Mapster

Omolola Solaru **Meet Patel** **Jonathon Vo** **Crystle Yi** **Neon Bui**
Georgia State University

Abstract

In this present era, social media attracts people to present their views [1] has a lot of impact in our lives. We all are well aware of the fact that social media has a lot of influence over people's mental Health like depression, dementia, schizophrenia etc. Although the usage of social media platform like Facebook, Instagram , twitter and software engineering together is not well understood, these mechanisms influence the software development practices. Software developers use and integrate into a wide range of tools ranging from code editing web-based portals . In our research project we would like to discuss about software engineering practices implemented in our project "Mapster" . We used resourced data, HTML/Javascript and SQL, in our project. Using this data gives way into utilizing the machine learning models and can be extended to deep learning methods such as CNN, AE [2] and in real live scenarios such as twitter analysis. We used the most compatible architectural model, which is the facade pattern from the structural design for our project.

1 Introduction

Mapster is a traffic monitoring system that finds the quickest route from the starting destination to the end destination. By using traffic data collected from multiple routes from point A to point B (i.e. no highway, no tolls, fastest route, suggested route, etc.), this product will be able to provide the user with a surplus amount of data that will assist the user with securing the best route.

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

The creation of the Mapster is to help reduce traffic overall for the user and other individuals. To do this, Mapster will redirect the driver to alternative route which will reduce the overall traffic. The data will be collected from a day-to-day basis, but will also be used as past data to provide the optimal solution. This data can and will be shared to certain figures like civil engineers, who will in turn use this data to help find better ways to reduce traffic in high traffic areas.

This product is not exactly innovative compared to certain products like Google maps and Apple maps. However, the main goal of this product is not the same as a mapping service. Th main goal is to monitor traffic patterns and record the data that will be used by other professions to understand and fix traffic. The secondary feature of Mapster is the mapping service which assists the product in collecting traffic information.

By assuring the most optimal route, Mapster will help reduce traffic before it starts and decrease traffic volume within high risk areas.It will be presented only as the basic function found on every smartphone,as our product utilizes the GPS of multiple phones to formulate a coherent database. It does require us to have an already established database, which we can provide using MapBox. Further changes and data in the time to come will be added to the database as more routes and traffic patterns appear.

2 Design Measures and Patterns

This section of the paper would showcase the different diagrams found within Mapster. This includes the class diagram, sequence diagram, and all the architectural models.

2.1 Class Diagram

The class diagram is used to construct and visualize a diagram of the classes, attributes, operations, and relationships within Mapster. With this diagram, you can understand how Mapster works on a more simple scale.

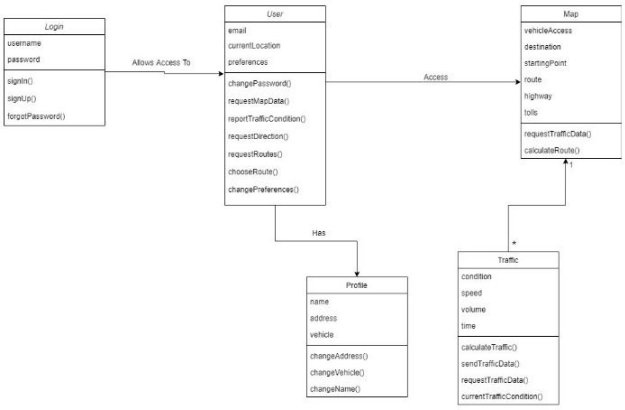


Figure 1: Class Diagram of Mapster

2.2 Sequence Diagram

Sequence diagram describes how and what order a group of objects work together. It is a behavior model of the dynamic behavior of the system, as it is an interaction model displaying the responses of the system. Essentially, they show what happens or what is suppose to take place when responding to external or internal stimulus.

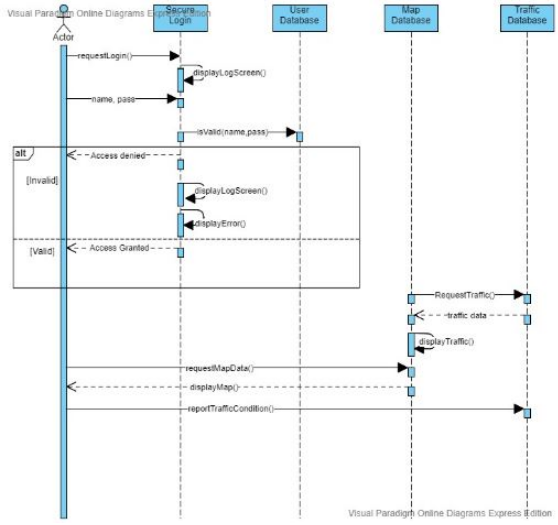


Figure 2: Behavioral Modeling (Dynamic Modeling)

2.3 Architectural Model

The architectural model will present the 4+1 view model. This means there will be a use case, logical view, process view, development view, and physical view of Mapster. Each of which is used by different professions based on their needs and requirements.

2.3.1 Use Case

The use case represents the general interaction of the Mapster program. This helps us identify the user requirement and highlight problems we might encounter later.

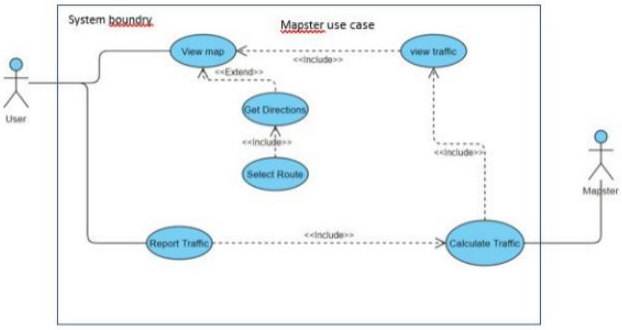


Figure 3: Use Case Diagram of Mapster

2.3.2 Logical View

The logical view is geared towards the developers and users to show how all the components logically connect with each other. It could also be considered a conceptual view, so it is more of just of a rough layout of how the components will be laid out.

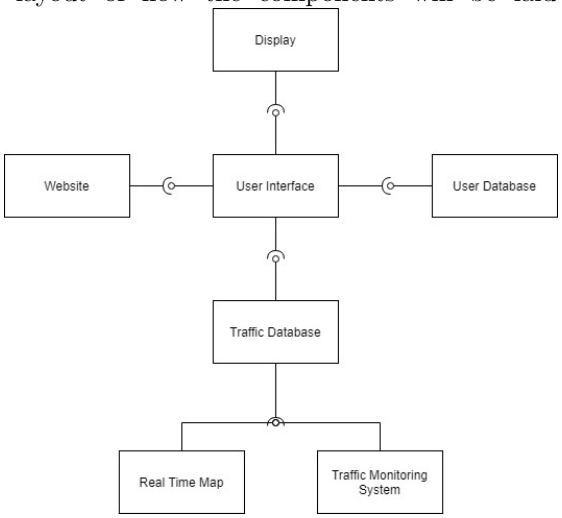


Figure 4: Logical View of Mapster

2.3.3 Process View

Similar to the logical view, the Process view is focused towards the developers but not users. The main purpose of the logical view is to showcase functionality and performance of the product. Process view models the dynamic aspects of the architecture and the behavior of its parts. Unlike the logical view, the process view has more detail because it shows which components interact with each other. As seen in Figure 5, the diagram shows how the different components of Mapster are able to interact with each other. As seen in Figure 5, it can be seen how every component interacts with the other components with the directional arrows.

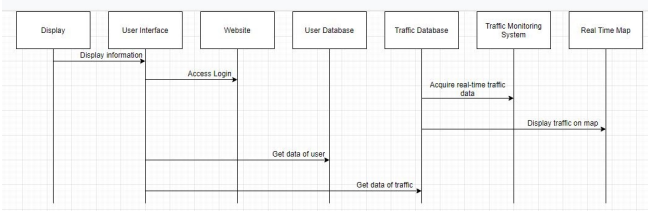


Figure 5: Process View of Mapster

2.3.4 Development View

The developmental view is focused towards the developers. This diagram shows how each component mentioned in the logical view will be executed. The development view's main concern to is address portability, build, and re-usability. As seen in Figure 6, every component's method of execution can be examined. For example, the user interface will be coded in HTML/JavaScript.

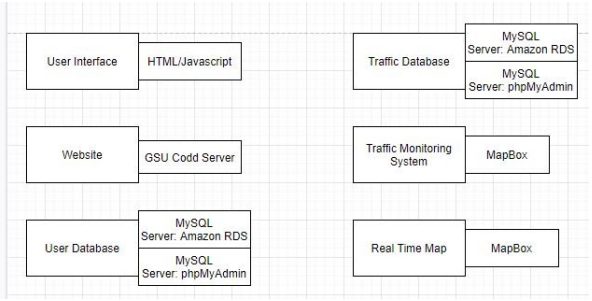


Figure 6: Development View of Mapster

2.3.5 Physical View

The physical view is what it sounds like, how the software will be mapped onto the hardware. Figure 7 shows how the databases will be displayed on the monitor as the UI and viewing methods.

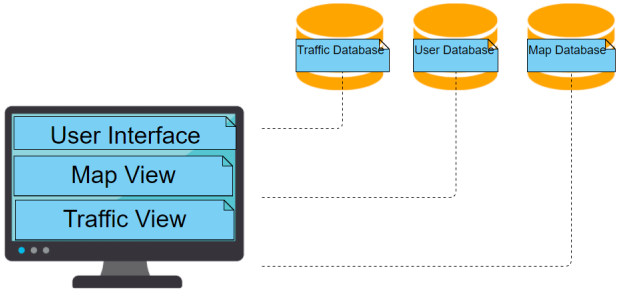


Figure 7: Physical View of Mapster

3 Implementation

The implementation part will explain both the backend and frontend of the project. This includes displaying the database of the backend and the completed page of the frontend. It will also state any coding languages and third-applications applications used to complete the project.

3.1 Backend

The backend of Mapster was developed utilizing SQL and phpMyAdmin. My phpMyAdmin is both a server and a database in which we can implement using SQL. The convenience and simplicity of phpMyAdmin allows us to easily connect the backend to the frontend, so if any data is entered through the frontend, it will be sent to the database and stored there. (This could be seen in Figure 2.) Mapster, itself, contains eight entities: login, permissions, role, route, test, traffic, user, and vehicle. Each of which has their own unique attribute, which can be seen from both Figure 1 and Figure 2. Due to the nature of the program, Mapster only requires a few entities since the most change can be found within the entities: traffic, login, and user.

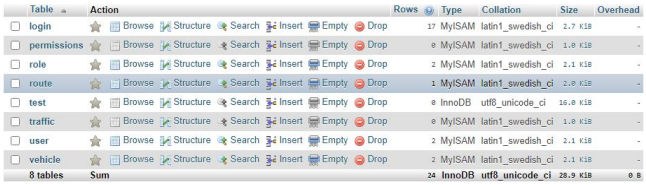
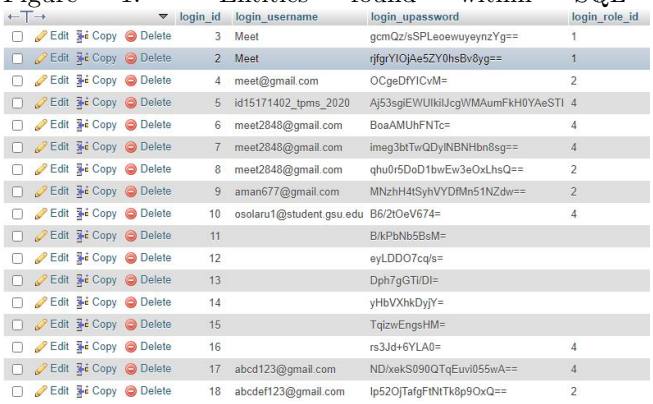


Table	Action	Rows	Type	Collation	Size	Overhead
login	Browse Structure Search Insert Empty Drop	17	MyISAM	latin1_swedish_ci	2.7 KiB	-
permissions	Browse Structure Search Insert Empty Drop	0	MyISAM	latin1_swedish_ci	1.0 KiB	-
role	Browse Structure Search Insert Empty Drop	2	MyISAM	latin1_swedish_ci	2.1 KiB	-
route	Browse Structure Search Insert Empty Drop	1	MyISAM	latin1_swedish_ci	2.0 KiB	-
test	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8_unicode_ci	16.0 KiB	-
traffic	Browse Structure Search Insert Empty Drop	0	MyISAM	latin1_swedish_ci	1.0 KiB	-
user	Browse Structure Search Insert Empty Drop	2	MyISAM	latin1_swedish_ci	2.1 KiB	-
vehicle	Browse Structure Search Insert Empty Drop	2	MyISAM	latin1_swedish_ci	2.1 KiB	-
8 tables	Sum	24	InnoDB	utf8_unicode_ci	28.9 KiB	0 B

Figure 1: Entities found within SQL



	login_id	login_username	login_upassword	login_role_id
Edit Copy Delete	3	Meet	gcmQz/sSPLeoewuyeynzYg==	1
Edit Copy Delete	2	Meet	rjfgrYIOjAe5ZY0hsBv8yg==	1
Edit Copy Delete	4	meet@gmail.com	OCgeDfYICvM=	2
Edit Copy Delete	5	id15171402_tpms_2020	Aj53sgiEWUIkilJcgWMAumFkH0YAeSTI	4
Edit Copy Delete	6	meet2848@gmail.com	BoaAMUhFNTc=	4
Edit Copy Delete	7	meet2848@gmail.com	imeg3btTwQDyINBNHbn8sg==	4
Edit Copy Delete	8	meet2848@gmail.com	qhu0r5DoD1bwEw3eOxLhsQ==	2
Edit Copy Delete	9	aman677@gmail.com	MNzhH4tSyhVYDfMn51NZdw==	2
Edit Copy Delete	10	osolaru1@student.gsu.edu	B6/2tOeV674=	4
Edit Copy Delete	11		B/kPbNb5BsM=	
Edit Copy Delete	12		eyLDDO7cq/s=	
Edit Copy Delete	13		Dph7gGTi/DI=	
Edit Copy Delete	14		yHbVXhkDyjY=	
Edit Copy Delete	15		TqizwEngsHM=	
Edit Copy Delete	16		rs3Jd+6YLA0=	4
Edit Copy Delete	17	abcd123@gmail.com	ND/xekS090QTqEuvi055wA==	4
Edit Copy Delete	18	abcdef123@gmail.com	lp52OjTafgFtNtTk8p9OxQ==	2

Figure 2: Data inputted into entities

3.2 Frontend

The frontend of Mapster was developed utilizing HTML, CSS, and Node js. The layout of the site was to allow the user to easily go through the framework of the design. Additionally, to ensure the user can view the traffic map panel, an map API called MapBox was integrated into the HTML code to enable the display of the custom map and its features.The allows the user to view the map as a means of seeing the traffic and routes to take to get from the source to destination effortlessly. HTML and CSS was utilized in the development of the home page, login page, and sign up page. The login page was assembled with the <form> element as the container for the variety types of input elements, such as: text fields, select fields, and sub-

mit buttons. It was then styled with CSS to make the design have a unique look. And this was completed for the other pages as well except the home page did not utilized the <form> element instead a <video> tag was used to ensure the layout of the background containing a mute,looping video with the logo at the center.

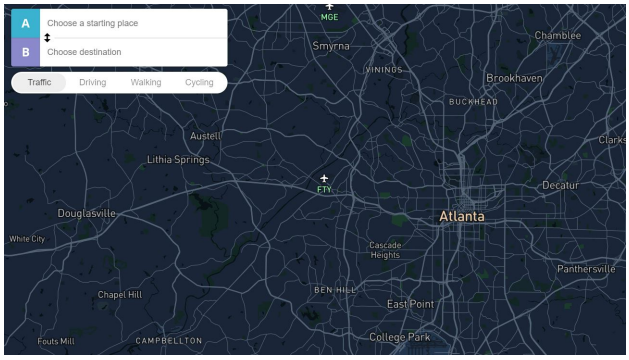


Figure 3: Image of Traffic Map

Figure 4: Image of Home page

4 Use Case and Requirements

4.1 Use Case No. 1

Use Case Name: Login System

Actors: User

Description: User logs in to the system. The input is sent to the systems to confirm credentials.

Alternate Path: User being unable to login gets sent to the "Forgot Password"

Pre-condition: Login Information

Requirement number: 1

Use Case Number: 1

Introduction: User information

Inputs: User credentials

Requirements Description: User input is required in order to continue.

Outputs: Input is sent to the system to confirm credentials.

4.2 Use Case No. 2

Use Case Name: Check Credentials

Actors: User

Description: User's login credentials are checked and determined whether is correct or not. If correct, the user will be given access.

Alternate Path: If the user is wrong, the user will not be given access and be returned to the login screen.

Pre-condition: Input to the login is needed.

Requirement number: 2

Use Case Number: 2

Introduction: User Credentials

Inputs: User credentials

Requirements Description: The input from Login screen is checked.

Outputs: Response from the system (allow access if login is right/deny if not)

4.3 Use Case No. 3

Use Case Name: Check all the roles of access

Actors: User

Description: The system will list out the roles the user is allowed access to.

Alternate Path: The user can logout to login screen

Pre-condition: User needs to be allowed access

Requirement number: 3

Use Case number: 3

Introduction: Roles of access

Inputs: Response from the system (Allows access from successful login)

Requirements Description: User is able to check the roles being able to access to

Outputs: Sends the user to Monitoring system

4.4 Use Case No. 4

Use Case Name: Traffic Monitoring System

Actors: User

Description: User is given access to a list of monitoring roles

Alternate Path: User can logout to login screen

Pre-condition: All access of roles are a success

Requirement number: 4

Use Case number: 4

Introduction: Traffic Monitoring System

Inputs: Successful roles of access

Requirements Description: Lists out all the monitoring roles

Outputs: The monitoring roles

4.5 Use Case No. 5

Use Case Name: Monitoring User System

Actors: User

Description: User is given access to a list of monitoring roles

Alternate Path: User can logout to login screen

Pre-condition: All access of roles are a success

Requirement number: 5

Use Case number: 5

Introduction: Monitoring User System

Inputs: access to User System

Requirements Description: Monitors the User System

Outputs: The User System

4.6 Use Case No. 6

Use Case Name: Monitoring Roles of User

Actors: User

Description: User can see what permissions they have

Alternate Path: User can logout to login screen

Pre-condition: Access to Traffic Monitoring System is fulfilled

Requirement number: 6

Use Case number: 6

Introduction: Monitoring User Roles

Inputs: Selecting User

Requirements Description: Shows User permissions

Outputs: Shows a list of permissions users can access.

4.7 Use Case No. 7

Use Case Name: Monitoring Login Report

Actors: User

Description: User can see the Login Report

Alternate Path: User can logout to login screen

Pre-condition: Access to Traffic Monitoring System is fulfilled

Requirement number: 7

Use Case number: 7

Introduction: Monitoring Login Report

Inputs: Selecting Login Report

Requirements Description: Selecting Login report to show Login Report

Outputs: Shows Login attempts: successes and fails

4.8 Use Case No. 8

Use Case Name: Monitoring Traffic Details

Actors: User

Description: User can see Traffic Detail

Alternate Path: User can logout to login screen

Pre-condition: Access to Traffic Monitoring System is fulfilled

Requirement number: 8

Use Case number: 8

Introduction: Traffic Detail

Inputs: Selecting Traffic Detail

Requirements Description: Selecting Traffic Details to monitor Traffic details

Outputs: Traffic details are shown to the user.

4.9 Use Case No. 9

Use Case Name: Monitoring Vehicle Profiles

Actors: User

Description: User can Monitor Vehicle Profiles

Alternate Path: User can logout to login screen

Pre-condition: Access to Traffic Monitoring System is fulfilled

Requirement number: 9

Use Case number: 9

Introduction: Monitor Vehicle Profiles

Inputs: Selecting Vehicle Profiles

Requirements Description: Selecting Vehicle Profiles to monitor Vehicle Profiles (Car type, etc)

Outputs: Shows Vehicle profile used with the user.

4.10 Use Case No. 10

Use Case Name: Monitoring Route Details

Actors: User

Description: User can Monitor Route Details

Alternate Path: User can logout to login screen

Pre-condition: Access to Traffic Monitoring System is fulfilled

Requirement number: 10

Use Case number: 10

Introduction: Monitor Route Details

Inputs: Selecting Route Details

Requirements Description: Selecting Route Details to monitor Route Details (Recommended route to take, etc)

Outputs: Shows Routes and possible recommended routes to take for the user.

4.11 Use Case No. 11

Use Case Name: Update the Fastest Routes

Actors: User

Description: User inputs destination for the system to figure out the fastest route.

Alternate Path: User can logout to login screen

Pre-condition: User needs to be allowed access

Requirement number: 11

Use Case number: 11

Introduction: Fastest Route Update

Inputs: Route destination

Requirements Description: Shortest route to destination is checked and updated.

Outputs: shows the shortest route

4.12 Use Case No. 12

Use Case Name: Monitoring Traffic Police

Actors: User

Description: Will check and monitor police near traffic.

Alternate Path: Canceling this brings the user back to the login

Pre-condition: User needs to be allowed access

Requirement number: 12

Use Case number: 12

Introduction: Monitoring Traffic Police

Inputs: clicking for police

Requirements Description: If police are on duty at the specified route or traffic, the system is to show their location.

Outputs: Police on duty shown.

5 Test Case

5.1 Test Case 1

Title: Login Page

Description: A registered user should be able to successfully login at https://osolaru1.github.io/Codd-SEP/TrafficCop.html

What is being tested: successful login vs unsuccessful. Login correctly transitions to the main page.

Test Input: Login details

Expected output: correct login details will allow access while incorrect login details will not.

Initialization: Login details on hand.

Precondition: the user must already be registered with an email address and password.

Assumption: A supported browser is being used.

Test Steps:

1. Navigate to Traffic Cop
2. In the 'username' field, enter the username of the registered user.
3. Enter the password of the registered user

Expected Result: A page displaying the user's inbox should load, showing any new message at the top of the page.

Actual results: Access to the full Mapster page.

Input: User and Pass (correct)

Output: Successful Access

Input: Random User and Random word for pass (Incorrect)

Output: Unsuccessful Access.

Tear Down: Make sure continued access to roles are allowed

Bugs found: None

5.2 Test Case 2

Title: Check Credentials

Description: User's login credentials are checked and determined whether is correct or not. If correct, the user will be given access.

What is being tested: successful login vs unsuccessful. Login correctly checks in databases.

Test Input: Login credentials

Expected output: Correct login shows the page, Incorrect doesn't

Initialization: login credentials on hand

Precondition: the user must already be registered with an email address and password.

Assumption: A supported browser is being used.

Test Steps:

1. Navigate to Traffic Cop
2. In the 'username' field, enter the username of the registered user.
3. Enter the password of the registered user
4. Wait for login to be successful or not.

Expected Result: Correct login credentials allows access, Incorrect doesn't

Actual results: Access to the full Mapster page.

Tear Down: Make sure continued access to roles are allowed

Bugs found: None

5.3 Test Case 3

Title: Check all the roles of access

Description: The system will list out the roles the user is allowed access to.

What is being tested: User is shown a list of roles via buttons of what they can access.

Test Input: accessing all roles of access

Expected output: a list of roles that have access to Mapster showing.

Initialization: access via login is successful

Precondition: Login is successful

Assumption: A supported browser is being used.

Test Steps:

1. After accessing the main page, go to roles of access.
2. Roles of allowed access is displayed

Expected Result: A page displaying the user's accessible roles should be seen or a section leading to the page with accessible roles.

Actual results: Access to the full Mapster page.

Tear Down: Make sure continued access to roles are allowed

Bugs found: None

5.4 Test Case 4

Title: Traffic Monitoring System

Description: User is given access to the monitorings system

What is being tested: User is shown a page of the monitoring system via buttons of what they can access.

Test input: accessing Monitoring System

Expected output: being able to access the Monitoring System

Precondition: All access of roles are a success

Assumption: A supported browser is being used.

Test Steps:

1. Navigate to Traffic Cop
2. In the 'username' field, enter the username of the registered user.
3. Enter the password of the registered user
4. After login, go to the traffic monitoring system section.

Expected Result: A page displaying the monitoring system should be seen.

Actual results: Access to the monitoring system.

Tear Down: Make sure continued access to roles are allowed

5.5 Test Case 5

Title: Monitoring User System

Description: User is given access to the User System

What is being tested: User is shown the User System.

Test Input: accessing monitoring user system

Expected output: monitor user system being displayed

Initialization: access via login is successful

Precondition: the user must already be registered with an email address and password.

Assumption: A supported browser is being used.

Test Steps:

1. Navigate to Traffic Cop
2. In the 'username' field, enter the username of the registered user.
3. Enter the password of the registered user
4. After login, go to the monitor user system.

Expected Result: A page displaying the user's accessible roles should be seen or a section leading to the page with accessible roles.

Actual results: Access to the full Mapster page with accessible roles.

Tear Down: Make sure continued access are allowed

Bugs found: None

5.6 Test Case 6

Title: Monitoring Roles of User

Description: User is given access to a list of monitoring roles

What is being tested: User is shown a list of roles via buttons of what they can access.

Test Input: accessing monitoring user system

Expected output: monitor roles system being displayed

Initialization: access via login is successful

Precondition: the user must already be registered with an email address and password.

Assumption: A supported browser is being used.

Test Steps:

1. Navigate to Traffic Cop
2. In the 'username' field, enter the username of the registered user.
3. Enter the password of the registered user
4. After login, go to the monitor roles section.

Expected Result: A page displaying the user's roles should be seen.

Actual results: Access to the full Mapster page with the monitor the roles of the user.

Tear Down: Make sure continued access are allowed

Bugs found: None

5.7 Test Case 7

Title: Monitoring Login Report

Description: User is given access to the Login Report.

What is being tested: User is shown the Login Report in the main menu

Test Input: accessing Login Report

Expected output: Login Reports being displayed

Initialization: access via login is successful

Precondition: the user must already be registered with an email address and password.

Assumption: A supported browser is being used. Test Steps:

1. Navigate to Traffic Cop
2. In the 'username' field, enter the username of the registered user.
3. Enter the password of the registered user
4. After login, go to the Login Report section.

Expected Result: A page displaying the Login Report should be seen.

Actual results: Access to the full Mapster page with the Login Report being accessible.

Tear Down: Make sure continued access are allowed

Bugs found: None

5.8 Test Case 8

Title: Monitoring Traffic Details

Description: User is given access to the Traffic Details.

What is being tested: User is shown the Traffic Details in the main menu

Test Input: accessing Traffic Details

Expected output: Traffic Details being displayed

Initialization: access via login is successful

Precondition: the user must already be registered with an email address and password.

Assumption: A supported browser is being used.

Test Steps:

1. Navigate to Traffic Cop
2. In the 'username' field, enter the username of the registered user.
3. Enter the password of the registered user
4. After login, go to the Traffic section.
5. Go to the details.

Expected Result: A page displaying the Traffic page with details should be seen.

Actual results: Access to traffic showing details is shown.

Tear Down: Make sure continued access are allowed

Bugs found: None

5.9 Test Case 9

Title: Monitoring Vehicle Profiles

Description: User can Monitor Vehicle Profiles

What is being tested: Vehicle Profiles shown upon checking traffic in traffic details.

Test Input: accessing Vehicle Profiles

Expected output: Vehicle Profiles shown

Initialization: access via login is successful

Precondition: the user must already be registered with user and password.

Assumption: a supported browser is being used. Login is successful.

Test Steps:

1. Navigate to the Traffic Details
2. In the Vehicle Profiles, click it.
3. Click the 'Next' button.

Expected Result: Police cars shown or other vehicle profiles around testing inputs.

Actual Result: Vehicle profiles shown. Police cars shown in denser traffic.

Tear Down: Make sure continued access to roles are allowed

Bugs found: none

5.10 Test Case 10

Title: Monitoring Route Details

Description: A registered user can monitor route details. This includes the shortest routes and an alternative link for traffic.

What is being tested: The Map API, Routes being marked correctly, UI showing properly and leading to pages for other options.

Test Input: accessing monitoring route details

Expected output: monitor route details being displayed

Initialization: access via login is successful

Precondition: the user must already be registered with user and password.

Assumption: a supported browser is being used. Login is successful.

Test Steps:

1. Navigate to the Route Details
2. In the Destination field, add any address.
3. Click the 'Next' button.

Expected Result: Results showing the shortest route to the destination as well as other options on the side.

Actual Result: Page not done yet. Testing for this function is not possible at this time.

Tear Down: Make sure continued access to roles are allowed

Bugs found: None

5.11 Test Case 11

Title: Update the Fastest Routes

Description: User can figure out the fastest route

What is being tested: Finding the fastest route to the destination

Test Input: accessing quickest route

Expected output: quickest route shown

Initialization: access via login is successful

Precondition: the user must already be registered with user and password.

Assumption: a supported browser is being used. Login is successful.

Test Steps:

1. Navigate to the Route Details
2. In the Destination field, add any address.
3. Click the 'Next' button.
4. Click on the fastest route.

Expected Result: Fastest or the shortest route shown.

Actual Result: Shortest route shown.

Tear Down: Make sure continued access to roles are allowed

Bugs found: none

5.12 Test Case 12

Title: Monitoring Traffic Police

Description: User can check out and monitor for police near traffic

What is being tested: seeing if traffic police will be shown at a traffic dense area

Test Input: monitor traffic police

Expected output: Showing any police near the area

Initialization: access via login is successful

Precondition: the user must already be registered with user and password.

Assumption: a supported browser is being used. Login is successful.

Test Steps:

1. Navigate to the Traffic Details
2. In the Details, click on traffic police
3. Expected Result: traffic police in area shown

Actual Result: Traffic police shown in traffic dense area. No police shown in non-dense area.

Tear Down: Make sure continued access to roles are allowed

Bugs found: none

Conclusion

This paper is the conclusive effort of a group of students from Georgia State University. It shows the different parts that are needed to create a fully-functioning application. Mapster was suppose to be a traffic-monitoring application that also functions as a mapping service, but it still lack many features that we had wanted. From the frontend, which utilizes HTML, CSS, and Node js to provide the design of the application, to the backend, which utilizes SQL and phpMyAdmin to provide a database to store the obtained information, this application showcases the work done by us. Even though this is the "final" report, the application and the group still has much to learn and improve.

References

[1] S. T. Sadasivuni and Y. Zhang, "Using gradient methods to predict twitter users' mental health with both covid-19 growth patterns and tweets," *second IEEE International Conference on Humanized Computing and Communication with Artificial Intelligence (HCCAI 2020) September 21-23, 2020 Irvine, CA, USA*.

[2] J. K. Mandivarapu, B. Camp, and R. J. Estrada, "Self-net: Lifelong learning via continual self-modeling," *Frontiers in Artificial Intelligence*, vol. 3, p. 19, 2020.