# PROJECT REPORT

# ON

# SMART TRAFFIC

**Submitted By:**

**Rishab Patnaik(6)**

**Sumeet Choudhary(5)**

# ACKNOWLEDGEMENT

Apart from our efforts, the success of this project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take immense pleasure in thanking and warmly acknowledging the continuous encouragement, invaluable supervision, timely suggestions and inspired guidance offered by our project mentor. Ashutosh Behura, Professor, School of Computer Engineering, KIIT University, in bringing this report to a successful completion.

We also express our sincere thanks to all our friends who have patiently extended all sorts of help for accomplishing this undertaking. Also, we would like to express our heartfelt thanks to each of our beloved parents for their blessings, for their help and wishes for the successful completion of this project.

Finally we extend our gratefulness to one and all that are directly or indirectly involved in the successful completion of this project work.

**Rishab Patnaik(6)**

**Sumeet Choudhary(5)**

# ABSTRACT

Traffic signals are one of the sectors where technology and advancements have not been utilized to the fullest potential. Most populated countries like India, China etc they have huge traffic congestions on road.

India's road condition are not expanding/improving in proportion with the increase in vehicle numbers. Times of India(newspaper) estimates that the annual cost of traffic congestion in India is **RS 60,000 crore** which includes fuel wastage and lost productivity. Much of this congestion could be addressed with Internet of Things technology.

Usually in any traffic signal , it gives green light only after a particular time period (usually 60 seconds) . But we believe that traffic signals should change only after it has analysed whatever lane is the busiest one. This can be analysed by thermal sensors that will analyse the heat of each lane. The busiest lane will be then given the green light.

Morever most of the traffic occurs due to people trying to find parking spaces. So we believe we can fix this problem by attaching sensors to the parking sensors to indicate wheather a particular space is free or not. These sensors wouldn't need much power as they would run on small lithium batteries that would run for considerable amount of time before replacing it.

These sensors in each case will upload the data to a controlling computer or to a cloud application on the internet and drivers could access the data in real time. Rather then seeking for empty parking spaces, drivers would know actually where to go.

# TABLE OF CONTENTS

# 1. INTRODUCTION

## 1.1 METHODOLOGY

The **spiral model** is a risk-driven process model generator for software projects. Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, or evolutionary prototyping.

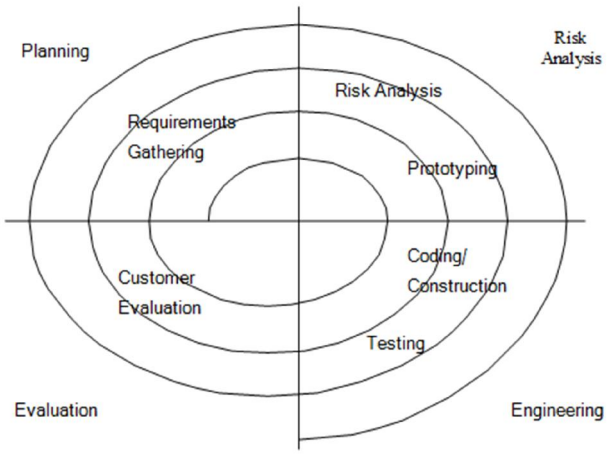
A software project repeatedly passes through these phases in iterations (called Spirals in this model). The baseline spiral, starting in the planning phase, requirements is gathered and risk is assessed. Each subsequent spiral builds on the baseline spiral.

**Planning Phase:** Requirements are gathered during the planning phase. Requirements like 'BRS' that is 'Business Requirement Specifications' and 'SRS' that is 'System Requirement specifications'.

**Risk Analysis:** In the **risk analysis phase**, a process is undertaken to identify risk and alternate solutions. A prototype is produced at the end of the risk analysis phase. If any risk is found during the risk analysis then alternate solutions are suggested and implemented.

**Engineering Phase:** In this phase software is **developed**, along with **testing** at the end of the phase. Hence in this phase the development and testing is done.

E**valuation phase:** This phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral.



# 1.2 RISKS INVOLVED

1) **Operational Risk**

- Sensors might be not in working condition giving wrong data.
- Quality of sensors and the products that are working in the system.
- Goverment constraints in the construction of the system.
- Is the system really required in that area?

2) **Technical Risk**

- Estimated size of the system.
- The size of the database created or used by the system.
- Number of changes according to the requirement of the system?
- Number of other products/systems with which this product must be interoperable.
- Speed of data access from remote database.
- Do analysis and design tools deliver method that is appropriate for the system to be built?

3) **Schedule Risk**

- Wrong time estimation
- Resources not tracked properly.
- Unexpected project scope expansion

# JUSTIFICATION FOR USING SPIRAL MODEL

- Spiral model helps in handling the unforeseen risks that can show up much later in the system.
- Spiral model provides the scope to build the prototype at every phase of the system development.
- Alternative solutions are evaluated to select the best possible solution.
- Reviewing the result of stages traversed so far with the public reduces the risks

# 1.3 PURPOSE

Internet of Things (IoT) links the objects of the real world to the virtual world, and enables anytime, anywhere connectivity for anything that has an ON and OFF switch. It constitutes to a world where physical objects and living beings, as well as virtual data and environments, interact with each other. Large amount of data is generated as large number of devices are connected to the internet. So this large amount of data has to be controlled and converted to useful information in order to develop efficient systems. In this paper, we focus on to an urban IoT system that is used to build Intelligent Traffic System (ITS). IoT based Intelligent Trafic Systems are designed to support the Smart City vision, which aims at employing the advanced and powerful communication technologies for the administration of the city and the citizens.

As the Wireless Sensor Networks have technologically developed more rapidly and more efficiently, they have become the key source for the development of IoT.We aim to manage traffic congestion with the help of IOT.We can calculate the congestion on each lane by using preassure pads. The roads will itself act as preassure pads and the iot system will let go of the cars which has maximum congestion on road(calculated through preassure pads).

Additional benefits include parking guidance. Rather than driving around the whole area looking for space, the drivers would be told over the WiFi about the vacant spaces available near to their location. In addition to this, the drivers would be intimated with the shortest possible paths to reach the destination so that carbon dioxide emissions can be controlled.

# 1.4 SCOPE

India’s road condition are not expanding/improving in proportion with the increase in vehicle numbers. Times of India(newspaper) estimates that the annual cost of traffic congestion in India is **RS 60,000 crore** which includes fuel wastage and lost productivity. Much of this congestion could be addressed with Internet of Things technology.

We believe by using this system we can reduce this cost to about 15%.

The system, replicates the human intelligence, but instead of a traffic policeman, "we want the system to do the job".

It is what manually a traffic policeman does. It will allow vehicles to pass in the street as long as there is traffic on it, and then after clearing it, he switches to the other street at the junction.

With a few minor changes to the road system we will be able to make a change that will be efficient for the years to come and save the system a lots of money.

# 1.5 TOOLS USED

1. **Arduino** is an open source, computer hardware and software company, project, and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world.
2. **Wamp Server**

   **WampServer** is a Windows web development environment. It allows you to create web applications with Apache2, PHP and a MySQL database. It also comes with PHPMyAdmin and SQLite Manager to easily manage your databases. WampServer installs automatically (installer), and its usage is very intuitive.
3. **MySQL**

   MySQL is the world's most popular open source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties.
4. **PHP**

   Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps
5. **STARUML**

   A sophisticated software modeller. It can create Entity-Relationship Diagrams (ERD). ERD is one of the most frequently used diagram for database modelling. Quick Edit to create elements and relationships at once such as sub-classes, supporting interfaces, etc.

# OBJECTIVE

- To create a smart traffic system to reduce the traffic congestion on the roads.
- Notify the users of the free parking space on his/her mobile.
- Traffic System will be sensing other factors like rain and etc. and the itensity of traffic lights will depend on that.

# ISSUES WITH EXISTING SYSTEM

- **Indian Traffic System:** Existing systems works on a deafault time of mostly 1 min and keep changing the lights based on that.
- This system is not a valid one because it doesnot take into account of the factor of traffic congestion.

# PROJECT PLANNING

## Requirement Gathering

- **Brain Storming**: Thinking about what the real problems that are taking place in today's world. We discussed among ourselves, the major problems in today's world and came to the conclusion of smart traffic system .
- **One-on-one interviews**: The most common technique for gathering requirements is to sit down with the public and ask them what they need. We asked different public, their problems while they deal with; while using the existing system.
- **Prototyping** is a relatively modern technique for gathering requirements. In this approach, you gather preliminary requirements that you use to build an initial version of the solution. We gathered these requirements to start building upon it.

# At Functional Requirements:

## R1. User arrives in lane

**Input:** Cars weight.

**Output**: Preassure pads increases the factor by 1 to show that a car has arrived.

## R1.2. Calculation of traffic signals time.

**Input:** Count of cars in each lane taken by preassure pads.

**Output**: The most congested lane will be given highest green signal time.

## R2. Users arrive at a paking area.

**Input:** User enter the location of which Parking area he is at.

**Output**: The parking systems map of which parking spot is wmpty is shown.

**Processing**: Accesing the database and then showing the records obtained from the sensors.

## R2.1 Update Parking Spots

**Input:** User makes his car park at a parking spot whose data is picked up by the sensors

**Output**: The spots are updated on the server of which parking spots are now empty and which are occupied

# Non-Functional Requirements:

**24x7 Availability**: The smart traffic system is available 24X7 managing traffic all the time.

**Performance Requirements**: Performance of the system depends on the response time and the speed of the data submission. The response time of the system is direct and the application is real-time. System should have a fast response time which depends on the efficiency of the implemented algorithm.If the system is not real-time it is not useful.

The first version of the system will have limited implementation hence there will be no need for a large network. However, it may grow depending on the increase in usage.

**Safety Requirements**: System has to check-

- If Database content is syntactically well formed.
- If another server is not running on the same port which might not allow Apache-MySQL server to run.
- If web forms with the services processing form input are consistent.
- Statically safe binding of code of session operations to variables defined with session scope.

**Security Requirements**: For security of the system the technique known as database replication should be used so that all important data is kept safe. In case of crash, the system should be able to backup and recover the data. Traffic Manager should only be given the access for database monitoring and insights from the data.

**Software Quality Attribute**: The system will have a simple and we will be using basic sensors which can easily collect dataset and act upon it smartly. Flexible service based architecture will be highly desirable for future extension.

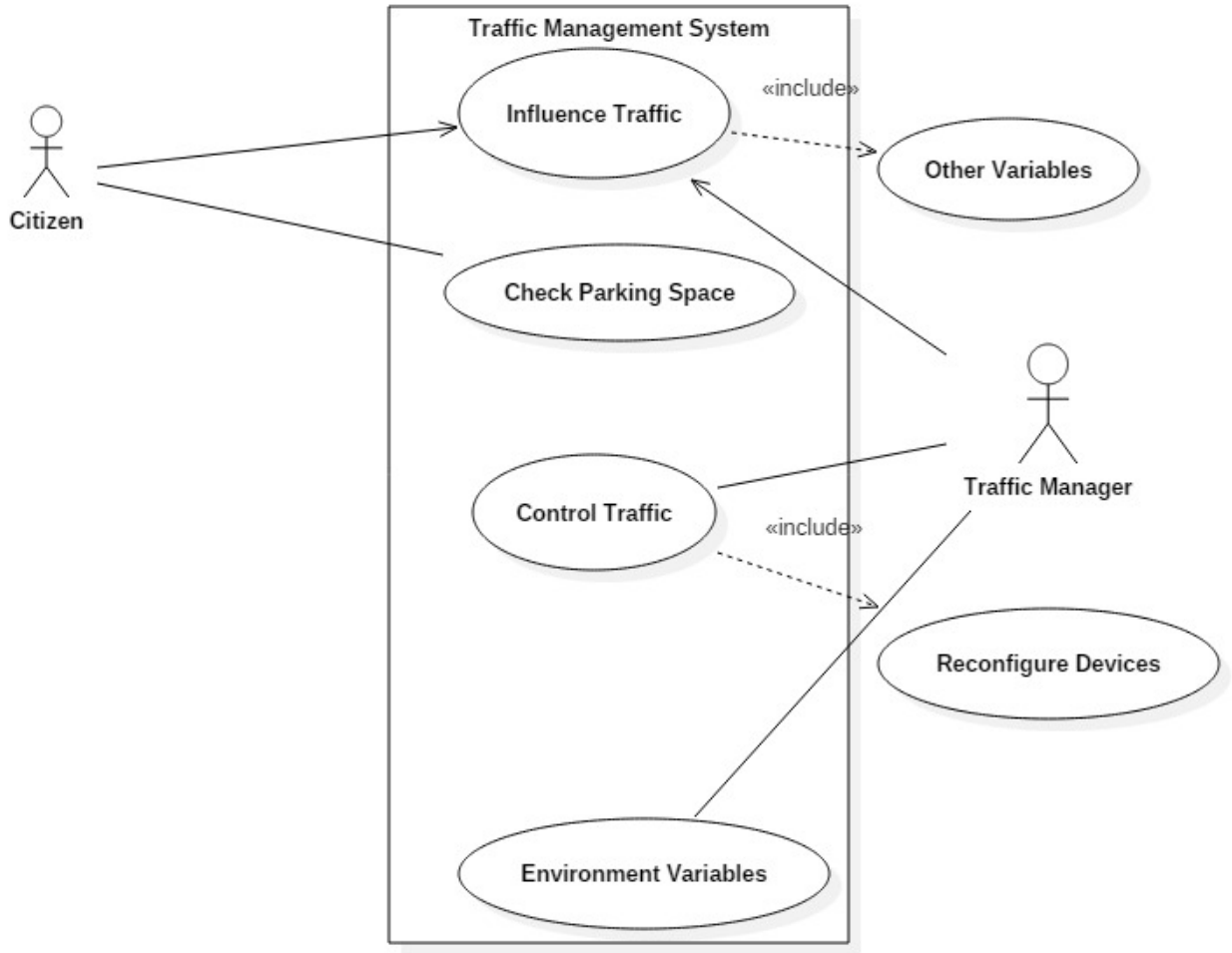
## 3.4 Constraints:

- The application will be useful only in the high or variable traffic areas.
- The citizen have to lookup the webpage for the parking space availibility from a table format.
- Only citizen having smartphone can utilize the services provided.
- Citizens who have the knowledge of accessing net facilities can only utilize the offered services

# 4. DESIGN

## 4.1Use-Case Diagram

An **use case diagram** at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.




Use Case Diagram for Traffic Management

## 4.2Class Diagram

In software engineering, a **class diagram** in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

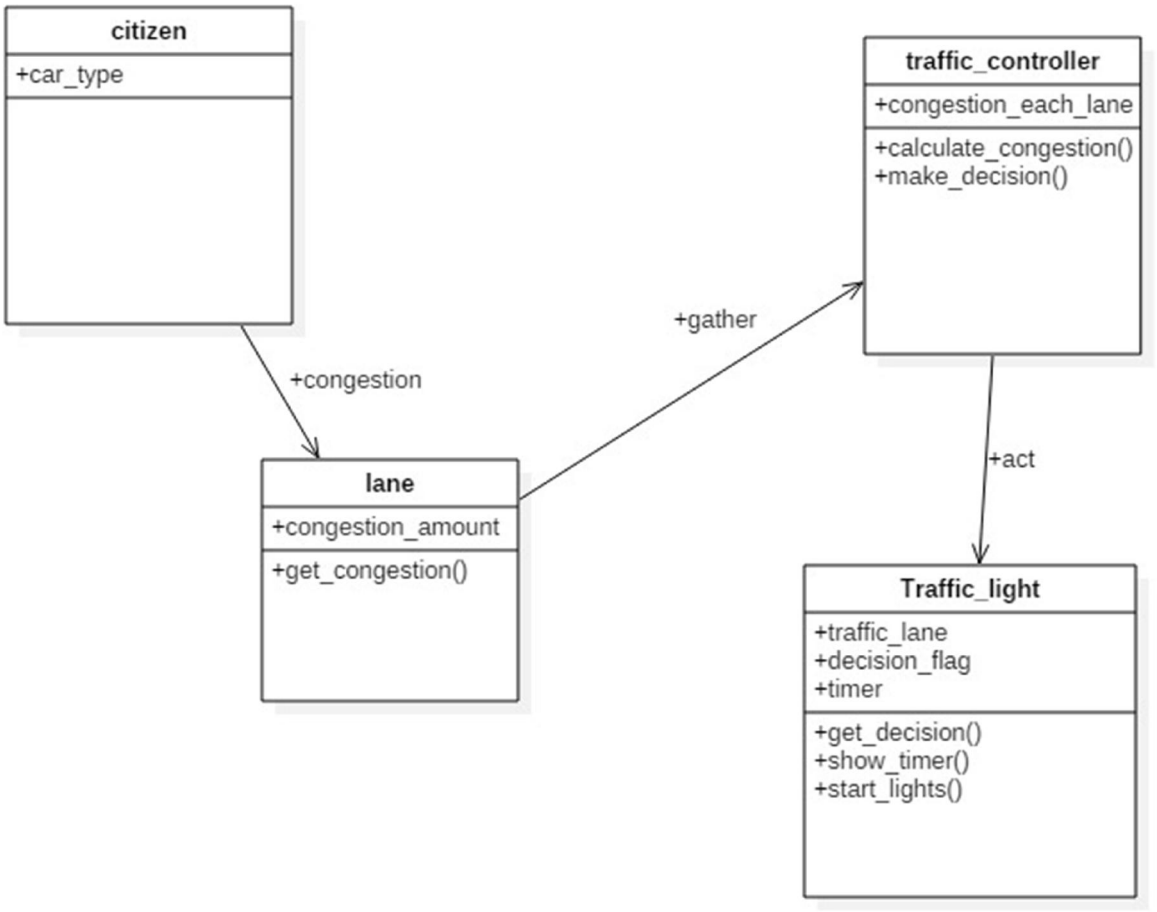
The class diagram is the main building block of object-oriented modelling. It is used both for general conceptual modelling of the systematics of the application, and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

In the diagram, classes are represented with boxes that contain three compartments:

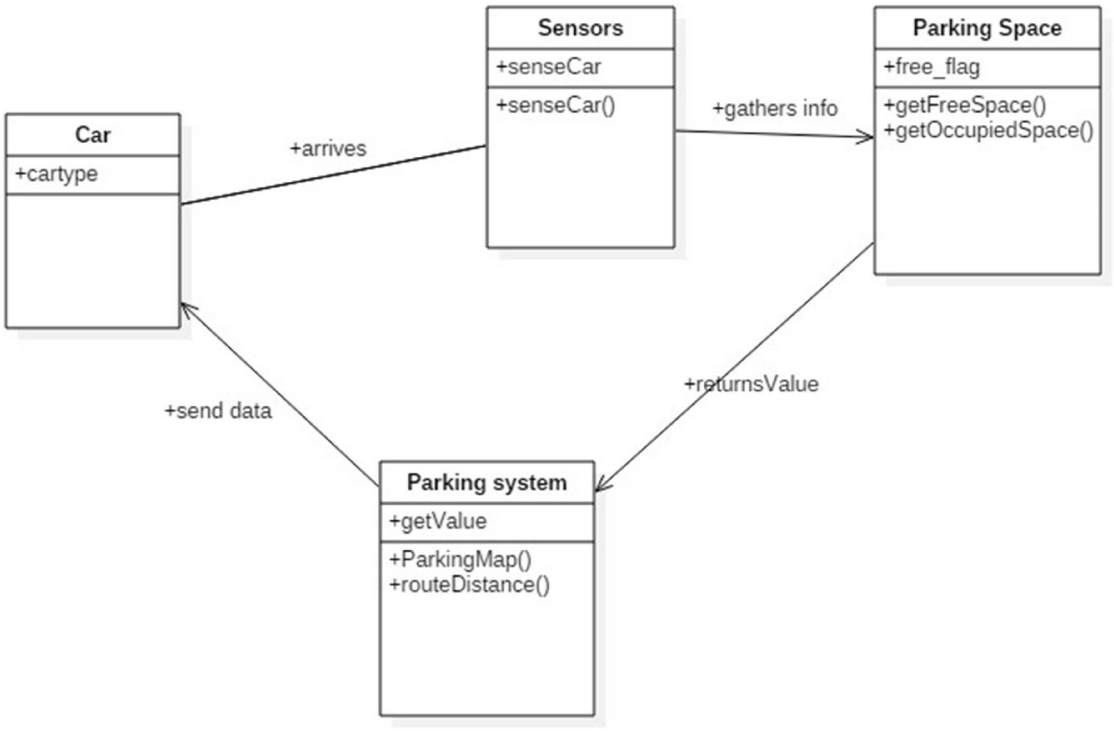
- The top compartment contains the name of the class. It is printed in bold and centered, and the first letter is capitalized.
- The middle compartment contains the attributes of the class. They are left-aligned and the first letter is lowercase.
- The bottom compartment contains the operations the class can execute. They are also left-aligned and the first letter is lowercase.

In the design of a system, a number of classes are identified and grouped together in a class diagram that helps to determine the static relations between them. With detailed modelling, the classes of the conceptual design are often split into a number of subclasses.

In order to further describe the behaviour of systems, these class diagrams can be complemented by a state diagram or UML state machine.




1.Class Diagram of Traffi Management




# 2.Class Diagram For Smart Parking

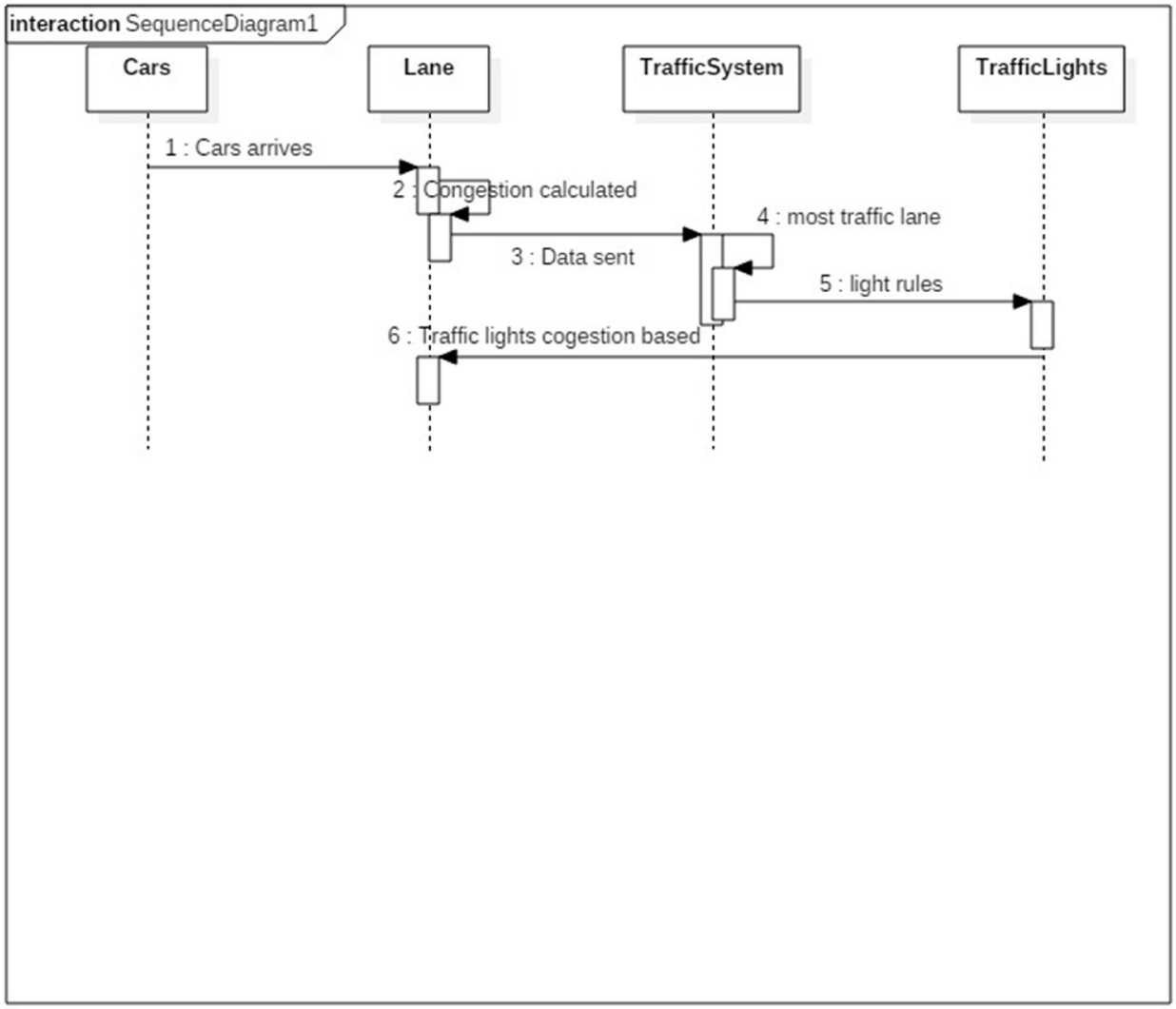
# 4.3 Sequence diagram

A **Sequence diagram** is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart.

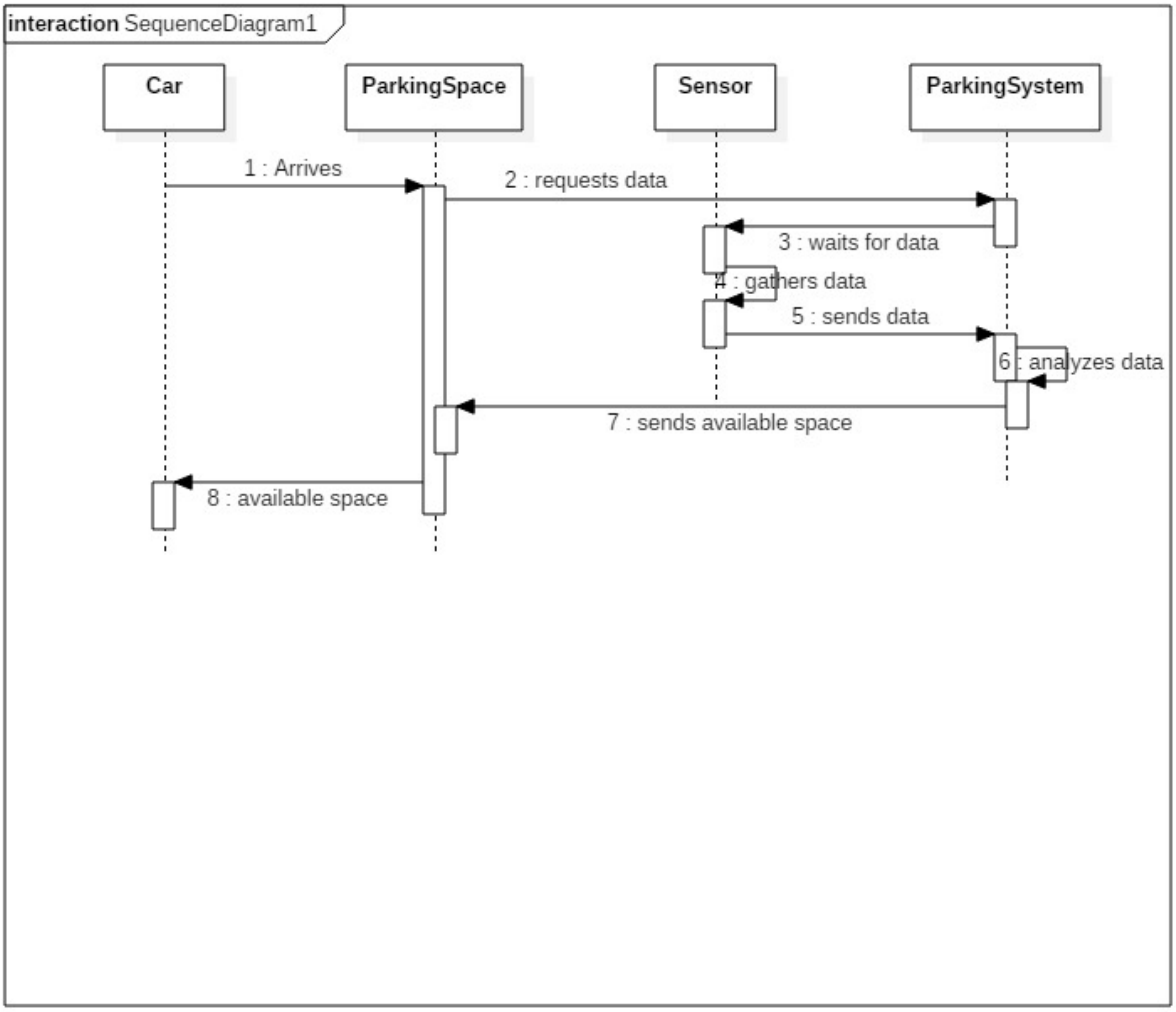
A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams** or **event scenarios**.

A sequence diagram shows, as parallel vertical lines (*lifelines*), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner

# 1.Smart Traffic Managment:



# 2.Smart Parking:



# IMPLEMENTATION

-We are going to use PHP and MYSQL to create a database and backend API's and it will help in storing User's cars is parked or not.

-Using HTML, CSS and java script for the web app and creating the front end of the project to make the software more attractive and User-friendly.

-We have also used Proteus Designer for designing the model of our project.

# CONCLUSION

Traffic Management will dynamically access the density of vehicles on each of the lanes in the real-time to handle the congestion in traffic. It will be able to differentiate between lanes according to congestion and provide proirity access to lane manages time with them. It will help in keeping all the information of the lane in a particular area .Because there are many cases where one lane having very few vehicles are provided with same priority which is not efficient.

It will help in dealing with the demerits of the present traffic system. Dealing with the traffic congestion and then working with the traffic system is very more efficient because there is no point in time oriented traffic system if there is more traffic on one lane and zero on other.

Morever many of the people cause a lot of traffic because they spend a lot of time in finding a parking space and also spent a lot of fuel too.

Smart traffic sees to all of these problem in a smart and simple way to manage all of the traffic and also provide a hassle free parking solution.

# BIBLIOGRAPHY


- https://www.w3schools.com/
- Software Engineering-Rajib Mall