Project Report

1. INTRODUCTION

1.1 Project Overview

Smart City Assistant is an AI-driven web application that enhances the sustainability and efficiency of urban living. It integrates multiple modules like environment monitoring, city health status, chatbot support, feedback analysis, and summarization using advanced LLMs.

1.2 Purpose

The purpose of this project is to build a centralized platform for citizens to interact with smart city services using AI capabilities.

2. IDEATION PHASE

2.1 Problem Statement

Modern urban cities lack centralized AI-based assistants for sustainable management and citizen engagement.

2.2 Empathy Map Canvas

Says: Wants real-time updates about the environment and city health.

Thinks: Is concerned about pollution and lack of centralized help.

Feels: Frustrated with current disconnected systems.

Does: Searches online or contacts municipal offices individually.

2.3 Brainstorming

We brainstormed modules that address environmental monitoring, chatbot support, feedback, summarization, and healthcare awareness.

3. REQUIREMENT ANALYSIS

Project Report

3.1 Customer Journey Map

Users interact with the platform via a simple UI, access services such as air quality check, feedback submission, summarization, and city health updates.

3.2 Solution Requirement

FastAPI or Flask backend for handling APIs

Frontend for user interaction (HTML/CSS/JS)

OpenAI or IBM Granite API integration

Static image resources

3.3 Data Flow Diagram

User input → Route handling → Model interaction (LLM/Custom) → Output to frontend

3.4 Technology Stack

Frontend: HTML, CSS, JavaScript

Backend: Python, Flask or FastAPI

Model: IBM Granite LLM / OpenAI

Deployment: Render

4. PROJECT DESIGN

4.1 Problem Solution Fit

AI-driven smart assistants are ideal for managing and simplifying multiple urban services in one place.

4.2 Proposed Solution

A modular AI assistant web app with LLM integration for feedback, chatbot, summarization, and environmental data processing.

4.3 Solution Architecture

Frontend (UI) ⇄ Backend (FastAPI/Flask) ⇄ Model (Granite/OpenAI) ⇄ Output response

Project Report

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Week 1: Ideation and wireframe

Week 2: Backend API setup

Week 3: LLM integration

Week 4: Frontend design

Week 5: Testing and deployment

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Tested API response times for multiple modules.

Tested frontend loading with images and dynamic responses.

Stress-tested summarizer and chatbot under concurrent load.

7. RESULTS

7.1 Output Screenshots

Screenshots of all modules were taken after successful integration and are attached in the appendix.

8. ADVANTAGES & DISADVANTAGES

Advantages

Unified platform for multiple smart city features.

User-friendly AI-based assistant.

Scalable and modular design.

Project Report

Disadvantages

Dependent on internet connectivity.

API limits from OpenAI/Granite.

Model performance depends on input quality.

9. CONCLUSION

This Smart City Assistant project demonstrates the potential of AI in public services, combining multiple helpful modules under one intelligent interface.

10. FUTURE SCOPE

Mobile version of the app.

Real-time data integration via sensors.

Voice interaction using speech-to-text APIs.

Support for multiple Indian languages.

11. APPENDIX

Source Code

GitHub Repository: https://github.com/akshaya615/smart-city-assistant

Dataset Link

This project uses live APIs and LLMs, so no static datasets were used.

GitHub & Project Demo Link

GitHub: https://github.com/akshaya615/smart-city-assistant

Live Demo: https://smart-city-assistant.onrender.com