



미니 스마트팜

팀원1, 팀원2, 정석진

발표 순서 |

미니 스마트팜 프로젝트

01 프로젝트 개요

02 아두이노

03 GUI

04 소감





프로젝트 개요





E사 제품

110,000원

적당한 가격

낮은 내구성

우노 미포함

작은 화분

수분 자동공급 불가



R사 제품

583,000원

높은 가격

높은 내구성

무선통신 및 확장성

작은 화분

수분 자동공급 불가



F사 제품

61,000원

낮은 가격

높은 내구성

영양제 의존성

짙은 워터펌프 청소

온도 조절 불가

프로젝트 개요

프로젝트 목적

기능에 비해서 너무 비싸거나, 결함이 있다.

작물재배의 기초원리는 어렵지 않다.

직접 만들어도 이것보다는 기능이 좋을것이다.

프로젝트 개요 |

프로젝트 목적

기능에 비해서 너무 비싸거나, 결함이 있다.

적정한 가격에 제대로 된 기능을
갖는

작물재배의 기초를 제공하는 어렵지 않다.

직접 만들어도 이것만으로는 부족하다.
'미니 스마트팜' 만들기



품명 : 12V 환풍팬
개수 : 2개
가격 : 4,400원



품명 : 5V 워터펌프
개수 : 2개
가격 : 2,400원



품명 : 수위센서
개수 : 1개
가격 : 400원



품명 : 토양수분센서
개수 : 1개
가격 : 6,000원



품명 : 라즈베리파이 4b
& 터치스크린
개수 : 1개
가격 : 200,000원



품명 : 생육용 LED
개수 : 1개
가격 : 12,900원



품명 : 상추 씨앗
개수 : 1세트
가격 : 1,000원

아두이노 | 재료



품명 : 아두이노 우노
개수 : 1개
가격 : 32,000원



품명 : DHT22
개수 : 1개
가격 : 6,600원



품명 : 1.5V 건전지
개수 : 8구
가격 : 2,000원



품명 : 4구 건전지 케이스(6V)
개수 : 2개
가격 : 1,800원



품명 : 브레드보드
개수 : 1개
가격 : 2,000원



품명 : 스펀지
개수 : 1개
가격 : 2,000원



품명 : 수납용 상자(65L)
개수 : 1개
가격 : 18,900원



품명 : 모종 트레이 & 화분
개수 : 1세트
가격 : 18,900원



품명 : 공급&배출 물통
개수 : 2개
가격 : -원



품명 : 원예용 상토
개수 : 4개
가격 : 4,000원



품명 : 워터펌프 호스(1m)
개수 : 1개
가격 : 3,000원



품명 : 보조배터리
개수 : 1개
가격 : 5,000원





품명 : 12V 환풍팬
개수 : 2개
가격 : 4,400원



품명 : 5V 워터펌프
개수 : 2개
가격 : 2,400원



품명 : 수위센서
개수 : 1개
가격 : 400원



품명 : 토양수분센서
개수 : 1개
가격 : 6,000원



품명 : 라즈베리파이 4b
& 터치스크린
개수 : 1개
가격 : 200,000원



품명 : 생육용 LED
개수 : 1개
가격 : 12,900원



품명 : 상추 씨앗
개수 : 1세트
가격 : 1,000원



품명 : 아두이노 우노
개수 : 1개
가격 : 32,000원



품명 : 1.5V 건전지
개수 : 8구
가격 : 2,000원



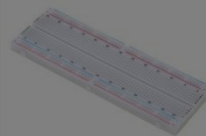
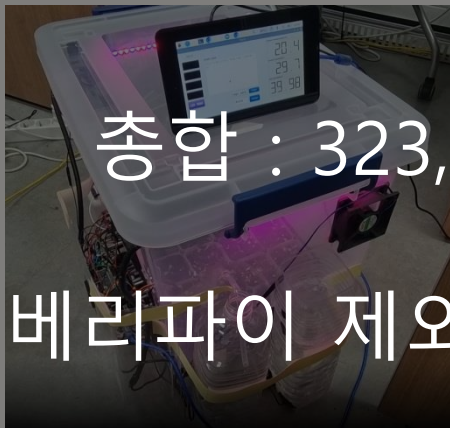
품명 : DHT22
개수 : 1개
가격 : 6,600원



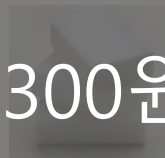
품명 : 4구 건전지 케이스(6V)
개수 : 2개
가격 : 1,800원

총합 : 323,300원

라즈베리파이 제외 : 123,300원



품명 : 브레드보드
개수 : 1개
가격 : 2,000원



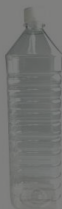
품명 : 스펀지
개수 : 1개
가격 : 2,000원



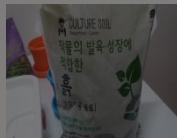
품명 : 수납용 상자(65L)
개수 : 1개
가격 : 18,900원



품명 : 모종 트레이 & 화분
개수 : 1세트
가격 : 18,900원



품명 : 공급&배출 물통
개수 : 2개
가격 : -원



품명 : 원예용 상토
개수 : 4개
가격 : 4,000원



품명 : 워터펌프 호스(1m)
개수 : 1개
가격 : 3,000원



품명 : 보조배터리
개수 : 1개
가격 : 5,000원



프로젝트 개요

프로젝트 구성

식물 생육장치 구성요소



센서 · 모터 · LED · 아두이노 · 식물 트레이

GUI 장치 구성요소

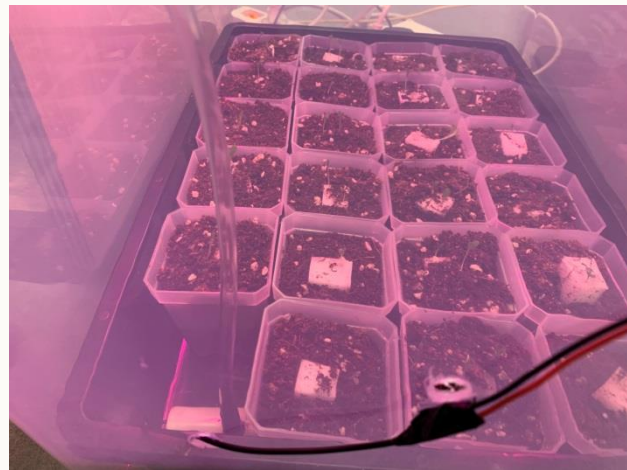


터치스크린 · 라즈베리파이 · 파이썬



프로젝트 개요 |

상추 생장 일지



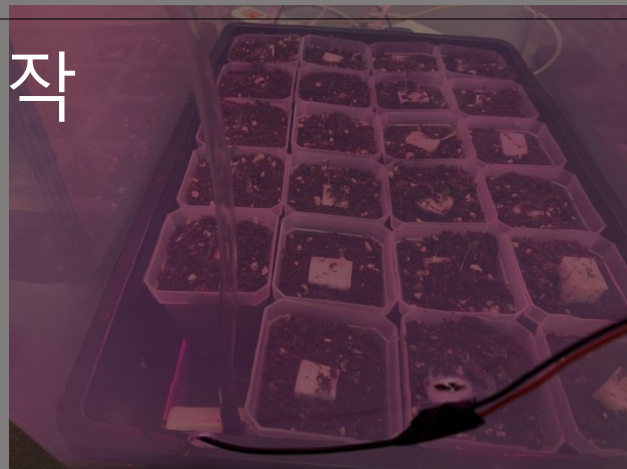
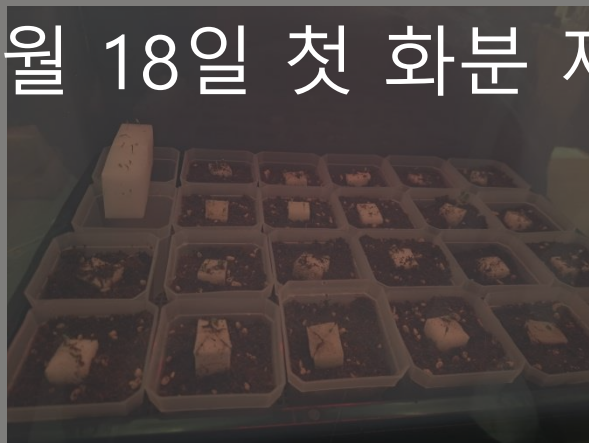


프로젝트 개요 |

상추 생장 일지



11월 18일 첫 화분 제작

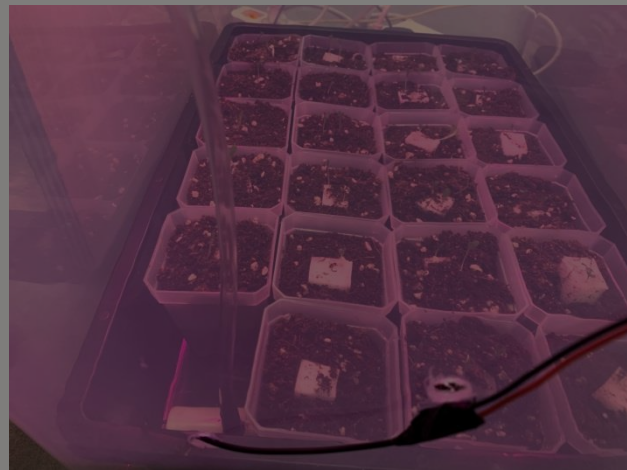
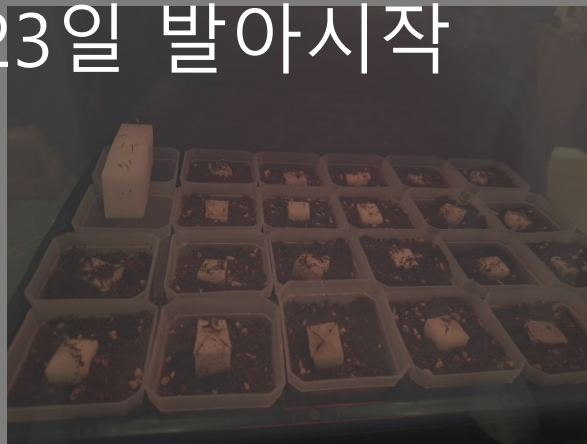


프로젝트 개요

상추 생장 일지

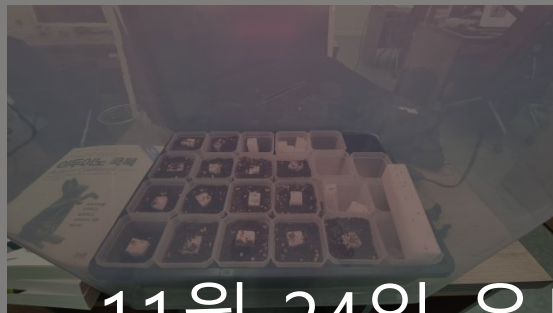


11월 23일 발아시작

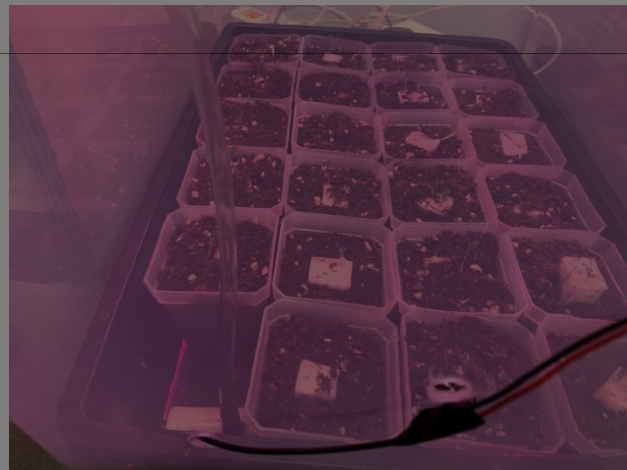


프로젝트 개요 |

상추 생장 일지



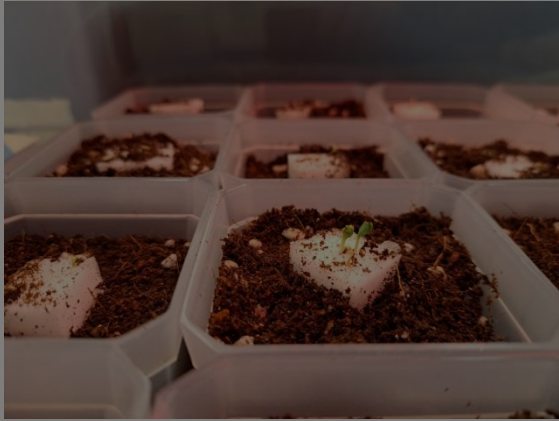
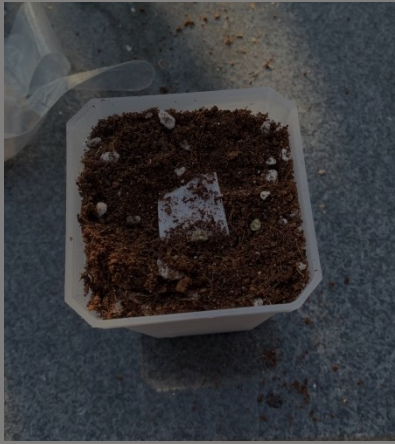
11월 24일 유묘 사진



프로젝트 개요 |

상추 생장 일지

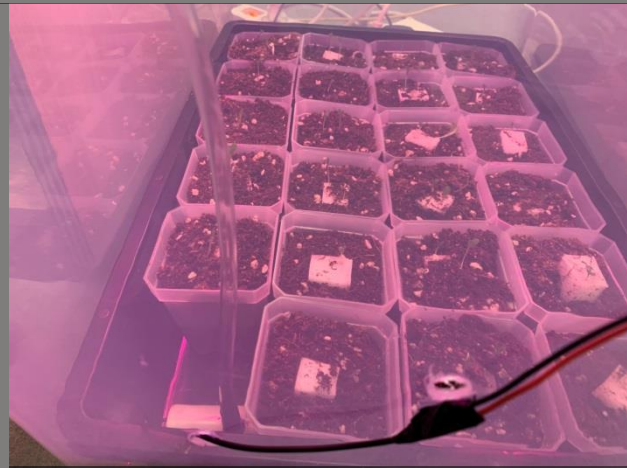
11월 25일 전체 발아 완료



프로젝트 개요 |

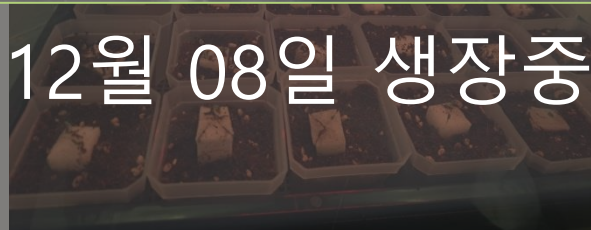
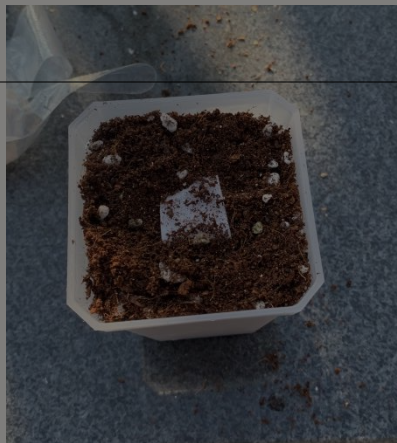
상추 생장 일지

11월 29일 생장중

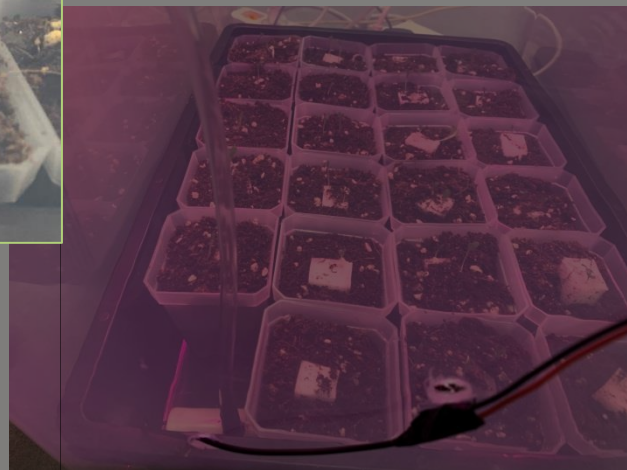


프로젝트 개요

상추 생장 일지



12월 08일 생장중



프로젝트 개요 |

스마트팜 실작동 영상



아두이노



아두이노 |

목차

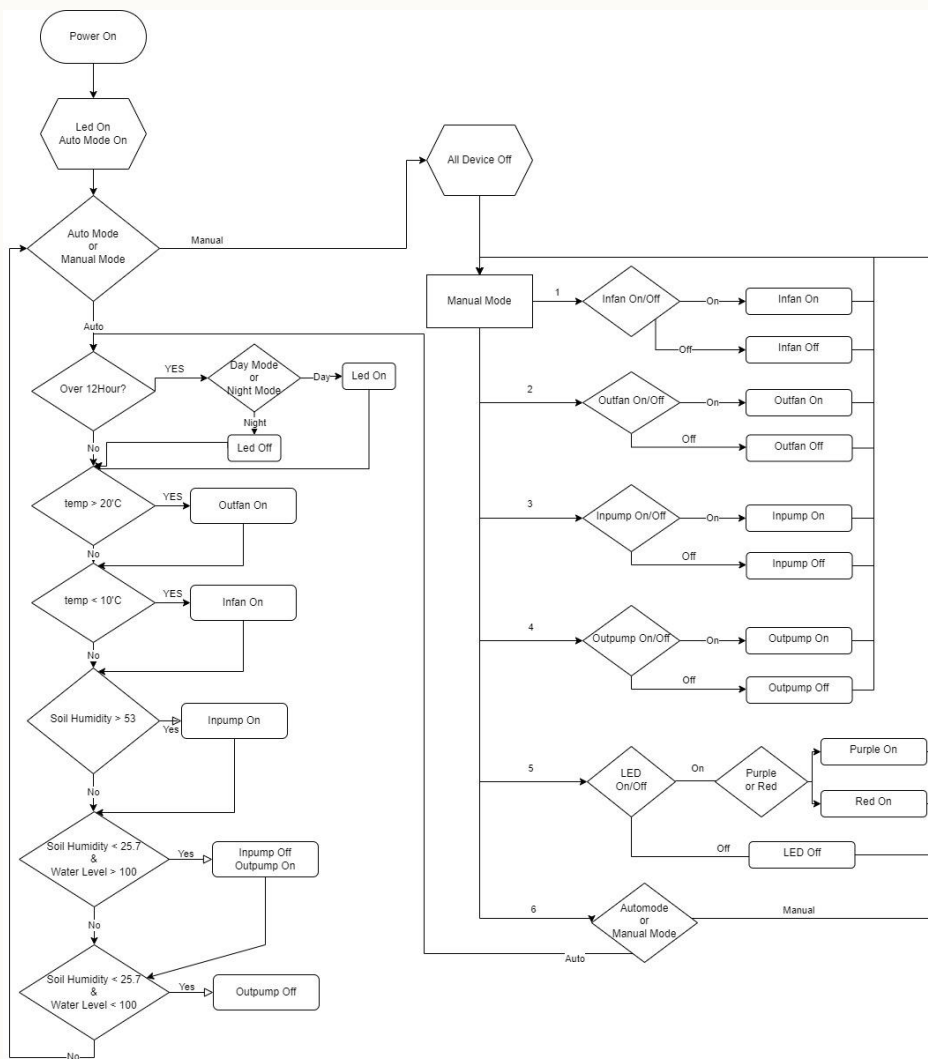
01 아두이노 순서도

02 Code



아두이노 |

Flowchart



특징

1. 초기설정 값 : 주간모드(LED 보라색 On), 자동제어 모드 On
2. 6번을 통해 Auto & Manual 전환 가능
3. Auto Mode -> Manual Mode 전환 시 모든 장치가 Off 된다.

아두이노

변수선언

```
1  #include <DHT.h>
2  #define DHTTYPE DHT22
3
4  float Humi,Temp; //온습도 센서값
5  float SoilHumi = 0; // 토양수분센서값
6  float SoilHumi_Per = 0;
7  int WTLV = 0; // 수위값
8
9  int ACInRelayPin = 4; // 흡기 시그널핀
10 int ACOutRelayPin = 5; // 배기 시그널핀
11 int WTInRelayPin = 7; // 수분공급 시그널핀
12 int WTLV = 6; //수분배출 시그널핀
13 int HTS = 8;
14 int LEDPin = 9;
15 int LED_Color_Pin = 3;
16 int LED_Color = 0;
17
18 unsigned long pro_Time = 0; // 딜레이없는 딜레이 구현용
19 int WT = 1000; // 1초마다 가동
20 unsigned long RevertTime = 43200000; // 12시간
21 unsigned long StartTime = millis();
```

SoilHumi => 토양수분센서(0~1023)

SoilHumi_Per : 저항값을 퍼센트로 저장

WTLV : 수위센서 (0~1023)



아두이노

변수선언

```
1  #include <DHT.h>
2  #define DHTTYPE DHT22
3
4  float Humi,Temp; //온습도 센서값
5  float SoilHumi = 0; // 토양수분센서값
6  float SoilHumi_Per = 0;
7  int WTLV = 0; // 수위값
8
9  int ACInRelayPin = 4; // 흡기 시그널핀
10 int ACOutRelayPin = 5; // 배기 시그널핀
11 int WTInRelayPin = 7; // 수분공급 시그널핀
12 int WTLV = 0; // 수위값
13 int HTS = 8;
14 int LEDPin = 9;
15 int LED_Color_Pin = 3;
16 int LED_Color = 0;
17
18 unsigned long pro_Time = 0; // 딜레이없는 딜레이 구현용
19 int WT = 1000; // 1초마다 가동
20 unsigned long RevertTime = 43200000; // 12시간
21 unsigned long StartTime = millis();
```

LEDPin : 릴레이모듈에 연결, ON/OFF조절

LED_Color_Pin: 색상조절을 위한 시그널핀

LED_Color(0~2): LED의 상태를 조절하는 토글



아두이노

변수선언

```
1  #include <DHT.h>
2  #define DHTTYPE DHT22
3
4  float Humi,Temp; //온습도 센서값
5  float SoilHumi = 0; // 토양수분센서값
6  float SoilHumi_Per = 0;
7  int WTLV = 0; // 수위값
8
9  int ACInRelayPin = 4; // 흡기 시그널핀
10 int ACOutRelayPin = 5; // 배기 시그널핀
11 int WTInRelayPin = 7; // 수분공급 시그널핀
12 int WTLV = 0; // 수위값
13 int HTS = 8;
14 int LEDPin = 9;
15 int LED_Color_Pin = 3;
16 int LED_Color = 0;
17
18 unsigned long pro_Time = 0; // 딜레이없는 딜레이 구현용
19 int WT = 1000; // 1초마다 가동
20 unsigned long RevertTime = 43200000; // 12시간
21 unsigned long StartTime = millis();
```

1초마다 실행하기 위해 설정해 놓은 변수



아두이노

변수선언

```
23 char SLOD; //Serial OrderData
24
25 DHT temp(HTS, DHTTYPE);
26
27 //매뉴얼 모드 토글구현용 TF
28 bool AutoFan = true;
29 bool InFan = false;
30 bool OutFan = false;
31 bool InPump = false;
32 bool OutPump = false;
33
34 //이벤트 발생계산용 변수
35 int EventValue = 0;
36
37
38 //낮모드 밤모드
39 bool DN_Mode = true;
```

DHT 센서값을 읽어오기 위한 준비



아두이노

변수선언

```
23 char SLOD; //Serial OrderData
24
25 DHT temp(HTS, DHTTYPE);
26
27 //매뉴얼 모드 토글구현용 TF
28 bool AutoFan = true;
29 bool InFan = false;
30 bool OutFan = false;
31 bool InPump = false;
32 bool OutPump = false;
33
34 //이벤트 발생계산용 변수
35 int EventValue = 0;
36
37
38 //낮모드 밤모드
39 bool DN_Mode = true;
```

자동모드와 수동모드를 구분하는 토글



아두이노

변수선언

```
23 char SLOD; //Serial OrderData
24
25 DHT temp(HTS, DHTTYPE);
26
27 //매뉴얼 모드 토글구현용 TF
28 bool AutoFan = true;
29 bool InFan = false;
30 bool OutFan = false;
31 bool InPump = false;
32 bool OutPump = false;
33
34 //이벤트 발생계산용 변수
35 int EventValue = 0;
36
37
38 //낮모드 밤모드
39 bool DN_Mode = true;
```

센서들 On/Off 토글 설정



아두이노

void setup()

```
41 void setup()  
42 {  
43   temp.begin(); // 온도센서값 읽을준비됨  
44   Serial.begin(9600); //비트레이트 9600으로 시리얼 통신 하겠다  
45   pinMode(A3,INPUT); // 토양수분센서 핀으로 입력받겠다  
46   pinMode(A2,INPUT); // 수위센서 핀으로 입력 받겠다  
47  
48   pinMode(ACInRelayPin, OUTPUT);  
49   pinMode(ACOutRelayPin, OUTPUT);  
50   pinMode(WTInRelayPin, OUTPUT);  
51   pinMode(WTOutRelayPin, OUTPUT);  
52   pinMode(LEDPin,OUTPUT);  
53   digitalWrite(LEDPin,HIGH);  
54   pinMode(LED_Color_Pin,OUTPUT);  
55 }
```

센서, 펌프, 환풍팬 출력 설정



아두이노

void setup()

```
41 void setup()  
42 {  
43     temp.begin(); // 온도센서값 읽을준비됨  
44     Serial.begin(9600); //비트레이트 9600으로 시리얼 통신 하겠다  
45     pinMode(A3,INPUT); // 토양수분센서 핀으로 입력받겠다  
46     pinMode(A2,INPUT); // 수위센서 핀으로 입력 받겠다  
47  
48     pinMode(ACInRelayPin, OUTPUT);  
49     pinMode(ACOutRelayPin, OUTPUT);  
50     pinMode(WTInRelayPin, OUTPUT);  
51     pinMode(WTOutRelayPin, OUTPUT);  
52     pinMode(LEDPin,OUTPUT);  
53     digitalWrite(LEDPin,HIGH);  
54     pinMode(LED_Color_Pin,OUTPUT);  
55 }
```

LED를 주간모드로 시작



```

56 void loop()
57 {
58     if(millis() > pro_Time + WT){
59
60         SoilHumi_Per = 0 ;
61         EventValue = 0; //이벤트 변수 초기화
62         Humi = temp.readHumidity(); // 습도읽어서 저장
63         Temp = temp.readTemperature(); // 온도읽어서 저장
64         SoilHumi = analogRead(A3); // 토양수분센서값 읽어서 저장
65         WTLV = analogRead(A2); // 수위센서값 읽어서 저장
66
67         if(Serial.available()){ // 만일 컴퓨터로부터 값을 받았다면 == 매뉴얼모드
68             SLOD = Serial.read();
69             if(AutoFan == false){ // 매뉴얼 모드일때만 12345 값을 입력받는다.
70
71
72                 if(SLOD == '1'){
73                     if(InFan == false){
74                         digitalWrite(ACInRelayPin,HIGH);
75                         InFan = true;
76
77                     }
78
79                     else if(InFan == true){
80                         digitalWrite(ACInRelayPin,LOW);
81                         InFan = false;
82
83                     }
84
85                 }
86
87             }
88
89             else if(SLOD == '2'){
90                 if(OutFan == false){
91                     digitalWrite(ACOutRelayPin,HIGH);
92                     OutFan = true;
93
94                 }
95                 else if(OutFan == true){
96                     digitalWrite(ACOutRelayPin,LOW);
97                     OutFan = false;
98
99                 }
100             }
101         }
102     }
103 }

```

1초마다 작동하도록 설정

수동조작 모드일 때 센서제어

아두이노

void loop()



```
void loop()
```

```
57 {  
58     if(millis() > pro_Time + WT){  
59  
60         SoilHumi_Per = 0 ;  
61         EventValue = 0; //이벤트 변수 초기화  
62         Humi = temp.readHumidity(); // 습도읽어서 저장  
63         Temp = temp.readTemperature(); // 온도읽어서 저장  
64         SoilHumi = analogRead(A3); // 토양수분센서값 읽어서 저장  
65         WTLV = analogRead(A2); // 수위센서값 읽어서 저장  
66  
67         if(Serial.available()){ // 만일 컴퓨터로부터 값을 받았다면 == 매뉴얼모드  
68             SLOD = Serial.read();  
69             if(AutoFan == false){ // 매뉴얼 모드일때만 12345 값을 입력받는다.  
70  
71  
72                 if(SLOD == '1'){  
73                     if(InFan == false){  
74                         digitalWrite(ACInRelayPin,HIGH);  
75                         InFan = true;  
76  
77                     }  
78  
79                     else if(InFan == true){  
80                         digitalWrite(ACInRelayPin,LOW);  
81                         InFan = false;  
82  
83                     }  
84                 }  
85  
86             }  
87  
88             else if(SLOD == '2'){  
89                 if(OutFan == false){  
90                     digitalWrite(ACOutRelayPin,HIGH);  
91                     OutFan = true;  
92  
93                 }  
94                 else if(OutFan == true){  
95                     digitalWrite(ACOutRelayPin,LOW);  
96                     OutFan = false;  
97  
98                 }  
99             }  
}
```

1초마다 작동하도록 설정

1번을 입력받은 경우 : 흡기팬 작동/

아두이노

void loop()



```
void loop()
```

```
57 {  
58     if(millis() > pro_Time + WT){  
59  
60         SoilHumi_Per = 0 ;  
61         EventValue = 0; //이벤트 변수 초기화  
62         Humi = temp.readHumidity(); // 습도읽어서 저장  
63         Temp = temp.readTemperature(); // 온도읽어서 저장  
64         SoilHumi = analogRead(A3); // 토양수분센서값 읽어서 저장  
65         WTLV = analogRead(A2); // 수위센서값 읽어서 저장  
66  
67         if(Serial.available()){ // 만일 컴퓨터로부터 값을 받았다면 == 매뉴얼모드  
68             SLOD = Serial.read();  
69             if(AutoFan == false){ // 매뉴얼 모드일때만 12345 값을 입력받는다.  
70  
71  
72                 if(SLOD == '1'){  
73                     if(InFan == false){  
74                         digitalWrite(ACInRelayPin,HIGH);  
75                         InFan = true;  
76  
77  
78                     }  
79  
80                     else if(InFan == true){  
81                         digitalWrite(ACInRelayPin,LOW);  
82                         InFan = false;  
83  
84                     }  
85  
86                 }  
87  
88                 else if(SLOD == '2'){  
89                     if(OutFan == false){  
90                         digitalWrite(ACOutRealyPin,HIGH);  
91                         OutFan = true;  
92  
93                     }  
94                     else if(OutFan == true){  
95                         digitalWrite(ACOutRealyPin,LOW);  
96                         OutFan = false;  
97  
98                     }  
99                 }  
}
```

1초마다 작동하도록 설정

2번을 입력받은 경우 : 배기팬 작동/중지

아두이노

void loop()



아두이노

void loop()

3번을 입력받은 경우 : 공급펌프 작동/중지

```
100     else if(SLOD == '3'){
101         if(InPump == false){
102             digitalWrite(WTInRelayPin,HIGH);
103             InPump = true;
104         }
105         else if(InPump == true){
106             digitalWrite(WTInRelayPin,LOW);
107             InPump = false;
108         }
109     }
110     else if(SLOD == '4'){
111         if(OutPump == false){
112             digitalWrite(WTOutRelayPin,HIGH);
113             OutPump = true;
114         }
115         else if(OutPump == true){
116             digitalWrite(WTOutRelayPin,LOW);
117             OutPump = false;
118         }
119     }
120 }
121 if(SLOD == '5'){
122     LED_Color++;
123     if(LED_Color > 2){
124         LED_Color = 0;
125     }
126     switch(LED_Color){
127         case 0 : digitalWrite(LEDPin,LOW); break;
128         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
129         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
130         default : break;
131     }
132 }
133
134 if(SLOD == '6'){
135     AutoFan = !AutoFan; //토글
136     digitalWrite(ACOutRealyPin,LOW);
137     digitalWrite(ACInRelayPin,LOW);
138     switch(LED_Color){
139         case 0 : digitalWrite(LEDPin,LOW); break;
140         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
141         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
142         default : break;
143     }
144 }
```



아두이노

void loop()

```
100 else if(SLOD == '3'){
101     if(InPump == false){
102         digitalWrite(WTInRelayPin,HIGH);
103         InPump = true;
104     }
105     else if(InPump == true){
106         digitalWrite(WTInRelayPin,LOW);
107         InPump = false;
108     }
109 }
110 else if(SLOD == '4'){
111     if(OutPump == false){
112         digitalWrite(WTOutRelayPin,HIGH);
113         OutPump = true;
114     }
115     else if(OutPump == true){
116         digitalWrite(WTOutRelayPin,LOW);
117         OutPump = false;
118     }
119 }
120 }
121 if(SLOD == '5'){
122     LED_Color++;
123     if(LED_Color > 2){
124         LED_Color = 0;
125     }
126     switch(LED_Color){
127         case 0 : digitalWrite(LEDPin,LOW); break;
128         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
129         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
130         default : break;
131     }
132 }
133
134 if(SLOD == '6'){
135     AutoFan = !AutoFan; //토글
136     digitalWrite(ACOutRelayPin,LOW);
137     digitalWrite(ACInRelayPin,LOW);
138     switch(LED_Color){
139         case 0 : digitalWrite(LEDPin,LOW); break;
140         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
141         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
142         default : break;
143     }
144 }
```

4번을 입력받은 경우 : 배출펌프 작동/중지



아두이노

void loop()

```
100 else if(SLOD == '3'){
101     if(InPump == false){
102         digitalWrite(WTInRelayPin,HIGH);
103         InPump = true;
104     }
105     else if(InPump == true){
106         digitalWrite(WTInRelayPin,LOW);
107         InPump = false;
108     }
109 }
110 else if(SLOD == '4'){
111     if(OutPump == false){
112         digitalWrite(WTOutRelayPin,HIGH);
113         OutPump = true;
114     }
115     else if(OutPump == true){
116         digitalWrite(WTOutRelayPin,LOW);
117         OutPump = false;
118     }
119 }
120 }
121 if(SLOD == '5'){
122     LED_Color++;
123     if(LED_Color > 2){
124         LED_Color = 0;
125     }
126     switch(LED_Color){
127         case 0 : digitalWrite(LEDPin,LOW); break;
128         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
129         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
130         default : break;
131     }
132 }
133
134 if(SLOD == '6'){
135     AutoFan = !AutoFan; //토글
136     digitalWrite(ACOutRealyPin,LOW);
137     digitalWrite(ACInRelayPin,LOW);
138     switch(LED_Color){
139         case 0 : digitalWrite(LEDPin,LOW); break;
140         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
141         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
142         default : break;
143     }
144 }
```

5번을 입력받은 경우 : LED 모드변경



아두이노

void loop()

```
100 else if(SLOD == '3'){
101     if(InPump == false){
102         digitalWrite(WTInRelayPin,HIGH);
103         InPump = true;
104     }
105     else if(InPump == true){
106         digitalWrite(WTInRelayPin,LOW);
107         InPump = false;
108     }
109 }
110 else if(SLOD == '4'){
111     if(OutPump == false){
112         digitalWrite(WTOutRelayPin,HIGH);
113         OutPump = true;
114     }
115     else if(OutPump == true){
116         digitalWrite(WTOutRelayPin,LOW);
117         OutPump = false;
118     }
119 }
120 }
121 if(SLOD == '5'){
122     LED_Color++;
123     if(LED_Color >2){
124         LED_Color = 0;
125     }
126     switch(LED_Color){
127         case 0 : digitalWrite(LEDPin,LOW); break;
128         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
129         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
130         default : break;
131     }
132 }
133 }
134 if(SLOD == '6'){
135     AutoFan = !AutoFan; //토글
136     digitalWrite(ACOutRealyPin,LOW);
137     digitalWrite(ACInRelayPin,LOW);
138     switch(LED_Color){
139         case 0 : digitalWrite(LEDPin,LOW); break;
140         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
141         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
142         default : break;
143     }
144 }
```

6번을 입력받은 경우 : 제어모드 변경



아두이노

void loop()

```
100 else if(SL0D == '3'){
101     if(InPump == false){
102         digitalWrite(WTInRelayPin,HIGH);
103         InPump = true;
104     }
105     else if(InPump == true){
106         digitalWrite(WTInRelayPin,LOW);
107         InPump = false;
108     }
109 }
110 else if(SL0D == '4'){
111     if(OutPump == false){
112         digitalWrite(WTOutRelayPin,HIGH);
113         OutPump = true;
114     }
115     else if(OutPump == true){
116         digitalWrite(WTOutRelayPin,LOW);
117         OutPump = false;
118     }
119 }
120 }
121 if(SL0D == '5'){
122     LED_Color++;
123     if(LED_Color > 2){
124         LED_Color = 0;
125     }
126     switch(LED_Color){
127         case 0 : digitalWrite(LEDPin,LOW); break;
128         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
129         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
130         default : break;
131     }
132 }
133 }
134 if(SL0D == '6'){
135     AutoFan = !AutoFan; //토글
136     digitalWrite(ACOutRealyPin,LOW);
137     digitalWrite(ACInRelayPin,LOW);
138     switch(LED_Color){
139         case 0 : digitalWrite(LEDPin,LOW); break;
140         case 1 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,0); break;
141         case 2 : digitalWrite(LEDPin,HIGH); digitalWrite(LED_Color_Pin,1); break;
142         default : break;
143     }
144 }
```

Auto -> Manual이 될 때 이전 LED의 상태를 불러오기 위한 장치



아두이노

void loop()

자동모드에서 12시간마다 LED제어

```
148 //주간이면 true 야간이면 false
149 if(AutoFan == true){
150     if(DN_Mode == false){
151         digitalWrite(LEDPin,LOW);
152     }
153     else{
154         digitalWrite(LEDPin,HIGH);
155         digitalWrite(LED_Color_Pin,0);
156     }
157 }
158
159 //AutoFan 이 true == 자동제어모드
160 if(AutoFan == true){ //자동제어 모드일 경우
161     if(Temp > 20){ //생육적온 최대20도
162         digitalWrite(ACOutRealyPin,HIGH);
163         digitalWrite(ACInRelayPin,LOW);
164         EventValue = EventValue +1;
165     }
166     if(10 >= Temp){ //생육적온 최소10도
167         digitalWrite(ACInRelayPin,HIGH);
168         digitalWrite(ACOutRealyPin,LOW);
169         EventValue = EventValue +1;
170     }
171     //온도정상, 습도 비정상이라면
172     if(Humi > 65){ //상대습도 65미만으로 설정
173         digitalWrite(ACOutRealyPin,HIGH);
174         EventValue = EventValue +2;
175     }
176     //온도, 습도 모두 정상이면 팬 정지
177     if(EventValue == 0){
178         digitalWrite(ACInRelayPin,LOW);
179         digitalWrite(ACOutRealyPin,LOW);
180     }
181 }
```



아두이노

void loop()

```
148 //주간이면 true 야간이면 false
149 if(AutoFan == true){
150     if(DN_Mode == false){
151         digitalWrite(LEDPin,LOW);
152     }
153     else{
154         digitalWrite(LEDPin,HIGH);
155         digitalWrite(LED_Color_Pin,0);
156     }
157 }
158
```

```
159 //AutoFan 이 true == 자동제어모드
160 if(AutoFan == true){//자동제어 모드일 경우
161     if(Temp > 20){ //생육적온 최대20도
162         digitalWrite(ACOutRealyPin,HIGH);
163         digitalWrite(ACInRelayPin,LOW);
164         EventValue = EventValue +1;
165     }
166     if(10 >= Temp){//생육적온 최소10도
167         digitalWrite(ACInRelayPin,HIGH);
168         digitalWrite(ACOutRealyPin,LOW);
169         EventValue = EventValue +1;
170     }
171     //온도정상, 습도 비정상이라면
172     if(Humi > 65){ //상대습도 65미만으로 설정
173         digitalWrite(ACOutRealyPin,HIGH);
174         EventValue = EventValue +2;
175     }
176     //온도, 습도 모두 정상이면 팬 정지
177     if(EventValue == 0){
178         digitalWrite(ACInRelayPin,LOW);
179         digitalWrite(ACOutRealyPin,LOW);
180     }
181 }
```

자동모드에서 Fan 제어

WHY?

Q. 온도를 10~20도 사이에 설정한 이유?
상추의 최적 발육온도가 10~20도이기 때문

Q. 습도를 65%이하로 설정한 이유?
과습 할 경우, 증산에 방해가 되며
병충해의 원인이 될 수 있기 때문



```

182 //760이상으로 건조하다면
183 ① if(SoilHumi > 760){
184     digitalWrite(WTInRelayPin,HIGH);
185     EventValue = EventValue +4; // 펌프이벤트 발생
186 }
187 ② else if(SoilHumi<480 && WTLV >100){ // 480 이하로 습하고, 수위가 100 이상일 때
188     digitalWrite(WTInRelayPin,LOW);
189     digitalWrite(WTOutRelayPin,HIGH);
190 }
191 ③ else if(SoilHumi<480 && WTLV <100) // 480 이하로 습하고, 수위가 100 이하일 때
192     digitalWrite(WTOutRelayPin,LOW);
193
194
195 }
196 if(millis() >= (StartTime+RevertTime)){
197     DN_Mode = !DN_Mode;
198     StartTime = millis();
199     EventValue = 9;
200 }
201 //아두이노에서 시리얼통신으로 값 보내는 내역
202
203 SoilHumi_Per = (1023-SoilHumi)/1023 * 100;
204
205 Serial.print(Temp);
206 Serial.print(',');
207 Serial.print(Humi);
208 Serial.print(',');
209 Serial.print(SoilHumi_Per);
210 Serial.print(',');
211 Serial.print(WTLV);
212 Serial.print(',');
213 Serial.println(EventValue);
214 //EventValue 해석 : 온도변화 1 습도변화 2 물공급변화 4, 온도와 습도 = 3, 온도와 펌프제어 = 5,
215 // 습도와 물공급 6, 전부터 7 주야간 전환 9
216
217 pro_Time = millis();
218 }
219 }

```

아두이노

void loop()

1. 토양이 건조할 경우 수분을 공급하고,
2. 충분히 적신 이후에 물을 배출하는 방식
3. 수위가 설정수위 이하로 내려갈 때 배출펌프 Off

WHY?

Q. 왜 SoilHumi>760, SoilHumi<480인가?
실제 측정해 본 결과. 건조한 흙과, 습한 흙의 값이 각각 760, 480 이었다.

Q. 왜 수위센서의 기준을 100으로 했는가?
수위센서의 측정값의 범위는 (0~1023)인데, 수위센서의 하단 일부만 닿아도 측정값이 600으로 측정되었다.

또한, 물에서 뿔 때는 곧바로 0이 되는 것이 아닌 100이하의 값이 나와서 100이라는 숫자를 물이 '있다'와 '없다'의 구분 기준으로 삼았다..



아두이노

void loop()

```
182 //760이상으로 건조하다면
183 if(SoilHumi > 760){
184     digitalWrite(WTInRelayPin,HIGH);
185     EventValue = EventValue +4; // 펌프이벤트 발생
186 }
187 else if(SoilHumi<480 && WTLV >100){ // 480 이하로 습하고, 수위가 100 이상일 때
188     digitalWrite(WTInRelayPin,LOW);
189     digitalWrite(WTOutRelayPin,HIGH);
190 }
191 else if(SoilHumi<480 && WTLV <100) // 480 이하로 습하고, 수위가 100 이하일 때
192     digitalWrite(WTOutRelayPin,LOW);
193
194
195 }
196 if(millis() >= (StartTime+RevertTime)){
197     DN_Mode = !DN_Mode;
198     StartTime = millis();
199     EventValue = 9;
200 }
201 //아두이노에서 시리얼통신으로 값 보내는 내역
202
203 SoilHumi_Per = (1023-SoilHumi)/1023 * 100;
204
205 Serial.print(Temp);
206 Serial.print(',');
207 Serial.print(Humi);
208 Serial.print(',');
209 Serial.print(SoilHumi_Per);
210 Serial.print(',');
211 Serial.print(WTLV);
212 Serial.print(',');
213 Serial.println(EventValue);
214 //EventValue 해석 : 온도변화 1 습도변화 2 물공급변화 4, 온도와 습도 = 3, 온도와 펌프제어 = 5,
215 // 습도와 물공급 6, 전부터 7 주야간 전환 9
216
217 pro_Time = millis();
218 }
219 }
```

12시간 주기로 주/야간 모드가 변경됨



```

182 //760이상으로 건조하다면
183 if(SoilHumi > 760){
184     digitalWrite(WTInRelayPin,HIGH);
185     EventVelue = EventValue +4; // 펌프이벤트 발생
186 }
187 else if(SoilHumi<480 && WTLV >100){ // 480 이하로 습하고, 수위가 100 이상일 때
188     digitalWrite(WTInRelayPin,LOW);
189     digitalWrite(WTOutRelayPin,HIGH);
190 }
191 else if(SoilHumi<480 && WTLV <100) // 480 이하로 습하고, 수위가 100 이하일 때
192     digitalWrite(WTOutRelayPin,LOW);
193 }
194
195 }
196 if(millis() >= (StartTime+RevertTime)){
197     DN_Mode = !DN_Mode;
198     StartTime = millis();
199     EventValue = 9;
200 }
201 //아두이노에서 시리얼통신으로 값 보내는 내역
202
203 SoilHumi_Per = (1023-SoilHumi)/1023 * 100;
204
205 Serial.print(Temp);
206 Serial.print(',');
207 Serial.print(Humi);
208 Serial.print(',');
209 Serial.print(SoilHumi_Per);
210 Serial.print(',');
211 Serial.print(WTLV);
212 Serial.print(',');
213 Serial.println(EventValue);
214 //EventValue 해석 : 온도변화 1 습도변화 2 물공급변화 4, 온도와 습도 = 3, 온도와 펌프제어 = 5,
215 // 습도와 물공급 6, 전부터 7 주야간 전환 9
216
217 pro_Time = millis();
218 }
219 }

```

아두이노 |

void loop()



Python에 Serial로 데이터를 보내주기 위한 코드

GUI

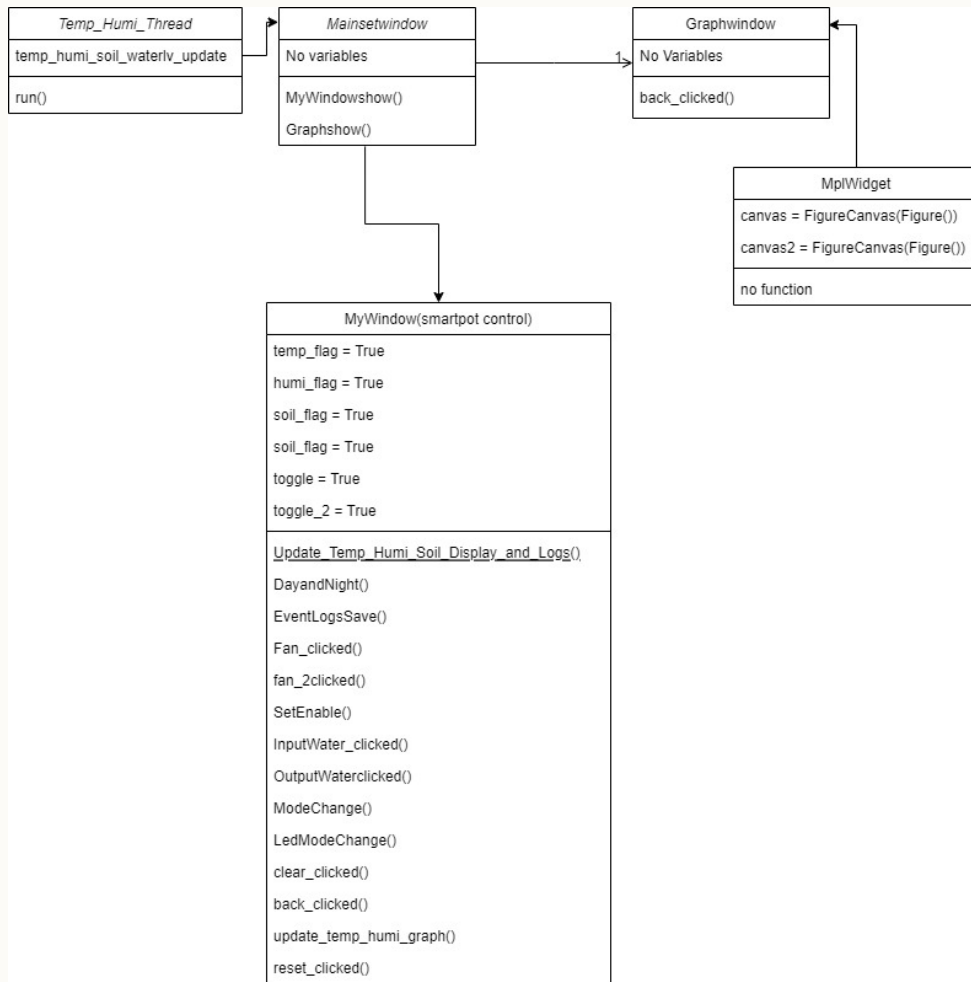


01 GUI Class 순서도

02 GUI 화면 구성

03 Logic

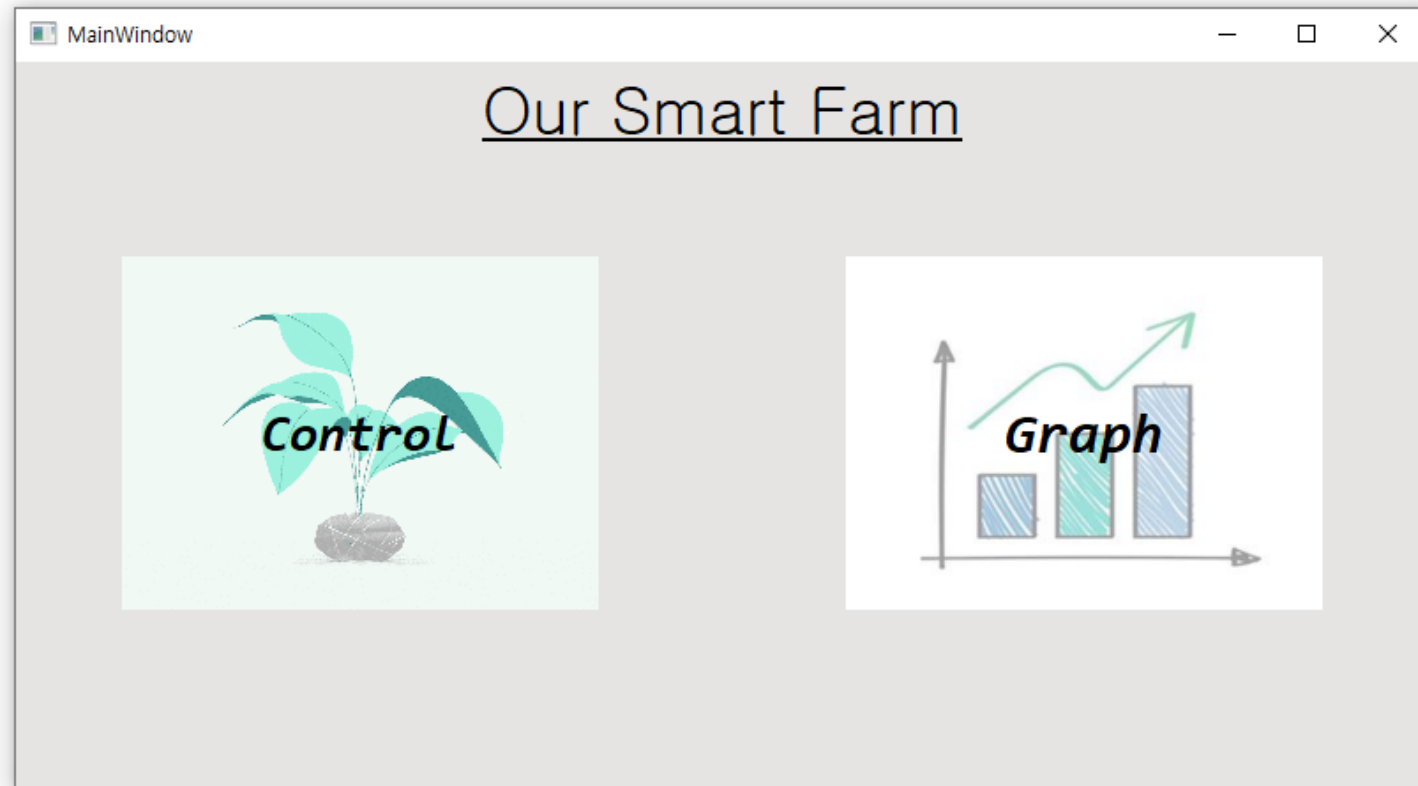




GUI

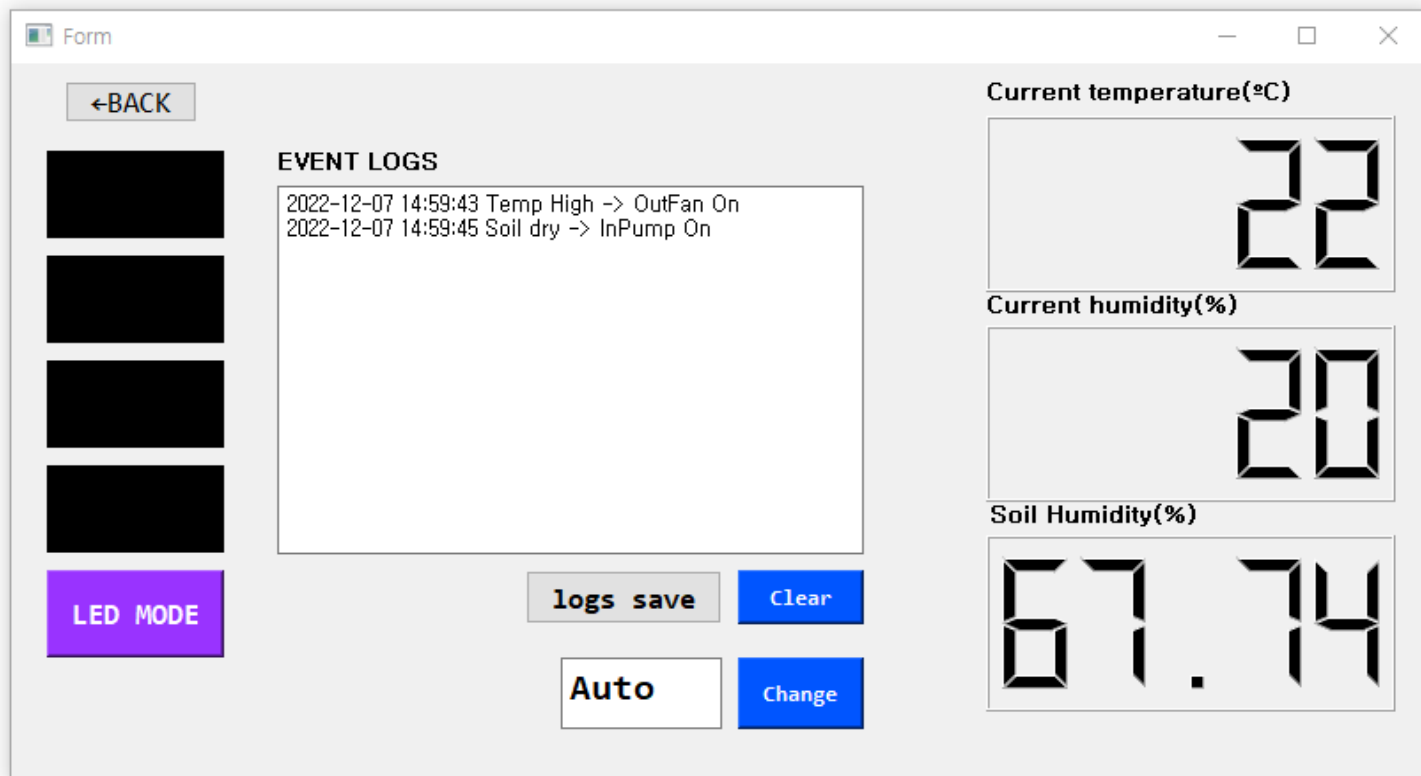
Class 순서도





GUI

Control UI (Auto)



The screenshot shows a GUI window titled "Form" with a standard Windows-style title bar (minimize, maximize, close buttons). The interface is divided into several sections:

- Left Sidebar:** Contains four black rectangular buttons stacked vertically. Below them is a purple button labeled "LED MODE".
- Top Left:** A grey button labeled "←BACK".
- Event Logs:** A section titled "EVENT LOGS" containing a text box with the following log entries:
 - 2022-12-07 14:59:43 Temp High -> OutFan On
 - 2022-12-07 14:59:45 Soil dry -> InPump On
- Bottom Left:** A white button labeled "Auto" and a blue button labeled "Change".
- Right Panel:** Displays three sensor readings in a digital font:
 - Current temperature(°C):** 22
 - Current humidity(%):** 20
 - Soil Humidity(%):** 67.74
- Bottom Center:** A grey button labeled "logs save" and a blue button labeled "Clear".



Form

←BACK

InFan

OutFan

Inpump

OutPump

LED MODE

logs save

Clear

Manual

Change

EVENT LOGS

2022-12-07 14:59:43 Temp High -> OutFan On
2022-12-07 14:59:45 Soil dry -> InPump On

Current temperature(°C)

22

Current humidity(%)

20

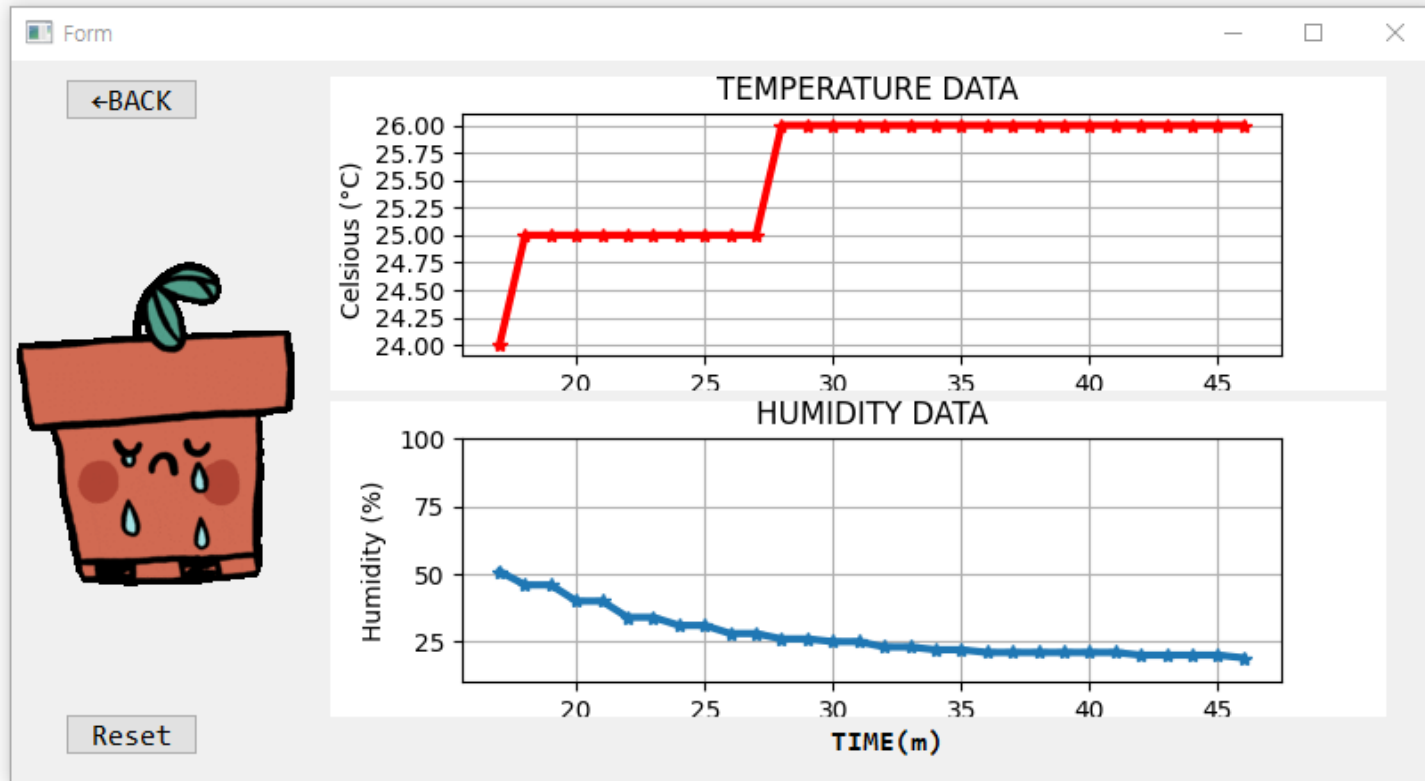
Soil Humidity(%)

68.23



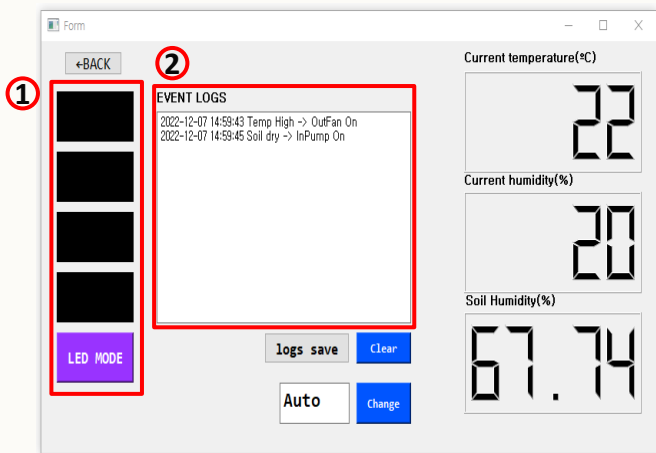
GUI

Graph UI



GUI |

Logic (Auto)



이벤트로그 기록 조건

1. 온도 10도 이하, 20도 이상
2. 습도 65%이상
3. 토양습도 25.7% 이하 53% 이상
4. 수위 100보다 낮을 때
5. 12시간 주기로 주/야간 모드 변경될 때

① 버튼 비활성화

```
self.Fan.setDisabled(True)
self.Fan_2.setDisabled(True)
self.LED_MODE.setDisabled(True)
self.InputWater.setDisabled(True)
self.OutputWater.setDisabled(True)
```

② 이벤트로그

```
currentdate = QDate.currentDate()
currenttime = QTime.currentTime()
if temp_float > 20 and self.temp_flag == True:
    self.textBrowser.append(currentdate.toString(Qt.ISODate) + " " + currenttime.toString() + " Temp High -> OutFan On")
    self.temp_flag = False
elif 10 < temp_float <= 20 and self.temp_flag == False :
    self.temp_flag = True
elif temp_float <= 10 and self.temp_flag == True:
    self.textBrowser.append(currentdate.toString(Qt.ISODate) + " " + currenttime.toString() + " Temp Low -> InFan On")
    self.temp_flag = False
```



GUI

Logic (Auto)

Form

+BACK

EVENT LOGS

2022-12-07 14:59:43 Temp High -> OutFan On
2022-12-07 14:59:45 Soil dry -> InPump On

LED MODE

logs save Clear

Auto Change

Current temperature(*C)
22

Current humidity(%)
20

Soil Humidity(%)
67.74

Qthread

```
class Temp_Humi_Thread(QThread): #온습도 쓰레드 클래스
    temp_humi_soil_waterlv_update = QtCore.pyqtSignal(float, float, float, int)
    DN_update = QtCore.pyqtSignal(int)
    def __init__(self):
        QThread.__init__(self)

    def run(self):
        while True:
            if py_serial.is_open == True:
                try:
                    serial_line=py_serial.readline()

                    serial_line_list = serial_line.split(b',')

                    temp_float =float(serial_line_list[0].decode())
                    humi_float = float(serial_line_list[1].decode())
                    soil_float = float(serial_line_list[2].decode())
                    waterlv_int = int(serial_line_list[3].decode())
                    DN_int = int(serial_line_list[4].decode())

                    self.temp_humi_soil_waterlv_update.emit(temp_float, humi_float, soil_float, waterlv_int)
                    self.DN_update.emit(DN_int)

                except:
```

시리얼모니터 Data

```
b'22.00,20.00,47.02,469,1\r\n'
[b'22.00', b'20.00', b'47.02', b'469', b'1\r\n']
22.00
```

GUI

Logic (Manual)

버튼 토글

Form

←BACK

InFan

OutFan

Inpump

OutPump

LED MODE

logs save Clear

Manual Change

EVENT LOGS

2022-12-07 14:58:43 Temp High -> OutFan On
2022-12-07 14:58:45 Soil dry -> InPump On

Current temperature(°C)

22

Current humidity(%)

20

Soil Humidity(%)

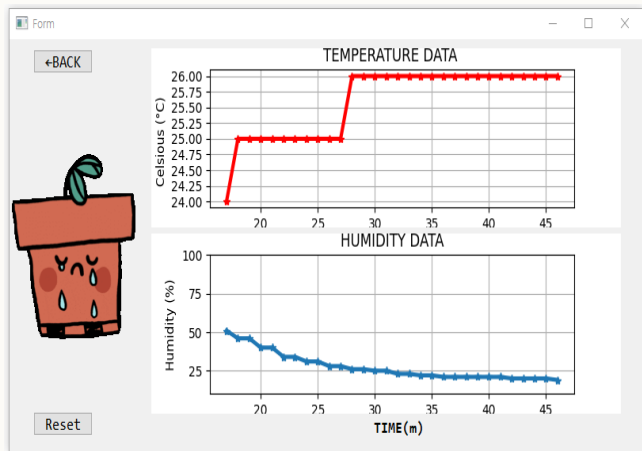
68.23

```
def fan_2clicked(self):  
  
    if self.Fan_2.isChecked():  
        py_serial.write(b'2')  
        self.MODE.setDisabled(True)  
    else :  
        py_serial.write(b'2')  
        self.SetEnable()
```



GUI

Logic (Graph)



MplWidget으로 승격

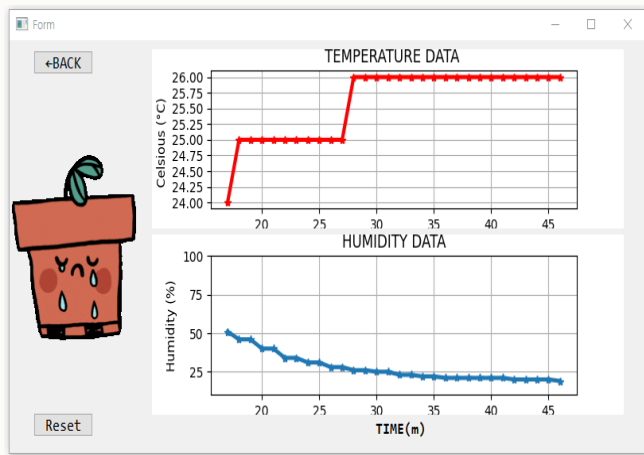
Object	Class
Form	QWidget
MplWidget	MplWidget

MplWidget 클래스

```
class MplWidget(QWidget):
    def __init__(self, parent = None):

        QWidget.__init__(self, parent)

        self.canvas = FigureCanvas(Figure())
        self.canvas2 = FigureCanvas(Figure())
        vertical_layout = QVBoxLayout()
        vertical_layout.addWidget(self.canvas)
        vertical_layout.addWidget(self.canvas2)
        self.canvas.axes = self.canvas.figure.plot()
        self.canvas2.axes = self.canvas2.figure.plot()
        self.setLayout(vertical_layout)
```

Graph plot

```
self.t = np.array([], dtype=float)
self.signal1 = np.array([], dtype=float)
self.signal2 = np.array([], dtype=float)
self.i = 0
self.interval = 60
```

```
def update_temp_humi_graph(self, temp_float, humi_float):
    if self.interval % 60 == 0:
        self.t = np.append(self.t, self.i)
        self.signal1 = np.append(self.signal1, temp_float)
        self.signal2 = np.append(self.signal2, humi_float)

        self.signal1 = self.signal1[-30:]
        self.signal2 = self.signal2[-30:]
        self.t = self.t[-30:]

        graph.MplWidget.canvas.axes.clear()
        graph.MplWidget.canvas.axes.plot(self.t, self.signal1, color='r', linewidth = '3', marker='+')
        graph.MplWidget.canvas.axes.grid(True, axis='y')
        graph.MplWidget.canvas.axes.grid(True, axis='x')
        graph.MplWidget.canvas.axes.yaxis.set_major_locator(MaxNLocator(10))
        graph.MplWidget.canvas.axes.yaxis.set_major_formatter(mtick.FormatStrFormatter('%1f'))
        graph.MplWidget.canvas.axes.set_ylabel('Celsious (°C) ')
        graph.MplWidget.canvas.axes.set_title('TEMPERATURE DATA ')
        graph.MplWidget.canvas.draw()

        graph.MplWidget.canvas2.axes.clear()
        graph.MplWidget.canvas2.axes.plot(self.t, self.signal2, linewidth = '3', marker='+')
        graph.MplWidget.canvas2.axes.grid(True, axis='y')
        graph.MplWidget.canvas2.axes.grid(True, axis='x')
        graph.MplWidget.canvas2.axes.set_ylabel('Humidity (%)')
        graph.MplWidget.canvas2.axes.set_title('HUMIDITY DATA')
        graph.MplWidget.canvas2.axes.set_ylim([10, 100])
        graph.MplWidget.canvas2.axes.set_xlabel('Time')
        graph.MplWidget.canvas2.draw()

        self.i += 1
        graph.btn_reset.clicked.connect(self.reset_clicked)
        self.interval += 1
```

역할 및 성과 |

※역할

아두이노에서 데이터와 시그널을 받아 GUI를 사용하여 수동제어를 담당하고 실시간으로 받은 데이터를 쓰레드를 사용하여 GUI에 출력하고 그래프로 표현

※성과

- 상추 씨앗이 잘 발아되어 좋은 성장 환경을 조성하도록 자동, 수동 제어하여 프로젝트 마지막 날까지 큰 문제 없이 잘 성장
- 아두이노에서부터 GUI로 실시간 데이터를 보내 출력할 때 Thread를 사용해야 한다는 것을 알았고 Thread 클래스를 만들 하나의 윈도우에만 가동하게 시켜야 한다는 것을 숙지
- 아두이노로 부터 받는 데이터를 가공하여 화면에 출력해주고 반대로 화면에서 시그널을 아두이노로 보내어 작동하도록 하는 시리얼 통신 숙지



A watercolor illustration of various green leaves and herbs, including parsley, basil, and other leafy greens, arranged around the central text. The leaves are painted in shades of green with visible veins and soft edges, creating a natural and fresh feel. The text "Thankyou" is centered in a simple, green, sans-serif font.

Thankyou