**Milestone 5: Streamlit Frontend UI Development**

**🎯 Objective:**

To design a clean, interactive **Streamlit dashboard** that enables real-time interaction with backend APIs—allowing users to chat, view KPI data, upload documents, explore eco tips, and submit feedback easily.

**✅ Step 1: Project Structure (Frontend)**

markdown

CopyEdit

frontend/

├── smart\_dashboard.py

├── components/

│ ├── chat\_assistant.py

│ ├── feedback\_form.py

│ ├── eco\_tips.py

│ ├── summary\_card.py

│ └── report\_generator.py

└── utils/

└── api\_utils.py

**✅ Step 2: Create Main Dashboard — smart\_dashboard.py**

python

CopyEdit

import streamlit as st

from components import chat\_assistant, eco\_tips, feedback\_form, summary\_card, report\_generator

st.set\_page\_config(page\_title="Smart City Assistant", layout="wide")

st.sidebar.title("🌆 Smart City Assistant")

page = st.sidebar.radio("Navigate", ["KPI Dashboard", "Chat Assistant", "Eco Tips", "Policy Summarizer", "Feedback", "Report Generator"])

if page == "KPI Dashboard":

summary\_card.render\_dashboard()

elif page == "Chat Assistant":

chat\_assistant.chat\_interface()

elif page == "Eco Tips":

eco\_tips.display\_tips()

elif page == "Policy Summarizer":

summary\_card.summarize\_policy()

elif page == "Feedback":

feedback\_form.feedback\_ui()

elif page == "Report Generator":

report\_generator.generate\_report\_ui()

**✅ Step 3: Components Overview**

**🔹 chat\_assistant.py**

python

CopyEdit

import streamlit as st

from utils.api\_utils import post\_request

def chat\_interface():

st.subheader("🧠 Ask the Sustainability Assistant")

user\_input = st.text\_input("Your Question:")

if user\_input:

response = post\_request("/chat", {"prompt": user\_input})

st.success(response["response"])

**🔹 eco\_tips.py**

python

CopyEdit

import streamlit as st

from utils.api\_utils import get\_request

def display\_tips():

st.subheader("🌱 Eco-Friendly Tips")

topic = st.text\_input("Enter topic (e.g., energy, water):")

if topic:

tip = get\_request("/get-eco-tips", {"topic": topic})

st.info(tip["tip"])

**🔹 feedback\_form.py**

python

CopyEdit

import streamlit as st

from utils.api\_utils import post\_request

def feedback\_ui():

st.subheader("📣 Submit Your Feedback")

name = st.text\_input("Name")

category = st.selectbox("Category", ["Environment", "Transport", "Water", "Other"])

message = st.text\_area("Message")

if st.button("Submit"):

result = post\_request("/submit-feedback", {"name": name, "category": category, "message": message})

st.success(result["message"])

**🔹 summary\_card.py**

python

CopyEdit

import streamlit as st

from utils.api\_utils import post\_request

def summarize\_policy():

st.subheader("📄 Policy Document Summarizer")

text = st.text\_area("Paste Policy Text Here")

if st.button("Summarize"):

result = post\_request("/upload-doc", {"text": text})

st.success(result["summary"])

def render\_dashboard():

st.subheader("📊 KPI Dashboard")

st.markdown("✅ City: Hyderabad \n🌫️ Air Quality: Good \n💡 Energy Usage: Moderate")

**🔹 report\_generator.py**

python

CopyEdit

import streamlit as st

from utils.api\_utils import post\_request

def generate\_report\_ui():

st.subheader("📈 Sustainability Report Generator")

city\_kpis = st.text\_area("Paste KPI Data (text format)")

if st.button("Generate Report"):

report = post\_request("/generate-report", {"kpi\_data": city\_kpis})

st.success(report["report"])

**🔹 api\_utils.py (Reusable API Calls)**

python

CopyEdit

import requests

BASE\_URL = "http://localhost:8000/api"

def post\_request(endpoint, data):

res = requests.post(f"{BASE\_URL}{endpoint}", json=data)

return res.json()

def get\_request(endpoint, params=None):

res = requests.get(f"{BASE\_URL}{endpoint}", params=params)

return res.json()

**✅ Milestone 5 Deliverables**

| **Component** | **Function** | **Status** |
| --- | --- | --- |
| smart\_dashboard.py | Main app UI | ✅ Done |
| chat\_assistant.py | Chat interface | ✅ Done |
| eco\_tips.py | Topic-based tip generator | ✅ Done |
| feedback\_form.py | Citizen input form | ✅ Done |
| summary\_card.py | KPI dashboard + policy summarizer | ✅ Done |
| report\_generator.py | Report generator UI | ✅ Done |
| api\_utils.py | REST API abstraction | ✅ Done |

**📌 Outcome:**

* Clean, modular **Streamlit interface**
* Live interaction with FastAPI backend
* Dynamic features like **chat**, **policy summarization**, **eco tips**, **feedback**, **reporting**, and **KPI dashboard**
* **Activity 5.1 & 5.2: Page Structure and UI Components**
* **✅ Activity 5.1: Page Structure with streamlit-option-menu**
* **🔹 Install the library (if not yet):**
* bash
* CopyEdit
* pip install streamlit-option-menu
* **📄 smart\_dashboard.py (Main UI entry)**
* python
* CopyEdit
* import streamlit as st
* from streamlit\_option\_menu import option\_menu
* from components import (
* summary\_card,
* feedback\_form,
* chat\_assistant,
* eco\_tips,
* policy\_summarizer,
* report\_generator,
* anomaly\_checker,
* kpi\_forecasting
* )
* st.set\_page\_config(page\_title="Smart City Assistant", layout="wide")
* # Sidebar navigation
* with st.sidebar:
* selected = option\_menu(
* menu\_title="Smart City Dashboard",
* options=["Dashboard", "Feedback", "Eco Tips", "Chat", "Policy Search", "Anomaly Checker", "KPI Forecasting"],
* icons=["bar-chart", "chat-dots", "leaf", "robot", "file-earmark-text", "exclamation-triangle", "graph-up"],
* default\_index=0,
* )
* # Page rendering logic
* if selected == "Dashboard":
* summary\_card.render\_dashboard()
* elif selected == "Feedback":
* feedback\_form.feedback\_ui()
* elif selected == "Eco Tips":
* eco\_tips.display\_tips()
* elif selected == "Chat":
* chat\_assistant.chat\_interface()
* elif selected == "Policy Search":
* policy\_summarizer.summarize\_policy()
* elif selected == "Anomaly Checker":
* anomaly\_checker.check\_anomalies()
* elif selected == "KPI Forecasting":
* kpi\_forecasting.forecast\_kpi()
* **✅ Activity 5.2: Build UI Components**
* **📄 summary\_card.py – Stylish KPI Boxes**
* python
* CopyEdit
* import streamlit as st
* def render\_dashboard():
* st.markdown("## 🌆 City KPIs - Live Snapshot")
* col1, col2, col3 = st.columns(3)
* col1.metric("Air Quality", "Good", "↑ 5%")
* col2.metric("Energy Usage", "Moderate", "- 3%")
* col3.metric("Water Consumption", "High", "↑ 10%")
* st.markdown("---")
* st.info("City: Hyderabad | Last Updated: Just now")
* **📄 chat\_assistant.py – Form for Chat Prompt**
* python
* CopyEdit
* import streamlit as st
* from utils.api\_utils import post\_request
* def chat\_interface():
* st.subheader("🧠 Ask About Sustainability")
* prompt = st.text\_input("Type your question:")
* if prompt and st.button("Ask"):
* response = post\_request("/chat", {"prompt": prompt})
* st.success(response["response"])
* **📄 feedback\_form.py – Feedback Form**
* python
* CopyEdit
* import streamlit as st
* from utils.api\_utils import post\_request
* def feedback\_ui():
* st.subheader("📣 Share Feedback")
* name = st.text\_input("Name")
* category = st.selectbox("Category", ["Transport", "Energy", "Water", "Other"])
* message = st.text\_area("Your Message")
* if st.button("Submit Feedback"):
* result = post\_request("/submit-feedback", {
* "name": name, "category": category, "message": message
* })
* st.success(result["message"])
* **📄 eco\_tips.py – Keyword Prompt and Result**
* python
* CopyEdit
* import streamlit as st
* from utils.api\_utils import get\_request
* def display\_tips():
* st.subheader("🌱 Get Eco-Friendly Tips")
* topic = st.text\_input("Enter Topic (e.g., water, waste):")
* if topic and st.button("Get Tip"):
* tip = get\_request("/get-eco-tips", {"topic": topic})
* st.info(tip["tip"])
* **📄 policy\_summarizer.py – Policy Text Input + Summary**
* python
* CopyEdit
* import streamlit as st
* from utils.api\_utils import post\_request
* def summarize\_policy():
* st.subheader("📄 Summarize Policy Document")
* text = st.text\_area("Paste policy text:")
* if text and st.button("Summarize"):
* result = post\_request("/upload-doc", {"text": text})
* st.success(result["summary"])
* **📄 report\_generator.py – PDF Sustainability Report (Text)**
* python
* CopyEdit
* import streamlit as st
* from utils.api\_utils import post\_request
* def generate\_report\_ui():
* st.subheader("📈 Sustainability Report")
* kpi\_data = st.text\_area("Enter KPI data (text)")
* if st.button("Generate Report"):
* report = post\_request("/generate-report", {"kpi\_data": kpi\_data})
* st.success(report["report"])
* **📄 anomaly\_checker.py – Placeholder for Anomaly Feature**
* python
* CopyEdit
* import streamlit as st
* def check\_anomalies():
* st.subheader("🚨 Anomaly Detection")
* st.warning("Coming Soon: Upload KPI .csv file to check for anomalies.")
* **📄 kpi\_forecasting.py – Placeholder for Forecasting Feature**
* python
* CopyEdit
* import streamlit as st
* def forecast\_kpi():
* st.subheader("📊 KPI Forecasting")
* st.warning("Coming Soon: Visualize predicted trends based on uploaded data.")
* **📌 Deliverables Summary**

| **Component** | **Description** | **Status** |
| --- | --- | --- |
| Sidebar Navigation | Using streamlit-option-menu | ✅ Done |
| smart\_dashboard.py | Page controller | ✅ Done |
| summary\_card.py | KPI display | ✅ Done |
| chat\_assistant.py | Chat UI | ✅ Done |
| feedback\_form.py | Feedback input form | ✅ Done |
| eco\_tips.py | Topic prompt → tip | ✅ Done |
| policy\_summarizer.py | Text → summary | ✅ Done |
| report\_generator.py | KPI → sustainability report | ✅ Done |
| anomaly\_checker.py | Placeholder | ✅ Done |
| kpi\_forecasting.py | Placeholder | ✅ Done |