IoT Based Real Time Data Collection for Dynamic Road Weight Measurement

Md Abdullah-Al-Forhad

ID: 2013–1–60–029

Md Nadim

ID: 2013–1–60–067

A thesis submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering

Department of Computer Science and Engineering
East West University
Dhaka-1212, Bangladesh

April, 2017

Declaration

We, hereby, declare that the work presented in this thesis is the outcome of the investigation performed by us under the supervision of Dr. Md. Shamim Akhter, Assistant Professor, Department of Computer Science and engineering, East West University. We also declare that no part of this thesis has been or is being submitted elsewhere for the award of any degree or diploma.

Signature

. . . .

(Md. Abdullah-Al-Forhad)

(ID: 2013-1-60-029)

Signature

. . . .

(Md. Nadim)

(ID: 2013-1-60-067)

Letter of Acceptance

This Thesis Project entitled *"IoT Based Real Time Data Collection for Dynamic Road Weight Measurement"* submitted by Md. Abdullah-Al-Forhad (ID: 2013-1-60-029) and Md. Nadim (ID: 2013-1-60-067) to the Department of Computer Science and Engineering, East West University, Dhaka, Bangladesh is accepted by the department in partial fulfillment of requirements for the Award of the Degree of Bachelor of Science and Engineering on April, 2017.

Supervisor

....

(**Dr. Md. Shamim Akhter**)

Assistant Professor, Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh

Chairperson

....

(**Dr. Ahmed Wasif Reza**)

Chairperson (Acting) and Associate Professor

Department of Computer Science and Engineering

East West University, Dhaka, Bangladesh

Abstract

Traffic congestion is an inevitable problem for metropolitan cities around the globe. A new and dynamic internet based decision making tool for traffic management system (TMS) was proposed and implemented in our previous work. The tool uses weather, road and vehicle related information and generates the routing decision. Thus, real-time data collection is important. There are many web portals hosting real time continuous weather data streams. However, their coverage is limited to city level congregate information and location/area based information is necessary for real time TMS decision. In addition, there is not a single web site that hosts real-time road and vehicle information. Internet of Things (IoT) can be a solution for this circumstance. So, an IoT based framework is proposed and implemented with several remote agents. Agents are securely (session management) interconnected with a cloud server and able to collect or exchange TMS data among themselves through GSM/GPRS/Bluetooth communication channels. Each agent is an embedded device with integrated sensors (including DHT11, Raindrops Sensor Module), Arduino microcontroller and GSM/GPRS/GPS Bluetooth enable modules.

Acknowledgments

First of all, We would like to thank almighty Allah for giving us strength, patience and knowledge to complete this thesis work.

We would like to express our sincere gratitude to our supervisor Dr. Md. Shamim Akhter for the continuous support, for his patience, motivation, and immense knowledge. His guidance helped us in all the time of work and writing of this thesis. We could not have imagined having a better supervisor and mentor for this thesis.

We would like to thank our friends for their feedback, cooperation and of course friendship.

In addition, We would like to express our gratitude to the faculty members of the Department of Computer Science and Engineering (CSE), East West University, Bangladesh who helped us with their feedback.

Last but not least, We would like to thank our family: our parents and to our brothers and sisters for supporting us.

Md Abdullah-Al-Forhad
April, 2017

Md Nadim
April, 2017

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Life in the metropolitan cities involves fast movement from one place to another. Faster movement requires faster transportation support and thus number of vehicles are increasing day by day and putting continuous pressure on the traffic flow of the cities. Traffic flow should be managed in order to keep pace with the development of those cities.

We have already proposed and implemented an internet based real time bi-directional traffic management support system [1][2][3][4][5]. Decision making of that system requires various information including road and vehicle related information, and several environmental attributes. Some of that information can be collected from various open data websites. However, their coverage is limited to city level. In addition, location/area based congregate information at an instant is necessary for accurate TMS decision making system. Moreover, vehicle related information (speed) collection requires building embedded devices. Devices need to communicate among themselves and transfer the acquired information to the TMS system. Subsequently, TMS decision making system will make routing related decision from those data. Internet of Things (IoT) is a network where remote agents are interconnected with each other and can be a solution for this circumstance. In this research work, we are going to propose an IoT based framework and implement it with several remote agents with the core fundamentals of IoT.

There are many solution models for IoT based TMS. The solutions are varying in respect to the technologies they used. Radio Frequency Identification (RFID) based IoT is best suited for smaller spaces, where the infrastructure is already in place to use it.

RFID requires specialized scanners to read and transmit data, and specific receivers [6]. Thus, RFID is avoided from our proposed system.

Our designed agents are securely (session management) interconnected with an authenticated cloud server and able to collect or exchange encrypted TMS data among themselves through GSM/GPRS/Bluetooth communication channels. Base64 encoding is used to encode and decode data. The proposed agent is mainly embedded device based on Arduino open source microcontroller [7]. Temperature sensor, rainfall sensor module and GSM/GPRS/GPS/Bluetooth module are used together to build the device embedded. In the figure 1.1 we can see the Process flow of Traffic Management System (TMS).

1.1 Motivation

There are many web portals hosting real time continuous weather data streams. However, their coverage is limited to city level congregate information and location or area based information is necessary for real time Traffic Management System (TMS) decision. In addition, there is not a single web site that hosts real-time road and vehicle information. Internet of Things (IoT) can be a solution for this circumstance.

1.2 Research Objectives

The main objective of this research is the implementation of a secure IoT based framework with several remote agents to collect real time data for dynamic road weight measurement of traffic management system. The other three sub objectives are,

- Design and implement an embedded device for real time data collection
- Design and implement a secure IoT infrastructure
- Cloud based system design and implementation for data storage

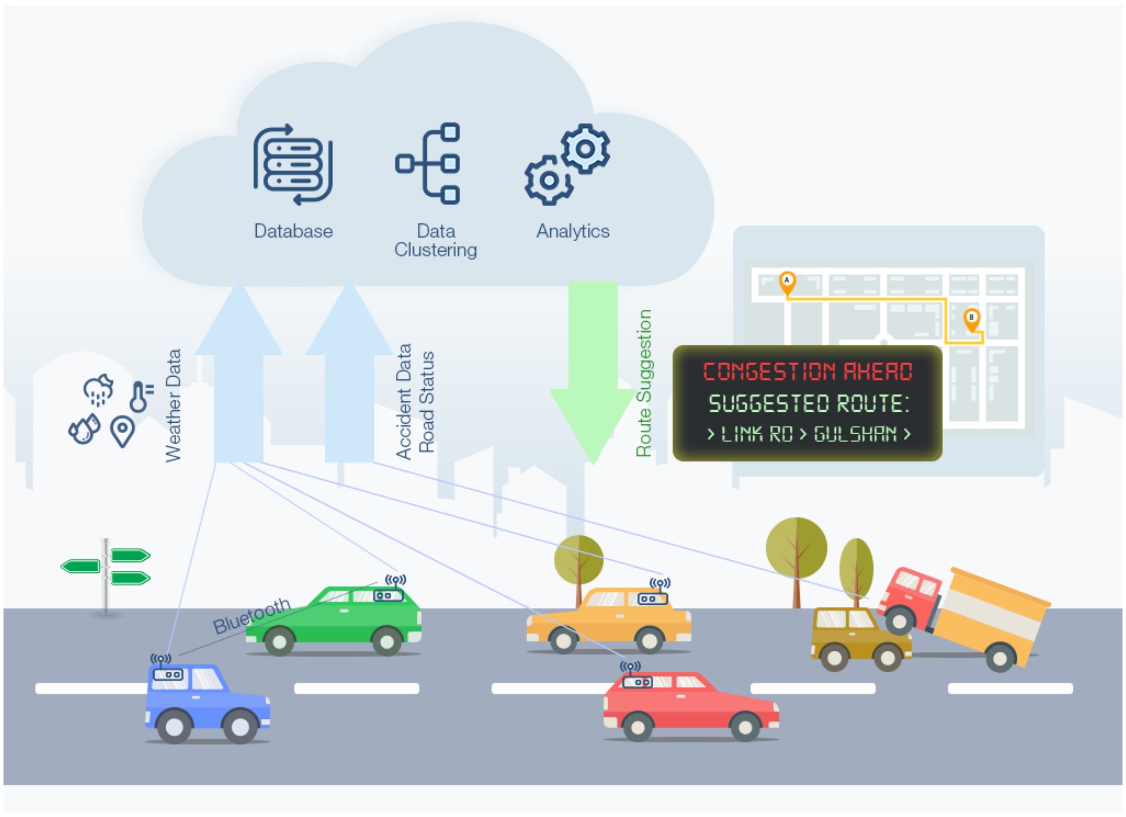


Figure 1.1: Process flow of TMS

1.3 Scope and Limitation

The devices can communicate with cloud server using GSM/GPRS based HTTP protocol. In addition, they can communicate with each other and also with mobile phone via a mobile application without internet but Bluetooth communication. However, the device has limitation not to communicate with server if both Bluetooth range and GSM coverage are not available.

1.4 Outlines of Thesis

The rest of this thesis is organized as follows: chapter 2 gives the background study of the work. Chapter 3 is about related work of this thesis. Chapter 4 demonstrates implementations of IoT based real time data collection model. Finally, chapter 5 concludes the work and shows the outline of our future plans.

Chapter 2
Background Study

2.1 Cloud Computing

In the past few years, cloud computing has transformed the IT landscape for both individuals and enterprises-from the way we access, store, and share information to how we communicate, collaborate, and process data. Cloud computing often referred as "The Cloud". It simply means delivery of computer services, i.e: servers, storage, databases, networking, software, analytics and more, over the internet [8]. Cloud computing restructured the way of thinking about hardware resources because with cloud computing we can use computer services without thinking about the hardware resources. There are many reasons to consider cloud computing over general server based computing, here are a few of them:

1. Cost efficiency
2. High Speed
3. Global Scale
4. Productivity
5. Performance
6. Reliability
7. Security

2.1.1 Service Models of cloud computing

Cloud computing providers provide their services according to different models. There are three standard model defined by NIST (National Institute of Standards and Technology) [9]. These are sometimes called the cloud computing stack, because they are built on top of one another. Those models are,

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

The figure 2.1 demonstrates the service ranges of different cloud computing models [10].

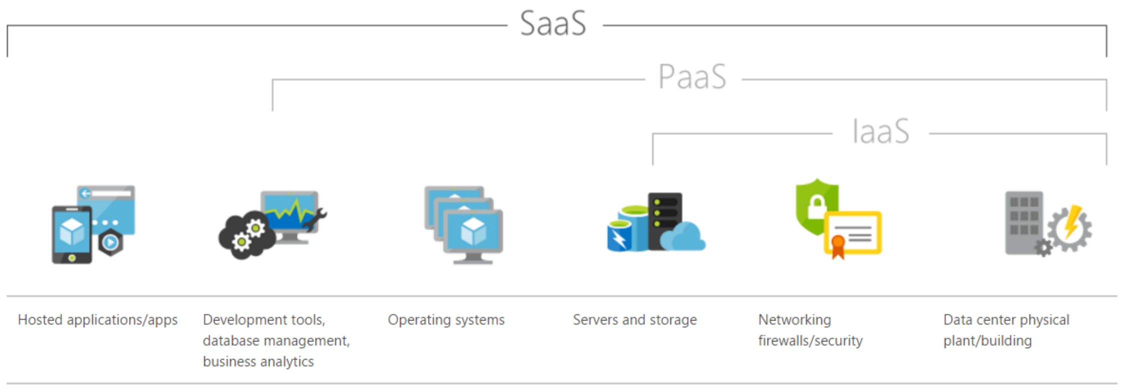

SaaS
PaaS
IaaS
Hosted applications/apps
Development tools, database management, business analytics
Operating systems
Servers and storage
Networking firewalls/security
Data center physical plant/building

Figure 2.1: Service Models of Cloud Computing.

2.1.1.1 Software as a Service (SaaS)

Software-as-a-service (SaaS) is a method for delivering software applications over the Internet. It allows users to connect to and use cloud-based apps over the Internet.

Common examples are email, calendaring, and office tools. In most cases SaaS is end user applications. With a SaaS offering user do not have to think about how the service is maintained. User only need to think about how to use that particular piece of software. Advantages of SaaS are:- [11]

- **Gain access to sophisticated applications.** To provide SaaS apps to users, you don't need to purchase, install, update, or maintain any hardware, middleware, or software. It even provides sopchisticated enterprise level applications which could be very costly for a new business to buy and maintain.
- **Pay only for what you use.** You also save money because the SaaS service automatically scales up and down according to the level of usage.
- **Use free client software.** Users can run most SaaS apps directly from their web browser without needing to download and install any software.
- **Mobilize your workforce easily.** SaaS makes it easy to "mobilize" your workforce because users can access SaaS apps and data from any Internet-connected computer or mobile device. You don't need to worry about developing apps to run on different types of computers and devices because the service provider has already done so.
- **Access app data from anywhere.**With data stored in the cloud, users can access their information from any Internet-connected computer or mobile device. As the data is stored in the cloud, there is no chance of losing the data even the system is failed.

2.1.1.2 Platform as a Service (PaaS)

Platform as a service (PaaS) is a complete development and deployment environment in the cloud, with resources that enable you to deliver everything from simple cloud-based apps to sophisticated, cloud-enabled enterprise applications. PaaS includes infrastructure-servers, storage, and networking-but also middleware, development tools, database management systems, and more. Advantages of PaaS are:- [12]

- **Cut coding time.** With pre-coded applications built into the platform, coding new apps takes less time with applications such as workflow, directory services, security features, search, and so on.
- **Pay only for what you use.** You also save money because the SaaS service automatically scales up and down according to the level of usage.
- **Add development capabilities without adding staff.** Platform as a Service components can give your development team new capabilities without your needing to add staff having the required skills.
- **Develop for multiple platforms.** Some service providers give you development options for multiple platforms, such as computers, mobile devices, and browsers making cross-platform apps quicker and easier to develop.
- **Use sophisticated tools affordably.** A pay-as-you-go model makes it possible for individuals or organizations to use sophisticated development software and business intelligence and analytics tools that they could not afford to purchase outright.
- **Support geographically distributed development teams.** Because the development environment is accessed over the Internet, development teams can work together on projects even when team members are in remote locations.
- **Efficiently manage the application lifecycle.** PaaS provides all of the capabilities that you need to support the complete web application lifecycle: building, testing, deploying, managing, and updating within the same integrated environment.

2.1.1.3 Infrastructure as a Service (IaaS)

Infrastructure as a service (IaaS) is an instant computing infrastructure, provisioned and managed over the Internet. Infrastructure as a service is the most basic service model of cloud computing. It provides companies with computing resources including servers, networking, storage, and data center space. Advantages of IaaS are:- [10]

- **Eliminates capital expense and reduces ongoing cost.** For start-ups, settings up and managing on-site datacenter can be costly. IaaS reduces that upfront expense, making it an economical option for start-ups and businesses testing new ideas.
- **Improves business continuity.** Achieving high availability, business continuity, and disaster recovery is expensive, since it requires a significant amount of technology and staff. But with IaaS, it can reduce this cost and access applications and data as usual during a disaster or outage.
- **Innovate rapidly.** As soon as you've decided to launch a new product or initiative, the necessary computing infrastructure can be ready in minutes or hours, rather than the days or weeks-and sometimes months-it could take to set up internally.
- **Respond quicker.** IaaS enables you to quickly scale up resources to accommodate extra needs in demand for your application- during the holidays, for example-then scale resources back down again when activity decreases to save money.
- **Focus on your core business.** IaaS frees up your team to focus on your organization's core business rather than on IT infrastructure.
- **Increase stability, reliability, and supportability.** With IaaS there's no need to maintain and upgrade software and hardware or troubleshoot equipment problems. With the appropriate agreement in place, the service provider assures that your infrastructure is reliable and meets SLAs.

- **Better security.** A cloud service provider can provide security for your applications and data that may be better than what you can attain in-house.

2.1.2 Deployment models

There are four ways to deploy cloud computing resources, [9]

- Public Cloud

The most recognizable model of cloud computing to many consumers is the public cloud model, under which cloud services are provided in a virtualized environment, constructed using pooled shared physical resources, and accessible over a public network such as the internet. Generally Public clouds are owned by cloud service providers. User access those service by their web browser and also manages their accounts. The most salient examples of cloud computing tend to fall into the public cloud model because they are, by definition, publicly available. Software as a Service (SaaS) offerings such as cloud storage and online office applications are perhaps the most familiar.

- Private Cloud

In Private cloud resources are exclusively used by single organization. In this cloud services and infrastructure are maintained on a private network. It is a particular model of cloud computing that involves a distinct and secure cloud based environment in which only the specified client can operate. As with other cloud models, private clouds will provide computing power as a service within a virtualized environment using an underlying pool of physical computing resource. However, under the private cloud model, the cloud is only accessible by a single organization providing that organization with greater control and privacy.

- Hybrid Cloud

A hybrid cloud is an integrated cloud service utilizing both private and public clouds to perform distinct functions within the same organization. Hybrid cloud gives better flexibility as it allows data to be shared with private and public cloud. Therefore, an organization can maximize their efficiencies by employing public cloud services for all non-sensitive operations, only relying on a private cloud where they require it and ensuring that all of their platforms are seamlessly integrated.

- Community cloud

The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations. Those community have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party.

2.2 IoT Architecture

The Internet of Things refers to the set of wireless/wired devices and systems that interconnect real sensors and actuators to the Internet. The term "Internet of Things" (IoT) denotes a trend where a large number of embedded devices employ communication services offered by the Internet protocols. Many of these devices, often called "smart objects", are not directly operated by humans, but exist as components in buildings or vehicles, or are spread out in the environment [13]. It is a broad term, which indicates the concept that increasingly pervasive connected devices (embedded within, attached to or related to "Things") will support various applications to enhance the awareness and the capabilities of users. The possibility of implementing "intelligence" in these pervasive systems and applications has also suggested the definition of "Smart" contexts, where digital and real-word objects cooperate in a cognitive and autonomic way to fulfil specific goals in a more efficient way than basic systems implemented on static rules and logic.

At the end of the day, IoT intends to synchronize the physical world with the virtual world by utilizing the Internet as the middleware to exchange data.

There are some particular requirements for IoT platforms to earn that title. For example, many requirements rise up out of the power available to IoT devices. Other requirements come from the path in which IoT devices are manufactured and used; the methodologies are more like conventional product design plan for consumers and also existing Internet approaches. Obviously, there are various existing prescribed procedures for the server-side and Internet availability that need to be considered in. We can summarize the general requirements into some key requirements: [14]

- Require Diverse Connectivity

The connectivity objective is that an IoT platform support as many modes of connection-wired and wireless-as possible. Wireless options include ANT+, Bluetooth, EDGE, GPRS, IrDA, LTE, NFC, RFID, Weightless, WLAN, ZigBee, and Z-Wave.

- Leverage Applications

IoT software applications provide much of the automation capabilities that make IoT solutions so valuable. These software and middleware applications help businesses drive down costs, increase efficiency, and improve regulatory compliance.

- Manage a Range of Devices

IoT sensors gather information about conditions in their vicinity, such as temperature or moisture level. IoT actuators perform specific tasks, such as turning things on or off, and recording information about its triggers and subsequent reactions.

- Generate Massive Amounts of Data

Devices that we discussed above don't just perform tasks. In most cases, they will also report on the tasks they perform. Through their connection to an IoT platform and to each other, they will transmit detailed data about their actions. Typically, there will be no need for human intervention in the process. The devices will simply send data, potentially in real-time, for storage and analysis. Functionally, therefore, an IoT platform must be able to support storing massive amounts of data.

- Require Powerful Analytics

The vast volumes of data discussed above have the potential to provide unprecedented insights into consumer behavior and preferences. Unlocking those insights, however, requires powerful analytics tools. A key IoT platform functionality, analytics solutions that will translate significant amounts data into useful and actionable insights.

- IoT Platforms and Security

Even with the recent attention given to security for IoT devices, it can be easy to overlook the need for end-to-end security for an IoT platform. Every part of a platform should be analyzed for security prospects. From internet connections to the applications and devices to the transmitted and stored data, there is a potential for an attack vector. Without question, the single most important non-functional requirement of an IoT platform is that it offers robust security.

2.2.1 A Framework Model

While choosing which approach to take for your IoT project a reference model may come in handy [13]. The table 2.1 contains explanation of each level and the figure 2.2 illustrates the IoT Reference Model.

2.3 Traffic Management System

Traffic jam is an on-going challenge, and is not showing any signs of improvement. It results losing valuable time (approx. 200%) being stuck in traffic jams and increases CO2

Level No	Level Name	Description
7	Collaboration and Processes	Involving people and business processes: providing real-time insights, decision support and collaboration options to other areas in the business.
6	Application	Reporting, analytics, control: diagnostic, predictive and prescriptive analytics (preferably autonomous) to control your things. Reporting: accounting, usage, state on your things and application (service).
5	Data Abstraction	Aggregation and access: aggregation of data to reports (required usage) and access of data (interface) for applications.
4	Data Accumulation	Storage: event persistence, filtering, sampling events, event based rules, event aggregation and complex event processing.
3	Edge (Fog) Computing	Data element analyses and transformation: filtering, cleaning and aggregating data, generating events and alarms.
2	Connectivity	Communication and processing units: providing security, reliable networking, protocol translation, switching and routing.
1	Physical Devices and Controllers	The Things in IoT: generating data (analogue/digital) and events, queried and controlled over the net.

Table 2.1: IoT Reference Model

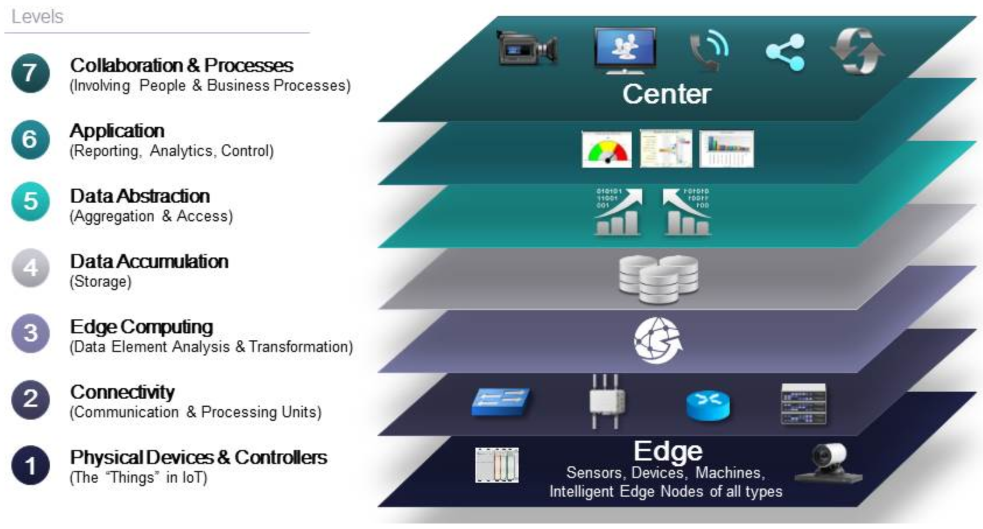

Levels
7 Collaboration & Processes
(Involving People & Business Processes)
6 Application
(Reporting, Analytics, Control)
5 Data Abstraction
(Aggregation & Access)
4 Data Accumulation
(Storage)
3 Edge Computing
(Data Element Analysis & Transformation)
2 Connectivity
(Communication & Processing Units)
1 Physical Devices & Controllers
(The "Things" in IoT)
Center
Edge
Sensors, Devices, Machines,
Intelligent Edge Nodes of all types

Figure 2.2: IoT Reference Model

emissions (approx. 300%) due to the present traffic situation. As a result, an up-to-date technological traffic management support system/tool has become a necessity for metro cities.

A new low cost, flexible, maintainable, and secure internet based traffic management system with real time bi-directional communication was proposed and implemented (in [1][2][3][4]) to assist and reduce the traffic situation. To determine the dynamic road weights in TMS, four (4) different environmental attributes - rain fall, temperature, wind, and humidity are considered. Rainfall is one of the most influential weather attributes to determine the road congestion in metro city, as the road segments are submerged due to the heavy rains, and makes slower traffic movements. The heat released from the engines, air-conditioners of the traffic stacked vehicles, may raise the overall temperature of the area. Thus, the current temperature helps to classify traffic congestion status of a particular road segment. Decision tree based logic is implemented over the simplified data to adjust the route weights in database. The figure 2.3 contains model of Traffic Management System.

2.4 Components

It is very obvious that the main concern behind building any device is making it cost efficient, reliable and user friendly. So, our main choice was open source Arduino microcontroller board. Temperature sensor, rainfall sensor module and GSM/GPRS/GPS/Bluetooth module are used together to build the device.

- Arduino UNO R3

Arduino is very popular and easy to use programmable board for creating project as it is an open source microcontroller based development board which develops an environment for writing software for the board. The table 2.2 contains Technical specification of Arduino Uno R3 [15].

- GSM/GPRS/GPS/Bluetooth SIM808 Shield

It is based on the latest SIM808 from SIMCOM, supports GSM/GPRS Quad-Band network and combines GPS technology for satellite navigation and also Bluetooth communication. The table 2.3 contains Technical specification of GSM/GPRS/GPS/Bluetooth SIM808 Shield [16].

- DHT11

DHT is usually a good option if we want to monitor indoor or outdoor temperature. The DHT sensor has two versions: DHT11 and DHT22. Both sensors are very good to measure the temperature and humidity, but the characteristics are different. In comparison with DHT11, the DHT22 is good to measure the temperature from -40 to 125C and has a higher accuracy than DHT11. Although, DHT11 cannot read a large range of

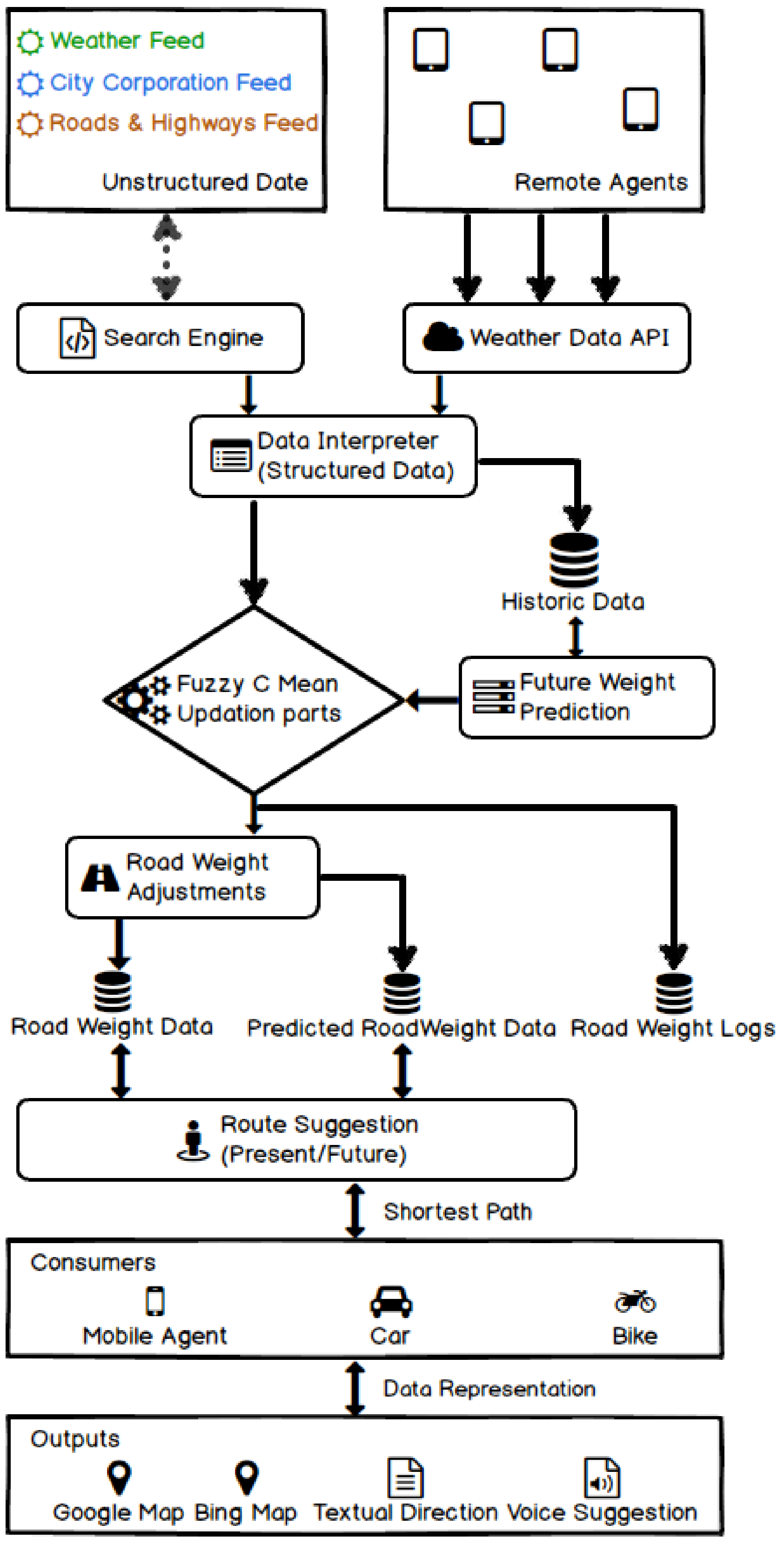


Figure 2.3: Traffic Management System Model

Figure 2.4: Arduino UNO R3

Figure 2.5: GSM/GPRS/GPS/Bluetooth SIM808 Shield

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Table 2.2: Technical Specification of Arduino Uno R3

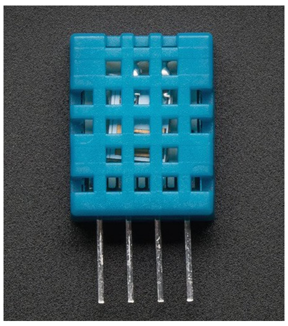


Figure 2.6: DHT11 Sensor

Quad-band	850/900/1800/1900MHz
GPRS multi-slot class12 connectivity	max. 85.6kbps(down-load/up-load)
GPRS mobile station	class B
Controlled by	SIMCOM enhanced AT Commands
Charging control	for Li-Ion battery
Supply voltage range	5V 28V
GPS	Integrated GPS/CNSS ,A-GPS,GPS NMEA protocol
Bluetooth Radio	ISM band from 2400-2480 MHz
Bluetooth Version	BT3.0
Logic voltage level	3.0V to 5.0V logic level
Power consumption	Low power consumption, 1mA in sleep mode
SIM	Standard SIM Card
Operation temperature	-40C 85C

Table 2.3: Technical Specification of GSM/GPRS/GPS/Bluetooth SIM808 Shield

temperature, it is smaller and less expensive than DHT22. We also dont need that extra range in the real world general scenario. The table 2.4 contains Technical specification of DHT11 [17].

Range	0 to 50 C
Accuracy	2 C
Voltage - Supply	3 V 30 V
Output type	Digital

Table 2.4: Technical Specification of DHT11

- Raindrops Sensor Module

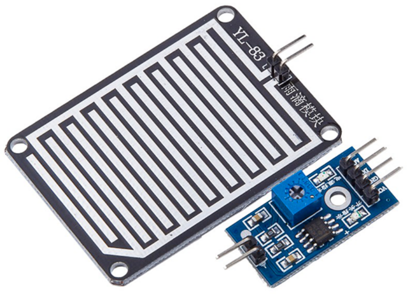


Figure 2.7: Raindrops Sensor Module

This module allows you measure moisture via analog output pins and it provides a digital output when a threshold of moisture is exceeded. The module is based on the LM393 op amp. It includes the electronics module and a printed circuit board that "collects" the rain drops. As rain drops are collected on the circuit board, they create paths of parallel resistance that are measured via the op amp.

The lower the resistance (or the more water), the lower the voltage output. Conversely, the less water, the greater the output voltage on the analog pin. A completely dry board for example will cause the module to output five volts [18].

Chapter 3
Related Works

In many research IoT has been used to solve road traffic problem. Xi Yu, Fuquan Sun and Xu Cheng [19] used IoT and cloud computing in *"Intelligent Urban Traffic Management System Based on Cloud Computing and Internet of Things"*. Mr. P.Sivasankar and B.Brindhavathy also used IoT on their work *"IOT Based Traffic Monitoring using Raspberry Pi"* [20]. Mahesh Lakshminarasimhan used IoT on his research *"IoT Based Traffic Management System"* [21]. J.Sherly and D.Somasundareswari also worked on related topic with title *"Internet of Things Based Smart Transportation Systems"* [22]. Most of the works stated above are based on RFID technologies. Radio Frequency Identification (RFID) is a pronunciation technology for identifying, tracing and counting real-life objects and in the research of IoT. The main objective of RFID is location tracking which means widelife monitoring and tracking, traffic movement control and parking management. With the help of RFID system if any car is stolen it can identify the location of car [23]. This technology is best suited for smaller spaces, where the infrastructure is already in place to use it [6]. RFID requires specialized scanners to read and transmit data, and without one specific to the proprietary receivers, there's no point. The dedicated infrastructure may be of great cost on a large scale, but on a small, localized scale, it will be incredibly powerful for both tracking and for providing information [24].

GPS is a very different beast from RFID. While it also uses radio waves to transmit data, it does so using, well, the global positioning system of 24 satellites, as opposed to specialized scanners here on the ground. Radio waves sent out from this system of satellites transmit their time and orbital data to receivers down on Earth [24]. Using GPS is better to get the accurate location and navigation data, thus it is easier to find the user's location from anywhere and suggests route accordingly. For RFID, scanners need to be set up almost in every route to get accurate result, but with our built in GPS system in the device it is easier to locate and collect data from any location and process it.

In the work *"Bi-Directional Traffic Management with Multiple Data Feeds for Dynamic Route Computation and Prediction System"* done by Md. Rahatur Rahman and Shamim Akhter [2], they used weather data to analyze road weight. Instead of using weather data from general sources, we are using our own sensors placed in vehicles to get the data. The advantage of this is, general weather data will show a broader result of the city/area level domain. So, the margin of difference among different locations will be very small. But our system will collect data directly from road along with other valuable info such as velocity. This will give a more accurate perception of each area and its affects.

Chapter 4

Implementations

4.1 Embedded Device

4.1.1 Circuit Connection

GSM/GPRS/GPS/Bluetooth SIM808 Shield is Arduino compatible so it sits on Arduino and directly connects to all pins of Arduino. The figure 4.1 contains Schematic Diagram of the Embedded Device.

To achieve serial port mode of GSM/GPRS/GPS/Bluetooth SIM808 Shield to communicate with Arduino UNO then jumper cap must connect 900R and 900T to U_R and U_T respectively in Serial Port Mode Selection Interface(P1). For serial communication we used port number RX=10 and Port number TX=11.

To power on the GSM/GPRS/GPS/Bluetooth SIM808 Shield automatically we, used a jumper cap to connect the "GND pin" of the control interface (J2) with "PWK Pin".

DTH11 sensor has 4 pins and we need three of them, those are VCC, Signal, GND and other one is not used. The DHT11 uses Signal pin to transfer sensor data digitally to Arduino. Signal pin of DHT11 is connected to Arduino via pin number 2. VCC and GND pins of DHT11 are connected to 5v pin and GND pin respectively. There is a 10K Ohm pull-up resistor connected from the signal line to 5V to make sure the signal level stays high by default.

Raindrop sensor module consists of 4 pins and 2 loop pins. Those 4 pins are A0(Analog Output), D0(Digital Output), GND and VCC. VCC and GND pins of Raindrop sensor

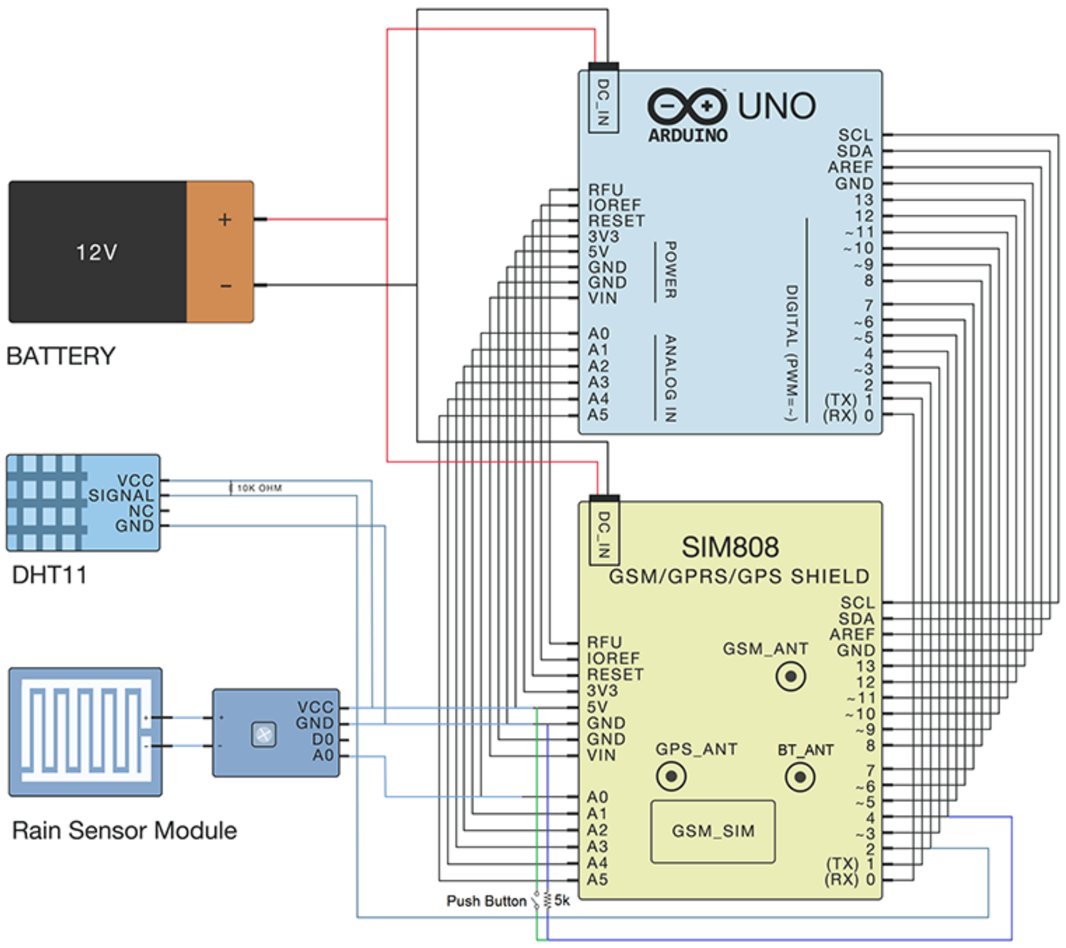


Figure 4.1: Schematic diagram of the embedded device

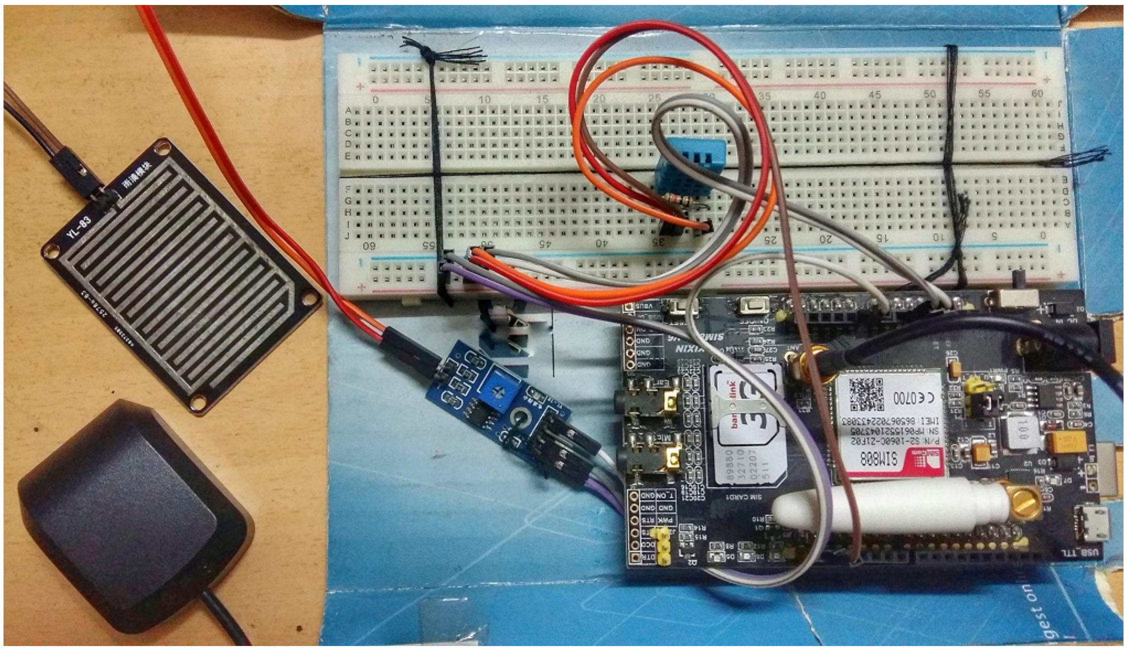


Figure 4.2: Actual Picture of the Embedded Device

module are connected to 5v pin and GND pin respectively and Pin A0 is connected to A0 pin of Arduino board. Pin D0 in not connected.

We have used a push button to turn on the built in LED of Arduino UNO of port 13. We connected a wire from digital pin 4 to one leg of the pushbutton. That same leg of the button connects through a pull-down resistor (here 10K ohm) to ground and the other leg of the button connects to the 5 volt supply of Arduino UNO.

When the push button is open (unpressed) there is no connection between the two legs of the push button, so the pin is connected to ground (through the pull-down resistor) and we read a LOW. When the button is closed (pressed), it makes a connection between its two legs, connecting the pin to 5 volts, so that we read a HIGH.

GSM/GPRS/GPS/Bluetooth SIM808 Shield and Arduino UNO board both are powered by 12V battery power. The figure 4.2 contains actual picture of the Embedded Device.

4.1.2 Arduino Programming

4.1.2.1 Setup

The Arduino Uno can be programmed with the (Arduino Software (IDE)). But there are also other IDEs available. We used Arduino IDE in our thesis which we installed on our Windows 10 OS based computer [15]. The figure 4.3 illustrates the setup of the embedded device with Arduino IDE in Windows 10 environment for programming.

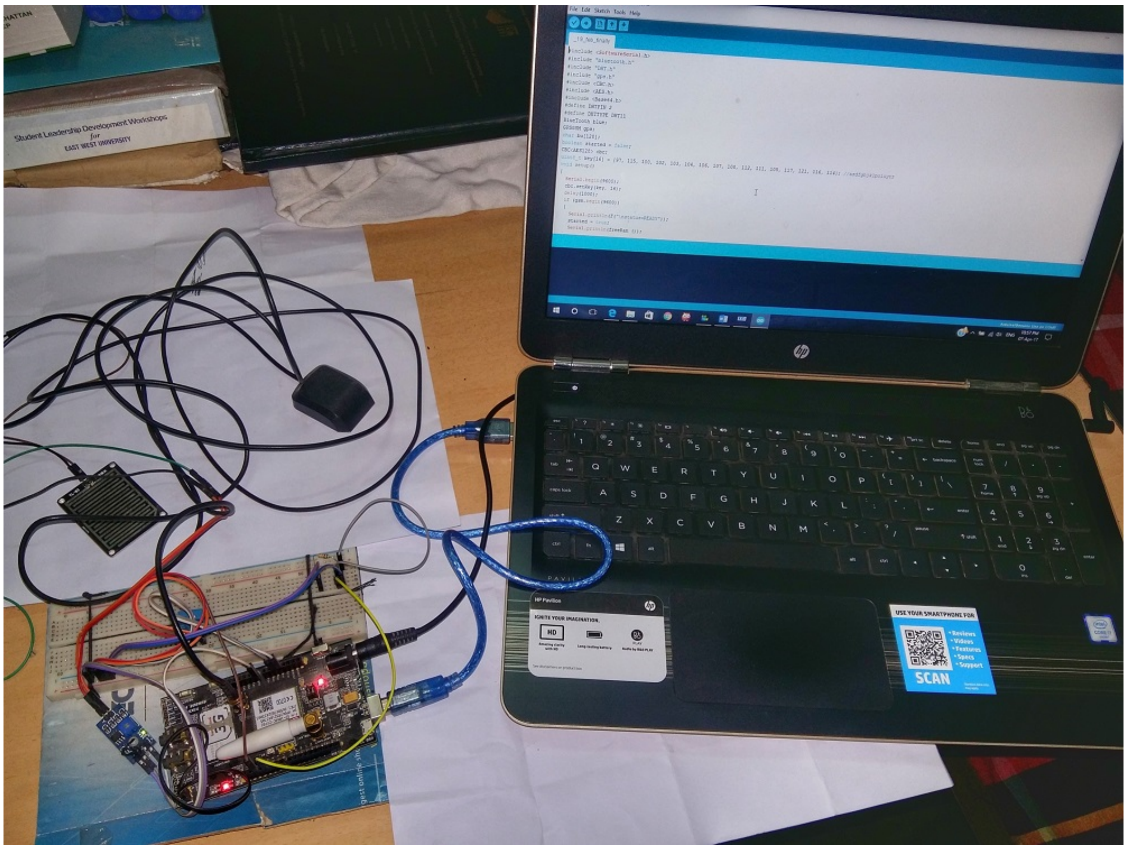


Figure 4.3: Setup of the Embedded device with Arduino IDE in Windows 10 environment

4.1.2.2 Arduino Sketch

Arduino Sketch is mainly divided into two functions which are void setup() and void loop(). Others functions must be called from those two function for use. For communication with other sensors, shield and module there are many libraries available. SoftwareSerial library mainly responsible for serial communication. In our sketch we also used DHT, gps, GSM, Bluetooth and Base64 library.

4.2 Data format

From our embedded device we receive Temperature in celsius and Relative Humidity in percentage. We receive GPS latitude, longitude data and the velocity data in Km/H. But rain sensor module gives analog values within range 0 to 1024. The table 4.1 contains meaning of various range for rain data.

Range	Meaning
0 to 10	No rain
10 to 45	Moderate rain
45 to 118	Heavy rain

Table 4.1: Formatting of Rain Sersor Data

4.3 Cloud Server

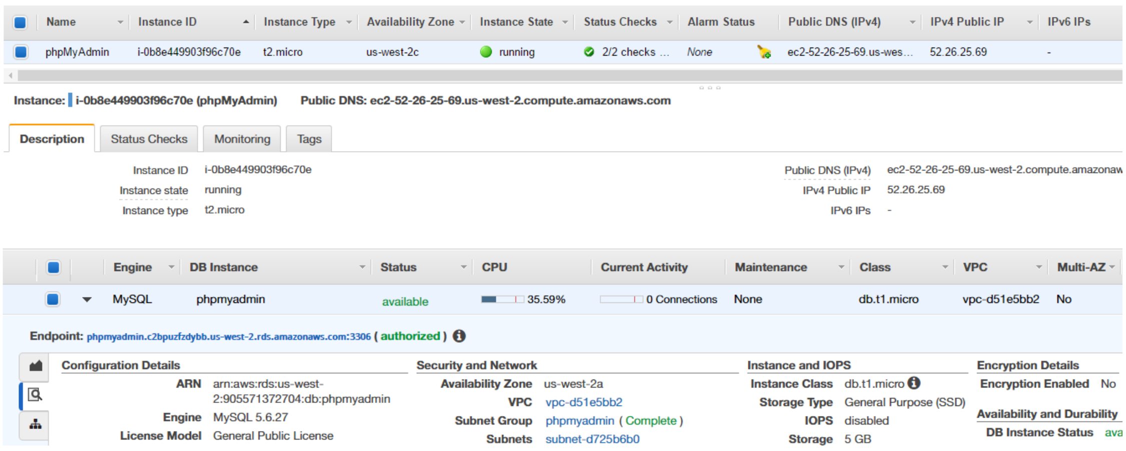
A cloud server is a logical server that is built, hosted and delivered through a cloud computing platform over the Internet. Traditionally there are two main options for hosting: shared hosting and dedicated hosting. Shared hosting is the cheaper option whereby servers are shared between the hosting provider's clients. One client's website will be hosted on the same server as websites belonging to other clients. This has several disadvantages including the fact that the setup is inflexible and cannot cope with a large amount of traffic. Cloud servers possess and exhibit similar capabilities and functionality to a typical server but are accessed remotely from a cloud service provider. Dedicated hosting is a much more advanced form of hosting, whereby clients purchase whole physical servers. Dedicated servers allow for full control over hosting. The downside is that the required capacity needs to be predicted. With cloud hosting clients get the best of both worlds. Resource can be scaled up or scaled down accordingly, making it more flexible and, therefore, more cost-effective. And it is more reliable and will be always up and running. Cloud API service is another reason of using cloud servers over normal servers. For our system, we are using Amazon's cloud service called Amazon Web Services(AWS). It provides both storage and server facility through Amazon RDS and Amazon EC2 instance services.

4.3.1 Setting up AWS

Amazon Web Services (AWS) is a secure cloud services platform, offering compute power, database storage, content delivery and other functionality to help businesses scale and grow. For our TMS we will be using Amazon EC2 and Amazon RDS instances. Amazon Elastic Compute Cloud (EC2) is a web service that provides secure, resizable compute capacity in the cloud. We will be installing Linux based virtual server, Ubuntu server 14.04 LTS (HVM) to manage the MySQL databases. Then we will use Amazon Relational Database Service (RDS) to install the MySQL Database. After Configuring both we will have the EC2 and RDS instances running.

4.3.2 Configuring Database

Now that our backend is set up and running, we can configure our database to store the data that are being sent from the embedded devices. For ease of use, we set up phpMyAdmin application to configure the MySQL database. There are 3 tables: Login,



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs
phpMyAdmin	i-0b8e449903f96c70e	t2.micro	us-west-2c	running	2/2 checks ...	None	ec2-52-26-25-69.us-wes...	52.26.25.69	-

Instance: i-0b8e449903f96c70e (phpMyAdmin) **Public DNS:** ec2-52-26-25-69.us-west-2.compute.amazonaws.com

Description | Status Checks | Monitoring | Tags

Instance ID	i-0b8e449903f96c70e	Public DNS (IPv4)	ec2-52-26-25-69.us-west-2.compute.amazonaw
Instance state	running	IPv4 Public IP	52.26.25.69
Instance type	t2.micro	IPv6 IPs	-

Engine	DB Instance	Status	CPU	Current Activity	Maintenance	Class	VPC	Multi-AZ
MySQL	phpmyadmin	available	35.59%	0 Connections	None	db.t1.micro	vpc-d51e5bb2	No

Endpoint: phpmyadmin.c2bpuzfzdybb.us-west-2.rds.amazonaws.com:3306 (authorized)

Configuration Details

ARN	arn:aws:rds:us-west-2:905571372704:db:phpmyadmin
Engine	MySQL 5.6.27
License Model	General Public License

Security and Network

Availability Zone	us-west-2a
VPC	vpc-d51e5bb2
Subnet Group	phpmyadmin (Complete)
Subnets	subnet-d725b6b0

Instance and IOPS

Instance Class	db.t1.micro
Storage Type	General Purpose (SSD)
IOPS	disabled
Storage	5 GB

Encryption Details

Encryption Enabled	No

Availability and Durability

DB Instance Status	ava

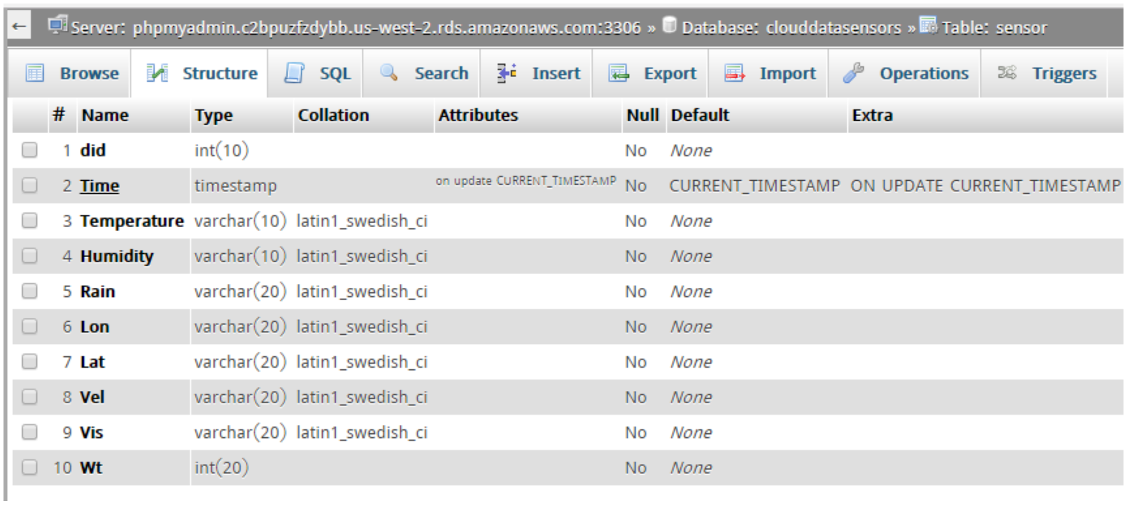
Figure 4.4: Amazon EC2 (Ubuntu) and RDS (MySQL) Instances Running

Device and Sensor. “Login” table stores the username and passwords, “Device” table stores the Mac address of the device and a device ID to identify the device the data is being sent from and the “Sensor” table stores all the data sent from the device. The data is sent with a mac address fixed to the device. The Device table cross match that mac and find the device id (did). Which then get fetched and displayed in the web. The figure 4.5 shows the structure of the sensor table.

There are 9 attributes in the table where the Time is the primary key. Other attributes are: did (Device id), Temperature, Humidity, Rain, Lon (Longitude), Lat (Latitude), Vel (Velocity) and Wt (Weight).

4.3.3 Cloud API: DreamFactory

The ability to enhance the cloud experience and have cross-cloud compatibility has helped form the Cloud API (Application Programming Interface) environment. We can access or update data to the server using the API. DreamFactory provides an easy and secure way to add a REST API to any MySQL database. With the free, open source DreamFactory REST API backend all we have to do is create a service for the database, then use the


←
Server: phpmyadmin.c2bpuzfzdybb.us-west-2.rds.amazonaws.com:3306 » Database: clouddatasensors » Table: sensor
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Triggers

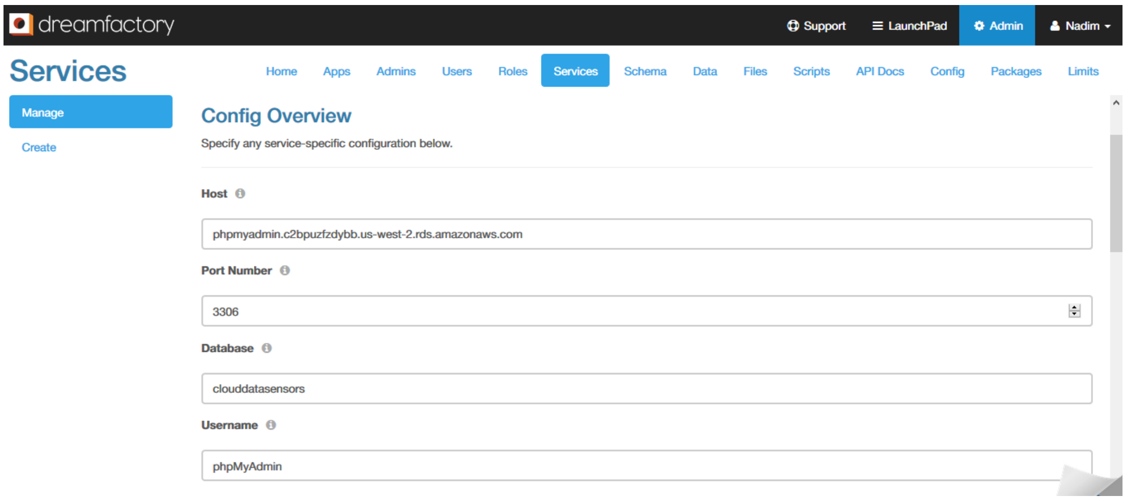
	#	Name	Type	Collation	Attributes	Null	Default	Extra
☐	1	**did**	int(10)			No	*None*	
☐	2	**Time**	timestamp		on update CURRENT_TIMESTAMP	No	CURRENT_TIMESTAMP	ON UPDATE CURRENT_TIMESTAMP
☐	3	**Temperature**	varchar(10)	latin1_swedish_ci		No	*None*	
☐	4	**Humidity**	varchar(10)	latin1_swedish_ci		No	*None*	
☐	5	**Rain**	varchar(20)	latin1_swedish_ci		No	*None*	
☐	6	**Lon**	varchar(20)	latin1_swedish_ci		No	*None*	
☐	7	**Lat**	varchar(20)	latin1_swedish_ci		No	*None*	
☐	8	**Vel**	varchar(20)	latin1_swedish_ci		No	*None*	
☐	9	**Vis**	varchar(20)	latin1_swedish_ci		No	*None*	
☐	10	**Wt**	int(20)			No	*None*	

Figure 4.5: Structure of the Sensor Table

auto-generated REST API to access that service. To set up the DreamFactory, we will again use the AWS service and Bitnami Launchpad will be the client to setup the virtual machine for the API. This will automatically create an EC2 instance for DreamFactory. This will logged the user in with a default email and password. The first screen will prompt the user to change the email and set a new password.

From the Services tab we can now configure the settings to connect with the desired database. In the figure 4.6 we can see the database configuration fields. Host is the public server address where the database is stored in. The default port 3306 is being used to connect. Then the username and password as set up during the database setup. After the service is set up successfully, we can avail the API service going to API Docs tab. Inside that, selecting the newly created service 'Cloud' (the name given for the service), will expand a list of operations we can do on the database. To access, update or delete any data on the database. The figure 4.7 shows a list of few API actions that can be performed in the database.

For example. If we want to get the device id, temperature, humidity and location of



Config Overview

Specify any service-specific configuration below.

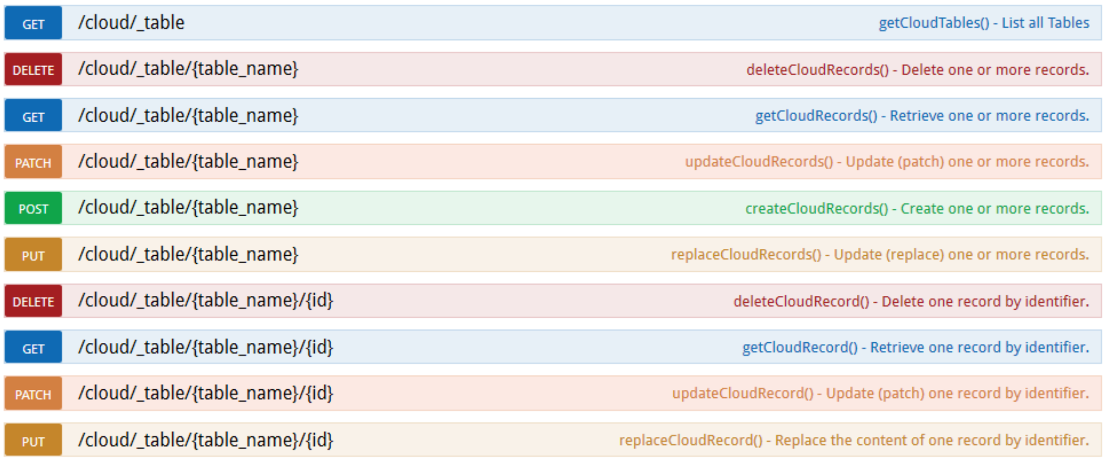
Host: phpmyadmin.c2bpuzfzdybb.us-west-2.rds.amazonaws.com

Port Number: 3306

Database: clouddatasensors

Username: phpMyAdmin

Figure 4.6: API to Database Connection Service Configuration



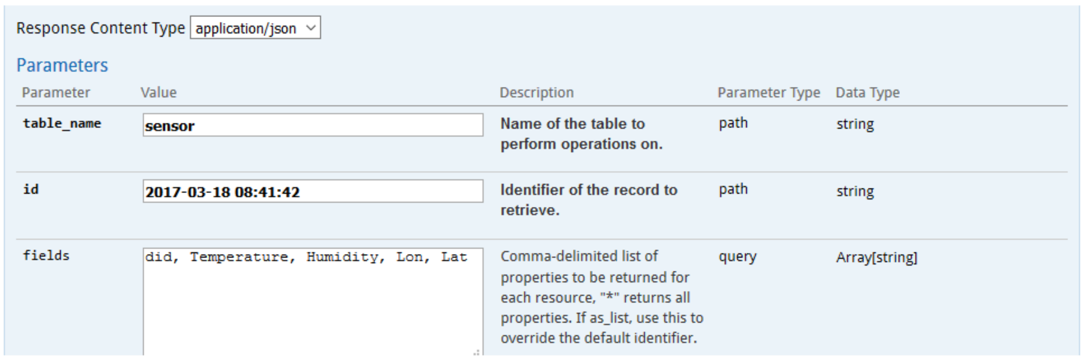
GET	/cloud/_table	getCloudTables() - List all Tables
DELETE	/cloud/_table/{table_name}	deleteCloudRecords() - Delete one or more records.
GET	/cloud/_table/{table_name}	getCloudRecords() - Retrieve one or more records.
PATCH	/cloud/_table/{table_name}	updateCloudRecords() - Update (patch) one or more records.
POST	/cloud/_table/{table_name}	createCloudRecords() - Create one or more records.
PUT	/cloud/_table/{table_name}	replaceCloudRecords() - Update (replace) one or more records.
DELETE	/cloud/_table/{table_name}/{id}	deleteCloudRecord() - Delete one record by identifier.
GET	/cloud/_table/{table_name}/{id}	getCloudRecord() - Retrieve one record by identifier.
PATCH	/cloud/_table/{table_name}/{id}	updateCloudRecord() - Update (patch) one record by identifier.
PUT	/cloud/_table/{table_name}/{id}	replaceCloudRecord() - Replace the content of one record by identifier.

Figure 4.7: A list of few actions of many that we can perform on the database

a specific timestamp: 2017-03-18 08:41:42, we can get this by selecting

```
GET /cloud/_table/{table_name}/{id}.
```

Which retrieves a record by an identifier (the timestamp in this case). In the parameters section we have to input the table name, identifier and the fields (columns) we want to see. The figure 4.8 shows the request and response body of a data retrieval.



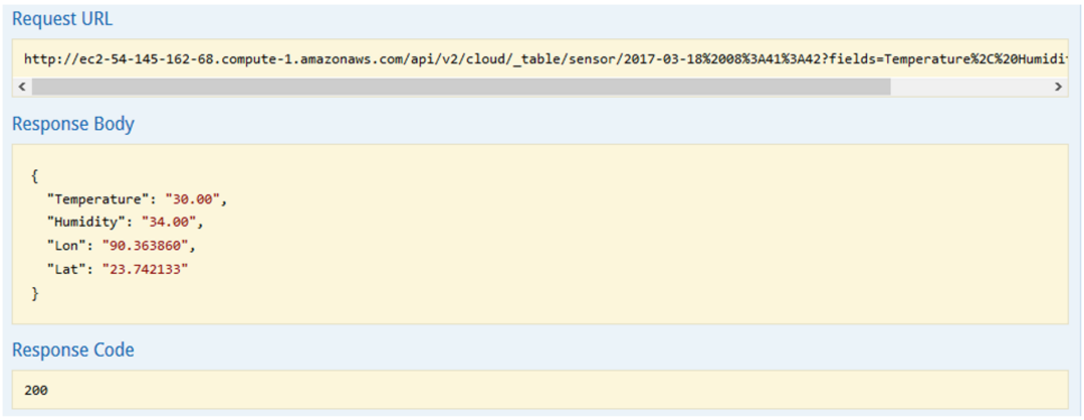
Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
table_name	sensor	Name of the table to perform operations on.	path	string
id	2017-03-18 08:41:42	Identifier of the record to retrieve.	path	string
fields	did, Temperature, Humidity, Lon, Lat	Comma-delimited list of properties to be returned for each resource, "*" returns all properties. If as_list, use this to override the default identifier.	query	Array[string]

Figure 4.8: Retrieving data using an identifier

The response of the request illustrated in the figure 4.9.



Request URL

```
http://ec2-54-145-162-68.compute-1.amazonaws.com/api/v2/cloud/_table/sensor/2017-03-18%2008%3A41%3A42?fields=Temperature%2C%20Humidi
```

Response Body

```
{
  "Temperature": "30.00",
  "Humidity": "34.00",
  "Lon": "90.363860",
  "Lat": "23.742133"
}
```

Response Code

```
200
```

Figure 4.9: Response body of the requested data

That's the simple way of accessing data using the DreamFactory's admin panel. But what if the user want to access it from a website or application? It can be done by a single command line or an URL request.

To access the same data through command line, we have to follow a specific format of request which is shown below:

```
curl -X GET --header 'Accept: application/json' --header 'X-
    DreamFactory-Api-Key: {API Key}' --header 'X-DreamFactory-
    Session-Token: {Session Token}' 'http://ec2-54-145-162-68.
    compute-1.amazonaws.com/api/v2/cloud/_table/{table name}/{Id
    }?{fields}
```

To access the data using URL, we have to follow this format:

```
http://ec2-54-145-162-68.compute-1.amazonaws.com/api/v2/cloud/
    _table/{table name}/{Id}?{fields}
?session_token={session_token}&api_key={API_Key}
```

The returned data from the URL will be this:

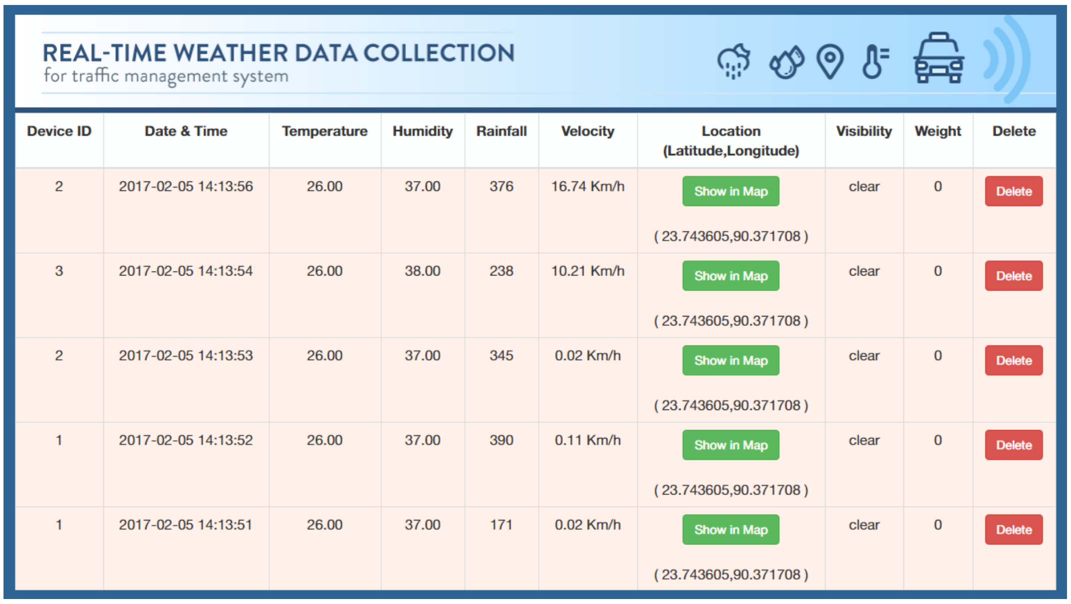
```
{"did":0,"Temperature":"30.00","Humidity":"34.00","Lon":"
    90.363860","Lat":"23.742133"}
```

4.3.4 Data Visualization on Web

The address of the website is `http://52.26.25.69/sensor/index.php`. HTML table is being used to visualize the data on web. The columns are 'Device ID', 'Date & Time', 'Temperature', 'Humidity', 'Rainfall', 'Velocity', 'Location (Latitude, Longitude)', 'Visibility', 'Weight'and 'Delete'. This is the initial structure of the table. Then the data are being fetched from the database to show in their own matching columns. A php script inside the index file first connects with the database server using servername, username, password and database name. Then it collects all the data using the sql query:

```
"SELECT * FROM `sensor` ORDER BY Time DESC";
```

In the table 4.2 the description of each columns explained with an example. The figure 4.10 illustrates the website view of the data.


REAL-TIME WEATHER DATA COLLECTION
for traffic management system

Device ID	Date & Time	Temperature	Humidity	Rainfall	Velocity	Location (Latitude,Longitude)	Visibility	Weight	Delete
2	2017-02-05 14:13:56	26.00	37.00	376	16.74 Km/h	Show in Map (23.743605,90.371708)	clear	0	Delete
3	2017-02-05 14:13:54	26.00	38.00	238	10.21 Km/h	Show in Map (23.743605,90.371708)	clear	0	Delete
2	2017-02-05 14:13:53	26.00	37.00	345	0.02 Km/h	Show in Map (23.743605,90.371708)	clear	0	Delete
1	2017-02-05 14:13:52	26.00	37.00	390	0.11 Km/h	Show in Map (23.743605,90.371708)	clear	0	Delete
1	2017-02-05 14:13:51	26.00	37.00	171	0.02 Km/h	Show in Map (23.743605,90.371708)	clear	0	Delete

Figure 4.10: Screenshot of data visualization on the web

4.4 Mobile Application

We developed an Android application for Android powered mobile phones to receive data from embedded device via Bluetooth and send it to cloud via mobiles internet (GSM,2G,3G,LTE,Wi-Fi etc) connection when embedded device fails to connect with internet or out of Bluetooth range of another embedded device. We named this application as BluSPP.

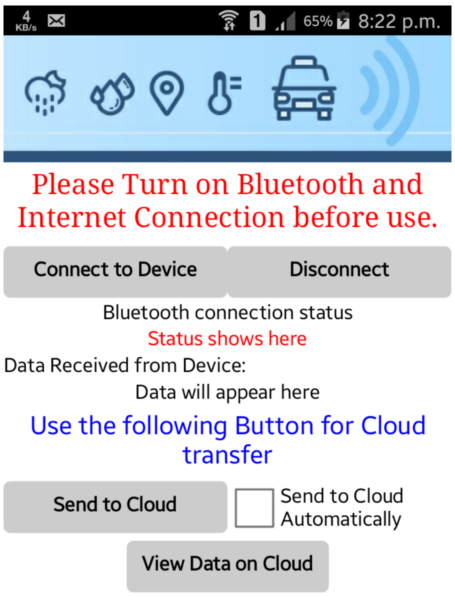
User is responsible for pair their mobile with our embedded device. After successful pair user can send connection request from this application. This application starts SPP profile connection with embedded device.

After successful SPP profile connection device starts sending sensor data to BluSPP and then BluSPP sends receiving data to cloud.

BluSPP uses user mobile phones internet so it may cost user some money for data charge which varies with mobile operator.

Users are able control how much data BluSPP sends to cloud with 'Send to cloud 'and 'Send to cloud automatically 'option of BluSPP application.

The figure 4.11 and 4.12 contains screenshot of two different screen of BluSPP.


Please Turn on Bluetooth and Internet Connection before use.

Connect to Device | Disconnect

Bluetooth connection status

Status shows here

Data Received from Device:

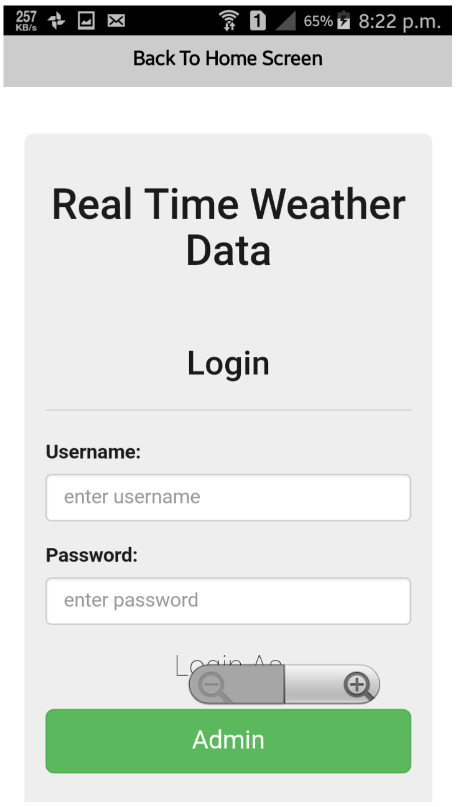
Data will appear here

Use the following Button for Cloud transfer

Send to Cloud | Send to Cloud Automatically

View Data on Cloud

Figure 4.11: Main screen of BluSPP application


Back To Home Screen

Real Time Weather Data

Login

Username: enter username

Password: enter password

Login As

Admin

Figure 4.12: Web login screen of BluSPP application

4.5 Security

4.5.1 User Authentication for Login

To access the website, the use has to go through a login page. There is a table "Login" stored in the database with username and password. After the user input the username and password, it got matched with the fetched data from the database and if matches, it will log the user in and a new session will be started. We have discussed about the session security and management in 4.5.2. This username also enables the data to load on the site. So, even if someone got access to the main webpage no data will be loaded until the correct username and password is given. The figure 4.13 shows the user login screen of the website.

4.5.2 Session Security

A session can be defined as a server-side storage of information that is desired to persist throughout the user's interaction with the web site or web application.

Instead of storing large and constantly changing information via cookies in the user's browser, only a unique identifier is stored on the client side (called a 'session id '). This session id is passed to the web server every time the browser makes an HTTP request. The web application pairs this session id with its internal database and retrieves the stored variables for use by the requested page.

On the login page and index page of the web, we have also added session security. After arriving on the login page, the user session starts by

```
session_start();.
```

It automatically creates a session id and puts it in the browser as a cookie. Then the login variables gets stored in the session and got retrieve each time after cross checking the session id. If the user logs out, the session id will expire and a new one will be generated for each new login sessions.

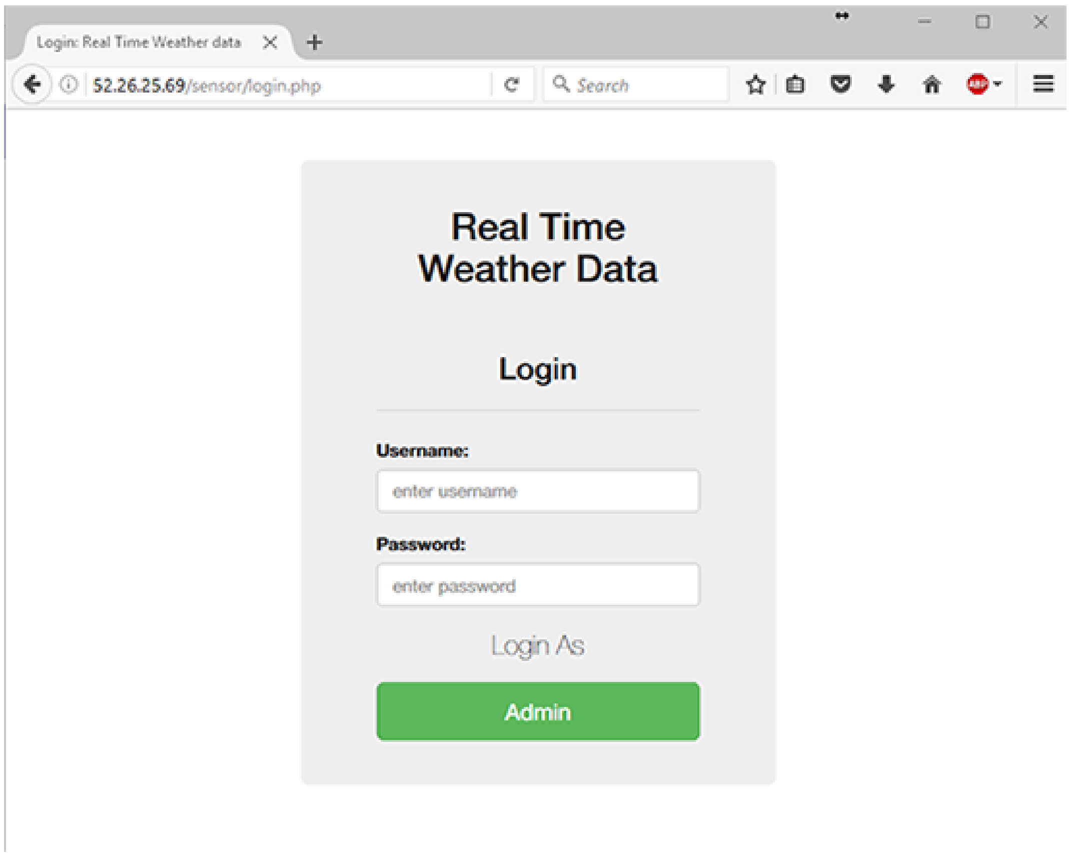


Figure 4.13: User login screen of the website

4.5.3 Bluetooth Security

Bluetooth technology uses short-wavelength radio transmissions in the ISM band 2400-2480 MHz is standard for exchanging data over short distances from fixed and mobile devices. It creates personal area networks (PANs) with high levels of security. Bluetooth was standardized as IEEE 802.15.1. The Bluetooth version of our device is BT3.0, which is also known as Bluetooth High Speed (HS) [25]. There are many Bluetooth profiles. Four basic Bluetooth profile supported by our embedded device are GAP/SDAP/SPP/-GOEP Profile. Secure Simple paring was introduced in Bluetooth version 2.1. We used SSP Just Works model paring in our device as it was designed for the situation where at least one of the pairing devices has neither a display nor a keyboard for entering digits. Just Works model of Secure Simple paring improves security through the addition of ECDH public key cryptography for protection against passive eavesdropping. Though it does not provide protection for man-in-the-middle attacks (MITM) [26].

Name	Description	Example Data
Device ID	The Identification number of the device, data sent from	02
Time	Time and Date of the data update	2017-03-27 15:54:17
Temperature	Shows the temperature data in Celsius	30.00
Humidity	Quantity of humid	46.00
Rainfall	Rainfall reading from the sensor.	6.56
Velocity	Velocity of the vehicle in km per hour	16.74 km/h
Location (Latitude, Longitude)	The combined value of Latitude and Longitude to get the exact location	23.742133, 90.363860
Weight	Calculated road weight value	2
Delete	Deletes the entire record of that time	Data has been deleted.

Table 4.2: Description of Columns with an Example of Data Visualization on Web

Chapter 5
Conclusions and Future work

The IoT Based Real Time Data Collection for Dynamic Road Weight Measurement is successfully implemented with active cloud based server and API and fulfils all the requirements to be an IoT based framework. Our sensor based device is capable of reading and collecting the required data and sends them securely to the database stored in a cloud based server. In case of GSM connection failure of a specific device, we have included a secondary option to transfer device data through Bluetooth which ensures easy data exchange and reduces potential data lost. Web based real time data visualization makes it convenient to see all the data in a clean, formatted and user friendly way.

So far, we successfully implemented the data collection part. In near future, we will integrate the data collection with other data processing TMS modules. We will also improve the data communication method using SMS and mobile app based notification, voice routing suggestion etc. On the web based routing suggestion, we will use Google Maps API to suggest the optimized route directly on the map in case of any congestion detect. The detection will be automatically done by users current position. User will be able to attain the service visiting our website. Enhanced security protocol will be included in near future.

Bibliography

[1] M. R. Rahman and S. Akhter, "Real time bi-directional traffic management support system with gps and websocket," in *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, Oct 2015, pp. 959–964.

[2] S. Nawrin, M. R. Rahman, and S. Akhter, "Exploreing k-means with internal validity indexes for data clustering in traffic management system," in *International Journal of Advanced Computer Science and Applications*, 2017.

[3] M. R. Rahman and S. Akhter, "Bi-directional traffic management support system with decision tree based dynamic routing," in *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, Dec 2015, pp. 170–175.

[4] S. Akhter, R. Rahman, and A. Islam, "Neural network nn based route weight computation for bi-directional traffic management system," *Int. J. Appl. Evol. Comput.*, vol. 7, no. 4, pp. 45–59, Oct. 2016. [Online]. Available: https://doi.org/10.4018/IJAEC.2016100103

[5] M. R. Rahman and S. Akhter, "Bidirectional traffic management with multiple data feeds for dynamic route computation and prediction system," in *International Journal of Intelligent Computing Research*, 2016.

[6] "Radio-frequency identification," [Online; accessed 1-April-2017]. [Online]. Available: https://en.wikipedia.org/wiki/Radio-frequency_identification

[7] "Arduino," [Online; accessed 20-February-2017]. [Online]. Available: https://en.wikipedia.org/wiki/Arduino

[8] "What is cloud computing?" [Online; accessed 22-March-2017]. [Online]. Available: https://azure.microsoft.com/en-in/overview/what-is-cloud-computing/

[9] T. G. Peter Mell, "The nist definition of cloud computing," in *Recommendations of the National Institute of Standards and Technology*, 2011.

[10] "What is iaas?" [Online; accessed 22-March-2017]. [Online]. Available: https://azure.microsoft.com/en-us/overview/what-is-iaas/

[11] "What is saas," [Online; accessed 23-March-2017]. [Online]. Available: https://azure.microsoft.com/en-us/overview/what-is-saas/

[12] "What is paas?" [Online; accessed 23-March-2017]. [Online]. Available: https://azure.microsoft.com/en-us/overview/what-is-paas/

[13] D. T. D. M. H. Tschofenig, J. Arkko, "Architectural considerations in smart object networking," Internet Requests for Comments, 2015. [Online]. Available: https://tools.ietf.org/html/rfc7452

[14] L. Lu, "Wise-paas introduction," Internet, 2017.

[15] "Arduino board uno," [Online; accessed 05-January-2017]. [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUno

[16] "Gprs gsm gps bluetooth all in one sim808 shield for arduino," in *https://store.roboticsbd.com/arduino-shield/322-gprs-gsm-gps-bluetooth -all-in-one-sim808-shield-for-arduino.html*, [Online; accessed 09-April-2017].

[17] "Temperature and humidity module dht11 product manual," [Online; accessed 09-April-2017]. [Online]. Available: https://akizukidenshi.com/download/ds/aosong/DHT11.pdf

[18] "Arduino rain sensor module guide and tutorial," in *https://henrysbench.capnfatz.com/henrys-bench/arduino-sensors-and-input/arduino-rain-sensor-module-guide-and-tutorial/*, [Online; accessed 09-April-2017].

[19] X. C. Xi Yu, Fuquan Sun, "Intelligent urban traffic management system based on cloud computing and internet of things," in *2012 International Conference on Computer Science and Service System*, 2012.

[20] M. P.Sivasankar and B.Brindhavathy, "Iot based traffic monitoring using raspberry pi," in *International Journal of Research in Engineering, Science and Technologies*, 2015.

[21] M. Lakshminarasimhan, "Iot based traffic management system," [Online; accessed 09-April-2017]. [Online]. Available: www.researchgate.net/publication/310036684_IoT_Based_Traffic_Management_System

[22] D. J.Sherly, "Internet of things based smart transportation systems," in *International Research Journal of Engineering and Technology*, 2015.

[23] D. v. D. p. G. D. Nikita Tendulkar, Komal Sonawane, "A review of traffic management system using iot," in *International Journal of Modern Trends in Engineering and Research*, 2016.

[24] allychevalier, "What is the difference between gps and rfid tracking?" [Online; accessed 09-April-2017]. [Online]. Available: http://www.brighthub.com/electronics/gps/articles/60599.aspx

[25] "Sim800 series bluetooth application note v1.04," in *http://cdn-shop.adafruit.com/product-files/1946/SIM800+Series_Bluetooth_Application_Note_V1.04.pdf*, [Online; accessed 09-April-2017].

[26] L. C. John Padgette, Karen Scarfone, "Guide to bluetooth security," in *Recommendations of the National Institute of Standards and Technology*, 2012.

Appendix A
List of Acronyms

AWS	Amazon Web Services
MITM	Man in The Middle Attack
TMS	Traffic Management System
IoT	Internet of Things
GSM	Global System for Mobile Communications
GPRS	General Packet Radio Service
GPS	Global Positioning System
RFID	Radio Frequency Identification
HTTP	Hypertext Transfer Protocol
SaaS	Software as a Service
PaaS	Platform as a Service
IaaS	Infrastructure as a Service
IT	Information Technology
ANT	Antenna
EDGE	Enhanced Data for GSM Evolution
IrDA	Infrared Detection and Array
LTE	Long Term Evolution
NFC	Near Field Communication
WLAN	Wireless Local Area Network
ML	Machine Learning

CO2	Carbon Dioxide
I/O	Input Output
PWM	Pulse Width Modulation
DC	Direct Current
SRAM	Static Random Access Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory
LED	Light Emiting Diode
SIM	Subscriber Identity Module
BT3.0	Bluetooth 3.0
GND	Ground
UNO	Universal Network Objects
A0	Analog Output
D0	Digital Output
VCC	Voltage at the Common Collector
EC2	Elastic Compute Cloud
RDS	Relational Database Service
Lon	Longitude
Lat	Latitude
REST	Representational State Transfer
API	Application program interface
URL	Uniform Resource Locator
HTML	Hypertext Mark-up Language
php	Hypertext Preprocessor
PAN	Personal area network
SSP	Shared Service Provider
SMS	Short Message Service