App.py

import streamlit as st

import folium

from streamlit\_folium import st\_folium

from sqlalchemy.orm import Session

import time

import yaml

from yaml.loader import SafeLoader

from streamlit\_authenticator import Authenticate

from models.base import get\_db, engine

from models.landmarks import Landmark

from models.transportation import TransportRoute

from data.landmarks import FAMAGUSTA\_CENTER

from utils.map\_utils import create\_landmark\_map, add\_landmark\_markers, filter\_landmarks

from utils.schedule\_utils import load\_bus\_schedules, display\_schedule\_table

from utils.filter\_utils import filter\_landmarks\_by\_distance, filter\_landmarks\_by\_type, filter\_landmarks\_by\_search

from utils.route\_utils import create\_route\_planner, add\_route\_to\_map

from database.init\_db import init\_database

# Initialize database on startup

init\_database()

# 🔵 Mock Bus Movement (Simulating real-time tracking)

BUS\_ROUTE = [

(35.1234, 33.9456), (35.1240, 33.9460), (35.1245, 33.9470),

(35.1250, 33.9480), (35.1255, 33.9490), (35.1260, 33.9500)

]

if "bus\_position\_index" not in st.session\_state:

st.session\_state.bus\_position\_index = 0

def update\_bus\_position():

"""Simulate real-time bus movement."""

st.session\_state.bus\_position\_index = (st.session\_state.bus\_position\_index + 1) % len(BUS\_ROUTE)

return BUS\_ROUTE[st.session\_state.bus\_position\_index]

# 🔵 Authentication Setup

with open("config.yaml") as file:

config = yaml.load(file, Loader=SafeLoader)

authenticator = Authenticate(

config["credentials"],

config["cookie"],

config["key"],

config["expiry\_days"]

)

# 🔵 Fetch Landmarks with Caching

def get\_landmarks(db: Session, search\_term: str = "", limit: int = 50) -> list:

"""Fetch landmarks with search filter and limit results."""

query = db.query(Landmark)

if search\_term:

search = f"%{search\_term.lower()}%"

query = query.filter(

(Landmark.name.ilike(search)) |

(Landmark.description.ilike(search))

)

return [landmark.to\_dict() for landmark in query.limit(limit).all()]

# 🔵 Main Streamlit App

def main():

st.set\_page\_config(

page\_title="Famagusta Bus System",

page\_icon="🚌",

layout="wide",

initial\_sidebar\_state="expanded"

)

# 🔵 Dark Mode Toggle

dark\_mode = st.sidebar.toggle("Dark Mode", value=False)

theme = "dark" if dark\_mode else "light"

st.markdown(f"""

<style>

body {{ background-color: {'#121212' if dark\_mode else '#FFFFFF'}; color: {'#FFFFFF' if dark\_mode else '#000000'} }}

</style>

""", unsafe\_allow\_html=True)

# 🔵 Authentication

name, authentication\_status, username = authenticator.login("Login", "main")

if authentication\_status:

authenticator.logout("Logout", "sidebar")

st.sidebar.write(f"Welcome \*{name}\*")

st.markdown("<h1 style='text-align: center; color: #1E88E5;'>Famagusta Bus System</h1>", unsafe\_allow\_html=True)

# 🔵 Tabs

tab3, tab2, tab1, tab4 = st.tabs(["Bus Schedules", "Route Planner", "Landmarks", "Live Bus Tracking"])

try:

# Fetch Database Session

db = next(get\_db())

# Cache landmarks in session state

if "landmarks" not in st.session\_state:

st.session\_state.landmarks = get\_landmarks(db)

landmarks = st.session\_state.landmarks

# 🔵 Live Bus Tracking

with tab4:

st.subheader("Live Bus Tracking")

col1, col2 = st.columns([3, 1])

with col1:

bus\_map = create\_landmark\_map(FAMAGUSTA\_CENTER["latitude"], FAMAGUSTA\_CENTER["longitude"])

bus\_lat, bus\_lon = update\_bus\_position()

folium.Marker((bus\_lat, bus\_lon), tooltip="Bus Location", icon=folium.Icon(color='red')).add\_to(bus\_map)

st\_folium(bus\_map, width=800)

with col2:

st.info(f"Current Bus Location: ({bus\_lat}, {bus\_lon})")

time.sleep(2) # Simulate movement

st.experimental\_rerun()

except Exception as e:

st.error(f"An error occurred: {str(e)}")

elif authentication\_status is False:

st.error("Username or password is incorrect")

elif authentication\_status is None:

st.warning("Please enter your username and password")

if \_\_name\_\_ == "\_\_main\_\_":

main()