# DOCUMENTATION

Team SL
14th June 2023
Hochschule Hamm-Lippstadt

## 1 Team Members

Kajeepan Umaibalan(kajeepan.umaibalan@stud.hshl.de)
Ravindu Athukorala(ravindu.athukorala@stud.hshl.de)
Dapsara Kapuge(dapsara.kapuge@stud.hshl.de)

## 2 Introduction

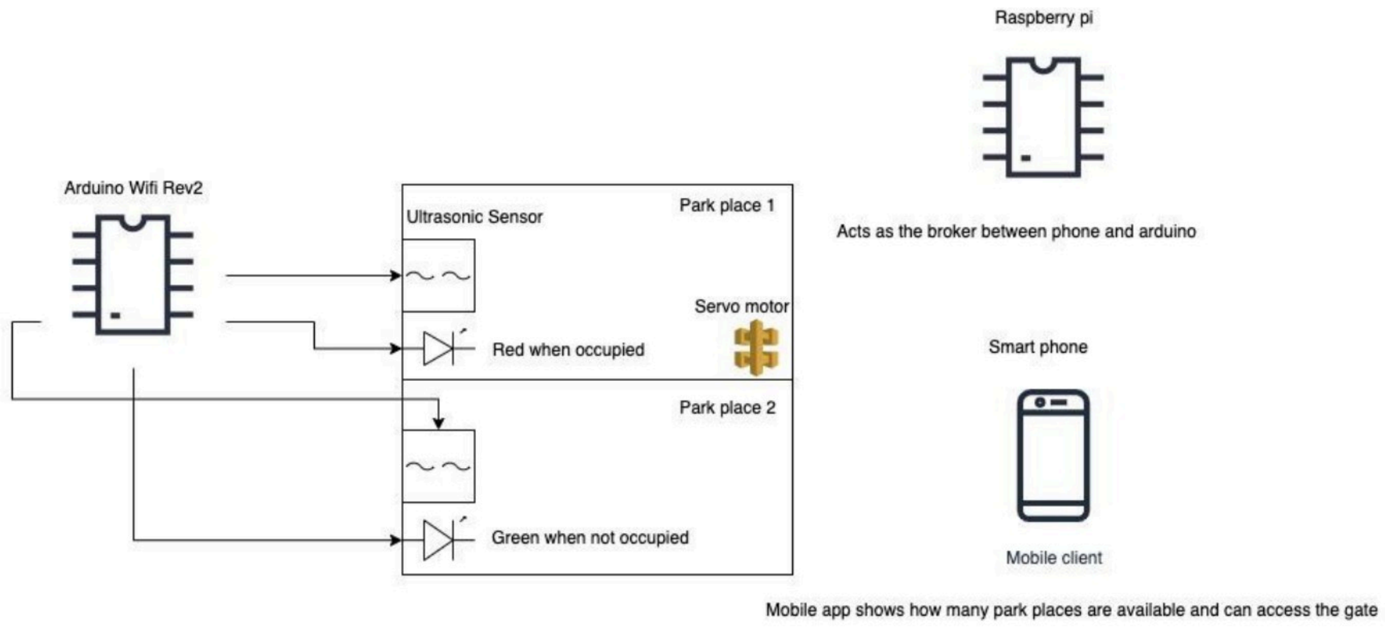
MQTT(Message Queuing Telemetry Transport) is an efficient communication protocol that is designed for IoT devices and applications. It is widely used for real time data exchange and connecting devices over networks.

MQTT operates on a publish and subscribe messaging model. The publishing device can publish messages while the subscribed device can receive the messages.

To connect to a phone using MQTT you need an MQTT broker device which acts as the hub for message routing. The phone that acts as a subscriber connects to the broker and subscribes to specific topics of interest. The other device that acts as the publisher also connects to the broker and publishes messages on the specific topic. When a message is published the broker device forwards that message to all the subscribed devices that are interested on that topic.

In our project we are using this communication protocol to get the data from arduino wifi Rev 2 to our mobile application. In our project Raspberry pi board acts as the Broker and the publisher is Arduino Wifi Rev 2 and the subscriber is our mobile application. Simple overview of the project is, we are making a smart park where the user can check the availability status of how many free parking spaces there are and control the park gates with a smartphone application. We are using 2 ultrasonic sensors, raspberry pi board, arduino wifi rev 2, servo motor, and RGB light as our main electronic compartments. As for the mobile application we are using thf our project.

## 3 Concept Description



Main application in our project is to help the user to get real time information about the car park and access the car park remotely.

For our project we are using - 2 Ultrasonic sensors (Sensor)

Servo motor (Actuator)

RGB light (Actuator)

Arduino Wifi Rev 2 (Device)

Raspberry Pi (Device)

IoT MQTT Panel (Application)

## 4 Project/Team Management

In managing tasks within our project, we followed a organized approach to ensure efficiency and productivity. Firstly, I begin by clearly defining the project goals and objectives, breaking them down into smaller, manageable tasks. This helps me prioritize and determine the critical path for the project. I then create a detailed project plan, outlining timelines, dependencies, and milestones. Utilizing project management software, I assign tasks to team members, ensuring each person's responsibilities align with their skills and expertise. Regular communication and progress meetings play a vital role in keeping everyone informed and addressing any challenges or roadblocks promptly. To stay on track, I regularly monitor the project's progress, tracking key metrics and adjusting timelines or resources as necessary. Finally, I maintain flexibility and adaptability, making sure to reassess priorities and adjust plans whenever unexpected changes or issues arise. By employing this structured approach, I effectively manage tasks, foster collaboration, and ensure the project's successful completion.

### Team members task

- **Ravindu Athukorala :-** As a part of the project, the role was to develop communication and implement connections with Raspberry Pi brokers Using MQTT. Two Arduinos were used in this project. One Arduino acts as a publisher, which sends the sensor sampling data to the broker using MQTT, and the other Arduino is used as a subscriber, which receives data from the broker in order to activate the actuator.

- **Kajeepan Umaibalan :-** the role was to install the operating system and mqtt application on the Raspberry Pi. We made a MQTT configuration to configure the Raspberry pi as a broker. And made communication with smartphones possible for data visualization. The smartphone also acts as a publisher and subscriber.

- **Dapsara Kapuge :** In the project, the role was to code and configure the Arduino IDE to set up the sensors and read data. I utilized the Arduino programming language and IDE to write the necessary code for sensor integration. This involved initializing the sensors, setting up appropriate pins and communication protocols, and implementing data reading algorithms. Through this role, I ensured the seamless integration of sensors into the Arduino platform, enabling accurate and reliable data acquisition for further processing.

## 5 Technologies

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol that is commonly used in IoT applications. It follows a publish-subscribe model, where devices publish data to specific topics, and other devices subscribe to those topics to receive the data. MQTT enables efficient and reliable communication between the Arduino, Raspberry Pi, and MQTT panel app.

The Raspberry Pi serves as the MQTT broker, which is a lightweight messaging protocol used for communication between devices in the Internet of Things (IoT) ecosystem. The MQTT broker facilitates the exchange of data between the Arduino and the MQTT panel app.

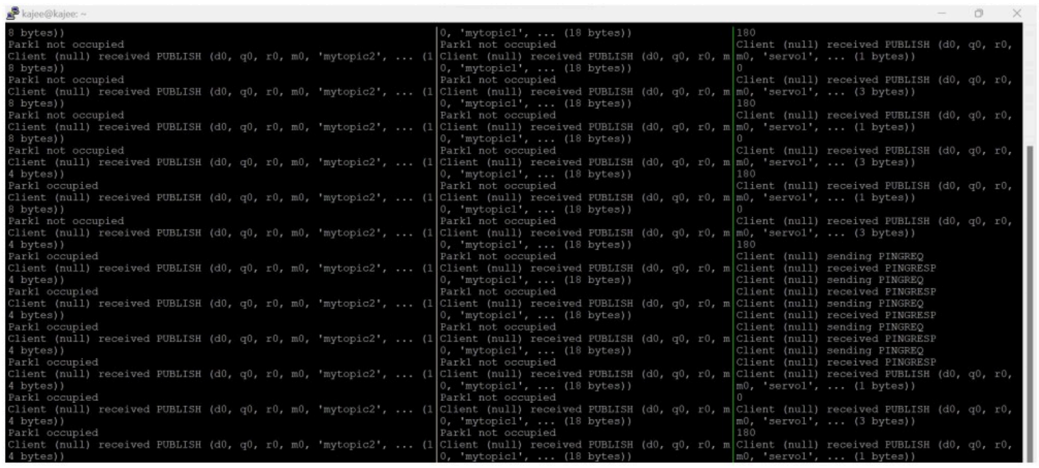
Arduino board with built-in WiFi capabilities is used. This allows the Arduino to connect to the local network and communicate with other devices, such as the Raspberry Pi and the MQTT panel app. The Arduino collects data from an ultrasonic sensor, which measures the availability of parking spots.

The MQTT IOT panel app is an application that allows users to monitor and visualize the availability of parking spots in real-time. It subscribes to the MQTT topics where the parking spot data is published, and displays the information in a user-friendly interface. The app provides users with up-to-date information on parking spot availability.

The ultrasonic sensor is used to detect the presence of vehicles in parking spots. It emits sound waves and measures the time it takes for the waves to bounce back after hitting an object. This data is then used to determine the availability of parking spots.

A servo motor is a type of motor that allows for precise control of angular position. In your project, the servo motor is used to control the opening and closing of the entrance gate. By integrating the servo motor with your system, you can remotely operate the gate through commands sent from the IoT app.

As applications, we used the Arduino IDE,Putty terminal, and IoT MQTT Panal App. A Raspberry pi was used as a broker, and MQTT was installed on it. The MQTT that was implemented on this project is not public. In order to access the topic through MQTT, The user has to input username **kajee** and password **1234: "`mosquitto_sub -u kajee -p 1234 -t <topic>`".**

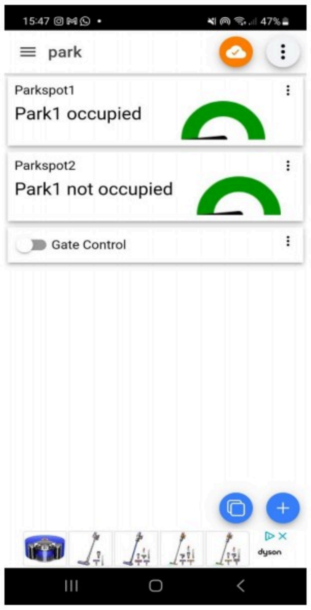

kajee@kajee: ~

```
8 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
8 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
8 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
8 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
4 bytes))
Park1 occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
8 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
4 bytes))
Park1 occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
4 bytes))
Park1 occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
4 bytes))
Park1 occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
4 bytes))
Park1 occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
4 bytes))
Park1 occupied
Client (null) received PUBLISH (d0, q0, r0, m0, 'mytopic2', ... (1
4 bytes))
```

```
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
Park1 not occupied
Client (null) received PUBLISH (d0, q0, r0, m
0, 'mytopic1', ... (18 bytes))
```

```
180
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (1 bytes))
0
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (3 bytes))
180
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (1 bytes))
0
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (3 bytes))
180
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (1 bytes))
0
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (3 bytes))
180
Client (null) sending PINGREQ
Client (null) received PINGRESP
Client (null) sending PINGREQ
Client (null) received PINGRESP
Client (null) sending PINGREQ
Client (null) received PINGRESP
Client (null) sending PINGREQ
Client (null) received PINGRESP
Client (null) sending PINGREQ
Client (null) received PINGRESP
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (1 bytes))
0
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (3 bytes))
180
Client (null) received PUBLISH (d0, q0, r0,
m0, 'servo1', ... (1 bytes))
```

The IoT MQTT Panel app uses three topics to subscribe and publish messages. For both ultrasonic sensors we implemented on  projects "mytopic1" and mytopic2," the payload was min 0 and max 1500. In publishing the message for a servo motor to turn it on and off through switches, which is assigned to the topic servo1," the payload min is 0 and max is 180 because the maximum angle of servo is 180. We have used zero-level service at Zero level.



The Arduino IDE platform was used to write c code to implement connections with MQTT and also to get sensor data. We have used two Arduinos, in which one acts as a publisher and the other acts as a Subscriber. In both Arduinos, we implemented code to connect the Arduino WiFinina to the same wireless network as the broker.

```
#include <WiFiNINA.h>
#include <PubSubClient.h>
#include <stdio.h>


const char* ssid = "Kajee06";
const char* password = "kajee19960106";
const char* mqtt_server = "192.168.186.1";
const int mqtt_port = 1883;
const char* mqtt_username = "kajee";
const char* mqtt_password = "1234";
```

The above code shows the configuration of WiFi and MQTT on Arduino WiFiNinna

```
client.setServer(mqtt_server, mqtt_port);

 while (!client.connected())
 {
  Serial.print("Attempting MQTT connection...");
  if (client.connect("arduinoClient", mqtt_username, mqtt_password))
  {
   Serial.println("connected");
  }
  else {
   Serial.print("failed, rc=");
   Serial.print(client.state());
   Serial.println(" try again in 5 seconds");
   delay(5000);
     }
 }
```

Implementing MQTT connections on the arduino board and broker is done using the above code provided

```
 client.publish("mytopic1","Park1 occupied");
 delay(50);
```

This code is used to publish the message to MQTT broker on the topic name mytopic1

```
 int position = atoi((char*)payload);

 if (strcmp(topic, "servo1/control") == 0) {
  servo1.write(position);
 } else if (strcmp(topic, "servo2/control") == 0) {
  servo2.write(position);
 }
}
```

This code is implemented for the purpose of subscribing topic from MQTT broker

Using all the codes, we can create a wireless MQTT communication network between sensors and actuators.

## 6 Summary

In order to develop a practical and user-friendly parking system, the smart parking spot concept combines a number of cutting-edge technology. An Arduino board with WiFi connectivity, a Raspberry Pi serving as a MQTT broker, and an ultrasonic sensor to gauge parking spot availability make up the main parts. Communication between devices is made simple by the MQTT protocol.

The entrance gate is also equipped with a servo motor that can be operated remotely via an IoT app. To provide a hassle-free parking experience, the MQTT panel app gives users real-time updates on parking spot availability.

Overall, this solution for a smart parking place combines the strength of MQTT messaging, a broker Raspberry Pi, Arduino with WiFi for data collecting, ultrasonic sensors for detection, a servo motor for gate control, and IoT apps for user engagement. As a result, a smart parking solution that improves user comfort and parking place management is created.

## 7 Sources and References.

- HiveMQ MQTT and MQTT 5 Essentials

- Tantitharanukul, Nasi & Osathanunkul, Kitisak & Hantrakul, Kittikorn & Pramokchon, Part & Khoenkaw, Paween. (2017). MQTT-Topics Management System for sharing of Open Data. 62-65. 10.1109/ICDAMT.2017.7904935.

- Delporte, F. (n.d.). *MQTT on Raspberry Pi - Send Sensor Data to HiveMQ Cloud withJavaandPi4J*. https://www.hivemq.com/article/mqtt-raspberrypi-part01-sensor-data-hivemqcloud-java-pi4j/

- *WiFiNINA Library Examples | Arduino Documentation*. (n.d.). https://docs.arduino.cc/tutorials/communication/wifi-nina-examples

Affidavit

I hereby confirm that I have written this paper independently and have not used any sources or aids other than those indicated. All statements taken from other sources in wording or sense are clearly marked. Furthermore, Iassure that this paper has not been part of a course or examination in the same or a similar version.

| | | |
|---|---|---|
| Name:- Kajeepan Umaibalan | Name:- Ravindu Athukorala | Name:-Dapsara Kapuge |
| Location:- Lippstadt | Location:-Lippstadt | Location:-Lippstadt |
| Date:- 12.07.2023 | Date:-12.07.2023 | Date:- 12.07.2023 |
| Signature:- [illegible] | Signature:- [illegible] | Signature:-Dapsara |