A PARTIAL PROJECT REPORT

ON

Avoiding Road Traffic Congestion using Dynamic Traffic Assignment Approach

SUBMITTED BY

Adil Hussain : B120314204
Mohammad Moheed Inamdar : B120314249
Ajay Singh Rajpurohit : B120314310
Utkarsh Alone : B120314207

UNDER THE GUIDENCE OF

Mrs. Deepti Nirwal

DEPARTMENT OF COMPUTER ENGINEERING
P.E.S MODERN COLLEGE OF ENGINEERING
PUNE - 411005.

[2017 - 18]

Progressive Education Society's
Modern College of Engineering
Department of Computer Engineering
Shivajinagar, Pune - 411005.

CERTIFICATE

This is to certify that the following students of Final Year Computer Engineering have successfully completed the preliminary analysis and design of project entitled "*Avoiding Road Traffic Congestion using Dynamic Traffic Assignment Approach*" for the organization "PES Modern College Of Engineering"

The Group Members names are: Adil Hussain
Mohammad Moheed Inamdar
Ajay Singh Rajpurohit
Utkarsh Alone.

This is in partial fulfilment of Bachelor of Computer Engineering under Savitribai Phule Pune University.

Date:

(Mrs. Deepti Nirwal)	(Prof. Dr. Mrs. S. A. Itkar)	
Internal Guide	Head	External Examiner
	Department of Computer Engineering	

Acknowledgement

It gives us pleasure in presenting the partial project report on **‘Avoiding Road Traffic Congestion using Dynamic Traffic Assignment Approach’**.

Firstly, we would like to express our indebted appreciation to our internal guide **Mrs. Deepti Nirwal**. Her constant guidance and advice played very important role in making the execution of the report. She always gave us her suggestions, that were crucial in making this report as flawless as possible.

We would like to express our gratitude towards **Prof. Dr. Mrs. S. A. Itkar** Head of Computer Engineering Department, PES Modern College of Engineering for her kind co-operation and encouragement which helped us during the completion of this report.

Also we wish to thank our Principal, **Prof. Dr. Mrs. K. R. Joshi** and all faculty members for their whole hearted co-operation for completion of this report. We also thank our laboratory assistants for their valuable help in laboratory.

Last but not the least, the backbone of our success and confidence lies solely on blessings of dear parents and lovely friends.

Adil Hussain
Mohammad Moheed Inamdar
Ajay Singh Rajpurohit
Utkarsh Alone

Contents

Abstract

Traffic management is a tedious task riddled with a lot of uncertainity of buildup and flow changes. The largest issue regarding Traffic Management is its unpredictability of changes. This Project aims to tackle this issue with the use of incremental changes that counteract signs of Traffic buildup. The idea is to detect changes in the traffic density (from cycle to cycle) and slowly increase/decrease signal timings for the growing/shrinking route density or overall.
This allows us to do 2 things at once, viz.

1. Prepare and provide proportional timings to each road of the signal and

2. Dynamically increase/decrease the overall signal cycle time based on density growth/reduction.

Our system is ideal for use in a Smart City environment or any environment that provides GSM internet or mobile data abilities near the roads.

List of Figures

List of Tables

List of Abbreviations

API .... Application Programming Interface

GTAPI .... Google Traffic API

JSON .... JavaScript Object Notation

XML .... eXtensible Markup Language

CSS .... Cascading Style Sheets

VANET .... Vehicular Ad-hoc NETwork

IoT .... Internet of Things

WSN .... Wireless Sensor Network

IDE .... Integrated Development Environment

GSM .... Global Systems for Mobile

MOTUS .... Microscopic Open Traffic Simulator : http://homepage.tudelft.nl/05a3n/

UI .... User Interface

P2P .... Peer to Peer

NIC .... Network Interface Card

1.

Introduction

1.1 Brief Description

The problem statement describes a problem that is very complex to deal with. It contains many variables like density of traffic, frequency of input, probability of failure, location of failure, area/capacity of channel, etc. Finding an equation that defines the flow in the network using these variables is a tedious task as the behaviour of such networks is not defined with respect to each variable. Also, conducting experiments over the network in an attempt to define relations between one variable and the traffic (keeping all other variables constant) is limited to simulations. Such experiments cannot be conducted in the real world.

Many solutions use IoT devices embedded inside vehicles (or VANET) that solve this problem by finding the density and urgency of each vehicle to adjust traffic signal timings. Other solutions use WSN to identify traffic density using infrared lasers across the road to mark low, medium and high traffic. This technique adapts the traffic signal based on the amount of backlog in each lane that builds up over the ”red” duration of the traffic signal for a given lane.

We observe that these variables have slow variations over time for the case of Road Traffic. This allows us to observe activities as they happen and make changes to accomodate the traffic. Our solution uses Incremental Approach since it is a common approach used to tackle problems in slowly changing systems. It solves these issues by developing a continual monitor-evaluate-modify loop that tends to adapt to problems with no single solution.

1.2 Problem Statement

Devise a Distributed Control system to solve the Traffic Assignment Problem on City Roads.

1.3 Objectives of the Project

1. To Observe Current Traffic Conditions.
2. To Mediate Road Traffic Flow.
3. To Dynamically Control Traffic Signals.

1.4 Scope of the Project

1. IoT Based Traffic Signal Control (Arduino)
2. Decentralized Algorithm to create changes in Signal timings

2.

Literature Survey

2.1 Literature Survey

Sr. No.	Title	Author	Journal	Purpose
1	An Integrated and Scalable Platform for Proactive Event-Driven Traffic Management	Alain Kibangou, Alexander Artikis *et. al*	ArXiv.org, March 2017	SPEEDD project, ML predictive approach
2	Self-organizing Traffic Lights	Carlos Gershen-son	Complex Systems, 2005	Naive Distributed Traffic Signal Control
3	Centralized and Localized Data Congestion Control Strategy for Vehicular Ad Hoc Networks Using a Machine Learning Clustering Algorithm	Nasrin Taherkhani and Samuel Pierre	IEEE Transactions on Intelligent Transporta-tion Systems, 2016	Example of Ad-Hoc Approach (VANET)

Table 2.1: Literature Survey

3.

Design Details

3.1 Requirements Analysis

3.1.1 Project Requirements:

The Software Requirements are:

- Arduino IDE
- Redis Database - JSON/XML
- GTAPI
- Apache Tomcat
- Sublime editor
- Python v3.5
- JavaScript - Vue.JS, JQuery
- CSS - Sass

The Hardware Requirements are:

- Arduino Uno Rev3
- 3G/4G GSM Shield

The System Requirements are:

- Windows/Linux Operating System
- MOTUS
- Python's matplotlib.

3.1.2 Problem Description:

The Issues in Indian Traditional Traffic Management systems are that they are:

1. Inflexible: They do not adapt to changing situations in real time. In case of *accidents* or *disputes*, the traffic piles up rather quickly into a 1-3 hour jam.

2. Inadaptive: Traditional approach do not optimize signal's waiting times for each line according to the traffic in the incoming lanes.

3. Uncoordinated: Situations that arise in one area usually produce effects on nearby regions, though the remedial measures do not reach these other areas quickly enough.

4. Manual or Preset Control: The timings of each signal and the ability to switch them on/off need human interaction which are slow and/or inefficient.

3.1.3 Solution Description:

We propose a solution to these problem by first making some assumptions:

- The roads do not vary in width by a huge factor near the signal (500m).
- People do not usually jump the signal.
- Most travellers carry their smartphones.

Our solution to the problem uses a Mathematical system that is similar to one which is built to balance weights.

Let's assume that on a signal, the densities of incoming traffic is D_N, D_E, D_W and D_S (may vary as per number of directions/lanes). We then use the summation of these densities to decide upon the cycle time (C).

$$C = \frac{\sum D_d}{C_{avg}} \tag{3.1}$$

where C_{avg} is the average Cycle time among it's neighbours. Using these we divide the cycle time C into 4 parts (as per number of directions/lanes). These 4 cycle parts are used as total alloted time per lane.

From this alloted time or slots, we first identify which are too low or too high. These are categorized as Min_C and Max_C respectively. This is to ensure there is some meaningful minimal and some bounding maximum cycle slots to consider. Note that though there is a single Min_C, there are multiple Max_C for allowing a one way routed traffic on a cross section (Work Day traffic flows are usually one directional).

Then we split the slot time into Red, Green and Yellow timings of the Signal. These values are then provided to the Signal for control.
The higher orders of Max_C (also known as Max_C_i) come with a correctional cost. This cost is added back to the Cycle slot in the following manner.

$$C_{corrected} = C + \sum_{i=1}^{i=n} Cost(i) \tag{3.2}$$

The tendency of this correctional cost is to release more traffic in order to reduce the incoming traffic. This causes the densities to appear dynamic. The net result of this operation over time is to increase the traffic fluidity while economizing the whole cycle time. When these densities approach each other, the higher orders of Max_C are reduced to either Max_C or lower cycle slot time. This tends to make original cycle slot to the corrected cycle time.

Hence this Mathematical system automatically balances itself which in turn smoothen out traffic. Consequently, this system works independent of it's own instances and hence is a purely decentralized system.

3.1.4 Project Outcome:

On completion, we will have desgined:

- User's Software perspective:
  1. A Signal Controller software.
  2. A web interface to see signal timings generated by the system.
  3. A web interface to check locations of jams and/or other bottlenecks.
  4. A web interface to change/modify the routes (actual signals and associated lanes are shut down or rerouted) on-the-fly.
- User's Hardware perspective:
  1. A Signal Controller board.

3.2 SRS

3.2.1 Necessary Functions:

- Dynamically scale signal timings as per lane's incoming traffic.
- Dynamically reduce/eliminate the onset of traffic jams.
- On-the-fly Route configuration and modification.
- Visible Timings (Exposed info) of each signal controller.
- Authentication and User creation for Traffic Control Administrator(s).
- Automatic Detection, Reporting and Adaptation to Signal Controller Failures.

3.2.2 Desirable Functions:

- Android application to warn users of new changes in the Traffic system.
- Android webview of web interface to see Traffic status.
- Provision for adapting to emergency services (ambulance,fire brigade, etc.)

3.2.3 Interface Requirements:

User interfaces:

1. UI for visitors to see signal timings.
2. UI for authentication of Traffic Control Administrators.
3. UI for editing (with support for undo/redo) routes and signal operation.
4. Exposed API for using non-authenticated functionalities outside of UI.

Hardware Interfaces

1. A hidden P2P communication strategy for real-time monitoring network status.
2. Automatic route expansion in case of signal controller failure.

Software Interfaces:

1. Hidden P2P localhost server for notifying lane neighbours.
2. Web server to Controller interface for operating on controller (switching on/off, changing route, watching timings).

Communication Interfaces:

1. 3G/4G GSM internet connection.
2. IoT connectivity between signal controllers.

3.2.4 System Features:

1. Speed - Use of Microcontroller makes sure that the hardware doesn't prove to be the bottleneck of the System's operation.
2. Consistency - Use of Microcontrollers allow the software to not generate corrupt structures as the code will not change or get corrupted from use (as is the case of overheated microprocessor systems).
3. Decentralized - No single point of failure exists in this system.
4. Adapative - System is capable of handling many situations by itself in real time.
5. Security - Inner working of Authentication and Modifications uses Hashes that are untraceable to their sources. Hence they mark events instead of Credentials or System's Data.

3.3 Design Phase :

3.3.1 Architecture Block Diagram

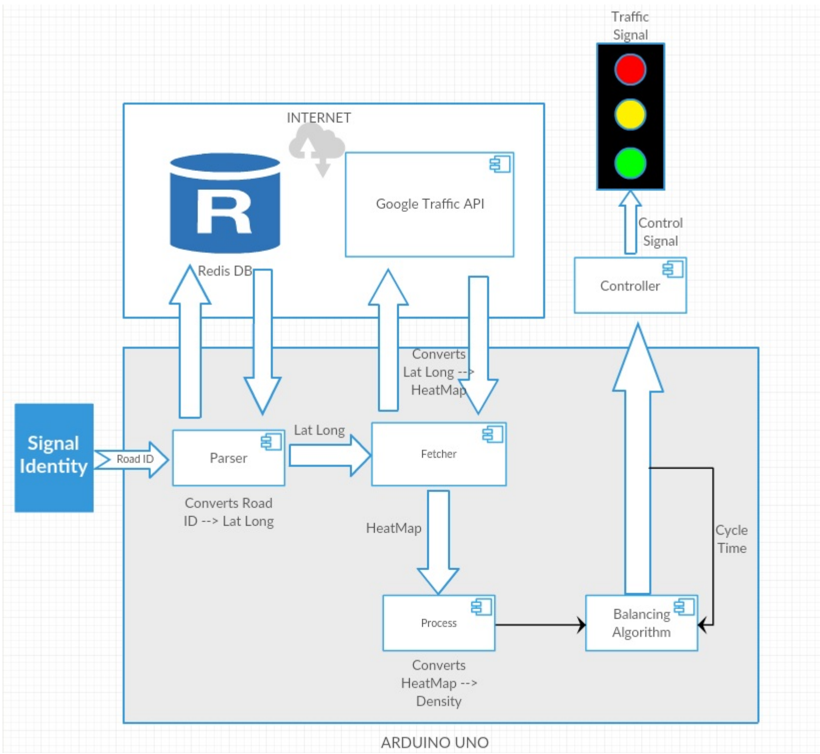

Figure 3.1: Architecture Diagram

3.3.2 Structural Diagrams

Class Diagram

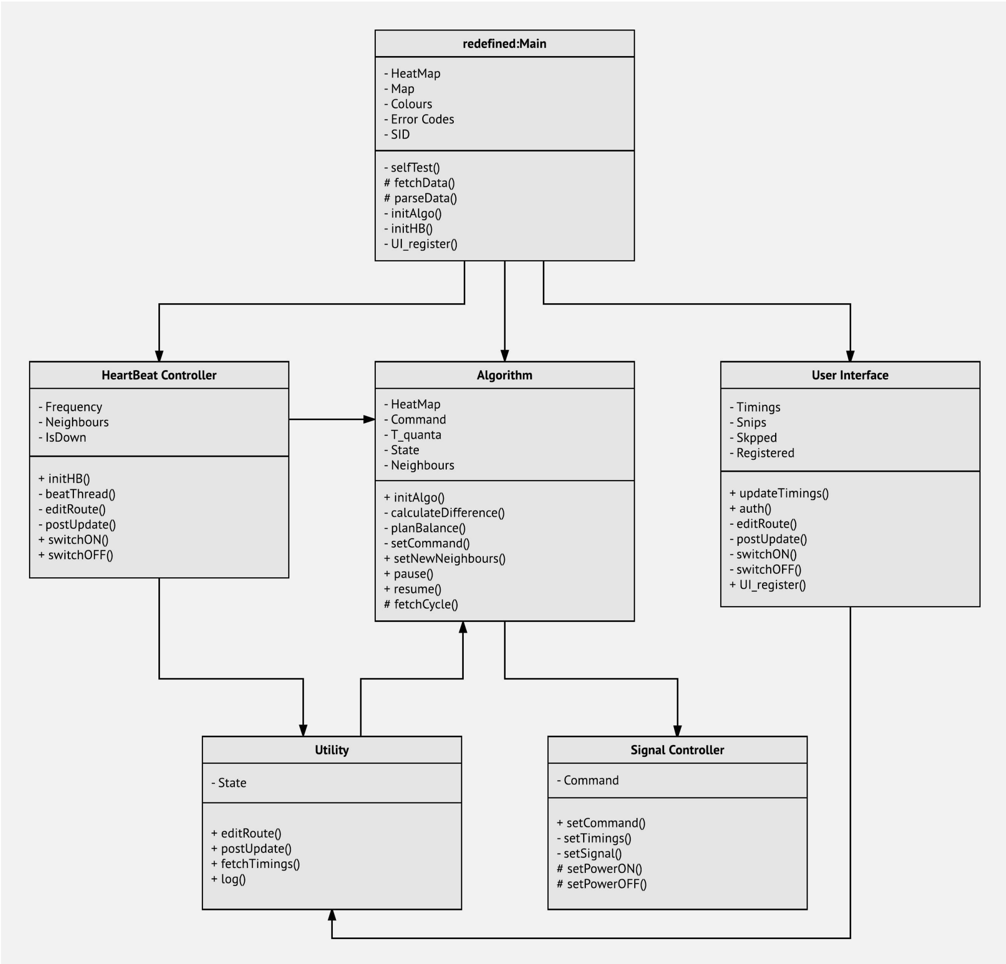

Figure 3.2: Class Diagram

Object Diagram

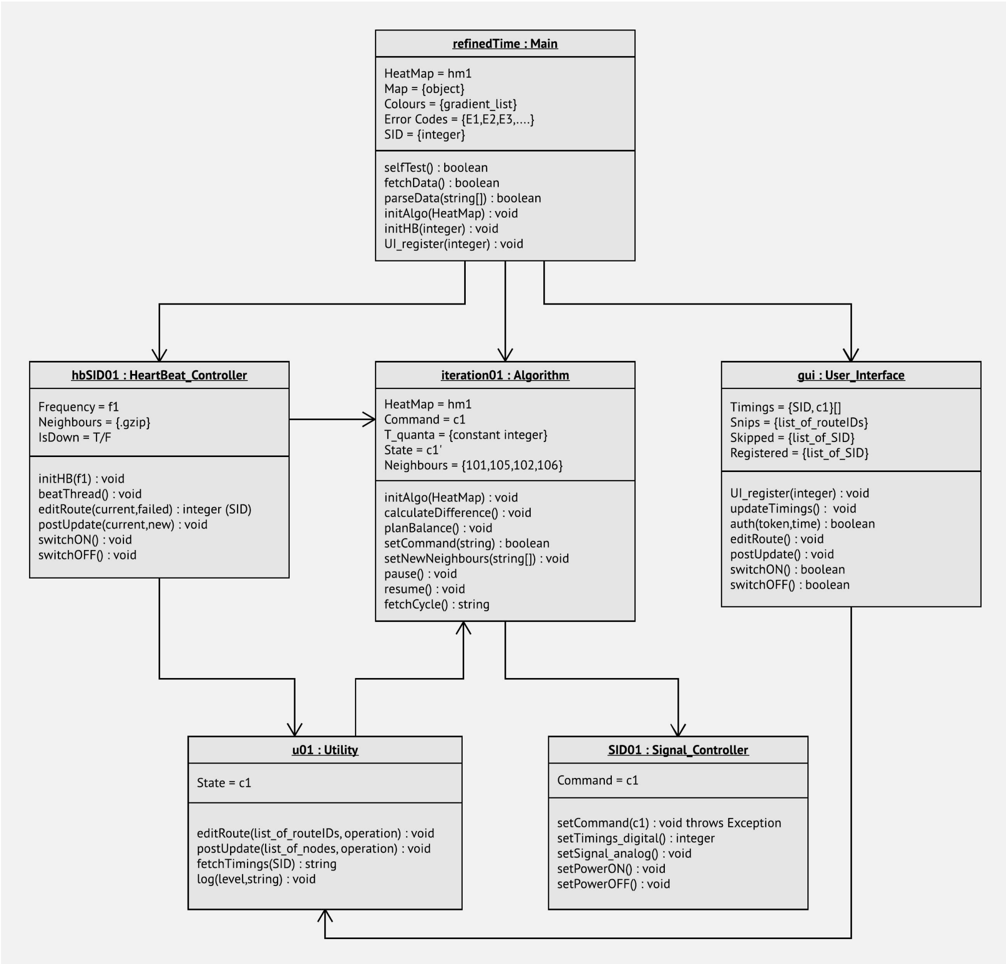

Figure 3.3: Object Diagram

Component Diagram

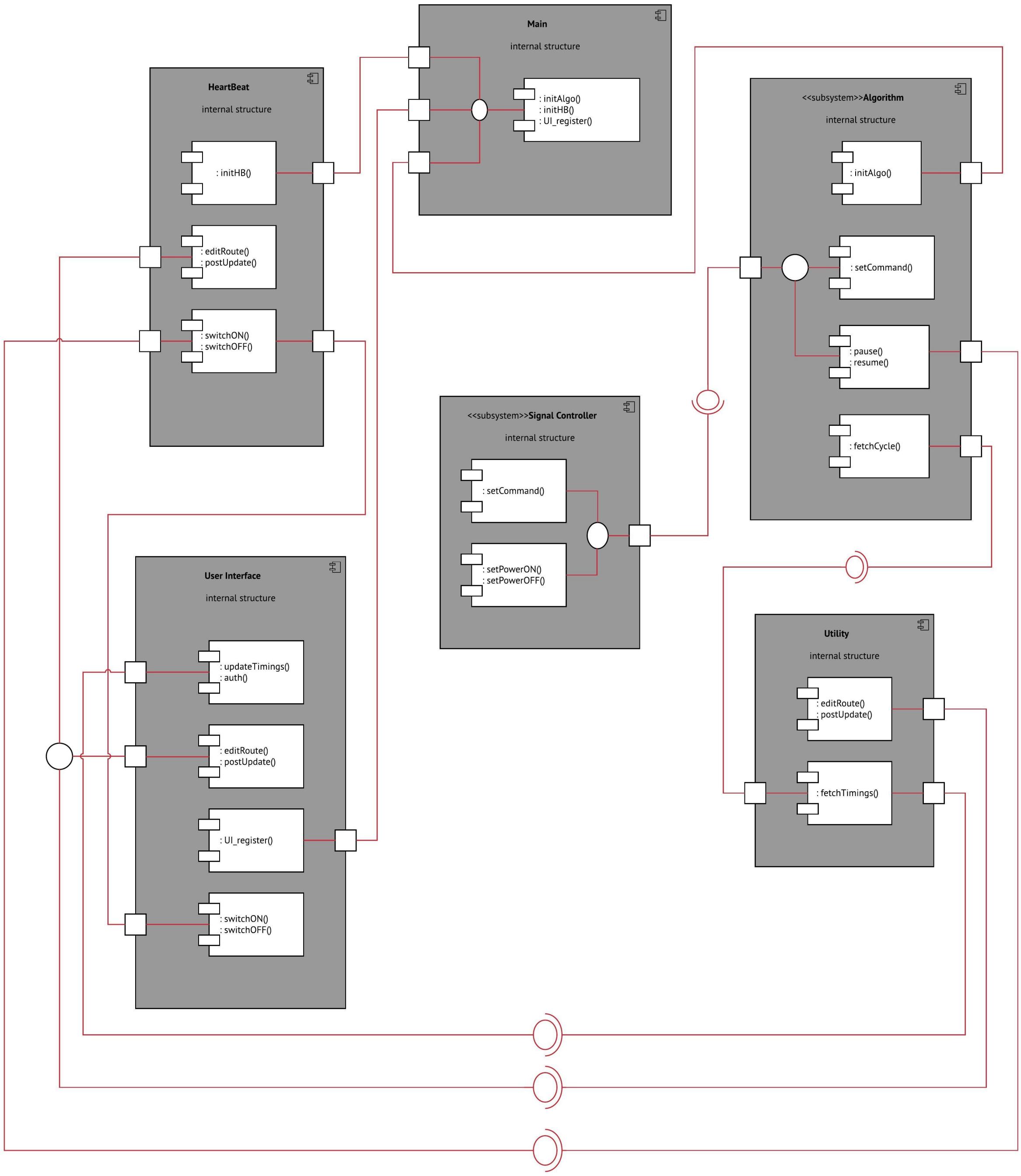

Figure 3.4: Component Diagram

3.3.3 Behavioural Diagrams

Use Case Diagram

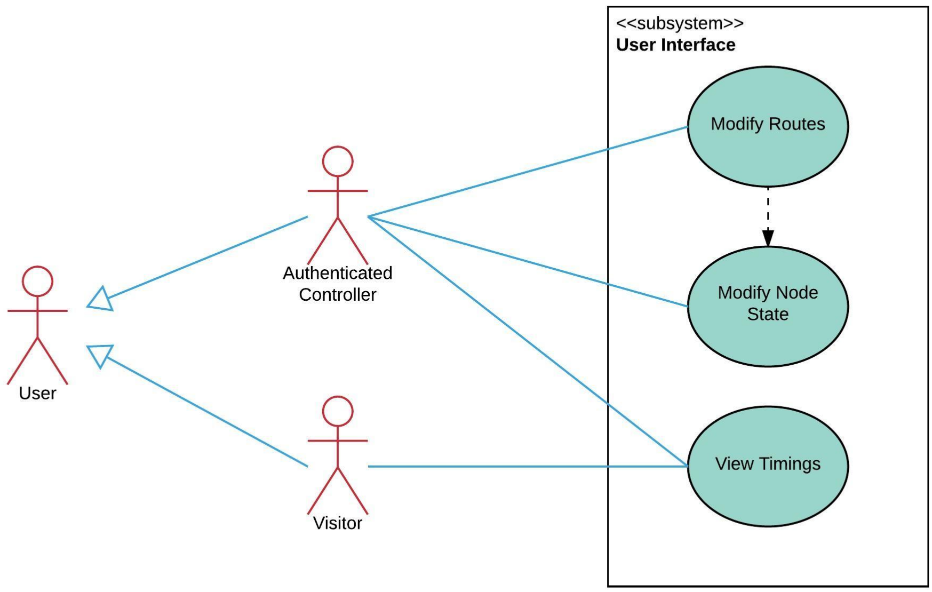

Figure 3.5: Use Case Diagram

StateChart Diagram

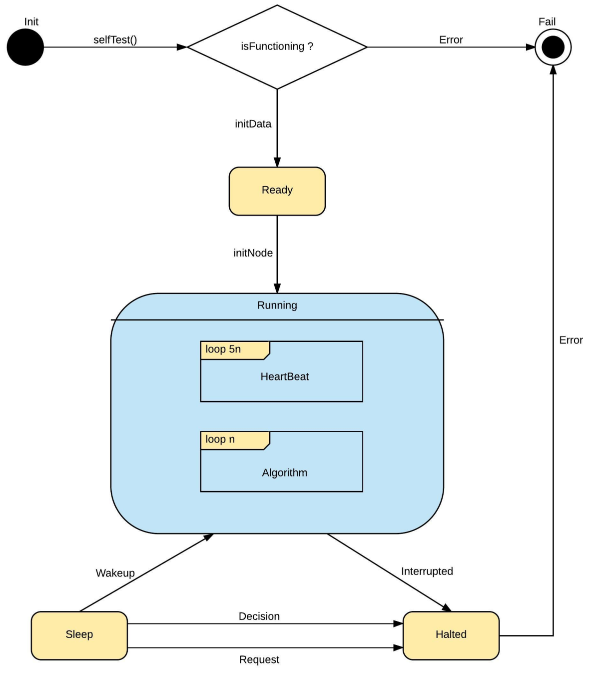

Figure 3.6: StateChart Diagram

Sequence Diagrams

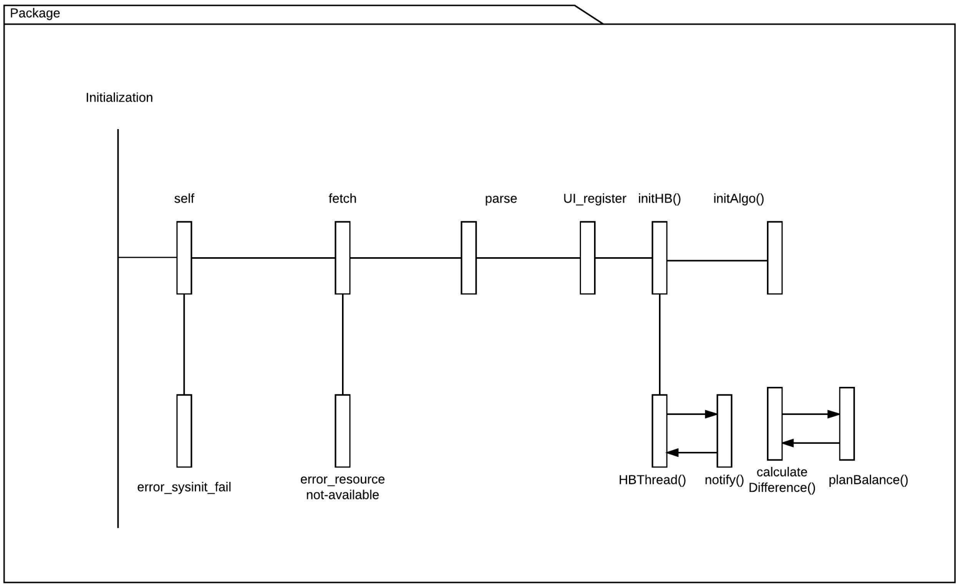

Figure 3.7: Initialization Sequence

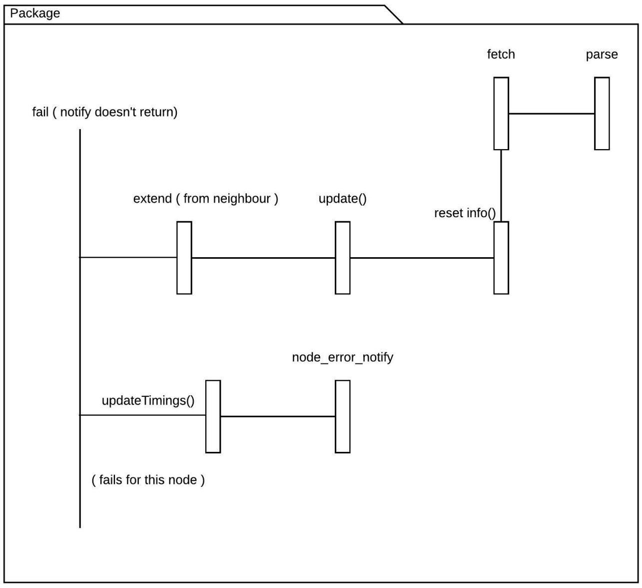

Figure 3.8: Sequence for Failure of a Node

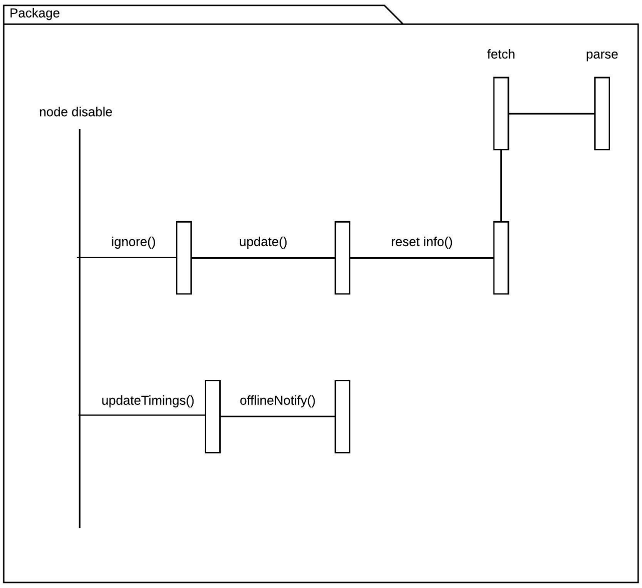

Figure 3.9: Node Removal Sequence

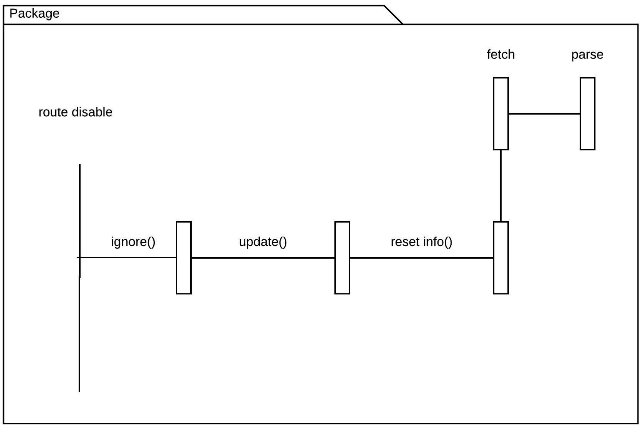

Figure 3.10: Route Modification Sequence

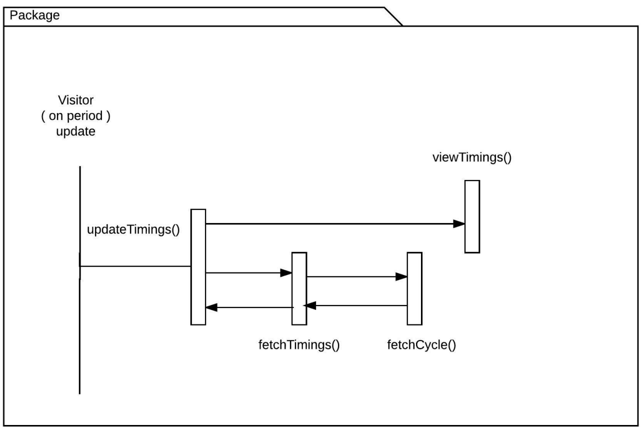

Figure 3.11: Visitor's Usage Sequence (uncached data)

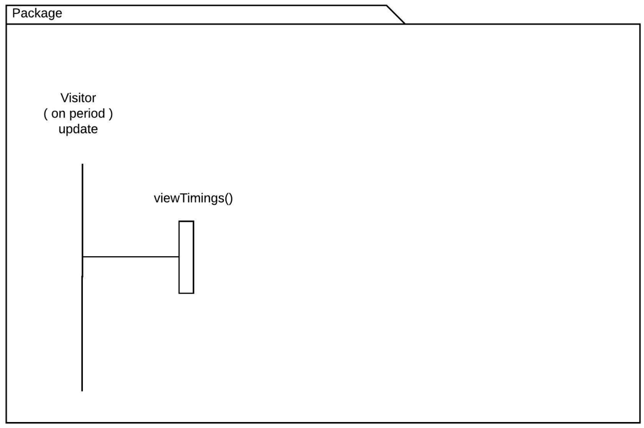

Figure 3.12: Visitor's Usage Sequence (cached data)

Activity Diagram

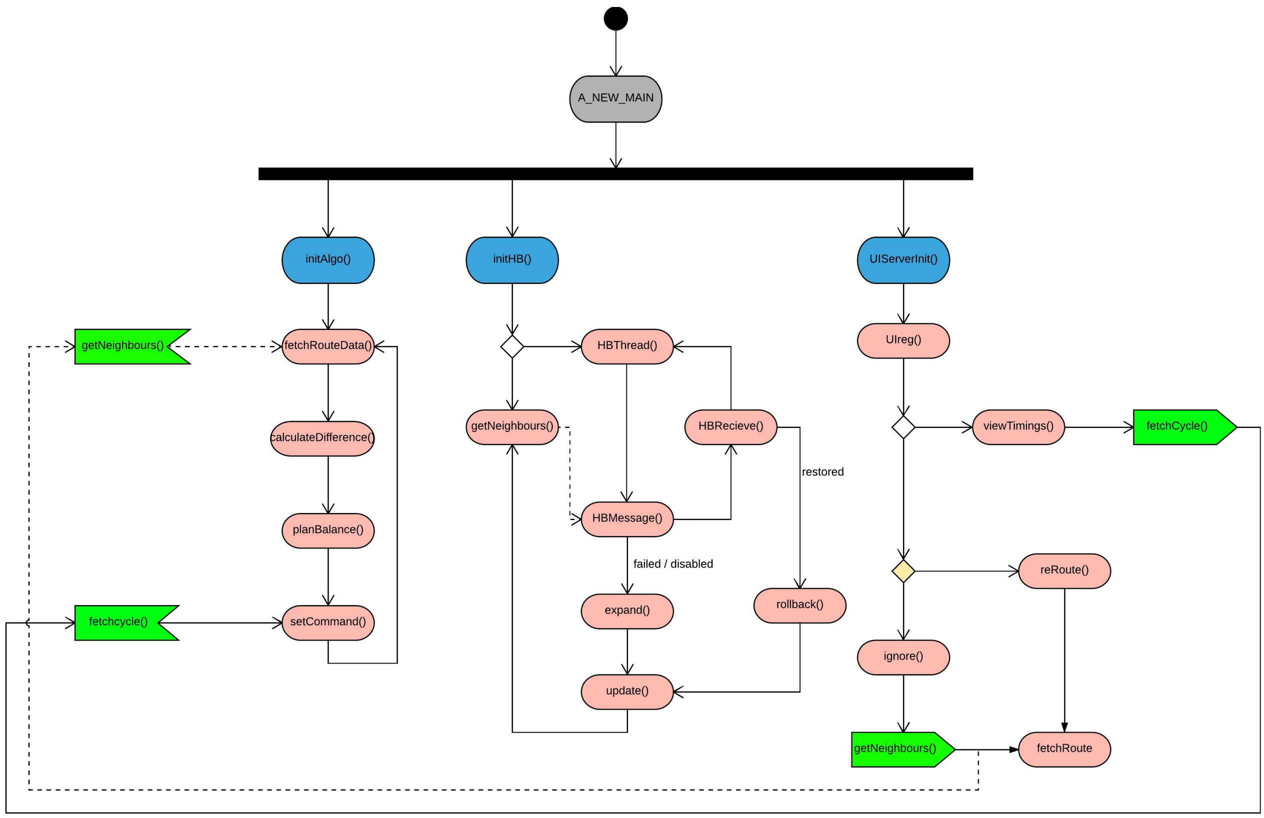

Figure 3.13: Activity Diagram

3.3.4 Entity Relationship Diagram

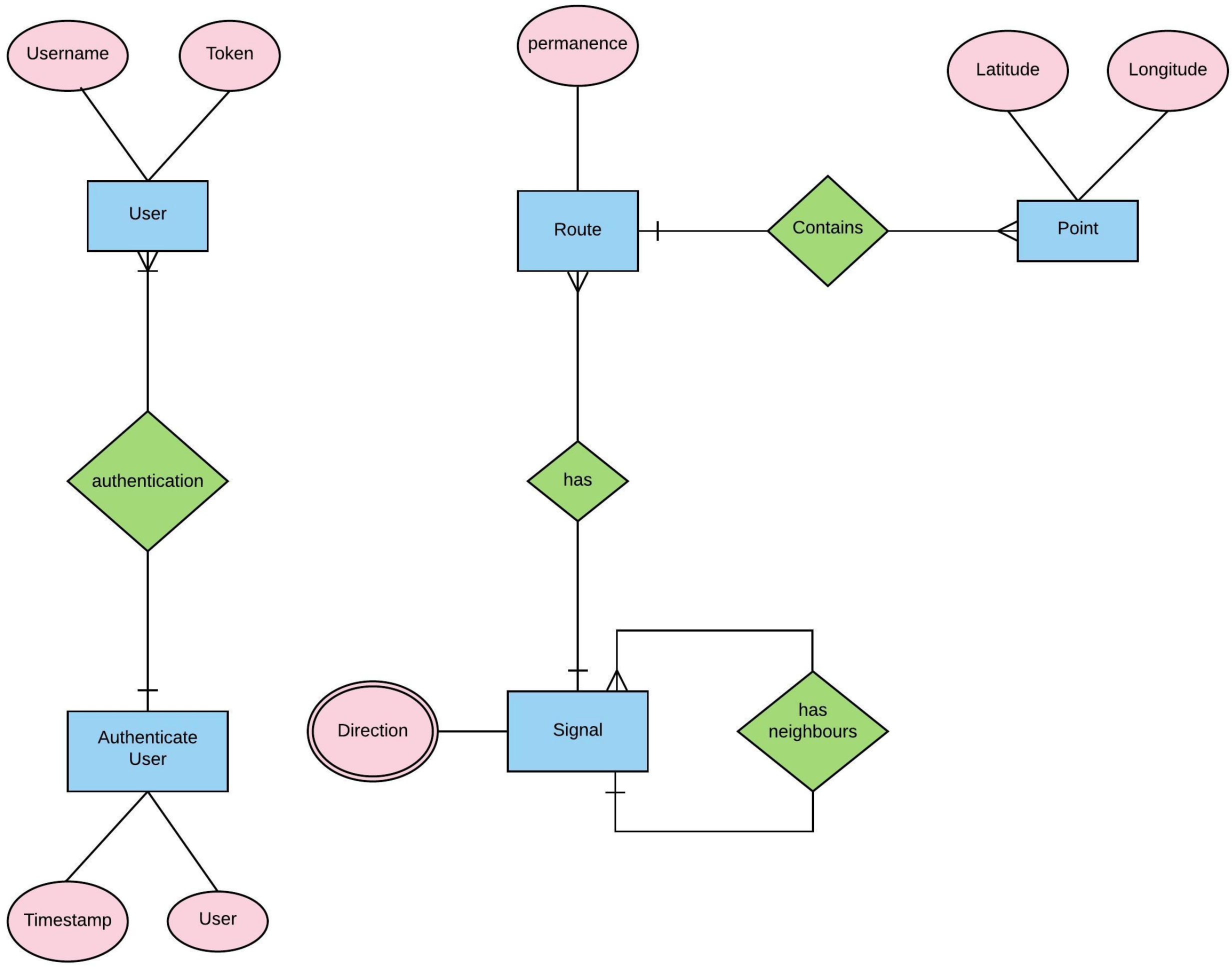

Figure 3.14: ER Diagram

3.4 Planning Phase :

3.4.1 Software and Hardware Requirements:

1. Software Requirements

   - Linux Operating System (Arch, Debian) for Internet access to Google's Traffic API.
   - A representation of roads as Google's *LatLong* Objects.
   - An interface to use/modify the Traffic Signal.

2. Hardware Requirements

   - Arduino UNO Rev.3 to control the signal.
   - A 3G or 4G enabled internet Dongle/WiFi NIC to use the Internet.

3. Front End Requirements:

   - JavaScript control to all Arduino Modules deployed in any route.

- JavaScript access to GTAPI to view route's traffic conditions.

Back End Requirements:

- JavaScript access to GTAPI to observe route section.
- JavaScript access to Redis Database to access known LatLong Objects for route sections.

3.4.2 Major milestones and deadlines:

1. Major Milestones:
    - Validation of GTAPI usage and reliability.
    - Developing Traffic Signal Controller in Python.
    - Setting up Internet Access on Arduino. (Demo the GTAPI access).
    - Developing the Traffic Management Algorithm.
    - Testing and Tuning of above Algorithm using MOTUS Simulator.
    - Using above Algorithm with Signal Controller.
2. Deadline for project completion: 18th March, 2018.

3.4.3 Structure of the database:

1. JSON documents(A) storing (Latitude,Longitude) pairs for points on/along a route section; accessible using Route Section IDs.
2. JSON documents(B) storing Route Section IDs and Controller IDs for each traffic signal; accessible using Controller IDs.
3. JSON Documents(C) storing Direction:Controller IDs next to a given Controller ID for each traffic signal; accessible using Controller IDs.
4. JSON Document(D) storing User's credentials (username and unique hashed token) for each User; accessible using username.
5. JSON Document(E) storing Authenticated user with their Timestamp (time of login) for each logged-in user; accessible using username.

3.5 Prototyping:

The diagrams below are an early(demonstrational) model of the project's outcome.

Figure 3.15: Hardware Prototype

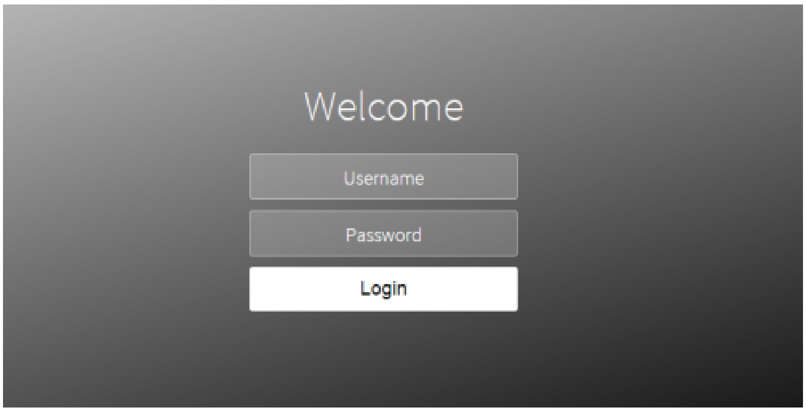

Figure 3.16: Login UI

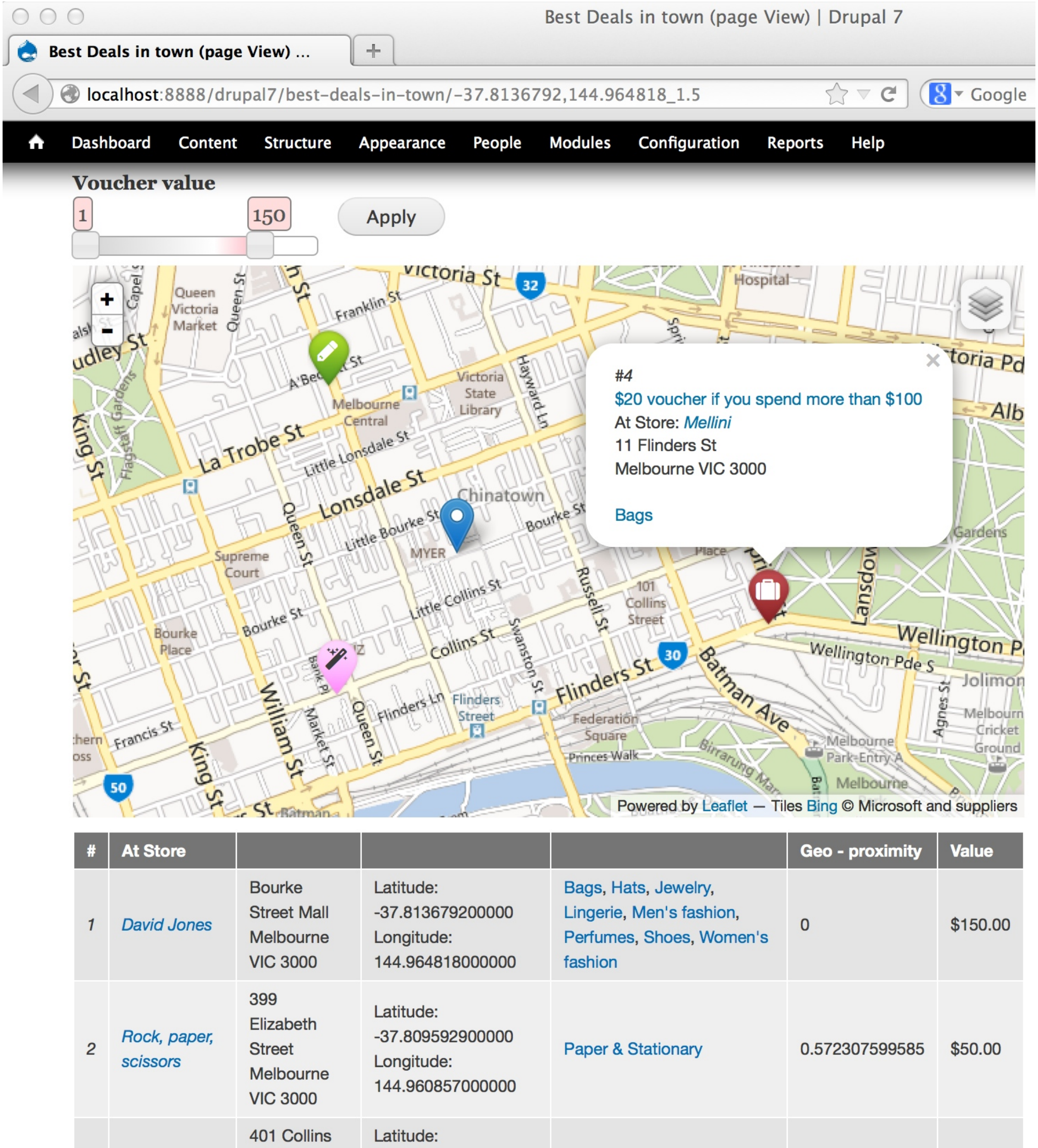

Figure 3.17: Visitor UI

All diagrams above are subject to changes in the future.

4.

Conclusion

We have implemented a stable solution to a long standing Traffic problem. Our system is able to automatically handle traffic without any Human intervention. It actively looks for signs of Traffic Jam buildup and prevents it. The system achieves this in a decentralized manner such that there is no single point of control/failure. Comparing to leading technologies established in this field, our approach aims to reduce costs while providing the same level of efficiency. One can also note that our system is deployable anywhere without any major modifications required to the existing setup.

5.

References

References

[1] Alain Kibangou, et. al., An Integrated and Scalable Platform for Proactive Event-Driven Traffic Management, *Arxiv.org*, 8th March 2017.

[2] Carlos Gershenso, Self-Organizing Traffic Lights, *Arxiv.org*, 5th Feb 2008.

[3] Moshe E. Ben-Akiva, et. al., A dynamic traffic assignment model for highly congested urban networks, *Elsevier*, 6th Feb 2012.

[4] Pu Wang, et. al., Understanding Road Usage Patterns in Urban Area, *SCIENTIFIC REPORTS*, vol 2, no. 1001, 20 December 2012.

[5] Nasrin Taherkhani and Samuel Pierre, Centralized and Localized Data Congestion Control Strategy for Vehicular Ad Hoc Networks Using a Machine Learning Clustering Algorithm, *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*, March 20, 2016.

6.

Annexure - A

Table 2 - IDEA Matrix

I	D	E	A
Innovative	Detect	Evaluate	Associate
Increase	Decrease	Effective	Acquire
Investment	Depict	Enhance	Analyse

Innovate	Traffic systems are either centralized or uncoordinated or need a lot of additions to the installation site. Here, the system uses only one addition which can be a replacement to existing signal control hardware and is decentralized which is **innovative**.
Increase	As the system uses real-time data from GTAPI (which is cached), the efficiency of the system will **increase**.
Investment	As the system uses commodity hardware, **investment is minimal** and installation is easy.

Detect	The system will **detect** the signs of traffic buildup and makes changes to the signal timings dynamically.
Decrease	The algorithm will **decrease** the buildup by increasing green time along the route and increasing overall cycle timings for heavily loaded signals.
Depict	The system UI **depicts** the timings and status of each traffic signal in real-time.

Evaluate	The algorithm **evaluates** the densities of traffic in each lane to decide timings
Effective	The densities are obtained in real-time allowing for steady and tiny changes in timings to be highly **effective**.
Enhance	The system can be **enchanced** by improving decisions about changes and their amount.

Associate	The project **associates** incremental approach to large distributed networks to allow optimal solutions to emerge.
Acquire	The density data of each lane is **acquired** from GTAPI.
Analyse	The system **analyzes** current density information and previous cycle's densities to produce current timings.

7.

Annexure - B

Table 3 - Test Cases

Test ID	Test Case Description	Input	Expected Output
1	User enters a Correct Username & Correct Password	Valid Username & Valid Password	Access Granted
2	User enters an Incorrect Username & Correct Password	Invalid Username & Valid Password	Access Denied
3	User enters a Correct Username & Incorrect Password	Valid Username & Invalid Password	Access Denied
4	User enters an Incorrect Username & Incorrect Password	Invalid Username & Invalid Password	Access Denied
5	User wants to modify routes	Valid changes	Changes saved in UI Buffer
6	User wants to modify routes	Invalid changes	Changes saved in UI Buffer
7	User wants to save modified routes	Valid changes	Changes saved in Redis Database
8	User wants to save modified routes	Invalid Changes	Changes not saved & are discarded
9	User wants to modify Node status	Switch ON	Node switched ON
10	User wants to modify Node status	Switch OFF	Node switched OFF
11	User wants to modify Node status	Switch ON then OFF	No Change in Node status
12	User wants to modify Node status	Switch OFF then ON	No Change in Node status
13	User wants to modify Node status	Combination of 0/1 cases of type 11 and m cases of 12	Output of case type 11

8.

Annexure - C

Table 4 - Progress Report

Sr. No.	Day and Date	Topics Discussed	Suggestion of Project Guide
1	Tuesday, 22nd August 2017	Base Paper, Topic, Tools Decided.	
2	Monday, 18th September 2017	Seeking Algorithm class/style.	
3	Thursday, 28th September 2017	Partial Presentation made.	
4	Wednesday, 11th October 2017	Synopsis made.	
5	Tuesday, 24th November 2017	Partial UML Diagrams completed (Structural).	
6	Monday, 11th December 2017	SRS Textual Content completed Partial Behavioural Diagrams completed (Use case, Statechart, Sequence).	
7	Wednesday, 13th December 2017	Completed UML Diagrams (Activity) and ER Diagram. Partial Project Report completed.	
8	February 2018 End	Data Collection Completed	
9	Mid January 2018	Algorithm Design Completed	
10	January 2018 End	Hardware Design and Testing Completed	
11	February 2018 End	UI Design and Testing Completed	
12	Mid March 2018	Algorithm Testing using Data Collected	