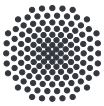


**University of Stuttgart**
Institute of Industrial Automation and
Software Engineering

**Research Project 3265**

Dynamic Geofences for Autonomous Transport Robots in Indoor Logistics

**Sowjanya Krishna**

# User Manual

Examiner: Prof. Dr.-Ing. Dr: h.c. Michael Weyrich
Supervisor: Hannes Vietz, M. Sc

**Document Version Management**

| Version | Author | QA | Date | Status | Changes |
|---|---|---|---|---|---|
| **0.1** | Sowjanya | Sa | 23.05.2022 | in progress | Creation |

# 0 Table of Contents

# 1 Installation Guide

## 1.1 Backend Installation

*syn_back* folder is the backend of the Dynamic Geofence Application, implemented in flask microframework. Below mentioned figure shows the overview of it.

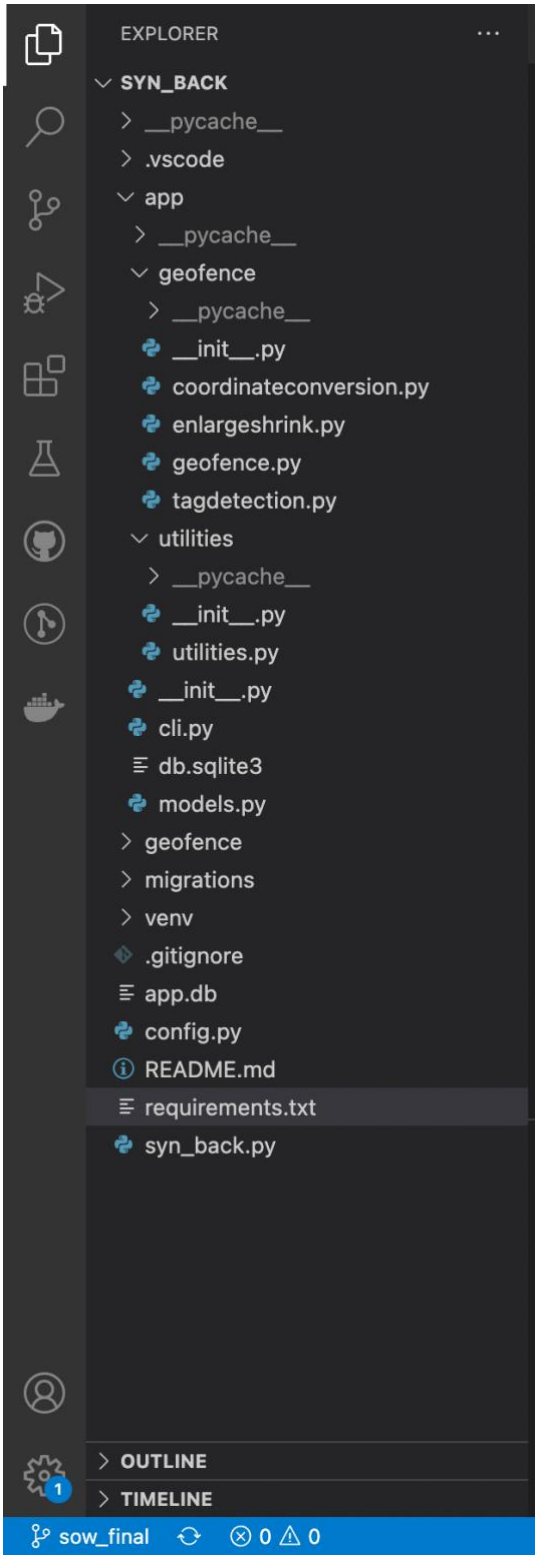



**Figure 1. Folder structure of syn_back**

Above mentioned software can be installed on a PC by following below mentioned steps:

1. Clone the source code from GitHub by cloning the repository or downloading the .zip file (link)
2. Open the application in any chosen IDE (Visual studio code or pycharm)
3. Open the terminal (on a chosen IDE) and run the command:

   ```
   pip install -r requirements.txt
   ```

   This installs all the required packages mentioned in the *requirements.txt* file.
4. The virtual environment files are stored in the name venv and can be found in the project directory.
5. The system needs to use this virtual environment. To activate the virtual environment, use the following command:
   For linux machine-

   ```
   $ source *syn_back folder location*/venv/bin/activate
   ```

   Eg: $ source /Users/'Username'/Downloads/syn_back/venv/bin/activate

   For windows machine-

   ```
   $ *syn_back folder location* \venv\Scripts\activate
   ```

   Eg: C:\Users\'Username'\venv\Scripts\activate
6. Flask needs to be told how to import it, by setting the FLASK_APP environment variable:

   ```
   (venv)$ export FLASK_APP=syn_back.py
   ```

## 1.2 Frontend Installation

*syn_front* folder is the frontend of the Dynamic Geofence Application, implemented in vue.js. Below mentioned figure shows the overview of it.

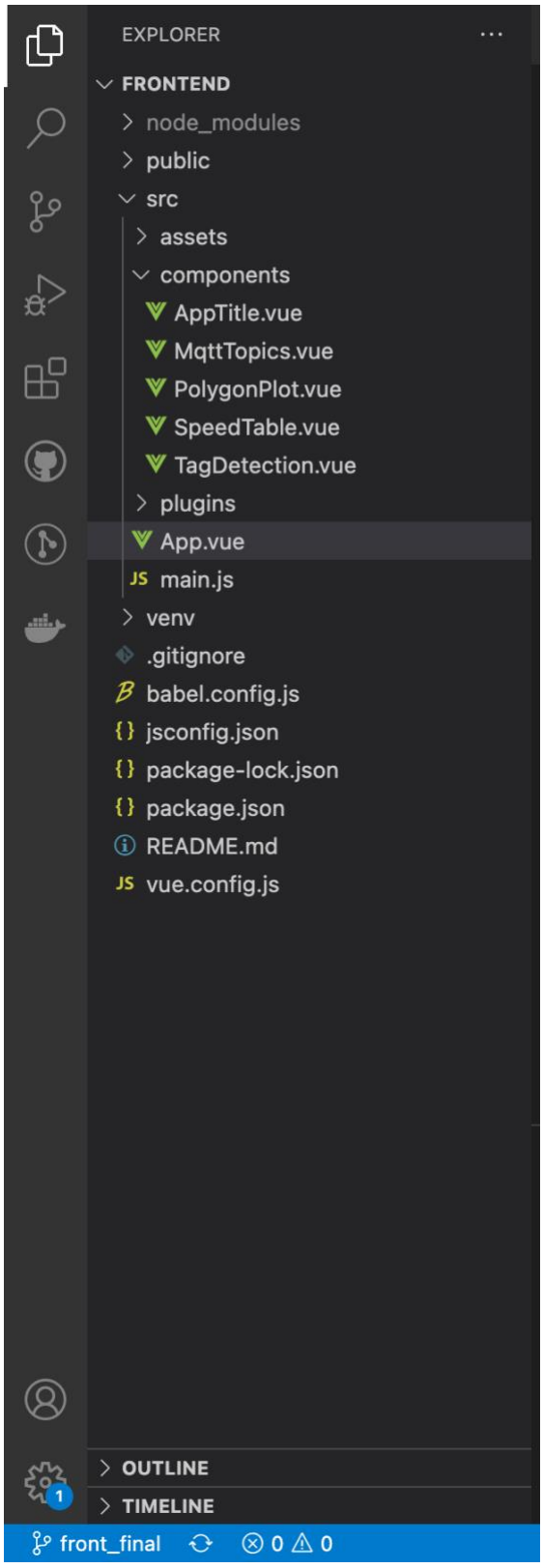



**Figure 2. Folder structure of frontend**

Above mentioned software can be installed on a PC by following below mentioned steps:

1. Clone the source code from GitHub by cloning the repository or downloading the .zip file (link)
2. Open the application in any chosen IDE (Visual studio code or pycharm)
3. To configure the app, navigate to the terminal and install the required npm packages.

```
npm install
```

# 2 Start-up

## 2.1 NAiSE GUI [1]

Once the backend(syn_back) and frontend(syn_front) are installed successfully with the necessary configurations. The next step is to connect to the NAiSE GUI in order to visualize the ARENA map, AGV's and the geofence operation in real-time.

The below mentioned steps need to be followed to connect to the NAiSE GUI:

1. To open the GUI on your PC, the user must type the IP-address in the browser. The devices must be connected to the NAiSE network.

```
http://192.168.0.147/login/
```

2. Once the application is running properly, the login page of the GUI will appear (seen in figure 3)




**Figure 3:Login-page of NAiSE GUI**

Add the credentials as mentioned below in the login page.

```
Username: naise
Password: NAiSEisnice
```

3. After adding in the necessary credentials, you can view the NAiSE GUI as seen in the figure 4. To access the geofence interface in the mapping mode, click on the third button on the top left part of the GUI for opening the ‘map-editor’.

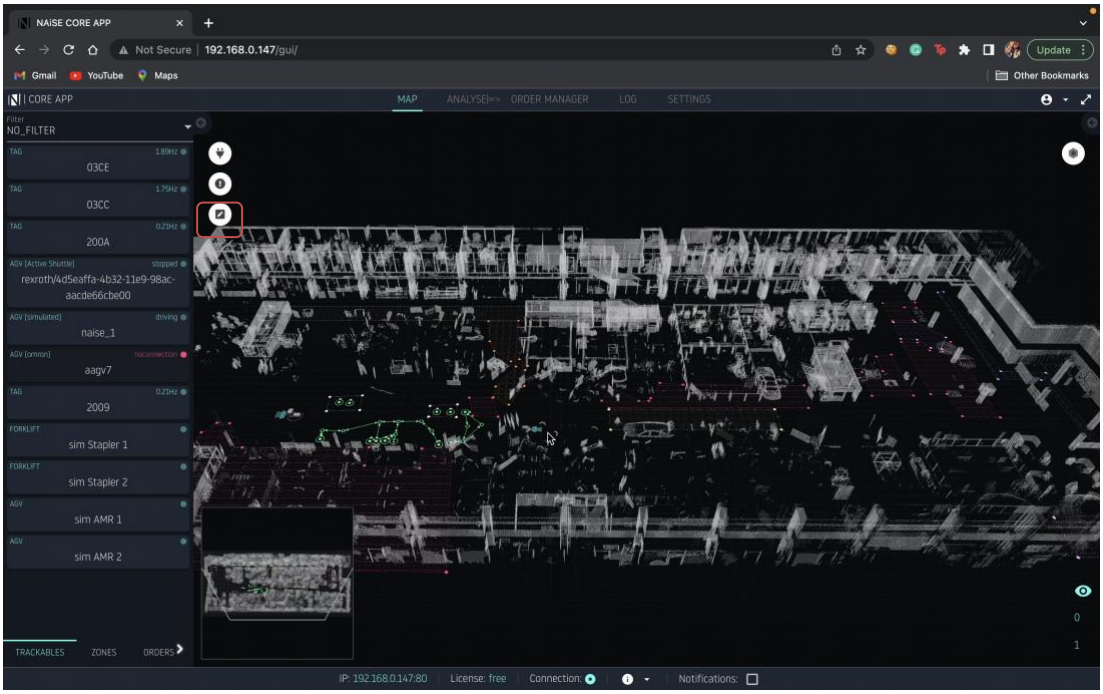





**Figure 4: Graphical User Interface of NAiSE Application**

4. When in mapping mode, the corner of the app will turn green smoothly changing its light intensity while on the top "edit mode" will be written as seen in figure 5. The user should select the *GEOFENCE* (highlighted in red box in figure 5).

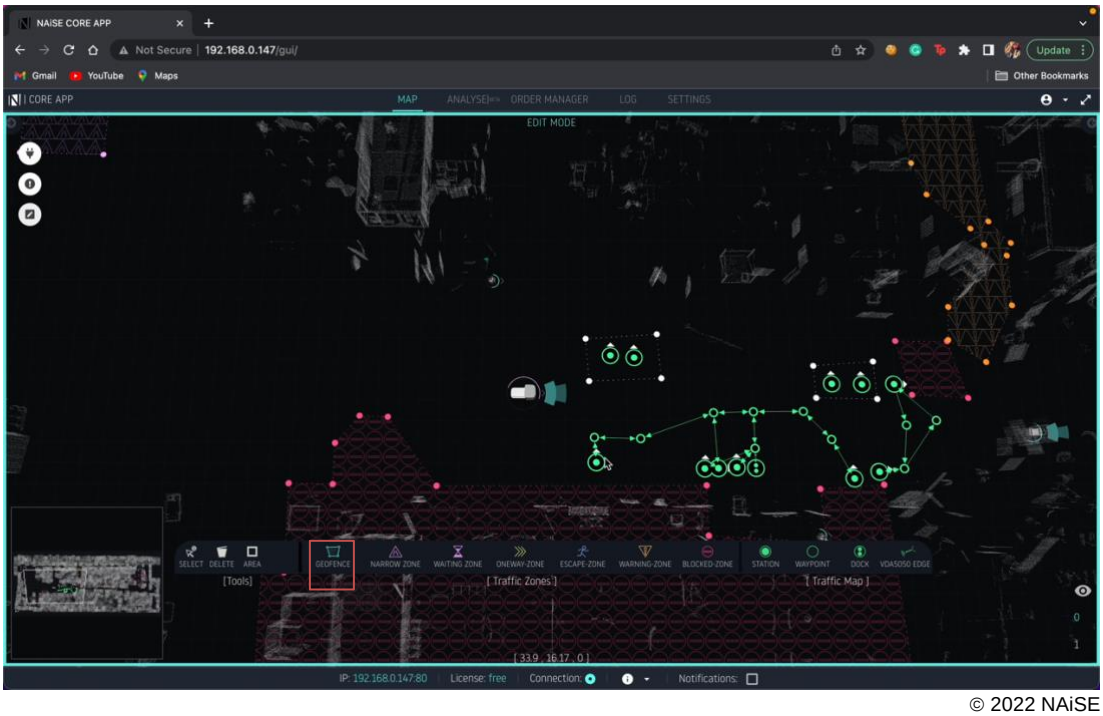





**Figure 5: NAiSE GUI Edit Mode**

5. The user can define any number of points for defining a geofence area around the AGV as seen in the figure 6. For closing the geofence, you can either click twice in the last point or close the loop by clicking back on the first point.

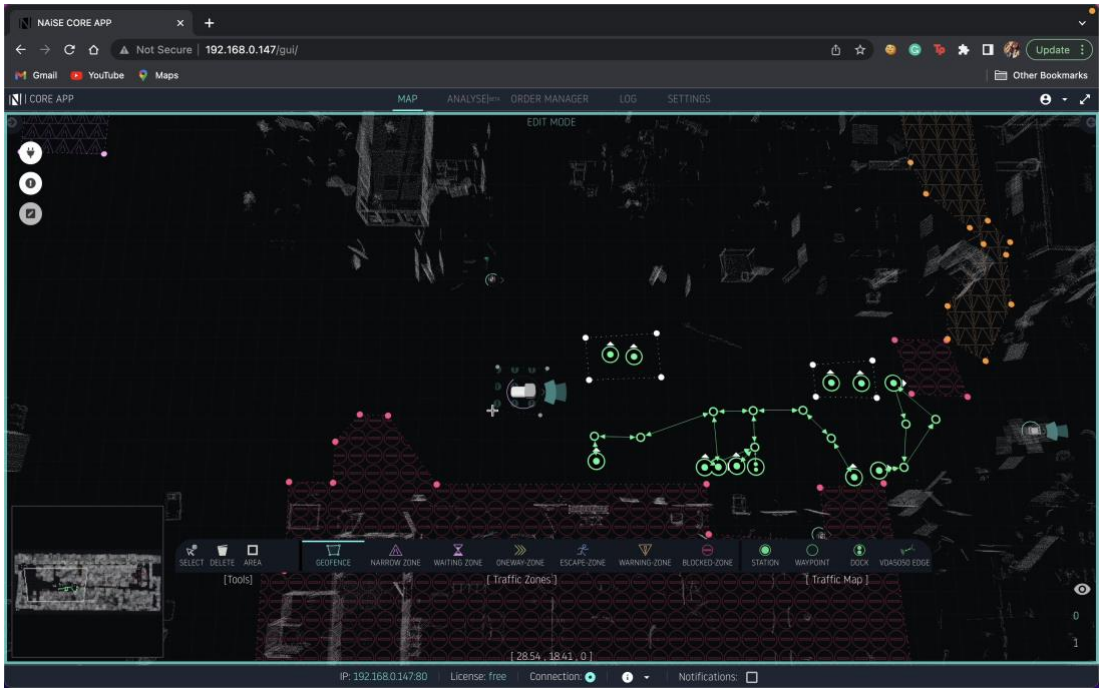





**Figure 6: Draw Geofence around the AGV**

6. Once a closed polygon (i.e,geofence) of any shape and any number of points is drawn around the AGV, a new window will pop up, in which the user needs to enter a name to the geofence-area as seen in the figure 7.
   Note: Keep a record of the AGV name that the geofence is drawn around, as well as the name of the geofence, this information needs to be added in the syn_front frontend page(refer to section 2.3 step 7).

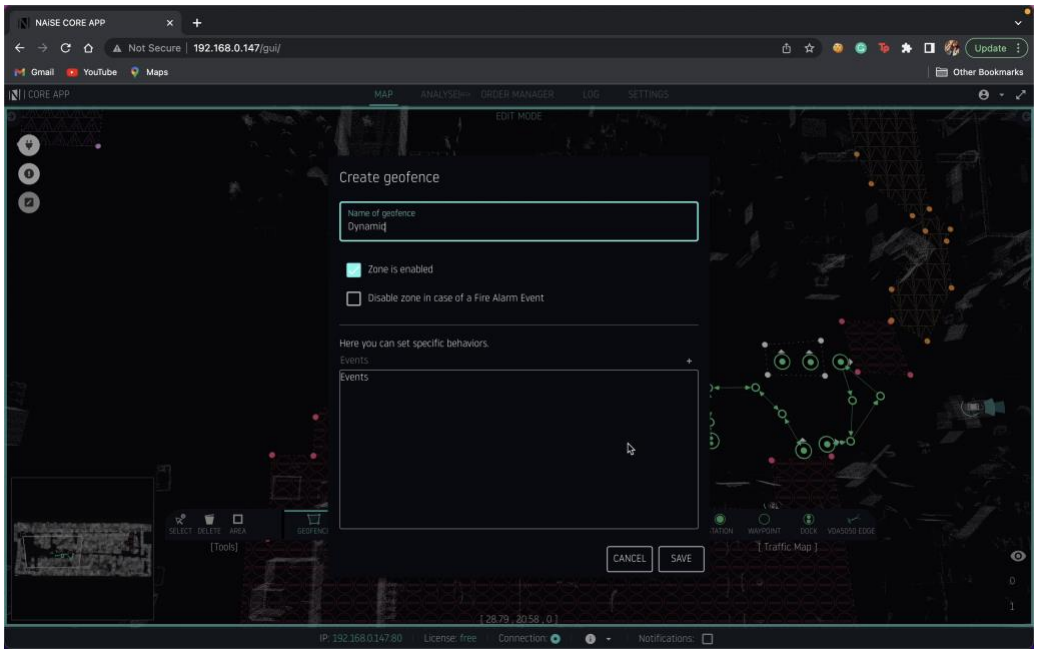





**Figure 7: Naming the Geofence-Area**

7. After creating a geofence, next step is to come out of the editing mode, this can be done by click again on the “map-editor” as mentioned in step 3. More information regarding the *Trackable* items (AGV and Tags) and geofence *Zones* can be found on the left section of the NAiSE GUI as seen in Figure 8.

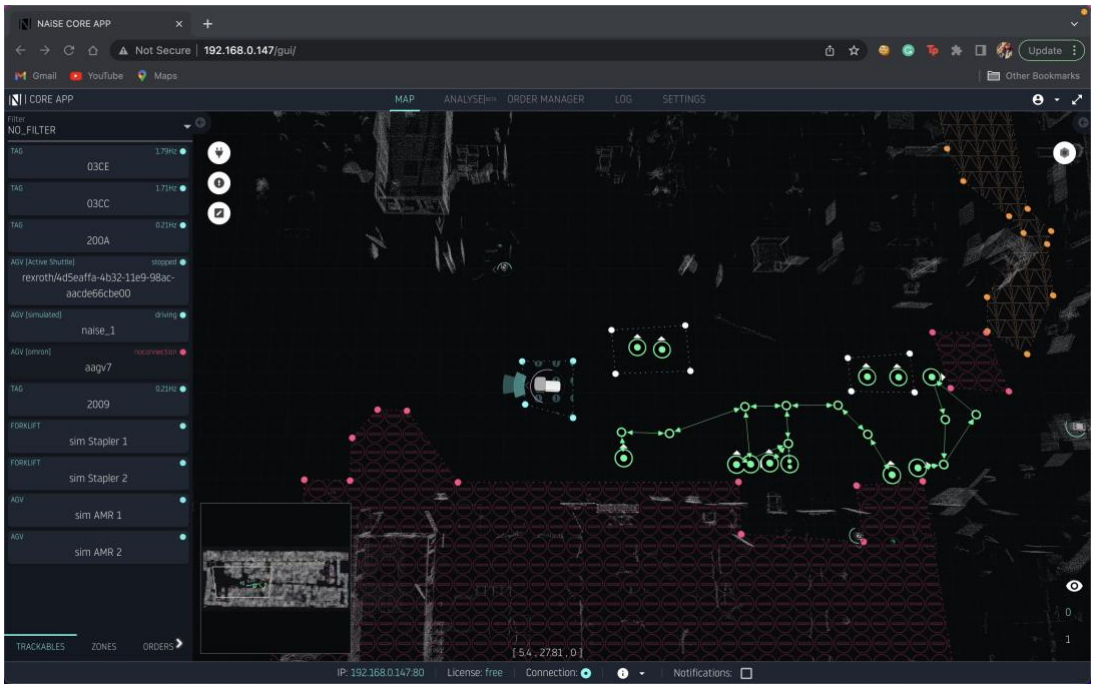





**Figure 8:NAiSE GUI main page**

## 2.2 Syn_back

Once the necessary configurations are done on the NAiSE GUI as mention in the section 2.1, the next step is to run our backend so it can communicate to the frontend, mosquitto broker and NAiSE server.

The below mentioned steps must be followed to run the backend:

1. For successful installation of the backend module refer to section 1.
2. Run the command in the terminal after activating the virtual environment.

```
(venv)$ flask run
```

3. The application will be running on *http://127.0.0.1:5000* as shown in figure 9.

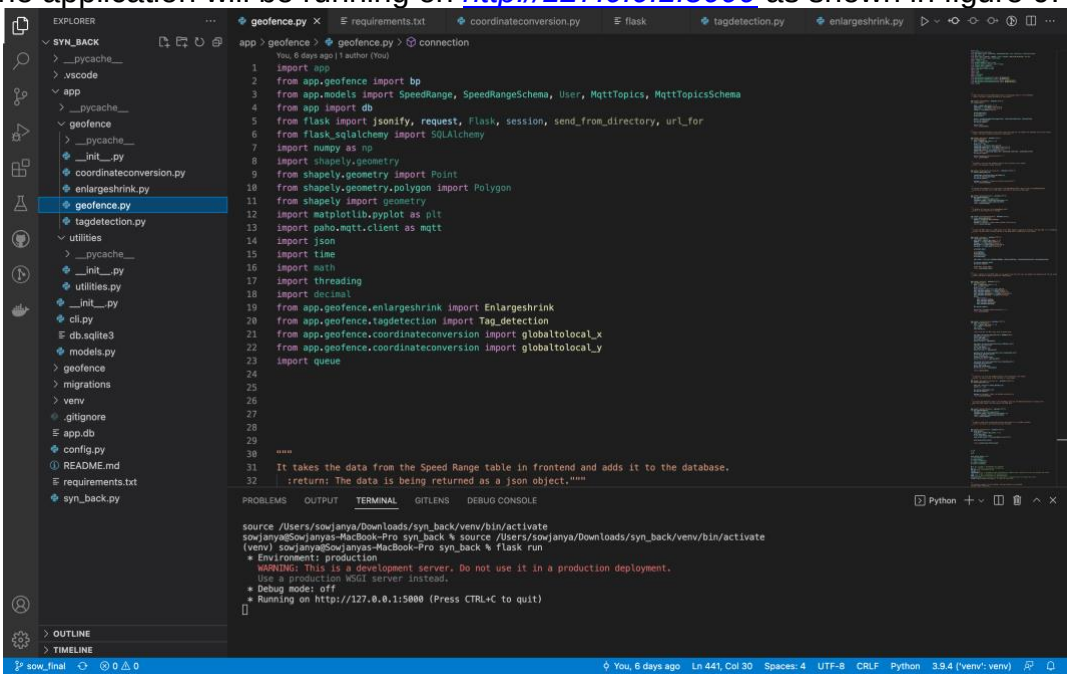



**Figure 9: Running syn_back from Terminal**

## 2.3 Syn_front

Now that the backend is up and running, the next step is to run the frontend application and to add the necessary information.

The below mentioned steps must be followed to run the frontend:

1. For successful installation of the frontend module refer to section 2
2. To run the app, start a Node development server. The application will open in the browser at http://localhost:8080 as seen in the figure 10.

```
npm run serve
```

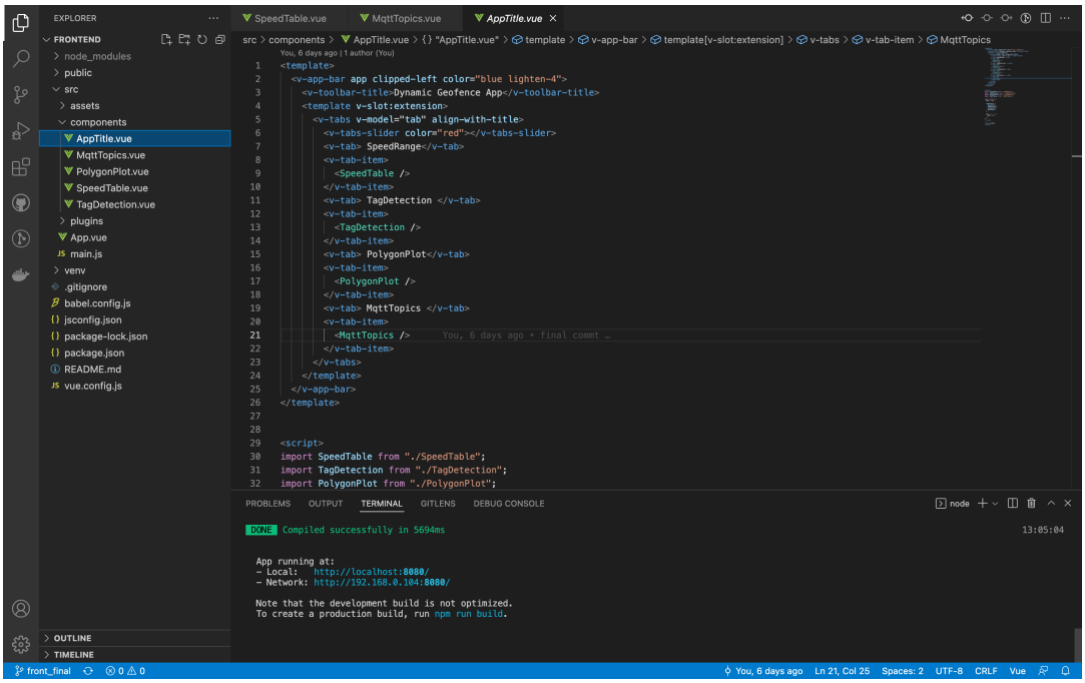



**Figure 10: Running syn_front fromTerminal**

3. Open the *Dynamic Geofence Application* in a web browser. This application consists of four tabs: 1)SpeedRange 2)Tag Detection 3)Polygon Plot 4)MQTT Topics as seen in figure 11.

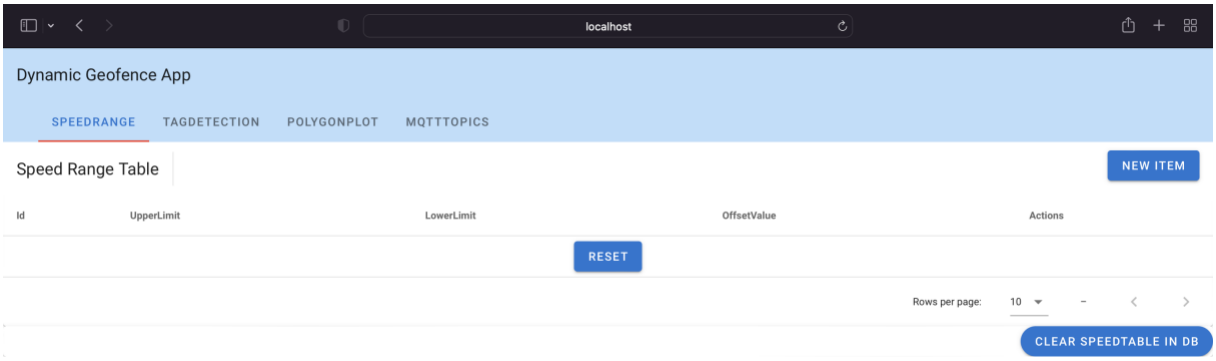



**Figure 11: Dynamic Geofence App**

4. Firstly, the user needs to add data into the *Speed Range Table.* By clicking on *NEW ITEM* on the top right corner of the page, a dialogue box will pop up where the user needs to add the speed range values as seen in figure 12 and click on save button to store the entry in the database.

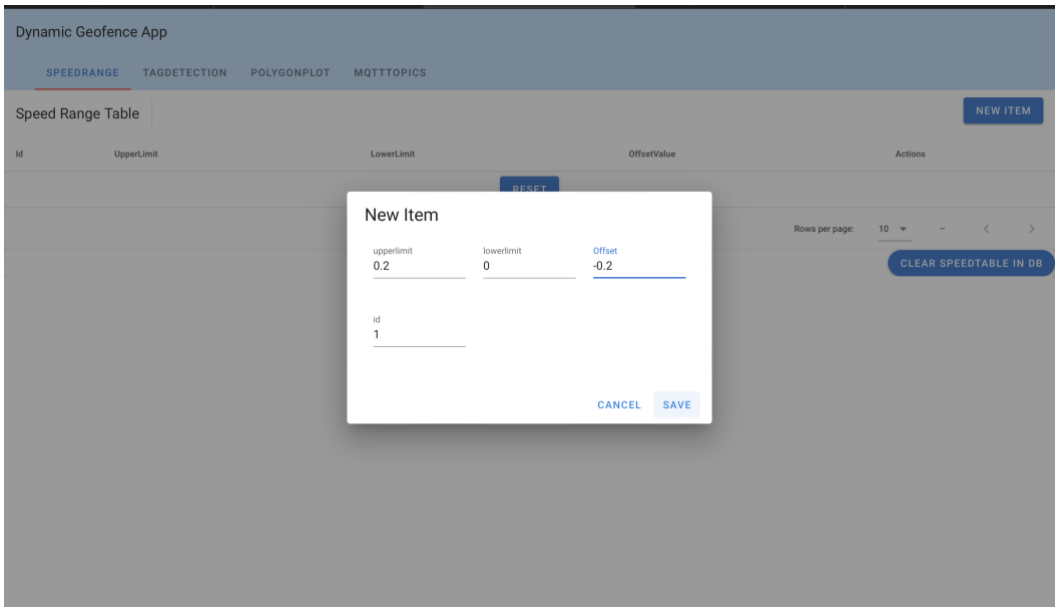



**Figure 12: Speed Range Input**

| **Column** | Description | Data |
|---|---|---|
| Id | Unique identifier for each row. | Whole numbers eg: 1,2,3… |
| Upper Limit | Upper range of the AGV's speed | Decimal numbers between the range 0 to 1 |
| Lower Limit | Lower range of the AGV's speed | Decimal numbers between the range 0 to 1 |
| Offset Value | The geofence offset. If the offset value is positive then the geofence is enlarged, If the value is negative then the geofence is shrunk. | Integer numbers. |

5. After the data entered in the speed range table as seen in the Figure 13. Each row can either be deleted or edited under the *Actions* column. The *Upperlimit*, *Lowerlimit* and *OffsetValue* can further be iterated to fit the users requirements. *To* clear the data from the table as well at the database click on the button *CLEAR SPEEDTABLE IN DB.*

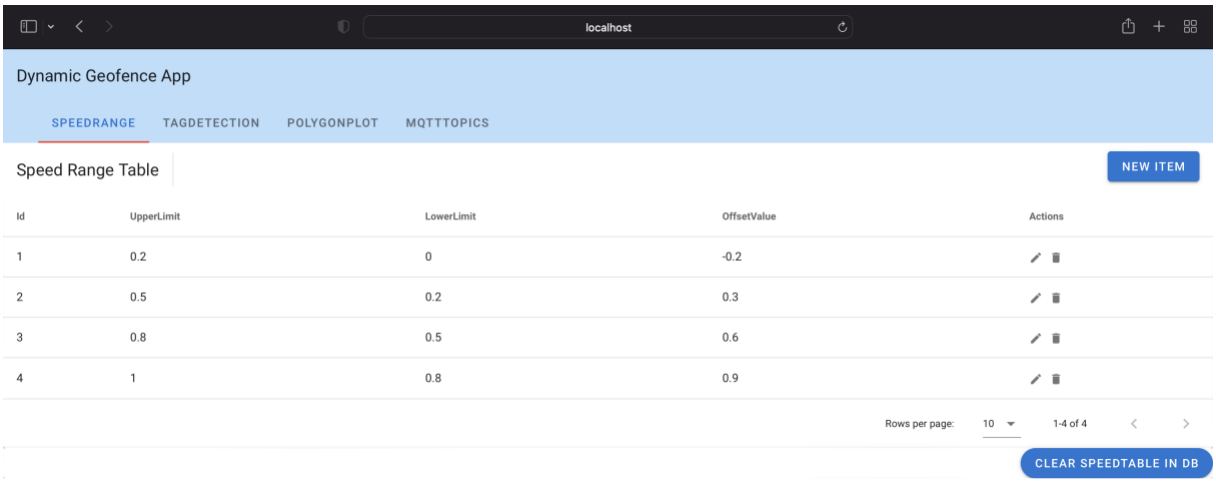



**Figure 13: SpeedRange Table**

6. Click on the *Tag detection* tab to add the *zone offset percentage* the value must be entered as decimal numbers ranging from 0 to 1 as per the users requirement as shown in figure 14.

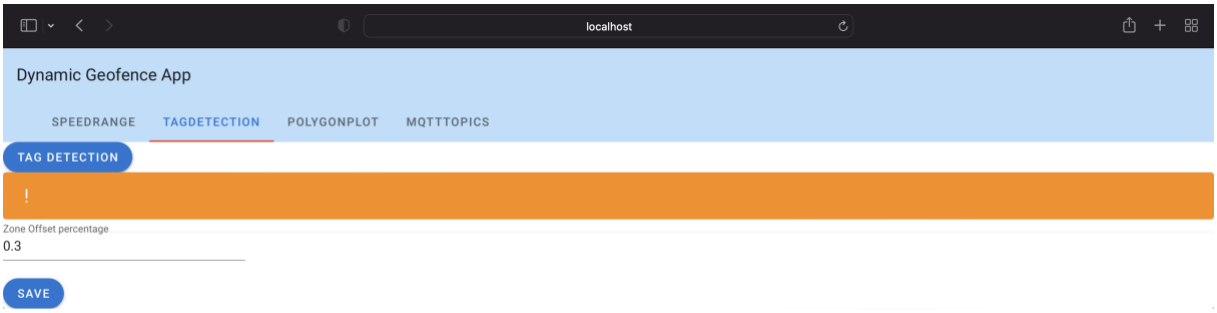



**Figure 14: Tag Detection**

7. Next, click on the MQTT Topics tab to view the MQTT table and select the check box corresponding to each row based on the AGV Name, AGV topic, Geofence ID, and Naise tag. The user also has option to create new entry by clicking on *NEW ITEM* button.

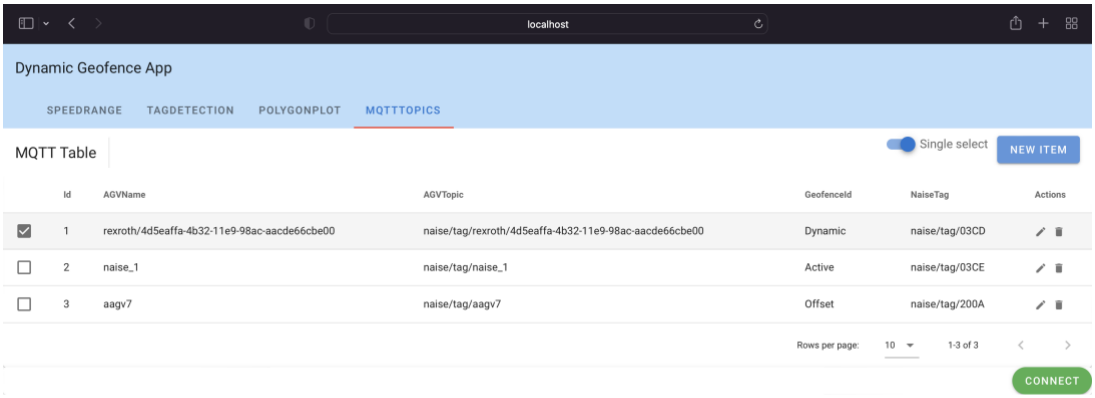



**Figure 15: MQTT Table**

| **UI components** | Description |
|---|---|
| Select box | User can select the row based on the topics that needs to be subscribed during run time |
| AGV Name | Name of the AGV as in NAiSE GUI |
| AGV Topic | MQTT topic to connected to the AGV and receive payload |
| Geofence Id | Name of the ID entered by the user in NAiSE GUI |
| Naise Tag | MQTT topic to connect to the NAiSE tag and receive real-time location |
| Actions | User can either edit or delete the respective row. |
| Connect | To connect to the MQTT broker |

8. The last step after adding all the necessary information is to click on the connect button in the bottom right corner to run the application.

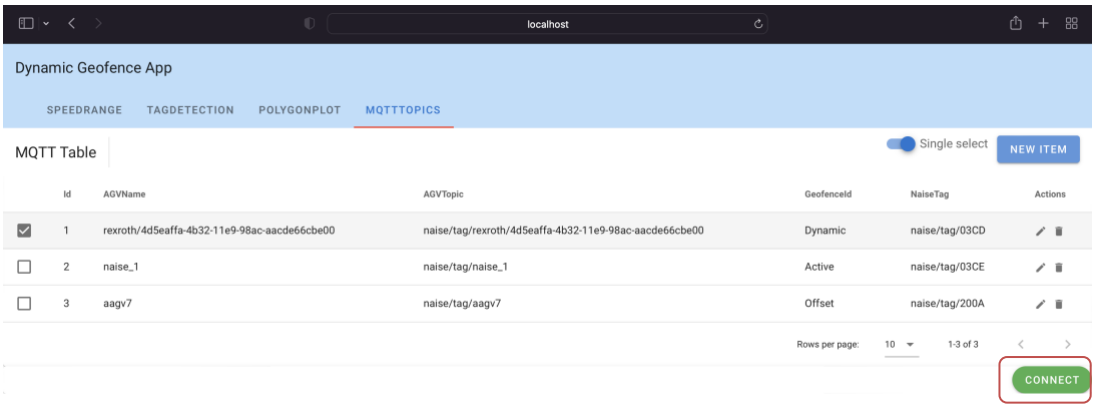



**Figure 16: Connect function**

# 3 Results

In this section we will visualize the working of enlarge/shrinking of geofence and tag detection in different zones of the geofence.

1. After clicking on the connect button as mention in section 2.3, step 8. At the syn_back terminal, '*Connected with result code 0'* is printed which states that the connection with the MQTT broker was successful. As shown in figure 17

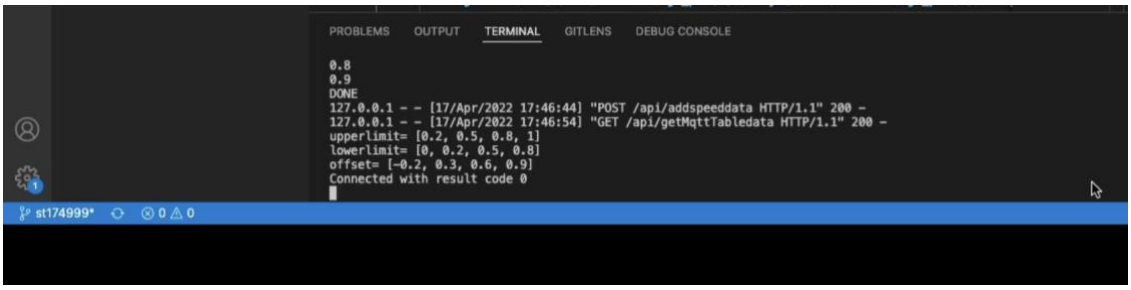
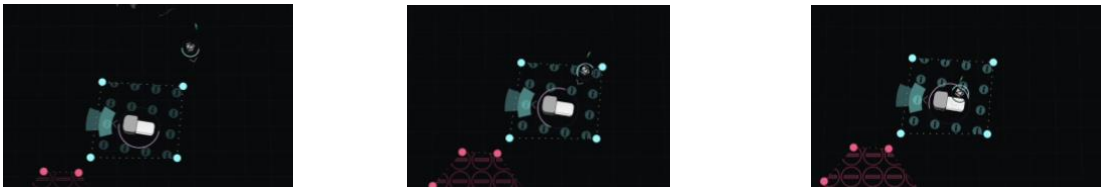



**Figure 17: Syn_back terminal**

2. To visualize the Tag detection function, open the NAiSE GUI web page to see the tag moving in real time.



a b c

**Figure 18: Tag detection a) Tag is outside the geofence b) Tag is in the blue zone of the geofence c) Tag is in the red zone of the geofence**

3. The warning based on which zone the tag is located will be sent to the Dynamic Geofence app.

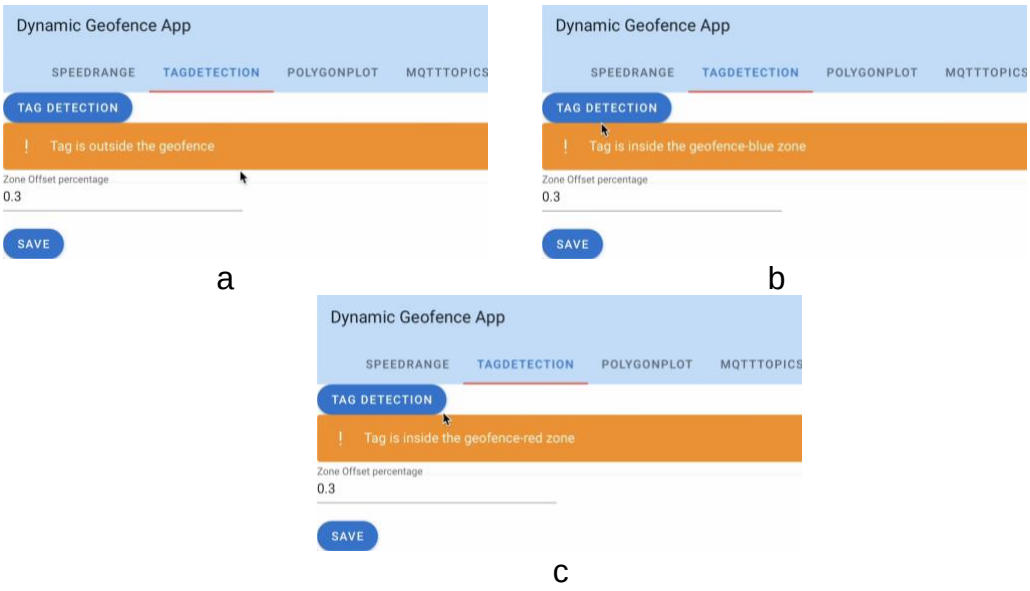



**Figure 19: Warning a) Tag is outside the geofence b) Tag is in the blue zone of the geofence c) Tag is in the red zone of the geofence**

4. To visualize the Enlarge/shrink function, open the NAiSE GUI and 1)first click on the AGV around which the geofence was drawn to open the control panel. 2)Select the *"TO POINT"* under navigation tab and 3)right click on any area of the map to drive the AGV to that point.

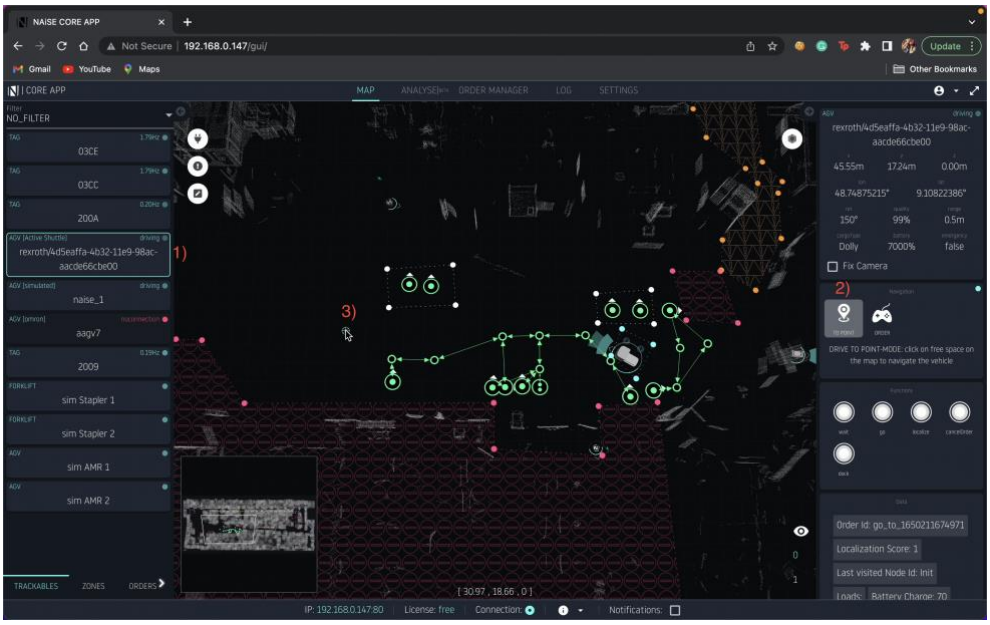
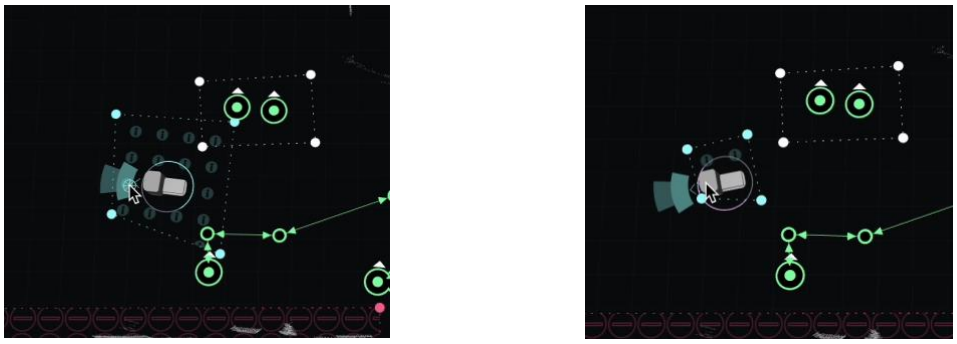





**Figure 20: Drive AGV in NAiSE GUI**

5. The enlarging and shrinking of geofence takes place as the speed of the AGV varies as seen in figure 21. The data from the speedrange table is queried at the backend from database and the respective offset value is applied based on the AGV speed.



a b

**Figure 21: Enlarge/Shrink a) Geofence is enlarged as AGV speed is high b) Geofence shrunk as AGV speed is less**

6. The Terminal prints the real-time AGV speed(agv speed) provided by the MQTT broker, as well as the corresponding offset value from the speedrange table and the tag warning.

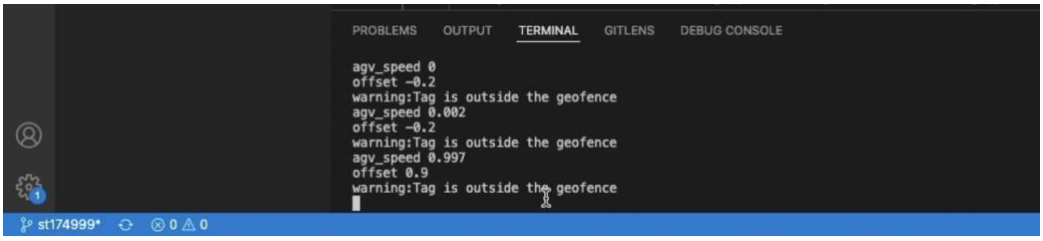



**Figure 22: Syn_back Terminal Output**

# *4* Monitoring process

To ensure that all MQTT message sequences are valid, the system may be executed to determine if it exhibits this behavior. A MQTT client software can be used to investigate MQTT behavior. MQTT Explorer is utilized in this validation. An MQTT client may be setup with the aid of this program by connecting to the MQTT broker. Topic subscribe and publish functionality are offered, same like with any other client. The below figure shows the published messages to the broker.

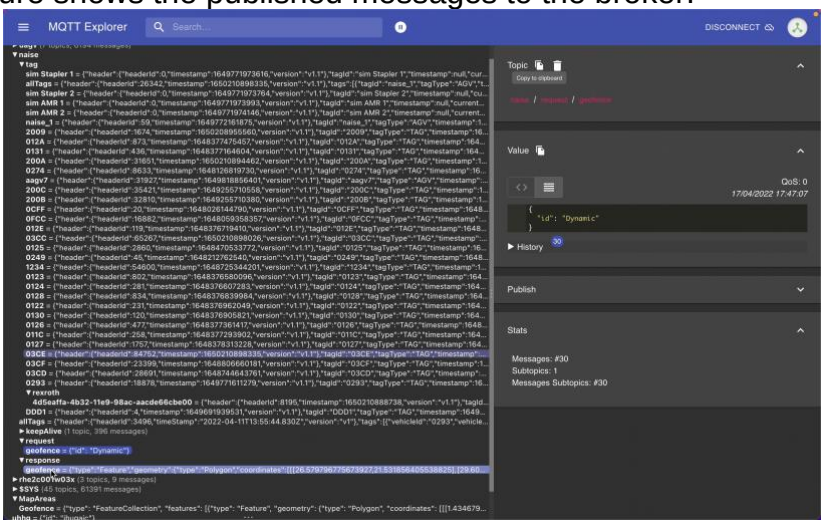



**Figure 23: MQTT Explorer**

# 5 Bibliography

[1] NAiSE,"NAiSE- Autonomous Intralogistics Manual v1.3.0".