README: Pedestrian Traffic Simulation – KSU Marietta Campus

1 Project Overview

This project simulates pedestrian traffic on the Kennesaw State University Marietta campus using the SUMO (Simulation of Urban MObility) simulator. Pedestrian flows are influenced by building occupancy schedules stored in a PostgreSQL database. The simulation outputs pedestrian density statistics and time-based movement data.

2 Dependencies Installation

2.1 Install SUMO (Windows)

1. Download SUMO from: `https://www.eclipse.dev/sumo/download/`
2. Extract the archive to `C:\SUMO`
3. Add `C:\SUMO\bin` to your system PATH:
   - Control Panel → System → Advanced → Environment Variables → Path → New → Paste the path
4. Verify installation:

```
sumo --version
```

2.2 Install Python (Windows)

1. Download Python 3.10–3.12 from `https://www.python.org/downloads/windows/`
2. During setup:
   - Check "Add Python to PATH"
   - Select "Customize installation" and ensure pip is selected
3. Verify installation:

```
python --version
pip --version
```

2.3 Create a Virtual Environment

```
python -m venv sim-env
sim-env\Scripts\activate
```

2.4 Install Python Dependencies

```
pip install traci psycopg2 pandas numpy
```

2.5 Install PostgreSQL (Windows)

1. Download and install PostgreSQL 15+ from `https://www.postgresql.org/download/windows/`
2. Launch pgAdmin and create a database named `mobinav`
3. Use the following connection parameters:

```
Host: localhost
Port: 5432
Username: postgres
Password: [your_password]
```

2.6 Create Required Tables

Run this schema in pgAdmin or the psql terminal:

```
CREATE TABLE building (
    building_id SERIAL PRIMARY KEY,
    name TEXT,
    address TEXT,
    street TEXT,
    state TEXT,
    zipcode TEXT,
    latitude NUMERIC(9,6),
    longitude NUMERIC(9,6),
    updated_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE occupancy_schedule (
    schedule_id SERIAL PRIMARY KEY,
    building_id INTEGER REFERENCES building(building_id),
    occupancy_value INTEGER,
    day TEXT,
    start_time TIME,
```

```
    end_time TIME,
    updated_at TIMESTAMP DEFAULT NOW()
);
```

2.7 Insert Test Data

```
INSERT INTO building (name, latitude, longitude)
VALUES
  ('Atrium Building', 33.9416, -84.5203),
  ('Academic Building', 33.9420, -84.5195);

INSERT INTO occupancy_schedule (building_id, occupancy_value, day, start_time
VALUES
  (1, 300, 'MWF', '08:00', '10:00'),
  (2, 200, 'MWF', '08:00', '10:00');
```

3 Running the Simulation

3.1 Configure the Simulation config.py

Edit `config.py` and set:

```
SUMO_CONFIG_FILE = "KSUMariettaConfig.sumocfg"
NET_FILE = "KSUMarietta.net.xml.gz"
SIMULATION_STEPS = 86400  # One full day
congested_lanes = set()  # set global congested lanes variable
stats_type = "MWF" # "day" value entered in occupancy_schedule
DB_NAME = "mobinav"
DB_USER = "postgres"
DB_PASS = "your_password"
DB_HOST = "localhost"
DB_PORT = "5432"
```

3.2 Configure the Simulation main.py

Edit `main.py` and set:

```
NUM_RUNS = 1 # set for number of runs desired to run automatically
```

3.3 Run the Simulation

From your activated environment and project root:

```
python main.py
```

3.4 Simulation Output

- `simulaion_stats_`*day*`_run_`*run number*`.csv` – contains timestamped pedestrian metrics
- Console logs showing:
  - Start of simulation
  - Number of pedestrians generated
  - Pedestrians skipped due to invalid edges or no route
  - Pedestrians re-routed due to congestion
  - End-of-run status

4 System Requirements

- **Operating System:** Windows 10+ or Linux
- **CPU:** Dual-core processor or better
- **RAM:** 8 GB recommended
- **Disk Space:** 500 MB free minimum
- **Python:** 3.10–3.12
- **SUMO:** 1.19.0 or higher
- **PostgreSQL:** Version 15 or later

5 Project Structure

```
main.py                        # Entry point
simulation.py                  # SUMO control and data logging
pedestrians.py                 # Pedestrian logic
routing.py                     # Routing and strategy
get_buildings_list.py          # Building utility
get_coordinates_occupancy.py# Occupancy and GPS mapping
config.py                      # Configuration parameters
KSUMariettaConfig.sumocfg      # SUMO simulation config
KSUMarietta.net.xml.gz         # Network file
*.xml, *.cfg                   # SUMO support files
```

6 Troubleshooting

- `ModuleNotFoundError: No module named 'traci'` → Run `pip install traci`
- `psycopg2.OperationalError` → Check database credentials and connection
- SUMO command not recognized → Ensure `sumo` is added to your PATH
- No CSV output → Check that simulation completes and paths in `config.py` are valid

7 Support

If you encounter issues, contact the simulation team via Microsoft Teams or email. Include logs, screenshots, or SQL output to help diagnose the problem.