AutoTracker Functional-Specification

Brian Quilty 18373856

Conor Marsh 19728351 18/11/22

1.Introduction

Autotracker is a mobile application for people who drive. Its purpose is to help drivers improve their driving skills. It will help to ensure that drivers do not exceed the speed limit by getting the speed of the vehicle the user is currently in and then getting the speed limit of the road through bing maps api. It uses sensors in the phone like accelerometer and gyroscope to determine what type of driver the user is e.g aggressive slow. The type of driver is obtained by machine learning algorithms like decision trees to predict driver behavior.

Machine Learning

The app will use a machine learning algorithm known as the decision tree classifier. It uses a dataset that was obtained from Kaggle which uses gyroscope’s x,y and z values and also accelerometers x, y and z values to determine if a driver is either aggressive, normal or slow. Using python the dataset is split into a classified column and the sensor value columns. The model is then used to predict the driver type based on the values passed in through the sensors in the phone.

1.2 business context

This mobile application could be sold to drivers of any sort of vehicle in order to help improve their driving experience. This could potentially save them from vehicle accidents through the functionality that the app provides, like driver behavior, excessive speed limit etc. Could also be extremely useful for learner drivers who want to see their driving progress over time and to make sure if they are improving. Driving learning businesses can distribute the app to help keep track of learners' driving behavior over time to see if they are improving.

2.1 Product / System Functions

Create Account / Login:

Upon opening the application for the first time the user will be prompted to create an account for the application. Once they have created the application, when they reopen the application, they will be prompted to sign into their account, they can also choose to remain logged in to the application if they do not want to repeat this process.

Start a journey:
The user will be able to press a button on the main screen to notify the application to start the journey.

End a journey:
The user will be able to press a button on the main screen to notify the application to end the journey.

Track speed of vehicle:
The application will use gps coordinates through their phone to determine the speed of the vehicle that they are currently in. The speed of the vehicle can be changed between km/h to m/h, it will also be displayed on the main screen of the application once the user has started to journey.

Notify if speed limit reached:
The application will compare the speed of the vehicle that has been tracked to the speed limit of the road that the vehicle is currently on. It will get the speed limit of the road by first getting the road you are currently on through GPS coordinates. Then it will use Bing maps road api to get the speed limit of those GPS coordinates. If the speed limit of the road has exceeded the speed limit it will emit a voice to notify you to slow down. This voice’s pitch and volume can be changed through settings.

Determine driver behavior:
The application will be able to determine the driver's behavior. It will classify their behavior into either aggressive, normal or slow through machine learning methods. It will use sensors in the android phone, specifically gyroscope and accelerometer. These sensors will be used in the machine learning algorithm which will use a kaggle dataset. There will be a separate screen to show driver behavior over time.

Get driver statistics:
The user will be able to use the application to get specific driving statistics like average speed, distance driven, driver behavior for specific journeys. They can rename journeys to give better visibility and delete journeys to save space on their phone.

2.2 **User Characteristics and Objectives**

Users will access the auto tracker application through their android mobile devices. Our users will be people who drive vehicles and specifically those drivers that want to improve their driving and want to improve their driving experience. This could be learner permit drivers and the general public who drive which will take up our target audience. Our application will be built to conform to the user's accessibility needs. We will ensure that the user can easily identify the functionality of the application. This will be done by ensuring that the text and buttons are visible to the user by making sure they are large enough and provide enough precise information to display the functionality of the app. The user interface will be easy to understand and intuitive allowing a seamless experience across the application. Although some of our features may be complicated, we will have to ensure that an everyday user with a small plethora of technical knowledge can use the application. The user's age will vary based on the country's driving age. This means our users could be 16+ years of age.

2.3 operational scenarios.

2.1 Starting a journey

A user can start a journey by clicking the start button on the main screen. This will begin to calculate the speed of the vehicle that you are currently in and also start monitoring the sensors in the phone. If the speed of the vehicle goes over the speed limit of the road a voice will emit from the phone notifying the user to slow down.

2.2 Ending a journey

A user can end the journey by clicking the end button once the journey has started. Once the journey is finished it will stop monitoring the sensors in the phone and the speed of the vehicle you are in.

2.3 Monitoring sensors

Every second the phone will get the current coordinates of the sensors accelerometer and gyroscope, it will then store these values inside an sql database which will be used later on to determine a users driving behavior.

2.4 Analyzing driver behavior

The application will be able to give a precise breakdown of the percentage of user driving behavior using charts and other tools to give the user a clear insight into their driving behavior and whether or not they are an aggressive, normal or slow driver. This will be based on all the journeys that the user has partaken in.

2.5 Voice software

The user will be able to adjust the volume and the type of voice that is emitted from the app. The voice speaks when the speed of the vehicle is greater than the speed limit of the road the user is currently driving on. The speed limit of the road the user is driving on is obtained through bings roadmap api.

2.6 Accessing journey stats

The user can access statistics based on previous journeys. They will be able to get the average speed, distance covered, times the speed limit was reached, breakdown of driver behavior for each individual journey they take in.

2.7 Deleting specific journeys

The user can delete journeys that have been built up over time, this can help save space.

2.8 Renaming journeys

The user can rename specific journeys to help identify unique journeys that they traveled.

2.9 Object detection of other road vehicles

The user can access information to help identify where other road vehicles are on the road

2.10 Traffic light detection system

The user can access system information which will warn of the arrival of oncoming traffic lights

2.4 Constraints

2.4.1 Machine Learning

Neither of us have used machine learning before. Ca4010 data warehousing goes into more details about machine learning so that module should help us with regards to machine learning for the project

2.4.2 Database storage

There could possibly be a large amount of data stored in the database on the phone, so we will have to ensure that the database isn't too large and that it takes up too much space on the phone.

2.4.3 Phone gps

Neither of us have worked with applications that deal with tracking a phone's gps coordinates and using those coordinates to build a functioning application.

2.4.4 Permission request

Our application will need to require permission to track a phone's GPS coordinates. If the user declines this request the application will not be able to fully function.

2.4.5 Phone's sensors

Although almost all smartphones have gyroscope and accelerometer sensors built into them, if the phone does not have them or is missing one then we will not be able to monitor the users driving behavior.

3. Functional Requirements

3.1 Starting a journey

Description:

Once the user enters the main screen of the app, they can indicate to the app that a car journey has started by clicking the start button. This will begin to monitor the driver's speed and sensors to help determine driver behavior.

Criticality:

Vital to the app as the main functionality for helping to improve driver behavior comes from starting the journey and letting the application know to monitor vehicle speed and mobile sensors.

Technical Issues:
Ensuring that the user has gps enabled, a notification must be sent if they do not. Most smartphones have gyroscope and accelerometer sensors, but old smartphones may not.

Dependencies:
User must be logged in and must press the start journey button.

3.2 Ending a journey

Description:
Once the user has stopped driving and they want to stop monitoring their vehicle they can notify the application that the journey has finished by pressing the stop button.

Criticality:
Critical to the app as the app needs to know when to analyze the values that have been passed through the application.

Technical Issues:
Making sure the app is able to analyze the values if the user never presses the stop button, will need to still be able to analyze the values the next time the user starts the app/ if the speed of the car remains at zero for some time.

Dependencies:
User must be logged in and a journey must have started.

3.3 Monitoring Sensors

Description:
Once the journey has started, the phone will use its built-in sensors, gyroscope and accelerometer and pass these coordinates to the sql database, they will be stored and then used in the machine learning algorithm to determine driver behavior.

Criticality:
Essential to the app as sensors will be used to determine driver behavior and give a better insight to drivers about their driving style, helping improve their driving.

Technical Issues:

Most smartphones have these sensors built into their hardware. Very old smartphones may not though so that could be a technical problem, nearly all smartphones would have these sensors built in however.
Dependencies:
User must be logged in and the journey must be started.

3.4 Analyzing driver behavior

Description:
On the main screen of the app the user will be able to enter a screen which will give them a breakdown of their behavior that has been built up over numerous journeys through graphs etc. It will determine what sort of driver they are based on sensor coordinates which will use machine learning to decide if they are an aggressive, normal or slow driver based on a Kaggle dataset.

Criticality:
Critical to the app, will give the user a visual representation of their driving behavior.

Technical Issues:
Making sure the graphs give a precise visual representation of the driving behavior could be an issue, has to make sure that the user can fully see the graphs and that they are big enough.

Dependencies:
Users must be logged in and must select the button to enter the screen.

3.5 Voice software

Description:
A settings screen will be used to adjust the pitch and the volume of the voice that notifies the user if the speed limit has been exceeded.

Criticality:
This is not the most important feature for the app but will help give the user a more personalized and overall better experience.

Technical Issues:
None.

Dependencies:

User must be logged in and must select the settings icon.

3.6 Accessing journey stats

Description:

The user can click a button to take them into a new screen which will display all the previous journeys that they have taken. Previous journeys will display the breakdown of driver behavior, average speed, total distance traveled and if the speed limit was broken.

Criticality:

Important to the app as the user can look at specific journeys to analyze their behavior on a specific day. Important to learner drivers for example, as they want to track progress over time, so seeing journeys they took previously and comparing them to now helps achieve this goal.

Technical Issues:

Holding this amount of data could be a problem. Might have to set a limit or ask the user to delete specific journeys if there is too much data.

Dependencies;

User must be logged in and they must have finished a journey previously.

3.7 Deleting specific journeys

Description:

A delete button beside the individual journeys screen will delete the data for that specific journey.

Criticality:

Critical to the app as the phone may not be able to handle so much data if there are a lot of journeys. Also, it will help to save data on the phone if the user doesn’t want the application to occupy a lot of space on its phone.

Technical Issues:

None

Dependencies:

User must be logged in and they must have previous journeys to delete.

3.8 Renaming specific journeys

Description:
A rename button beside the individual journeys will help the user to rename specific journeys.
Criticality:
Not extremely critical but gives the user a more personalized and immersive feel to the application.

Technical Issues;
None

Dependencies:
User must be logged in and they must have previous journeys to rename.

3.9 Object detection of other road vehicles

Description:
Implementation of object detection sufficient to be able to recognise another road vehicle and approximate distance from the user's vehicle

Criticality:
Important but may involve technical complications which could limit its implementation, this would not be considered a major failing if it proves the case.

Technical Issues;
Will require refined programming implementations using the OpenCV programme.

Dependencies:
User will have to be logged in

3.10 Traffic light detection system

Description:
Implementation of object detection sufficient to be able to recognise an oncoming set of traffic lights

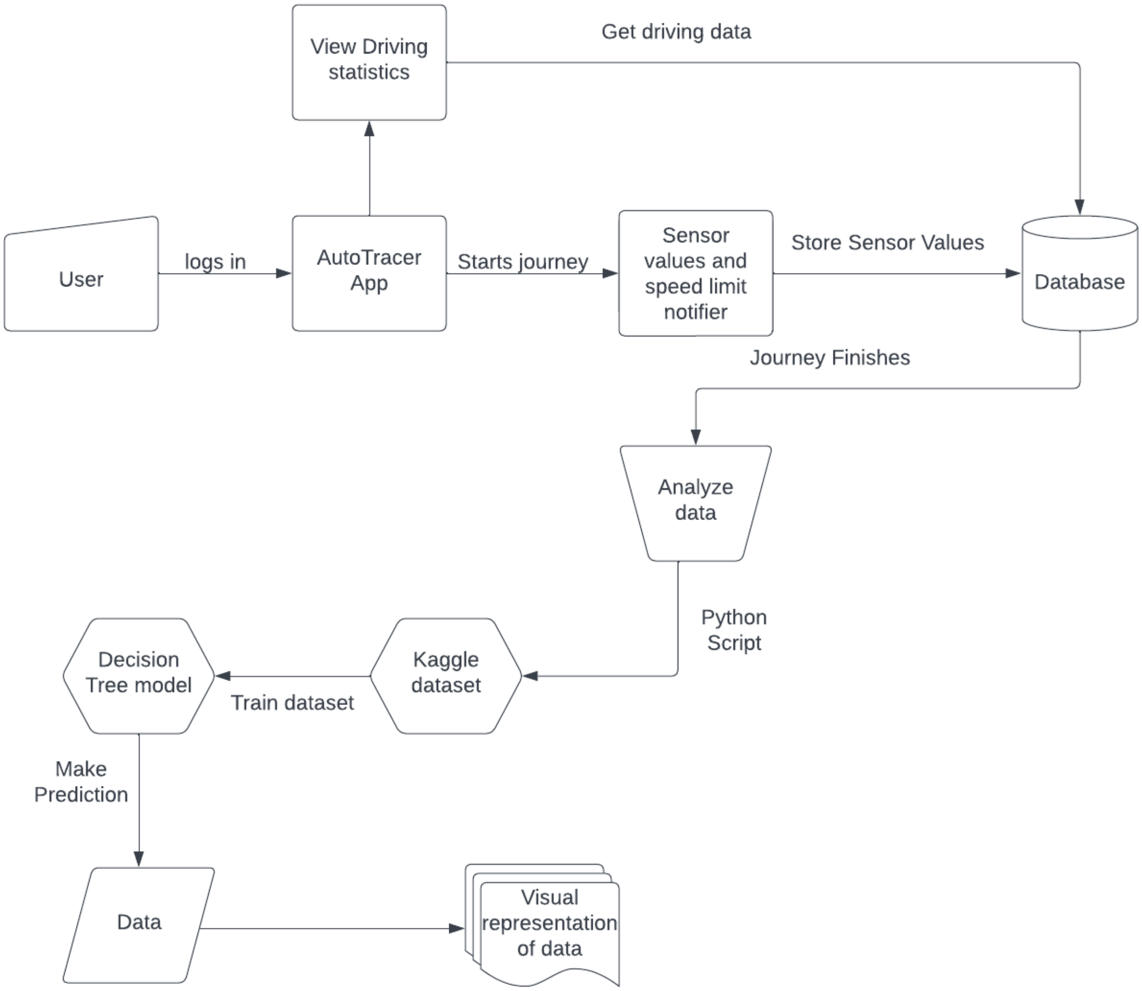
Criticality:
Less important than vehicle detection from the perspective of developing the app as a safety mechanism, as the user should not be put in such a position by the app that they would miss a set of traffic lights using their own eyesight. Involving technical

complications which could limit its implementation, some measure of successful implementation would be considered a success.

Technical Issues;
Will require refined programming implementations using the OpenCV programme.

Dependencies:
User will have to be logged in

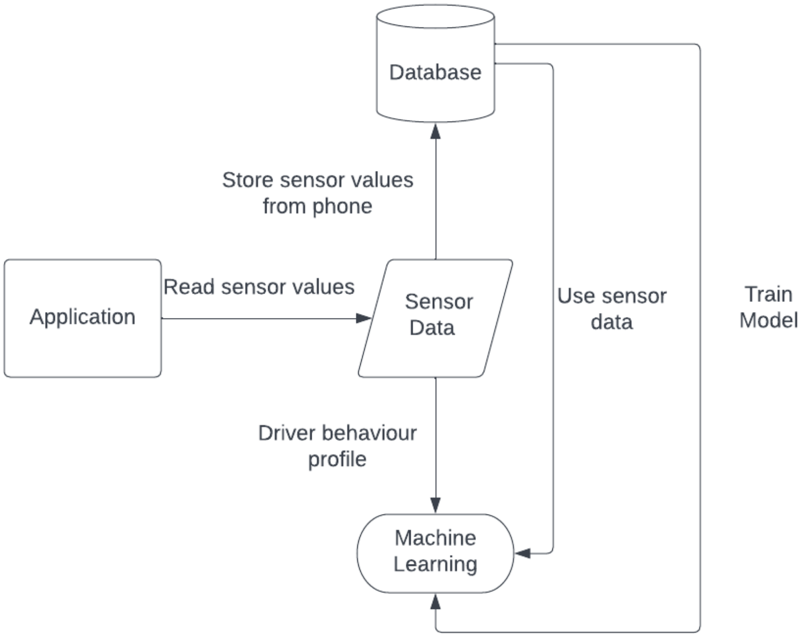
4. System Architecture



This diagram conveys how components of the application combine and work together. Once a user logs in they can start a car journey or view their driving statistics. If a user has built up journey data, they can view various statistics regarding their overall journeys or specific journeys through database values. They can start a new journey, the model will be trained using the Kaggle dataset, the decision tree model will then make a prediction on driver behavior, giving percentages on how much the driver was an aggressive, normal or slow driver. based on the sensor values parameters that are passed through. A new screen will display the driver behavior and data. The app will also notify users of excessive speed limits if the journey has started.

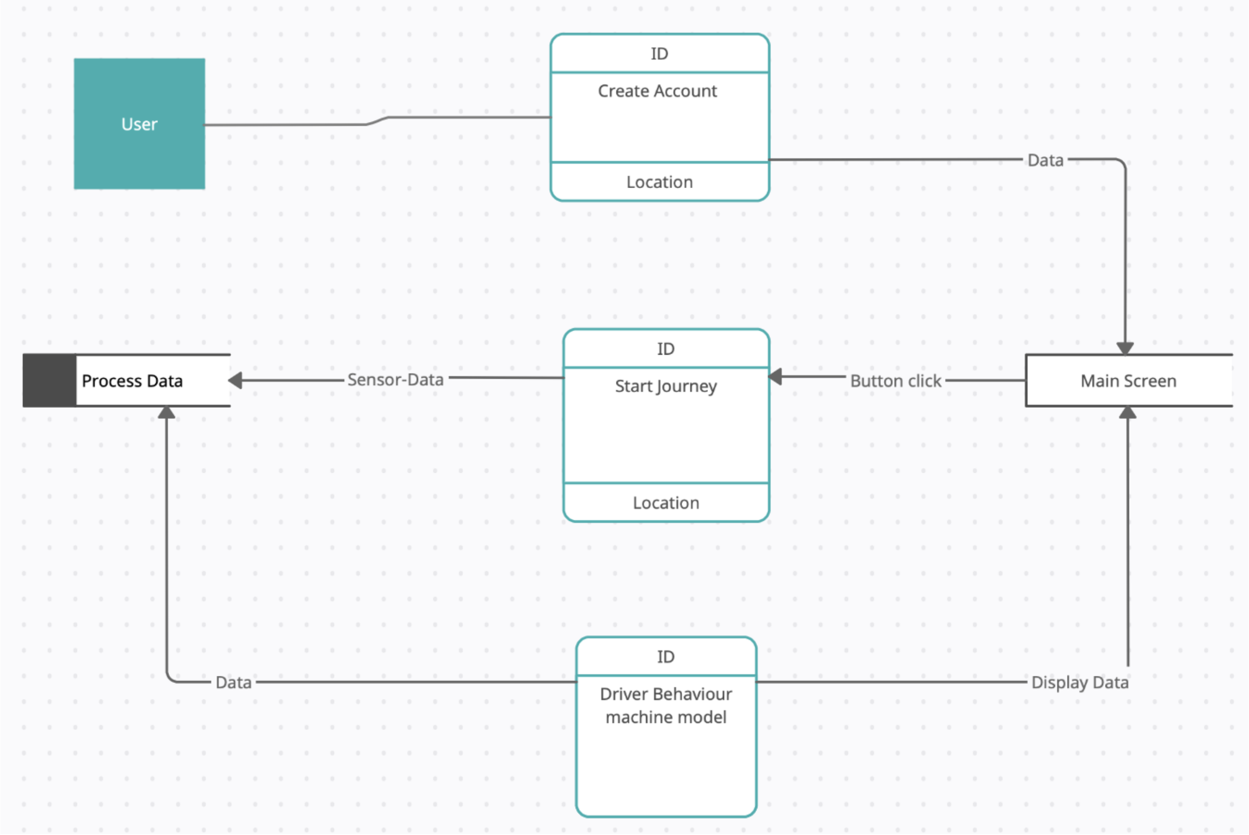
5. High level design

5.1 Context Diagram



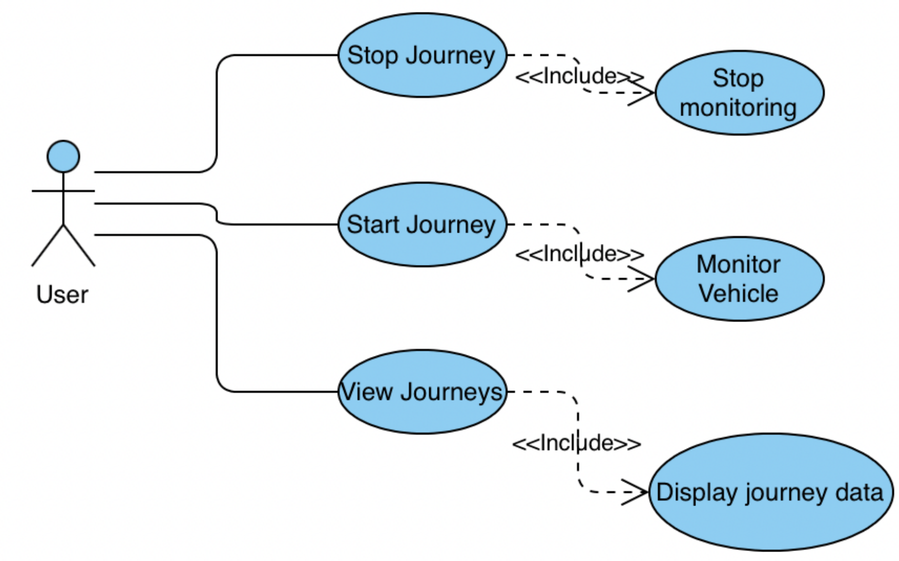
A context diagram is known as a level 0 diagram. They are meant to be simple to give a general overview of the application and show the overall process of it.

5.2 Data Flow diagram



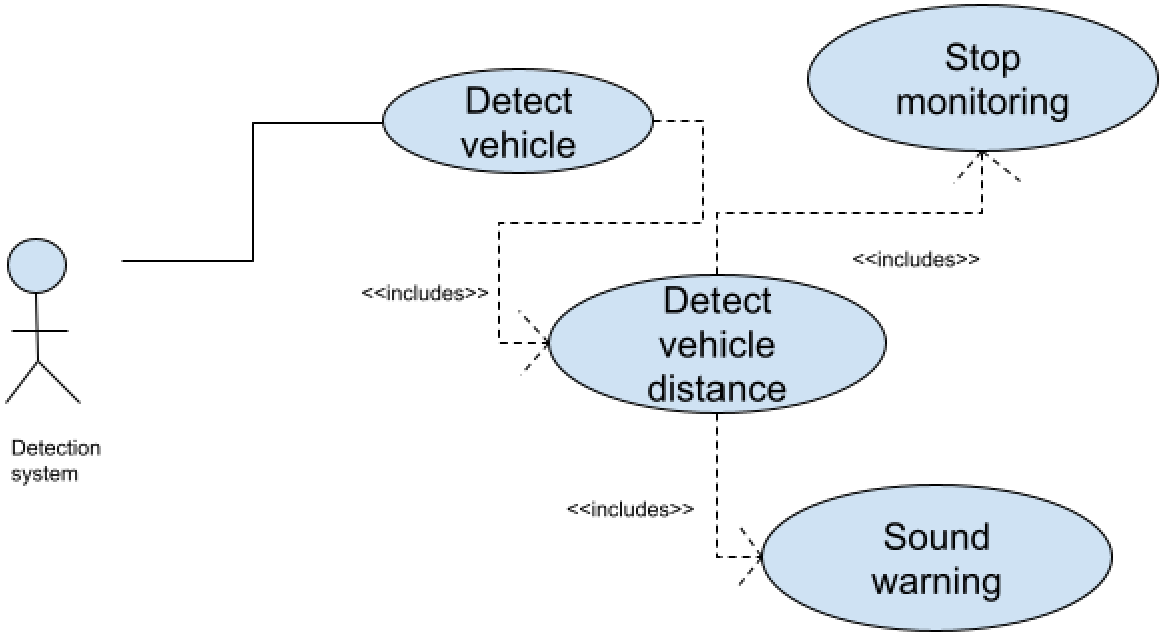
This is the data flow diagram for our application. It displays the process a user will go through when using the application. The first step a user will take is to create an account, this process is displayed in the diagram. The main screen of the app can start a journey through a button click, the data is processed from monitoring the vehicle, the machine model to describe driver behavior uses the processed data. All the data from monitoring the vehicle is then displayed on the main screen.

5.3 Use Case diagram – Start journey.



A use case diagram is how the user will interact with the system. This diagram is used to depict how the user will interact with the Journey functionality of the application.

5.4 Use Case diagram – Vehicle detection.



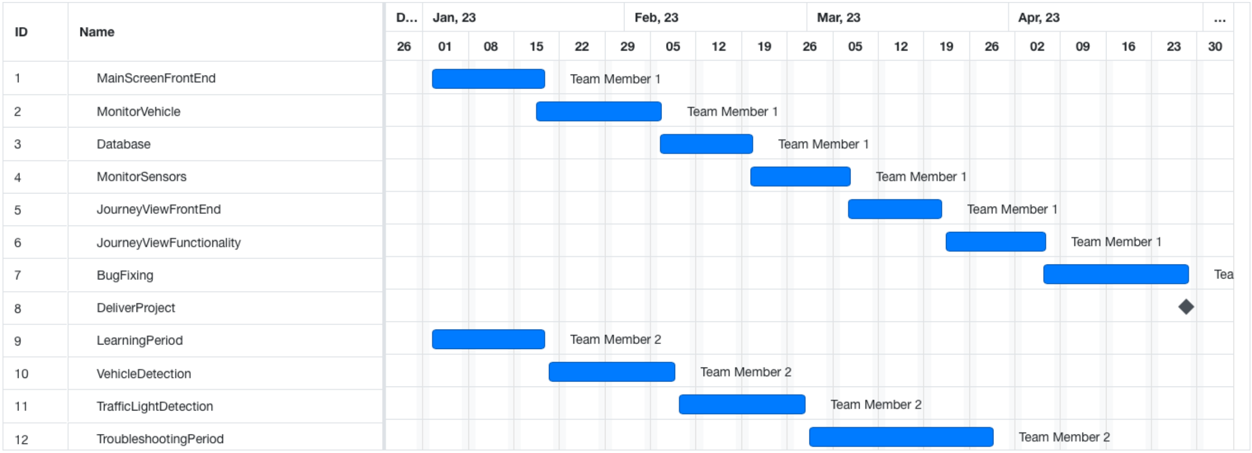
6. Preliminary Schedule

Software/Hardware tools Development Environment : Android Studio

Database : SQLite

Frontend development : Javascript, xml Backend development : Javascript, python

Gantt representation:



ID	Name
1	MainScreenFrontEnd
2	MonitorVehicle
3	Database
4	MonitorSensors
5	JourneyViewFrontEnd
6	JourneyViewFunctionality
7	BugFixing
8	DeliverProject
9	LearningPeriod
10	VehicleDetection
11	TrafficLightDetection
12	TroubleshootingPeriod