Documentation

Cypher

17 june 2022

1 Team members

Listing of the individual team members

1. Sheikh Muhammad Adib Bin Sh Abu Bakar
2. Zafirul Izzat Bin Mohamad Zaidi
3. Muhammad Farid Izwan Bin Mohamad Shabri
4. Muhammad Iqbal Bin Mohd Fauzi

2 Introduction

For our project we plan to make a smart gate controller that is mainly used for controlling people entering a restricted area. For instance, entering a theme park. Our smart gate controller is capable of counting people who are entering and leaving an area through the gate and blocking people from entering the restricted area. To make it more compatible with the big area environment, this system supports controlling and monitoring the gate from distance. This is where we use the concept of the Internet of Things(IoT) where all the components in our system are capable of connecting to the Internet so that the admin can control the gate even though he or she is not there. Since an area could have many gates, we will group all the sensors and actuators for a gate and connect them to a node. This is where we apply the wireless sensors network concept to connect all the nodes with a master controller.

3 Concept description

The smart gate controller allows the administration of a restricted area like a theme park, zoo or stadium to control the entrance of the area by closing and opening the gate from distance and get the information about the new coming visitor such as the high and the temperature of the visitor, total number of visitor and a current number of visitor. This smart gate controller could also notify the admin if there is a fire near the gate, this feature is to avoid any visitor from entering the area when there is a fire.

To realise that the admin can access the system from distance, wireless network communication is implemented, where all the sensors and actuators are connected to only one node and the node can communicate with the admin known as a client through a server.

Those connections use the MQTT protocol. This means that the wifi protocol is also implemented.

The concept of the smart gate controller is visualised in the form of a requirements diagram and overall system architecture as shown in Figure 1 and 2. Table 1 is also included to make clear about the devices, components and application that we use throughout this project.

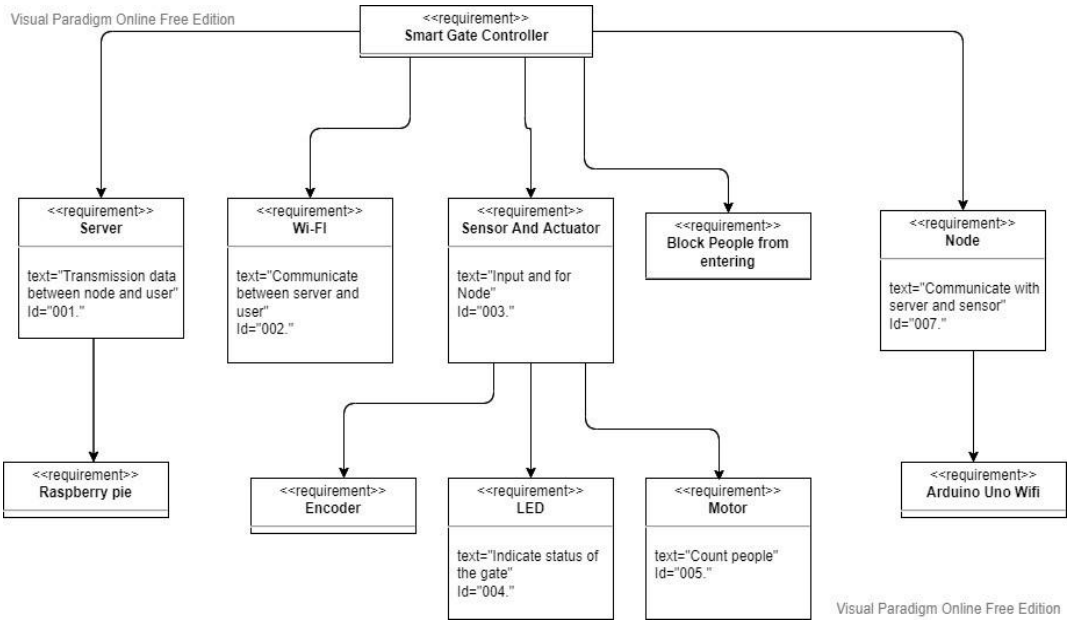

Figure 1 : Requirement diagram

The devices, components and actuators used in the system are listed below.

Type	Name	Function
Devices	Arduino UNO wifi	As node
	Raspberry pi	As Server / Broker
	Mobile phone	As a client
	Computer	
Components (Sensors/Actuators)	Fire sensor	Detect fire
	Distance sensor	Detect distance
	Temperature sensor	Detect temperature
	led	Detect fire, lock state, wifi and MQTT connection indicator
	Step motor	Gate locker
Apps	MQTT explorer	On Windows
	MQTT Clients	On Android and IOS

Tabe 1: The devices, components and apps used in the system

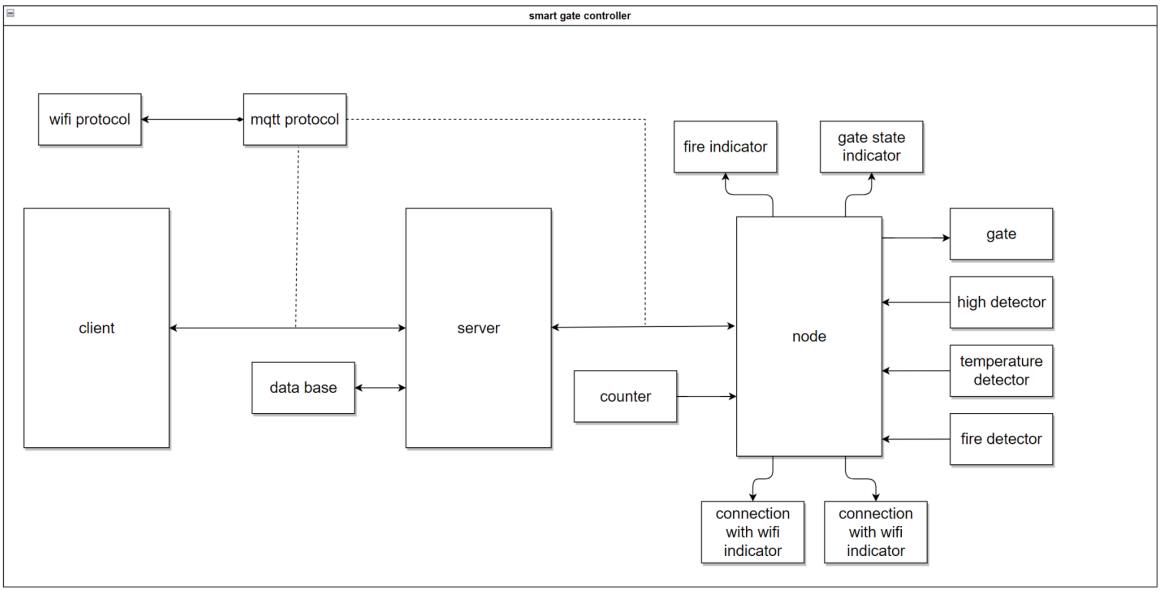

Figure 2: Smart gate overview architecture

4 Project/Team management

For project management, we use the scrum method where we divide the time that we have to develop this product into 11 cycles and each cycle takes 1 week. To make it more efficient, we have two minimum meetings in each cycle. In the first meeting, we will discuss and brainstorm together the big task that we need to complete in the cycle. As a result, the sub-tasks are created and distributed fairly among the team members. The second meeting is to verify the current progress and make improvements.

Since we use the scrum method, we still help each other in completing each sub-task to ensure the product of each cycle is at its highest and finest quality. The division of the tasks is shown in table 2.

cycle	Task	Subtask	Actor
1	Idea and discussion	Concept	All members
2	Concept model	Use case	Zafirul Izzat Farid Izwan
		Activity	
		Requirement	
		Sequence	

		System architecture	Sheikh Adib Muhammad Iqbal
		Draft system architecture	
		Class	
3	Verifying model	All models	All members
4	Improvement and synchronisation	All models	All members
5	Discuss for devices, sensors and actuators	Listing all devices, sensors and actuators	All members
6	Component test	Class for gate + test	Farid Izwan Muhammad Iqbal
		Class for temperature + test	
		Class for fire alarm detector + test	
		Class for counter + test	Zafirul Izzat Sheikh Adib
		Class for height+ test	
7	Source file	Wifi	Sheikh Adib
		Mqtt	
		System behaviour	Zafirul Izzat Farid Izwan
		API	
8	Server preparation	Prepare platform for installation of raspberry pi	All members
		Install mosquito	
		Test mosquito with arduino	
9	Integration management	Software management	Zafirul Izzat Muhammad Iqbal
		Hardware management	Sheikh Adib Farid Izwan
10	Simulation for a chosen scenario	Test publish topic	All members
		Test subscribe topic	
11	Documentation	Concept description	Farid Izwan
		Introduction	Sheikh Adib

		Project/Team management	
		Technologies	Muhammad Iqbal
		Implementation	Zafirul Izzat
		Use case	

Table 2 : Tasks divi*sion*

5 Technologies

In our smart gate controller, many sensors and actuators are used to simulate our system. The list of the sensors and actuators, their functions and the reasons why we choose them are shown in Table 3.

Sensors / Actuators	Functions	Reasons
Flame Sensor (KY-026)	To detect if any fire occurs	Wavelength between 760 - 1100nm (most sensitive)
Temperature Sensor (KY-013)	To measure the temperature of visitor	Can measure temperature in the range of -55 °C up to +125 °C
		lower resistance on higher temperature.
Ultrasonic Sensor Module (KY-050)	To measure the height of visitors	Can measure height in the range of 2 cm to 300cm which is the range of human height.
Rotary encoder (KY-040)	To count the number of visitors	Act as an indicator of either visitor entering (clockwise rotation) or visitor leaving (anticlockwise rotation)
RGB 5mm LED Module (KY-016)	Green = unlock	Act as an indicator to show whether the gate is locked or unlocked.
	Red = lock	
28BYJ-48 DC 5V 12V Stepper Motor	To control the gate either open or close	can be positioned accurately, one 'step' at a time.
LED (blue and	To indicate whether there is	There is no display connected to

yellow)	fire (blue LED)	the node, so we use led as the indicators
	To indicate whether the connection with the server is established or not.(yellow LED)	
	To indicate whether the connection with the server is established or not. (yellow LED)	

Table 3: Sensors and actuators

In our system, the main communication protocol that is used for the communication between the components in our system is the MQTT protocol. In this protocol, the client, the admin and the node will send a message to the server as a command. If a client subscribes to a specific topic, for example, they will always receive messages from the server. The server acts as a broker in this protocol. According to table 4, there are six topics in our project that can be subscribed by the clients, such as topic gate, topic new visitor, and so on. As a result of its client subscription capability, the system can be more reliable and efficient in certain circumstances.

More than one client, for instance, node and admin, will subscribe to the topics, by using web-based applications, phone apps and desktop apps. For example, we are using the “easymqtt” app, which can be found in the Apple Store or Google Play stores and the “ MQTT-explorer” app on Windows. In order to use the MQTT protocol, the wifi protocol will be used to transmit data between the server and the client.

In developing the software part, we use C++ with Arduino API on Arduino UNO WIFI as the platform for the node. By installing Mosquito software, Raspberry PI acts as a server or a broker for our system.

6 Implementation

The project implementation consists of designing the static structure of the system and its environment. This design entails many diagrams in order to make it more clear for implementing the system prototype.

First, the reader will be exposed to a class diagram. Then, to know the system and the environment in more detail, use case models will assist the reader. To make it more concrete, the activity diagram is explained based on the use case model. Last but not least, with the help of the previous diagram, we generalise our system into the system architecture.

In this class diagram, we have included both system components and environments. As shown in Figure 3, we can see that the smart gate controller is the centre between all devices in the system and also the environment. The smart gate controller contains features like people count, getting the temperature and measuring the height of the visitor. Since our smart gate controller gets input from both environment and system, then the information is passed through to the controller to take further actions, we can generalise the interactions and come up with the overall system architecture.

Other than the smart gate controller, we have also defined all the entities of each component. As can be seen in the diagram, there are a class of gate, counter, flame detector, temperature detector and height detector. To make the connection work very well on the Arduino Wifi Rev2 board, each smart device and also the application has the same attribute which is the pin number. In gate, it consists of two functions. The first one is the lock that will lock the gate if the admin or the user sends an appropriate command. Then, it also has an unlock function to unlock the gate. Next, the counter. The functions get the number of visitors and update the number of visitors is to know the total of visitors visited in the whole day and the other one is to know the exact number of visitors in that place at a time. Furthermore, a flame detector is also included. The function of the flame is to detect fire. So, if there is fire, then the blue led will turn on to indicate there is fire. Now, we move on to the temperature detector. In this component, the function is to get the temperature of the coming visitor. Whenever a visitor enters the gate, it will give the temperature of the visitor. So, when the client subscribes to a topic called get temperature, the client will obtain the data of the visitor's temperature. Then, for the height detector, we have to get height as a function. As a piece of additional information, in one of the scenarios, the visitor's height is taken when the visitor enters the gate. Last but not least, we also included other environments such as visitors, sensors, fire and clients as shown in Figure 3.

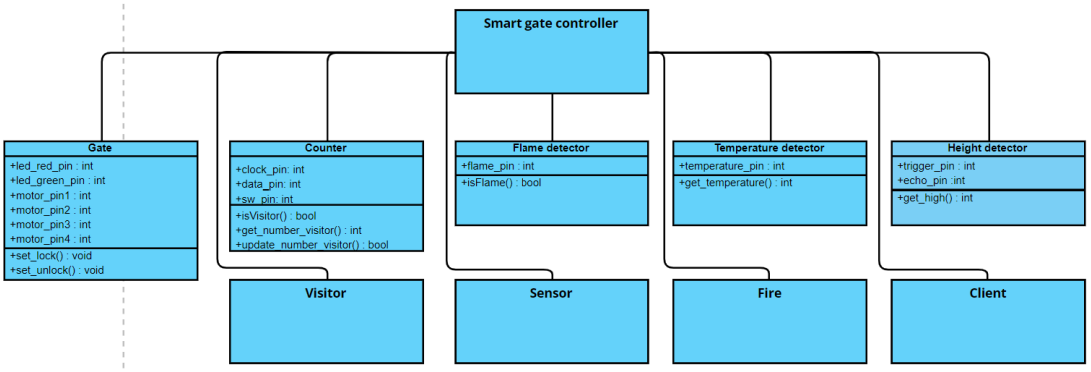

Figure 3: Class diagram

Our next step is to define the intended behaviour of our system using use cases. We chose this modelling as one of our next methods since it allows us to create a system from the standpoint of the user reader. Now, in Figure 4, we describe our use case diagram. With both of the information that we gathered from the diagrams, we have a clearer vision and better understanding of the overall smart gate controller system. This is important as we can now develop a more concrete system architecture with refined details. That is exactly what we did in this documentation for the reader to easily digest our overall system architecture.

As we can see, the outside of the system consists of the user, a node which acts as a middle man in the system and server. The user or also known as the admin can have information on the number of visitors. The user is also capable of sending requests to the server. The main thing about the user is that the user can control the gate remotely. So, the gate can be opened and closed based on the command given by the user. From the node perspective, the node can count visitors entering and leaving with the help of sensors. The node can also get the gate state which is either lock or unlock. Lastly, we move to the server. The server, which is the Raspberry PI, can store the data both from the user and node. Not to forget, the most important part is the data of the system itself. After getting a little bit of knowledge about the overview of our system, we will move to our next approach which is the activity diagram.

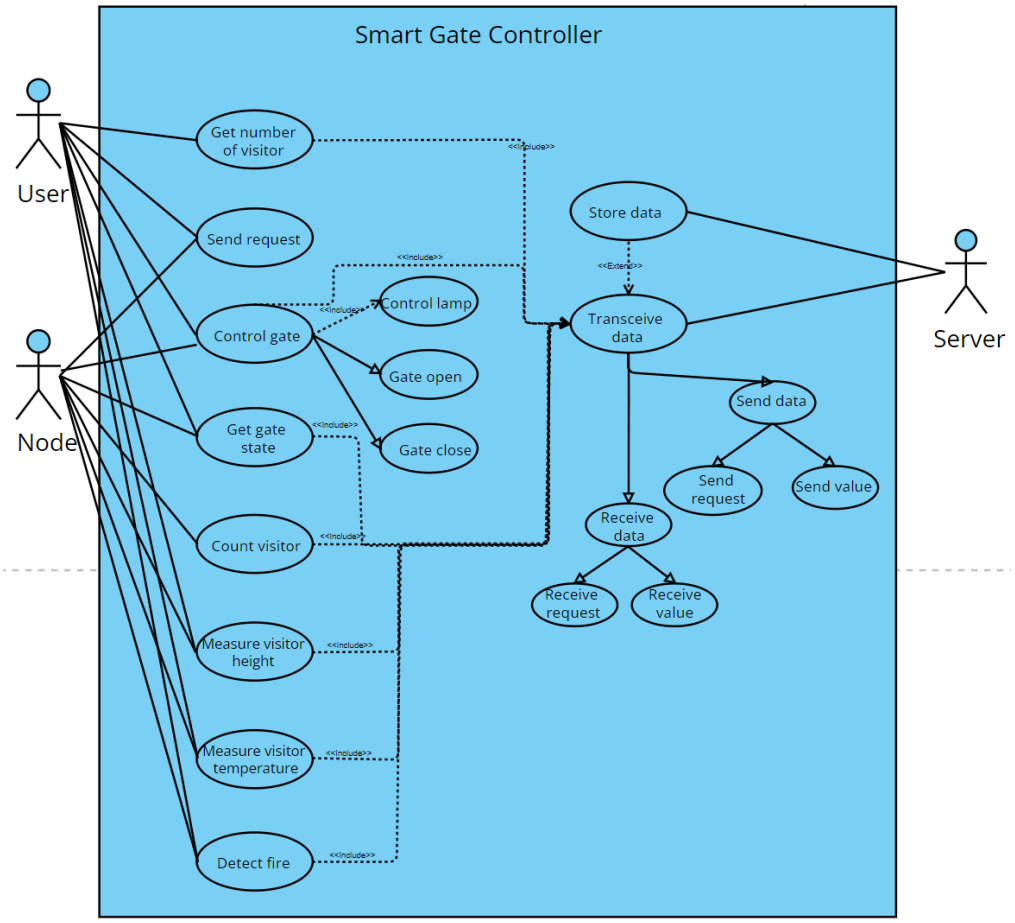

Figure 4: Use case diagram

To see more details about the scenario, the activity diagram is modelled. Figure 5 shows how the overall scenario for Smart Gate Controller will be run. Firstly, it will connect to the server. After connecting to the server, it will send data, either the gate will open or close. If the gate is closed, then a green LED will turn on. On the other hand, if the gate is opened, a red LED will pop out.

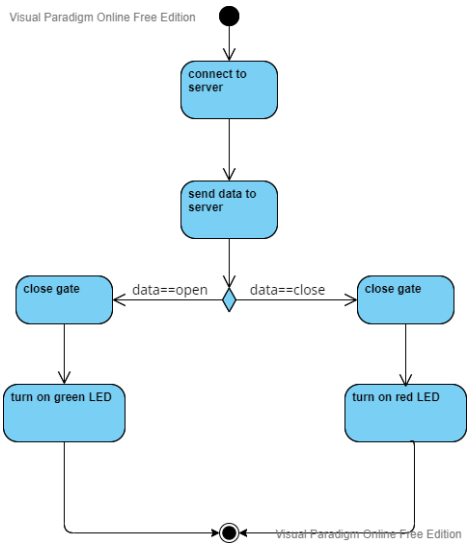

Figure 5: Activity diagram client (node)

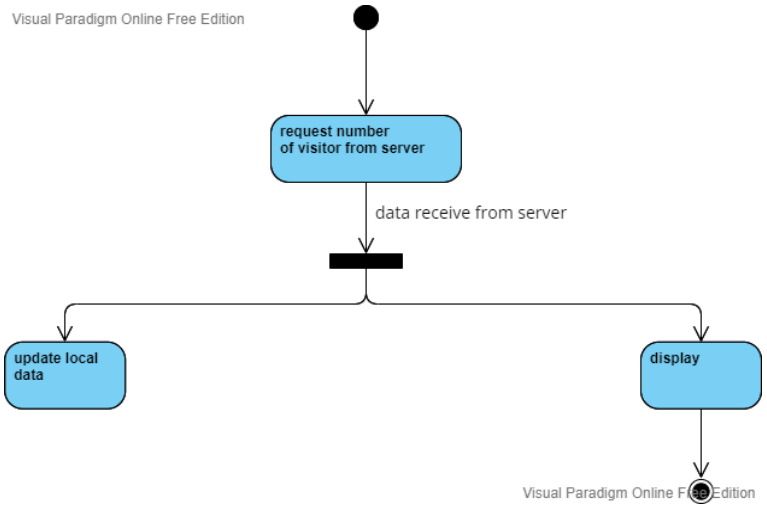

Figure 6: Activity diagram client (admin)

Then, we move to the next Activity diagram. Figure 6 explains how users get the data about the number of visitors. The user will request a number of visitors from the server. Then, the data will be displayed and also local data will be updated once the data is received.

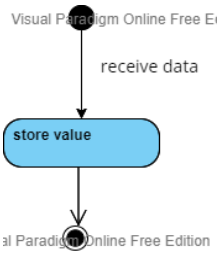

Figure 7: Activity diagram server/broker

Figure 7 shows how data is stored. In the beginning, it will receive data, and then the received data will be stored.

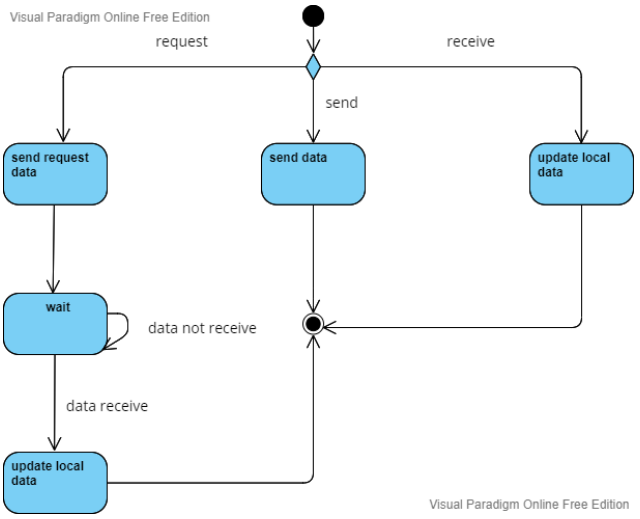

Figure 8: Activity diagram on how the data is transmitted

The diagram in Figure 8, describes how the data is transmitted. From the initial node, there is a decision node on whether the data is requested, received or sent. If the data is requested, then, the required data will be sent. Then, go to wait for the state, if the data is not received. If requested data is received, local data will be updated. Besides that, from the decision node, sending data and updating local data are parts of how data is transmitted.

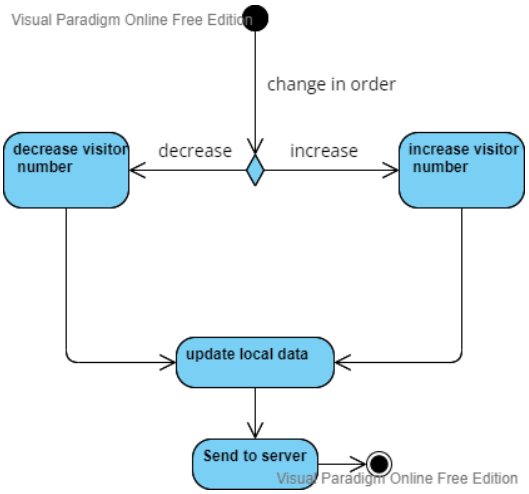

Figure 9: Activity diagram on how the visitor counter works

In Figure 9, explains how the visitor counter works. If there is a change in the number of visitors, if it is either increasing or decreasing, the counter will count the latest number of visitors. Once the latest number is calculated, the local data will be updated and will be sent to the server.

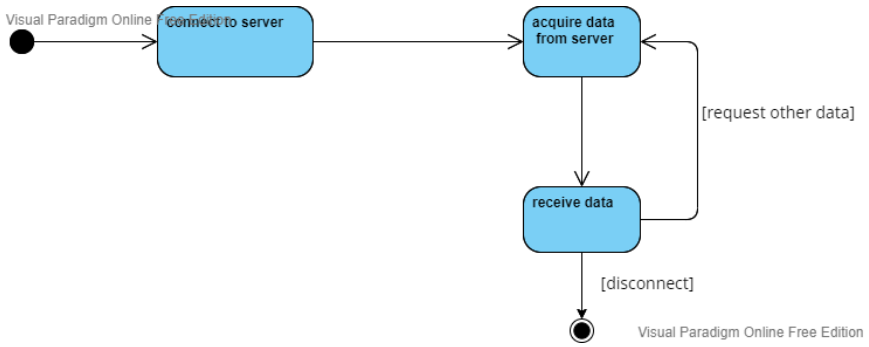

Figure 10: Activity diagram on how the server will receive data

Figure 10 shows how the server will receive data. Initially, the server is connected. The server will request the data. The requested data then will be received by the server. Once the server is done receiving data, it will be disconnected.

With the help of the previous models, we can now form our system architecture. Based on Figure 11 explains the system architecture of the smart gate controller. It consists of Client, Server and Node. For each subsystem, it will define more details, about what happens in every subsystem.

Figure 11: System architecture

Figure 12 shows what the architecture of the client looks like. The client subsystem consists of 5 main components: input interface, output interface, controller, database and communication system.

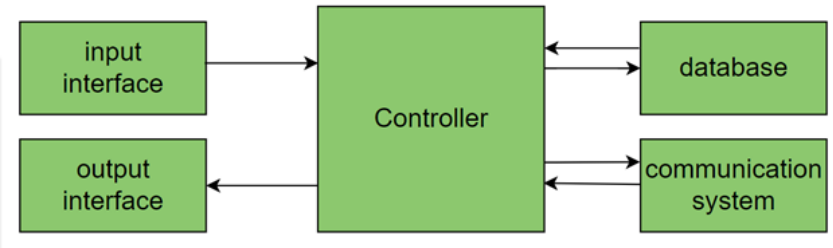

Figure 12: Subsystem architecture of client

For the server subsystem, there are 3 main components that play a big role to make sure the server is good. The three components are the communication system, data manager and database as shown in Figure 13.

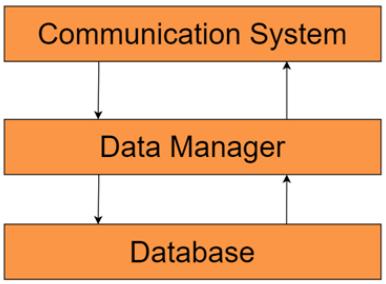

Figure 13: Subsystem architecture of server

Lastly, Figure 14 shows the node subsystem. Node subsystems are made up of 5 main components: communication system, data manager, sensor, actuator and database.

For our project, we will use a rotary sensor to count the number of people. The DC motor is used to open and close the gate. Last but not least is LED which acts as an indicator for visitors to enter and leave.

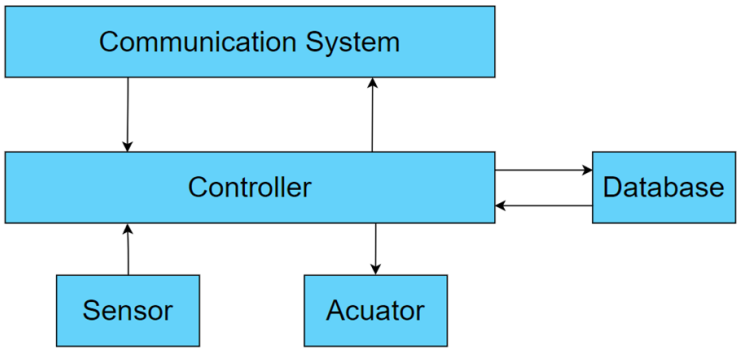

Figure 14: Subsystem architecture of node

7 Use Case

The instructions on how to use and set up the system are described here. This will assist the reader to simulate the smart gate controller system. As additional information, the reader can refer to Figure 15 in order to have a grasp of overview on how the hardware setup of our system.

A. Set up manual

1. Power up the server

First we turn on the server.

2. Power up the node

After the server is fully on, we must turn on the node. This action will enable the node to connect to the broker which is the server. The wifi indicator will be light on if the

node is successfully connected to the wifi and the server indicator will be light on if the node is successfully connected to the server.

3. Connect the client to the server

To connect to the server, first, we must install an MQTT application on the phone as an example. In our case, we use easy MQTT as our application to publish and subscribe.

The set-up manual is visualised in Figure 15.

B. Simulation manual

1. How to publish

The user has to publish the topic that we made as listed in table 4. To help the reader, we also include a picture of how to publish in Figure 16.

2. How to subscribe

The user has to subscribe to the topic that we made as listed in table 4. The result of publication and subscription are visualised in Figure 16.

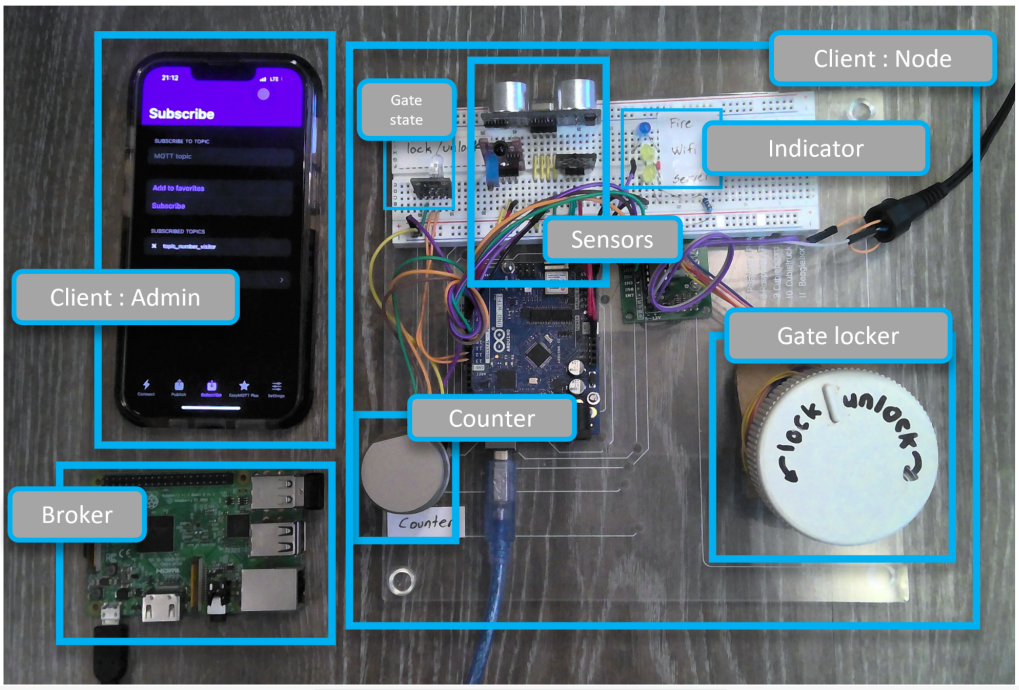

Figure 15: Hardware system setup

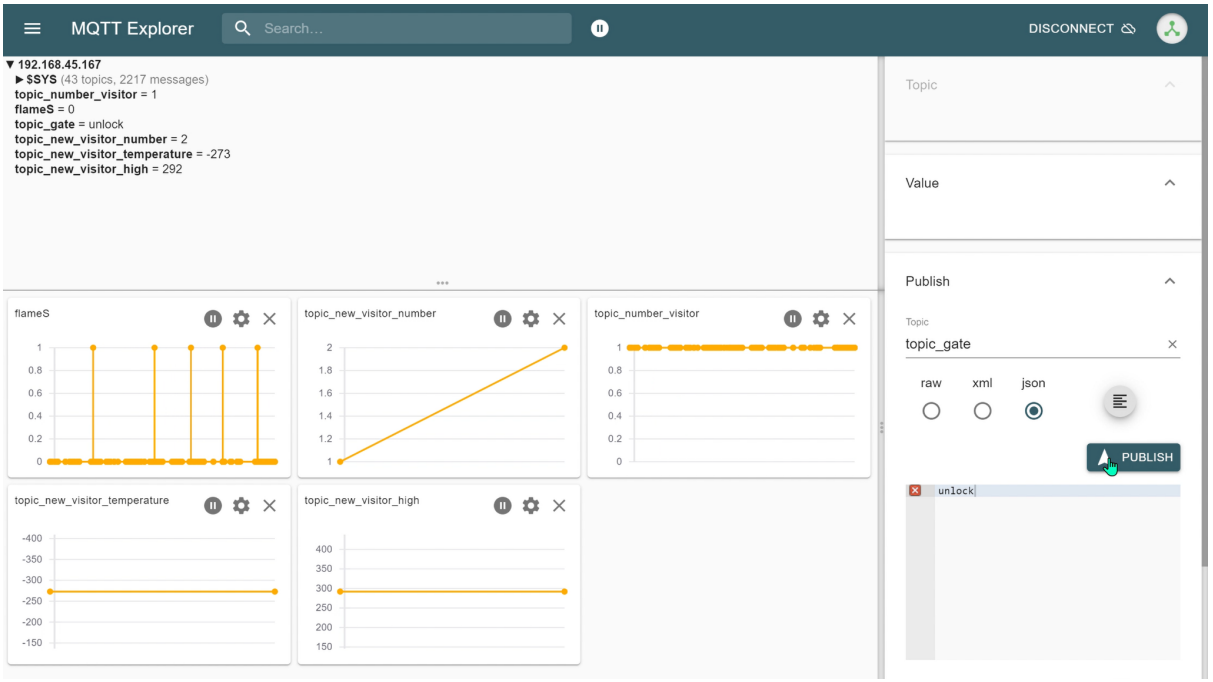

Figure 16: Publish and subscribe using MQTT application

C. Application environment

Now we can state that our system has been effectively implemented from a little notion to a robust Smart gate controller. The major reason we picked this project is that we want to show the world that our system can be implemented at any place as long as the requirements are met. There are a few examples of where our Smart gate controller can be implemented.

All of our smart gate controller's features are applicable in theme parks. It is common for a theme park to have a lot of things that must be monitored. For example, the visitor's height must be monitored. It is because some of the rides have a minimum height requirement to use them. Regarding the topic of temperature, it is critical to determine whether the visitor is in good health or not in order to avoid affecting others. Furthermore, due to the current situation with the Covid-19 virus, the temperature of the visitors is required to avoid the spread of Covid-19. When discussing the stadium, not all features will be utilised. It is because the stadium does not require the visitors' height. Only the temperature, visitor count, and other topics are required. The same is true for the zoo, shopping mall, and cinema. The topic height is not required, but the other topics are required so that the admin can monitor the temperature, number of visitors, and total number of visitors.

Topic	Command	Action	Return value	Remarks
topic_gate	lock	gate lock, red led	lock	Admin has

		on		the authority to lock and unlock the gate
	unlock	gate unlock, green led on	unlock	
topic_new_visitor_high	-	-	New visitor height	Only when a new visitor enters
topic_new _vistor_temperature	-	-	New visitor temperature	Only when a new visitor enters
topic_number_visitor	-	-	Current number visitor	Total number visitor at that time
flame_s	-	Fire indicator light on	Return fire state	Led blue turn on when fire
topic_new_visitor_number	-	-	Total number visitor	Total number for the whole day

Table 4: Topics

8 Sources/References

1. embedded-hardware-lab/Projects-Cypher (github.com)
2. MQTT library = arduino-libraries/ArduinoMqttClient: ArduinoMqttClient Library for Arduino (github.com)
3. Wifi library = WiFiNINA - Arduino Reference