릴레이와 온습도 센서를 활용한 화재시 전기 차단 소켓



[프로젝트 개요 3](#_Toc153577908)

[프로젝트 배경 및 목적 3](#_Toc153577909)

[프로젝트 진행 계획 3](#_Toc153577910)

[프로젝트 설계 4](#_Toc153577911)

[사용 부품 4](#_Toc153577912)

[회로도 5](#_Toc153577913)

[프로젝트 계발 계획 5](#_Toc153577914)

[프로젝트 개발 과정 6](#_Toc153577915)

[하드웨어 구성 6](#_Toc153577916)

[센서 데이터 생성 7](#_Toc153577917)

[릴레이 모듈 제어 10](#_Toc153577918)

[프로세스 간 통신(Socket) 12](#_Toc153577919)

[MariaDB 데이터 통합 20](#_Toc153577920)

[MQTT 통합 26](#_Toc153577921)

[데이터 모니터링 & 시각화 32](#_Toc153577922)

[프로젝트 결과 35](#_Toc153577923)

[결론 36](#_Toc153577924)

# 프로젝트 개요

## 프로젝트 배경 및 목적

* 전선 지중화 사업 진행으로 거리에서 전선을 육안으로 확인할 수 없게 되었다. 전선은 시간이 지나며 노후화가 진행된다. 그로 인해 피복이 약해서 열에 취약해지고 화재의 위험으로 이어질 수 있다.
* 온습도 센서를 이용하여 주변 온도와 습도를 측정한다. 화재, 전선의 저항으로 인한 온도 상승 등 비정상적인 온도 상황이 발생한다면 전선에 흐르는 전류를 차단하여 화재를 예방하고, 2차 사고를 방지 한다.

## 프로젝트 진행 계획

1. 온습도 센서를 통해 온도와 습도 데이터를 생성 한다.
2. 생성된 온도 값에 따라 릴레이 모듈을 제어한다.(정상적인 상황에서는 릴레이 모듈에 연결된 소켓에 전류가 흐르지만 일정 온도 이상 올라가게 된다면 전류를 차단 한다.)
3. 소켓 통신을 통해 온도와 습도 정보를 서버 측으로 전송 한다.
4. 서버는 클라이언트로 받은 데이터를 MariaDB에 저장 한다.
5. 클라이언트에서 MQTT를 사용해 값을 전달 한다.
6. MariaDB에 저장된 데이터를 실시간으로 모니터링 및 시각화 한다.

# 프로젝트 설계

## 사용 부품

* MCU : 라즈베리파이 4 model B

전자제품, 전자 부품, 회로 구성요소, 패시브 회로 부품이(가) 표시된 사진

자동 생성된 설명

* 온습도 센서 : DHT-11

회로 구성요소, 패시브 회로 부품, 전자 부품, 전자 공학이(가) 표시된 사진

자동 생성된 설명

* 릴레이 모듈 : SZH-EK082

텍스트, 회로 구성요소, 전자 부품, 패시브 회로 부품이(가) 표시된 사진

자동 생성된 설명

* 브레드보드, 점퍼 케이블

패턴, 줄무늬, 직물, 줄무늬의이(가) 표시된 사진

자동 생성된 설명

## 회로도

도표, 평면도이(가) 표시된 사진

자동 생성된 설명

## 프로젝트 계발 계획

텍스트, 스크린샷, 폰트, 직사각형이(가) 표시된 사진

자동 생성된 설명

# 프로젝트 개발 과정

## 하드웨어 구성

전자제품, 케이블, 전기 배선, 전자 공학이(가) 표시된 사진

자동 생성된 설명

## 센서 데이터 생성

텍스트, 폰트, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명

통신 방식 : 1 – wired

19~26행에서 다음과 같은 동작을 진행해 주었다.

### 소스코드

|  |
| --- |
| // sensor.c  #include <wiringPi.h>  #include <stdio.h>  #include <stdlib.h>  #include <stdint.h>  #include <unistd.h>  #define DHT\_PIN 7  #define MAX\_TIMINGS 85  int dht\_data[5] = {0, 0, 0, 0, 0}; s  void readDHTData() {  uint8\_t last\_state = HIGH;  uint8\_t counter = 0;  uint8\_t j = 0, i;  dht\_data[0] = dht\_data[1] = dht\_data[2] = dht\_data[3] = dht\_data[4] = 0;  pinMode(DHT\_PIN, OUTPUT);  digitalWrite(DHT\_PIN, HIGH);  delay(10);  digitalWrite(DHT\_PIN, LOW);  delay(18);  digitalWrite(DHT\_PIN, HIGH);  delayMicroseconds(30);  pinMode(DHT\_PIN, INPUT);  for (i = 0; i < MAX\_TIMINGS; i++) {  counter = 0;  while (digitalRead(DHT\_PIN) == last\_state) {  counter++;  delayMicroseconds(2);  if (counter == 255) {  break;  }  }  last\_state = digitalRead(DHT\_PIN);  if (counter == 255) {  break;  }  // 처음 두 비트는 항상 0이므로 스킵  if ((i >= 4) && (i % 2 == 0)) {  dht\_data[j / 8] <<= 1;  if (counter > 16) {  dht\_data[j / 8] |= 1;  }  j++;  }  }  }  int main() {  if (wiringPiSetup() == -1) {  fprintf(stderr, "WiringPi initialization failed\n");  return 1;  }  while (1) {  readDHTData();  float humidity = (float)((dht\_data[0]) + dht\_data[1]);  float temperature = (float)(((dht\_data[2])) + dht\_data[3]);  if (dht\_data[2] & 0x80) {  temperature \*= -1.0;  }  printf("Temperature: %.2f°C, Humidity: %.2f%%\n", temperature, humidity);  delay(2000);  }  return 0;  } |

실행 방법

1. gcc -o sensor sensor.c -lwiringPi
2. ./sensor

### 실행결과

스크린샷, 소프트웨어, 텍스트, 컴퓨터이(가) 표시된 사진

자동 생성된 설명

보고서와 같이 첨부

파일명 : 1.DHT11 결과.mp3

## 릴레이 모듈 제어

### 소스코드

|  |
| --- |
| // relay.c  #include <wiringPi.h>  #include <stdio.h>  #include <stdlib.h>  #include <stdint.h>  #include <unistd.h>  #define Relay\_PIN 26  void Relay\_On() {  pinMode(Relay\_PIN, OUTPUT);  digitalWrite(Relay\_PIN, HIGH);  }    void Relay\_Off() {  pinMode(Relay\_PIN, OUTPUT);  digitalWrite(Relay\_PIN, LOW);  }  int main(void) {  if (wiringPiSetup() == -1) {  fprintf(stderr, "WiringPi initialization failed\n");  return 1;  }  while (1) {  Relay\_On();  delay(5000);  Relay\_Off();  delay(5000);  }  return 0;  } |

실행 방법

1. gcc -o relay relay.c -lwiringPi
2. ./relay

### 실행결과

전자제품, 전자 기기, 컴퓨터 하드웨어, 케이블이(가) 표시된 사진

자동 생성된 설명

보고서와 같이 첨부

파일명 : relay 결과.mp3

## 프로세스 간 통신(Socket)

|  |
| --- |
| // Common.h  #include <sys/types.h> // basic type definitions  #include <sys/socket.h> // socket(), AF\_INET, ...  #include <arpa/inet.h> // htons(), htonl(), ...  #include <netdb.h> // gethostbyname(), ...  #include <unistd.h> // close(), ...  #include <fcntl.h> // fcntl(), ...  #include <pthread.h> // pthread\_create(), ...  #include <poll.h> // poll()  #include <sys/epoll.h> // epoll()  #include <stdio.h> // printf(), ...  #include <stdlib.h> // exit(), ...  #include <string.h> // strerror(), ...  #include <errno.h> // errno  // Windows 소켓 코드와 호환성을 위한 정의  typedef int SOCKET;  #define SOCKET\_ERROR -1  #define INVALID\_SOCKET -1  // 소켓 함수 오류 출력 후 종료  void err\_quit(const char \*msg)  {  char \*msgbuf = strerror(errno);  printf("[%s] %s\n", msg, msgbuf);  exit(1);  }  // 소켓 함수 오류 출력  void err\_display(const char \*msg)  {  char \*msgbuf = strerror(errno);  printf("[%s] %s\n", msg, msgbuf);  }  // 소켓 함수 오류 출력  void err\_display(int errcode)  {  char \*msgbuf = strerror(errcode);  printf("[오류] %s\n", msgbuf);  } |

### 소스코드(Server)

|  |
| --- |
| // server.cpp  #include "Common.h"  #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <string.h>  #include <sys/wait.h>  #include <time.h>  // TCP Socket Parameters  #define SERVERPORT 9000  #define BUFSIZE 512  int \_recv\_ahead(SOCKET s, char \*p) {  static \_\_thread int nbytes = 0;  static \_\_thread char buf[1024];  static \_\_thread char \*ptr;  if (nbytes == 0 || nbytes == SOCKET\_ERROR) {  nbytes = recv(s, buf, sizeof(buf), 0);  if (nbytes == SOCKET\_ERROR) {  return SOCKET\_ERROR;  }  else if (nbytes == 0)  return 0;  ptr = buf;  }  --nbytes;  \*p = \*ptr++;  return 1;  }  int recvline(SOCKET s, char \*buf, int maxlen) {  int n, nbytes;  char c, \*ptr = buf;  for (n = 1; n < maxlen; n++) {  nbytes = \_recv\_ahead(s, &c);  if (nbytes == 1) {  \*ptr++ = c;  if (c == '\n')  break;  }  else if (nbytes == 0) {  \*ptr = 0;  return n - 1;  }  else  return SOCKET\_ERROR;  }  \*ptr = 0;  return n;  }  int main(int argc, char \*argv[]) {  int retval;  SOCKET listen\_sock = socket(AF\_INET, SOCK\_STREAM, 0);  if (listen\_sock == INVALID\_SOCKET) err\_quit("socket()");  struct sockaddr\_in serveraddr;  memset(&serveraddr, 0, sizeof(serveraddr));  serveraddr.sin\_family = AF\_INET;  serveraddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);  serveraddr.sin\_port = htons(SERVERPORT);  retval = bind(listen\_sock, (struct sockaddr \*)&serveraddr, sizeof(serveraddr));  if (retval == SOCKET\_ERROR) err\_quit("bind()");  retval = listen(listen\_sock, SOMAXCONN);  if (retval == SOCKET\_ERROR) err\_quit("listen()");  SOCKET client\_sock;  struct sockaddr\_in clientaddr;  socklen\_t addrlen;  char buf[BUFSIZE + 1];  while (1) {  addrlen = sizeof(clientaddr);  client\_sock = accept(listen\_sock, (struct sockaddr \*)&clientaddr, &addrlen);  if (client\_sock == INVALID\_SOCKET) {  err\_display("accept()");  break;  }  char addr[INET\_ADDRSTRLEN];  inet\_ntop(AF\_INET, &clientaddr.sin\_addr, addr, sizeof(addr));  printf("\n[TCP Server] Connect IP addr=%s, Port Number=%d\n",  addr, ntohs(clientaddr.sin\_port));  while (1) {  retval = recvline(client\_sock, buf, BUFSIZE + 1);  if (retval == SOCKET\_ERROR) {  err\_display("recv()");  break;  }  else if (retval == 0)  break;  printf("[TCP/%s:%d] %s", addr, ntohs(clientaddr.sin\_port), buf);  }  close(client\_sock);  printf("[TCP Server] Disconnect IP addr=%s, Port Number=%d\n",  addr, ntohs(clientaddr.sin\_port));  }  close(listen\_sock);  return 0;  } |

실행 방법

1. gcc -o server server.c -lwiringPi
2. ./server

### 소스코드(Client)

|  |
| --- |
| // client.cpp  #include "Common.h"  #include <wiringPi.h>  #include <stdio.h>  #include <stdlib.h>  #include <stdint.h>  #include <unistd.h>  #define DHT\_PIN 7  #define MAX\_TIMINGS 85  #define Relay\_PIN 26  char \*SERVERIP = (char \*)"127.0.0.1";  #define SERVERPORT 9000  #define BUFSIZE 50  // Function to send data to the server  void sendDataToServer(SOCKET sock, float temperature, float humidity) {  char buf[BUFSIZE];  int len;  len = snprintf(buf, BUFSIZE, "%.2f %.2f\n", temperature, humidity);  printf("[TCP Client] Send Data: %s", buf);  int retval = send(sock, buf, len, 0);  if (retval == SOCKET\_ERROR) {  err\_display("send()");  }  }  void Relay\_On() {  pinMode(Relay\_PIN, OUTPUT);  digitalWrite(Relay\_PIN, HIGH);  }  void Relay\_Off() {  pinMode(Relay\_PIN, OUTPUT);  digitalWrite(Relay\_PIN, LOW);  }  int dht\_data[5] = {0, 0, 0, 0, 0};  void readDHTData() {  uint8\_t last\_state = HIGH;  uint8\_t counter = 0;  uint8\_t j = 0, i;  dht\_data[0] = dht\_data[1] = dht\_data[2] = dht\_data[3] = dht\_data[4] = 0;  pinMode(DHT\_PIN, OUTPUT);  digitalWrite(DHT\_PIN, HIGH);  delay(10);  digitalWrite(DHT\_PIN, LOW);  delay(18);  digitalWrite(DHT\_PIN, HIGH);  delayMicroseconds(30);  pinMode(DHT\_PIN, INPUT);  for (i = 0; i < MAX\_TIMINGS; i++) {  counter = 0;  while (digitalRead(DHT\_PIN) == last\_state) {  counter++;  delayMicroseconds(2);  if (counter == 255) {  break;  }  }  last\_state = digitalRead(DHT\_PIN);  if (counter == 255) {  break;  }  if ((i >= 4) && (i % 2 == 0)) {  dht\_data[j / 8] <<= 1;  if (counter > 16) {  dht\_data[j / 8] |= 1;  }  j++;  }  }  }  int main() {  if (wiringPiSetup() == -1) {  fprintf(stderr, "WiringPi initialization failed\n");  return 1;  }    pinMode(Relay\_PIN, OUTPUT);  digitalWrite(Relay\_PIN, LOW);    int retval;  SOCKET sock = socket(AF\_INET, SOCK\_STREAM, 0);  if (sock == INVALID\_SOCKET) err\_quit("socket()");  struct sockaddr\_in serveraddr;  memset(&serveraddr, 0, sizeof(serveraddr));  serveraddr.sin\_family = AF\_INET;  inet\_pton(AF\_INET, SERVERIP, &serveraddr.sin\_addr);  serveraddr.sin\_port = htons(SERVERPORT);  retval = connect(sock, (struct sockaddr \*)&serveraddr, sizeof(serveraddr));  if (retval == SOCKET\_ERROR) err\_quit("connect()");  while (1) {  readDHTData();  float humidity = (float)((dht\_data[0]) + dht\_data[1]);  float temperature = (float)(((dht\_data[2])) + dht\_data[3]);  if (dht\_data[2] & 0x80) {  temperature \*= -1.0;  }  // printf("Temperature: %.2f°C, Humidity: %.2f%%\n", temperature, humidity);    if (temperature >= 50.00) {  digitalWrite(Relay\_PIN, HIGH);  }  else {  digitalWrite(Relay\_PIN, LOW);  }  sendDataToServer(sock, temperature, humidity);  delay(2000);  }  return 0;  } |

실행 방법

1. gcc -o client client.c -lwiringPi
2. ./client

### 실행결과

텍스트, 소프트웨어, 멀티미디어 소프트웨어, 스크린샷이(가) 표시된 사진

자동 생성된 설명

보고서와 같이 첨부

파일명 : socket결과.mp3

## MariaDB 데이터 통합

데이터베이스 환경

* 데이터베이스명 : sensor
* 테이블명 : sensordata
* 테이블 구조

텍스트, 스크린샷, 멀티미디어 소프트웨어, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

### 소스코드

* 서버측에서 MariaDB에 저장하는 코드를 추가했다.

|  |
| --- |
| // Server.cpp  #include "Common.h"  #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <string.h>  #include <sys/wait.h>  #include <mariadb/mysql.h>  #include <time.h>  // MariaDB Connection pameters  #define DB\_HOST "localhost"  #define DB\_USER "root"  #define DB\_PASSWORD "password"  #define DB\_NAME "sensor"  // TCP Socket Parameters  #define SERVERPORT 9000  #define BUFSIZE 512  void processValues(char\* buf) {  MYSQL \*conn = mysql\_init(NULL);  if (conn == NULL) {  fprintf(stderr, "mysql\_init() failed\n");  exit(EXIT\_FAILURE);  }  if (mysql\_real\_connect(conn, DB\_HOST, DB\_USER, DB\_PASSWORD, DB\_NAME, 0, NULL, 0) == NULL) {  fprintf(stderr, "mysql\_real\_connect() failed\n");  mysql\_close(conn);  exit(EXIT\_FAILURE);  }  char query[255];  sprintf(query,"create table if not exists sensordata (id INT AUTO\_INCREMENT PRIMARY KEY, time DATETIME, temp FLOAT(10), humi FLOAT(10))");  if (mysql\_query(conn, query) != 0) {  fprintf(stderr,"MariaDB query execution failed: %s\n", mysql\_error(conn));  }  char\* values[MAX\_VALUES];  int valueCount = 0;  char\* token = strtok(buf, " ");  while (token != NULL && valueCount < MAX\_VALUES) {  values[valueCount] = token;  valueCount++;  token = strtok(NULL, " ");  }  time\_t currentTime;  struct tm \*localTime;  time(&currentTime);  localTime = localtime(&currentTime);  char currentTimeString[20];  sprintf(currentTimeString, "%04d-%02d-%02d %02d:%02d:%02d",  localTime->tm\_year + 1900, localTime->tm\_mon + 1, localTime->tm\_mday,  localTime->tm\_hour, localTime->tm\_min, localTime->tm\_sec);  sprintf(query, "INSERT INTO sensordata (time, temp, humi) VALUES ('%s', '%s', '%s')",currentTimeString, values[0], values[1]);  if (mysql\_query(conn, query) != 0) {  fprintf(stderr,"MariaDB query execution failed: %s\n", mysql\_error(conn));  }  }  int \_recv\_ahead(SOCKET s, char \*p) {  static \_\_thread int nbytes = 0;  static \_\_thread char buf[1024];  static \_\_thread char \*ptr;  if (nbytes == 0 || nbytes == SOCKET\_ERROR) {  nbytes = recv(s, buf, sizeof(buf), 0);  if (nbytes == SOCKET\_ERROR) {  return SOCKET\_ERROR;  }  else if (nbytes == 0)  return 0;  ptr = buf;  }  --nbytes;  \*p = \*ptr++;  return 1;  }  int recvline(SOCKET s, char \*buf, int maxlen) {  int n, nbytes;  char c, \*ptr = buf;  for (n = 1; n < maxlen; n++) {  nbytes = \_recv\_ahead(s, &c);  if (nbytes == 1) {  \*ptr++ = c;  if (c == '\n')  break;  }  else if (nbytes == 0) {  \*ptr = 0;  return n - 1;  }  else  return SOCKET\_ERROR;  }  \*ptr = 0;  return n;  }  int main(int argc, char \*argv[]) {    int retval;  SOCKET listen\_sock = socket(AF\_INET, SOCK\_STREAM, 0);  if (listen\_sock == INVALID\_SOCKET) err\_quit("socket()");  struct sockaddr\_in serveraddr;  memset(&serveraddr, 0, sizeof(serveraddr));  serveraddr.sin\_family = AF\_INET;  serveraddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);  serveraddr.sin\_port = htons(SERVERPORT);  retval = bind(listen\_sock, (struct sockaddr \*)&serveraddr, sizeof(serveraddr));  if (retval == SOCKET\_ERROR) err\_quit("bind()");  retval = listen(listen\_sock, SOMAXCONN);  if (retval == SOCKET\_ERROR) err\_quit("listen()");  SOCKET client\_sock;  struct sockaddr\_in clientaddr;  socklen\_t addrlen;  char buf[BUFSIZE + 1];  while (1) {  addrlen = sizeof(clientaddr);  client\_sock = accept(listen\_sock, (struct sockaddr \*)&clientaddr, &addrlen);  if (client\_sock == INVALID\_SOCKET) {  err\_display("accept()");  break;  }  char addr[INET\_ADDRSTRLEN];  inet\_ntop(AF\_INET, &clientaddr.sin\_addr, addr, sizeof(addr));  printf("\n[TCP Server] Connect IP addr=%s, Port Number=%d\n", addr, ntohs(clientaddr.sin\_port));  while (1) {  retval = recvline(client\_sock, buf, BUFSIZE + 1);  if (retval == SOCKET\_ERROR) {  err\_display("recv()");  break;  }  else if (retval == 0)  break;  printf("[TCP/%s:%d] %s", addr, ntohs(clientaddr.sin\_port), buf);  processValues(buf);  }  close(client\_sock);  printf("[TCP Server] Disconnect IP addr=%s, Port Number=%d\n", addr, ntohs(clientaddr.sin\_port));  }  close(listen\_sock);  return 0;  } |

실행 방법

1. gcc -o server server.cpp -lwiringPi -lmariadb
2. ./server
3. mariaDB 접속
4. select \* from sensordata;

### 실행결과

스크린샷, 텍스트, 디자인, 가전용품이(가) 표시된 사진

자동 생성된 설명

보고서와 같이 첨부

파일명 : mariaDB 결과.mp3

## MQTT 통합

### 소스코드

|  |
| --- |
| // Server.cpp  #include "Common.h"  #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <string.h>  #include <sys/wait.h>  #include <mariadb/mysql.h>  #include <time.h>  // MQTT  #include <MQTTClient.h>  // MariaDB Connection parameters  #define DB\_HOST "localhost"  #define DB\_USER "root"  #define DB\_PASSWORD "password"  #define DB\_NAME "sensor"  // TCP Socket Parameters  #define SERVERPORT 9000  #define BUFSIZE 512  // MQTT parameters  #define MAX\_VALUES 10  #define MAX\_TEXT 512  #define MQTT\_BROKER\_ADDRESS "tcp://localhost:1883"  #define MQTT\_CLIENT\_ID "temperature\_sensor\_client"  #define MQTT\_TOPIC "test/temphumi"  #define MQTT\_TOPIC1 "test/temp"  #define MQTT\_TOPIC2 "test/humi"  char topics[2][10] = {"test/temp", "test/humi"};  struct message{  long msg\_type;  char msg\_text[MAX\_TEXT];  };  void removeNewLine(char\* str) {  char\* newlineChar = strchr(str, '\n');  if (newlineChar != NULL) {  \*newlineChar = '\0';  }  }  void sendToMQTT(char \*topic, const char \*value) {  MQTTClient client;  MQTTClient\_connectOptions conn\_opts = MQTTClient\_connectOptions\_initializer;  int rc;  MQTTClient\_create(&client, MQTT\_BROKER\_ADDRESS, MQTT\_CLIENT\_ID, MQTTCLIENT\_PERSISTENCE\_NONE, NULL);  conn\_opts.keepAliveInterval = 20;  conn\_opts.cleansession = 1;  if ((rc = MQTTClient\_connect(client, &conn\_opts)) != MQTTCLIENT\_SUCCESS) {  fprintf(stderr, "Failed to connect to MQTT broker : %d\n", rc);  exit(EXIT\_FAILURE);  }  MQTTClient\_message pubmsg = MQTTClient\_message\_initializer;  pubmsg.payload = (void \*)value;  pubmsg.payloadlen = strlen(value);  pubmsg.qos = 1;  pubmsg.retained = 0;  MQTTClient\_publishMessage(client, topic, &pubmsg, NULL);  MQTTClient\_disconnect(client, 10000);  MQTTClient\_destroy(&client);  }  void processValues(char\* buf) {    MYSQL \*conn = mysql\_init(NULL);  if (conn == NULL) {  fprintf(stderr, "mysql\_init() failed\n");  exit(EXIT\_FAILURE);  }  if (mysql\_real\_connect(conn, DB\_HOST, DB\_USER, DB\_PASSWORD, DB\_NAME, 0, NULL, 0) == NULL) {  fprintf(stderr, "mysql\_real\_connect() failed\n");  mysql\_close(conn);  exit(EXIT\_FAILURE);  }  char query[255];  sprintf(query,"create table if not exists sensordata (id INT AUTO\_INCREMENT PRIMARY KEY, time DATETIME, temp FLOAT(10), humi FLOAT(10))");  if (mysql\_query(conn, query) != 0) {  fprintf(stderr,"MariaDB query execution failed: %s\n", mysql\_error(conn));  }  char\* values[MAX\_VALUES];  int valueCount = 0;  char\* token = strtok(buf, " ");  while (token != NULL && valueCount < MAX\_VALUES) {  values[valueCount] = token;  valueCount++;  token = strtok(NULL, " ");  }  time\_t currentTime;  struct tm \*localTime;  time(&currentTime);  localTime = localtime(&currentTime);  char currentTimeString[20];  sprintf(currentTimeString, "%04d-%02d-%02d %02d:%02d:%02d",  localTime->tm\_year + 1900, localTime->tm\_mon + 1, localTime->tm\_mday,  localTime->tm\_hour, localTime->tm\_min, localTime->tm\_sec);  sprintf(query, "INSERT INTO sensordata (time, temp, humi) VALUES ('%s', '%s', '%s')",currentTimeString, values[0], values[1]);  if (mysql\_query(conn, query) != 0) {  fprintf(stderr,"MariaDB query execution failed: %s\n", mysql\_error(conn));  }  removeNewLine(values[1]);  sendToMQTT(topics[0], values[0]);  sendToMQTT(topics[1], values[1]);  printf("Token Data:%s %s %s\n", currentTimeString,values[0], values[1]);  int \_recv\_ahead(SOCKET s, char \*p) {  static \_\_thread int nbytes = 0;  static \_\_thread char buf[1024];  static \_\_thread char \*ptr;  if (nbytes == 0 || nbytes == SOCKET\_ERROR) {  nbytes = recv(s, buf, sizeof(buf), 0);  if (nbytes == SOCKET\_ERROR) {  return SOCKET\_ERROR;  }  else if (nbytes == 0)  return 0;  ptr = buf;  }  --nbytes;  \*p = \*ptr++;  return 1;  }  int recvline(SOCKET s, char \*buf, int maxlen) {  int n, nbytes;  char c, \*ptr = buf;  for (n = 1; n < maxlen; n++) {  nbytes = \_recv\_ahead(s, &c);  if (nbytes == 1) {  \*ptr++ = c;  if (c == '\n')  break;  }  else if (nbytes == 0) {  \*ptr = 0;  return n - 1;  }  else  return SOCKET\_ERROR;  }  \*ptr = 0;  return n;  }  int main(int argc, char \*argv[]) {  int retval;  SOCKET listen\_sock = socket(AF\_INET, SOCK\_STREAM, 0);  if (listen\_sock == INVALID\_SOCKET) err\_quit("socket()");  struct sockaddr\_in serveraddr;  memset(&serveraddr, 0, sizeof(serveraddr));  serveraddr.sin\_family = AF\_INET;  serveraddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);  serveraddr.sin\_port = htons(SERVERPORT);  retval = bind(listen\_sock, (struct sockaddr \*)&serveraddr, sizeof(serveraddr));  if (retval == SOCKET\_ERROR) err\_quit("bind()");  retval = listen(listen\_sock, SOMAXCONN);  if (retval == SOCKET\_ERROR) err\_quit("listen()");  SOCKET client\_sock;  struct sockaddr\_in clientaddr;  socklen\_t addrlen;  char buf[BUFSIZE + 1];  while (1) {  addrlen = sizeof(clientaddr);  client\_sock = accept(listen\_sock, (struct sockaddr \*)&clientaddr, &addrlen);  if (client\_sock == INVALID\_SOCKET) {  err\_display("accept()");  break;  }  char addr[INET\_ADDRSTRLEN];  inet\_ntop(AF\_INET, &clientaddr.sin\_addr, addr, sizeof(addr));  printf("\n[TCP Server] Connect IP addr=%s, Port Number=%d\n", addr, ntohs(clientaddr.sin\_port));  while (1) {  retval = recvline(client\_sock, buf, BUFSIZE + 1);  if (retval == SOCKET\_ERROR) {  err\_display("recv()");  break;  }  else if (retval == 0)  break;  printf("[TCP/%s:%d] %s", addr, ntohs(clientaddr.sin\_port), buf);  processValues(buf);  }  close(client\_sock);  printf("[TCP Server] Disconnect IP addr=%s, Port Number=%d\n", addr, ntohs(clientaddr.sin\_port));  }  close(listen\_sock);  return 0;  } |

실행 방법

1. gcc -o server server.cpp -lmariadb -lpaho-mqtt3c
2. ./server
3. mosquitto\_sub -t test/temp
4. mosquitto\_sub -t test/humi

### 실행결과

스크린샷, 소프트웨어, 멀티미디어 소프트웨어, 텍스트이(가) 표시된 사진

자동 생성된 설명

보고서와 같이 첨부

파일명 : mqtt결과.mp3

## 데이터 모니터링 & 시각화

### 소스코드

|  |
| --- |
| # view.py  import streamlit as st  import pandas as pd  import time  import pymysql  import altair as alt  def generate\_data():  db = pymysql.connect(host="localhost", user="root", password="password", charset="utf8")  cursor = db.cursor(pymysql.cursors.DictCursor)  cursor.execute('USE sensor;')    select\_query = 'SELECT \* FROM sensordata'  cursor.execute(select\_query)    df\_org = cursor.fetchall()  df\_org = pd.DataFrame(df\_org)  df = pd.DataFrame(df\_org, columns=['temp', 'humi'])  df['time'] = pd.to\_datetime(df\_org['time'])    return df  try:  data = generate\_data()  st.title("temp/humi real time dashboard")  # Altair 차트 생성  chart = alt.Chart(data).mark\_line().encode(  x='time:T',  y=alt.Y('temp:Q', title='Temperature (°C)'),  color=alt.value('red')  ).properties(  width=600,  height=300  ) + \  alt.Chart(data).mark\_line().encode(  x='time:T',  y=alt.Y('humi:Q', title='Humidity (%)'),  color=alt.value('blue')  )  chart\_component = st.altair\_chart(chart, use\_container\_width=True)  ph1 = st.empty()  ph2 = st.empty()  while True:  new\_data = generate\_data()  data = pd.concat([data, new\_data])  updated\_chart = alt.Chart(data).mark\_line().encode(  x='time:T',  y=alt.Y('temp:Q', title='Temperature (°C)'),  color=alt.value('red')  ).properties(  width=600,  height=300  ) + \  alt.Chart(data).mark\_line().encode(  x='time:T',  y=alt.Y('humi:Q', title='Humidity (%)'),  color=alt.value('blue')  )  chart\_component.altair\_chart(updated\_chart)  ph1.metric(  label="Temp",  value=f"{data.iloc[-1,0]} °C"  )  ph2.metric(  label="Humi",  value=f"{data.iloc[-1,1]} %"  )  time.sleep(1)  except Exception as e:  st.title(f"Error: {e}") |

실행 방법

1. ./server 실행
2. ./client 실행
3. streamlit run view.py 실행

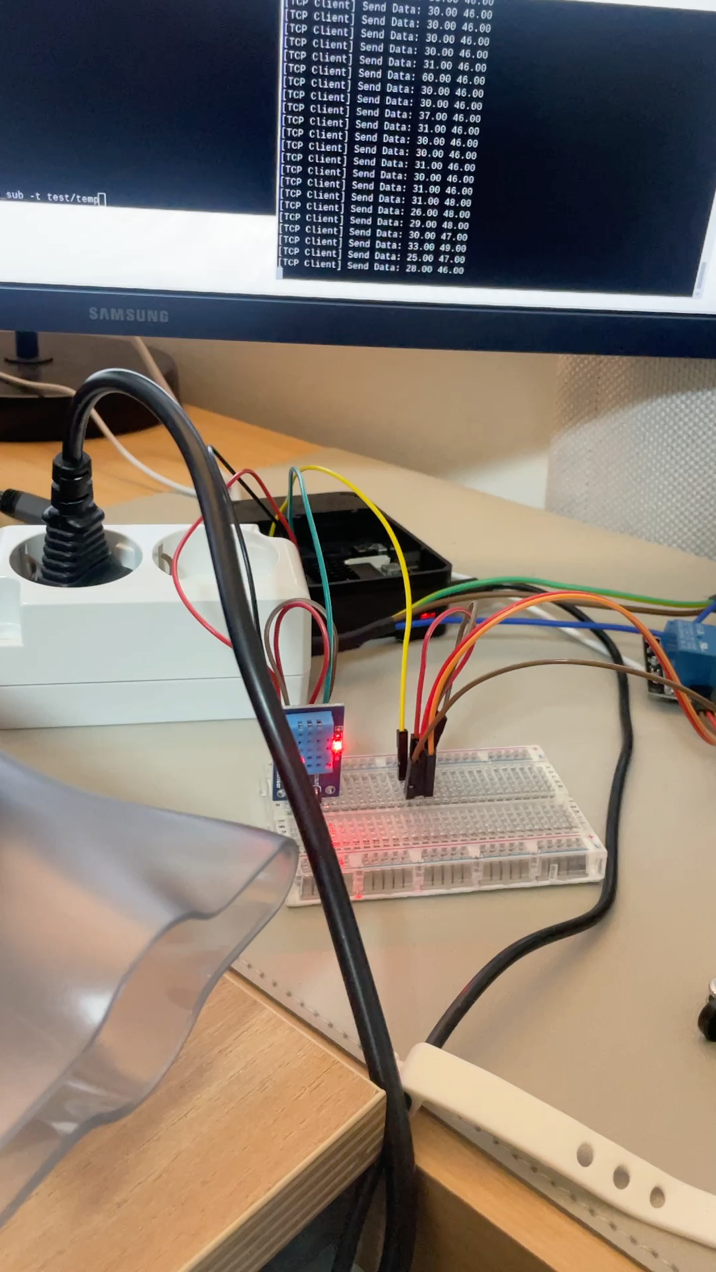
### 실행결과

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

# 프로젝트 결과

* DHT 11로부터 온습도 데이터를 생성하여 소켓 통신 성공했다.
* 클라이언트 측에서 서버 측으로 전달한 데이터를 mariaDB를 활용해 저장했다.
* 서버는 mariaDB에 저장하며 MQTT로 데이터를 보내줬다.
* MQTT로 subscribe를 하면 값을 읽을 수 있었다.
* mariaDB에 저장된 값을 불러와 데이터 모니터링 및 시각화를 했다.



보고서와 같이 첨부

파일명 : 최종결과.mp3

# 결론

현재는 50도가 넘었을 때 꺼지도록 설정했다.

센서의 값이 튀는 경우가 있는데 이때 50도가 넘어서 꺼졌다 켜지는 모습이다.

인위적으로 50도 이상 만드는게 어려워서 튀는 값을 가지고 시연했다.

전선 지중화 등 매설 된 케이블, 눈으로 확인 할 수 없는 전선관 속의 케이블 등을 관리하거나 사고가 발생했을 때 2차 사고 발생을 예방할 수 있고, 온습도 센서를 통해 주변 상황을 모니터링할 수 있다.