

# **Full Stack Development Project report**

**1. Introduction**

**2. Project Overview**

**3. Architecture**

**4. Setup Instructions**

**5. Folder Structure**

**6. Running the Application**

**7. API Documentation**

**8. Authentication**

**9. User Interface**

**10. Testing**

**11. Screenshots or Demo**

**12. Known Issues**

**13. Future Enhancements**

## 1. Introduction

Project Title: Sustainable Smart City Assistant using IBM Granite LLM

Team Members & Roles:

Name	Role
Singuluri Chandra Sagar	Team Leader, LLM Integration & Deployment
Sesetti Mohana Krishna	Gradio UI Developer & Model Testing
Tibirisetti Satish	KPI Forecasting & Anomaly Detection
Sangani Kasu Raju	Citizen Feedback & Eco Tips Module

## 2. Project Overview

Purpose:

The application allows city residents to report civic issues (e.g., potholes, streetlights, garbage) directly to authorities, ensuring real-time communication and resolution.

- Features:
- Issue reporting with category, description, image, and location
- Admin panel for issue tracking and status updates
- Real-time notifications to users
- User feedback and resolution rating system
- Public transparency dashboard

## 3. Architecture

Frontend: Built using Gradio. Provides interactive UI with embedded Python logic.

Backend: Implemented in Python using FastAPI or Flask. Handles APIs and DB logic.

Database: MongoDB for storing reports, user data, feedback, and status.

## 4. Setup Instructions

- Prerequisites:
- Python 3.10+
- MongoDB (Local or Atlas)
- Pip (Python package installer)
- Installation Steps:

- `git clone https://github.com/sagarnaidu04/sustainable_smart_city.git`
- `cd smart-city-gradio`
- `pip install -r requirements.txt`
- `python app.py`

## 5. Folder Structure

```
/smart-city-gradio
├── app.py          # Main Gradio app
├── Requirements.txt #requirement needed
```

## 6. Running the Application

`python app.py`

Gradio will host the app locally at `http://localhost:7860`.

## 7. API Documentation

List of available backend endpoints:

Endpoint	Method	Description
<code>/report</code>	POST	Submit a new issue
<code>/issues</code>	GET	Retrieve all issues
<code>/update-status</code>	PUT	Admin updates issue status
<code>/feedback</code>	POST	User submits feedback

## 8. Authentication

Light user verification (e.g., email/username-based). Admin login handled via password input in Gradio UI.

## 9. User Interface

Built using Gradio components:

- Textbox: Issue title, description
- Dropdowns: Category selection
- Image Upload: Upload proof images
- Map Picker or Text Input: Location info

- Admin UI: Status updates and feedback views

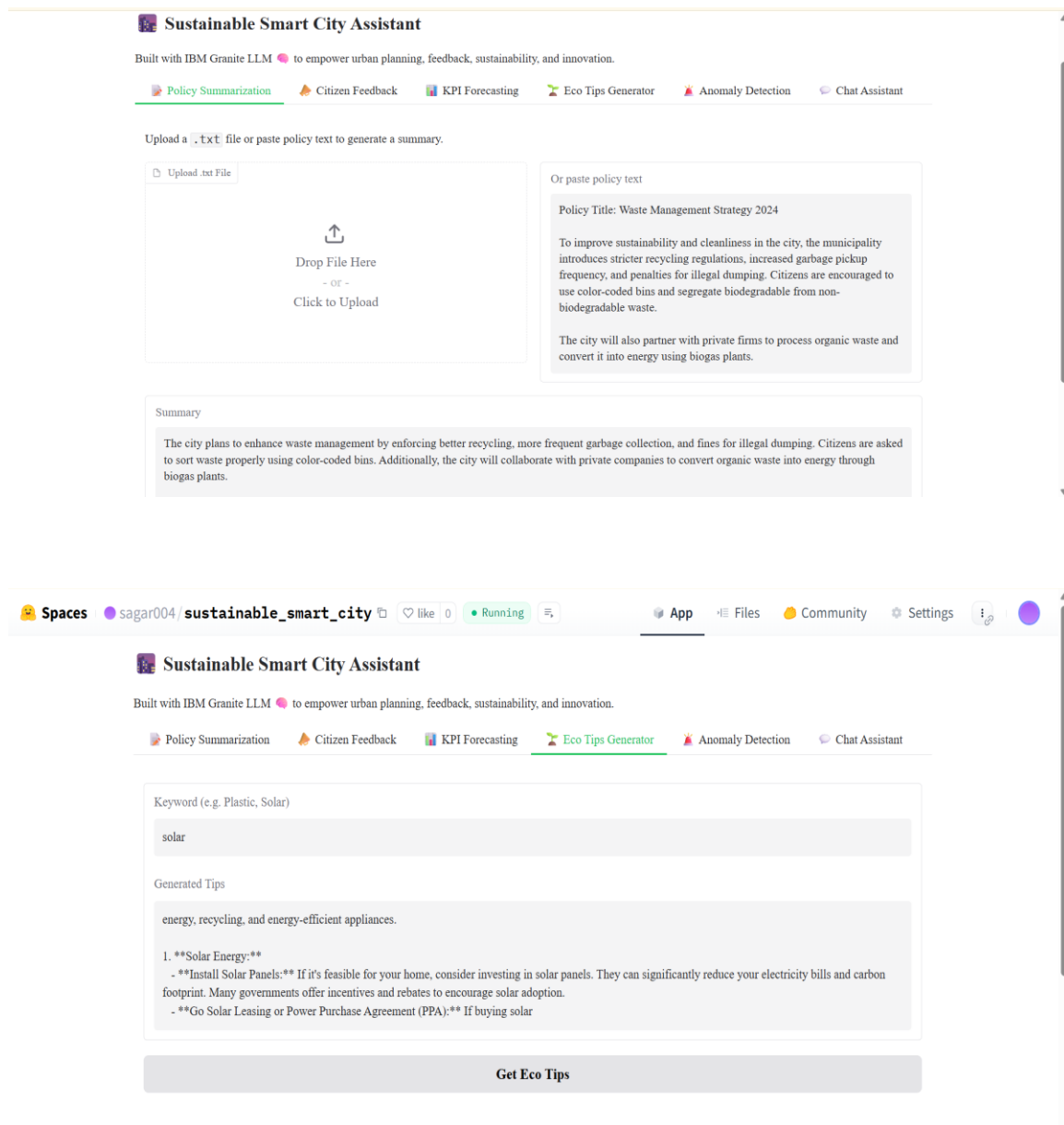
## 10. Testing

Manual testing via Gradio interface and unit testing of core functions using pytest/unittest.

## 11. Screenshots or Demo

Link to live demo or hosted app:

[https://huggingface.co/spaces/sagar004/sustainable\\_smart\\_city](https://huggingface.co/spaces/sagar004/sustainable_smart_city)



## 12. Known Issues

Limited session persistence in shared Gradio links.

UI optimization required for smaller screens.

Takes large amount of time to return output because the models used are too large to load.

## 13. Future Enhancements

- Secure authentication for users and admins
- Real-time tracking via WebSockets
- Email/SMS notification integration
- Analytics dashboard for authorities
- AI-based automatic issue classification