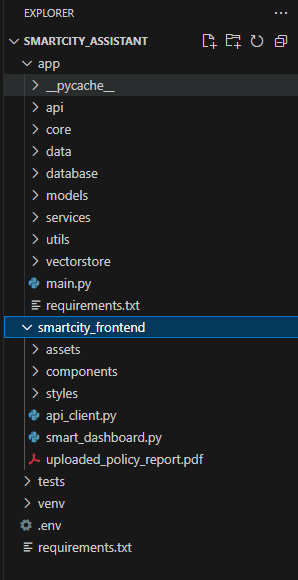
**Development Flow**

Phase 1 – Project Initialization

Modular Folder Structure Defined: Created separate folders for app/api, services, vectorstore,

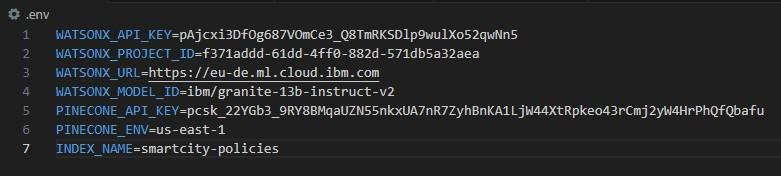
core, frontend/components, and utils for organized and scalable development.



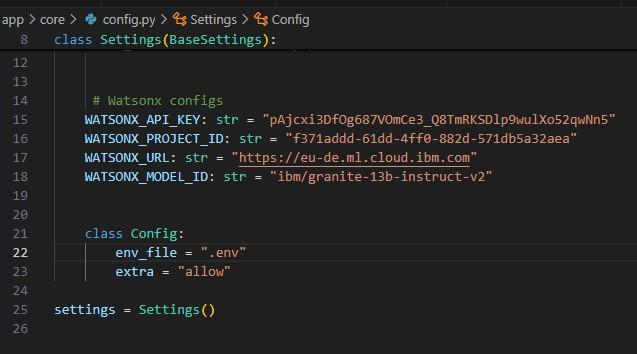
Environment Setup:

.env file created with keys for Pinecone and Watsonx. config.py loads environment variables securely using pydantic.

.env file



Config.py file



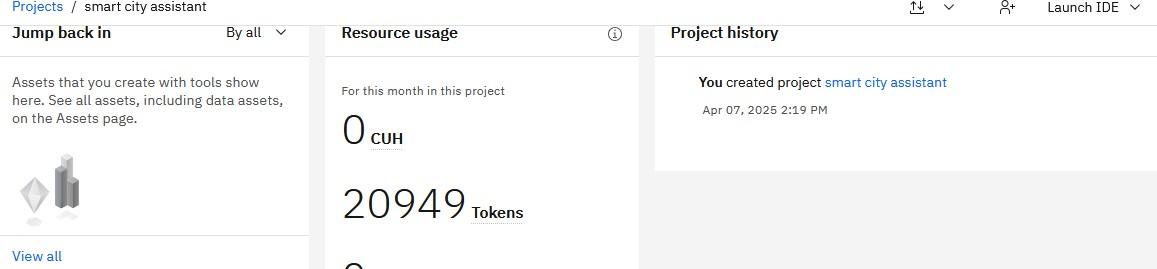
Pinecone Initialization:

pinecone\_client.py written to initialize the Pinecone vector index (smartcity-policies). Ensured creation with correct dimension=384 matching embedding model.

Phase 2 – IBM Watsonx Integration

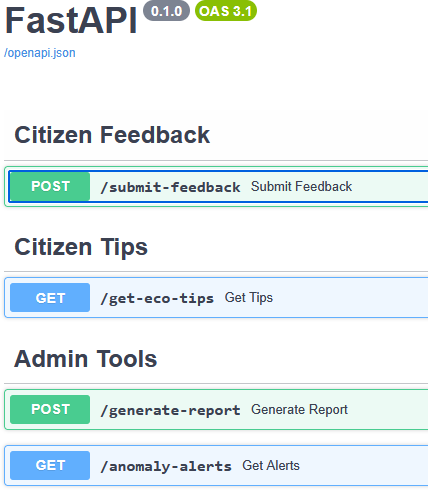
Watsonx Key & Model Configuration: Set up .env with:

WATSONX\_API\_KEY, PROJECT\_ID, MODEL\_ID



Endpoint Testing:

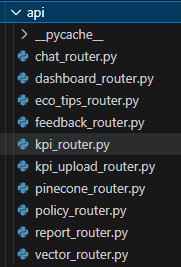
Validated /chat, /policy/summarize, and /get-eco-tips FastAPI routes using Swagger UI.



Phase 3 – Backend API Routers API Routes Implemented:

Developed modular routers:

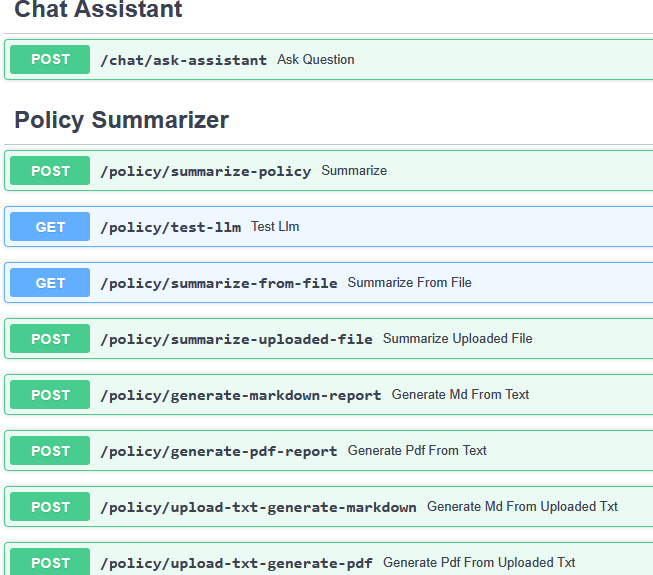
* chat\_router.py
* feedback\_router.py
* eco\_tips\_router.py
* kpi\_upload\_router.py
* anomaly\_checker.py
* vector\_router.py, etc.



Testing & Validation:

Each route tested for:

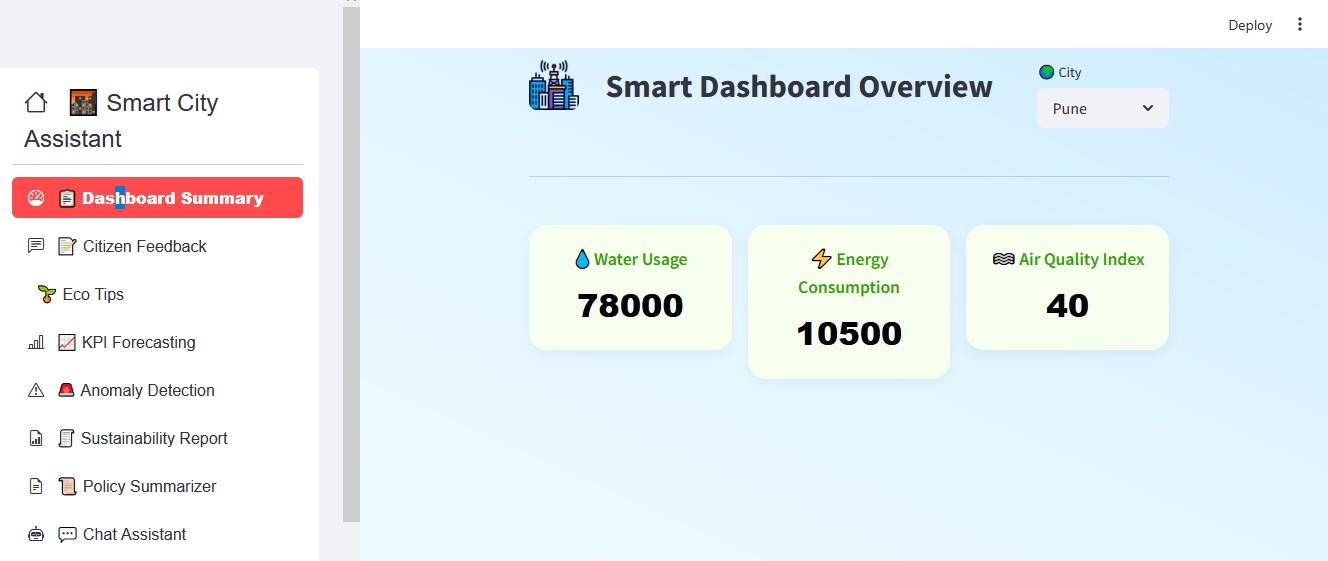
* JSON payload correctness
* File upload parsing
* Error handling & logging
* Swagger auto-documentation generation



Phase 4 – Frontend UI Design

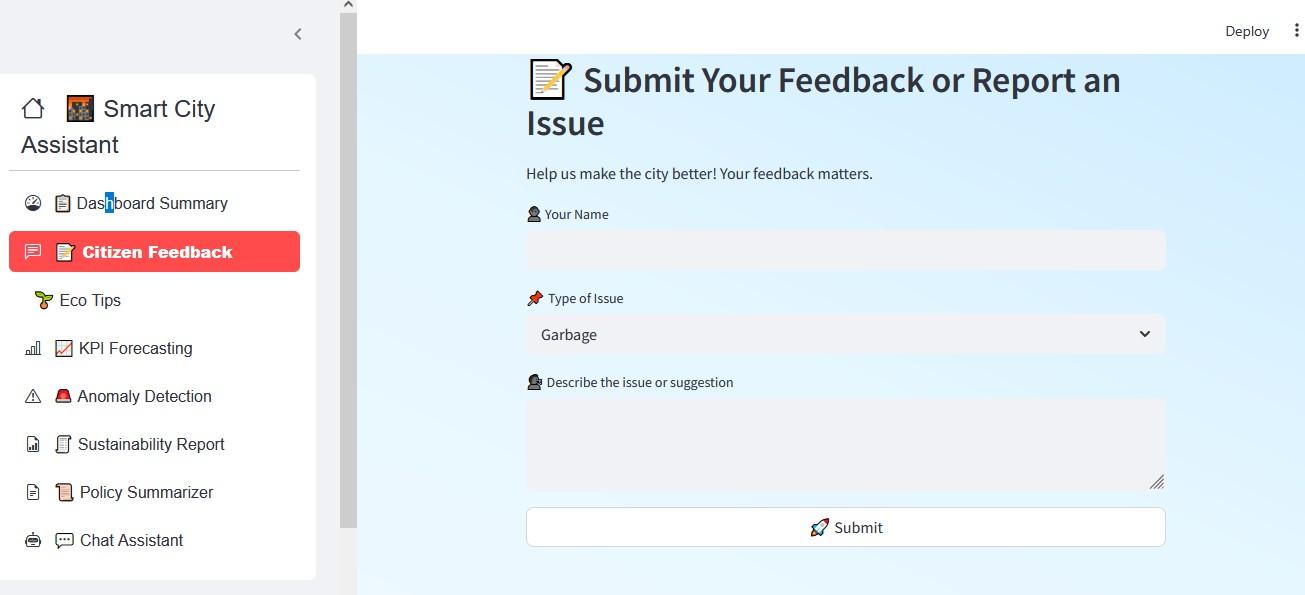
Streamlit UI Structure Implemented:

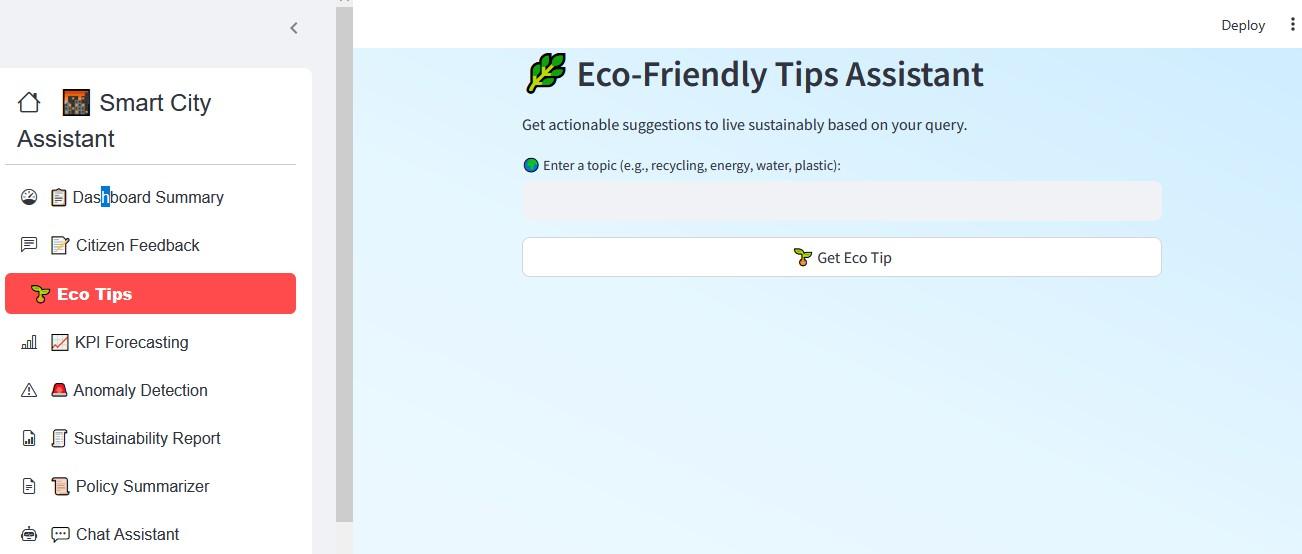
Created central file smart\_dashboard.py with conditional rendering for each module using sidebar navigation.



Component Development:

Developed reusable Streamlit components: summary\_card.py – Beautiful KPI cards chat\_assistant.py – Text prompt and AI reply feedback\_form.py, eco\_tips.py, report\_generator.py, etc.





UI Enhancements Done:

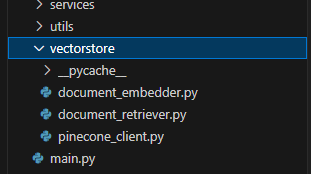
Gradient backgrounds

Icon-rich sidebar using streamlit-option-menu Rounded buttons, font styles, padding fixes

Phase 5 – Pinecone & Document Embedding

Embedding Logic Built:

Created document\_embedder.py and document\_retriever.py using sentence-transformers.



Phase 6 – Report Generation & Deployment

Granite LLM Report Generator:

report\_generator.py takes city name and KPI data, generates detailed city sustainability report using Granite LLM prompts.

Markdown & PDF Support:

Output formatted to text block for copy/paste or PDF download (optional).

End-to-End Integration Testing:

Final dashboard tested on all 8 features: KPI dashboard, feedback form, policy summarization, eco tips, chat, anomaly check, vector search, report generation.

**Development Flow**