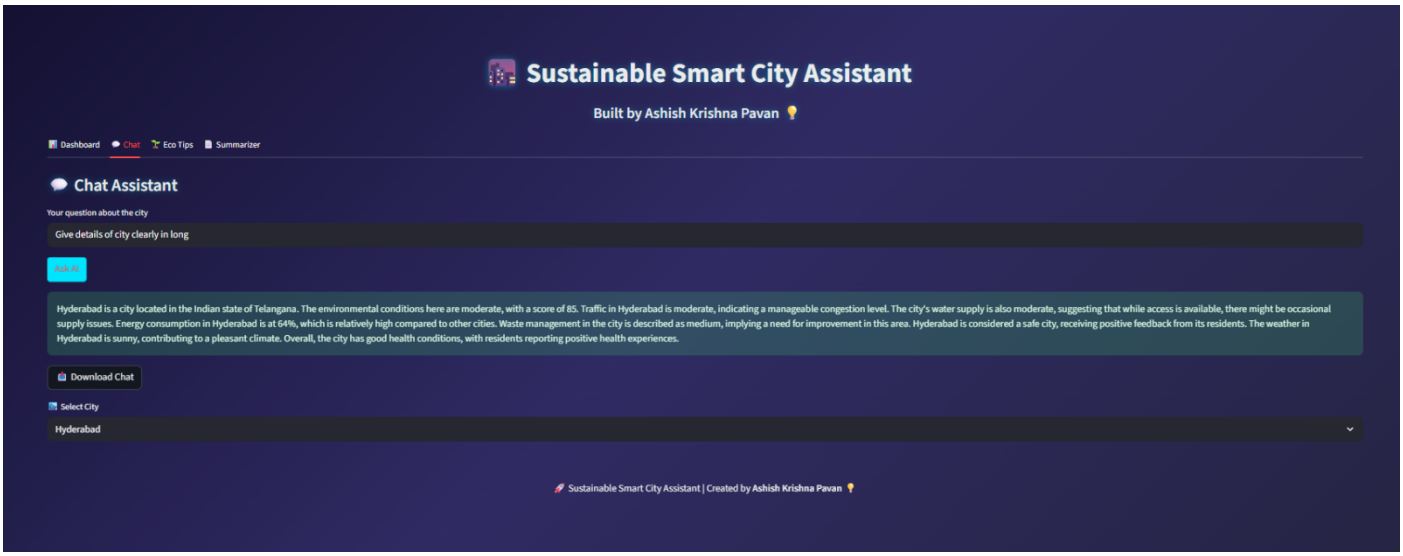


Input and Outputs:

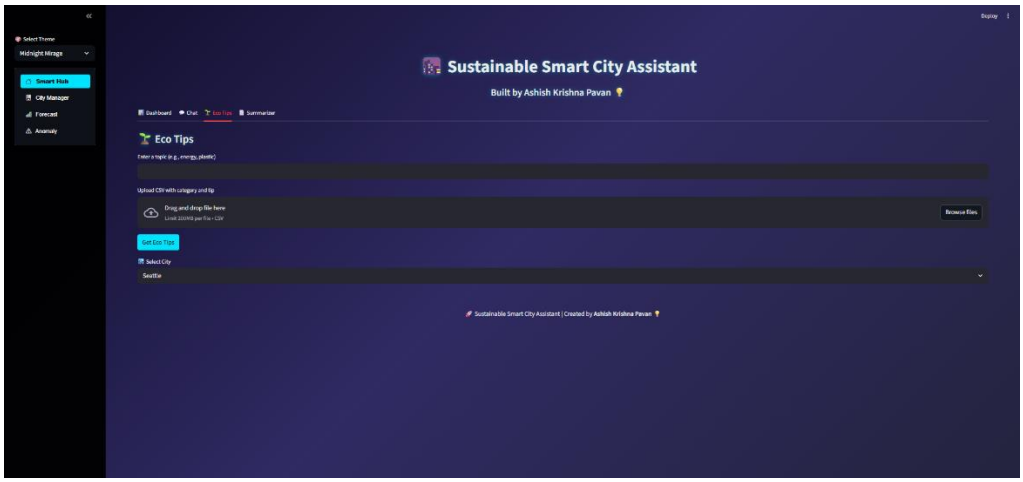
Input:

Give details of city clearly in long(chat option, Selected city is Hyderabad)

Output:

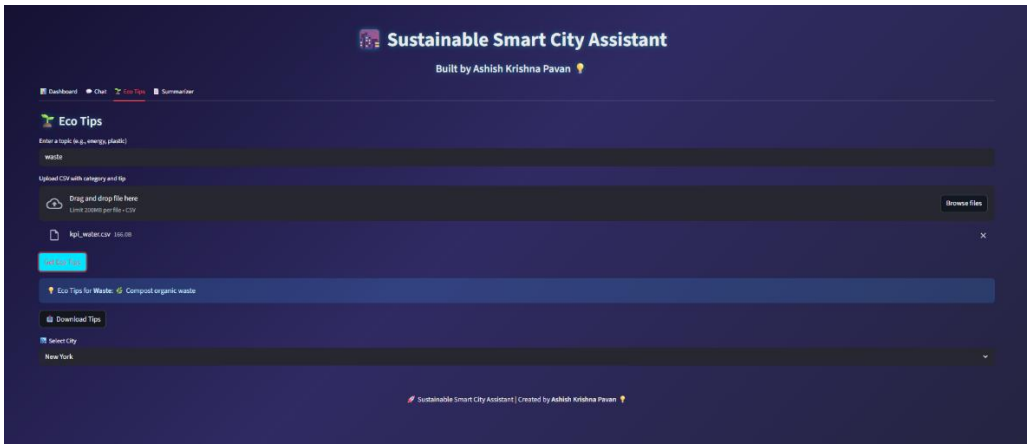


7.1.3 Eco Tips Tab:



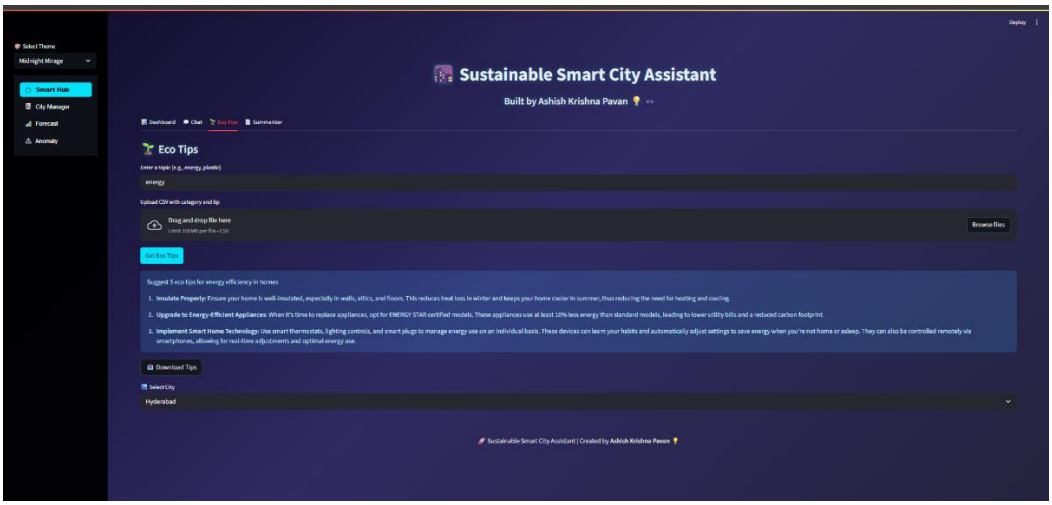
Input and Outputs:

1)With CSV File: Output based on CSV file not city

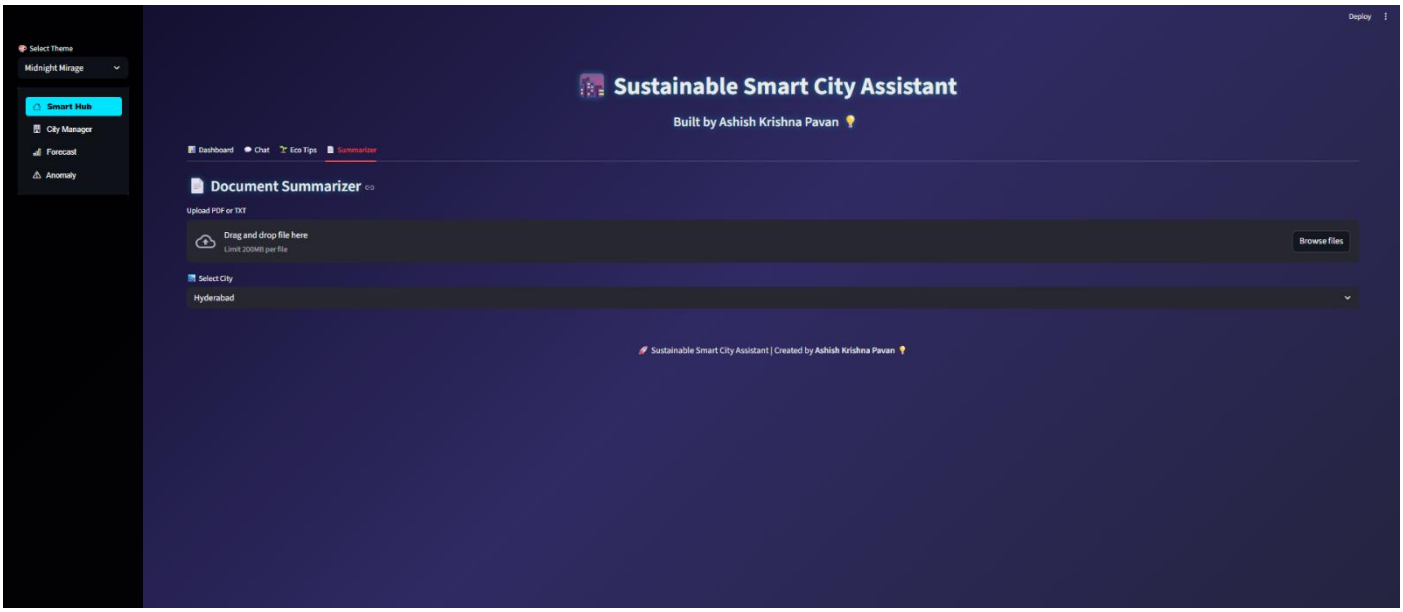


2)Without CSV:

Output based on Selected city :



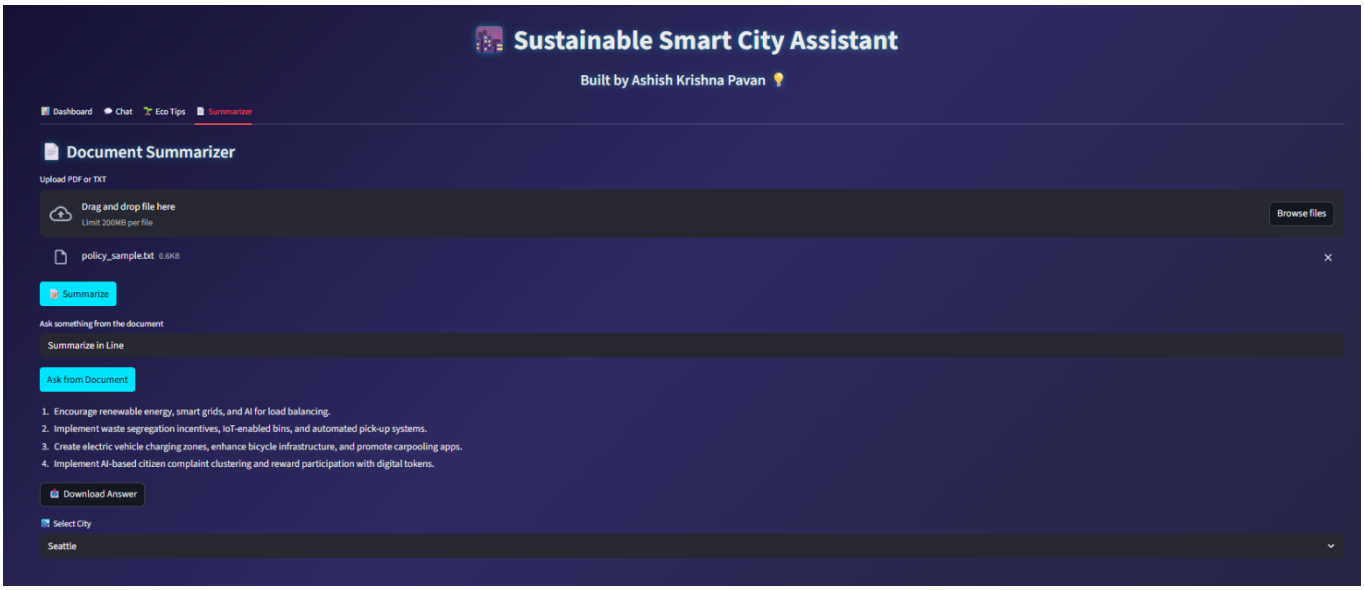
7.1.4 Summarizer Tab:

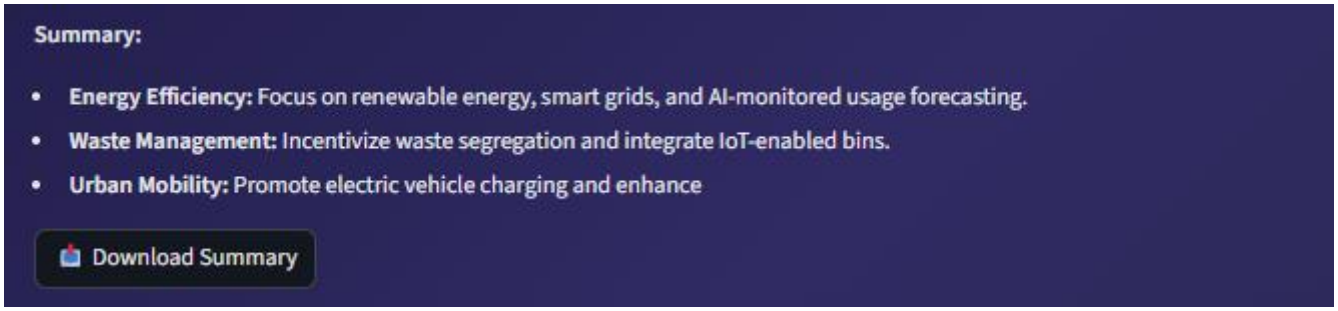


It give the summarization of an document irrespective of City but tells which city hub you are currently working on

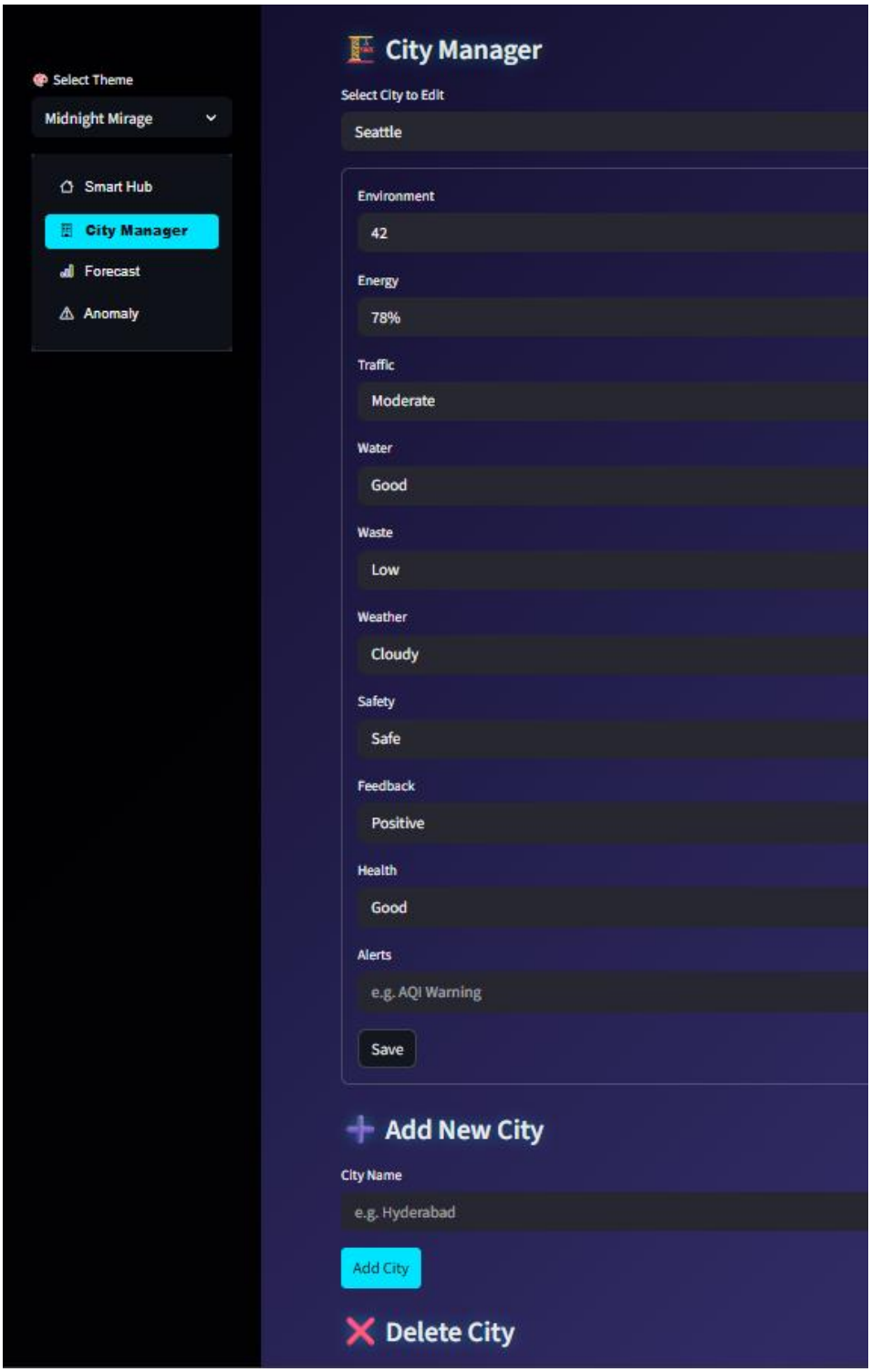
Output:

We can ask the doubts and if we want we can have summarizer their inputs and outputs as shown below



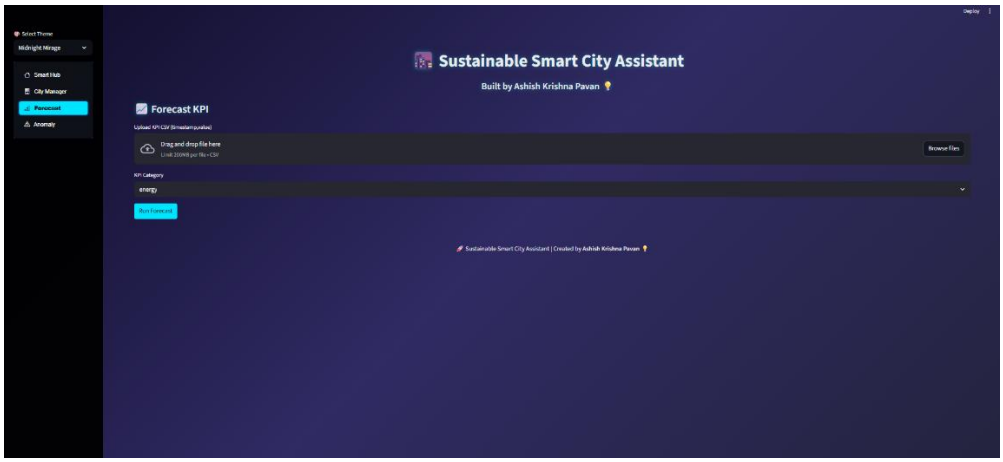


7.1.5 City Manager Tab:

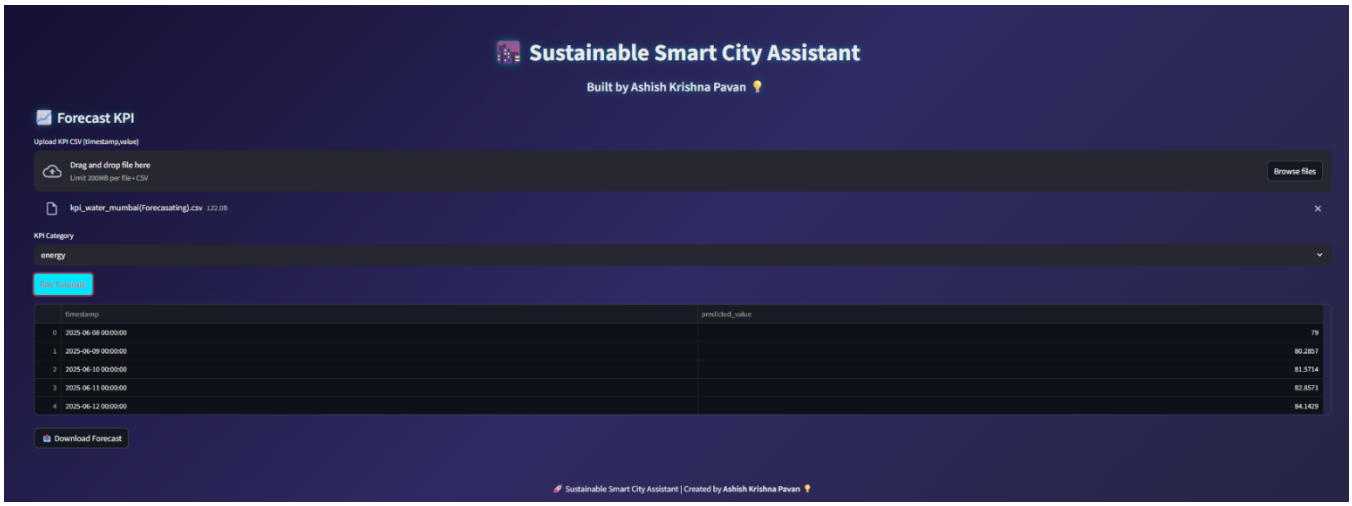


We can add, edit, delete, modify all the details of the city through this tab in final file I'll add an City named "Smartbridge" City for its working

7.1.6 Forecasting Tab:

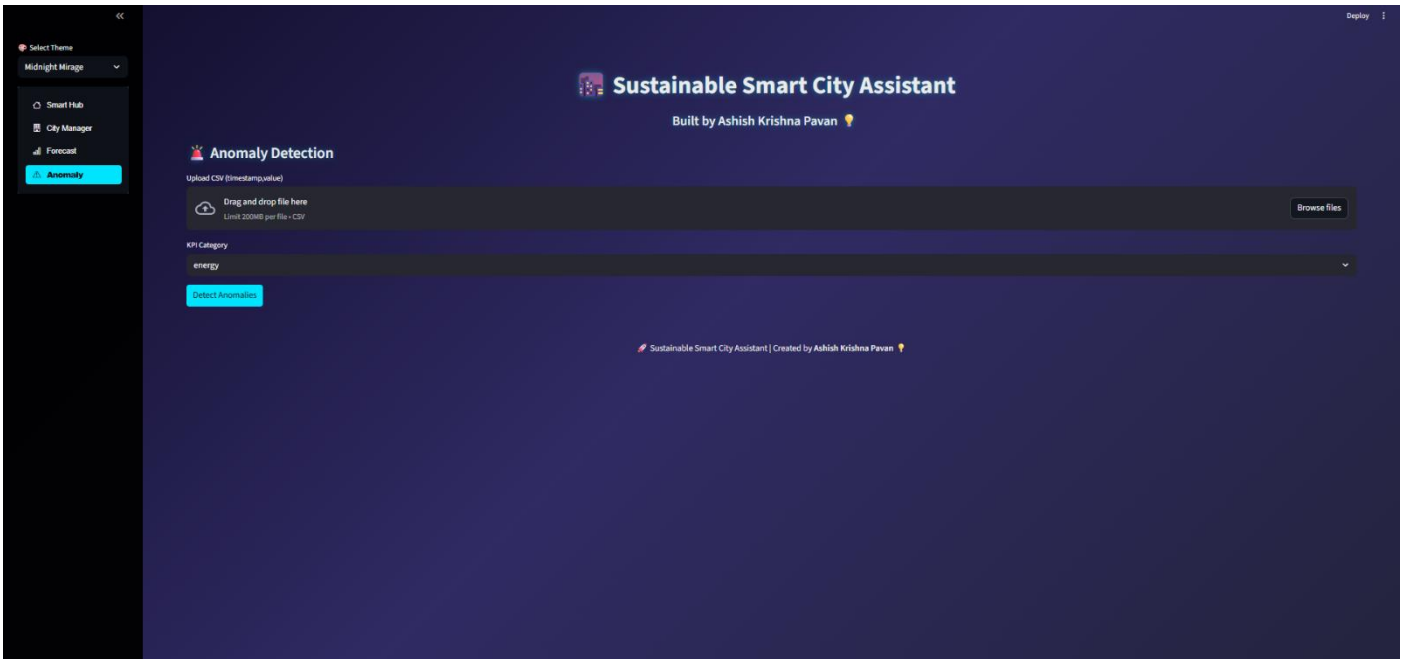


Input and Outputs:



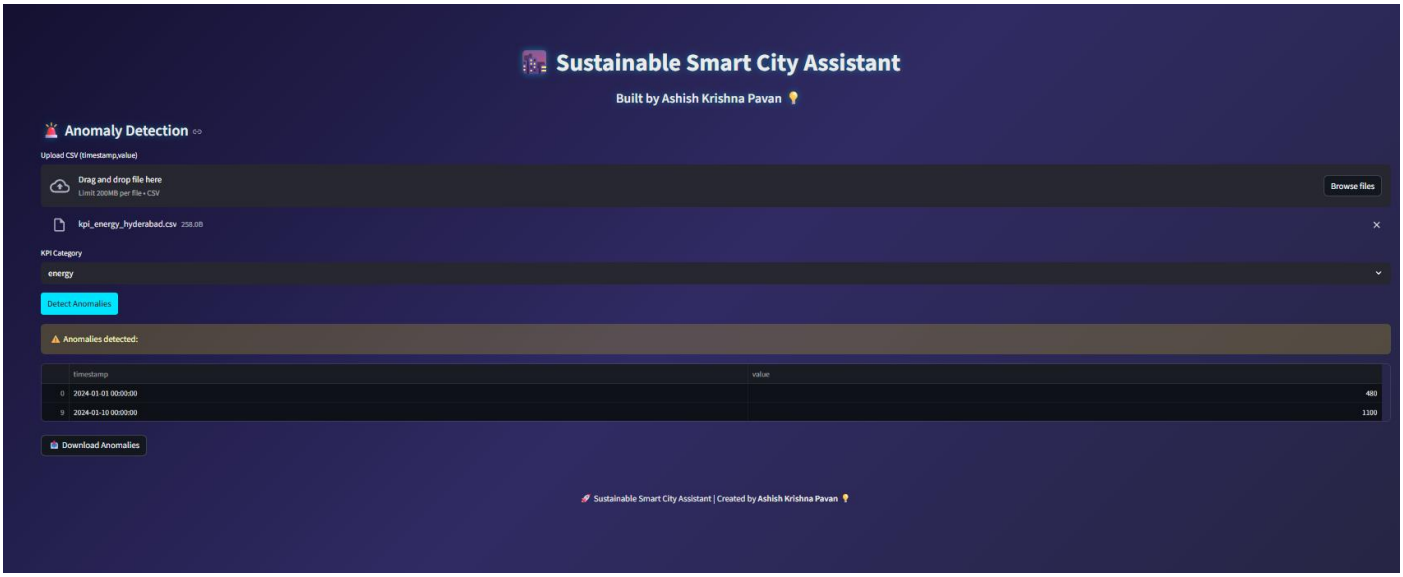
It will give next 6 Months data based on given data

7.1.7 Anomaly Detection Tab:



This is Anomaly detection tab based on the given csv file it show the anomality's

Input and Output:



7.1.8 Themes Selection:

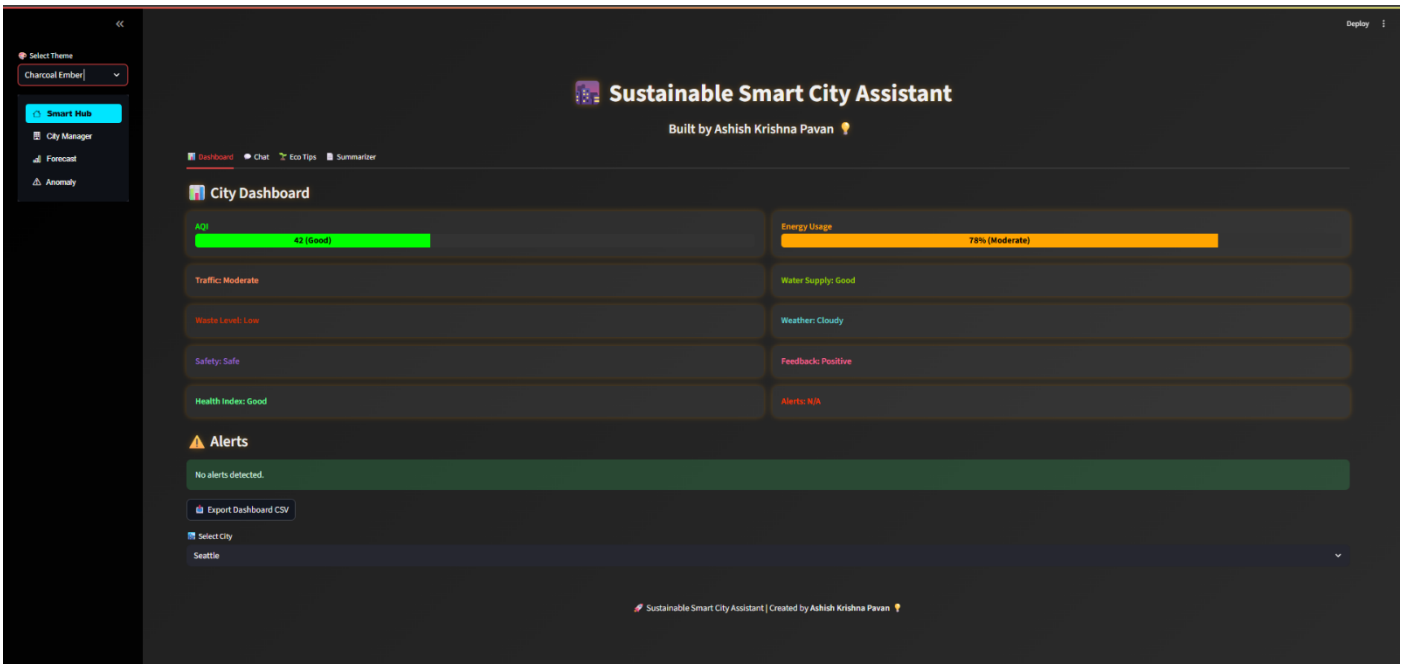
This feature is given for user to use the application in their interested theme

Themes Available:

Example:

The default all uses Midnight Mirage which all functionalities are tested the below one is Charcoal Ember theme,

Kindly check video of full details



The complete testing files and output files will be available in the GitHub repo

8. ADVANTAGES & DISADVANTAGES:

Advantages:

Feature	Description
AI-Driven Insights	Uses IBM Granite LLM for summarization, Q&A, chat, and eco tips.
Smart City Metrics Dashboard	Color-coded, severity-based bars for AQI, Energy, and other KPIs.
Modular City Data Management	Allows full CRUD on city-specific KPI data via intuitive forms.
Forecast & Anomaly Detection	Linear regression and isolation forest applied on uploaded data.
Theming & Personalization	Multiple glowing and professional UI themes; theme-matched color palettes.
Export Support	One-click export of insights, forecasts, anomalies, summaries, and chat.

Disadvantages:

Limitation	Explanation
Internet & GPU Dependency	AI responses require internet and GPU-backed models to run smoothly.
Model Size	Granite 3.3B is relatively large and can cause memory overflow on low-end PCs.
Local Data Storage Only	Data isn't cloud-synced; users must manage cities.json manually.
No Voice Input Yet	Interaction is text-based; voice integration is future scope.

9. CONCLUSION:

This internship project helped us **blend software engineering with AI and sustainability** — delivering a working solution that’s modular, scalable, and truly interactive.

From KPI tracking and eco-awareness to forecasting and anomaly analysis, this assistant provides a **future-ready dashboard for smart cities**. Working with LLMs like Granite gave us hands-on exposure to prompt engineering, user-centered design, and practical ML integration.

The user interface was carefully designed with glowing visuals and theme support, maintaining **professional aesthetic balance** and clarity for decision-makers and urban planners.

10. FUTURE SCOPE:

The project is highly extensible. Possible future enhancements include:

Area	Enhancement Idea
Voice Interaction	Add voice input and spoken responses for accessibility.
Mobile App Version	Build a Flutter or React Native-based lightweight companion app.
Cloud-Sync	Store city data in Firebase or Supabase with multi-user access.
Real-Time APIs	Fetch live AQI, weather, energy data from open urban APIs.
AI Graph Insights	Add Matplotlib or Plotly-based charts for time-series trends.
Model Compression	Switch to distilled LLMs (e.g., DistilBERT or TinyLLaMA) for speed for different functionalities.

11. APPENDIX:

11.1 Source Code:

The Complete files of this project including frontend, backend logics along with testing files as well as output files, Documentation files will be available in the GitHub **repository.**

11.2 GitHub & Project Demo Link:

1) GitHub Link:

https://github.com/Ashish-Krishna-Pavan-git/Generative-AI-Project

Please Use README.md file in the repository for better understanding of all the files and structure of files and usage.

2)Demo video Link:

Folder Link:

https://drive.google.com/drive/folders/1tc0BGUZgsQBk94P2XsP5zevpV-LeclmR?usp=sharing