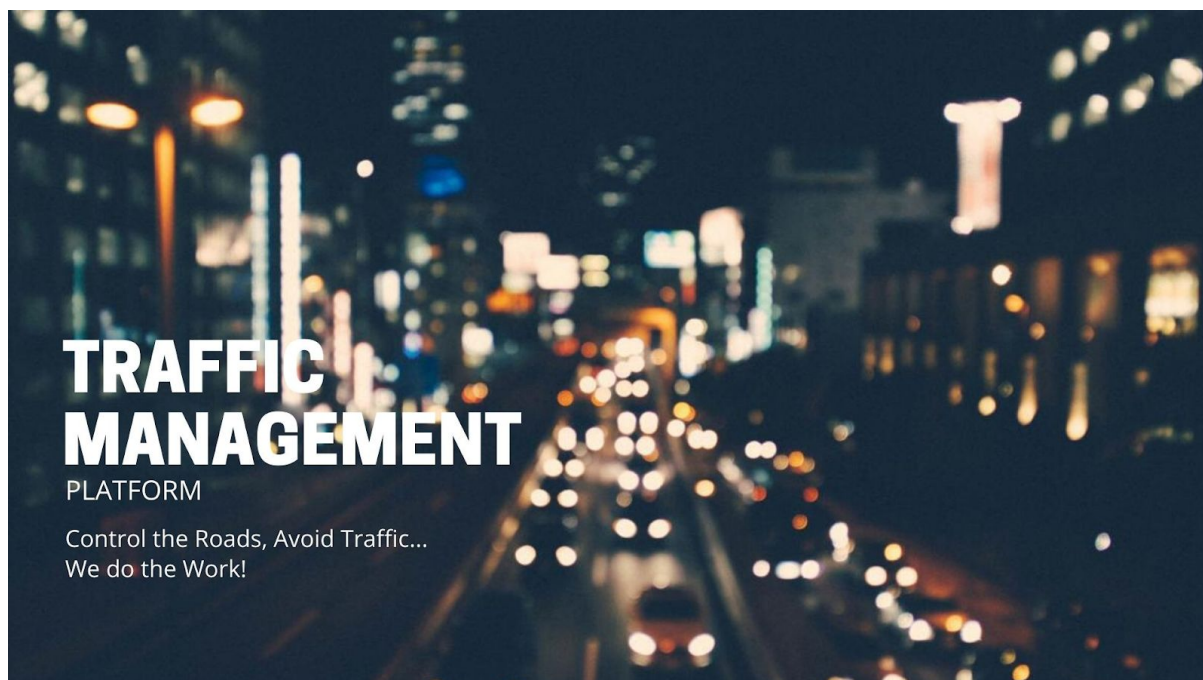


Traffic Management Platform



Project developed by:

- David Rocha nº 84807 (dmtar@ua.pt)
- Francisco Morgado nº 85009 (fmpfmorgado@ua.pt)
- Cristiano Santos nº 79671 (jcristianosm@ua.pt)
- Leandro Cardoso nº 80311 (leandrocardoso@ua.pt)
- João Soares nº 79955 (joao.f.soares@ua.pt)

Collaborators:

- Joaquim Macedo (jmacedo@ua.pt)
- Pedro Teixeira (pedro.teix@ua.pt)
- Paulo Vasconcelos (paulobvasconcelos@ua.pt)

Advisors:

- José Alberto Fonseca (jaf@ua.pt)
- Joaquim Ferreira (jjcf@ua.pt)

Abstract

Road traffic management is an area that is not fully developed at the moment. In other words, there was space for the emergence of new tools to help monitoring the flow in certain zones, more specifically the beaches at Barra and Costa Nova. With this said, our purpose was to design and build a tool that allowed the visualization of the important information regarding this issue. With some input of people already working on this area, we managed to put together an application that allows data visualization and comparison in different time intervals, responding to their requests. There is still more work to do, but this project can be seen as a foundation to the future of road traffic management.

Acknowledgments

Before starting the report itself, we would like to acknowledge and thank the contributions of all people involved in this project, besides our team. Starting from the advisors, José Alberto Fonseca and Joaquim Ferreira, that were always available to gather with the team, responding to our doubts, as well as helping with what they could, providing a vision of someone that is much more experienced than us. Secondly, Joaquim Macedo, who has experience in the area, showed and requested us what was actually necessary to be on the application, providing some crucial information to make it not just usable but also required. Lastly, Pedro Teixeira and Paulo Vasconcelos for the suggestions made during the project, as well as their input in various other technical details.

Table of Contents

1. Introduction	5
2. State of Art	5
3. Conceptual Modeling	6
3.1 Calendar and task division	6
3.1.1 Responsibilities in the first module	6
3.1.2 Responsibilities in the second module	6
3.2 Functional Requirements	7
3.2.1 High priority	7
3.2.2 Medium priority	7
3.2.3 Low priority	7
3.3 Non Functional Requirements	8
3.3.1 High priority	8
3.3.2 Medium priority	8
3.3.3 Low priority	8
3.4 Personas and motivations	9
3.4.1 Users	9
3.4.2 Administrators	10
3.5 Use Cases	11
4. Implementation	12
4.1 Architecture	12
4.2 Vehicle Detection	13
4.2.1 Used Model	13
4.2.2 How it Works	13
4.3 Vehicle Tracking	14
4.3.1 Deep SORT	14
4.4 Database - UML Diagram	15
4.5 SWOT Analysis	16
5. Results and Discussion	17
5.1 Usability Tests	17
5.2 Image Identification and Tracking	17
6. Conclusion	18

1. Introduction

Road safety is a topic that is imperative to study nowadays, we have intense road traffic more than ever and it is important to learn from it, especially dealing with it. Therefore, the main goal behind Intelligent Transport Systems and vehicular network communications, is to provide better road safety, since there is a large number of the current accidents and deaths that could be avoided if the vehicles had the capacity to trade information between them, with the road infrastructure and other road users. In this context, the PASMO project, Plataforma Aberta para o desenvolvimento e experimentação de Soluções para a MObilidade, developed by the Instituto de Telecomunicações, installed a set of equipments at Costa Nova and Barra beaches, as well as the Barra bridge and the A25. One of the use cases of the project is the classification of the traffic at both beaches. The goals of this project are to improve the classification of the traffic at Barra and Costa Nova beaches, by merging data from traffic classification radares and video cameras.

The traffic radars are capable of identifying, although with significant errors, various object classes, such as light vehicles, heavy vehicles, bikes and persons. It is intended to use the footage from the cameras, with the same viewfield of the radares, to improve classification results, recurring to image processing techniques to detect objects in real time, such as YOLO or equivalent. The data from four radars and four cameras are available in real time to be processed. Besides vehicle classification, it shall be created a front-end that allows the visualization of the data, either by the public, that will have access to generic data, or by public entities, that will be allowed to consult statistics and other aggregates.

2. State of Art

Platforms that focus on road safety are something new, even though they are so important, there are few applications that focus on this topic such as Google Maps, Waze etc. What we want to accomplish with this project is something that not only can give all the important info to traffic managers (which they will need to evaluate the global scenario) but to also to give the information that normal citizens can also understand and do their daily travel aware of the traffic situation around the areas which we mentioned before.

- ### 3.1.2 Responsibilities in the second module

3.2 Functional Requirements

3.2.1 High priority

- Chart visualisation.
- In/out traffic flow in real time.
- Information should be efficient.
- Refreshed zon/device selection should be done with the minimap.

3.2.2 Medium priority

- Direct comparison between zones.
- Direct comparison between different time periods
- In/Out traffic flow of different time periods.
- One or more selected radar/zone to analyse data.

3.2.3 Low priority

- Statistic visualisation related to meteorological data.

3.3 Non Functional Requirements

3.3.1 High priority

- Distinction between users on the information they can access.
- Java implementation.
- Dashboards with Graph.js.
- Database implemented in SQL.
- Database should update every 15 minutes.
- Use of YOLO to identify vehicles.
- 10% increase in the success ratio on identifying vehicles.

3.3.2 Medium priority

- Database security, especially against SQLInjection.
- Create extensive documentation.
- Database should update every 10 minutes.
- 15% increase in the success ratio on identifying vehicles.

3.3.3 Low priority

- Database should update every 5 minutes.
- 20% increase in the success ratio on identifying vehicles.

3.4 Personas and motivations

3.4.1 Users

Maria Assunção:

Maria is 65 years old, and retired. During her free time, she likes to travel, but doesn't enjoy traffic. To plan her trips, and to know the best roads and time periods to drive, she would like to have easy and efficient ways to check seasons with the biggest affluence, as well as the most congested roads, to be able to avoid them.



Paulo Rodrigues:

Paulo is 32, and is the president of Cascavais. He believes that having a constant motorisation and a precise register of traffic related data in that area, allowing him to control and reinforce the areas with the biggest affluence, as well as having statistics of different time periods, to be able to pick the best places to advertising and public constructions would benefit Cascavais.



Alberto Sousa:

Alberto is 42 and is the traffic manager of Cascavais. It would make his job easier if he had access to a platform where every data related to road traffic, past and present, was available, so that he could plan with more efficiency every change that would benefit the region, such as police reinforcement in areas with constant traffic jam, and the addition of road signs in areas with a high number of accidents.



3.4.2 Administrators

Lara Silva:

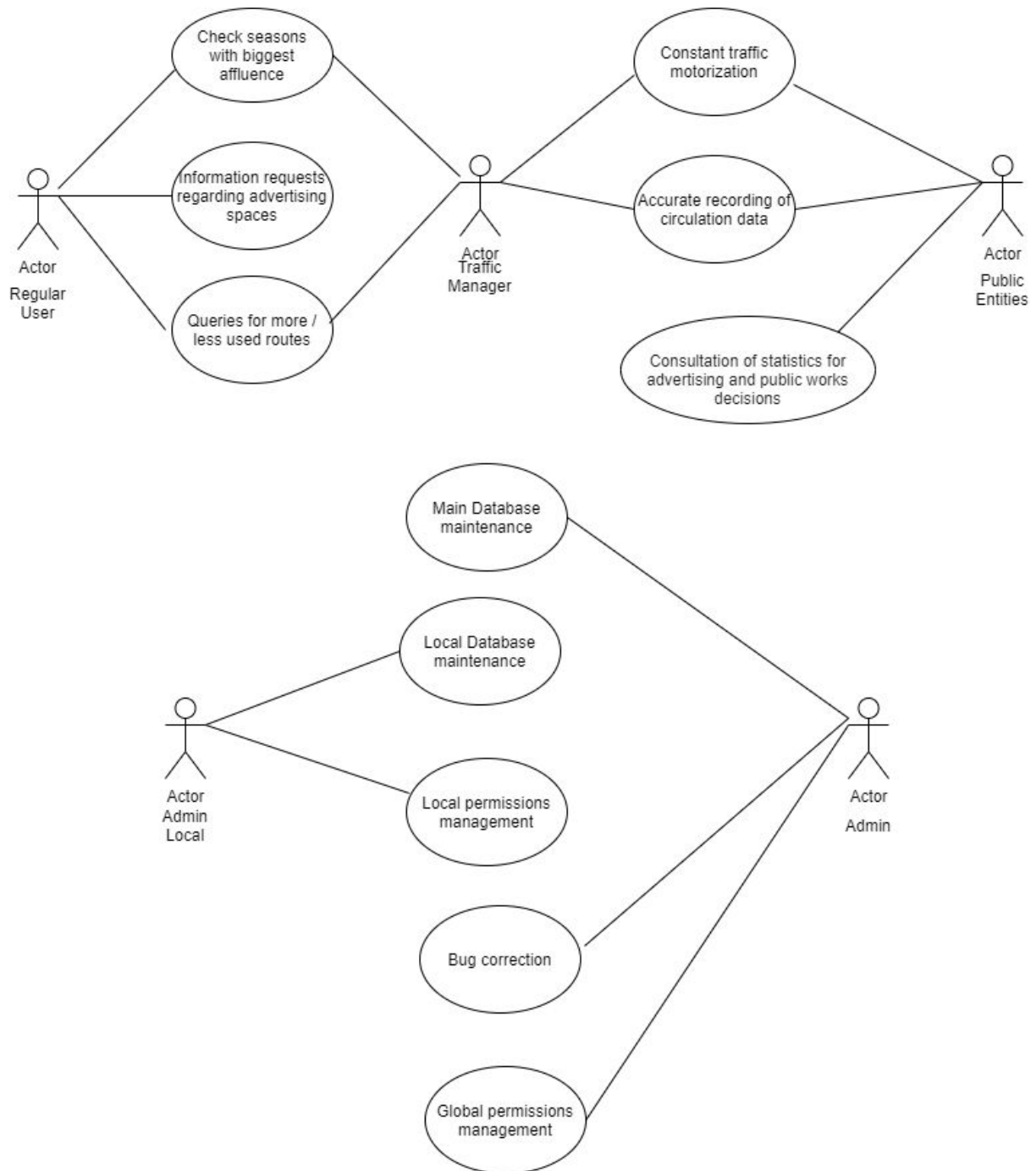
Lara is a local administrator of the implemented system. It is part of her work to maintain the local database, as well as manage permissions of access of data by other areas, after permissions from the owners of the service.

**Mário Santos:**

Mário is the administrator of the platform, and it is part of his work to maintain the mais database, as well as correct some bugs that could appear, in addition to make some improvements to increase the quality of the service to the platform users.

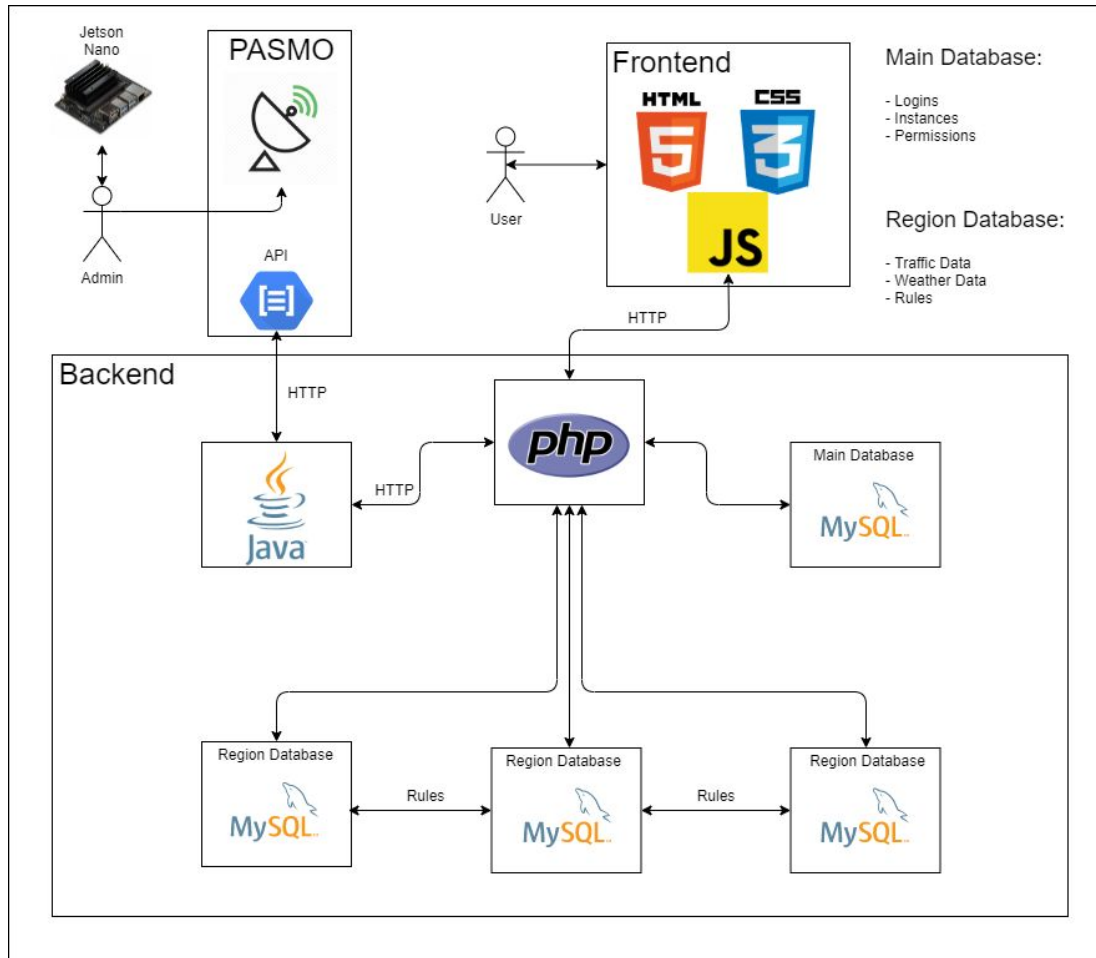


3.5 Use Cases



4. Implementation

4.1 Architecture



The chosen architecture is multi-tenant, which means that every public entity is responsible to manage their database, so that we only provide the model to the regional database, in order to make it communicate with the central one. It is a web app, where all the front end is developed in HTML, CSS and JavaScript. The backend will use Java to obtain data from the PASMO API, MYSQL to create the database and PHP to create a bridge between the java program, the javascript and the database.

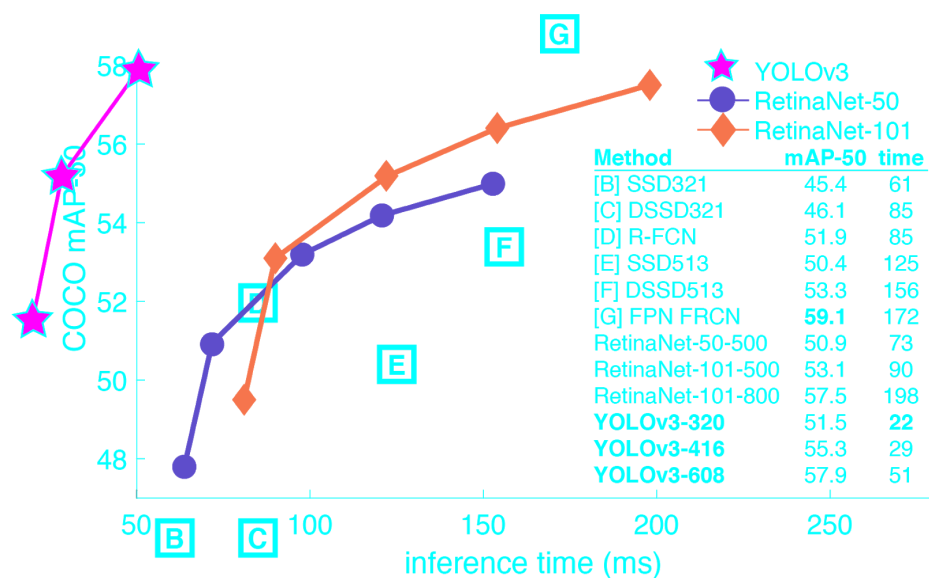
4.2 Vehicle Detection

Since object detection on videos demands a high graphical performance, we used a Jetson Nano, which is board with a powerful GPU.



4.2.1 Used Model

The chosen model was YOLO, why use YOLO and not any other existing model? As shown in the image below, YOLOv3 is more efficient (time wise) and produces better results as an object recognition tool than most of the other models.



4.2.2 How it Works

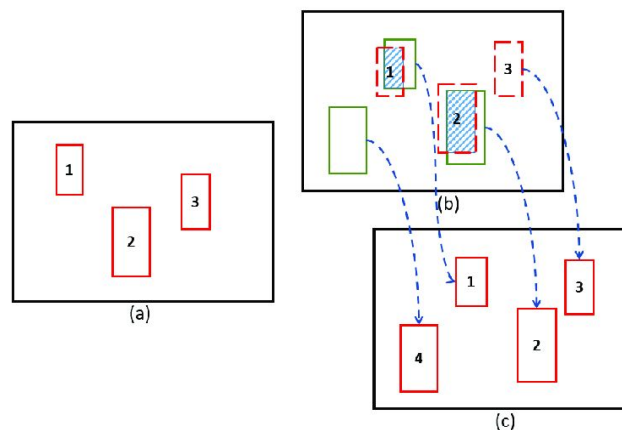
At YOLO, we apply a single neural network (created with the darknet framework) to each full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities.

4.3 Vehicle Tracking

While working on the vehicle detection, we came to the conclusion that we would need to track each vehicle to be able to count them, otherwise every detected object in each video frame would be counting as a different vehicle. For this we used the Deep SORT algorithm, a variation of SORT (Simple Object Real Time Tracker).

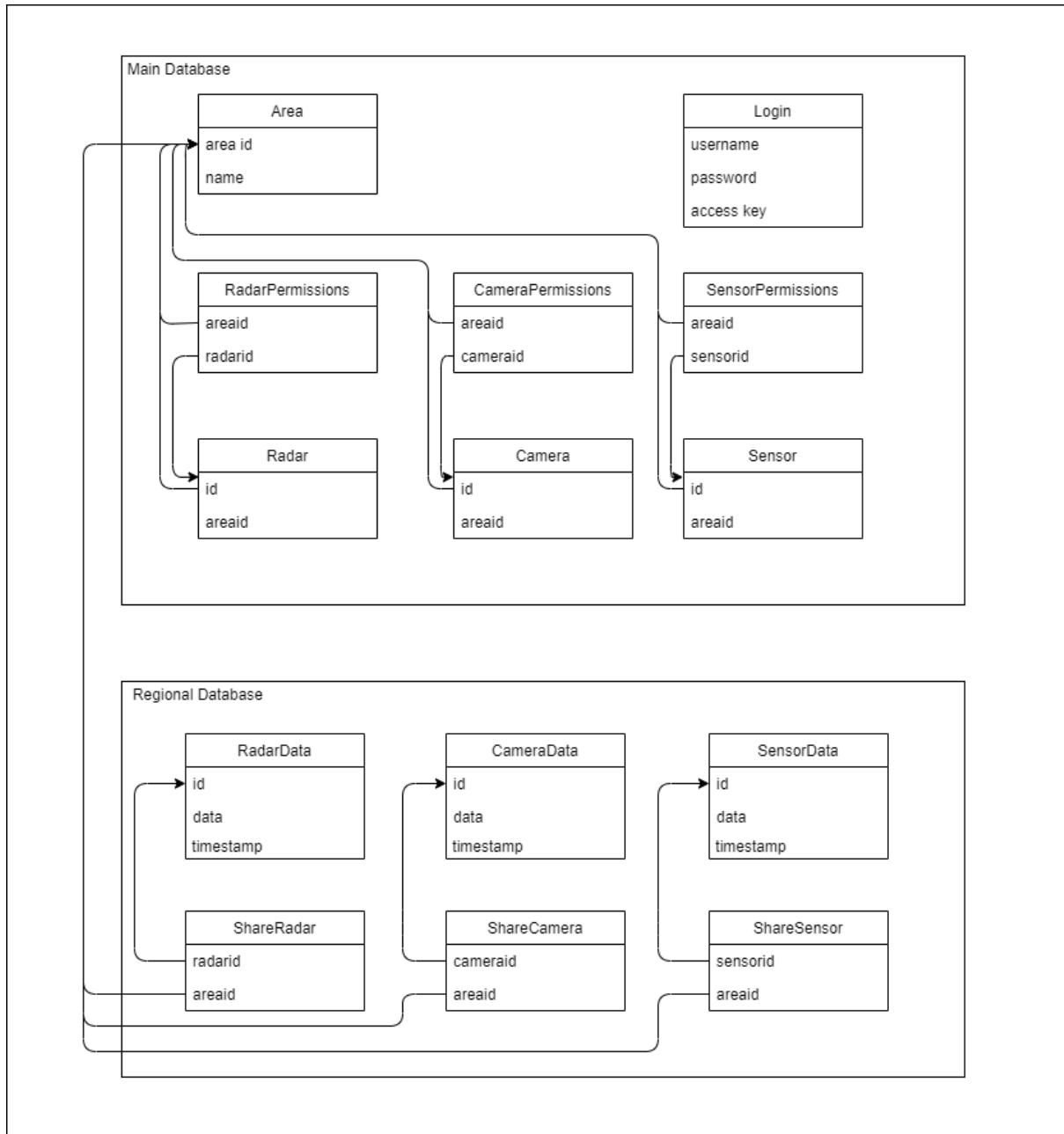
4.3.1 Deep SORT

By assigning each detected object a ID and saving it's last position, in the next video frame it is possible to predict if the new detected object is in fact the same from the previous frame.



As seen above, in frame A there were 3 objects detected, those objects position and ID values are saved in memory, and in frame B it is checked if the new detections position coincide with the saved ones, and if so, assigns the new position to its respective ID.


4.4 Database - UML Diagram



The main database stores user login data and permissions, equipment ids, as well as the regions where they're installed.

The region databases will store specific data the user pretends, and permissions, to provide capacity to trade information with other radars from other areas, if the user allows it.

4.5 SWOT Analysis

SWOT	 Traffic Management Platform
<p>Strengths:</p> <ul style="list-style-type: none"> -Part of an existing project - PASMO. -Groundbreaking technology that merges Camera and Radar Data. Traffic data is stored in our system. -Field sensors such as parking sensors, wi-fi hotspots and radars provide a lot of information. -Our software can be adapted to several types of radars. 	<p>Weaknesses:</p> <ul style="list-style-type: none"> -The stored data may not be 100% correct. -Vehicles success classification rate is not 100%. -The capacity of our database may be affected by the budget of our project. -The database will need permanent maintenance. -The capacity of our servers may be affected by the budget of our project. -The servers will need permanent maintenance.
<p>Opportunities:</p> <ul style="list-style-type: none"> -Our system can be implemented in other areas. -The data can be provided to advertisement companies, respecting data privacy rules. -Possible increase in individual transport due to recent rules and fears. -Possibility of generating relevant information to new areas. 	<p>Threats:</p> <ul style="list-style-type: none"> -Huge decrease in traffic may impossibilite much needed testing. The API we are using is unstable and frequently changed. -Equipment malfunction, may report incorrect data. -Equipment theft and vandalism, that requires replacements.

5. Results and Discussion

5.1 Usability Tests

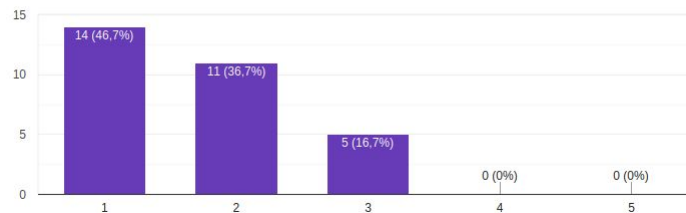
Since the beginning the group focused on a user centered design while developing the web application. Every week we got together with Joaquim Macedo, a professional regarding the traffic area, who helped us in knowing what would be useful and how the information should be presented.

Even though his help was crucial, verifying the usability of our application with outside users that never played with the platform was something that we planned from the beginning.

The tests were made by assigning the user to some tasks, and taking feedback in order to know if the tasks were easily achieved, as well as knowing how the application felt to their eyes. Below we can see the results of two of the questions made after the user finished with its tasks.

Grau de dificuldade a encontrar informação desejada.

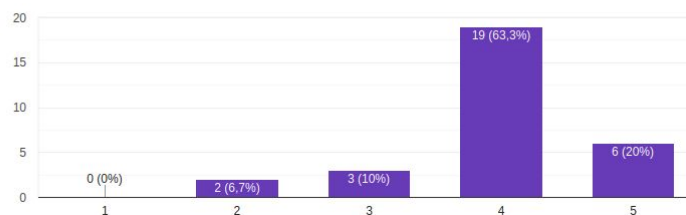
30 respostas



Overall we think that the feedback was positive, although there were some improvements that we have done after reading all the responses, like having the option to select just one date to get information of, instead of being forced to select a date to compare.

Atratividade gráfica do sistema.

30 respostas



5.2 Image Identification and Tracking

Through the creation of the neural networks many performance issues started to emerge, YOLO weights are formed by iteratively processing images and labels and assembling them in the neural network, during these trains Jetson's memory and CPU usage were constantly at 100%, getting segmentation faults when trying a complex train, because even the swap memory was getting saturated. Because of this, extensive research about CPU vs GPU performance in image detection, as well as memory efficiency when creating big neural networks, had to be made in order to not compromise the expectations that the group had.

In spite of these optimizations, we still didn't have a weight that could precisely identify heavy vehicles and bikes. Changes were made to the YOLO weight creation configuration file in order to further optimize the neural network creation, and we got results that were reflected in the decrease of the time needed for this task (78 hours to 26), with the cost of degrading the weight quality. With this, we accomplished a YOLO weight made with eight thousand batches with four images each.



These images are taken from the compilation of videos from the radars installed in Barra and Costa Nova, and here we can find some of the results accomplished. Through the visualization of the counters created we can easily check if the camera identification is working correctly or if it is missing any important objects.

In the end, the group was able to get together all the tools needed to calibrate the cameras installed, from the creation of the YOLO weights to the assertion of its correct behaviour.

6. Conclusion

Regarding the web application, the group feels that all the objectives were completed, we acquired knowledge about tools and languages that we never used, the required infrastructure and the complexity that full fledged web applications require at the structural level. With the Traffic Management Platform, a fair chance to improve traffic classification and road safety exists, and the addition of new functionalities should be the priority when moving forward.

The services provided by our development in the vehicle detection field should provide a satisfactory base when calibrating de equipments, by using the neural network creation tool based on darknet, to create YOLO weights with custom datasets, and our application to unify the vehicle detection to the videos and compile them together, verifying its results.

In conclusion, a contribution to the smart traffic management field was achieved, by providing a web application to facilitate the management itself, as well as tools to efficiently improve the quality of the data presented in the application. A great amount of knowledge and skills were acquired with the development process and by dealing with a real system, that will benefit us as future developers.

Internet References

- Nvidia 'Getting started with Jetson Nano Developer Kit':
<https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit>
- Pysource 'YOLO V3 - Install and run yolo on nvidia jetson nano':
<https://pysource.com/2019/08/29/yolo-v3-install-and-run-yolo-on-nvidia-jetson-nano-with-gpu/>
- Tzutalin's Labellmg: <https://github.com/tzutalin/labellmg>
- Francium Tech 'Custom object training and detection with YOLOv3, Darknet and OpenCV':
<https://blog.francium.tech/custom-object-training-and-detection-with-yolov3-darknet-and-opencv-41542f2ff44e>
- JetsonHacks 'Jetson Nano - Use more memory':
<https://www.youtube.com/watch?v=JXv39FGi-nw>
- JetsonHacksNano's installSwapeFile:
<https://github.com/JetsonHacksNano/installSwapfile>
- Pjreddie 'YOLO: Real-Time Object Detection': <https://pjreddie.com/darknet/yolo/>
- The AI Guy 'YOLOv3 in the CLOUD: Install and Train Custom Object Detector (Free GPU)': <https://www.youtube.com/watch?v=10joRJt39Ns>
- timebutt 'How to train YOLOv2 to detect custom objects':
<https://timebutt.github.io/static/how-to-train-yolov2-to-detect-custom-objects/>
- Bountysourc 'Training YOLOv3 with own dataset':
<https://www.bountysource.com/issues/56401814-training-yolov3-with-own-dataset>
- Pjreddie's Darkne': <https://github.com/pjreddie/>
- Towards data science 'Implementing YOLO on a custom dataset':
<https://towardsdatascience.com/implementing-yolo-on-a-custom-dataset-20101473ce53>
- Ultralytics's 'Train Custom Data':
<https://github.com/ultralytics/yolov3/wiki/Train-Custom-Data>
- erik lindernoren's yolov3.weight and darknet53.conv.74:
<https://gitmemory.com/issue/eriklindernoren/PyTorch-YOLOv3/201/503310806>
- Bamwani's 'Vehicle counting using python YOLO':
<https://github.com/bamwani/vehicle-counting-using-python-yolo?fbclid=IwAR1LuPQ3HLJJOINwXHuAQx3OYxILz-fwZAnc-MvahUM-iwSgJTOW5GcIN0E>