

**A CLOUD SOLUTION FOR VISUALIZING AND NAVIGATING MOBILE IOT  
SENSOR NODES**

**BY**

**SEE TIAN XIN**

**A REPORT**

**SUBMITTED TO**

**Universiti Tunku Abdul Rahman**

**in partial fulfillment of the requirements**

**for the degree of**

**BACHELOR OF COMPUTER SCIENCE (HONS)**

**Faculty of Information and Communication Technology  
(Kampar Campus)**

**JANUARY 2019**

## REPORT STATUS DECLARATION FORM

**Title:** A Cloud Solution for Visualizing and Navigating Mobile IoT Sensor Nodes

\_\_\_\_\_  
\_\_\_\_\_

**Academic Session:** January 2019

I SEE TIAN XIN  
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in  
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

\_\_\_\_\_  
(Author's signature)

\_\_\_\_\_  
(Supervisor's signature)

**Address:**  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

Supervisor's name

**Date:** \_\_\_\_\_

**Date:**

**A CLOUD SOLUTION FOR VISUALIZING AND NAVIGATING MOBILE  
IOT SENSOR NODES**

**BY  
SEE TIAN XIN**

**A REPORT  
SUBMITTED TO  
Universiti Tunku Abdul Rahman  
in partial fulfillment of the requirements  
for the degree of  
BACHELOR OF COMPUTER SCIENCE (HONS)  
Faculty of Information and Communication Technology  
(Kampar Campus)**

**JANUARY 2019**

## DECLARATION OF ORIGINALITY

I declare that this report entitled “**A CLOUD SOLUTION FOR VISUALIZING AND NAVIGATING MOBILE IOT SENSOR NODES**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : \_\_\_\_\_

Name : \_\_\_\_\_

Date : \_\_\_\_\_

## **ACKNOWLEDGEMENTS**

I would like to express my sincere thanks and appreciation to my supervisor, Dr. Ooi Boon Yaik who has given me this bright opportunity to engage in an IoT visualization project. A million thanks to you.

I would also like to thank UTAR Undergraduate Research Scheme (URS) and CREST Undergraduate Research Project for sponsoring on this project.

Finally, I must say thanks to my parents and my family for their love, support and continuous encouragement throughout the whole Final Year Project.

## **ABSTRACT**

This project is to develop a cloud solution for visualizing and navigating mobile IoT sensor nodes. The field of study of this project is related to IoT and distributed system. The problem that this project will solve is to give indoor map support for mobile sensors. So, this project focuses on building a solution to solve the lack of support for mobile IoT sensor nodes and insufficient indoor map support for the existing visualization platforms. As mobile IoT devices are more widely used nowadays, it is a need to have a visualization platform that can fully support mobile IoT devices. The methodology adopted to develop the visualization platform is by using prototyping model. There are two prototypes being produced before developing the final product. The scope of this project is to build an extension from the existing data visualization tools instead of building the whole visualization tool from scratch. It is because there are many existing tools available and it is redundant to build all the existing functions that are available in the market. Some important functions that are included in the visualization platform are able to track the real-time location of the mobile sensors, visualize its current location on an indoor map, and to display different types of data collected from the sensors in a 2D graph. Hence, instead of developing everything from scratch, this project will focus on developing an extension that is able to solve the problem statements as mentioned.

## TABLE OF CONTENTS

<b>TITLE PAGE</b>	<b>i</b>
<b>DECLARATION OF ORIGINALITY</b>	<b>ii</b>
<b>ACKNOWLEDGEMENTS</b>	<b>iii</b>
<b>ABSTRACT</b>	<b>iv</b>
<b>TABLE OF CONTENTS</b>	<b>v</b>
<b>LIST OF FIGURES</b>	<b>viii</b>
<b>LIST OF TABLES</b>	<b>x</b>
<b>LIST OF ABBREVIATIONS</b>	<b>xi</b>
<b>CHAPTER 1: INTRODUCTION</b>	<b>1</b>
1.1 Problem Statement	1
1.2 Background and Motivation	1
1.3 Objectives	4
1.4 Proposed Approach/Study	4
1.5 Highlight of Achievement	7
1.6 Report Organization	7
<b>CHAPTER 2: LITERATURE REVIEW</b>	<b>8</b>
2.1 Review on ThingSpeak	8
2.2 Review on Freeboard	9
2.3 Review on IBM Watson Analytics	9
2.4 Review on Atrius Solution Builder	10
2.5 Review on Tableau	11
2.6 Comparison on the Reviewed Data Visualization Platforms	12
<b>CHAPTER 3: SYSTEM DESIGN</b>	<b>16</b>
3.1 System Design Diagrams	16
3.1.1 System Overview Diagram	16
3.1.2 Use Case Diagram	17

3.1.3 Use Case Description	18
3.1.4 ER Diagram	21
3.1.5 Class Diagram	22
3.1.6 Sequence Diagram	23
3.1.7 Communication Diagram	24
<b>CHAPTER 4: IMPLEMENTATION</b>	<b>25</b>
4.1 Methodology	25
4.2 Tools	25
4.2.1 Platform	25
4.2.2 Programming Language	25
4.2.3 Software Tools	26
4.2.4 Hardware Tools	26
4.3 Implementation Phases	28
4.3.1 Setting up AWS EC2	28
4.3.2 Developing Visualization of Indoor Map	28
4.3.3 Communication between Cloud Server and Visualization Framework	28
4.3.4 Calculation of Best Route for Mobile IoT Sensors	29
4.3.5 Developing 2D Line Graph	30
4.3.6 AGV Calibration	31
4.3.7 Two-way Communication between Mobile IoT Sensors and Cloud Server	32
<b>CHAPTER 5: TESTING</b>	<b>33</b>
5.1 Testing between Visualization Framework and Mobile IoT Sensors	33
5.1.1 Real-time Location Update Test	33
5.1.2 Best Route Calculation Test	35
5.2 Testing on the AGV	37
5.2.1 QR Code Distance Test	37



5.2.2 AGV Travelling Speed Test	38
5.3 Issues and Challenges	38
<b>CHAPTER 6: CONCLUSION</b>	<b>39</b>
6.1 Project Review, Discussions and Conclusions	39
6.2 Future Work	40
<b>BIBLIOGRAPHY</b>	<b>41</b>
<b>APPENDICES</b>	<b>44</b>
Appendix A	44
<b>POSTER</b>	<b>47</b>
<b>PLAGIARISM CHECK RESULT</b>	<b>48</b>
<b>CHECKLIST</b>	<b>51</b>

## LIST OF FIGURES

<b>Figure Number</b>	<b>Title</b>	<b>Page</b>
Figure 1-2-1	One-dimensional line graph by Michael Florent	2
Figure 1-2-2	Tag cloud	2
Figure 1-2-3	Sparkline example	3
Figure 1-2-4	Trendalyzer example	3
Figure 1-4	Flowchart of the whole development process	6
Figure 2-1-1	ThingSpeak IoT System	8
Figure 2-2-1	Freeboard's Dashboard	9
Figure 2-3-1	Outlook of IBM Watson Analytics	10
Figure 2-4-1	Overview of Atrius Solution Builder	11
Figure 2-5-1	Tableau's dashboard	12
Figure 2-6-1	Atrius Solution Builder's indoor floor map	13
Figure 3-1-1	System Overview	16
Figure 3-1-2	Use Case Diagram	17
Figure 3-1-4	ER Diagram	21
Figure 3-1-5	Class Diagram	22
Figure 3-1-6	Sequence Diagram	23
Figure 3-1-7	Communication Diagram	24
Figure 4-2-4-1	Omni wheel robot	27
Figure 4-2-4-2	Air temperature and humidity sensor	27
Figure 4-3-2	Visualization for indoor map	28
Figure 4-3-3-1	PHP script to get indoor map coordinates from database	29
Figure 4-3-3-2	PHP script to get moving robots' location from database	29
Figure 4-3-4	User select the destination point to move for moving robots	30
Figure 4-3-5	Air Temperature and Humidity Line Chart	31
Figure 4-3-6	QR code's top left corner as the correction point	31
Figure 5-1-1-1	Original Location of Moving Robot	33

Figure 5-1-1-2	Updated location of moving robot 1	33
Figure 5-1-1-3	Updated location of moving robot 2	34
Figure 5-1-1-4	Updated location of moving robot 3	34
Figure 5-1-1-5	Moving Robot back to its Original Location	35
Figure 5-1-2-1	Selected destination point for moving robot RB001	35
Figure 5-1-2-2	Calculated best route	36
Figure 5-1-2-3	Mobile IoT sensor movement according to instruction	36
Figure 5-1-2-4	Mobile IoT sensor reached its destination point	37

## LIST OF TABLES

<b>Table Number</b>	<b>Title</b>	<b>Page</b>
Table 2-6	Comparison table between five data visualization platforms	12
Table 3-1-3-1	UC001 Use case description	18
Table 3-1-3-2	UC002 Use case description	18
Table 3-1-3-3	UC003 Use case description	19
Table 3-1-3-4	UC004 Use case description	19
Table 3-1-3-5	UC005 Use case description	20
Table 3-1-3-6	UC006 Use case description	20
Table 5-2-1	Result of QR Code Distance Test	37
Table 5-2-2	Comparison of the results taken from Travelling Speed Test	38

## LIST OF ABBREVIATIONS

<i>IoT</i>	Internet of Things
<i>FYP</i>	Final Year Project
<i>HTML5</i>	Hyper Text Markup Language 5
<i>MQTT</i>	Message Queuing Telemetry Transport
<i>REST</i>	Representational State Transfer
<i>FTP</i>	File Transfer Protocol
<i>API</i>	Application Program Interface
<i>WAN</i>	Wide Area Network
<i>2D-graph</i>	Two-Dimensional Graph
<i>ER Diagram</i>	Entity-Relationship Diagram
<i>AWS</i>	Amazon Web Services
<i>EC2</i>	Elastic Compute Cloud
<i>MySQL</i>	My Structured Query Language
<i>QR CODE</i>	Quick Response Code
<i>PHP</i>	Hypertext Preprocessor
<i>CSS</i>	Cascading Style Sheets
<i>JS</i>	JavaScript
<i>VNC</i>	Virtual Network Computing
<i>SSH</i>	Secure Shell
<i>AGV</i>	Automatic Guided Vehicle
<i>DFS</i>	Depth First Search
<i>cm</i>	Centimeter

## **CHAPTER 1: INTRODUCTION**

### **1.1 Problem Statement**

As for this project, a visualization platform is developed for the use of mobile IoT sensor nodes. The problem domain of this project is the data visualization platform. There are a lot of visualization platforms in the market which have different functions and features available. The problem of these existing platforms is that they do not support mobile sensor nodes. Hence, a visualization platform that can fully support mobile IoT sensor nodes is needed to be developed.

The problem statements of this project can be summarized as below:

1. Existing IoT visualization platforms do not support mobile IoT sensor nodes.
2. Indoor map support for visualization platforms is not sufficient to visualize mobile sensors data.

The problem statement of this project is the existing IoT visualization platforms are not able to support mobile IoT sensor nodes. It needs to be taken into considerations as most of our IoT devices nowadays are not static. The second problem statement is the indoor map support for visualization platforms is not sufficient to visualize mobile sensors data. If there is a mobile sensor moving around a designated area, the indoor map will become extremely important and helpful in visualizing where the mobile sensor is currently located at. Hence, this project will focus on solving the two problem statements mentioned above.

### **1.2 Background and Motivation**

Data visualization has been around far longer than computers have existed. From simple cave drawings to the dashboards we have today that display and connect millions of data. Although the term “Data Visualization” just has its firm roots in the 20<sup>th</sup> Century, the idea is as old as time (Horne, 2017). The first representation of statistical data is created on 1644 by Michael Florent Van Langren, a Flemish astronomer (Friendly, et al., 2010).



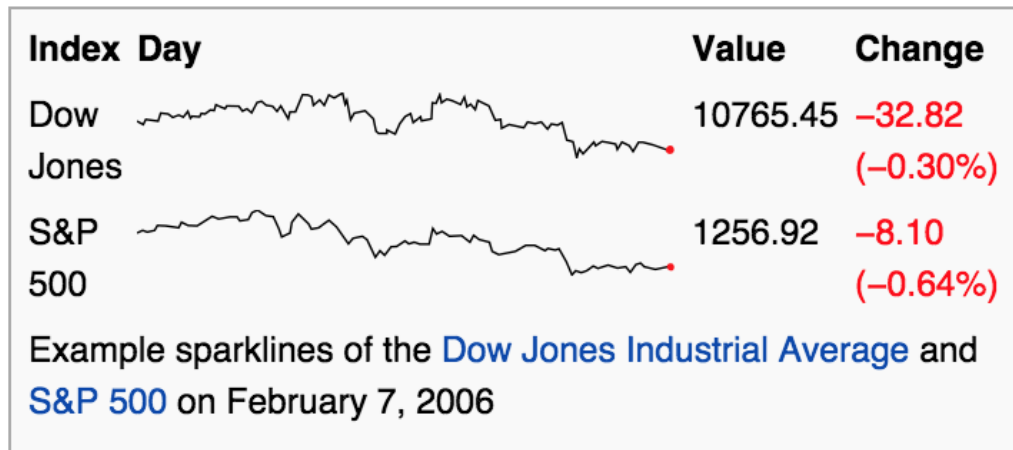


Figure 1-2-3: Sparkline example (Stock prices 2018)

The third one is the Trendalyzer, which is also known as Gapminder, invented by Swede Hans Rosling (Gapminder, 2018). It is a moving bubble chart where information is displayed in five different variables which are the x and y axes, bubble size, bubble colour and a time variable (Gray, 2015).

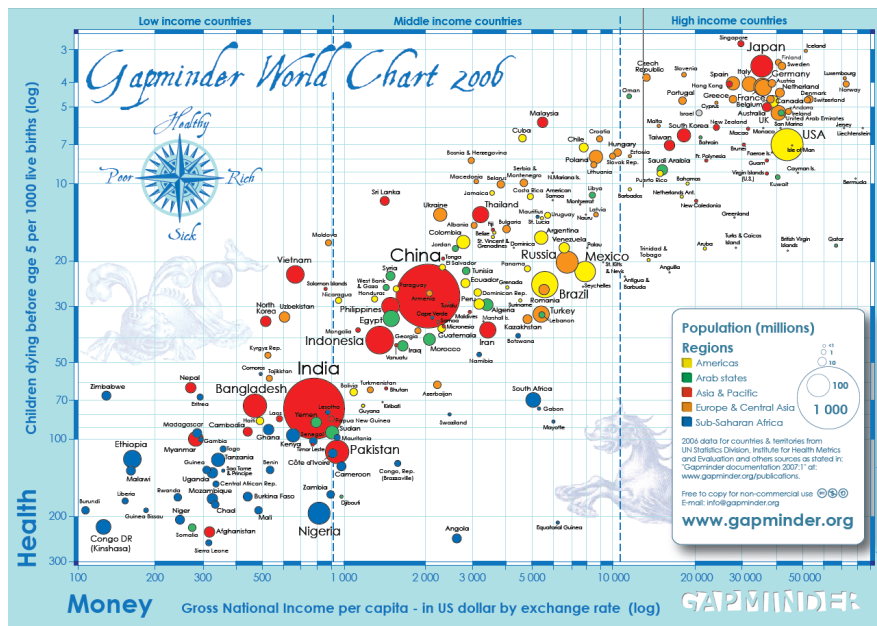


Figure 1-2-4: Trendalyzer example (Trendalyzer 2018)

Although there are a lot of visualization platforms in the market which have different functions and features available, these existing platforms do not support mobile sensor nodes. Hence, a visualization platform that can fully support mobile IoT sensor nodes is needed to be developed. It is important to have this visualization



platform as there is a growing demand on IoT devices which does not only stay at a certain location. The use of robots is becoming more and more popular nowadays. Therefore, this highlights the importance of having a visualization platform that supports mobile IoT nodes.

### **1.3 Objectives**

The objectives of this project can be summarized as below:

1. To develop an IoT visualization platform that can support mobile IoT sensor nodes.
2. To extent the existing indoor map support to support visualization for mobile IoT sensor nodes.

The first objective of this project is to develop an IoT visualization platform that can support mobile IoT sensor nodes. The real-time location of mobile sensors can be tracked and stored in the database by having support for mobile sensors. The second objective is to extent the existing indoor map support to support visualization for mobile IoT sensor nodes. As this project only focus on the extension instead of developing a whole visualization platform, the indoor map support is done as the extension for the existing visualization tools to be able to map the real-time location of the moving mobile sensors.

### **1.4 Proposed Approach/Study**

The proposed methods which are involved in the solutions are gathering the needed requirements, producing prototypes and evaluation on the prototypes. Figure 1-4 below shows the flowchart of this project's development process. There will be a total of two prototypes being produced before the final product. The first prototype is being produced in FYP1 and have the following features:

1. A database with few tables that are used to record the real-time location and other data recorded from the mobile IoT sensors.
2. A visualization platform that can display an indoor mapping with the precise real-time location of the mobile IoT sensors.

The second prototype will be developed in the FYP2 phase and it will have the following features:

1. The system will be able to send instructions to the mobile IoT sensors of which route to go.
2. The system will be able to calculate a best route for the mobile IoT sensors without colliding with other sensors and obstacles.

After the second prototype is being produced, a final prototype, which is the final product will be produced with the additional features as below:

1. A 2D-graph that can display the other data collected from the mobile IoT sensors such as temperature and air humidity.
2. Conclude the overall features and display it out in a nice visualized form.

There are a few tools that are used in this project. One of most important pieces of technology is the existing data visualization tool. As mentioned in the project objectives, this project focuses only on building the extensions for data visualization platform without starting from the scratch. Hence, the need of existing data visualization tool is crucial in this project. Besides, a cloud server is needed to host the data collections and server of the data visualization platform. The other two technologies to be used in the evaluation stage are robots and sensors. Since this project aims to build extensions of data visualization tool for mobile sensors, the need of sensors to move around is needed. Therefore, robots will be used to enable the sensors to move around.

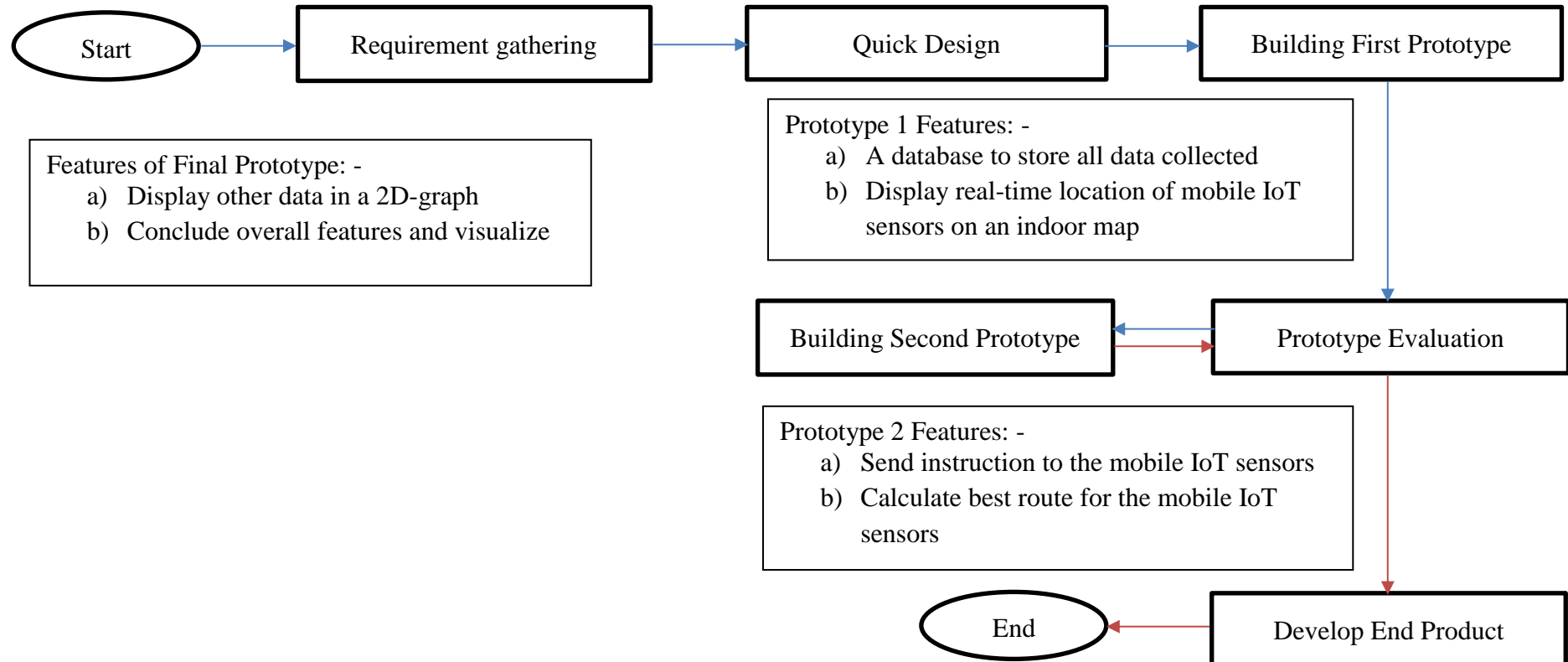


Figure 1-4: Flowchart of the whole development process

### **1.5 Highlight of Achievement**

There are a few highlights that have been achieved in this project to fulfill the project objectives.

1. An indoor map visualization that can display the real-time location of mobile IoT sensors.
2. Two-way communication between the mobile IoT sensors and cloud server.
3. Calculate best route for the mobile IoT sensors to move from one point to another.
4. 2D line graph to display the air temperature and humidity data collected from the mobile IoT sensors.

### **1.6 Report Organization**

This report consists of 6 chapters, which are Introduction, Literature Review, System Design, Implementation, Testing and Conclusion. Chapter 1 is the overview of this project, which includes the problem statement, project objectives, proposed approach and highlights of this project. In Chapter 2, reviews and comparison on different existing visualization framework and comparison on the proposed approach of this project with the existing technology are done. Chapter 3 shows the overall system design and how the project is developed in detail. In the fourth chapter, details of how the implementation of this project is done and what tools are used in the implementation are explained. For the next chapter, which is chapter 5, all the testing techniques that are used to test this system are described in detail. For the last chapter, it concludes what have been achieved in this project and also the future work that can be made to make further developments.

## CHAPTER 2: LITERATURE REVIEW

Data visualization tool is a widely used software in helping users to visualize their data. It is important to visualize our data as it allows us to visually access a huge amount of data into a simplified and easily digestible way. There are more than hundreds of data visualization tools available, whether it is open source or not. Out of hundreds of those, top five on them are reviewed, namely ThingSpeak (ThingSpeak 2018), Freeboard (Freeboard 2018), IBM Watson Analytics (IBM Watson Analytics 2018), Atrius Solution Builder (Atrius Solution Builder 2018) and Tableau (Tableau Software 2018).

### 2.1 Review on ThingSpeak

ThingSpeak is an IoT investigation platform service that enables you to aggregate, envision and analyze your information streams live in the cloud (ThingSpeak 2018). It is launched by the creators of Matlab. For users that are familiar with Matlab, they will find ThingSpeak easy to use for them as well. According to Degeler (2015), for non Matlab users, it is not too hard to learn, as there are good tutorials and documentations provided. The good part about it is that ThingSpeak gives instant representation of data presented by your gadgets to the cloud. The figure 2-1-1 below summarizes how ThingSpeak work with several smart devices connected to it.

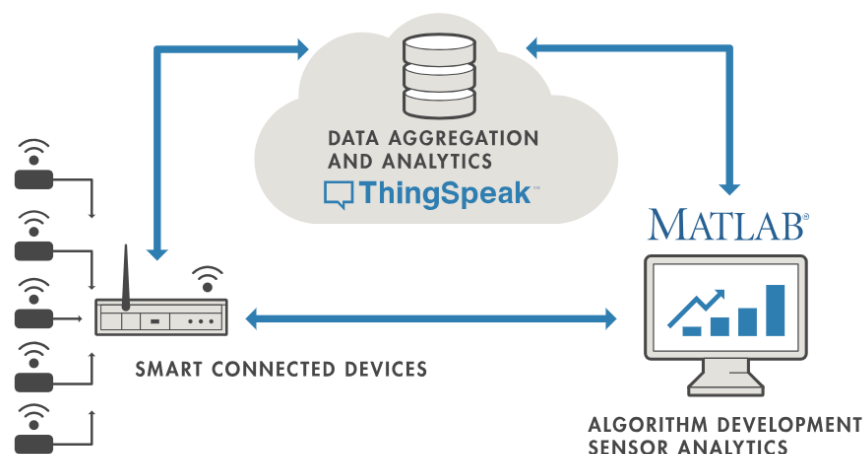


Figure 2-1-1: ThingSpeak IoT System (ThingSpeak 2018)

## 2.2 Review on Freeboard

Freeboard is a purpose-built visualization tool for IoT. It has a dashboard that allows you to create different widgets while sharing it immediately with anyone you like. The best thing about it is that it is simple with its drag and drop interface, which attracts users with zero knowledge in programming. Similar to ThingSpeak, it is completely open source. The other point that makes it even more user friendly is the widgets, which can either be created from scratch or chosen from pre-defined ones (Degeler 2015). The figure 2-2-1 below shows a screen capture of the dashboard from Freeboard.

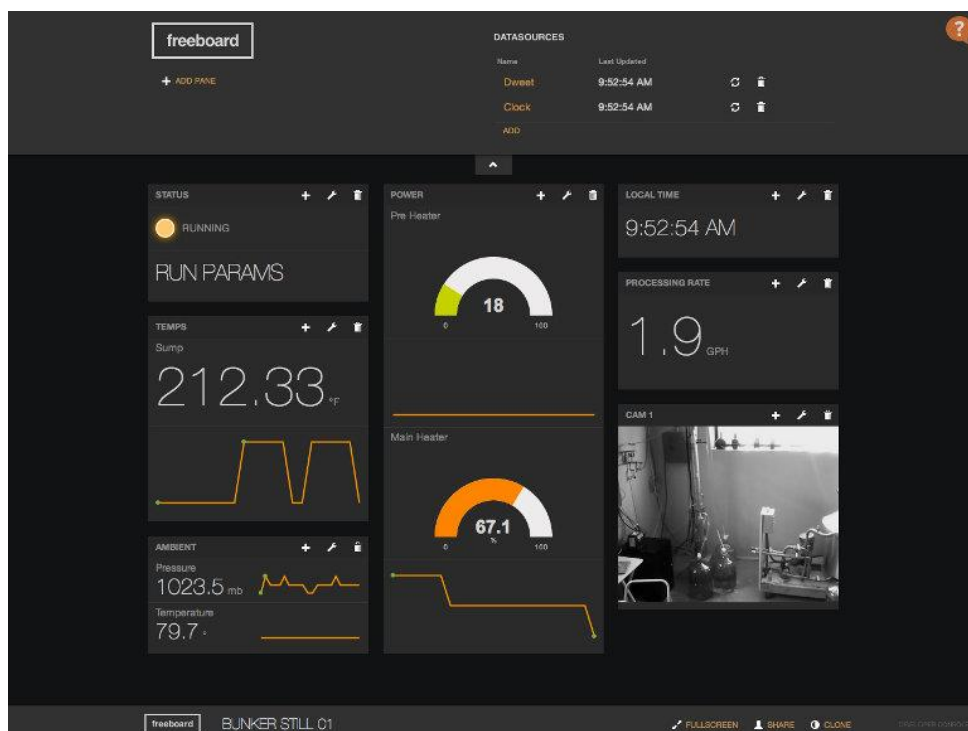


Figure 2-2-1: Freeboard's Dashboard (IoT Visual Freeboard 2018)

## 2.3 Review on IBM Watson Analytics

IBM Watson Analytics is a smart information analysis and representation service on the cloud. It is developed by the IBM company that intends to provide benefits of advanced analytics without complexity (IBM Watson Analytics 2018). It is a ground-breaking yet simple-to-utilize cloud-based tool for data science and business investigation. It is easy to learn because of its profoundly progressed analytics engine works with an outstanding natural language querying platform. According to Baker (2018), IBM Watson Analytics has as much as 32 business connectors which includes

spreadsheets, OneDrive, PayPal, Twitter and etc. IBM Watson Analytics also allows you to query directly from various databases such as Oracle, MySQL, Microsoft Azure and many more. Figure 2-3-1 below shows the outlook of IBM Watson Analytics.

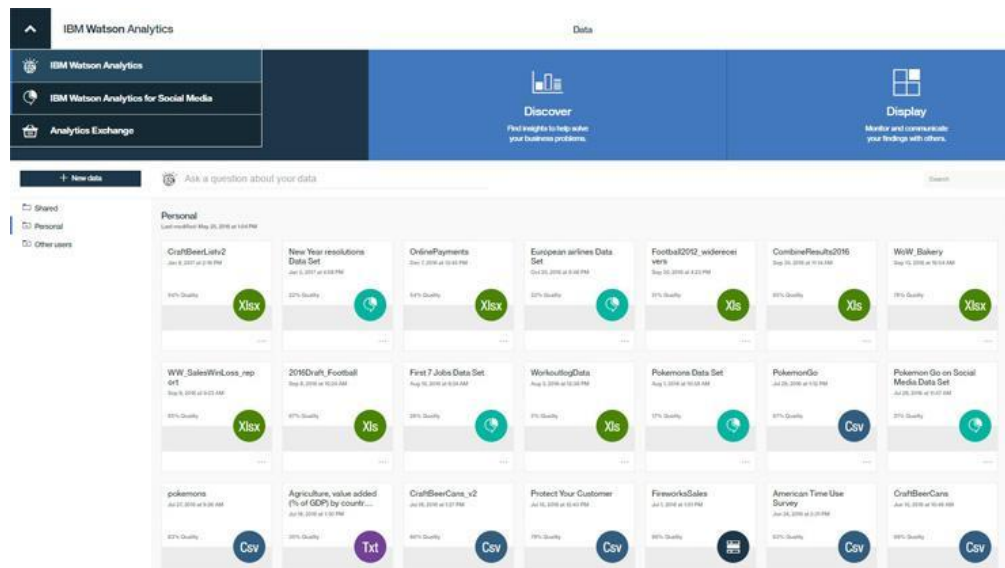


Figure 2-3-1: Outlook of IBM Watson Analytics (IBM Watson Analytics – Homepage Dropdown Menu 2018)

## 2.4 Review on Atrius Solution Builder

Atrius Solution Builder, which is previously known as DGLux5, is an IoT visualization tool by AcuityBrands. It is a HTML5 browser based integrated development environment (IDE) (Atrius Solution Builder 2018). According to Khvostionov (n.d.), it requires no plugins for any browser and no mobile apps needed as well. Atrius Solution Builder features a graphical data-driven toolbox which helps in the rapid build of interactive web applications and dashboards that are visually rich. Like Freeboard, it allows you to drag-and-drop which can maximize productivity and helps user to develop faster. It is a cross-platform application, which means user only need to develop once for desktop, tablet and mobile devices. Figure 2-4-1 below shows how the overview of Atrius Solution Builder.

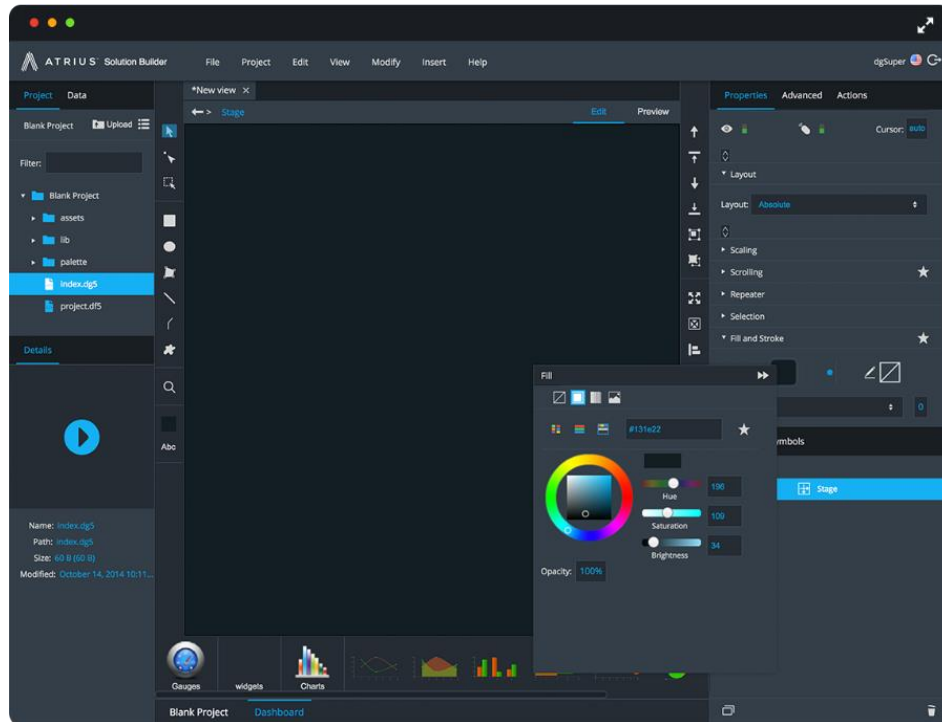


Figure 2-4-1: Overview of Atrius Solution Builder (Atrius IDE, 2018)

## 2.5 Review on Tableau

Last but not least, Tableau is one of the early data visualization tools in the market. Although the number of competitors is rapidly growing along with the rise of Big Data and IoT, Tableau remains a strong contender as a data visualization tool. It allows user to use the drag and drop manner to perform complex and impressively interactive data visualization. User can easily highlight sections and drill-down into charts effortlessly. Tableau also has a robust mobile client developed with touch-optimized controls which make accessing and viewing data much easier. One amazing thing about Tableau is that it will automatically recognize and make adjustment if the user is using the mobile app (Post, 2018). Figure 2-5-1 below shows the dashboard of Tableau.



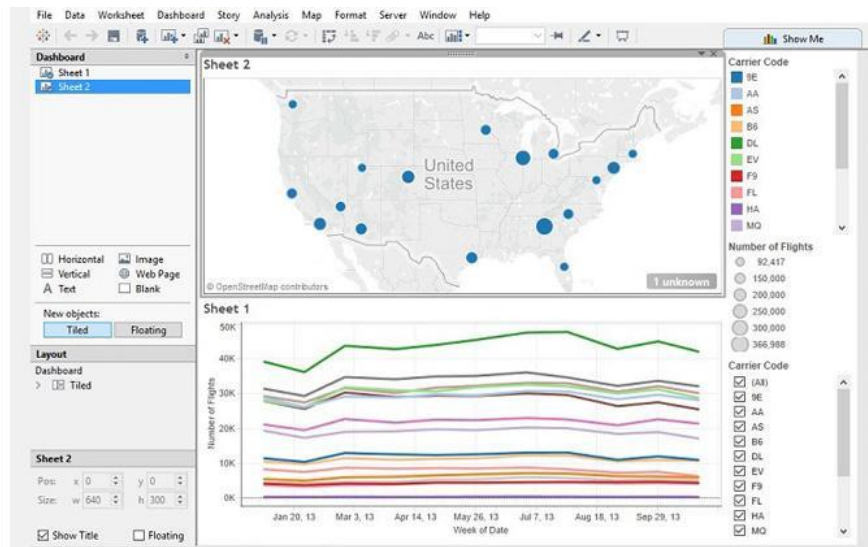


Figure 2-5-1: Tableau’s dashboard (Tableau Desktop – Dashboard 2018)

### 2.6 Comparison on the Reviewed Data Visualization Platforms

After reviewing five of the above data visualization platforms, a comparison table is done to compare a few major criteria between these five platforms. Table 2-6-1 below shows the comparison table of the five data visualization platforms.

Criteria	Outdoor Map Support	Indoor Map Support	Connectivity	Interactive Visualization	Cloud/On-Premise	Mobile Sensor
ThingSpeak	Yes	No	MQTT/REST	Yes	Cloud	No
Freeboard	Yes	No	REST	Yes	Cloud	No
IBM Watson Analytics	Yes	No	MQTT/REST/FTP	Yes	Hybrid	No
Atrius Solution Builder	Yes	No	MQTT	Yes	Hybrid	No
Tableau	Yes	No	MQTT/REST	Yes	Hybrid	No

Table 2-6: Comparison table between five data visualization platforms

There are six criteria that are compared in the above table. One of the criteria is outdoor map support. Outdoor map support means that the data visualization tools must be able to support worldwide map. The data visualization tools must have the ability to visualize a map that displays where the location of your IoT devices are located. For most of the platforms compared above, they support Google Map API, means users can use the Google Map widget inside those platforms. Outdoor map support helps user to visualize their data easily by just viewing on the map, rather than just displaying coordinates or addresses.

Indoor map support means user can view the interior floor plan of a certain building, house, and factory. The visualization platform can visualize the indoor floor plan and allows you to interact with it. From the five platforms that I have reviewed, only one of them has this feature, which is the Atrius Solution Builder. It can visualize any indoor floor map of your choice. The advantages of having indoor map support is so that we can know the exact location of the IoT devices when they go indoor. Figure 2-6-1 below shows an example of how the indoor floor plan of Atrius Solution Builder looks like.

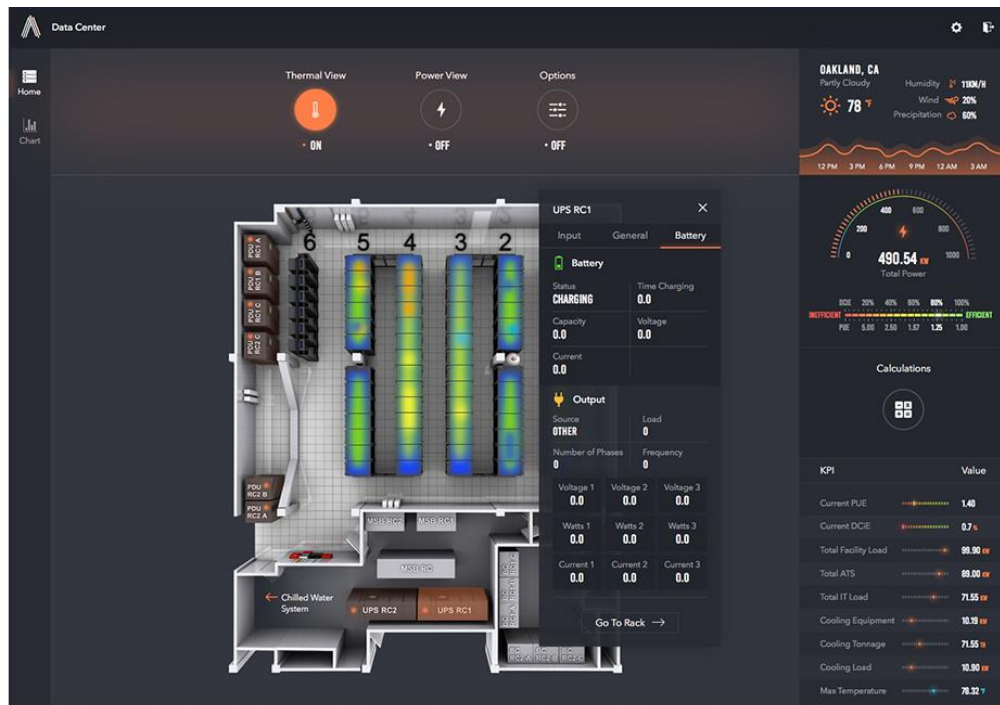


Figure 2-6-1: Atrius Solution Builder’s indoor floor map (Atrius IDE Project Example 2018)

Connectivity is the types of protocol used for uploading or transferring data from IoT devices to the server of the data visualization platform. There are a few types of connectivity that are widely used by several visualization platforms, which are MQTT and REST. These connectivity protocol enables IoT devices to connect and exchange data with the platform. It also creates open doors for more straightforward coordination of the physical world into computer-based framework. Most of the five reviewed visualization tools use MQTT and REST while IBM Watson Analytics also uses FTP for user to upload files manually.

Interactive visualization basically means the interaction between humans and computers generated graphic illustrations of information. There are two criteria that a visualization needs to satisfy to be considered as interactive, which are human input and response time. Human input means that the visualization can allow human to control or alter some aspect of the information of the visual representation. Response time means any changes made by the human must be updated in the visualization, which is also called real-time task.

The fifth criteria from the comparison table above is cloud and on-premise. It is important to distinguish whether a data visualization platform is running on cloud or on-premise. The main differences between cloud and on-premise is that cloud is accessed via the internet, and it is typically hosted by a third-party vendor (Software, n.d.). On-premise is mostly installed in the user's computer itself. According to Lysakowska (2017), since cloud is accessed via the internet, it requires reliable internet access, while on-premise itself does not rely on the internet. It is accessible via its own private WAN connections. One of the good things about using cloud is that it is quick for deployment. Unlike on-premise, where a longer implementation time is usually needed. Cloud service is affordable as it is a model with low total cost of ownership and charges are by the "pay as you go" payment model. For on-premise, you need high cost to develop the whole system in order to use it. However, there are some advantages for on-premise. One of them is it is usually more secure compare to the cloud. Since cloud is hosted in a third-party vendor, if security issues happen to that cloud service provider, data might be compromised. Therefore, it is important to select a well-established cloud service provider to mitigate such risk.

The last criteria from the comparison table above is mobile sensor, which means sensor that can move around rather than placed at a static location. After reviewing several data visualization tools, most of them does not support sensors or

devices that are mobile. It is an important criterion to be included in the IoT visualization platforms as devices are not necessarily stationary at a certain location. They might be moving around to different locations. This also means that the IoT visualization platform must be able to update the real-time location of those mobile devices. By having these functions available, the platform will be able to collect and visualize data collected from the mobile devices from different location points.

In summary, after reviewing and comparing several existing IoT data visualization platforms, a few important criteria for this project have been concluded to solve the problem statements. The first problem statement is the existing IoT visualization platforms do not support mobile IoT sensor nodes, as proven from the reviews above. Hence, one of the most important criteria is to be able to support mobile sensors. Besides, according to the second problem statement, indoor map support for existing visualization platforms is not sufficient to visualize mobile sensors data, which is also proven in the above reviews. Therefore, visualization for real-time location on an indoor map is needed, which means we need an indoor map support that is able to visualize where the mobile sensors are currently located at.

## CHAPTER 3: SYSTEM DESIGN

The overall system that is developed for this project consists of a robot, a sensor, a cloud server and data visualization tool. As this project focus on building an extension for mobile IoT sensors, the robot is used to carry the sensor for it to move around. The sensor sends data to the cloud server using MQTT protocol. The cloud server then collects all the data sent by the sensor from different checkpoints and store them accordingly. While the cloud server is collecting and storing data, the data visualization platform gets the data collected by the cloud server using REST API and visualize it for display. To be able to visualize the data collected from the mobile sensor, an extension to support mobile IoT sensor nodes is needed in the system, which is the main scope of this project.

### 3.1 System Design Diagrams

#### 3.1.1 System Overview Diagram

Figure 3-1-1 below illustrates the overview of the system as described in Section 3.2 above.

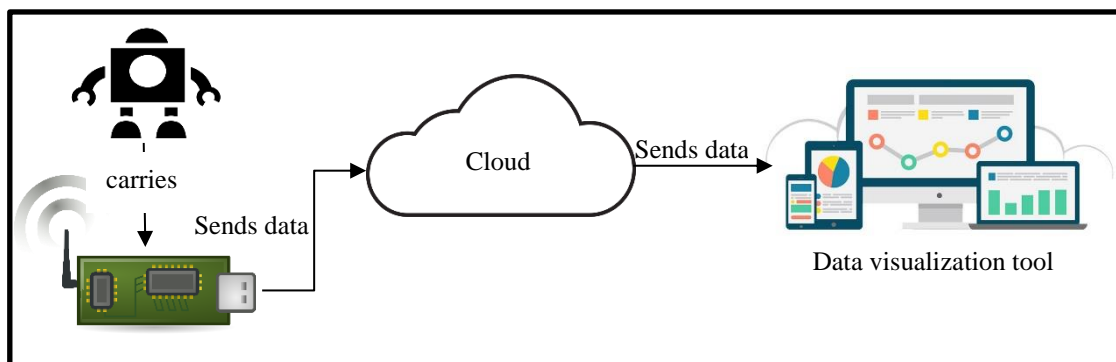


Figure 3-1-1: System Overview

### 3.1.2 Use Case Diagram

Figure 3-1-2 below shows the Use Case Diagram of the system. It illustrates what the user can do on the visualization and how the mobile sensors interact with the cloud server.

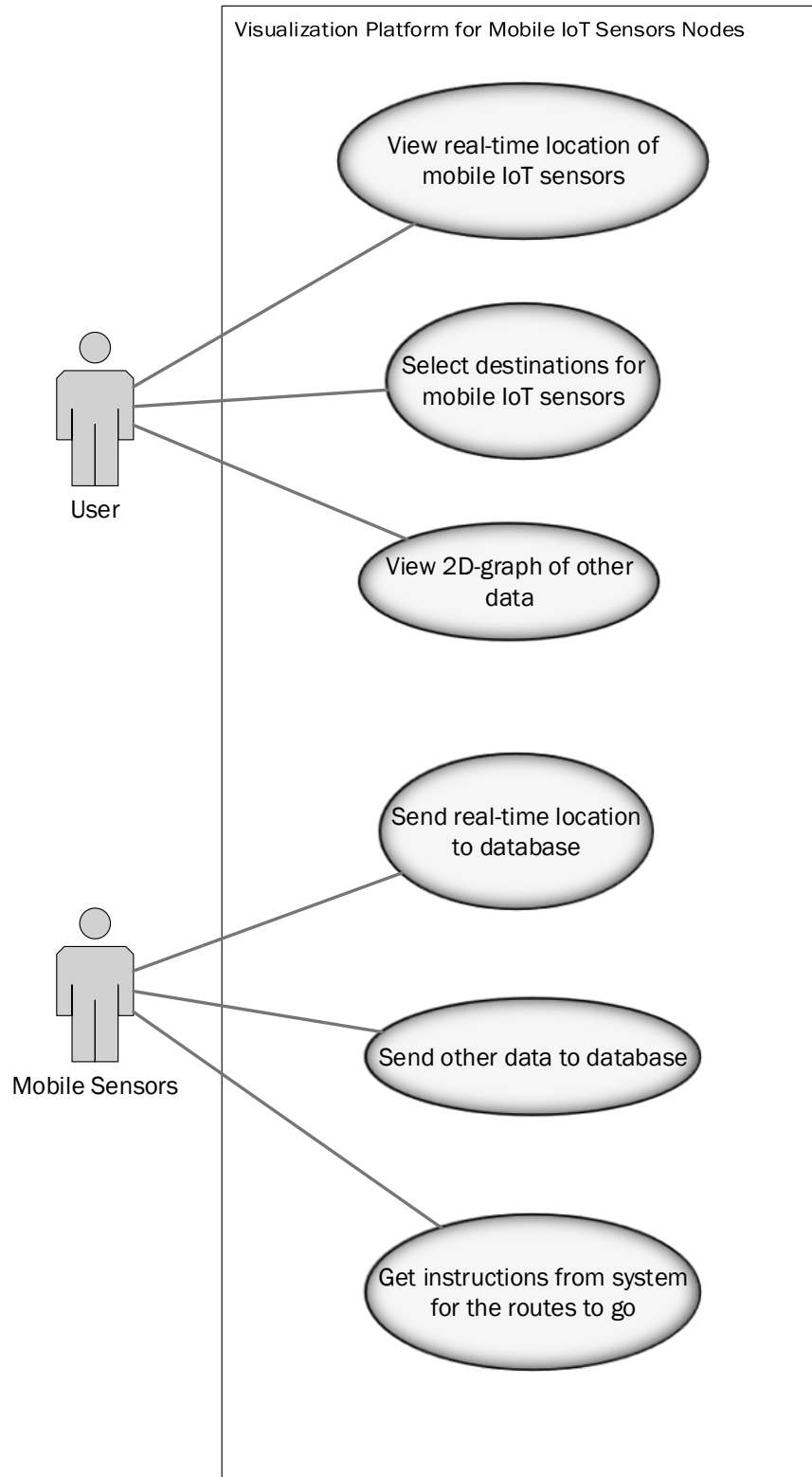


Figure 3-1-2: Use Case Diagram

### 3.1.3 Use Case Description

Tables 3-1-3-1 to 3-1-3-6 are the Use Case Description based on the Use Case Diagram as previously shown in Figure 3-3-2.

Name	View real-time location of mobile IoT sensors
ID	UC001
Level	High
Actor	User
Stakeholders and Interests	User – wants to view the real-time location of mobile IoT sensors
Description	This use case describes how the user view the real-time location of mobile IoT sensors
Trigger	When wanted to view real-time location of mobile IoT sensors
Type	External

Table 3-1-3-1: UC001 Use case description

Name	Select destinations for mobile IoT sensors
ID	UC002
Level	High
Actor	User
Stakeholders and Interests	User – wants to select destinations for mobile IoT sensors
Brief Description	This use case describes how the user decide which destinations the mobile IoT sensors go
Trigger	When wanted to give instructions to mobile IoT sensors
Type	External

Table 3-1-3-2: UC002 Use case description

Name	View 2D-graph of other data
ID	UC003
Level	High
Actor	User
Stakeholders and Interests	User – wants to view other data in a 2D-graph
Description	This use case describes how the user view other data collected from the mobile IoT sensors in a 2D-graph
Trigger	When wanted to view other data collected by mobile IoT sensors
Type	External

Table 3-1-3-3: UC003 Use case description

Name	Send real-time location to database
ID	UC004
Level	High
Actor	Mobile Sensors
Stakeholders and Interests	Mobile Sensors – wants to send its real-time location to the database
Description	This use case describes how the mobile sensors send its real-time location to the database
Trigger	When wanted to send real-time location to the database
Type	External

Table 3-1-3-4: UC004 Use case description



Name	Send other data to database
ID	UC005
Level	High
Actor	Mobile Sensors
Stakeholders and Interests	Mobile Sensors – wants to send other data to database
Description	This use case describes how the mobile sensors send other data collected such as air humidity and temperature to the database
Trigger	When wanted to send other collected data to the database
Type	External

Table 3-1-3-5: UC005 Use case description

Name	Get instructions from system for which route to go
ID	UC006
Level	High
Actor	Mobile Sensors
Stakeholders and Interests	Mobile Sensors – wants to get instructions from the system about the route to go
Description	This use case describes how the mobile sensors get instructions from the system for which routes to go
Trigger	When wanted to get instructions from system
Type	External

Table 3-1-3-6: UC006 Use case description

### 3.1.4 ER Diagram

Figure 3-1-4 below shows the ER diagram with seven tables which are also constructed in the database in the cloud. These seven tables are used to store all the important data that are needed in the system.

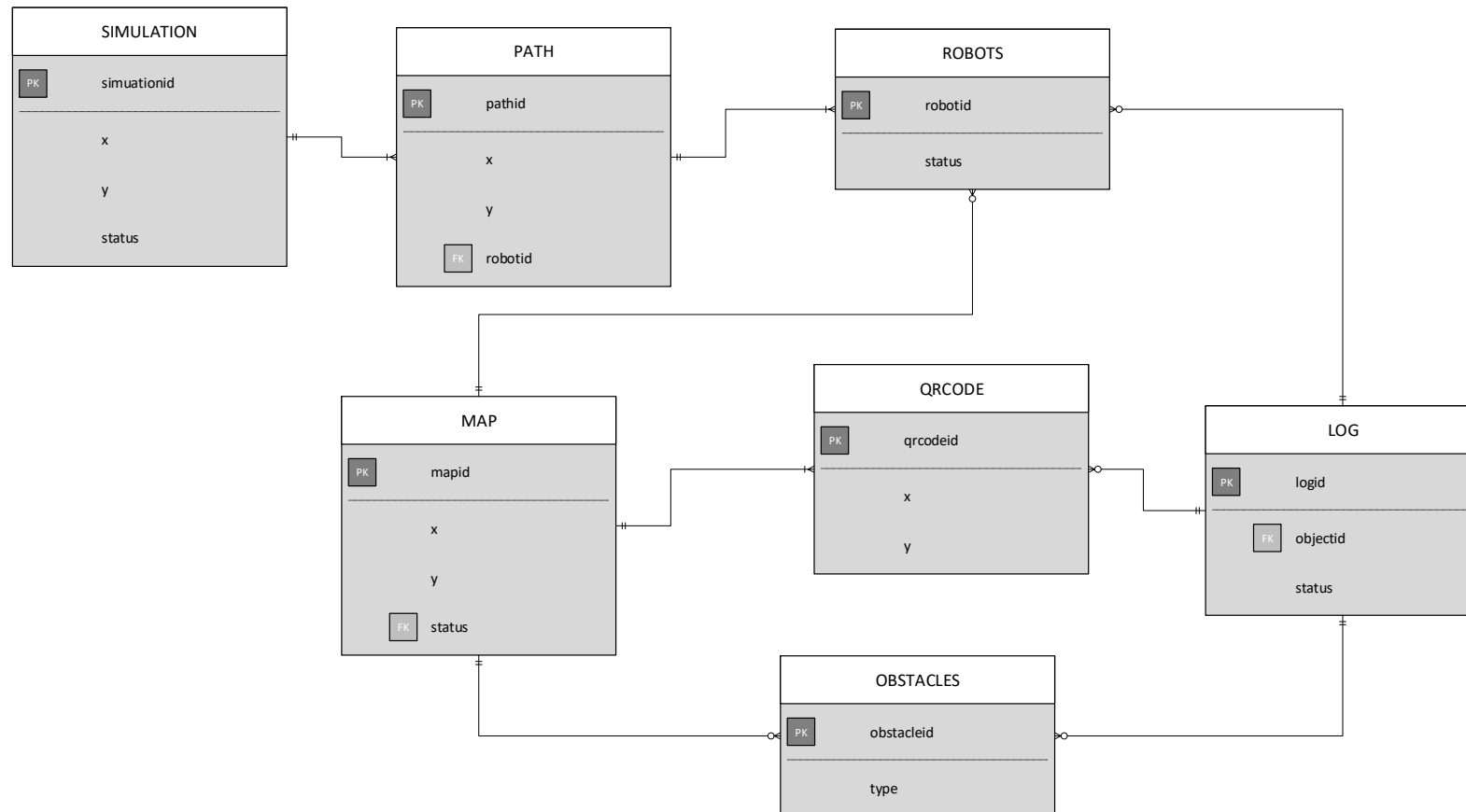


Figure 3-1-4: ER Diagram

### 3.1.5 Class Diagram

Figure 3-1-5 below is the Class Diagram that illustrates the structure of the system of this project.

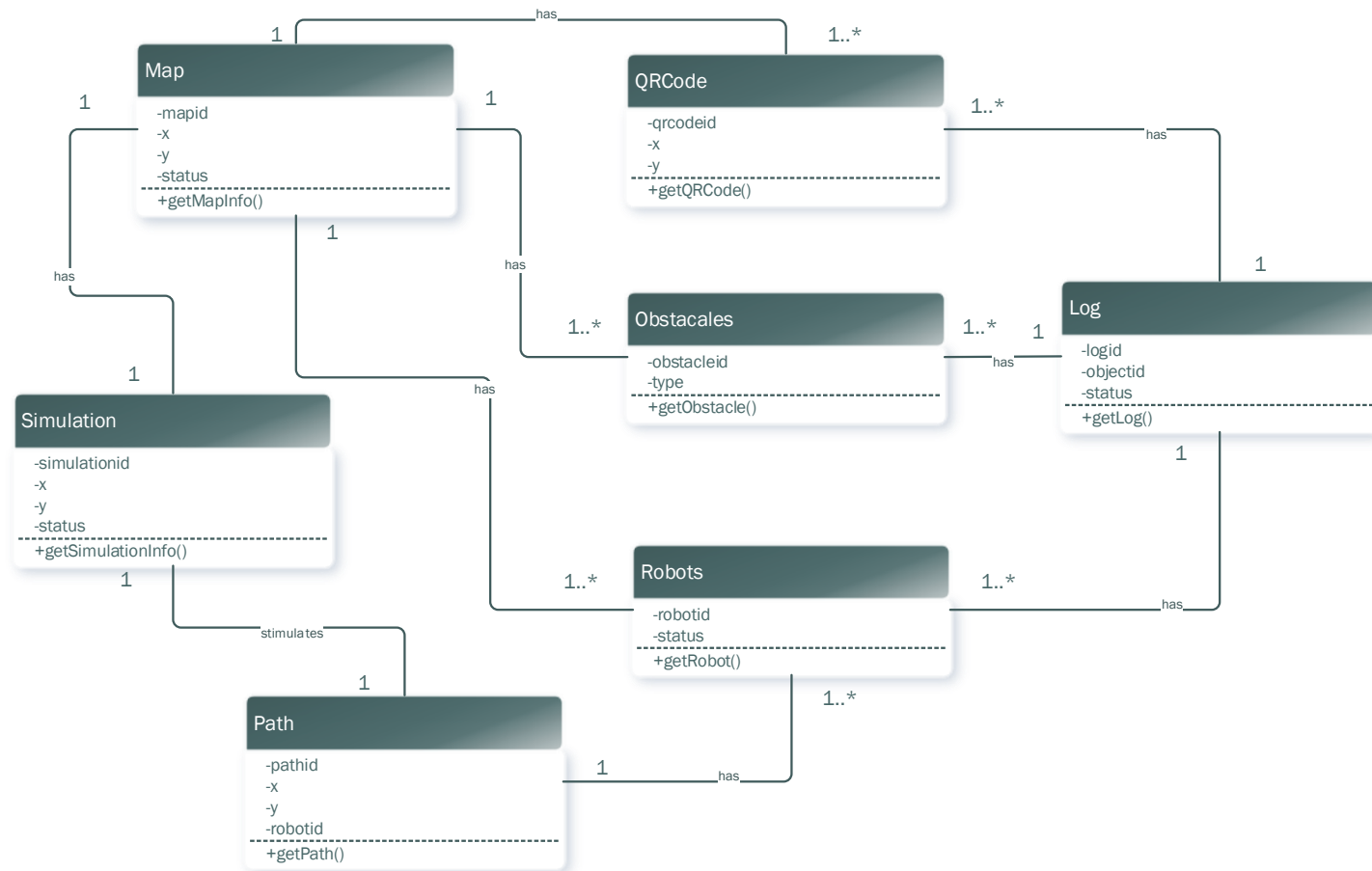


Figure 3-1-5: Class Diagram

### 3.1.6 Sequence Diagram

Figure 3-1-6 below shows the sequence diagram of how the user interact with the system interface and how the system get data out from the database. It is arranged in time sequence accordingly.

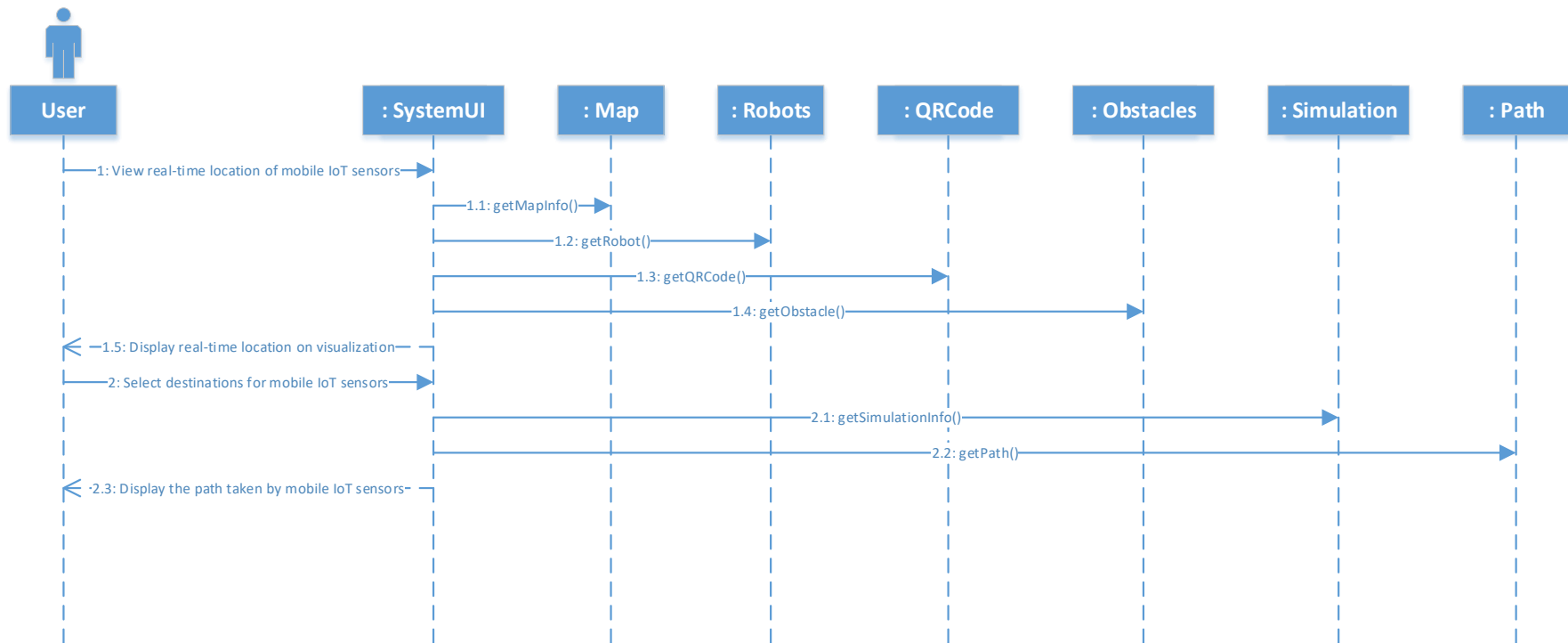


Figure 3-1-6: Sequence Diagram

### 3.1.7 Communication Diagram

Figure 3-1-7 below is the communication diagram which illustrates the interaction between objects in the system.

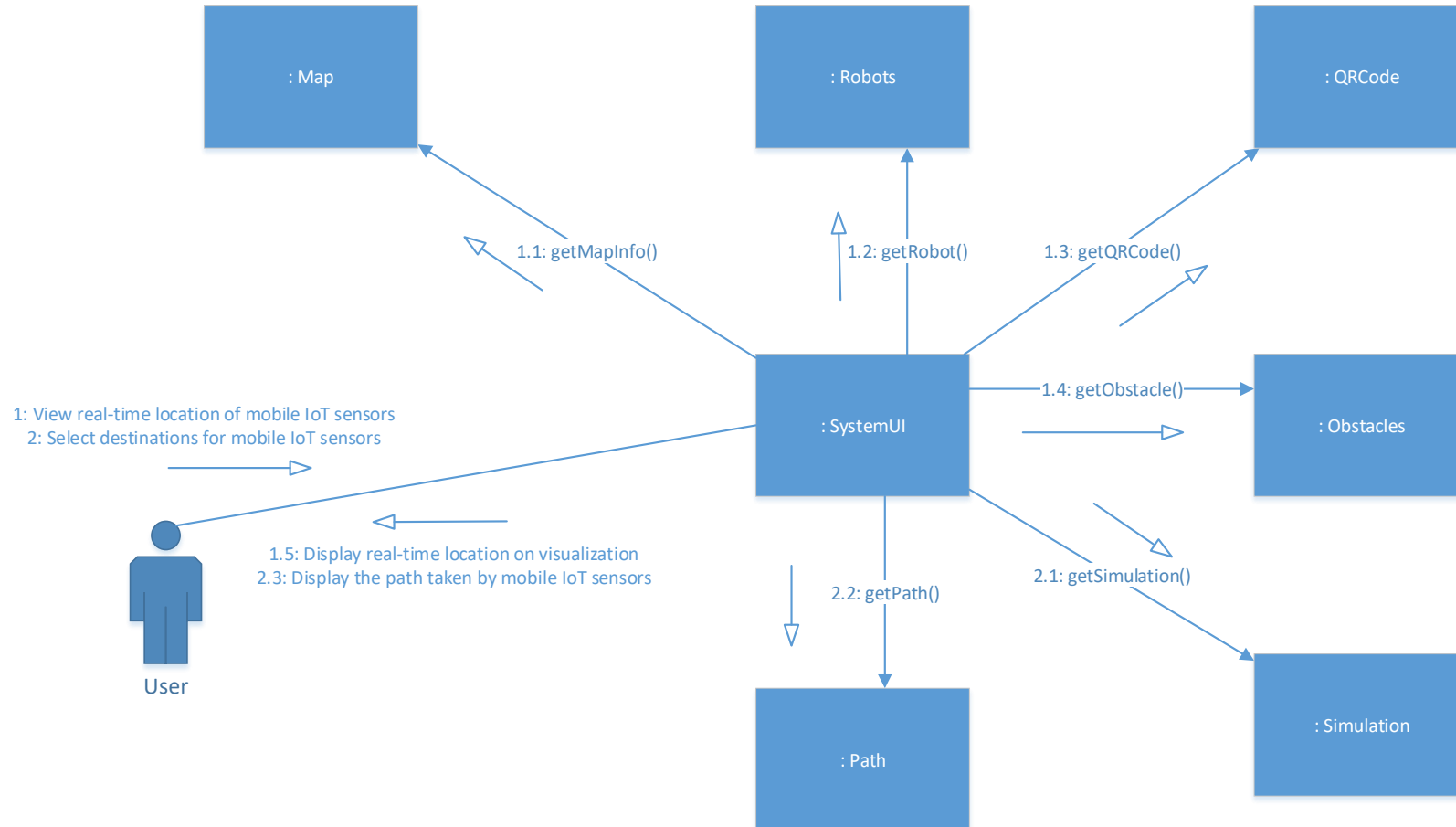


Figure 3-1-7: Communication Diagram

## **CHAPTER 4: IMPLEMENTATION**

### **4.1 Methodology**

Based on the proposed method, the implementation phase focuses on producing prototypes. In FYP1, the first prototype was produced with the following features:

1. A database with few tables that are used to record the real-time location and other data recorded from the mobile IoT sensors.
2. A visualization platform that can display an indoor mapping with the precise real-time location of the mobile IoT sensors.

Therefore, in FYP2, the second and final prototypes are done with all remaining features done, which includes:

1. The system will be able to send instructions to the mobile IoT sensors of which route to go.
2. The system will be able to calculate a best route for the mobile IoT sensors without colliding with other sensors and obstacles.
3. A 2D-graph that can display the other data collected from the mobile IoT sensors such as temperature and air humidity.

All the features above are combined together and produce the final product.

### **4.2 Tools**

#### **4.2.1 Platform**

##### **◆ Amazon AWS EC2**

The cloud platform used in this project is the Amazon Elastic Compute Cloud (Amazon EC2) which act as the web service, server and database for the system.

#### **4.2.2 Programming Language**

##### **◆ HTML**

Hypertext Markup Language (HTML) is used to develop the extension of visualization framework in a web page.

◆ **CSS**

Cascading Style Sheets (CSS) is used together with HTML to further enhance the style of the visualization framework.

◆ **JavaScript**

JavaScript (JS) is also used along with HTML and CSS that program the indoor mapping that is visualized on the web page done by HTML and CSS.

◆ **PHP**

Hypertext Preprocessor (PHP) is used to enable communication between cloud server and visualization framework.

◆ **Python**

Python is used to enable communication between the mobile IoT sensors and cloud server. It is also used to do calibration on the mobile IoT sensors while it moves.

#### 4.2.3 Software Tools

◆ **MobaXterm**

It is used to connect to the AWS EC2 platform to configure and deploy the server. It is also used to remotely connect to the Raspberry Pi on the mobile IoT sensors through SSH connection.

◆ **VNC Viewer**

Same as MobaXterm, VNC Viewer is also used to remote access to the Raspberry Pi on the mobile IoT sensors.

#### 4.2.4 Hardware Tools

◆ **Mobile Robots (AGVs)**

Two mobile robots are used in this project, which are the omni wheel robots.



Figure 4-2-4-1: Omni wheel robot

◆ **Air temperature and humidity sensor**

It is attached on the mobile robots and used to collect air temperature and humidity data.

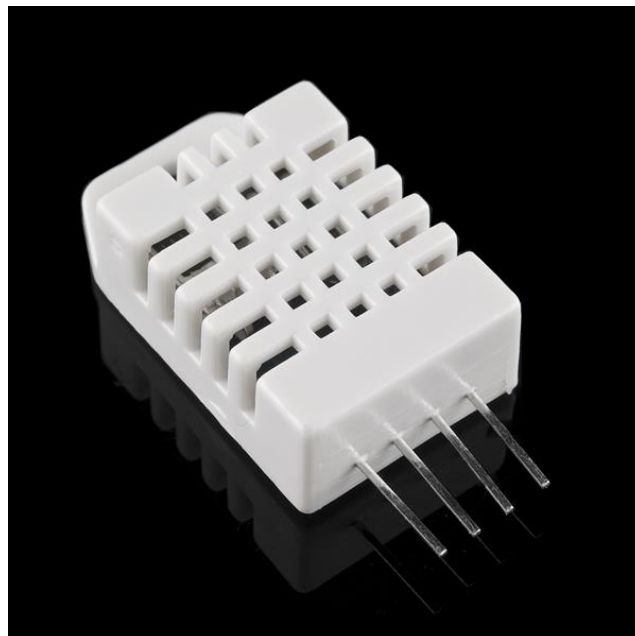


Figure 4-2-4-2: Air temperature and humidity sensor



### 4.3 Implementation Phases

#### 4.3.1 Setting up AWS EC2

An AWS EC2 instance is created to serve as the cloud server to host the database and act as the web service for the visualization framework. After creating an EC2 instance, MySQL database is set up to store all the data needed which are the indoor map coordinates, the real-time location of the robot and the data collected from the sensor which is attached on the mobile robots.

#### 4.3.2 Developing Visualization of Indoor Map

After the cloud platform are set up successfully, the next step is developing the visualization of indoor mapping. The web page is created by using HTML, CSS and JavaScript for the programming of indoor mapping. Figure 4-3-2 below shows the completed visualization of indoor map.

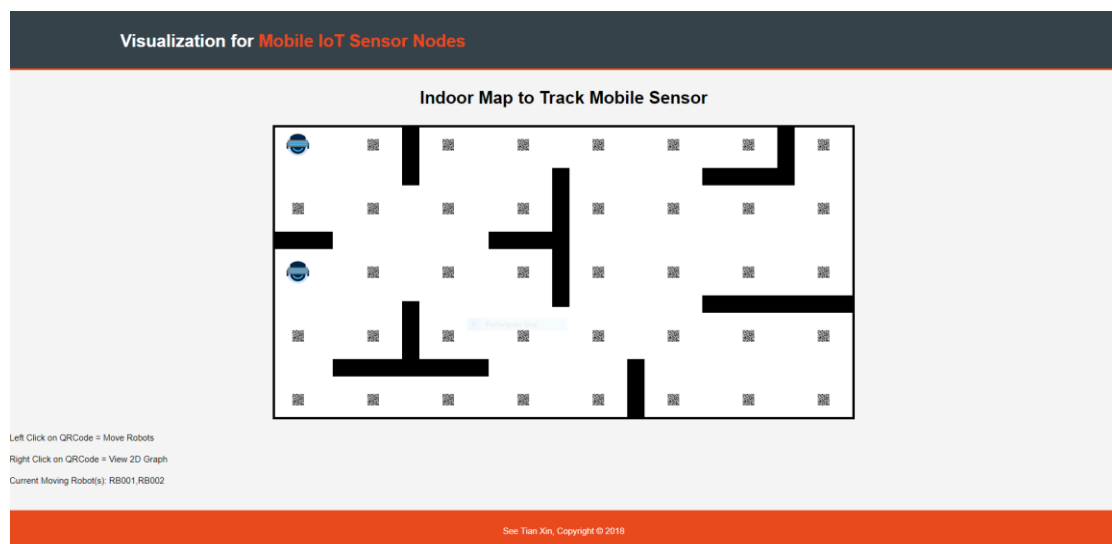


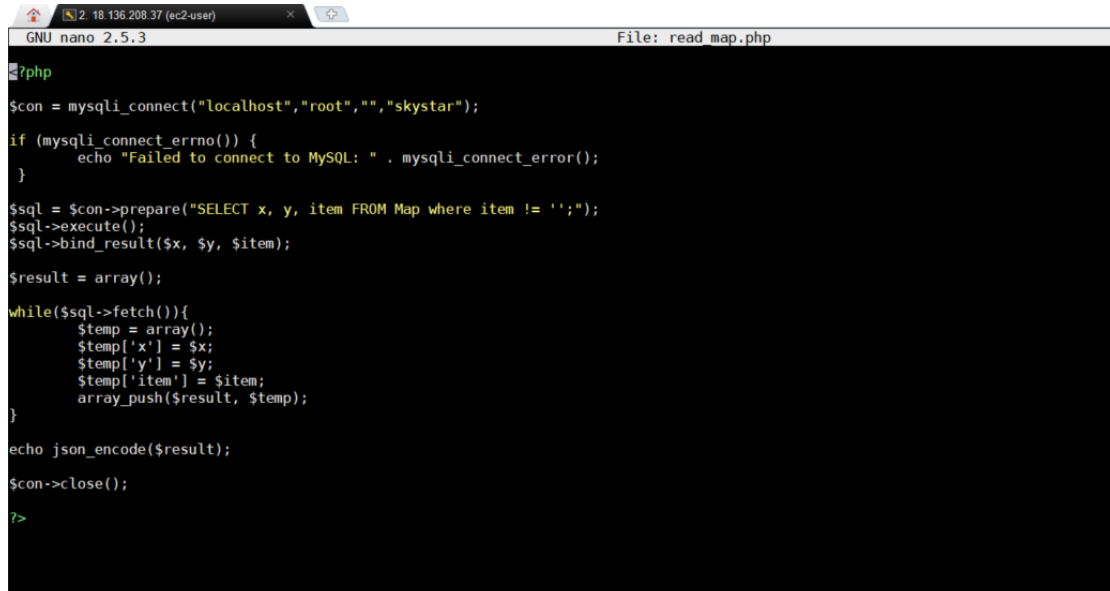
Figure 4-3-2: Visualization for indoor map

#### 4.3.3 Communication between Cloud Server and Visualization Framework

After having both cloud server and visualization framework, the visualization framework need to get data from the server in order to update the real-time location of the robot in its visualized indoor map. Therefore, the communication between visualization and cloud server is needed.

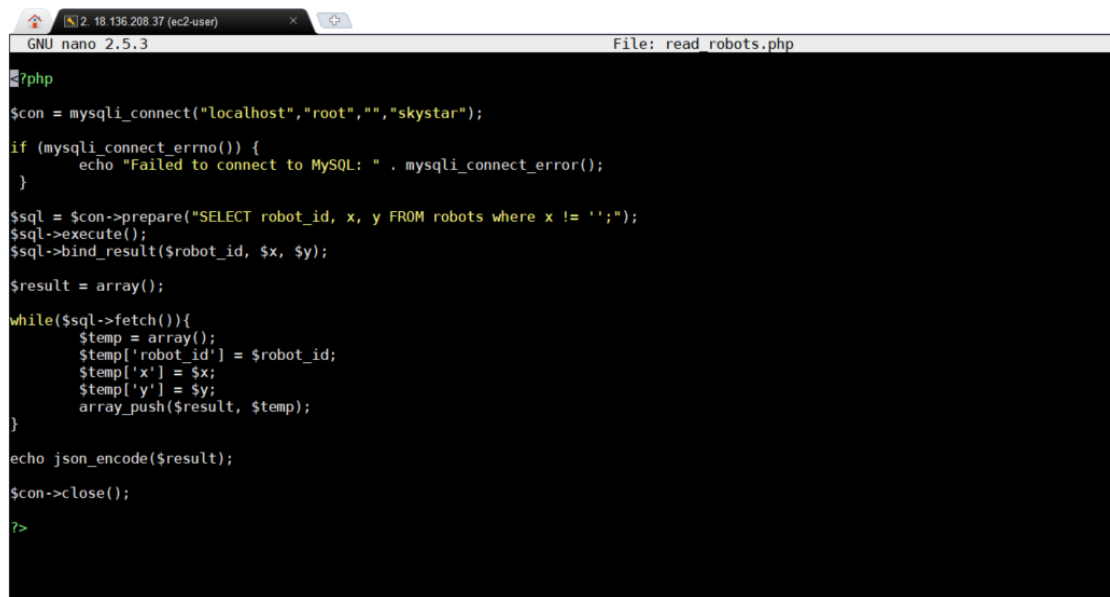
PHP scripts are written to enable communication between the visualization framework and cloud server. It helps the visualization framework to get the indoor map coordinates and real-time location of the moving robots from the database server

and return it to the framework. The visualization then updates the indoor map and moving robots' location according to the data it gets from the database server. Figures 4-3-3-1 and 4-3-3-2 below show the PHP script written to pull data from the database server.



```
GNU nano 2.5.3 File: read_map.php
?php
$con = mysqli_connect("localhost","root","","skystar");
if (mysqli_connect_errno()) {
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
$sql = $con->prepare("SELECT x, y, item FROM Map where item != ''");
$sql->execute();
$sql->bind_result($x, $y, $item);
$result = array();
while($sql->fetch()){
    $temp = array();
    $temp['x'] = $x;
    $temp['y'] = $y;
    $temp['item'] = $item;
    array_push($result, $temp);
}
echo json_encode($result);
$con->close();
?>
```

Figure 4-3-3-1: PHP script to get indoor map coordinates from database



```
GNU nano 2.5.3 File: read_robots.php
?php
$con = mysqli_connect("localhost","root","","skystar");
if (mysqli_connect_errno()) {
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
$sql = $con->prepare("SELECT robot_id, x, y FROM robots where x != ''");
$sql->execute();
$sql->bind_result($robot_id, $x, $y);
$result = array();
while($sql->fetch()){
    $temp = array();
    $temp['robot_id'] = $robot_id;
    $temp['x'] = $x;
    $temp['y'] = $y;
    array_push($result, $temp);
}
echo json_encode($result);
$con->close();
?>
```

Figure 4-3-3-2: PHP script to get moving robots' location from database

### 4.3.4 Calculation of Best Route for Mobile IoT Sensors

The algorithm used to calculate the best route for mobile IoT sensors to move from one point to another is the Depth First Search (DFS) algorithm. DFS is a recursive algorithm that implements exhaustive searches that moves until the very end

of the current path, and then backtrack on the same path to search for nodes. The algorithm is developed using the PHP script which gets the request from user selecting the destination point for the moving robots. Figure 4-3-4 below shows the user selecting a QR Code which is the destination point for the robot 'RB001' to move to.

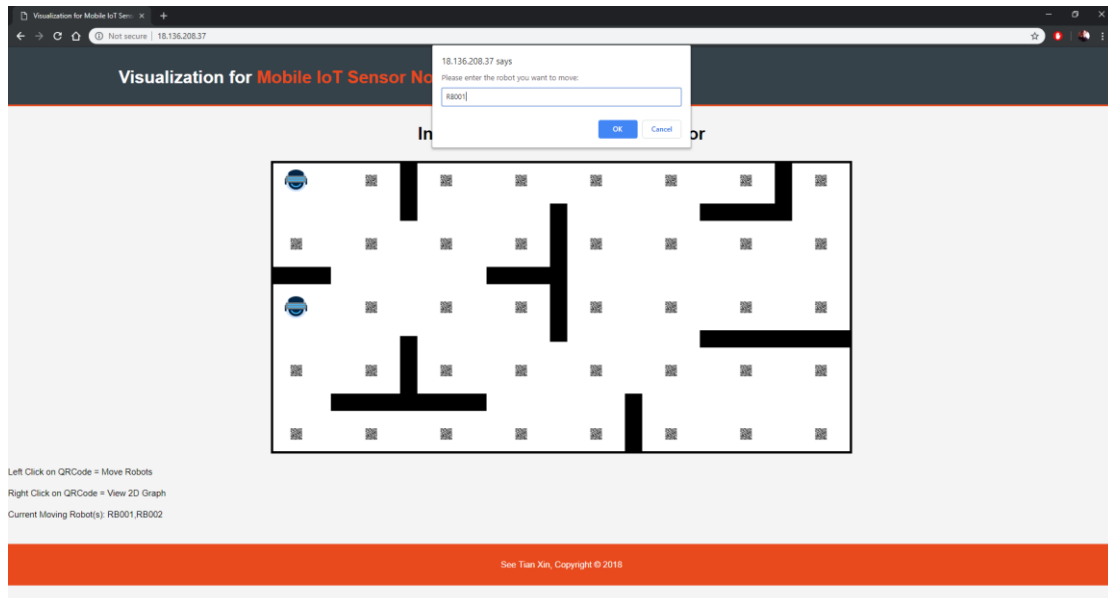


Figure 4-3-4: User select the destination point to move for moving robots

After the user selects a destination point, the visualization framework triggers the PHP script to calculate the best route for the robots to move to the destination point without colliding with the other moving robots and obstacles. After calculation, the PHP script then update the database server with the route it calculated.

### 4.3.5 Developing 2D Line Graph

A 2D line graph to visualize the data collected from the sensors which are fixed on the mobile robots are developed by using Chart.js, the open source HTML5 charts. Figure 4-3-5 below shows the 2D line chart of the air temperature and humidity collected from the sensor.

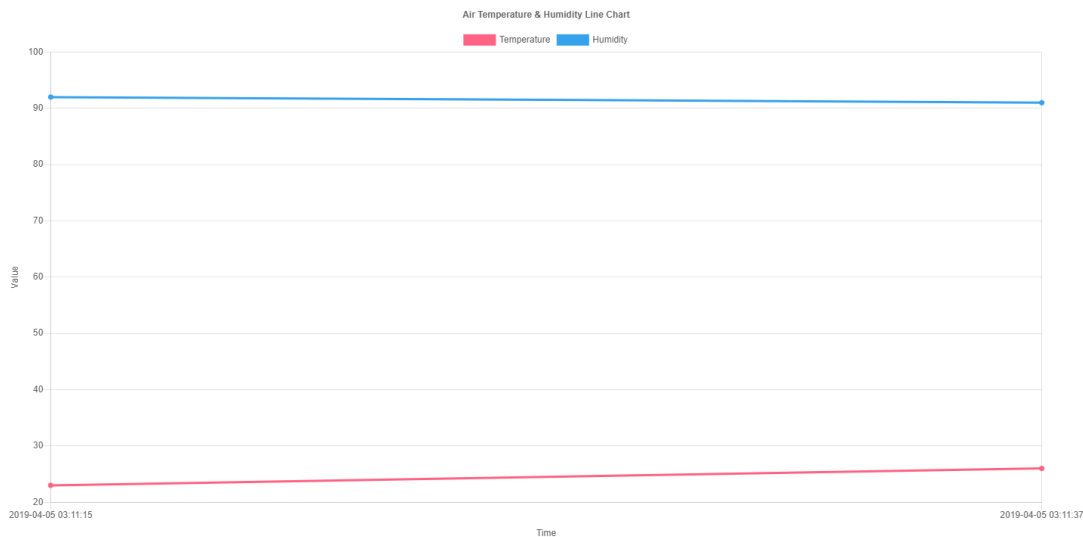


Figure 4-3-5: Air Temperature and Humidity Line Chart

#### 4.3.6 AGV Calibration

The calibration of the moving robots (AGV) is done by using the QR code as the correction point. The calibration is done to ensure that the AGV is moving in a straight line. When the AGV reaches a QR code, it scans the QR code by using the Pi camera attached on the AGV. Besides sending data to the cloud server, it gets the coordinate of the top left point of the QR code as the point to do calibration. The python code of the calibration is shown in Appendix A. Figure 4-3-6 below shows the corner of the QR code that the AGV takes as the correction point.

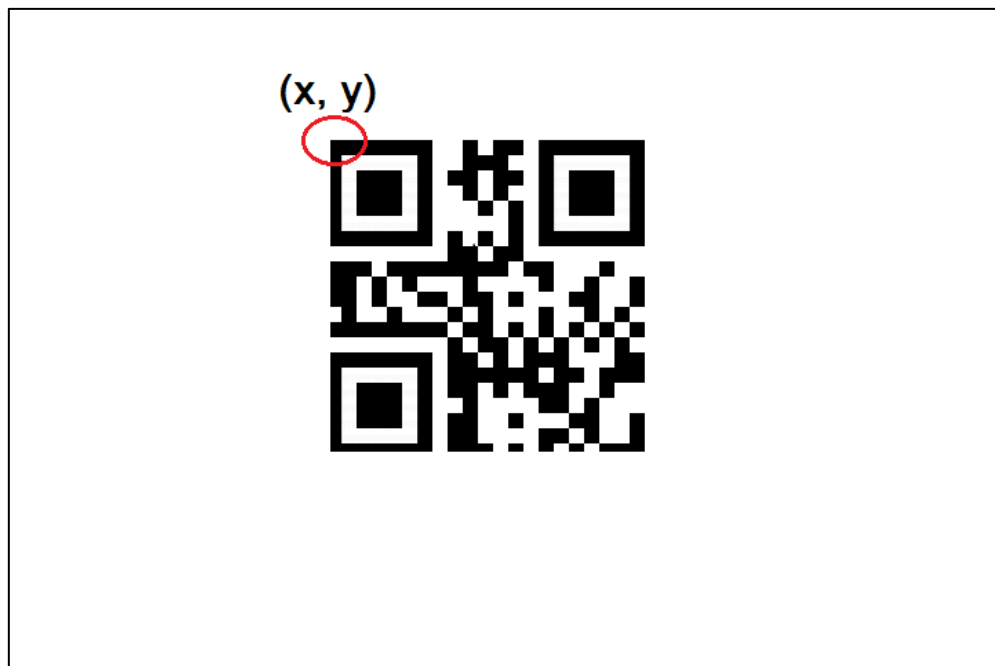


Figure 4-3-6: QR code's top left corner as the correction point

#### **4.3.7 Two-way Communication between Mobile IoT Sensors and Cloud Server**

To enable two-way communication between mobile IoT sensors and cloud server, PHP scripts and Python codes are needed to be developed. The Python codes are written in the Raspberry Pi that connects with the mobile IoT sensors which can request, get and send data to the cloud server. The PHP scripts are written in the cloud server to get the requests and data from the mobile IoT sensors and send back the data requested.

## CHAPTER 5: TESTING

### 5.1 Testing between Visualization Framework and Mobile IoT Sensors

#### 5.1.1 Real-time Location Update Test

The first test to conduct is the real-time location update of the mobile IoT sensors on the indoor map visualization. In the test conducted, the mobile IoT sensor is moved from one QR code to another while updating its real-time location by capturing the QR code and send its location to the database server. The visualization then gets its real-time location from the database server and reflects the real-time location update on the visualized indoor map. Figure 5-1-1-1 below shows the original location of the mobile IoT sensor and figure 5-1-1-2 shows the real-time location on the visualized indoor map after the actual mobile IoT sensors move forward to the next QR code.

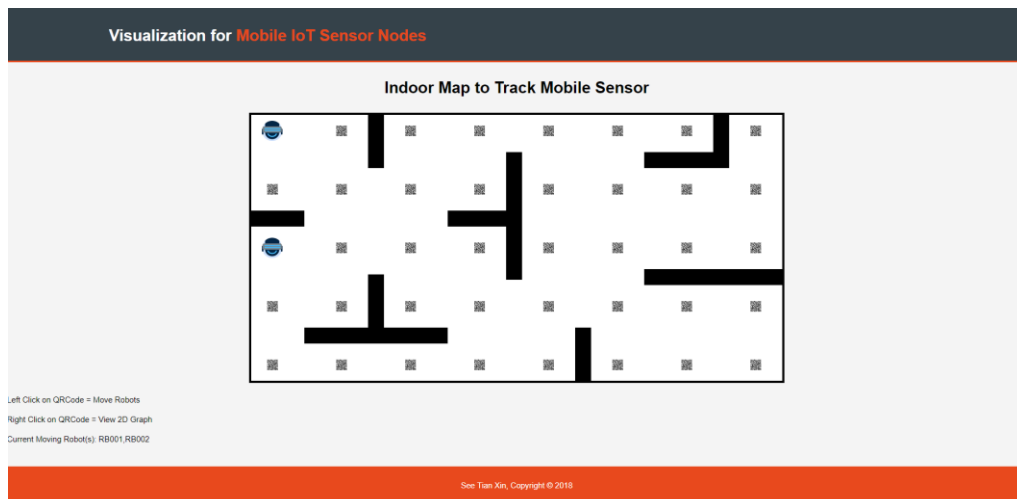


Figure 5-1-1-1: Original Location of Moving Robot

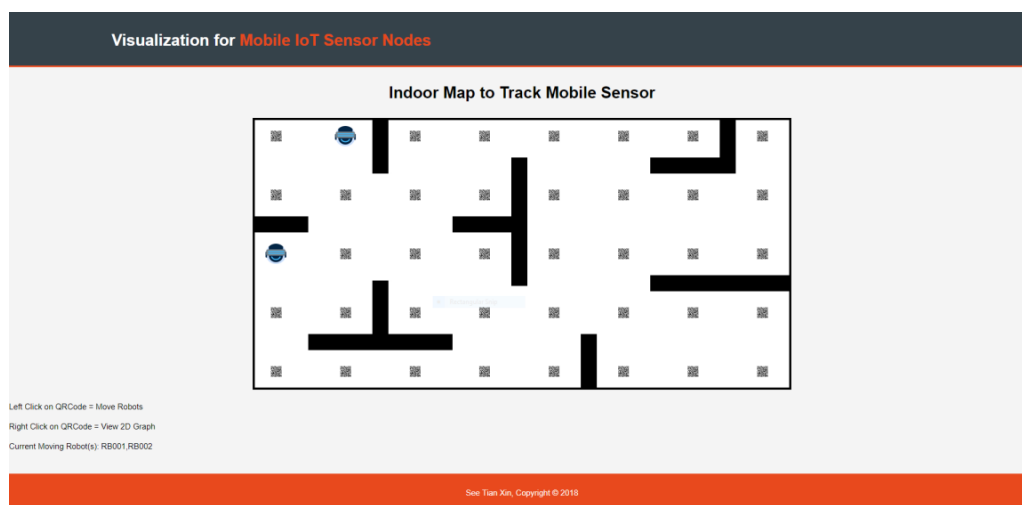


Figure 5-1-1-2: Updated Location of Moving Robot 1

## Chapter 5: Testing

Next, the mobile IoT sensor continues to move in a square before going back to its original location. Figure 5-1-1-3, 5-1-1-4 and 5-1-1-5 show the real-time location updates of the visualization of how the mobile IoT sensor moves back to its original location.

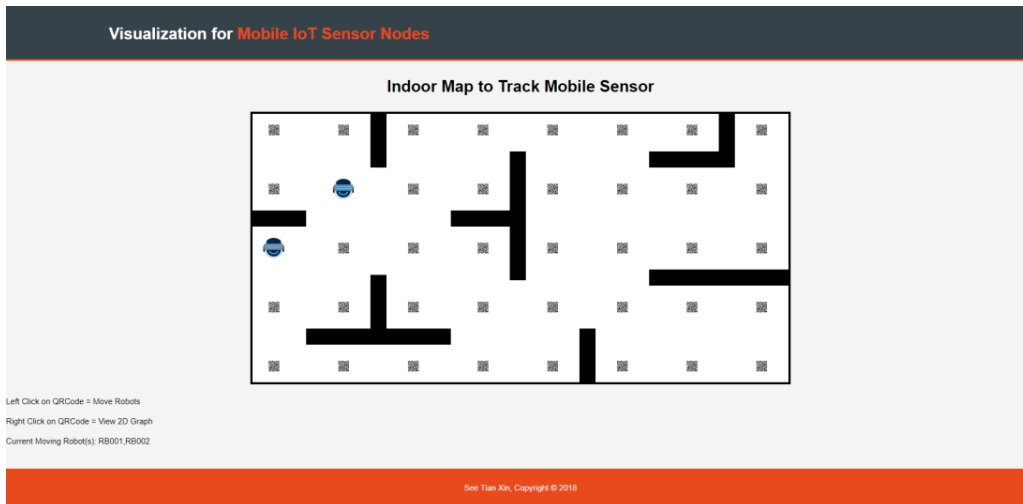


Figure 5-1-1-3: Updated Location of Moving Robot 2

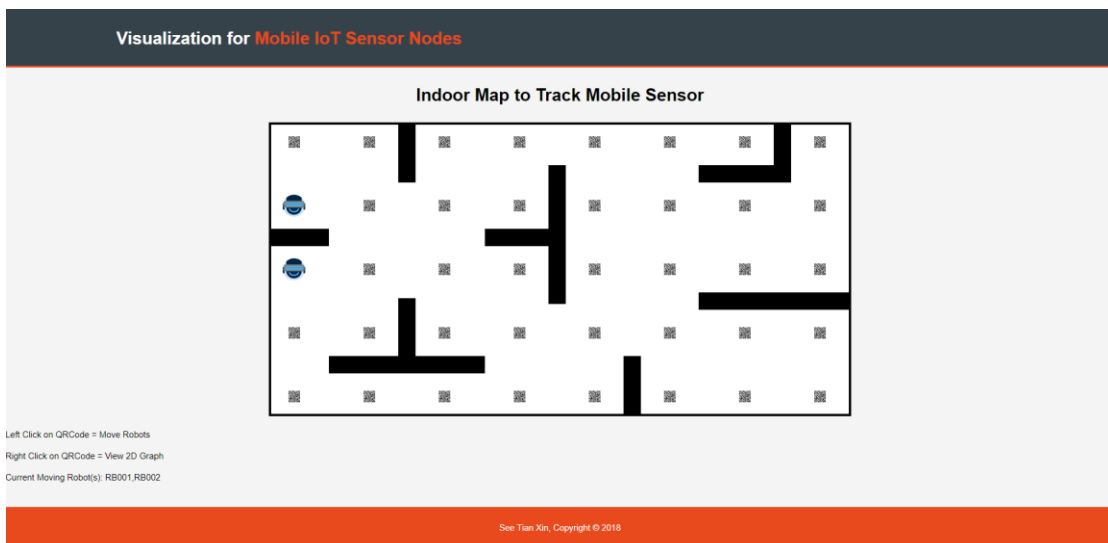


Figure 5-1-1-4: Updated Location of Moving Robot 3

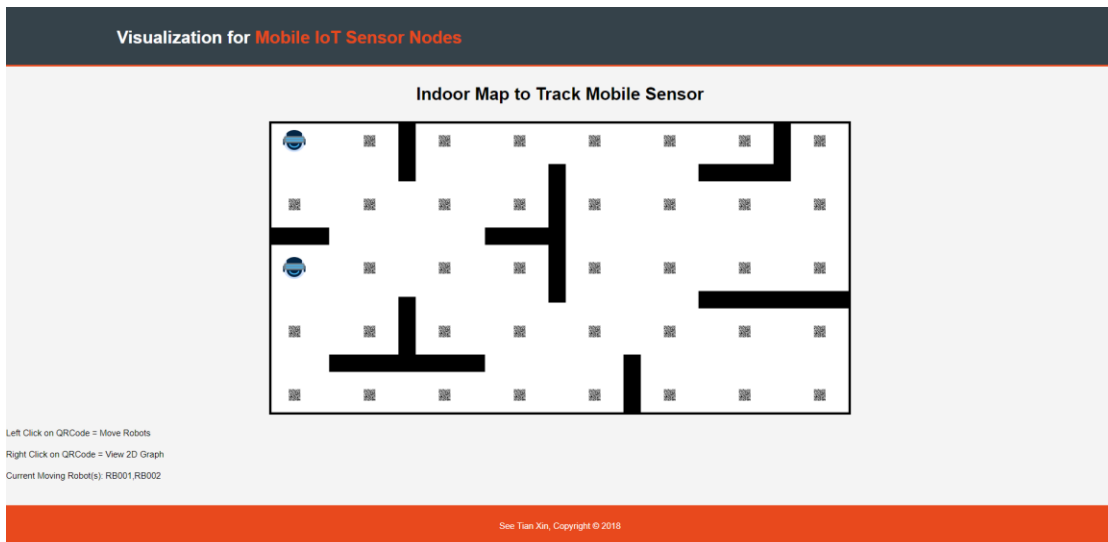


Figure 5-1-1-5: Moving Robot back to its Original Location

From this test, it is proven that the visualization framework can reflect the real-time location update of the real-life moving robots.

### 5.1.2 Best Route Calculation Test

In this test, the best route calculation is conducted to prove that it can send its calculated route to the real-life moving robot and instruct the robot to move according to the route calculated. For the testing, a destination point is selected for a mobile IoT sensor. Figure 5-1-2-1 shows the selected destination point circled in the visualization.

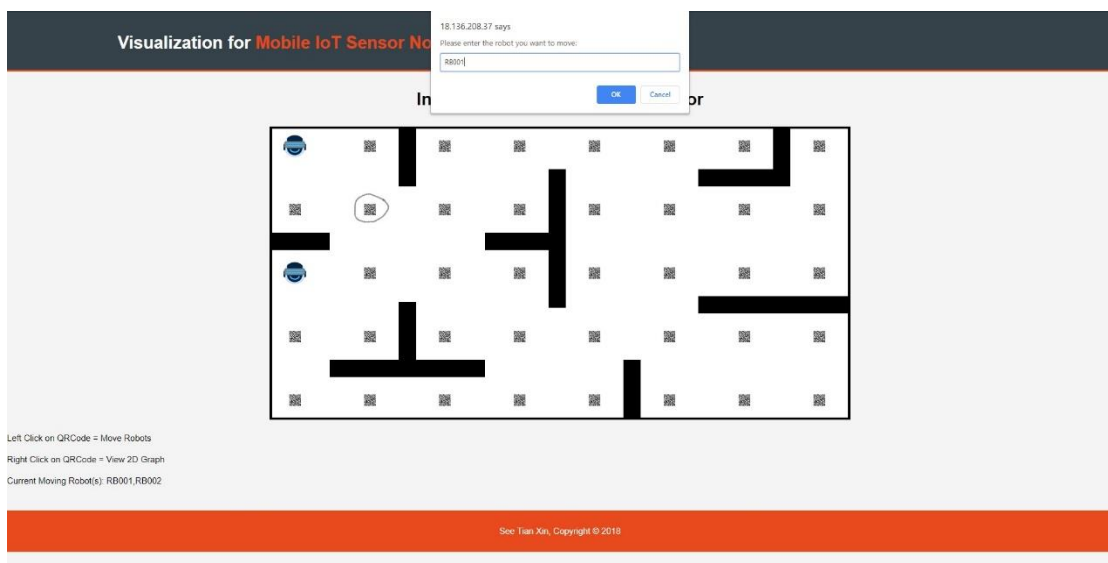


Figure 5-1-2-1: Selected destination point for moving robot RB001



After pressing the OK button on the prompt, a PHP script is triggered and the calculation for the best route is started. After done with the calculation, the calculated route is updated to the database server. The mobile IoT sensor then get the instructions of the route from the database server and execute the instructions. The calculated route is shown in the figure below.

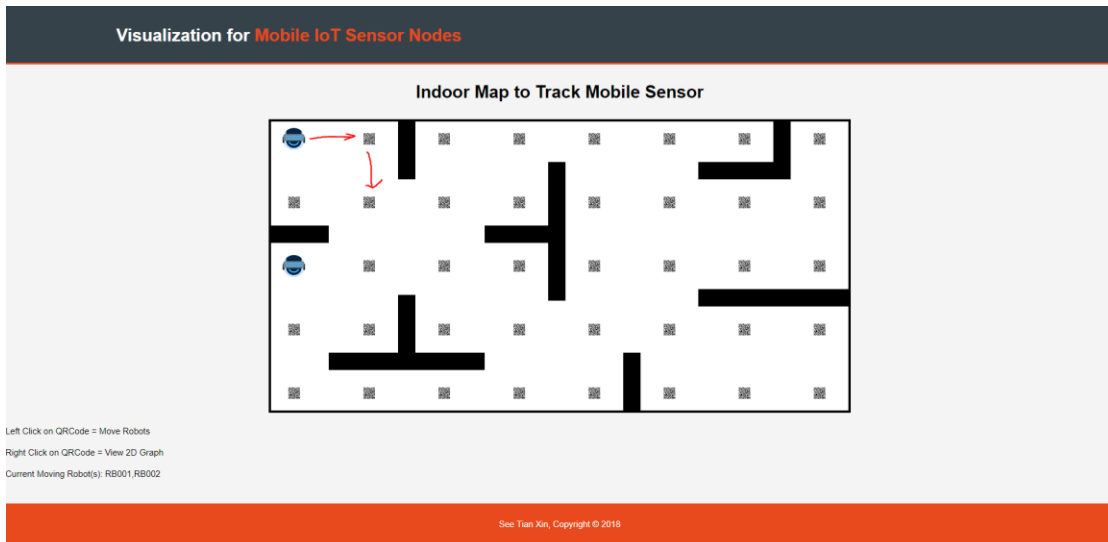


Figure 5-1-2-2: Calculated best route

The mobile IoT sensor then moved according to the calculated route. Figure 5-1-2-3 and 5-1-2-4 show how the mobile IoT sensor moved based on the calculated route.

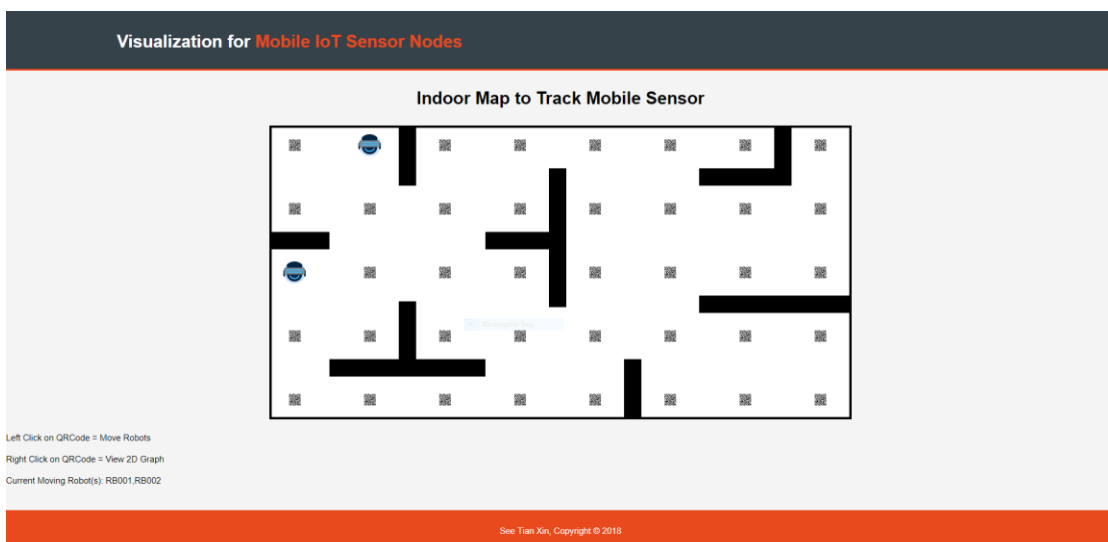


Figure 5-1-2-3: Mobile IoT sensor movement according to instruction

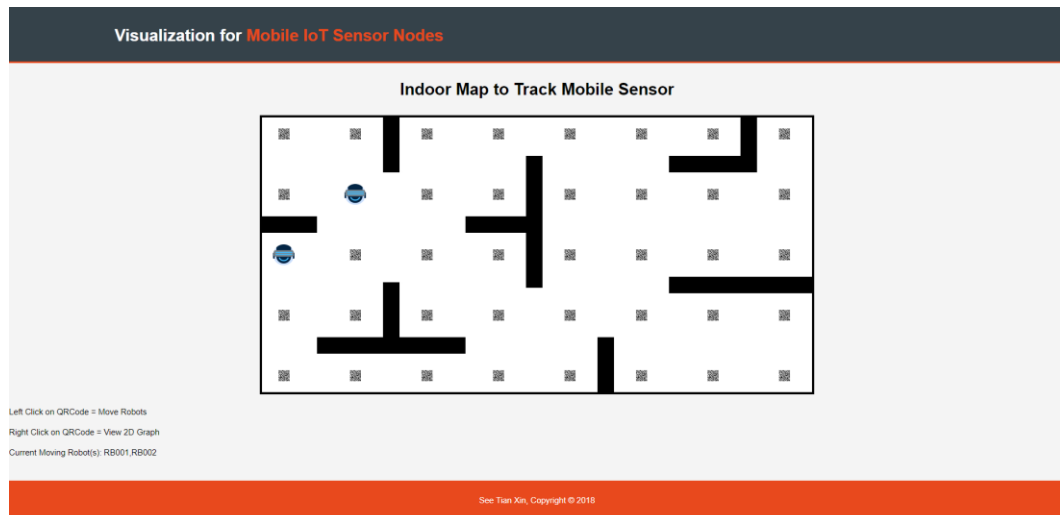


Figure 5-1-2-4: Mobile IoT sensor reached its destination point

From this test, it is proven that the best route calculation can calculate a correct route for the mobile IoT sensor and enable the mobile IoT sensor to move according to the route calculated towards its destination point.

## 5.2 Testing on the AGV

### 5.2.1 QR Code Distance Test

For the test on AGV, how far the QR code can be separated is crucial and needed to be tested. It is because of the AGV used in this project is the omni wheel robot. For the AGV to move in a perfectly straight line, the three wheels need to have perfect coordination. Since the three wheels do not have the perfect coordination, which is the limitation, the QR code is needed for calibration when the AGV moves slightly slanted.

A few distances are tested, which are 30cm, 50cm, 80cm and 100cm. Table 5-2-1 below shows the approximate distance the AGV slanted away when it reached the second QR code.

Distance between QR Codes (cm)	Distance Slanted Away (cm)
30	1.0
50	1.5
80	2.8
100	5.0

Table 5-2-1: Result of QR Code Distance Test

When the distance between two QR codes reaches 100cm, the AGV are slanted slightly further, and it results in the Pi camera attached on the AGV unable to capture the QR code. Therefore, to solve the problem, the next test is carried out.

### 5.2.2 AGV Travelling Speed Test

In this test, the hypothesis is that if the AGV moves from a slower speed to a faster speed, the slanting of the AGV will be reduced. It also solves the problem in Test 5.2.2 above if the hypothesis is true. In order to prove the hypothesis, the distance of two QR codes are set to be 100cm, and the slanting distances are compared between the original speed of the AGV and the modified speed from slower to faster speed. Table 5-2-2 below shows the comparison of the original speed and the modified speed.

Speed of AGV	Distance Slanted Away (cm)
Original speed	5.0
Modified speed (from slower to faster)	1.0

Table 5-2-2: Comparison of the results taken from Travelling Speed Test

From the results above, it is proven that the hypothesis is true, therefore the problem mentioned in Test 5.2.1 is solved with the AGV moving from a slower speed to faster speed, which is an idea taken from how the car moves in real world.

### 5.3 Issues and Challenges

There are a few issues and challenges faced while developing this project. The challenges faced are as below:

1. The accuracy of the real-time location of the mobile IoT sensors.
  - The real-time location displayed on the visualization might have delays with the actual mobile IoT sensors that are moving on the actual world.
2. The calculated routes might not be the best routes.
  - The shortest routes sometimes do not mean it will save the energy resources and time taken to travel from a point to another point.

## **CHAPTER 6: CONCLUSION**

### **6.1 Project Review, Discussions and Conclusions**

Visualization for mobile IoT devices are important as IoT devices that are moving around are used in many areas. Since there are inefficient support for mobile IoT devices, hence the problems are that the existing visualization platforms do not provide full support for mobile IoT devices and the support for indoor mapping is also not sufficient to visualized mobile IoT sensors.

Therefore, this project is proposed based on the problems that are faced, which is to develop an extension which supports the visualization of indoor mapping for mobile IoT sensors. Instead of developing a full visualization tool that has many existing functionalities, this project only focuses on the development of indoor map support for mobile IoT sensors and visualization of mobile sensors data.

In FYP1, the first prototype is being developed, which includes the setup of cloud platform, AWS EC2. The database server and web service are then setup in the cloud server. After that, the next feature that is done is the visualization framework can display the real-time location of several mobile IoT sensors on a visualized indoor map. Whenever the mobile sensors move, the visualized map will reflect on its movement.

For the second prototype, which is done in FYP2, the server can send instructions to the mobile IoT sensors, instead of just receiving data from them. This enables two-way communication between the cloud server and the mobile IoT sensors. Besides, the calculation for the best route is also done in this stage to enable the mobile IoT sensors to move to its destination point without colliding with obstacles and other mobile sensors. For the final product, a 2D line chart is added to complete the features of the system. The 2D line chart visualized the data collected from the sensors, which are air temperature and humidity. After the system is done, it is integrated with the real-life mobile IoT sensors. A series of tests are conducted to ensure the system is working with the implementation of mobile IoT sensors.

From the features mentioned above, the project objectives are achieved, which are:

1. To develop an IoT visualization platform that can support mobile IoT sensor nodes.
2. To extend the existing indoor map support to support visualization for mobile IoT sensor nodes.

Hence, as a conclusion, the project objectives as mentioned above and the main focus of this project are achieved, which is to develop an extension of visualization framework which supports indoor mapping for mobile IoT sensor nodes.

## **6.2 Future Work**

There are some improvements that can be done for further development of this project. The visualization framework can be further improved in terms of website design. Besides, the algorithm for best route calculation can also be replaced with other more efficient algorithms that can calculate the shortest and most efficient route for the mobile IoT sensors to travel. Furthermore, the calibration of the AGV can be further improved by not only do it while it reaches a QR code, but also calibrate it while the AGV is moving.

## BIBLIOGRAPHY

- Atrius IDE, 2018. Available from: <[https://www.acuitybrands.com/-/media/Images/AcuityBrands%20New/Solutions/Internet%20of%20Things/Atrius/Integrated%20Development%20Environment/Atrius-IDE\\_A-Single-Development-and-Visualization-Platform-block-icons-screen-Image\\_1144x1260px%20png.png?h=1260&la=en&w=1144&hash=1CCE04B028823F45046990E3038797B57EB67D24](https://www.acuitybrands.com/-/media/Images/AcuityBrands%20New/Solutions/Internet%20of%20Things/Atrius/Integrated%20Development%20Environment/Atrius-IDE_A-Single-Development-and-Visualization-Platform-block-icons-screen-Image_1144x1260px%20png.png?h=1260&la=en&w=1144&hash=1CCE04B028823F45046990E3038797B57EB67D24)>. [18 July 2018].
- Atrius IDE Project Example, 2018. Available from: <[https://www.acuitybrands.com/-/media/Images/AcuityBrands%20New/Solutions/Internet%20of%20Things/Atrius/Integrated%20Development%20Environment/Project%20Examples/Atrius-IDE\\_Project-Example-1\\_938x664px%20jpg.jpg](https://www.acuitybrands.com/-/media/Images/AcuityBrands%20New/Solutions/Internet%20of%20Things/Atrius/Integrated%20Development%20Environment/Project%20Examples/Atrius-IDE_Project-Example-1_938x664px%20jpg.jpg)>. [18 July 2018].
- Atrius Solution Builder, computer software 2018. Available from: <<https://www.acuitybrands.com/solutions/internet-of-things/atrus/atrus-integrated-development-environment>>. [18 July 2018].
- Baker, P., 2018. *IBM Watson Analytics - Review 2018 - PCMag Asia*. [Online] Available at: <https://sea.pcmag.com/ibm-watson-analytics/14301/review/ibm-watson-analytics> [Accessed 18 July 2018].
- Degeler, A., 2015. *Data Visualization Tools For IoT*. [Online] Available at: <https://stanfy.com/blog/data-visualization-tools-for-iot/> [Accessed 18 July 2018].
- First Data Visualization Ever, 2018. Available from: <<https://datahero.com/wp-content/uploads/2015/04/First-Data-Visualization-Ever.png>>. [31 July 2018].
- Freeboard, computer software 2018. Available from: <<https://freeboard.io>>. [18 July 2018].
- Friendly, M., Valero-Mora, P. & Ulargui, J. I., 2010. The First (Known) Statistical Graph: Michael Florent van Langren and the "Secret" of Longitude. *The American Statistician*, Volume 0, p. 0.
- Gapminder, 2018. *Trendalyzer*. [Online] Available at: <https://www.gapminder.org/tag/trendalyzer/> [Accessed 31 July 2018].
- Gray, P., 2015. *Data Visualization Throughout History*. [Online] Available at: <https://datahero.com/blog/2015/04/02/data-visualization-throughout-history/> [Accessed 31 July 2018].
- Horne, J., 2017. *The History of the History of Data Visualization*. [Online] Available at: <https://www.idashboards.com/blog/2017/11/29/the-history-of-the-history-of-data-visualization/> [Accessed 31 July 2018].

- IBM Watson Analytics, computer software 2018. Available from: <<https://www.ibm.com/watson-analytics>>. [18 July 2018].
- IBM Watson Analytics – Homepage Dropdown Menu, 2018. Available from: <[https://www.pcmag.com/image\\_popup/0,1740,iid=521858,00.asp](https://www.pcmag.com/image_popup/0,1740,iid=521858,00.asp)>. [18 July 2018].
- IoT Visual Freeboard, 2018. Available from: <<https://stanfy.com/wp-content/uploads/2015/09/iot-visual-freeboard.jpg>>. [18 July 2018].
- Khvostionov, D., n.d. *Internet of Things Data Visualization: Interview With CTO and Co-Founder of DGLogik Dennis Khvostionov*. [Online] Available at: <https://www.postscapes.com/iot-voices/interviews/internet-things-visualization-tools-interview-cto-co-founder-dglogik-dennis-khvostionov/> [Accessed 18 July 2018].
- Lysakowska, S., 2017. *Cloud Vs On Premise Software: Which is Best For Your Business?*. [Online] Available at: <https://www.xperience-group.com/blog/cloud-vs-on-premise-software/> [Accessed 18 July 2018].
- Post, T., 2018. *Straight Talk: Review of Tableau Software, the Pros and Cons*. [Online] Available at: <https://www.yurbi.com/blog/straight-talk-review-of-tableau-software-the-pros-and-cons/> [Accessed 18 July 2018].
- Software, S., n.d. *On-Premises vs. Cloud Computing*. [Online] Available at: <https://sherpasoftware.com/blog/on-premise-vs-cloud-computing/> [Accessed 18 July 2018].
- Tableau Desktop – Dashboard, 2018. Available from: <[https://www.pcmag.com/image\\_popup/0,1740,iid=469301,00.asp](https://www.pcmag.com/image_popup/0,1740,iid=469301,00.asp)>. [18 July 2018].
- Tableau Software, computer software 2018. Available from: <<https://www.tableau.com/>>. [18 July 2018].
- Tag Cloud, 2018. Available from: <[http://3.bp.blogspot.com/-aepyVr\\_3pt0/U3QB\\_D\\_nNVI/AAAAAABmzU/6xPuRaFH33c/s1600/Tag+cloud.png](http://3.bp.blogspot.com/-aepyVr_3pt0/U3QB_D_nNVI/AAAAAABmzU/6xPuRaFH33c/s1600/Tag+cloud.png)>. [31 July 2018]
- ThingSpeak, 2018. Available from: <[https://thingspeak.com/assets/Signup\\_TSP\\_ML\\_image-9211a489730f1c04d24cde0954123901.svg](https://thingspeak.com/assets/Signup_TSP_ML_image-9211a489730f1c04d24cde0954123901.svg)>. [18 July 2018].
- ThingSpeak, computer software 2018. Available from: <<https://thingspeak.com/>>. [18 July 2018].

Trendalyzer, 2018. Available from: <<https://datahero.com/wp-content/uploads/2015/04/Trendalyzer.png>>. [31 July 2018].



## APPENDICES

### Appendix A

#### Python Code for QR Code Calibration

```
import os, sys
import picamera
import numpy
import zbar
import zbar.misc
from scipy.misc import imread as read_image
from move_alexbot2 import AlexBot

camera = picamera.PiCamera()
camera.resolution = (480,270)
ab = AlexBot()

def calibrate(ab):
    count = 0
    flag_Xpos = 0
    flag_Ypos = 0
    while(count < 10):
        camera.capture('up1.jpg')
        img =
zbar.misc.rgb2gray((read_image('up1.jpg')))

        scanner = zbar.Scanner()
        results = scanner.scan(img)

        if (len(results) > 0):
            x = results[0].position[0][0]
            y = results[0].position[0][1]

            print(x)
            print(y)

            if (x < 120):
                print ("left")
                ab.move("left",0.05)

            elif (x > 240):
                print ("right")
                ab.move("right",0.05)

            else:
                flag_Xpos = 1
                print ("Ok!")

        if (y < 70):
            print ("forward")
            ab.move("forward",0.05)
```

```

        elif (y > 135):
            print ("backward")
            ab.move("backward",0.05)

        else:
            flag_Ypos = 1
            print ("Ok!")

    if (flag_Xpos == 1 and flag_Ypos == 1):
        break

    count = count + 1

count = 0
flag_X1pos = 0
flag_X2pos = 0
flag_X3pos = 0
flag_X4pos = 0
while(count < 10):
    camera.capture('up1.jpg')
    img =
zbar.misc.rgb2gray((read_image('up1.jpg')))

    scanner = zbar.Scanner()
    results = scanner.scan(img)

    if (len(results) > 0):
        x1 = results[0].position[0][0]
        x2 = results[0].position[1][0]
        x3 = results[0].position[2][0]
        x4 = results[0].position[3][0]
        print(x1)
        print(x2)
        print(x3)
        print(x4)

        if (abs(x1 - x2) > 10):
            if( x1 > x2 ):
                print("rotate clockwise")
                ab.move("clockwise",0.05)
            else:
                print("rotate anticlockwise")
                ab.move("anticlockwise",0.05)
        else:
            print("Ok!")
            break

    count = count + 1

```

```

count = 0
flag_Xpos = 0
flag_Ypos = 0
while(count < 10):
    camera.capture('up1.jpg')
    img =
zbar.misc.rgb2gray((read_image('up1.jpg')))

    scanner = zbar.Scanner()
    results = scanner.scan(img)

    if (len(results) > 0):
        x = results[0].position[0][0]
        y = results[0].position[0][1]

        print(x)
        print(y)

        if (x < 120):
            print ("left")
            ab.move("left",0.05)

        elif (x > 240):
            print ("right")
            ab.move("right",0.05)

        else:
            flag_Xpos = 1
            print ("Ok!")

        if (y < 70):
            print ("forward")
            ab.move("forward",0.05)

        elif (y > 135):
            print ("backward")
            ab.move("backward",0.05)

        else:
            flag_Ypos = 1
            print ("Ok!")

    if (flag_Xpos == 1 and flag_Ypos == 1):
        break

    count = count + 1

calibrate(ab)

```

# POSTER



Universiti Tunku Abdul Rahman

## A Cloud Solution for Visualizing and Navigating Mobile IoT Sensor Nodes

Building a solution to solve the lack of support for mobile IoT sensor nodes and insufficient indoor map support for the existing visualization platforms.

By See Tian Xin

### Introduction

#### Problem Statement:

- Existing IoT visualization platforms do not support mobile IoT sensor nodes.
- Indoor map support for visualization platforms is not sufficient to visualize mobile sensors data.

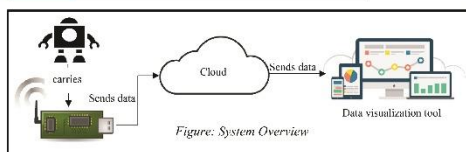
#### Project Scope:

To build an extension from existing data visualization tools which can support mobile IoT sensors.

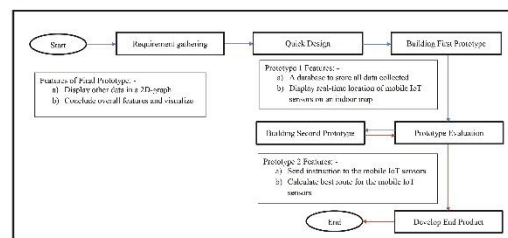
#### Project Objectives:

- To develop an IoT visualization platform that can support mobile IoT sensor nodes.
- To extend the existing indoor map support to support visualization for mobile IoT sensor nodes.

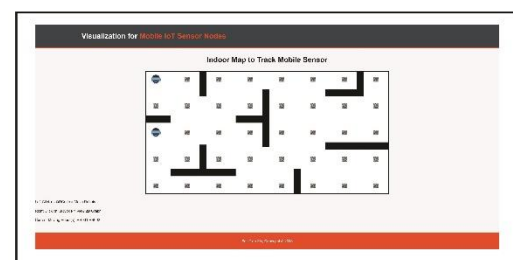
### System Overview



### Methodology



### Result



### Conclusion

As a conclusion, the project objectives as mentioned and the main focus of this project are achieved, which is to develop an extension of visualization framework which supports indoor mapping for mobile IoT sensor nodes.

Supervisor: Dr. Ooi Boon Yaik

# PLAGIARISM CHECK RESULT

The screenshot displays the Turnitin Feedback Studio interface. The main document area shows the following text:

**22**  
**CHAPTER 1: INTRODUCTION**  
**1.1 Problem Statement**  
As for this project, a visualization platform is developed for the use of mobile IoT sensor nodes. The problem domain of this project is the data visualization platform. There are a lot of visualization platforms in the market which have different functions and features available. The problem of these existing platforms is that they do not support mobile sensor nodes. Hence, a visualization platform that can fully support mobile IoT sensor nodes is needed to be developed.

The problem statements of this project can be summarized as below:

1. Existing IoT visualization platforms do not support mobile IoT sensor nodes.
2. Indoor map support for visualization platforms is not sufficient to visualize mobile sensors data.

The problem statement of this project is the existing IoT visualization platforms are not able to support mobile IoT sensor nodes. It needs to be taken into considerations as most of our IoT devices nowadays are not static. The second problem statement is the indoor map support for visualization platforms is not sufficient to visualize mobile sensors data. If there is a mobile sensor moving around a designated area, the indoor map will become extremely important and helpful in visualizing where the mobile sensor is currently located at. Hence, this project will focus on solving the two problem statements mentioned above.

**1.2 Background and Motivation**  
Data visualization has been around far longer than computers have existed. From simple cave drawings to the dashboards we have today that display and connect millions of data. Although the term "Data Visualization" just has its firm roots in the 20<sup>th</sup> Century, the idea is as old as time (Horne, 2017). The first representation of statistical data is created on 1644 by Michael Florent Van Langren, a Flemish

The right-hand side of the interface shows the 'Match Overview' panel with a total match percentage of 6%. A list of 12 matches is provided:

Match Number	Source	Match Percentage
1	Submitted to Universiti ... Student Paper	1%
2	Submitted to American... Student Paper	<1%
3	Submitted to Waikato I... Student Paper	<1%
4	Submitted to Coventry ... Student Paper	<1%
5	Submitted to University... Student Paper	<1%
6	Submitted to University... Student Paper	<1%
7	Hui-Zhong Zhuang, Shu... Publication	<1%
8	Submitted to Universiti ... Student Paper	<1%
9	Submitted to Universiti ... Student Paper	<1%
10	Submitted to Deakin U... Student Paper	<1%
11	Submitted to University... Student Paper	<1%
12	Submitted to CSU, San ... Student Paper	<1%

At the bottom of the interface, the status bar shows: Page: 1 of 40, Word Count: 6387, Text-only Report, High Resolution On, and a search icon.

Turnitin

https://www.turnitin.com/newreport.asp?eq=0&eb=0&esm=0&oid=1107120170&m=0&svr=322&r=44.76255...

preferences

turnitin Originality Report

Processed on: 07-Apr-2019 22:32 +08  
ID: 1107120170  
Word Count: 6387  
Submitted: 2

FYP2  
By See Tian Xin

Similarity Index  
6%

Similarity by Source  
Internet Sources: 1%  
Publications: 0%  
Student Papers: 6%

Document Viewer

exclude quoted exclude bibliography exclude small matches mode: show highest matches together

**CHAPTER 1: INTRODUCTION 1.1 Problem Statement** As for **20**  
**this project,**

a visualization platform is developed for the use of mobile IoT sensor nodes. The problem domain of this project is the data visualization platform. There are a lot of visualization platforms in the market which have different functions and features available. The problem of these existing platforms is that they do not support mobile sensor nodes. Hence, a visualization platform that can fully support mobile IoT sensor nodes is needed to be developed. The problem statements of this project can be summarized as below: 1. Existing IoT visualization platforms do not support mobile IoT sensor nodes. 2. Indoor map support for visualization platforms is not sufficient to visualize mobile sensors data. The problem statement of this project is the existing IoT visualization platforms are not able to support mobile IoT sensor nodes. It needs to be taken into considerations as most of our IoT devices nowadays are not static. The second problem statement is the indoor map support for visualization platforms is not sufficient to visualize mobile sensors data. If there is a mobile sensor moving around a designated area, the indoor map will become extremely important and helpful in visualizing where the mobile sensor is currently located at. Hence, this project will focus on solving the two problem statements mentioned above. 1.2 Background and Motivation Data visualization has been around far longer than computers have existed. From simple cave drawings to the dashboards we have today that display and connect millions of data. Although the term "Data Visualization" just has its firm roots in the 20th Century, the idea is as old as time (Horne, 2017). The first

**representation of statistical data is created on 1644 by Michael Florent Van Langren, a Flemish astronomer** **12**

(Friendly, et al., 2010). Figure 1-2-1: One-dimensional line graph by Michael Florent (First Data Visualization Ever 2018) With the invention of computer and internet, data visualization very became popular as it enables us to represent a bunch of information in a way that is simple for our kin to interpret. There are a lot of progress that has been done since the first ever data visualization, and there are three main milestones in data visualization

- 1% match (student papers from 28-Nov-2011)  
[Submitted to Universiti Teknologi Malaysia](#)
- < 1% match (student papers from 26-Apr-2011)  
[Submitted to American University in Dubai](#)
- < 1% match (student papers from 10-Apr-2012)  
[Submitted to Waikato Institute of Technology](#)
- < 1% match (student papers from 24-Sep-2007)  
[Submitted to Coventry University](#)
- < 1% match (student papers from 24-Aug-2009)  
[Submitted to University of Hertfordshire](#)
- < 1% match (student papers from 10-Apr-2014)  
[Submitted to University of Kent at Canterbury](#)
- < 1% match (publications)  
[Hui-Zhong Zhuang, Shu-Xin Du, Tie-Jun Wu, "Real-Time Path Planning for Mobile Robots", 2005 International Conference on Machine Learning and Cybernetics, 2005](#)
- < 1% match (student papers from 28-Nov-2011)  
[Submitted to Universiti Teknologi Malaysia](#)
- < 1% match (student papers from 28-Nov-2016)  
[Submitted to Universiti Malaysia Sabah](#)
- < 1% match (student papers from 13-May-2018)  
[Submitted to Deakin University](#)
- < 1% match (student papers from 05-Apr-2019)  
[Submitted to University of Bristol](#)
- < 1% match (student papers from 20-Sep-2014)

<b>Universiti Tunku Abdul Rahman</b>			
<b>Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)</b>			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



**FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY**

<b>Full Name(s) of Candidate(s)</b>	SEE TIAN XIN
<b>ID Number(s)</b>	1503313
<b>Programme / Course</b>	Bachelor of Computer Science (HONS)
<b>Title of Final Year Project</b>	A Cloud Solution for Visualizing and Navigating Mobile IoT Sensor Nodes

<b>Similarity</b>	<b>Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)</b>
<b>Overall similarity index: <u>6%</u></b>  <b>% Similarity by source</b> Internet Sources: <u>1</u> % Publications: <u>0</u> % Student Papers: <u>6</u> %	
<b>Number of individual sources listed of more than 3% similarity: <u>0</u></b>	
<b>Parameters of originality required and limits approved by UTAR are as Follows:</b> (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

***Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.***

\_\_\_\_\_  
Signature of Supervisor

\_\_\_\_\_  
Signature of Co-Supervisor

Name: \_\_\_\_\_

Name: \_\_\_\_\_

Date: \_\_\_\_\_

Date: \_\_\_\_\_



## UNIVERSITI TUNKU ABDUL RAHMAN

### FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

#### CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	1503313
Student Name	See Tian Xin
Supervisor Name	Dr Ooi Book Yaik

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Front Cover
√	Signed Report Status Declaration Form
√	Title Page
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

<p>I, the author, have checked and confirmed all the items listed in the table are included in my report.</p>  <p>_____</p> <p>(Signature of Student)</p> <p>Date:</p>	<p>Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction.</p>  <p>_____</p> <p>(Signature of Supervisor)</p> <p>Date:</p>
--	--



