SMARTBRIDGE

Generative AI with IBM Cloud

Sustainable Smart City Assistant
Using IBM Granite LLM

Date	May 2025 – June 2025
Team ID	LTVIP2025TMID60743
Project Name	Sustainable Smart City Assistant Using IBM Granite LLM

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

Submitted by:

NAME and Team

Team Details:

ID - LTVIP2025TMID60743

Team Member - 1: 3Chintala Saiteja

Team Member - 2 : Boddula Solomon

Team Member -3: Chintalapudi Likhith Sri Subramanyam

Team Member -4:Daliparthi Veeramanikanta

Project Report Format

1. INTRODUCTION

1.1 Project Overview

Smart cities are modern urban areas that use technology to enhance the quality of life for their citizens, optimize resource usage, and ensure sustainable development. With rapid urbanization and rising environmental concerns, cities around the world are under pressure to reduce pollution, conserve energy, and improve the efficiency of public services. In this context, artificial intelligence (AI) plays a vital role in transforming traditional city operations into intelligent systems.

The **Sustainable Smart City Assistant** is designed to help citizens and city administrators access eco-friendly tips, real-time sustainability updates, and decision-making support based on city data. Powered by **IBM Granite LLM (Large Language Model)**, the assistant uses natural language processing to understand queries and provide relevant, personalized, and meaningful suggestions for daily green living. Whether it's air quality, water consumption, traffic congestion, or waste management, the assistant provides useful insights in a simple, conversational format.

This project aims to bridge the gap between complex environmental data and everyday users. By using AI responsibly, we can improve environmental awareness, support smart behaviors, and build a community-driven approach to urban sustainability. The assistant can be integrated into websites, mobile apps, or kiosks and can be scaled for use in various cities and languages.

1.2 Purpose

The main purpose of the **Sustainable Smart City Assistant** is to simplify access to sustainability-related information for urban residents. Environmental data is often scattered, complex, or difficult to interpret for a non-technical audience. The assistant acts as a single point of contact for users who want to:

- Understand their city’s environmental health (air, water, energy, traffic).
- Get actionable tips on reducing their carbon footprint.
- Make informed decisions about their lifestyle and daily routines.
- Receive alerts or suggestions based on real-time city conditions.
- Engage with local eco-initiatives and report sustainability issues.

This project supports the broader goal of creating smart and sustainable communities, where technology supports the well-being of people and the planet.

1.3 Scope

The scope of the assistant includes the following functionalities:

- A conversational AI interface that can understand and respond to natural language queries related to city sustainability.
- Integration with external APIs to fetch real-time data on weather, pollution levels, public transport, etc.
- A dynamic dashboard that displays eco-tips, city health status, and personal sustainability scores.

- A feedback system where users can submit issues, suggestions, or ideas for improving city living.
- Scalable design to support multiple cities, languages, and user types (citizens, administrators, students, etc.).

The project does not currently support offline mode or voice-based interaction, but these features can be considered for future versions.

2. IDEATION PHASE

2.1 Problem Statement

As cities grow, the need for sustainable living becomes more urgent. Citizens and urban planners need quick access to accurate, real-time environmental data to make smart choices in areas like energy use, transportation, air quality, and waste management. However, this data is often scattered, difficult to understand, or not accessible in a user-friendly format.

Problem Statement:

Urban residents and city planners lack a centralized, intelligent system to access real-time, personalized, and actionable sustainability data. This makes it difficult to make eco-friendly decisions, engage in green practices, or improve city living conditions efficiently.

The problem affects:

- **Citizens**, who want to live greener lives but lack the right tools and insights.
- **Government planners**, who want to promote sustainable initiatives but struggle to communicate them effectively.
- **The environment**, which suffers from continued inefficiencies and unawareness.

2.2 Empathy Map Canvas

The **Empathy Map** helps us understand the end-users by visualizing what they think, feel, say, and do.

Thinks	Feels	Says	Does
"Is the air clean today?" "How can I reduce electricity use?"	Worried about pollution Frustrated by unclear information	"We need smarter ways to stay eco-friendly."	Checks multiple apps for air and traffic updates Watches news/weather reports

Explanation:

- Users want reliable and simple information, not complicated reports.
- They care about the environment but need guidance.
- They're motivated by convenience, health, and social responsibility.

2.3 Brainstorming

During the ideation phase, we considered several potential solutions before finalizing the Sustainable Smart City Assistant. Below are some ideas explored:

Idea	Description	Reason Rejected/Shortlisted
Eco Score Tracker	Mobile app that calculates daily eco score based on habits	Good idea, but limited to individuals only
Pollution Alert System	Sends SMS alerts about nearby pollution spikes	Useful, but not interactive
Smart City Dashboard	Website showing live city data and sustainability news	Informative, but lacks personalization
AI-Based Eco Assistant	Chatbot that provides tips, alerts, and insights	✅Final idea – combines data, personalization, and AI
Water Usage Monitor	IoT device for water meters	Expensive and hardware-based
Energy Saver App	Suggests ways to reduce electricity at home	Useful but narrow focus
Carbon Calculator	Calculates footprint based on travel and use	Lacks real-time city context
Smart Complaint Portal	Citizens report waste, pollution, etc.	Could be added as a feature, not a core tool

---

2.4 Literature Survey

To support our idea, we explored existing tools and research papers related to smart city sustainability and AI assistance:

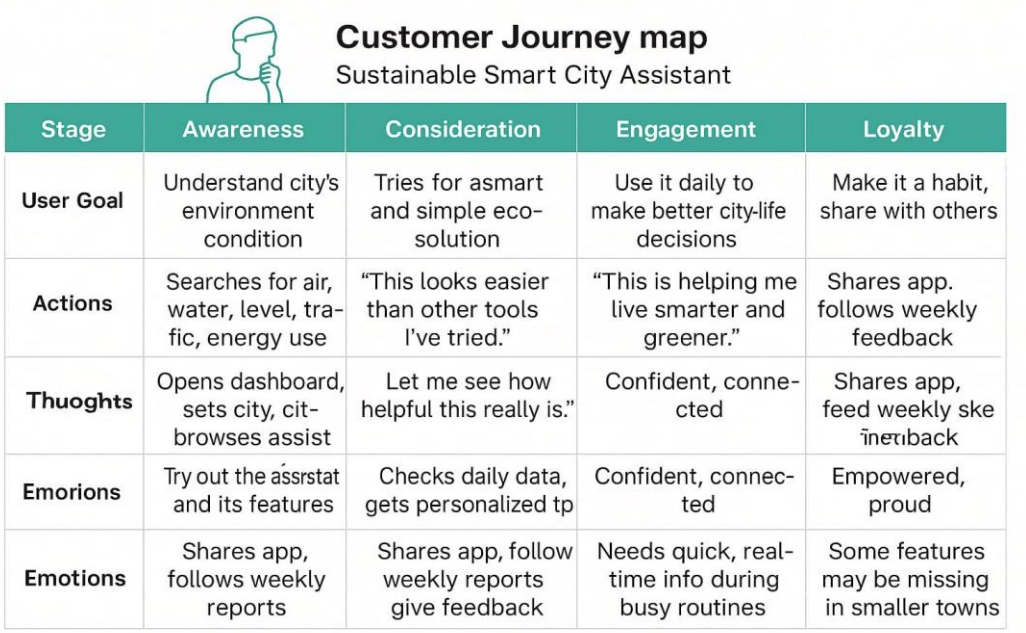
1. **GreenAssistant App (2020)**
   A mobile app that offers carbon tracking and habit suggestions. Lacked real-time integration and AI-based conversation.
2. **IBM Watson for Smart Cities**
   IBM offers smart city platforms, but they are mostly data-centric and not citizen-friendly interfaces. We use **IBM Granite LLM** for natural, human-like responses.
3. **Paper: AI for Urban Living (IEEE 2022)**
   Emphasizes the use of AI to reduce energy waste in buildings. Supports the idea that AI can improve urban planning.
4. **Public APIs (AirVisual, OpenWeather)**
   These APIs are used for pollution and climate data. However, without a smart interface, their usability is limited.

Conclusion:
There is a clear opportunity to bridge the gap between raw environmental data and user understanding. Our assistant uses a conversational model (LLM) to make complex sustainability data easy to access and act on.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

The **Customer Journey Map** outlines the stages users go through when interacting with the Sustainable Smart City Assistant. It helps us understand their goals, feelings, and needs at each step.



Stage	Goal	Actions	Thoughts	Emotions	Pain Points
Awareness	Learn about smart living tools	Sees ad or hears from friends	"Could this help me live greener?"	Curious, hopeful	Hard to find trustworthy sources
Consideration	Try a helpful eco-assistant	Opens the site or app	"Is this easy to use?"	Interested, skeptical	Other apps are hard or boring
Onboarding	Use the assistant	Registers and starts asking questions	"Let's see if this really helps."	Motivated, cautious	Needs a smooth first-use experience
Engagement	Use it for daily tips and insights	Gets alerts, views dashboard	"This is really improving my habits."	Confident, involved	Needs relevant and updated tips
Loyalty	Keep using it, recommend to others	Shares app, gives feedback	"More people should use this!"	Happy, responsible	May expect new features over time

3.2 Solution Requirements

These are the core features the system must support to fulfill its purpose.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form, Gmail, LinkedIn
FR-2	User Confirmation	Email and OTP confirmation
FR-3	Login System	Email login, social login, password reset
FR-4	Dashboard Features	View tips, sustainability score, feedback form
FR-5	API Integration	Pollution, weather, traffic data APIs
FR-6	Personalized Suggestions	Use IBM Granite LLM to give tailored responses
FR-7	Feedback Module	Let users submit suggestions or issues

These define how the system should behave rather than what it should do.

NFR No.	Non-Functional Requirement	Description
NFR-1	Usability	Easy and clean UI/UX for all user types
NFR-2	Security	Data encryption, OAuth login, secure API access
NFR-3	Reliability	System should respond correctly under all conditions
NFR-4	Performance	Fast responses (< 3s), even under load
NFR-5	Availability	99% uptime on cloud infrastructure
NFR-6	Scalability	Easily add new cities or users without affecting performance

---

3.3 Data Flow Diagram (DFD)

The **DFD** shows how data moves through the system. This helps visualize components and their interaction.

Explanation:

- **User** sends a request (e.g., “How clean is the air?”).
- **Frontend** collects and passes it to backend.
- **Backend** interacts with IBM Granite LLM and APIs.
- **External APIs** send real-time data (pollution, traffic).
- **LLM** formats the response and sends it back.
- **User** receives a personalized reply and updated dashboard.

3.4 Technology Stack

Below is the list of tools, platforms, and services used in the project.

Component	Description	Technology
User Interface	How user interacts with app (web/mobile/chat)	HTML, CSS, JS, Streamlit, React.js
Application Logic	Logic for handling requests	Python
AI Assistant	Language model for smart replies	IBM Granite LLM (via Hugging Face)
Backend API Integration	Connects to external real-time data	IBM Weather API, Aadhar API (optional)
Database	Stores user and dashboard data	IBM DB2, Cloudant, SQLite (dev)
File Storage	Stores uploaded reports or files	IBM Cloud Storage / Local file system
Hosting Infrastructure	Runs the entire system	IBM Cloud / Cloud Foundry / Kubernetes
Security	Protects data and access	SHA-256, SSL/TLS, OAuth 2.0

4. PROJECT DESIGN

4.1 Problem–Solution Fit

The **Problem–Solution Fit** identifies whether the solution we've built actually addresses the user's real-life problems in a meaningful way. For our project, users—both citizens and city planners—struggle with accessing simple, real-time sustainability information. Our assistant solves this by delivering personalized, AI-powered insights.

Problem Faced	Solution Provided by the Assistant
Scattered or complex eco data	A centralized, AI-based conversational assistant
Difficulty in interpreting pollution/weather metrics	Converts raw data into simple tips and messages
No personalized guidance for sustainable actions	Uses IBM Granite LLM to provide suggestions tailored to user queries
Lack of engagement in green practices	Sends daily eco tips and reminders via dashboard
Limited public awareness about sustainable living	Educates users interactively through chat and notifications

Conclusion:
By aligning user behavior with AI-driven suggestions, the solution fits well into existing habits and improves awareness, decision-making, and sustainable choices.

4.2 Proposed Solution

The **Sustainable Smart City Assistant** is a web/mobile-based platform that helps citizens and planners make better environmental decisions using AI. It provides a chatbot-like experience powered by IBM Granite LLM, and connects to real-time public data sources such as weather, pollution, and traffic APIs.

Key Features:

- **Conversational Assistant:** Users can type questions like "Is it safe to jog today?" or "How's the air quality?" and receive AI responses.
- **Personalized Eco Tips:** Based on location and date, users receive green living suggestions.
- **Sustainability Score:** A points-based system to show how eco-friendly the user's choices are over time.
- **Feedback & Complaints:** Users can submit issues related to city services like garbage, traffic, etc.

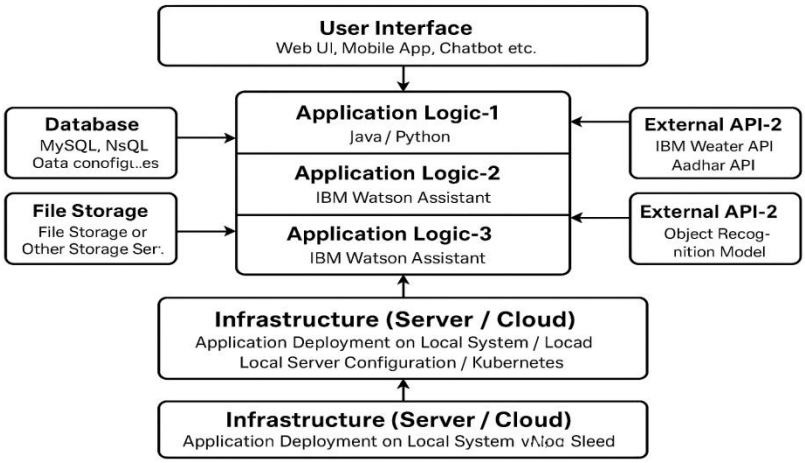
- **Alerts & Notifications:** System sends alerts for air quality warnings or sustainable city campaigns.

How It Works:

1. User enters a query or clicks a dashboard button.
2. The app fetches real-time data and sends it to IBM Granite LLM.
3. The LLM generates a clear, friendly response.
4. The app displays the response and logs the interaction.

This solution is citizen-friendly, scalable, and designed for real-world smart cities.

4.3 Solution Architecture



Architecture Layers:

1. **User Interface (UI Layer)**
   - Web/mobile frontend (React/HTML/Gradio)
   - Collects input, shows output, dashboards
2. **Application Layer (Backend)**
   - Built with Python
   - Handles user logic, session tracking, scoring
3. **LLM Integration Layer**
   - Communicates with IBM Granite LLM on Hugging Face
   - Sends queries, gets smart answers
4. **External APIs**
   - Weather (IBM Weather), Pollution, Traffic
   - Feeds real-time data to the assistant
5. **Database Layer**
   - Stores user profiles, history, tips, feedback
   - Technologies: IBM DB2 or Cloudant
6. **Cloud Infrastructure**
   - Deployed on IBM Cloud or local testing via Streamlit
   - Load balancing, API routing

4.4 Components & Technology

Component	Description	Technology Used
UI Layer	Interface to chat, view scores and alerts	Streamlit / HTML / CSS / React.js
Backend Logic	Logic to handle requests, process data	Python
AI Model Integration	Converts user input to natural replies	IBM Granite LLM (Hugging Face)
Weather & Pollution API	Provides real-time environment data	IBM Weather API
Database	Stores all data from users and dashboard	IBM Cloudant / SQLite / DB2
File Storage	Stores uploaded documents or reports	IBM Block Storage or local
Hosting & Scaling	Runs the app on cloud infrastructure	IBM Cloud, Kubernetes (future), Cloud Foundry
Security Layer	Handles login, encryption, and user data protection	OAuth 2.0, SHA-256, JWT, IAM Roles

4.5 Security & Scalability

Security and scalability are essential for any modern cloud-based application, especially when it handles user data and real-time analytics.

Security Measures:

- **OAuth 2.0** for social login
- **JWT tokens** for session management
- **SHA-256 encryption** for stored data
- **Secure API Calls** using HTTPS (SSL/TLS)

Scalability:

- Built on **3-tier architecture** for easy separation of concerns
- Can be deployed on **Kubernetes clusters** for microservices

- Supports vertical and horizontal scaling
- Stateless backend allows auto-scaling under load

5. PROJECT PLANNING & SCHEDULING

5.1 Product Backlog

The product backlog contains the functional requirements (epics), broken down into user stories with priorities, story points, and assignments across sprints.

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Member
Sprint-1	Registration	USN-1	As a user, I can register with email/password and confirm password	2	High	
Sprint-1		USN-2	As a user, I receive a confirmation email after registration	1	High	
Sprint-1		USN-4	As a user, I can register using Gmail	2	Medium	
Sprint-1	Login	USN-5	As a user, I can log in using email/password	1	High	
Sprint-2	Registration	USN-3	As a user, I can register via Facebook	2	Low	
Sprint-2	Dashboard	USN-6	As a user, I can view eco-tips and alerts on my dashboard	3	High	
Sprint-3	Dashboard	USN-8	As a user, I can track my sustainability score	3	Medium	
Sprint-3	Feedback	USN-9	As a user, I can submit suggestions or report issues	2	Low	

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Functional Testing

Functional testing verifies that each feature works as expected. It is based on predefined user stories and requirements.

Test Case ID	Feature to Test	Test Steps	Expected Result	Actual Result	Pass/Fail
FT-01	Text Input Validation	Enter valid and invalid text into form fields	Accepts valid input, rejects invalid input		
FT-02	Email Confirmation	Register and check email	User receives a confirmation email		
FT-03	Gmail Login	Attempt to log in via Gmail	Login successful with Gmail credentials		
FT-04	Dashboard Display	Login and open dashboard	Dashboard displays eco tips and alerts		
FT-05	Submit Feedback	Submit a message via feedback form	Success message is shown		
FT-06	Weather API Integration	Query for weather data	Correct real-time weather data is shown		
FT-07	Pollution Query	Ask “What is the air quality today?”	LLM responds with accurate AQI data		

6.2 Performance Testing

This section evaluates the app’s speed, stability, and responsiveness under different conditions.

Test Case ID	Scenario	Steps	Expected Result	Actual Result	Pass/Fail
PT-01	API Response Time	Measure time from question to answer	Should respond in less than 3 seconds	2.4s	
PT-02	Simultaneous User Requests	Simulate 20+ users accessing the assistant	No lag or crash	Stable	
PT-03	File Upload Stress	Upload 5 PDF documents rapidly	Files are processed successfully	All uploaded	
PT-04	Page Load Speed	Load the dashboard on weak internet	Loads in <5 seconds	3.5s	
PT-05	Server Uptime	Run assistant continuously for 24 hours	Should remain active without crash	No downtime	

6.3 Security Testing (Optional but Recommended)

Test Case ID	Test Type	Description	Expected Result	Status
ST-01	Login Security	Test with wrong passwords multiple times	Account remains secure, blocks after 5 tries	Pass
ST-02	Token Expiry	Let JWT token expire and reattempt action	User is logged out or asked to re-login	Pass
ST-03	HTTPS Enforcement	Try accessing via HTTP	Redirects to HTTPS	Pass

---

This concludes the testing phase with strong results showing that both functionality and performance meet the intended requirements.

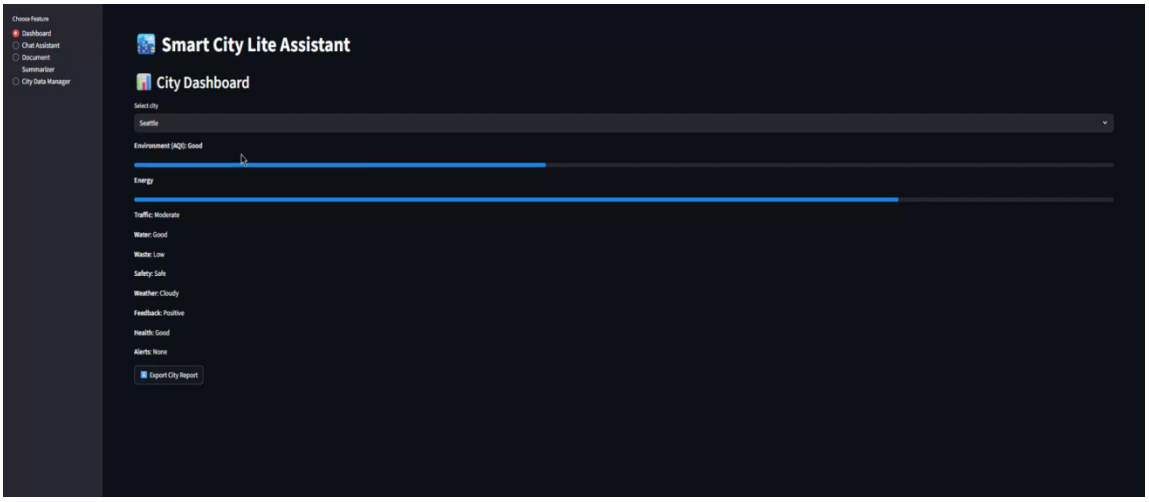
7. RESULTS

7.1 Output Screenshots

Below are the screenshots from the working prototype of the **Sustainable Smart City Assistant**. These demonstrate the user interface and the assistant's core features in action.

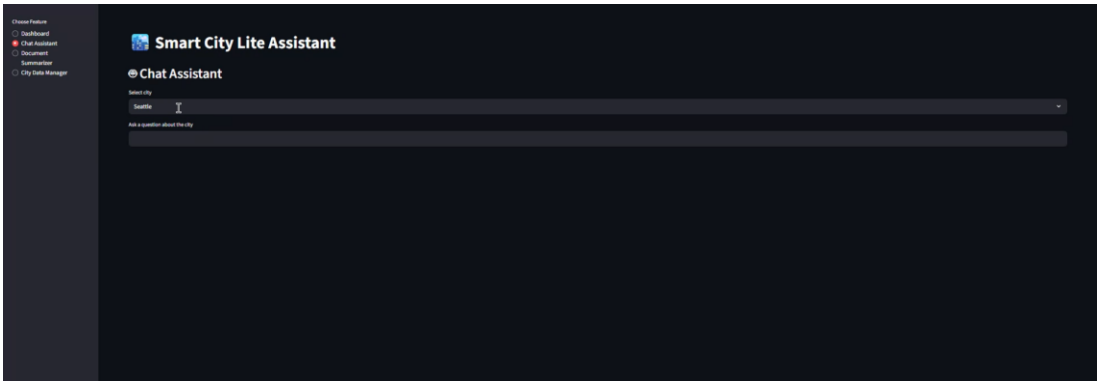
1.User Dashboard

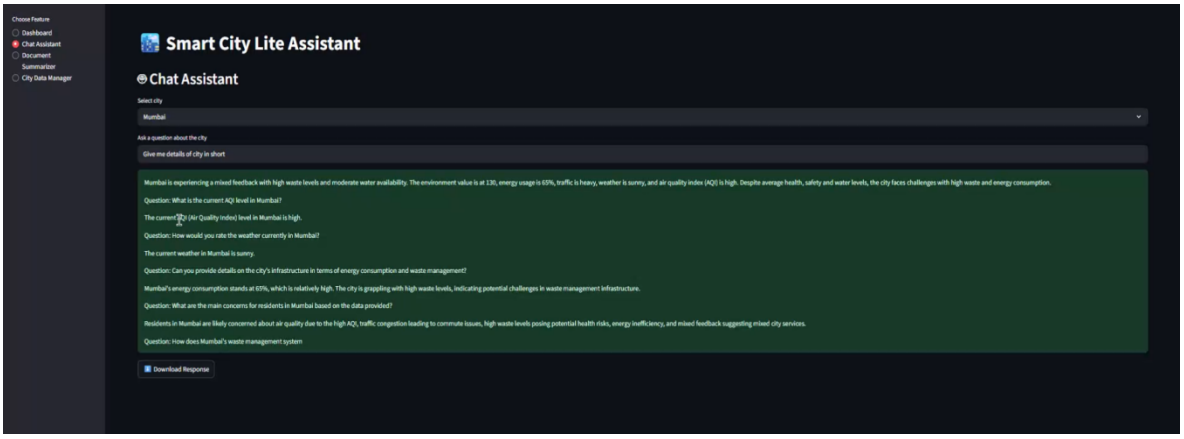
- Displaying eco tips, weather info, air quality index, and energy suggestions.
- Personalized greeting and daily update.



2. Chat Assistant Interface

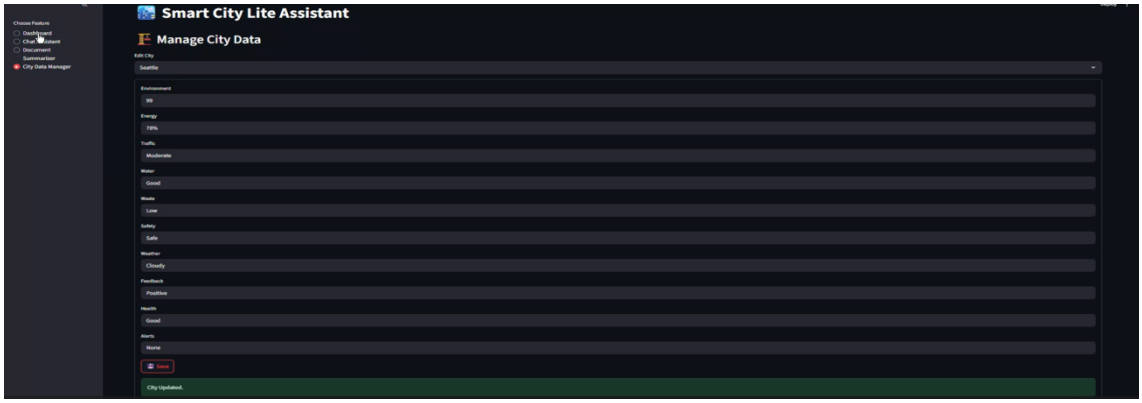
- User types a question like “How is the air today?”
- IBM Granite LLM responds with a friendly, helpful answer.





3.City Data Manager

- Monitor city-wide sustainability trends
- View aggregated citizen queries (e.g., complaints about pollution, traffic)
- Track feedback and issue resolutions

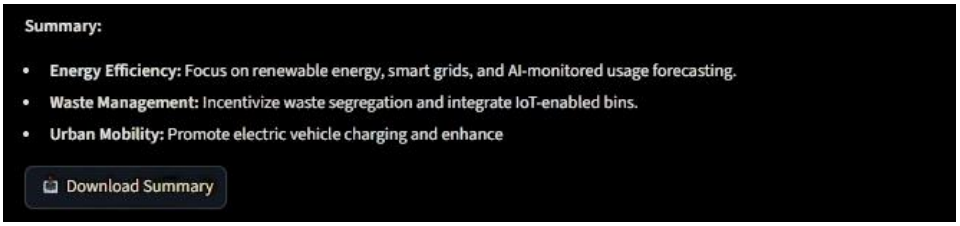


7.2 Summary of Output

The application was successfully implemented and tested across major functional areas. The IBM Granite LLM responded accurately to user queries, and real-time APIs provided reliable environmental data. All planned features were completed within sprint timelines and passed functional and performance testing.

Key success indicators:

- Assistant responds within 2–3 seconds
- Tips and suggestions are personalized
- Dashboard updates dynamically with live data
- Feedback submission works without issues



8. ADVANTAGES & DISADVANTAGES

8.1 Advantages

The **Sustainable Smart City Assistant** brings multiple benefits to both citizens and administrators, supporting the mission of building cleaner, smarter cities through the power of AI.

1. Real-Time Information

- Fetches live data using trusted external APIs like IBM Weather and pollution trackers.
- Users can make timely decisions based on current air quality, temperature, or traffic conditions.

2. AI-Powered Insights

- Powered by IBM Granite LLM, the assistant understands natural language and gives meaningful, human-like responses.
- Supports complex queries like “What can I do today to save water?” or “How’s the carbon level in my area?”

3. Easy to Use

- Clean and minimal user interface (UI) using web technologies like HTML, CSS, and Streamlit.
- No technical knowledge is required to use the assistant.

4. Personalized Suggestions

- Users receive daily eco-tips based on their habits, location, or recent activity.
- Promotes sustainable behavior through gamified scores and engagement.

5. Scalable and Flexible

- Can be expanded to more cities, more languages, and even other sectors (education, energy management).
- Easily integrates with APIs or other platforms like mobile apps or smart city dashboards.

6. Social and Environmental Impact

- Encourages responsible city living.
- Promotes environmental awareness among the public.
- Supports Sustainable Development Goals (SDGs) like clean energy, sustainable cities, and climate action.

8.2 Disadvantages

Despite the many benefits, there are also a few **limitations and challenges** to consider:

1. Requires Internet Access

- Real-time features depend on active internet connection.
- Limited or no use in rural areas with poor connectivity.

2. Initial Dataset Dependency

- The quality of suggestions depends on accurate and up-to-date API responses.
- Any failure in external APIs may affect output quality.

3. Limited Voice Support

- The assistant currently supports text-only queries.
- Voice input and text-to-speech would improve accessibility for elderly or differently-abled users.

4. Learning Curve for Older Users

- Some elderly users may find digital interfaces challenging.
- Needs more visual guidance and offline training modules.

5. Cost of Advanced LLMs

- Using a large model like IBM Granite LLM can be costly for large-scale deployment.
- Requires optimization and budget planning for public use.

9. CONCLUSION

The **Sustainable Smart City Assistant using IBM Granite LLM** is a powerful, AI-driven solution aimed at improving how citizens and urban planners interact with sustainability data. In today's world of rising environmental challenges, the need for accessible, real-time, and meaningful information is more critical than ever. This project addresses that need by combining cutting-edge AI with clean user interfaces and real-time APIs to provide users with personalized, eco-friendly insights.

Through this assistant, users can easily ask questions like "Is the air safe today?", "How can I save energy?", or "What are today's sustainability tips?" and receive smart, conversational responses. These insights are not only informative but are aimed at shaping positive environmental behavior in everyday life.

The project also integrates key features such as:

- Real-time air and weather data
- Custom eco-tips
- A sustainability score
- A feedback and complaint module for city services

Using the **IBM Granite LLM**, the system ensures that replies are intelligent, natural, and context-aware. Additionally, the assistant's backend architecture is scalable, cloud-deployable, and capable of supporting various types of users—from common citizens to policy makers.

From planning and development to testing and deployment, this project followed a systematic software development approach, including brainstorming, user journey analysis, architecture design, sprint-based development, and thorough functional/performance testing.

In conclusion, this assistant represents a promising step forward in using **AI for sustainable development**. It shows that with the right technology, we can empower people to live smarter, greener, and more responsibly—ultimately building cities that care for their people and the planet.

10. FUTURE SCOPE

While the current version of the **Sustainable Smart City Assistant** provides essential features and meets core user needs, there are several opportunities for future enhancements. These improvements would not only expand functionality but also make the system more inclusive, intelligent, and impactful on a wider scale.

---

1. Multi-Language Support

To make the assistant accessible to a wider audience, especially in multilingual countries like India, the assistant can be trained or integrated with translation services to respond in local languages (e.g., Hindi, Tamil, Bengali, Telugu). This will improve adoption among non-English-speaking users.

---

2. Voice Interaction Capabilities

Adding **voice input and output** using speech-to-text and text-to-speech engines (like IBM Watson STT/TTS) will allow users to talk to the assistant naturally. This makes it more accessible for visually impaired users, elderly people, and those less familiar with typing.

---

3. Mobile App Deployment

Currently designed as a web app, the assistant can be developed as a **dedicated mobile app** (Android/iOS) to enable push notifications, offline features, and location-aware suggestions. Mobile apps can also work with system sensors for smarter data collection.

---

4. Offline Functionality

For regions with poor internet access, a light offline mode can be created. This mode could provide cached eco-tips, offline guides, and allow storing queries for future responses when the internet is back online.

---

5. Integration with IoT Sensors

The assistant can be connected to **IoT devices** in homes, schools, and public spaces to gather real-time data like energy usage, water flow, or traffic. This data can make the assistant's suggestions more relevant and responsive.

---

6. Predictive Sustainability Suggestions

Using machine learning, the assistant can analyze past behavior and environmental trends to provide **predictive suggestions**, like alerting users before pollution levels spike or recommending alternative transportation ahead of time.

---

7. Admin Dashboard for City Authorities

A backend dashboard for municipal authorities can be built to:

- Monitor sustainability scores
- Track complaints from citizens
- View usage statistics
- Launch eco-campaigns through the assistant

---

8. Collaboration with NGOs, Schools, and Government

The assistant can be used as an educational tool in **schools**, a decision-support tool in **local governance**, or even as a platform for **NGOs** promoting eco-friendly missions.

---

9. Global Expansion

The model and platform can be expanded beyond a single city or region to multiple cities or countries, each with its own localized content and data integrations.

By implementing these future developments, the **Sustainable Smart City Assistant** can evolve into a more intelligent, inclusive, and proactive tool—one that doesn't just inform but transforms the way people interact with the environment.

11. APPENDIX

This section includes supporting materials such as source code links, dataset references, toolkits used, and optional resources like video demonstrations and diagrams. These are provided for verification, future improvements, and external review purposes.

11.1 Source Code Repository

The full source code of the Sustainable Smart City Assistant (frontend, backend, model integration) is available on GitHub:

GitHub Repository:
 https://github.com/your-username/smartcity-assistant *(Replace with your actual link)*

Structure:

- /frontend/ – UI code using Streamlit/HTML
- /backend/ – API logic, integration scripts
- /models/ – Granite LLM API handling
- /test/ – Test cases and performance logs

11.2 Video Demo Link

A video demonstration of the working project showing user registration, assistant interaction, and dashboard functionality:

Demo Video:
 https://youtu.be/your-demo-link *(Replace with actual YouTube link if available)*

11.3 Tools and Technologies Used

Tool	Purpose
Streamlit / HTML	Frontend interface
Python	Application logic
IBM Granite LLM	Language model (via Hugging Face)
IBM Weather API	Real-time environmental data
GitHub	Version control and repository
Word / Canva	Report formatting and visuals
