# Multi Agent Systems Project


***Completed by:***

**Heni WALHA**
**Anis BOUHAMED**


---

***Subject:***

*** Public Transport Mobility Simulation***

---

***Academic Year:***

**2023 - 2024**

# ACKNOWLEDGMENTS:

We express our heartfelt gratitude to Dr. Zargayouna Mahdi, whose guidance and expertise have been instrumental in shaping the vision and execution of the multi agent systems' project. Dr. Mahdi's unwavering support and insightful feedback have been a beacon throughout this short journey.

We extend our sincere appreciation to the entire team involved in the development of this project. Each team member has contributed their unique skills and dedication, creating a collaborative environment that fostered innovation and excellence. This project stands as a testament to the collective effort and commitment of the team. Thank you all for your invaluable contributions and unwavering support.

# TABLE OF CONTENT

- Contents

# TABLE OF CONTENT

Chapter

# 1 - INTRODUCTION

Hey there! Ever wondered how buses and people navigate a city? Our project is like a cool video game where we get to make a mini city with roads and bus stops and see how passengers and buses move around. But hold on, we're not just doing the usual stuff – we're adding some surprises!

Think of it like this: our mini city has roads going all over, and bus stops in just the right spots. We've got passengers, like friends going places, some walking and some catching buses that have their own schedules, like a movie script. But here's where it gets exciting – we're throwing in problems, like a road suddenly being blocked or a bus taking a break. It's a bit crazy, but that's what makes our virtual city feel alive! And the best part? We want to see how all this craziness affects how long it takes for people to get where they're going.

Our goal? To set up our mini city by figuring out the rules, drawing our city map, giving life to the buses, and deciding how everyone moves around.

This isn't your typical school project – it's a fun trip into how buses and people mix and mingle in the city. So, get ready for the ride as we dive into the lively world of public transport – where it's not just about roads and stops, but about buses, roads, and the folks they carry through the twists and turns of city life!

# 2 SYSTEM DESIGN

The code is divided into three main classes:

## 2.1 City Class:

**Attributes:**

**size**: Represents the size of the city grid.
**roads**: 2D list representing the city grid.
**bus_stops**: List of randomly generated bus stops within the city.
**inaccessible_routes**: Set of routes that are currently blocked.

## 2.2 PublicTransportVehicle Class:

**Attributes:**

**route**: List representing the bus route (sequence of bus stops).
**next_stop**: Index indicating the next bus stop on the route.
**position**: Current position of the bus on the city grid.
**inaccessible_routes:** Set of routes the bus cannot traverse.
**passengers**: List of passengers currently on the bus.

**Methods:**

move: Moves the bus towards the next stop, considering obstacles and alternative routes.

## 2.3 Passenger Class:

**Attributes:**

**origin**: Starting position of the passenger.
**destination**: Desired destination of the passenger.
**position**: Current position of the passenger.
**inaccessible_routes**: Set of routes the passenger cannot traverse.
**BusTaken:** Flag indicating if the passenger has taken a bus.
**BusStartPosition:** Starting position of the bus the passenger took.
**Bus:** Reference to the bus taken by the passenger.
**foot_steps:** Number of steps taken by foot.
vehicle_steps: Number of steps taken using public transportation.

**Methods:**

move: Moves the passenger towards the destination, considering obstacles and alternative routes.

# 3 TECHNOLOGIES USED

## 3.1 Python:

Our programming language of choice, Python, served as the backbone for developing the entire simulation. Known for its readability and versatility, Python enabled us to implement the complex logic underlying the movements of passengers and public transport vehicles. Its extensive libraries also facilitated efficient data manipulation and algorithm implementation.

## 3.2 Matplotlib:

For the visual aspect of our simulation, we harnessed the power of Matplotlib, a popular plotting library in the Python ecosystem. Matplotlib allowed us to create vibrant and dynamic visualizations, turning the abstract movements of agents into tangible graphics. This not only enhanced our understanding of the simulation but also provided a user-friendly interface for conveying the project's dynamics.

## 3.3 Flask:

Flask, a lightweight and versatile web framework, was instrumental in building an interactive user interface. We leveraged Flask to create a web application that allows users to engage with and observe the simulation in real-time. This not only enhanced the user experience but also facilitated easy deployment, making our simulation accessible to a broader audience

## 3.4 Machine Learning Model:

Adding a touch of machine learning sophistication, we incorporated scikit-learn (sklearn) into our toolkit. This component added a layer of predictive analytics to our simulation, offering insights into potential outcomes under different scenarios.

This machine learning model employs XGBoost to predict traffic volume intensity, integrating factors like rain, time of day, and historical patterns. Trained on a comprehensive dataset, the model's ensemble of decision trees captures intricate relationships, offering accurate predictions for urban traffic management. With a focus on multi-agent systems, this model enhances decision-making by providing insights into anticipated traffic conditions, enabling agents to plan optimal routes and activities based on real-time predictions.

## 3.5 Conclusion

By seamlessly integrating Python, Matplotlib, Flask, and scikit-learn, our technology stack brought synergy to the project, transforming it from lines of code into a dynamic, interactive, and insightful simulation of urban mobility. Each technology played a distinct role, contributing to the holistic and engaging nature of our endeavor.

Chapter

# 4

# - IMPLEMENTATION

The simulation is orchestrated by the simulate function. It involves the creation of a city, buses with random routes, and passengers with random origins and destinations. The simulation runs for a specified number of steps.

## 4.1 City Initialization:

Random bus stops are generated within the city.
An empty set is created for inaccessible routes.

## 4.2 Vehicle Initialization:

Buses are created with random starting positions and routes.
The route is a random permutation of the bus stops.
Buses are added to the list of vehicles.

## 4.3 Passenger Initialization:

Passengers are created with random origins and destinations.
Passengers are added to the list of passengers.

## 4.4 Simulation Steps:

Disturbances are introduced by blocking a random route in the city.
Vehicles (buses) move within the city, considering obstacles and picking up/dropping off passengers.
Passengers move towards their destinations, considering obstacles and deciding whether to take public transportation.
The simulation pauses for a specified duration to visualize the changes in the city.
Output is cleared to display the updated information.

## 4.5 Visualization:

The simulation is enhanced with a visual representation using the matplotlib library. The plot_city function is responsible for creating a dynamic plot that illustrates the city grid, bus stops, road blockages, vehicles, and passengers. Each step of the simulation is displayed, providing a visual understanding of the transportation system's dynamics.

Chapter 5 - CHALLENGES FACED

## 5.2 Real-Time Plot Updates on Web Interface:

### 5.2.1 Description:

Enabling the web interface to dynamically update and reflect the evolving state of the simulation in real-time posed a significant challenge. Traditional approaches struggled to provide instantaneous updates.

### 5.2.2 Solution:

Leveraging Socket.IO, we established a bidirectional communication channel between the server and the web interface. This technology allowed for seamless real-time updates, ensuring that users could witness the dynamic movements of vehicles and passengers as the simulation progressed, creating a more engaging and interactive user experience.

## 5.3 Ensuring Clear Visualization for Users

### 5.3.1 Description:

As the simulation unfolded, clarity in visual representation became crucial. The rapid succession of movements required a mechanism to pause and display the evolving state at each step, enhancing user comprehension.

### 5.3.2 Solution:

To address this challenge, we employed the plt.pause function from Matplotlib. This technique introduced pauses at strategic points in each iteration, allowing the visualization to catch up and providing users with a clear, step-by-step presentation of the simulation's progression. This ensured that the changes were not only real-time but also easily digestible for users interacting with the web interface.

## 5.4 Dynamic Movement of vehicles and Passengers towards Destinations:

### 5.4.1 Description:

Implementing a mechanism to ensure that vehicles and passengers consistently moved toward their respective destinations required a thoughtful approach. Achieving this in a dynamic and responsive manner posed a unique set of challenges.

### 5.4.2 Solution:

We adopted a directional movement strategy, where each vehicle and passenger recalculated their paths in each iteration based on their current position and the coordinates of their destinations.

Chapter

# 6

# - ADDITIONAL FUNCTIONALITIES:

## 6.1 Flask Application for Enhanced Interaction:

### 6.1.1 Description:

In our pursuit of a more engaging user experience, we integrated a Flask web application into our project. This not only facilitated real-time interaction with the simulation but also allowed users to visualize and comprehend the intricate movements of passengers and vehicles. The Flask app acted as a bridge between the simulation engine and the user interface, offering a seamless and accessible way for users to interact with the dynamics of the simulated city.

## 6.2 Machine Learning Model for Traffic Intensity Prediction

### 6.2.1 Description:

Taking a step beyond the core simulation, we introduced a machine learning model dedicated to predicting traffic intensity. This additional layer of sophistication allowed us to analyze and forecast the impact of disturbances on traffic flow. Leveraging scikit-learn, our model tapped into historical data generated by the simulation to provide insights into potential traffic patterns under varying conditions. This predictive aspect enhances the project's analytical capabilities, offering users a glimpse into the potential consequences of disruptions on urban traffic .

# - CONCLUSION

In wrapping up our academic venture into simulating public transport mobility, we've found that even simple projects can unfold a world of insights. Our aim was straightforward: to create a user-friendly platform that captures the essence of urban transportation systems.

The Flask web app proved to be a handy addition, making our simulation not just a code-based exercise but an interactive experience. Users could now observe, in real-time, the comings and goings of passengers and vehicles in our virtual city.

On top of that, we added a sprinkle of sophistication with a machine learning model. Nothing too fancy, just a tool to predict traffic intensity. It's a modest touch that opens up possibilities for analyzing the impact of disruptions on city traffic.

As we bring this project to a close, we acknowledge its simplicity. It might not be the most groundbreaking endeavor, but it's a small step into the world of urban mobility simulations. We've learned, we've coded, and we've seen a little bit more into the complexities of public transportation in cities. In the realm of academic projects, it might be a small fish, but every fish adds something to the pond.

Our gratitude extends to Dr. Zargayouna Mahdi, our team, and all involved. Together, we've sown seeds of innovation and commitment to an educational journey transcending boundaries. We envision a future where technology empowers education, fostering a supportive ecosystem.