RISE KRISHNA SAI PRAKASAM GROUP OF INSTITUTIONS

Ongole

INTERNSHIP REPORT

ON

"Sustainable Smartcity AI Assistant"

Submitted in partial fulfillment of the requirements of the Virtual Internship Program

Organized by

SMART BRIDGE

Submitted by

Ramayanapu Navya Sri

Aravapalli Hiranmai Sri

Sighakolli Venkata Sujatha

Achyuth Vemula

TEAM ID:

LTVIP2025TMID37446

Smart Bridge

June 2025

Index

BRAINSTORMING &IDEATION

Objectives:

1. Identify Core Urban Sustainability Challenges
    - Pollution, traffic congestion, energy waste, public service inefficiency.
2. Define AI Assistant Capabilities
    - Enable natural interaction with smart city data.
    - Deliver intelligent suggestions and real-time insights.
3. Align Technology with Purpose
    - Match tools (Granite LLM, ML, HTML, Unicorn) to user needs and system goals.
4. Promote Citizen Engagement & Awareness
    - Use AI to encourage eco-friendly behaviour and informed decisions.

Key Points:

- User-Cantered Design: Focus on ease of use for both citizens and city planners.
- Data-Driven Decisions: Use ML models to analyze traffic, energy, and pollution data.

- AI Interaction: Leverage IBM Granite LLM for smart, conversational responses
- Real-Time Support: Use Unicorn to manage fast, scalable backend communication.
- Modular Features: Plan for features like route optimization, pollution alerts, and energy usage tips.
- Sustainability First: Every feature must contribute to environmental and social sustainability.

REQUIREMENT ANALYSIS

Objectives:

1. **Define Functional Requirements**
    - Identify what the AI assistant must do (e.g., respond to queries, give eco-friendly suggestions).
2. **Determine Non-Functional Requirements**
    - Set expectations for performance, scalability, usability, and security.
3. **Identify User Roles and Needs**
    - Understand what citizens, city planners, and other users expect from the assistant.

1. Functional Requirements

Natural Language Interface:
Use IBM Granite LLM to enable human-like conversations for user queries.

- **Real-Time Data Access**:
  Provide live updates on traffic, air quality, energy usage, etc.
- **Smart Suggestions**:
  Use ML to recommend eco-friendly actions (e.g., low-emission travel routes, energy-saving tips).
- **User-Friendly Frontend**:
  Build an interactive, responsive HTML interface for users

2. Non-Functional Requirements

- **Performance**:
  Quick response times using **Unicorn** (Unicorn) as a fast ASGI server.
- **Scalability**:
  Capable of handling multiple users and data streams in real time.
- **Security**:
  Ensure secure handling of user data and API communications.

- **Reliability**:
  High uptime with minimal failures or downtime.

3. User Requirements

- **Citizens**:
  - Get real-time updates on their local environment.
  - Ask questions about sustainability and city services.
- **City Planners/Authorities**:
  - Access trend insights and analytics for decision-making.
  - Use predictive tools to optimize urban planning.

4. Technical Requirements

- **AI Engine**:
  IBM Granite LLM for advanced natural language understanding and generation.
- **Frontend**:
  HTML/CSS/JavaScript for interface design and interactivity.
- **Backend**:
  Python-based API using **Unicorn** to serve AI and ML functionality.

- **Machine Learning Models**:
  For predictions and recommendations based on historical and live.

PROJECT DESIGN

Objective

To build an AI-powered assistant that helps improve urban sustainability through smart insights on energy, transport, waste, and environment.

Technology Stack

- **Frontend:** HTML (UI for user interaction)
- **Backend:** Python (Fast API/Starlette with **Unicorn** ASGI server)
- **AI Engine:** IBM **Granite LLM** (natural language understanding)
- **Machine Learning:** Energy forecasting, traffic prediction, waste optimization
- **Deployment:** Cloud (optional: IBM Cloud or Docker)

Key Features

- AI chat assistant for sustainability queries
- Real-time data insights (e.g., pollution, traffic)
- Smart ML predictions for planning and efficiency
- Clean, browser-based interface

System Flow

1. User asks question via HTML interface
2. Backend (Unicorn) sends to IBM Granite LLM or ML models
3. AI processes query and returns insights
4. Response shown in chat or dashboard

Security

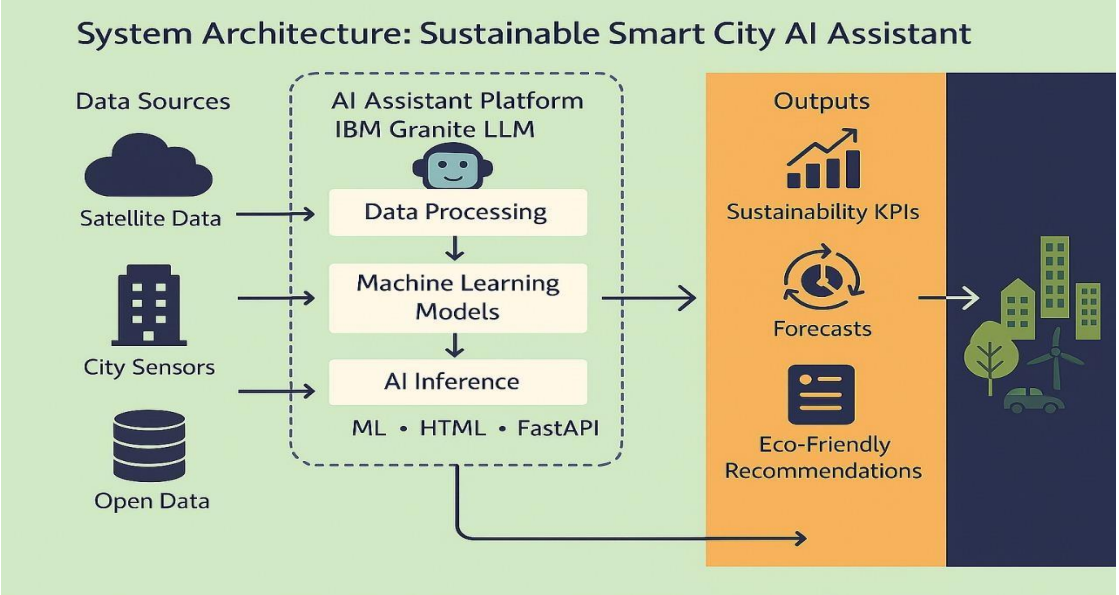
- Token-based API
- HTTPS
- GDPR-compliant data handling

Main Modules

- index.html – User interface
- main.py – Backend API (Fast API + Unicorn)

System Architecture Diagram:

Sustainable Smart City AI Assistant



PROJECT PLANNING

Objectives

- Develop an AI-powered assistant for smart, sustainable urban living.
- Use IBM Granite LLM for natural language interaction.
- Provide real-time insights on energy, traffic, and environment.
- Predict and optimize city resources using ML models.
- Deliver a web-based solution accessible to citizens and city admins.

Key Points

- **Frontend:** HTML-based user interface for input and results.
- **Backend:** Python server using Fast API + Unicorn (ASGI).
- **AI Engine:** IBM Granite LLM for understanding user queries.
- **ML Models:** Forecasting (e.g., traffic, energy, pollution).

- **Data Flow:** User → API → LLM/ML → Response → UI.
- **Security:** Token-based API access, HTTPS, data privacy.
- **Deployment:** Cloud-ready, scalable with Docker/Kubernetes.

Task Allocation:

Members	Tasks
Ramayanapu Navya Sri	• Set up Fast API backend and Uvicorn server • Create API routes for chat, KPI, reports • Handle backend logic for report generator • Design Streamlit/HTML UI layout and sidebar
Aravapalli Hiranmai Sri	• Build chat interface and input form • Apply final UI styling and theme • Integrate IBM Granite LLM into backend • Design and test prompts for Chat Assistant • Implement KPI Forecasting logic • Create AI-powered report response

Sighakolli Venkata Sujatha	• Test all features (chat, forecast, report) • Deploy using Uvicorn (VS Code/Colab) • Write project documentation and README • Handle bug fixes and integration QA

TimeLine and Milestones:

Date	Milestone
Week-1	• Requirement Specifications • Environment initialization &API Key Setup • AI Model Integration
Week-2	• Backend API Development • Stream Frontend UI Development • Pinecone semantic search integration
Week 3	• ML-based Forecasting and anomaly detection • Sustainability Report Generation
Week 4	• Chat Assistant Creation • Final integration & testing

PROJECT DEVELOPMENT

Objectives

- Build a smart AI assistant to support sustainable city management.
- Integrate IBM Granite LLM for natural language understanding.
- Use ML to predict and optimize urban services (energy, traffic, waste).
- Develop a responsive web interface for easy public and admin access.
- Deploy a scalable, secure backend using Unicorn (ASGI server).

Key Development Points

- Frontend:
  - Developed using HTML
  - Simple, interactive UI for asking questions and viewing insights
- Backend:
  - Built with Python using FastAPI + Unicorn

    - Handles API routing, data processing, and async responses
- AI Integration:
    - Connected to IBM Granite LLM for interpreting user queries
    - Delivers smart, actionable sustainability suggestions
- ML Integration:
    - Models for energy consumption, traffic prediction, pollution trends
    - Trained on real-world smart city datasets
- Testing & Optimization:
    - Functionality, performance, and security tested
    - Refined model outputs and AI responses for relevance
- Deployment:
    - Deployed to cloud with scalability support
    - Secure APIs and real-time responsiveness enabled

FUNCTIONAL & PERFOMANCE TESTING

Objectives

- Ensure the application works as intended across all features.
- Validate AI responses, ML predictions, and system reliability.
- Measure performance under different user loads and conditions.
- Identify and fix bugs, bottlenecks, or failures.

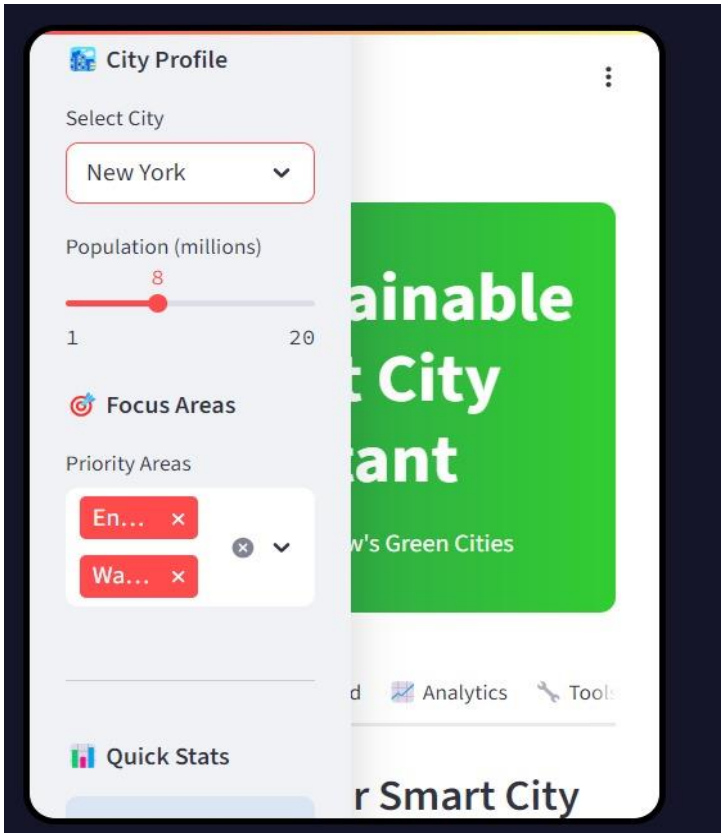
Key Functional Testing Points

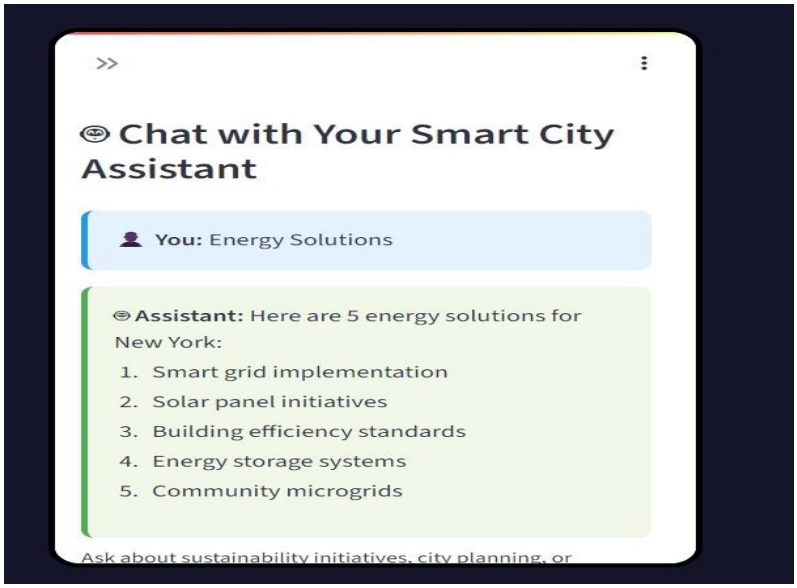
- Chat Input/Output Test – Validate LLM response accuracy.
- ML Model Output Test – Verify energy/traffic predictions.
- API Endpoint Test – Ensure all routes work correctly.
- UI Functionality Test – Forms, buttons, and data display.
- Data Flow Test – From user input → backend response.

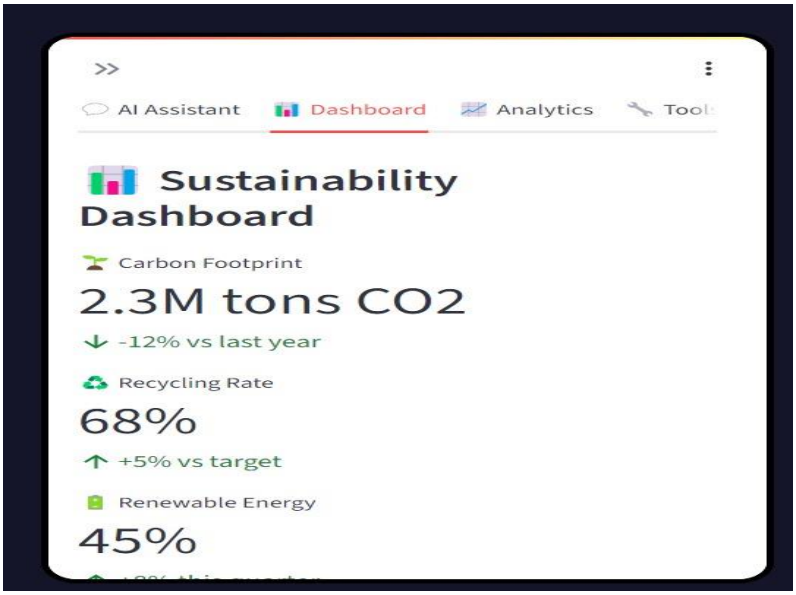
Key Performance Testing Points

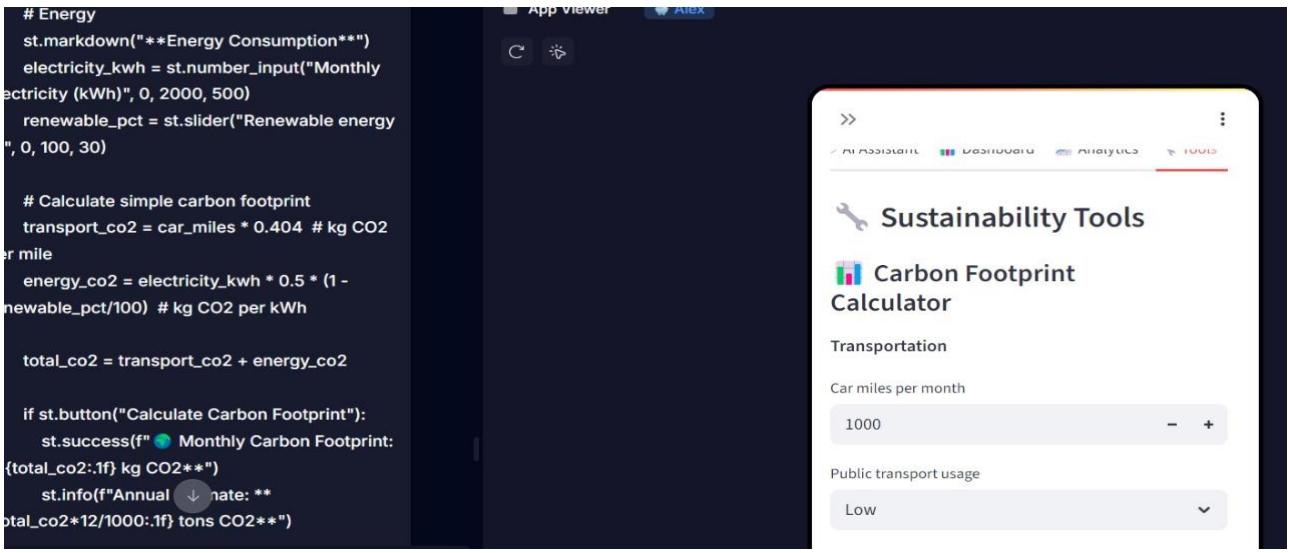
- Load Testing – Test with multiple concurrent users.
- Response Time – Measure time taken for LLM + ML outputs.
- Scalability Test – Check Unicorn server handling with high traffic.
- Memory & CPU Usage – Ensure models run within limits.
- Stability Test – Long-session interaction and uptime monitoring.

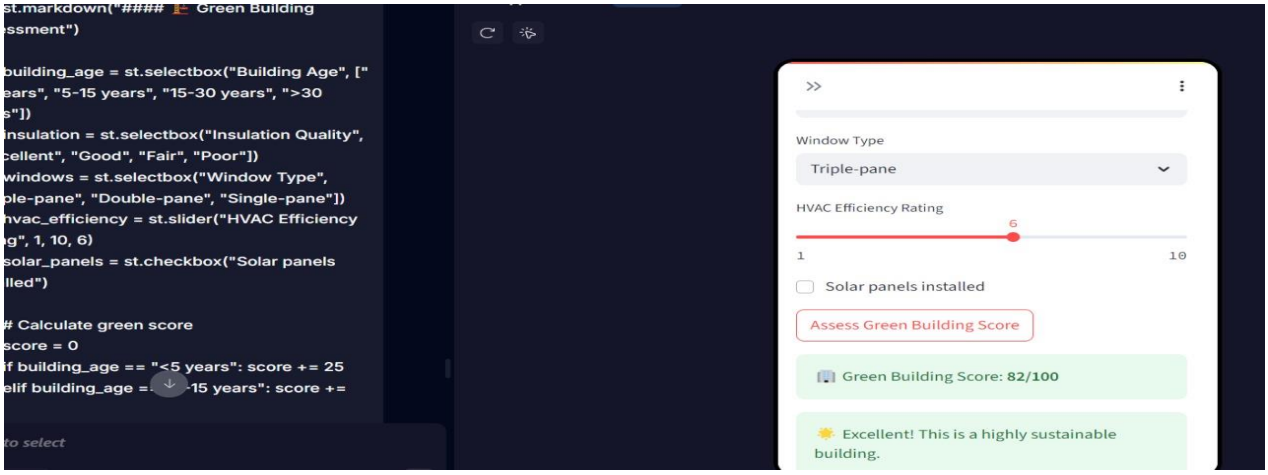
DEPLOYMENT:




Chat with Your Smart City Assistant
You: Energy Solutions
Assistant: Here are 5 energy solutions for New York:
1. Smart grid implementation
2. Solar panel initiatives
3. Building efficiency standards
4. Energy storage systems
5. Community microgrids
Ask about sustainability initiatives, city planning, or


AI Assistant
Dashboard
Analytics
Sustainability Dashboard
Carbon Footprint
2.3M tons CO2
-12% vs last year
Recycling Rate
68%
+5% vs target
Renewable Energy
45%


# Energy
st.markdown("**Energy Consumption**")
electricity_kwh = st.number_input("Monthly
ctricity (kWh)", 0, 2000, 500)
renewable_pct = st.slider("Renewable energy
", 0, 100, 30)
# Calculate simple carbon footprint
transport_co2 = car_miles * 0.404 # kg CO2
r mile
energy_co2 = electricity_kwh * 0.5 * (1 -
newable_pct/100) # kg CO2 per kWh
total_co2 = transport_co2 + energy_co2
if st.button("Calculate Carbon Footprint"):
st.success(f" Monthly Carbon Footprint:
{total_co2:.1f} kg CO2**")
st.info(f"Annual nate: **
tal_co2*12/1000:.1f} tons CO2**")
App Viewer
Alex
Sustainability Tools
Carbon Footprint Calculator
Transportation
Car miles per month
1000
Public transport usage
Low



Conclusion:

The *CityP'AI'Rtner* project effectively demonstrates the potential of integrating advanced large language models like IBM Granite into smart city solutions. By providing AI-driven insights, sustainability recommendations, KPI forecasting, and an interactive assistant, the application supports urban planners, administrators, and citizens in fostering eco-friendly, data-informed decisions. With a user-friendly interface powered by HTML-JS/Streamlit, and a scalable backend using FastAPI, the assistant is a step toward digitized urban management. This project not only highlights the role of generative AI in city governance but also lays the groundwork for future expansions involving real-time IoT integration, environmental monitoring, and predictive analytics to build truly sustainable and intelligent cities.

Future Scope

The CityP'AI'Rtner assistant can be enhanced by integrating real-time IoT data for live environmental monitoring. Future versions may include multilingual support for broader accessibility. Predictive analytics and machine learning can forecast urban sustainability trends more accurately. Integration with government databases and citizen feedback systems can enable smart policy recommendations.

Appendix:

Source code: Provided in GitHub repository

GitHub –

Demo Video: provided in GitHub and also through drive link

Drive –

https://drive.google.com/file/d/1lZQl42JBUDv9a9PaSFMAE9bssp1hC7RE/view?usp=drivesdk