Tmemo-2020-47

종합설계 프로젝트 수행 보고서

프로젝트명	라즈베리파이를 이용한 스마트 환풍기
팀번호	S4-11
문서제목	수행계획서( ) 2차발표 중간보고서( ) 3차발표 중간보고서( ) 4차발표 중간보고서( ) 최종결과보고서(O)

2020.12.03

팀원 : 이 상 준 (팀장)

정 진 식

최 광 복

지도교수 : 노 영 주 교수 (인)

문서 수정 내역

작성일	대표작성자	버전(Revision)	수정내용	비고
2020.01.17	최광복	1.0	수행계획서	최초작성
2020.04.08	정진식	1.1	2차 중간보고서	수정
2020.05.01	이상준	1.2	2차 중간보고서	수정
2020.06.26	이상준	1.3	학기말발표	수정
2020.11.25	정진식	1.4	최종발표	최종작성

문서 구성

진행단계	프로젝트 계획서 발표	중간발표1 (2월)	중간발표2 (4월)	학기말발표 (6월)	최종발표 (10월)
기본양식	계획서 양식	계획서 양식	계획서 양식	계획서 양식	계획서 양식
포함되는 내용	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~3) 참고자료	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~4) 참고자료	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~5) 참고자료	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~7) 참고자료	Ⅰ. 서론 (1~6) Ⅱ. 본론 (1~7) Ⅲ. 결론 (1~2) 참고자료

이 문서는 한국산업기술대학교 컴퓨터공학부의 “종합설계” 교과목에서 프로젝트 “라즈베리파이를 이용한 스마트 환풍기”를 수행하는 팀원(S4-11, 이상준, 정진식, 최광복)들이 작성한 것으로 사용하기 위해서는 해당 팀원들의 허락이 필요합니다.

목 차

Ⅰ. 서론

1. 작품 선정 배경 및 필요성

- 스위치 환풍기의 경우 수동으로 환풍기를 제어해야 하는 불편함 존재
- 깜빡하여 환풍기를 끄지 않아 지속적으로 환풍기가 작동할 시, 월 약 7,000 ~ 8,000원의 전기세가 낭비됨
- 다양한 센서들을 통해 습도와 온도를 자동으로 조절하고 스스로 켜지거나 꺼지는 편리한 환풍기가 필요

2. 기존 연구/기술 동향 분석

GiGA Genie 홈 IoT	• 매우 빈번한 작동 오류, 사용 불가능한 상황이 자주 발생 • 로그아웃 되어버리고 비번 아이디는 찾지도 못함
KT GiGA IoT 홈매니저	• 재실행 마다 kt 로그인 유지가 안되고 핀 번호 새로 설정 • 로그인 방법이 번거로움 사용범위에 따라 간편 로그인 기능 필요
U+스마트 홈(IoT@home)	• 앱에 접속했는데 기기 오작동 빈번 • 안드로이드 10 사용 불가

3. 개발 목표

WIFI를 이용하여 습도 및 온도를 DB에 저장하고 어디서나 웹을 통해 확인 가능하며, 습도 및 온도에 따라 스스로 작동하는 IOT 환풍기를 개발

4. 팀 역할 분담

이상준(팀장)	정진식	최광복
• Amazon RDS 환경 설정 • 스프링 환경 설정 • 웹 UI 및 기능 구현	• 라즈베리파이 파이썬 프로그래밍 • 라즈베리파이 회로, 센서 연결	• 라즈베리파이 파이썬 프로그래밍 • 웹 UI 및 기능 구현

5. 개발 일정

	항목	12월	1월	2월	3월	4월	5월	6월	7-9월	10월
추진 일정	요구사항 정의 및 분석	■								
	시스템 설계		■							
	라즈베리파이 코딩		■	■						
	웹 서버, DB 구축			■	■					
	모듈 통합 및 수정					■	■			
	시험 및 데모							■		
	최종 보고서								■	
	최종 발표									■

6. 개발 환경

PC	CPU	Intel Core i5-3230M 2.60GHz
	Memory	8GB RAM
	Graphic	GeForce GTX 750 Ti
	HDD	SSD 128GB
	운영체제	Windows 10 64-bit
Amazon RDS	스토리지	20 GiB
	Memory	1GB RAM
	엔진	MySQL Community
	엔진 버전	8.0.17
기타	개발 Tool	Eclipse, Python IDLE
	개발 언어	JAVA, Python

Ⅱ. 본론

1. 개발 내용

라즈베리파이에 탑재된 와이파이 모듈을 이용하여 라즈베리파이와 MySQL 데이터베이스 서버를 연동한다.

라즈베리파이와 온/습도 센서(DHT-22)를 연결하여 온/습도 센서가 설치된 곳의 온습도를 측정할 수 있다.

라즈베리파이와 5V 릴레이 모듈을 연결하고, 릴레이 모듈과 환풍기를 연결하여

코드로 구현된 On/Off 동작 알고리즘에 따라 라즈베리파이가 릴레이 모듈을 제어하여 환풍기의 전원을 제어할 수 있다

와이파이 통신을 이용해 측정한 온습도, 전원 On/Off 여부 등의 데이터를 라즈베리파이에서 데이터베이스로 송신할 수 있다.

웹 서버와 MySQL 데이터베이스를 연동하여 데이터베이스에 있는 데이터들을 서버로 가져올 수 있다.

가져온 데이터 값들을 시간에 따라 그래프로 표현하여 온/습도 변화 추이를 파악할 수 있다.

로그인에 성공하여 권한을 가진 사람만이 그래프를 통해 온/습도 데이터를 확인할 수 있다.

2. 문제 및 해결방안

- 환풍기가 작동하여 효과적으로 습도와 온도를 조절할 수 있는가?

해결방안 - 힘이 좋은 환풍기를 선택하여 공기 순환이 잘 일어날 수 있도록 함

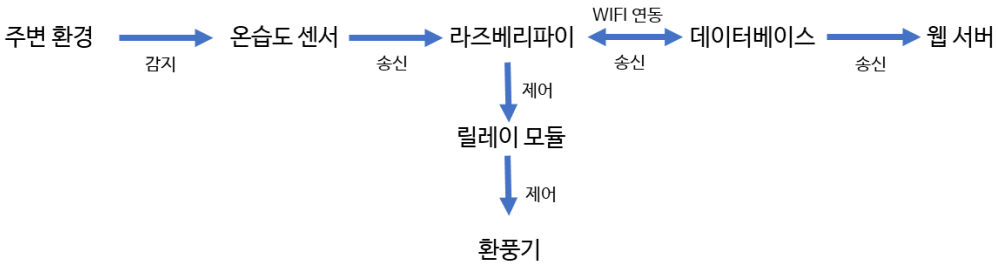
- 온도, 습도 센서의 정확도는 믿을만한가?

해결방안 - 고정밀 온/습도(온도, 습도) 센서 DHT-22를 사용 함

- 웹에서 환풍기 데이터의 조회가 가능한가?

해결방안 - 주기적으로 환풍기의 상태를 데이터베이스로 송신하여 웹에서 주어진 시간의 데이터 값을 조회할 수 있도록 한다.

3. 시험 시나리오



- 온습도 센서가 설치된 주변 환경의 실내/실외 온습도 측정
- 측정한 온습도 값을 라즈베리파이로 전송
- 측정한 온습도 값을 이용하여 코드로 구현한 알고리즘(내부의 온습도가 외부의 온습도대비 설정 기준보다 높을 경우 릴레이 모듈 ON)에 따라 릴레이 모듈 제어
- 릴레이 모듈이 신호를 받으면 연결된 환풍기를 제어
- 라즈베리파이에 내장된 와이파이 모듈을 통해 mysql 데이터베이스 서버와 연동
- 온습도 값과 측정 시간, 환풍기 on/off여부를 데이터베이스로 전송
- 웹 서버에서 로그인에 성공하면 데이터베이스의 값을 가져와 시간에 따른 온습도 변화를 그래프로 표현

4. 상세 설계

◆ 라즈베리파이

(1) 온/습도 읽기

Adafruit_DHT API를 사용하여 Adafruit_DHT.read_retry함수로 센서의 온/습도 값을 읽어온다. 함수 인자에 핀 번호를 다르게 하여 내부 온/습도 센서 값과 외부 온/습도 센서 값을 따로 읽어와 변수에 저장한다. 온/습도 읽기는 반복적으로 시도하고 읽어온 값을 통해 환풍기를 제어하고 1분에 한번 DB에 값을 저장한다(너무 많은 데이터가 저장되는 것 방지).

(2) 릴레이 모듈 / 환풍기 제어

Rpi.GPIO 라이브러리를 사용하여 GPIO 객체를 생성하고 setup()을 통해 릴레이

모듈의 핀 번호와 연결한 뒤, output()의 인자로 True를 주어 릴레이 모듈을 켜거나 False를 주어 릴레이 모듈을 끌 수 있다.

환풍기가 내부 기압을 떨어뜨려 외부 공기를 내부로 들어오게 하는 방식이기 때문에 내부 온습도와 외부 온습도를 비교하여 내부 온도가 외부 온도보다 0.2도 높거나 내부 습도가 외부 습도보다 2% 높은 경우 릴레이 모듈을 켜서 환풍기를 작동시키고, 반대의 경우에는 릴레이 모듈을 꺼서 환풍기의 작동을 멈춘다.

단, 온도가 지나치게 낮아지는 것을 방지하기 위해 온도가 20도 아래로 내려가면 조건을 무시하고 환풍기 작동을 중지한다.

(3) DB연동 및 값 전송

pymysql API를 사용하여 pymysql.connect()를 통해 사전에 생성한 MySQL DB와 연결하고 연결한 객체의 cursor()를 통해 커서 객체를 가져온다.

커서 객체의 execute()를 통해 쿼리문을 실행하며, INSERT 쿼리를 통해 온/습도 측정 시간, 내부 온/습도 값, 환풍기 ON/OFF여부에 대한 값을 DB에 저장한다.

◆ 웹 서버

(1) Amazon RDS 연결

datasource bean을 생성할 때, url property에 RDS의 ARN 값을 넣어준다. mapper 파일에 작성한 sql을 통해 DB로부터 온도 및 습도 데이터를 가져와 JSON 형식으로 변환하여 jsp파일에서 googleChart를 통해 데이터를 보여준다. 데이터는 사용자가 선택한 날짜(24시간)의 온습도 데이터를 보여준다.

(2) 그래프 표현

GoogleChart API를 이용하여 데이터베이스에서 가져온 값들을 그래프로 표현한다. x축에 시간, y축에 온/습도를 표시하여 꺾은선그래프로 표현하고 특정 날짜 검색을 통해 특정 구간의 온/습도 그래프를 조회할 수 있다.

◆ 데이터베이스

웹 서버 AWS에서 생성한 RDS MySQL DB에서 스키마를 생성하고 스키마에서 온습도 데이터 테이블, 로그인에 필요한 테이블을 만든 뒤 다음과 같이 구성한다.

• 온습도 테이블

컬럼명	time	hud_in	temp_in	ison	serial number	hud_out	temp_out
의미	측정 시간	내부 습도	내부 온도	On/Off 여부	기기 번호	외부 습도	외부 온도
데이터 형식	datetime	float	float	boolean	varchar (100)	float	float

• 로그인 테이블

	Serialnumber	passwd
의미	기기 번호	비밀번호
데이터 형식	varchar(100)	varchar(100)

5. Prototype 구현

• 사전에 구현한 파이썬 프로그램을 저장해둔 라즈베리파이에 전원을 연결하여 전원을 공급

• 라즈베리파이에 전원 공급이 된 후 온습도 센서에서 측정한 데이터가 DB로 전송되는지 확인

• 내부 환경에 열 / 습도를 가했을 경우 자동으로 릴레이 모듈에 신호를 주어 환풍기가 켜지는지 확인

• 환풍기의 충분한 작동으로 인해 온/습도가 외부 온/습도에 비하여 낮아졌을 때 환풍기가 동작을 멈추는지 확인

• 웹 서버에 접속하여 로그인 후 db에 저장된 온습도 값을 제대로 받아왔는지 확인 / 그래프로 잘 표현되었는지 확인

6. 시험 / 테스트 결과


외부 환경과 내부 환경을 아래 사진과 같이 구현하였다. 외부환경에는 라즈베리파이와 외부 온/습도, 릴레이 모듈을 설치하였고 내부 환경에는 환풍기, 내부 온/습도 센서를 설치하였다. 실험을 위해 처음 내부와 외부의 온습도를 비교하고 외부에 입김 또는 드라이기로 열과 습도를 가해 내부 온습도를 외부보다 크게 높도록 조정하여 환풍기가 켜지는 것을 확인. 환풍기가 켜진 후 내부의 온습도가 떨어지는 것을 확인하고 내부 온습도가 외부 온/습도 대비 기준치보다 낮아질 때 환풍기가 꺼지는 것을 확인한다.

처음 전원을 켠 후 putty를 통해 온습도와 ON/OFF 상태를 출력하여 데이터를 확인해보았다. Temp와 Humidity의 첫 인자가 내부, 두 번째 인자가 외부이며 내부 온도가 외부 온도보다 0.9만큼 높기 때문에 환풍기가 켜졌다.

```
Temp=(23.6*,22.7*)    Humidity=(53.4%,55.2%)   State=1
```

환풍기가 켜지고 일정 시간이 지난 후 내부 온도가 외부 온도보다 0.2도 높아지는 순간 환풍기가 꺼지는 것을 확인했다.

```
Temp=(22.8*,22.6*)    Humidity=(54.2%,54.3%)   State=0
```

다음으로 내부에 인위적으로 습도를 가해 내부 습도를 크게 높였다. 그러자 환풍기가 켜지는 것을 확인했다.

```
Temp=(23.8*,23.9*)    Humidity=(80.5%,61.6%)   State=1
```

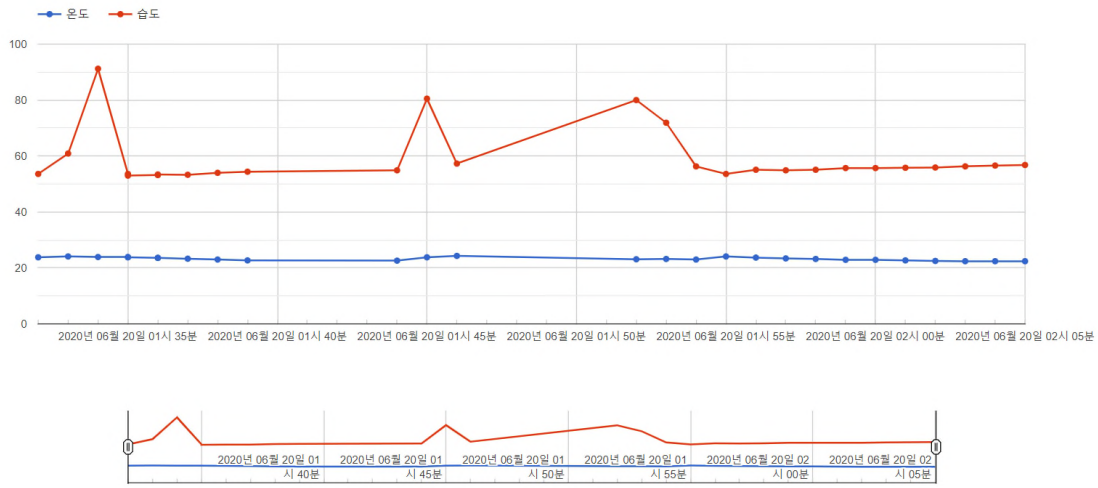
환풍기가 작동하고 시간이 지나자 내부 습도가 외부 습도 대비 기준치만큼 낮아졌고 환풍기가 꺼지는 조건을 만족하였다. 그러나 초기 습도를 높이는 과정에서 내부 온도도 높아졌고 내부 온도가 외부 온도 대비 기준치보다 높기 때문에 환풍기는 계속 작동한다.

```
Temp=(24.3*,23.9*)    Humidity=(58.8%,57.8%)   State=1
```

내부 온도가 외부 온도 대비 기준치보다 낮아지게 되자 환풍기가 꺼지는 것을 확인하였다.

```
Temp=(23.8*,23.7*)    Humidity=(53.2%,53.6%)   State=0
```

위의 과정에서 얻은 데이터들을 데이터베이스에 저장하고 웹 페이지에서 그래프를 확인한 결과는 다음과 같다.



7. Coding & DEMO

(1) 라즈베리파이

- 릴레이 모듈 4번 핀번호에 초기 설정

GPIO.setmode(GPIO.BCM)

```
GPIO.setup(4,GPIO.OUT)
```

- mysql 연결 후 커서 객체 가져옴

```
db = pymysql.connect(
host="rasberry1.cormbfdskqps.ap-northeast-2.rds.amazonaws.com",
port=3306, user="admin", passwd="rasberry", db="rasberry")
cursor = db.cursor()
```

- 측정시간 저장

```
nowTime = datetime.datetime.now()
nextMinute = nowTime.strftime("%M")
```

- 3번 핀에 연결된 내부 온습도 센서로부터 값 읽기

```
humidity_in, temperature_in = Adafruit_DHT.read_retry(Adafruit_DHT.DHT22,
3)
```

- 2번 핀에 연결된 외부 온습도 센서로부터 값 읽기

```
humidity_out, temperature_out =
Adafruit_DHT.read_retry(Adafruit_DHT.DHT22, 2)
```

- 내부 온도가 외부 온도+0.2보다 높거나 내부습도가 외부 습도+2보다 높을 경우 릴레이 모듈을 키고 반대의 경우 끔. 또한 내부 온습도가 각각 20도, 50% 이하일 경우 릴레이 모듈을 끈다.

```
if humidity_in > humidity_out+2 or temperature_in > temperature_out +0.2 :
        if temperature_in > 20 :
                GPIO.output(4,True)
                IsOn = 1
        else :
                GPIO.output(4,False)
                IsOn = 0
else:
        GPIO.output(4,False)
        isOn = 0
```

- 1분마다 데이터베이스에 측정값 저장

```
if preMinute != nextMinute:
insert_sql = 'insert into temp_hud values(%s, %s, %s, %s, %s, %s, %s)'
cursor.execute(insert_sql,                    (nowTime,temperature_in,
humidity_in,isOn,'testserial', temperature_out, humidity_out));
db.commit()
```

preMinute = nextMinute

(2) 웹 서버

웹 서버를 구축할 때 Spring 프레임워크와 Tomcat9.0을 사용하였고,
데이터베이스 접근용 프레임워크로 MyBatis 사용
프론트 화면 구현을 위해 GoogleChart와 Bootstrap사용

dataSource 객체에 AWS RDS MySQL8.0의 주소를 url property에 설정한다(rasberry1.cormbfdskqps.ap-northeast-2.rds.amazonaws.com), RDS 인스턴스에 탄력적 IP 주소를 할당하여 인스턴스를 껐다 켜도 주소가 바뀌지 않음.

dataSource 객체를 SqlSessionFactory 객체에 주입, SqlSessionTemplate에 SqlSessionfactory를 주입, DAO에 SqlSessionTemplate를 주입하여 사용한다.

context:component-scan을 이용하여 bean을 자동으로 주입
InternalResourceViewResolver가 Controller의 return 값에 prefix(/WEB-INF/views/), suffix(.jsp)를 이용하여 jsp파일을 찾도록 함

HttpSession session = req.getSession() 세션을 가져온다, 없으면 생성한다
RasberryloginVO login = rasberryloginService.login(vo) 사용자가 입력한 정보로 로그인 실행

session.setAttribute("member", null) login이 비정상이면
rttr.addFlashAttribute("msg", false) 실패 메시지를 전송하고 함수 종료

session.setAttribute("member", login); login이 정상이면 세션 설정후 함수 종료

DB에서 특정날짜의 데이터를 가져오는 SQL

```
<select id="selectAll" resultType="com.lsj.domain.RasberryVO">
   SELECT * from temp_hud ORDER BY time DESC
</select>
```

사용자가 입력한 아이디와 비밀번호가 맞는지 확인

```
<select id="login" resultType="com.lsj.domain.RasberryloginVO">
   SELECT serialnumber, passwd FROM rasberry WHERE serialnumber =
   #{serialnumber} AND passwd = #{passwd}
</select>
```

GoogleChart를 이용하기 위해서 데이터베이스로부터 데이터를 가져와 JSONArray, JSONObject를 이용하여 데이터의 형식을 json으로 변환 후, model객체를 사용해 jsp에서 데이터를 사용할 수 있도록 하였다

```
List<RasberryVO> temp = rasberryService.readRasberryList();

JSONArray jsonArray = new JSONArray();
JSONObject jsonObject = new JSONObject();

for (int i = 0; i < temp.size(); i++) {
        jsonObject.put("temp_in", temp.get(i).getTemp_in());
        jsonObject.put("hud_in", temp.get(i).getHud_in());
        jsonObject.put("temp_out", temp.get(i).getTemp_out());
        jsonObject.put("hud_out", temp.get(i).getHud_out());
        jsonObject.put("id", temp.get(i).getSerialNumber());
        jsonObject.put("time", temp.get(i).getTime());
        jsonObject.put("isOn", temp.get(i).getIsOn());

        jsonArray.add(jsonObject);
}

model.addAttribute("jsonList", jsonArray);
```

json으로 넘어온 데이터를 GoogleChart에 삽입하여 꺾은 선 그래프 형태로 사용자에게 정보를 제공한다. 데이터는 사용자가 날짜를 선택하면, 선택한 날짜의 온습도 데이터를 그래프로 확인할 수 있다.

```
var dataRow = [];
var jsonData = ${jsonList};

for(var ele in jsonData){
```

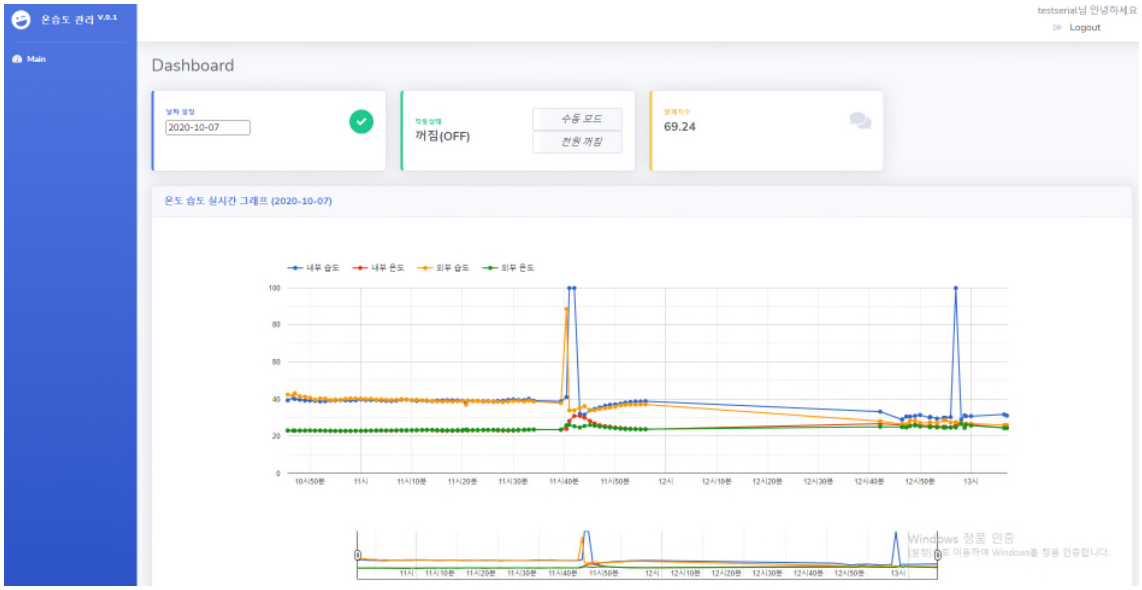
```
            var date = new Date();
            var timestamp = jsonData[ele].time.time;
            var date = new Date(timestamp);
            dataRow = [
            new Date(date.getFullYear(), date.getMonth(), date.getDate(), date.getHours(),
            date.getMinutes() ), jsonData[ele].temp_in, jsonData[ele].hud_in
            ];
            data.addRow(dataRow);
}
```

Ⅲ. 결론

1. 연구 결과 및 향후 연구과제

처음에 라즈베리파이의 전원을 연결하면 라즈베리파이가 연결할 수 있는 와이파이를 탐색하여 자동으로 연결한다. 연결에 성공하면 라즈베리파이에 코딩된 알고리즘에 따라 온/습도 센서를 통해 온/습도 값을 읽어온다. 온/습도 값을 읽어오는 과정에서 약간의 딜레이가 생기는 경우가 있었지만 계획 상의 1초에 한 번 온/습도를 읽어오는 것에 지장은 없었다. 읽어온 값을 라즈베리파이의 알고리즘에 따라 환풍기를 자동으로 제어하였고 온/습도 값과 환풍기 상태 등을 데이터베이스에 저장하였다.

데이터베이스는 웹사이트에 로그인하기 위해 환풍기의 시리얼 넘버와 환풍기의 패스워드로 구성된 로그인 테이블, 온/습도와 작동 상태를 확인할 수 있는 상태 테이블, 웹사이트에서 설정한 환풍기의 수동/자동 작동 모드를 확인할 수 있는 제어 테이블로 구성하였다. 웹사이트의 제품 등록 버튼을 통해 환풍기의 시리얼 넘버와 로그인할 때 사용할 패스워드를 입력하면 로그인 테이블에 저장된다. 그리고 라즈베리파이에서 온/습도 값과 환풍기의 작동 상태를 1분에 한 번씩 상태 테이블로 저장하고 있고 웹 사이트에서 설정한 환풍기의 자동/수동 모드를 제어 테이블에 저장하고 있으며 라즈베리파이의 알고리즘에서 제어 테이블의 값을 읽어와 제어 상태에 따라 알고리즘을 진행한다.

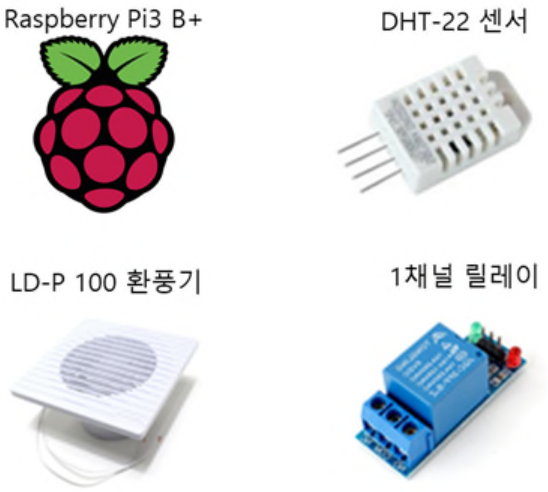


웹사이트는 아마존 RDS서버를 구축하여 설계한 것으로 주소만 알고 있으면 누구나 접속 가능하다. 처음 웹사이트에 접속하면 로그인 화면으로 이동하고 제품 등록(회원가입 역할) 버튼을 통해 제품을 등록하거나 로그인 할 수 있다. 위의 그림은 데이터베이스에 저장된 환풍기의 상태를 웹사이트에 로그인하여 확인한 모습이다. 날짜 설정을 통해 특정 날짜의 온/습도 그래프를 확인할 수 있으며 아래의 작은 그래프를 통해 확인할 시간 범위를 조정할 수 있다. 작동 상태 박스에서 환풍기의 작동 상태를 확인할 수 있으며 자동모드를 선택하면 라즈베리파이의 알고리즘에 따라 자동 모드로 환풍기가 작동하고 수동 모드를 선택하면 라즈베리파이의 알고리즘을 무시하고 웹 사이트에서 환풍기를 수동으로 키거나 끌 수 있다.

현재 작품에서 환풍기를 자동으로 작동시키기 위한 라즈베리파이 알고리즘으로 내부, 외부 온/습도를 비교하고 한겨울과 같이 추운 날씨에 작동하는 것을 방지하기 위해 특정 온도보다 낮을 시 작동하지 않도록 되어 있는데 주변 환경에 따라 더 스마트하게 환풍기를 작동시키기 위해서는 보다 세부적인 조건을 연구하여 환풍기 작동 알고리즘을 설정해야 할 것으로 보여진다. 또한 기존에 존재하는 스마트 환풍기보다 저비용, 저전력으로 환풍기를 작동시키는 것이 목적인데 1분마다 추가되는 각각의 환풍기에 대한 상태를 유지할 수 있는 대용량의 데이터베이스를 사용해야 한다는 점이 딜레마가 될 수 있으며 이에 대한 대책을 연구해야 할 것으로 보인다.

2. 작품 제작 소요 재료 목록

작품을 설계하면서 소요된 재료로는 환풍기와 라즈베리 파이, 라즈베리파이를 통해 환풍기를 제어할 수 있는 1채널 릴레이, 온습도 값을 읽어 라즈베리파이에 전달할 수 있도록 DHT-22온습도 센서를 사용하였고, 물질적인 것을 제외하면 환풍기의 상태를 저장할 수 있는 데이터베이스 서버와 웹사이트를 구성하기 위한 아마존 RDS 서버를 사용하였다.



참고자료

환풍기 전력 사용량(https://blog.kepco.co.kr/82)

AWS 개발자 안내서
(https://docs.aws.amazon.com/ko_kr/iot/latest/developerguide/sdk-tutorials.html)

RasberryPi 공식 사이트(https://www.raspberrypi.org/help/)

Pymysql(https://pymysql.readthedocs.io/en/latest/modules/)

GoogleChart 가이드(https://developers.google.com/chart)

Bootstrap 문서(https://getbootstrap.com/docs/4.5/getting-started/introduction/)