

# 2017년도 창의적 공학설계 최종보고서

2017. 12. 22

주관기관	숭실대학교 IT대학 소프트웨어학부
과 목	창의적 공학설계
담당교수	최 종 선
프로젝트	Feel Light

<div>Feel Light</div> <div>날씨 표현을 위한 아두이노 스마트 무드등</div>				
소 속	숭실대학교 IT 대학 소프트웨어학부			
분반 (00조)	나반 3조			
팀 명	Take A Look!			
팀 원 (총: 5명)	이름	학번	개인별 역할	팀내 기여도
	장원준	20170323	서버, 앱, 통신개발	20
	권혁진	20170262	외관, 앱디자인 내부 LED	20
	문태진	20170280	LED 패턴, 하드웨어, 아두이노	20
	손용락	20170296	설계/디자인, 외관 제작	20
	조세현	20170330	회로 구성, LCD	20
팀 구성	<div> <div> <div>팀장 : 장원준</div> <ul style="list-style-type: none"> <li>프로젝트 총괄</li> <li>nodeJS 서버 개발</li> <li>안드로이드 앱 개발</li> <li>아두이노 통신 개발</li> </ul> </div> <div> <div>팀원 : 권혁진</div> <ul style="list-style-type: none"> <li>외관 제작</li> <li>앱 디자인</li> <li>하드웨어 내부 LED 제작</li> <li>외관 제작</li> <li>LED 패턴 제작</li> </ul> </div> <div> <div>팀원 : 문태진</div> <ul style="list-style-type: none"> <li>LED 패턴 제작 총괄</li> <li>아두이노</li> <li>하드웨어 내부 LED 제작</li> <li>LED 배선 관리</li> </ul> </div> <div> <div>팀원 : 손용락</div> <ul style="list-style-type: none"> <li>하드웨어 설계/디자인</li> <li>앱 디자인</li> <li>외관 제작</li> <li>LED 패턴 제작</li> </ul> </div> <div> <div>팀원 : 조세현</div> <ul style="list-style-type: none"> <li>회로 구성</li> <li>LCD 컨트롤</li> <li>LED 패턴 제작</li> <li>외관 제작</li> </ul> </div> </div>			

## 〈 한 글 요약 문 〉

팀 이름		3조 - Take A Look		
제목		Feel Light - 날씨 표현을 위한 아두이노 스마트 무드등		
배경 및 당위성		기존의 분위기 연출이 한계인 무드등이 날씨 확인이 한계를 극복하기 위해서, 사용자에게 날씨에 대한 정보를 시각적으로 주어 오늘의 날씨에 따라 우산 등의 용품을 준비할 수 있도록 하기 위하여 설계되었다. 침실 등의 일상적인 생활공간에서 사용되며 날씨를 실시간으로 확인해 오늘의 날씨에 따른 분위기 연출을 가능케 하는 제품이다.		
제안 내용	최종 목표	이 프로젝트를 통해 사용자는 스마트폰을 사용하지 않아도 실시간으로 날씨 확인이 가능하며 비나 눈, 폭염과 같은 날씨에 미리 준비할 수 있게 된다. 또한 집 안에서도 바깥의 날씨와 동일한 분위기를 연출하는 것도 가능하다.		
	주요 내용	본 프로젝트는 날씨 확인이 가능한 스마트 무드등으로 기능은 4가지가 있다. 무드등 모드 - 기본적인 분위기 연출 날씨 모드 - 날씨에 따른 패턴 출력 온도 모드 - 온도에 따른 패턴 출력 미세먼지 모드 - 미세먼지 지수에 따른 패턴 출력		
	개발 방법	하드웨어	기동을 제작하여 LCD와 LED strip 회로를 내부에 연결한 후 벽면에 고정 후 아크릴로 외부를 덮어씌움	
소프트웨어		Android Studio를 통한 어플리케이션 개발 SK weather planet으로부터 받은 API데이터를 NodeJS 서버와 HTTP통신으로 아두이노와 LED, LCD에 전달		
기대 효과		스마트폰을 사용하지 않아도 실시간으로 날씨 확인이 가능해지며 비나 눈, 폭염과 같은 날씨에 사용자가 미리 준비할 수 있게 된다. 또한 집 안에서도 바깥의 날씨와 동일한 분위기를 연출할 수 있다.		
중심어	날씨데이터		모드설정	통신
	API		안드로이드	NodeJS

## < 목 차 >

1. 프로젝트 개요	-----	00
1-1. 개발 기술(또는 결과물)의 개요	-----	01
1-2. 개발 기술(또는 결과물)의 필요성	-----	02
1-3. 개발 기술의 예상효과 및 활용방안	-----	04
2. 관련 기술 현황	-----	01
2-1. 사례조사	-----	01
3. 기술개발의 수행내용 및 결과	-----	05
3-1. 개발목표	-----	05
3-2 개발내용	-----	05
3-2-1 개발 내용 및 범위	-----	07
3-2-1 구현내용	-----	10
3-2-2 제작과정	-----	13
3-2-3 실험결과	-----	16
4. 기술개발의 활용계획	-----	17
5. 사용자 매뉴얼	-----	18
6. 개발 후기	-----	24

## < 그림 목차 >

[그림 1] 만화 ‘뽀식’ 그림1	-----	01
[그림 2] 만화 ‘뽀식’ 그림2	-----	02
[그림 3] 만화 ‘뽀식’ 그림3	-----	02
[그림 4] 만화 ‘뽀식’ 그림4	-----	03
[그림 5] Feel Light 완성품 예시, 로고	-----	03
[그림 6] 웨더큐브 제품 사진	-----	01
[그림 7] Tempescope 제품 사진	-----	02
[그림 8] Smart Light 제품 사진	-----	03
[그림 9] Luminich Color Lamp 제품 사진	-----	04
[그림 10] 소프트웨어 구성도	-----	08
[그림 11] 하드웨어 구성도	-----	09
[그림 12] LED strip 연결	-----	13
[그림 13] 하드웨어 기본회로 연결 완성	-----	14
[그림 14] 하드웨어 완성 후	-----	14

## < 표 목차 >

[표 1] 시간대에 따라 다르게 나타나는 실행화면과 날씨표현	- 18
[표 2] 모드 설정화면과 컬러테이블을 통한 사용자 정의 색 설정화면	- 19
[표 3] 날씨모드 실행 시 표현되는 날씨	----- 20
[표 4] 날씨모드 실행 시 표현되는 날씨	----- 20
[표 5] 날씨모드 실행 시 표현되는 날씨	----- 21
[표 6] 온도 변화에 따른 무드등 LED의 변화	----- 22
[표 7] 미세먼지 지수에 따른 무드등 LED의 변화	----- 23

## 1. 프로젝트의 개요

### 1-1 개발 기술의 개요

본 프로젝트는 날씨 표현을 위한 아두이노 스마트 무드등이다. 기존에 존재하던 분위기 연출의 역할만을 하던 무드등의 한계를 극복하고 사용자에게 날씨에 대한 정보를 시각적으로 주어 오늘의 날씨에 따라 우산 등의 용품을 준비할 수 있도록 하기 위하여 설계되었다. 총 200개의 LED를 컨트롤하여 맑음, 흐림, 비, 눈, 천둥, 비와 천둥, 눈과 천둥, 폭염, 한파 총 9가지의 날씨를 표현하고 오늘의 온도, 미세먼지 등급 등을 추가로 표현한다. 또한 무드등답게 사용자가 원하는 색상으로 무드등의 색상을 바꾸는 모드가 존재한다. Feel Light는 침실 등의 일상적인 생활공간에서 사용되며 오늘의 날씨에 따른 분위기 연출이 주 목적이다. 그렇기 때문에 날씨를 실시간으로 확인하고, 날씨에 따른 특별한 연출로 집 안에서도 바깥의 날씨와 동일한 분위기를 느낄 수 있도록 하는 무드등을 개발하고자 한다. 이 프로젝트를 통해 사용자는 스마트폰을 사용하지 않아도 실시간으로 날씨 확인이 가능하며 비나 눈, 폭염과 같은 날씨에 미리 준비할 수 있게 된다. 또한 집 안에서도 바깥의 날씨와 동일한 분위기를 연출하는 것도 가능하다.

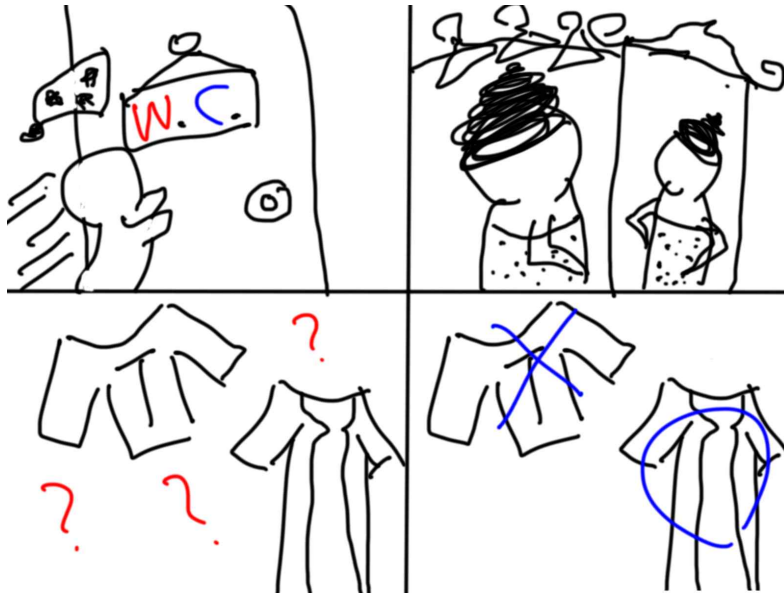
### 1-2 개발 기술의 필요성

본 프로젝트의 개발 기술의 필요성은 주인공이 ‘뽀식’ 인 다음 만화를 보며 알 수 있다.



<그림 1> 만화 ‘뽀식’ 그림 1

여느 때와 같이 아침에 눈을 뜬 뽀식은 평소보다 늦게 깨어났음을 깨닫게 된다.



<그림 2> 만화 ‘뽀식’ 그림 2

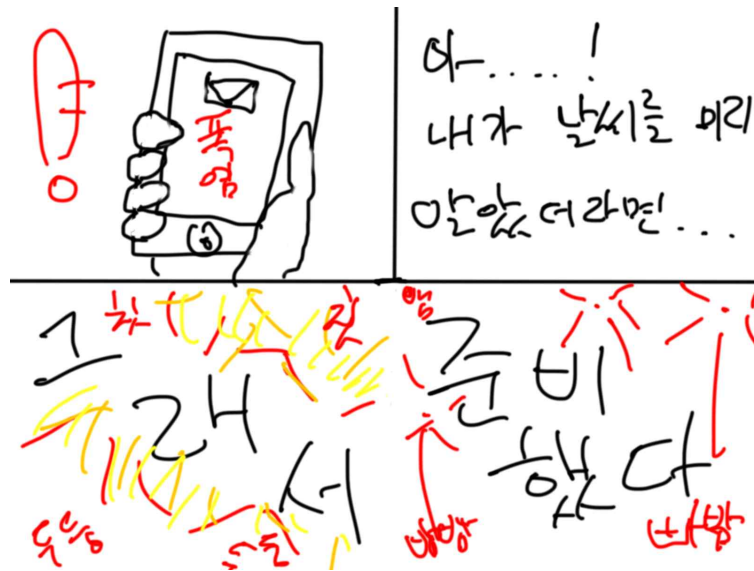
서둘러 나갈 준비를 한다. 밖의 상황은 고려하지 않은 채, 시원하게 입고 나갈지 약간 따뜻하게 입고 나갈지를 고민하게 된다. 결국 코트를 입고 밖을 나서게 되는데..



<그림 3> 만화 ‘뽀식’ 그림 3

문을 열고 발걸음을 내딛은 순간 날씨는 폭염에 가까움을 깨닫게 된다.





<그림 4> 만화 ‘뽕식’ 그림 4

때마침 폭염경보 문자가 울리게 되고, 그는 생각하게 된다.

‘만약 내가 날씨를 미리 알았더라면..?’



<그림 5> Feel Light 완성품 예시, 로고

다음과 같은 상황을 피하기 위해 날씨 확인이 가능한 스마트 무드등 Feel Light의 필요성이 대두된다고 할 수 있다.

### 1-3 개발 기술의 예상효과 및 활용방안

본 프로젝트의 예상효과는 다음과 같다. 스마트폰을 사용하지 않아도 실시간으로 날씨 확인이 가능해지며 비나 눈, 폭염과 같은 날씨에 사용자가 미리 준비할 수 있게 된다. 또한 집 안에서도 바깥의 날씨와 동일한 분위기를 연출할 수 있다.

본 프로젝트의 활용방안은 두 가지로 구분할 수 있다. 이는 외출 여부에 따른 분류로 외출이 필요한 경우에는 미리 밖의 날씨, 미세먼지, 온도를 알아내어 준비를 할 수 있다. 반대로 집에 있는 경우에는 당일 날씨에 따라 무드등이 변하기 때문에 사용자는 집 안에서도 바깥 날씨의 분위기를 느낄 수 있다.

## 2. 관련 기술 현황

### 2-1 관련 사례 및 분석

#### 2-1-1 Weather Cube

##### [개요]

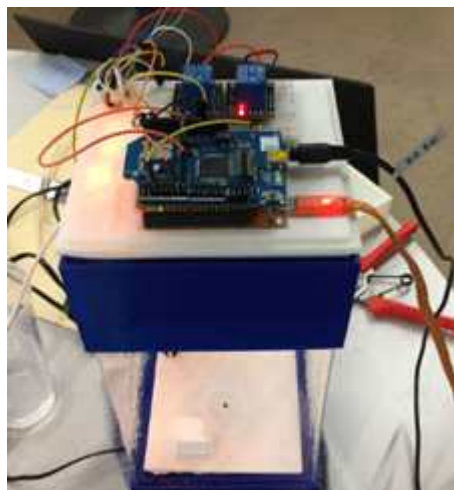
날씨를 시각적으로 표현하는 장치로, 가습기 모듈과 LED, 워터펌프를 사용하여 천둥, 비, 흐림, 맑음을 표현할 수 있다. 플랫폼으로는 Orange Board를 사용하였다. 날씨 API를 Wifi Shield를 통해 읽어온 후, 데이터를 분석하여 당일의 날씨를 나타내는 패턴을 작동시킨다. 통신 방식으로는 HTTP를 사용한다. 맑음과 천둥의 경우는 상단의 LED를 사용하여 표현하고, 흐린 날씨는 가습기 모듈을 통해 큐브 내부를 뿌영게 만든다. 읽어온 데이터가 비일 때는 상단의 워터펌프를 가동시켜 물이 떨어지는 효과를 연출한다.

##### [장점]

- 적은 양의 LED를 통해서 날씨를 표현가능
- 배수처리를 통한 회로의 불안요소 해결
- 가습기 모듈을 통한 흐린 날씨의 시각적 표현

##### [개선점]

- 바깥으로 돌출된 회로
- 아쉬운 배선



[그림 6] 웨더큐브 제품 사진

## 2-1-2 Tempescope

### [개요]

일본인 켄 카와모토가 기획한 제품으로, 앞서 서술한 2-1-1의 웨더큐브와 유사하다. 차이점은 Bluetooth 모듈을 사용한 하드웨어-어플리케이션 간 통신이 가능하고, 어플리케이션을 사용하여 원하는 지역의 날씨를 확인할 수 있다. 날씨표현은 2-1-1의 웨더큐브와 동일하다. 맑은 날씨와 천둥은 LED를 사용하고, 흐린 날씨는 가습기 모듈을 사용하며 비는 워터펌프를 가동시킨다. 플랫폼으로는 EAGLE를 사용한다.

### [장점]

- 배수처리를 통한 회로의 불안요소 해결
- 가습기 모듈을 통한 흐린 날씨의 시각적 표현
- 사용자의 위치 설정 가능

### [개선점]

- IOS한정 지원, Android 어플리케이션은 지원하지 않음
- 현재 개발 진행중인 상품이므로 결함 존재
- 날씨 표현 종류의 부족



[그림 7] Tempescope 제품 사진

### 2-1-3 Smart Light

#### [개요]

Arduino Uno R3 플랫폼을 사용한 기본적인 무드등 연출 방법을 소개하는 예시 작품이다. 아두이노 플랫폼과, LED, 외부를 둘러싸는 A4용지로 구성되어있다. 어플리케이션을 통해 색을 변경할 수 있도록 하기 위해 블루투스 모듈도 장착하였다.

#### [장점]

- 아주 간단한 제작
- 어플리케이션을 통한 색 변경 가능

#### [개선점]

- 회로를 A4용지 한 장으로 둘러싸 불안정한 구조
- 실외에서 사용 불가능



[그림 8] Smart Light 사진

[개요]

Arduino Uno R3 플랫폼을 사용한 무드등이다. 은은한 불빛을 내기 때문에 밤에 취침등 또는 보조등으로 사용한다. 적외선 센서를 이용하여 스위프 제스처를 인식할 수 있고, 이를 이용하여 빛의 색상, 밝기, 변화, 전원을 변경한다. 날씨 구현 여부와 통신 기능은 존재하지 않는다.

[장점]

- 스위프 제스처를 사용한 간편한 조작

[개선점]

- 구현 가능한 색상이 적음



[그림 9] Luminch Color LED Lamp 사진

### 3. 기술 개발의 수행내용 및 결과

#### 3-1 개발 목표

본 프로젝트에서는 아두이노를 활용하여 날씨 정보를 사용자에게 제공하기 위한 스마트 무드등을 개발한다. 날씨 기능 뿐만 아니라 미세먼지, 온도, 사용자 지정 모드 등 여러 모드를 제공한다. 그리고 이를 컨트롤하는 앱을 개발한다.

#### 3-2 개발내용

개발목적	연구개발 내용	성과내용
무드등 모드	<ul style="list-style-type: none"> <li>• 사용자가 원하는 색을 Feel Light에 출력하는 기능 개발</li> <li>• 컬러테이블을 사용하여 사용자 편의성 제공</li> </ul>	<ul style="list-style-type: none"> <li>• 사용자가 원하는 색상을 쉽게 출력할 수 있음</li> <li>• 간편한 UI를 제공함</li> </ul>
날씨 모드	<ul style="list-style-type: none"> <li>• 날씨에 우선순위를 두어 사용자에게 중요하다고 판단하는 날씨를 우선적으로 제공하는 기능을 개발</li> <li>• 총 9개의 날씨 맵을 구현</li> </ul>	<ul style="list-style-type: none"> <li>• 비, 눈, 맑음, 흐림, 천둥, 비와 천둥, 눈과 천둥, 폭염, 한파 맵 구현</li> <li>• 각각의 맵은 날씨 정보를 성공적으로 제공할 뿐만 아니라 그 날씨의 분위기를 사용자가 느낄 수 있도록 함</li> <li>• 날씨를 실시간으로 API를 통해 데이터를 제공받음. 사용자는 실시간으로 중요한 날씨 정보를 확인할 수 있음</li> </ul>

미세 먼지 모드	<ul style="list-style-type: none"> <li>• LED로 미세먼지가 날리는 것을 표현</li> <li>• 미세먼지 등급을 참고하여 등급에 따라 LED 패턴 변화</li> </ul>	<ul style="list-style-type: none"> <li>• 파랑, 초록, 주황, 빨강의 색상을 이용하여 효율적으로 등급 표현</li> <li>-각각 좋음, 보통, 나쁨, 매우 나쁨을 의미함</li> <li>• 미세먼지 분위기를 연출하는데 성공</li> </ul>
온도 모드	<ul style="list-style-type: none"> <li>• 10도를 기준으로 온도가 올라갈수록 붉게 출력, 온도가 내려갈수록 파랗게 출력하는 기능 개발</li> <li>• 사용자가 현재 온도를 쉽게 짐작할 수 있게 함</li> </ul>	<ul style="list-style-type: none"> <li>• rgb를 1도가 올라가거나 내려갈 때마다 7씩 바꾸어 이를 구현</li> </ul>
앱	<ul style="list-style-type: none"> <li>• 현재의 종합 날씨 정보를 제공하는 기능 개발</li> <li>• Feel Light의 모드 컨트롤러</li> </ul>	<ul style="list-style-type: none"> <li>• 사용자에게 종합 날씨 정보를 제공하는 앱 개발</li> <li>• 4가지 모드를 컨트롤 할 수 있음</li> </ul>



### 3-2-1 개발 내용 및 범위

Feel Light는 아두이노 메가, LED strip, WI-FI 모듈을 주 부품으로 사용하여 사용자에게 날씨 정보를 LED 맵으로 제공하는 아두이노 스마트 무드등이다. 주요기능으로는 무드등 모드, 날씨 모드, 미세먼지 모드, 온도 모드가 있다. 모든 모드는 기초적으로 앱에서 통신을 받아 제어된다.

첫 번째 기능은 무드등 모드이다. 앱에서 무드등 모드를 누르면, 컬러 테이블이 나오고, 사용자는 여기서 자신이 원하는 색을 골라 무드등으로 출력할 수 있다. 검은 빛은 없기 때문에 어두운 계열의 색상은 구현이 힘들다. 하지만 이를 제외한 거의 모든 색상은 구현이 가능하다. 사용자는 이 기능을 통하여 자신의 기분, 취향에 따라 무드등을 조절할 수 있다.

두 번째 기능은 날씨 모드이다. 앱에서 날씨 모드를 누르면, 미리 정해둔 날씨 우선순위를 기반으로 사용자에게 실시간 날씨를 LED 맵으로 보여준다. 날씨는 맑음, 비, 눈, 흐림, 천둥, 비와 천둥, 눈과 천둥, 폭염, 한파가 존재하며 각각의 우선순위가 존재한다. 그 우선 순위는 비와 번개 > 비 > 눈과 번개 > 눈 > 폭염 또는 한파 > 흐림 > 맑음 순이다. 현재로부터 24시간의 날씨 정보를 SK weather planet에서 제공하는 API를 기반으로 사용자에게 제공한다. 그리고 날씨 모드에서 출력하는 날씨는 24시간의 날씨에서 가장 우선순위가 높은 날씨이다. 사용자는 이러한 데이터를 보고 갑작스러운 비, 폭염 과 같은 사용자에게 영향을 줄 수 있는 날씨를 일찍 파악하고 대비할 수 있다. 뿐만 아니라 Feel Light의 날씨 모드는 사용자가 그날의 날씨 분위기를 연출할 수 있게 한다.

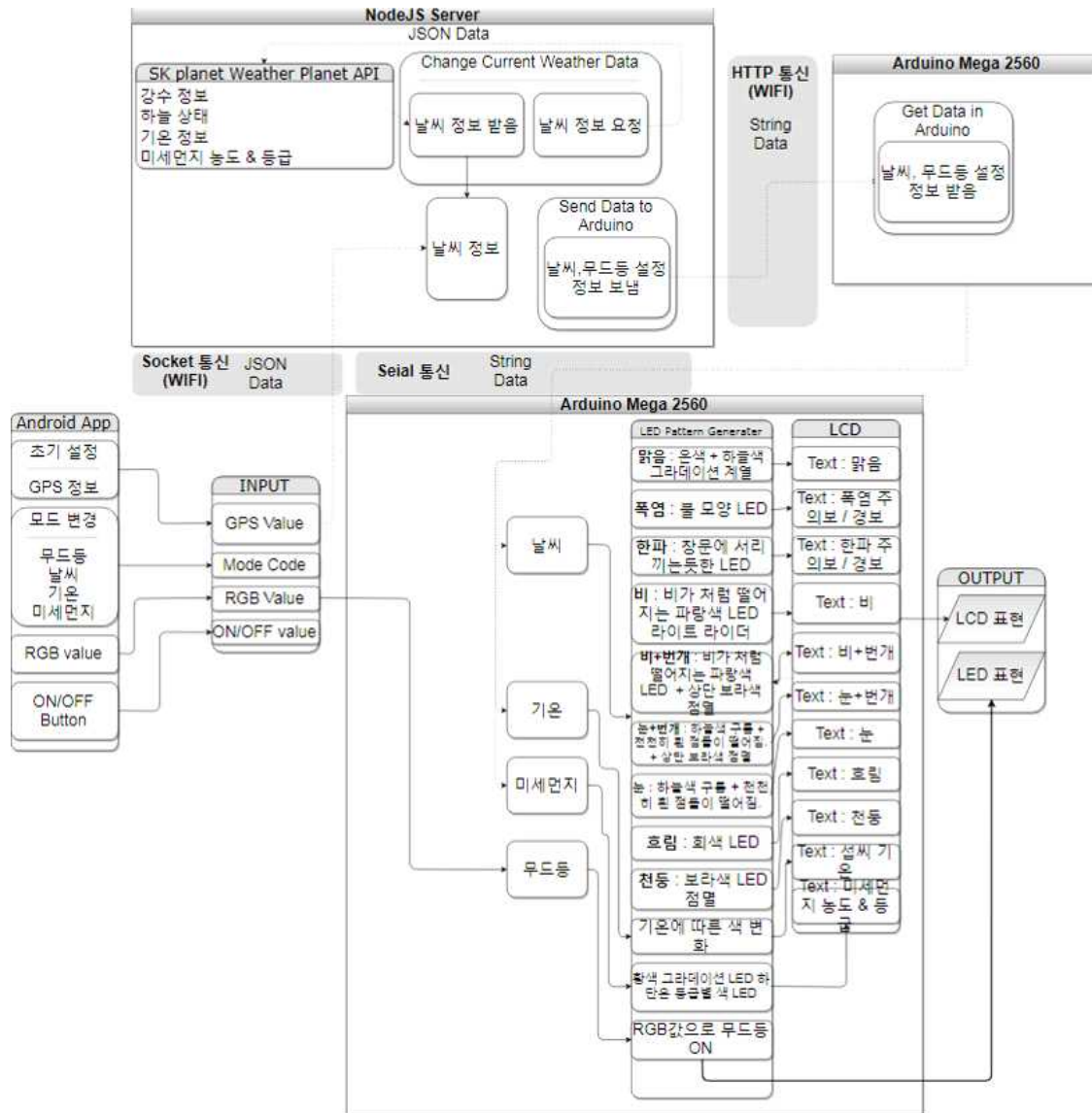
세 번째 기능은 온도 모드이다. 동일한 API에서 제공받은 데이터를 토대로 사용자에게 온도 정보를 제공하는 모드이다. 온도는 10도가 베이스 값이며 하얀색으로 표현된다. 여기서 온도가 상승하면 Feel Light는 점점 붉게 변하고, 온도가 하강하면 점점 푸르게 변한다. 1도가 변할 때 마다 rgb값이 7만큼 변화하도록 하여 이러한 기능을 구현했다. 최대 영상 40도, 최소 영하 20도까지의 온도 정보를 제공한다.

네 번째 기능은 미세먼지 모드이다. 동일한 API에서 제공받은 데이터를 토대로 사용자에게 실시간 미세먼지 정보를 제공하는 모드이다. 하단 부분이 그 날의 미세먼지 등급에 따라 색이 변한다. 파랑색은 좋음, 초록색은 보통, 주황색은 나쁨, 빨간색은 매우 나쁨을 의미한다. 미세먼지 등급의 기준은 기상청의 표준을 따랐으며, 색의 상징은 네이버 표준을 따랐다. 상단 부분에는 주황빛과 노란빛이 흘러가는 듯한 연출을 하여 모래바람을 연상케 하였다.

지금까지 제공된 모든 모드는 자체 제작한 Feel Light 앱에서 컨트롤 할 수 있다. Feel Light 앱은 모드를 컨트롤 할 뿐만 아니라 우리가 제공하는 모든 날씨 정보를 한 화면에 출력한다.

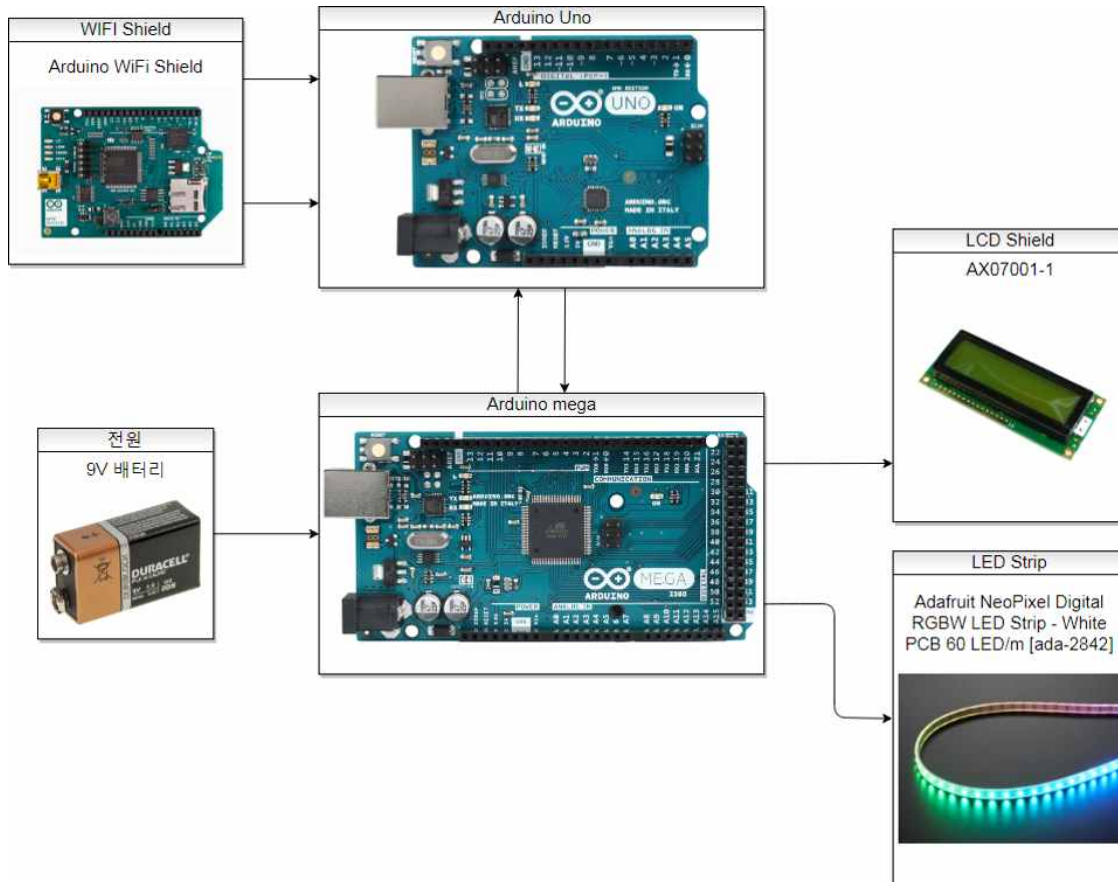
여기까지가 Feel Light의 핵심 기능에 대한 설명이었다. 다음 그림은 Feel Light의 작동 방식을 그림으로 구현화 한 것이다.

## - 소프트웨어 구성도



<그림 10> 소프트웨어 구성도

-하드웨어 구성도



<그림 11> 하드웨어 구성도

### 3-2-1 구현 내용

#### ○주요기능 1

통신 - 날씨 api 받아와서 데이터 처리하기.

Sk weather api 에서 날씨 정보를 받아왔다. 3시간별 날씨, 매 분 업데이트되는 현재 날씨, 한파정보, 폭염 정보, 미세먼지 정보 를 불러왔다. 모두 nodeJS 서버에서 받아왔으며 모든 데이터는 JSON 형태로 받아와 파싱하였다. 날씨와 한파, 폭염 정보의 우선순위를 정하는 것이 어려웠다. 사람들이 어떻게 날씨를 받아들이고 우리 앱이 어떤 상황에서 쓰일지 고민하였다. ‘비’ 가 올때 사람들은 우산을 챙겨야 하므로 상위 우선순위에 두었고, 맑음일수록 우선순위를 낮추었다. 폭염과 한파의 우선순위는 그 사이에 배치하였다. 또한 api 를 사용하다가 업데이트 주기를 어떻게 정해야 자연스러울까, 외부 날씨와 싱크를 맞추까 고민을 많이 하였다. 처음에는 1시간단위로 하였다가 눈이오는데 앱에서는 눈이 안오는 현상을 발견하여 모든 데이터를 20분 단위로 받아 자연스러움을 더했다.

#### ○주요기능 2

통신 - 아두이노에서 서버와 통신

아두이노와 서버와 통신을 위해서 처음에는 socket 방식을 사용하려 했다. 그러나 아두이노가 싱글 스래드인점과 아두이노에 SocketIO 라이브러리가 모두 신 버전에서 작동하지 않아서 사용하지 못하였다. 이것을 해결하기위해 서버에서 http - get 방식으로 라우터를 하나 파고 여기로 http - get 방식으로 아두이노가 접근하여 받은 데이터를 주어진 토큰을 기준으로 자르는 작업을 하였다. 이를통해 아두이노에 성공적으로 데이터가 들어갈 수 있었지만 http 통신은 서버와 클라이언트가 계속 연결되어있어야해서 싱글 스래드인 아두이노에서 문제가 되었다. 이를 해결하기위해 시분할 라이브러리를 사용하여 가상으로 멀티 스래드를 돌려볼려 했지만 부피가 작은 2가지 작업이라면 되었겠지만 부피가 큰 LED 컨트롤과 http 통신 두개가 동시에 작동되어야 했기 때문에 이 방식을 사용하지 못하였다. 따라서 물리적으로 하나의 스래드를 추가하는 방식을 사용하여 통신용 아두이노 하나를 더 넣었다. 통신용 아두이노에서 받은 데이터를 LED 컨트롤용 메가에 옮기기 위해 시리얼 통신을 사용하여 데이터를 전송하였다.

### ○주요기능 3

#### LED 컨트롤

LED컨트롤은 기본적으로 아두이노에 미리 존재하던<Adafruit\_NeoPixel.h> 라이브러리를 사용하여 구현하였다. 위의 라이브러리를 이용할 시 LED스트라이프 한 줄을 1차원 배열처럼 사용이 가능했기 때문에 간단하게 LED패턴을 만들 수 있었다. <Adafruit\_NeoPixel.h> 라이브러리를 사용할 때 주의사항은 사용할 LED의 종류를 정확히 알아두어야 한다는 것이다. 사용할 LED가 RGB 형일 경우 기본 설정에서Adafruit\_NeoPixel(NUMPIXELS, PIN\_NUM, NEO\_GRB + NEO\_KHZ800)를 사용해야 하지만 RGBW일 경우에는Adafruit\_NeoPixel(NUMPIXELS, PIN\_NUM, NEO\_RGBW + NEO\_KHZ800)을 사용해야 한다. 이를 제대로 하지 못하고 컴파일 할 시 오류가 나오지는 않지만 LED의 색 조절이 제대로 이루어 지지 않는다.

또한 LED패턴을 만들때 코드를 반복문을 통하여 최대한 간결하게 나타내야 한다. 패턴이 적을 경우에는 전체 코드의 길이가 짧아 상관이 없지만 패턴이 많아질 경우 코드의 양이 많아지면 아두이노 자체에서 심하게 버퍼링이 생긴다. 이를 해결하기 위해서는 위에서 말한 것 처럼 반복문과 조건문을 통하여 코드를 최대한 간결하게 만들어 주면 해결된다.

LED를 납땀할 때에도 주의할 점이 있었다. LED의 납땀 부분을 보면 위쪽에는 Din과 아래쪽에는Dout으로 써있는 부분이 있는데, Din부분에 납땀을 해야한다. Dout부분에 점퍼선을 납땀할 경우 LED의 전원이 들어오지 않는 문제가 발생하였다.

이후 패턴을 만들 때는 각각의 날씨와 모드에 맞는 패턴을 반복문과 난수를 이용하여 구현하였다.

### ○주요기능 4

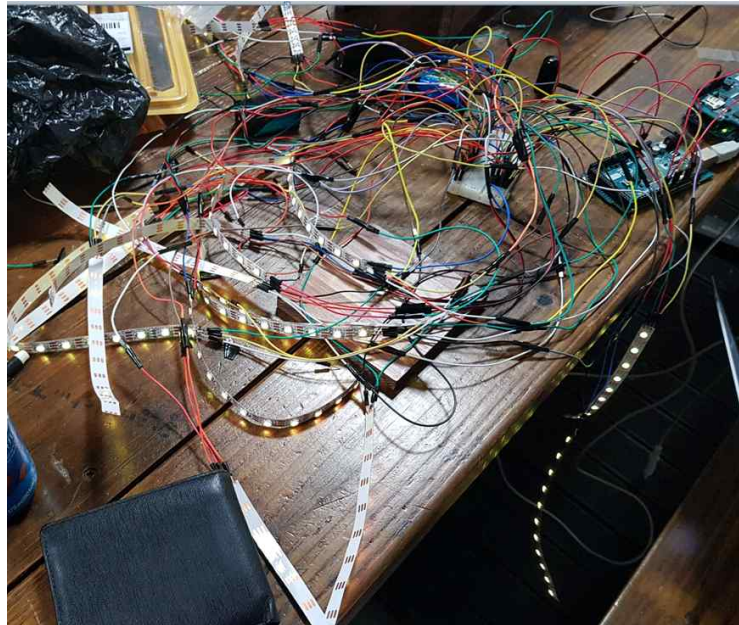
#### 안드로이드 앱

안드로이드 앱을 만들면서 Tabbed layout 을 구현하면서 많은 문제가 발생했었다. Main Activity 위에 2개의 Fragment 를 올려야 하는 상황이었다. 한 Fragment 에서 Socket 통신을 통해 받은 데이터를 다른 Fragment와 Main Activity 에 옮겨야 하는상황에서 문제가 발생하였다. Fragment 의 생명 주기 때문에 Fragment 에서 Activity 를 불러오거나 다른 Fragment 를 불러올때 Attached 상태가 아닌 Fragment 를 불러오면 바로 getActivity 나 Fragment 에서 오류가 나와 데이터를 옮기지 못하기도 하였다. 이 문제는 isAdded() 함수와 getActivity 의 null 체크를 통해 해결하였다. 두 번째 문제는 Fragment 에서 UI 수정 문제였다. Main Thread 가 아닌 다른곳에서 UI를 수정하기 위해서는 많은 작업을 거쳐야 했다. Feel Light 의 안드로이드 앱에서는 runOnUiThread 메소드를 사용하여 외부 Thread 에서 UI 업데이트를 구현하여 이 문제를 해결하였다. 세 번째 문제는 SocketIO 라이브러리 에 있었다. SocketIO Server NodeJS 는 2.0 이상 버전으로 업데이트가 되어 1.0 버전과 호

환성이 없었다. 이 호환성 문제는 github 에서 1.0 버전과 호환되는 오픈소스를 구해서 해결할 수 있었다.

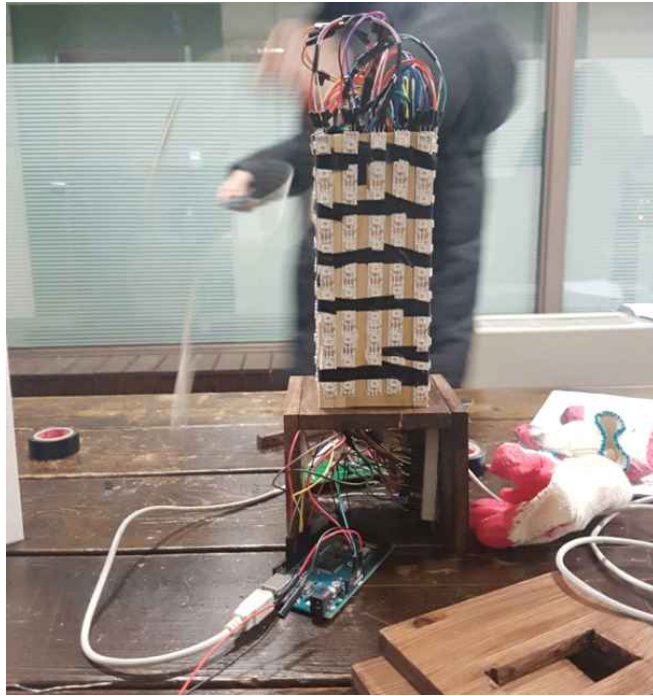
### 3-2-2 제작과정

#### - 하드웨어 제작과정



[그림 12]LED strip 연결

하드웨어 내부에 들어갈 LED strip를 브레드보드에 연결한 후 이를 Arduino Mega에 연결하였다. 또한 통신 또한 진행하기 위해 Arduino Uno R3을 Mega와 연결하였다. Uno와 Mega 사이에서 시리얼 통신을 통해 받아온 데이터를 Strip에 전송하여 적절한 패턴을 이끌어내도록 한다.



[그림 13] 하드웨어 기본 회로 연결 완성

브레드보드를 바닥기둥 벽면에 고정시킨 후, 내부기둥을 통해 LED strip를 바깥으로 빼낸다. 이후 절연테이프를 이용하여 Strip을 내부기둥 벽면에 고정시키도록 한다.

- 하드웨어 완성



[그림 14] 하드웨어 완성 후

그림 13에서 회로 연결을 끝낸 하드웨어 외부에 나무판자와 아크릴 등을 부착하여 내부 회로가 드러나지 않도록 하고, 기둥 등을 부착하여 세부 디자인을 완성시킨다



## - 소프트웨어 제작 과정

### 1. API 데이터

SK 웨더플래닛 API를 신청하여 그날의 날씨 데이터를 받아온 후, 맑음, 비, 흐림 등 여러 날씨가 존재할 경우 우선순위를 정하여 어떤 LED 패턴을 출력할지를 정하였다.

### 2. 아두이노 서버 간 통신

서버에서 API 데이터를 받아온 후 LED에 데이터를 전달하는 두 가지 작업을 동시에 수행해야 하므로 아두이노를 2개 사용하였다. 와이파이 실드를 부착하여 Node JS 서버에서 날씨 데이터를 받아온 후 이를 Uno에서 가공한다. 이후 시리얼 통신으로 Mega에 전송하고, 이후 LED와 LCD에 데이터를 전송하도록 하였다.

### 3. LED 컨트롤

Uno로부터 받은 데이터에 따라 각각의 패턴을 출력하도록 하였고, 알고리즘을 이용하여 한 날씨 내에서 일정한 패턴이 출력되는 것이 아닌 무작위 패턴이 출력되도록 하였다.

### 4. 어플리케이션

Android Studio를 이용하여 컨트롤 어플리케이션 Feel Light를 제작하였다.

### 5. LCD

라이브러리를 활용하여 Uno로부터 받은 데이터에 따라 적절한 문구를 출력하도록 하였다.

### 3-2-3 실험결과

구 분		개발방법	실험 결과
무드등	무드등 모드	4 면에 11칸 짜리 LED 스트립 5개를 부착하여 무드등을 제작. 사용자가 앱에서 칼라 테이블을 조작하면 서버를 통하여 그 데이터를 메가로 전송하고 무드등의 색을 바꿈	사용자가 원하는 대로 무드등의 색이 바뀜
	날씨 모드	API를 이용하여 날씨 정보를 제공받고 이에 따라 날씨 패턴을 보여줌	총 9개의 날씨 패턴을 구현함. 각각의 날씨 패턴은 우선순위에 의하여 무드등에 출력됨
	온도 모드	API를 이용하여 온도 정보를 제공받고, 이에 따라서 무드등의 색을 변화시킴	온도 정보가 색으로 표현됨
	미세먼지 모드	API를 이용하여 미세먼지 정보를 제공받고, 이에 따라서 미세먼지 패턴을 보여줌	등급에 따라 실시간으로 변화하는 미세먼지 패턴을 구현함
서버 통신	안드로이드와 서버 연결	socket i.o를 사용하여 JSON 방식으로 통신함	안드로이드에서 내린 명령이 서버에 도달함
	날씨 API	SK weather planet의 날씨 API를 사용함. Node JS를 사용해서 받은 날씨 정보를 파싱해서 주기적으로 저장함.	실시간 날씨 정보가 저장됨
	아두이노와 서버 연결	Http 방식으로 아두이노와 서버를 연결함. 통신용 아두이노와 LED용 아두이노는 시리얼로 통신함	성공적으로 서버에서 데이터를 아두이노로 넘겨줌
앱	날씨 정보	API를 이용하여 제공받은 날씨 정보를 종합하여 사용자에게 보여줌	사용자가 간편한 UI를 통하여 쉽게 24시간 안의 날씨 정보를 확인할 수 있음
	무드등 컨트롤	안드로이드에서 서버로 데이터를 넘기면, 그 데이터를 아두이노로 전송함	안드로이드에서 내린 명령으로 무드등을 컨트롤 할 수 있게 됨

## 4. 개발결과의 활용계획

### 1. 무드등 모드

무드등 모드에서는 사용자가 원하는 색상을 선택하여 feel flight에 반영할 수 있다. 사용자가 RGB값을 입력하는 것이 아닌 컬러 테이블을 이용하여 직접 눈으로 색상을 선택하기 때문에 선택이 쉽다. 하지만 앱에서 설정하는 색상과 실제 등에서 나오는 색상은 서로 완벽하게 일치하지 않으므로 사용자가 원하는 색상을 100% 표현할 수는 없다. 무드등 모드는 사용자가 원하는 색상으로 등의 색깔을 변하게 할 수 있도록 한다.

### 2. 날씨 모드

날씨 모드에서는 현재 날씨를 SK weather planet에서 받아와 등에 반영한다. 실제 날씨를 표현한다는 점에선 문제가 없지만, 사용자가 원하는 날씨를 볼 수 없다는 단점이 있다. 간단한 예로 여름에 눈이 오는 것을 볼 수 없듯이 Feel Light에서는 실제로 그 날씨가 실현되지 않으면 다른 날씨는 사용자가 원하더라도 볼 수 없다. 날씨 모드는 사용자로 하여금 눈, 비가 오거나 혹은 번개가 칠 경우 우산 등의 용품을 대비할 수 있도록 도움을 준다.

### 3. 온도 모드

온도 모드에서는 수치에 따라서 더울수록 빨강, 추울수록 파랑, 10도에 가까울수록 하얀색으로 오늘의 온도를 표현한다. 온도 차이가 심할 경우 등으로 온도의 변화를 확인하는 것이 가능하지만, 세세한 온도의 변화는 관측하기 어렵다. 하지만 극심한 온도 차이를 관측할 수 있기 때문에 사용자에게 온도가 급변할 경우에 대비할 수 있도록 한다.

### 4. 미세먼지 모드


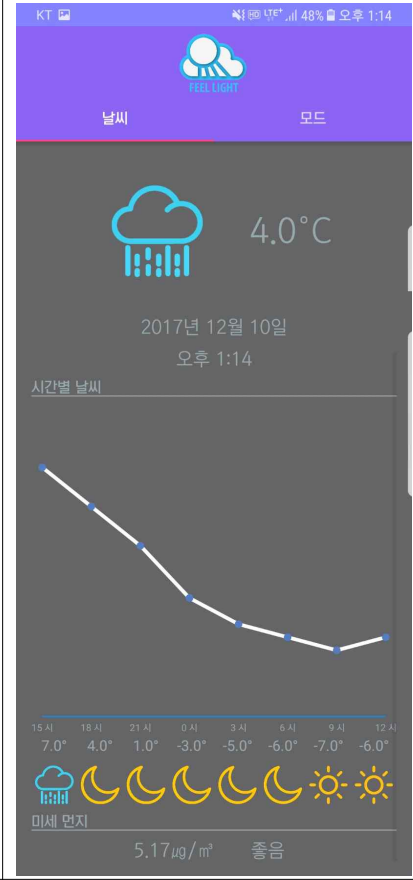
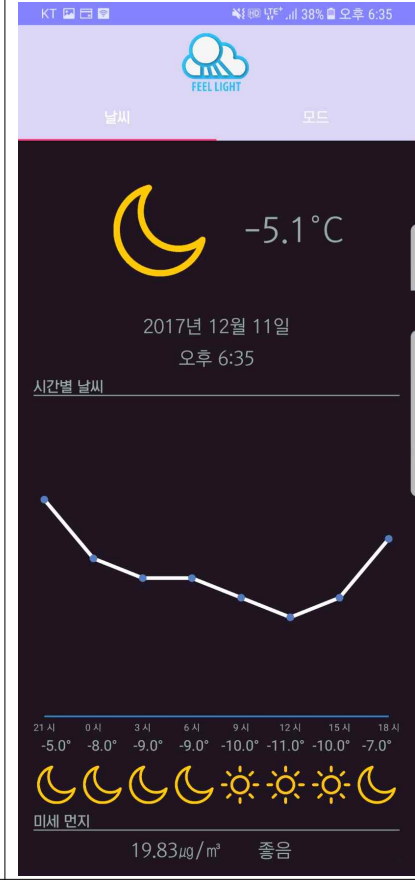
미세먼지 모드에서는 등으로 미세먼지가 날아다니는 것을 표현한다. 그리고 하단부에 미세먼지 농도에 따라 좋음 파랑, 보통 초록, 나쁨 주황, 매우 나쁨 빨강으로 사용자에게 미세먼지 농도에 따른 등급을 알려준다. 이를 통해 사용자가 미세먼지를 대비할 수 있도록 한다.

### 5. 앱 컨트롤러

앱을 통해 사용자는 무드등의 모드를 조절할 수 있으며 날씨 앱을 사용하는 것과 같이 오늘의 날씨를 시간별로 확인할 수 있고 미세먼지의 경우 농도를 수치로 확인할 수 있다.


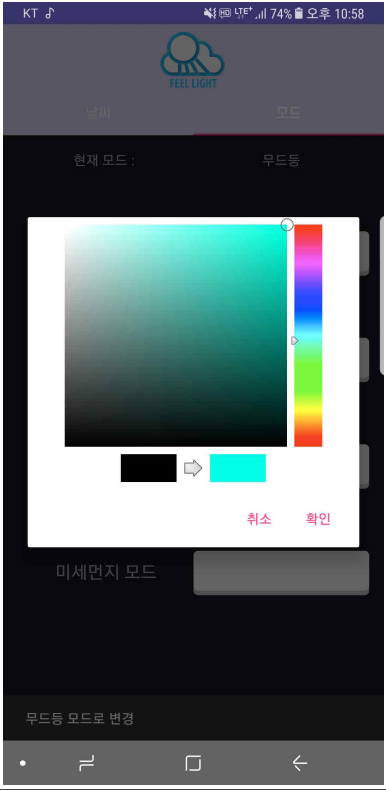
## 5. 사용자 메뉴얼

### 1. 앱 컨트롤러

		
오전(아침) 시간대	낮 시간대	밤 시간대

<표 1> 시간대에 따라 다르게 나타나는 실행화면과 날씨표현

Feel Light 어플리케이션 실행 시 시간대별로 실행 테마가 다르게 나타난다. 상단에는 현재 날씨와 기온이 출력되고, 아래에는 시간별 날씨와 날씨 아이콘, 그리고 미세먼지 값이 출력된다. 스와이프 혹은 ‘모드’ 버튼 터치 시 무드등 모드, 날씨 모드, 온도 모드, 미세먼지 모드의 4가지 모드 선택이 가능하다.

	
<p>모드 설정 화면</p>	<p>무드등 모드 - 컬러테이블 설정 화면</p>

<표 2> 모드 설정화면과 컬러테이블을 통한 사용자 정의 색 설정화면

스와이프 혹은 터치를 통해 모드 화면으로 넘어갈 경우, 사용자가 원하는 모드를 설정할 수 있다. 기본적으로 분위기 연출을 위한 무드등 모드, 날씨를 표현하는 날씨모드, 온도를 색으로 표현하는 온도모드, 미세먼지 정도를 표현하는 미세먼지 모드가 있다.

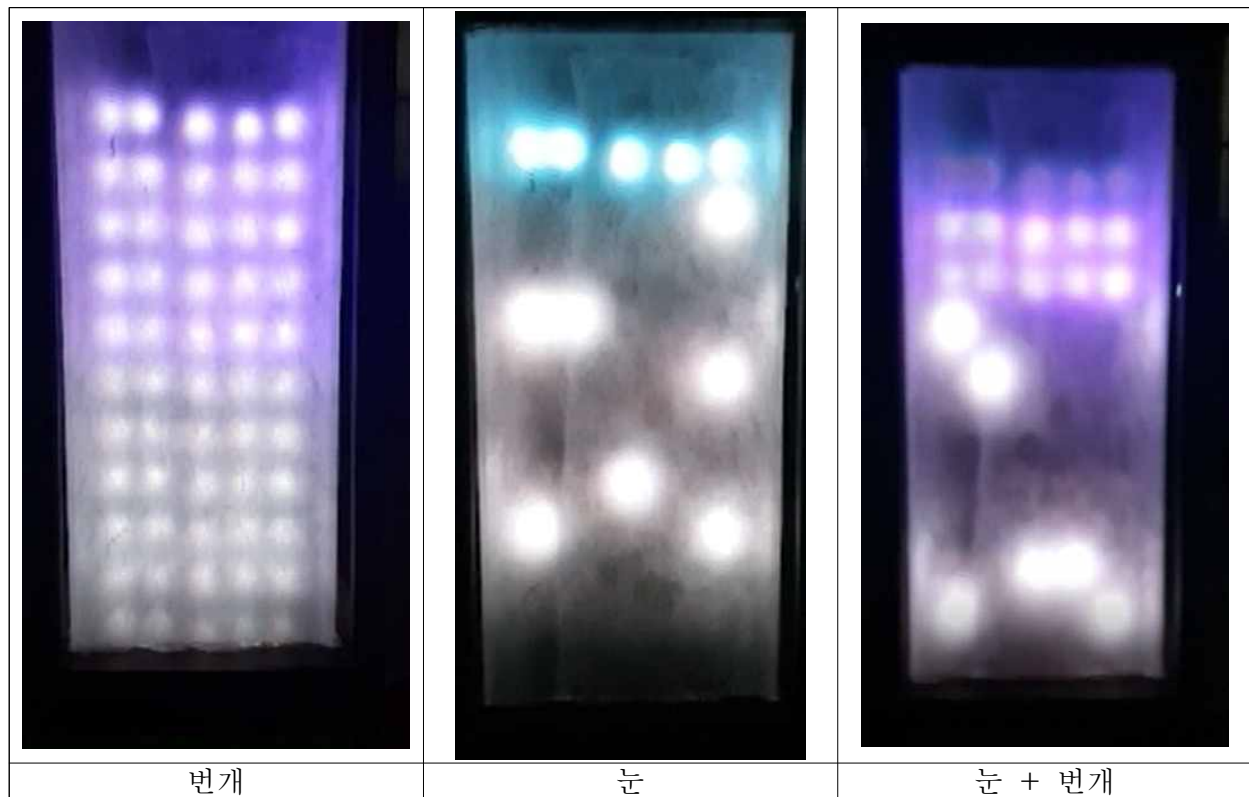
## 2. 무드등 모드

무드등 모드 터치 시 <표 2>의 두 번째 그림과 같이 나타난다. 사용자는 컬러테이블 설정을 통한 사용자 정의 색을 무드등에 적용시킬 수 있다.

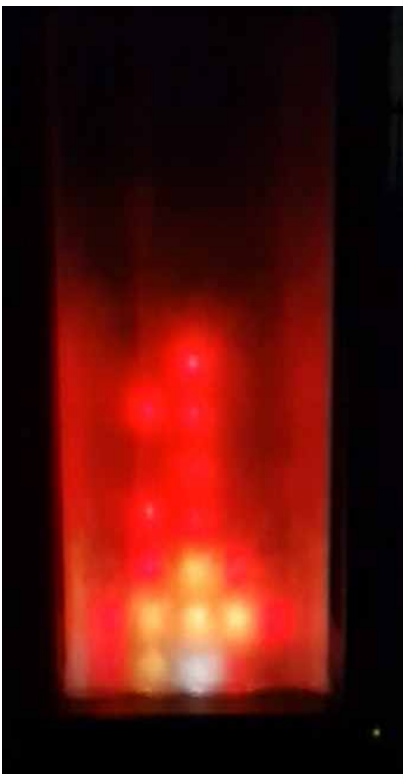


### 3. 날씨 모드



<표 3> 날씨 모드 실행 시 표현되는 날씨



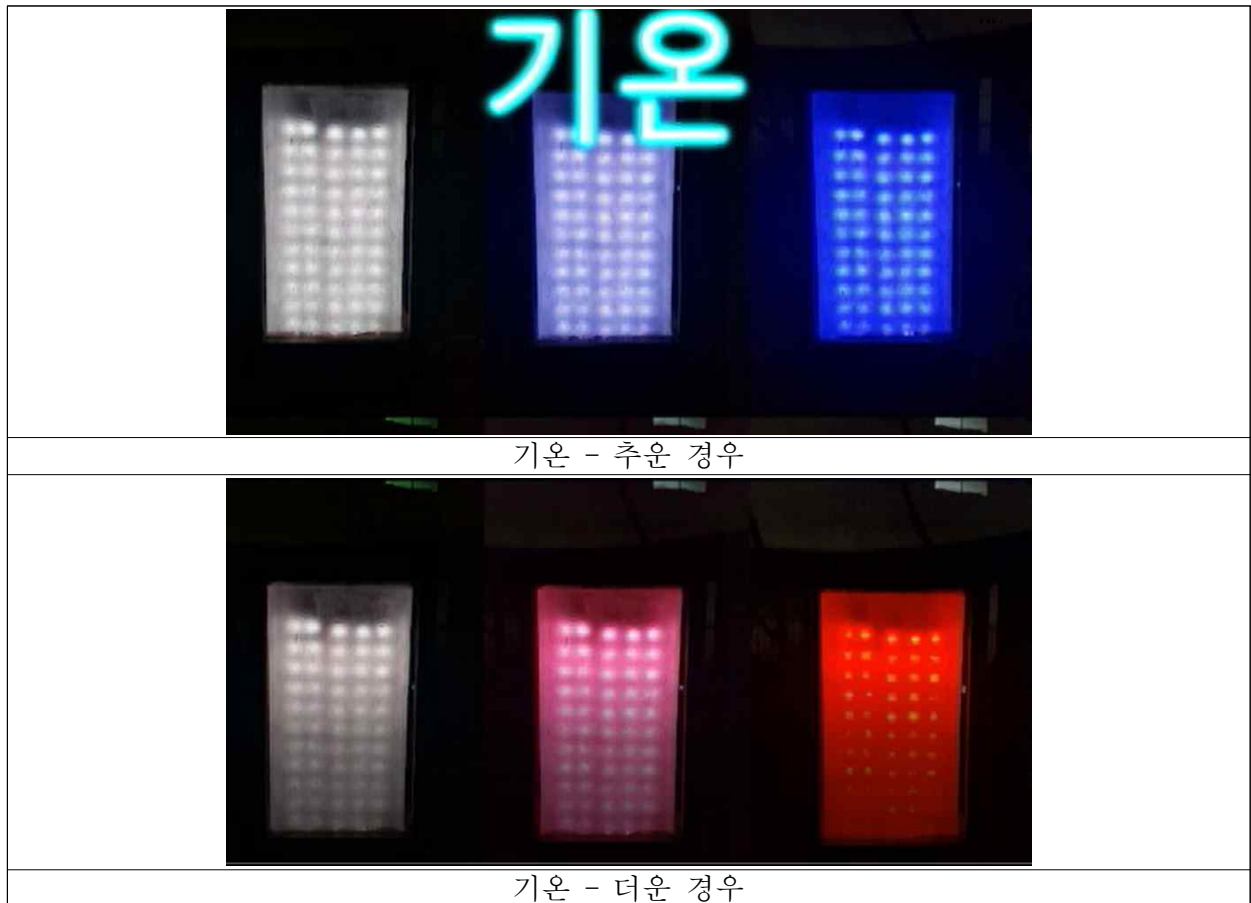
<표 4> 날씨 모드 실행 시 표현되는 날씨

		
폭염	한파	비 + 번개

<표 5> 날씨 모드 실행 시 표현되는 날씨

다음 <표 3>, <표 4>, <표 5>는 날씨 모드를 실행 시 API 데이터를 분석한 결과를 통해 날씨 데이터를 표현한다. 비와 눈은 무작위 알고리즘을 통해 LED 도트가 떨어지는 것을 표현하였고, 번개는 상단 LED의 점등. 폭염은 불꽃표현으로 나타내었고, 한파의 경우는 한 점에서 무작위로 퍼져나가며 얼음이 얼어가는 형태를 형상화한 것이다.

#### 4. 온도 모드


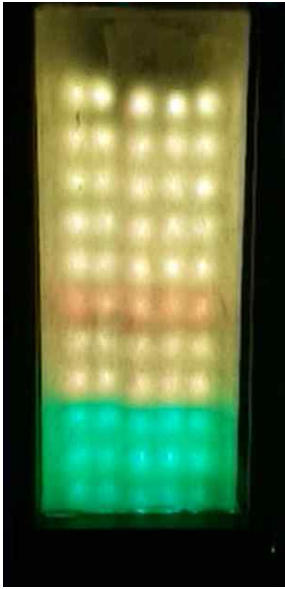

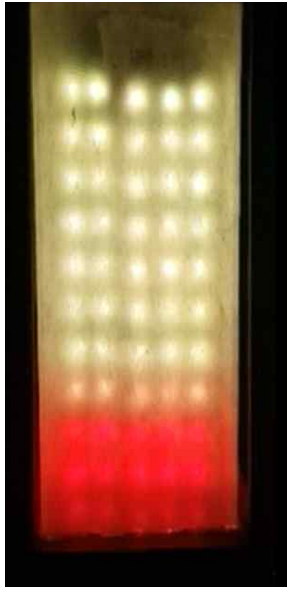


<표 6> 온도 변화에 따른 무드등 LED의 변화

온도 모드를 실행할 경우 붉은색 계열 혹은 푸른색 계열의 무드등이 나타나게 된다. 이는 10도를 기준으로 하여, 10도보다 높을 경우 RGB중 R값을 증가시키고, 10도보다 낮을 경우 B값을 증가시킨다. 따라서 최대 40도까지, 최소 영하 30도까지 표현된다. 그 이상 혹은 그 이하의 온도를 데이터로 받을 경우, 최댓값 혹은 최솟값의 출력과 동일하게 적용된다.



## 5. 미세먼지 모드

			
좋음	보통	나쁨	매우 나쁨

<표 7> 미세먼지 지수에 따른 무드등 LED의 변화

미세먼지 모드를 실행하면 SK플래닛에서 받아온 API데이터를 통해 미세먼지의 단계를 구분한다. 좋음의 경우 하단을 파란색, 보통의 경우는 초록색, 나쁨은 주황색 계열, 매우 나쁨은 하얀색 계열로 표현한다. 상단은 미세먼지가 날아다니는 모습을 표현한다.

## 6. 개발 후기

에로사항	개발하면서 가장 어려웠던 기능에 대한 극복방법
RGB LED로 인한 장애 요소	초기 계산 결과 608개 가량의 LED가 필요했다. 또한 RGB LED는 각 핀에 0~255의 값을 전달하지만 시프트 레지스터는 0 또는 1의 값만 넘겨지기 때문에 LED Strip을 사용하여 이 문제를 해결했다. 또한 LED Strip을 사용함으로써 공간을 절약하고 가격 부담을 줄일 수 있었다.
Socket 통신으로 인한 장애 요소	Android SocketIO-client 라이브러리를 사용하여 앱과 통신하는 데 성공했지만 Arduino SocketIO-client 라이브러리는 작동하지 않았다. 또한 아두이노가 Single Thread이기 때문에 LED와 동시에 컨트롤이 불가능했다. 아두이노와 서버와의 통신을 http에서 get 방식으로 바꾸고 통신용 Uno를 추가로 설치해 LED용 MEGA와 서로 시리얼 통신을 함으로써 문제를 해결하였다.
배선으로 인한 장애 요소	점퍼선이 엉키고 꼬여 회로가 혼란해졌고 점퍼선이 Bread board에서 계속 분리되었다. 이 문제는 절연 테이프를 사용하여 연관된 점퍼선끼리 분리시키고 Bread board와 접착함으로써 문제를 해결하였다.
LED 딜레이로 인한 장애 요소	LED로 만들어둔 패턴이 자연스럽게 보이지 않고 딜레이가 발생했다. 반복되는 코드를 묶고 주석을 지워 코드를 최적화 시키자 문제가 해결되었다.

팀원	개발 후기 (소감)
장원준	<p>이번 창의적 공학설계 Feel Light 제작 프로젝트는 잘 진행되고 잘 마무리된 것 같다.</p> <p>초기 알파버전부터 기획을 수정하여 최종 버전이 나올때 까지 여러가지 어려움이 있었지만 잘 해결되었다. 각 팀원들이 자신의 일에서 자발적으로 노력한 결과라고 생각한다.</p> <p>이 프로젝트의 메인 개발자로서 Feel Light 는 한 단계 도약의 발판이 되었다. 제한된 시간에 많은 기능을 구현해야하는 상황에서 프로그래밍 실력이 한 단계 성장한것 같다.</p> <p>이 프로젝트의 경험을 기반으로 다음 프로젝트때는 더욱 체계적인 프로젝트를 진행할수 있을것 같다.</p>
권혁진	<p>이번 프로젝트는 상당히 잘 진행된 것 같다. 팀원들과의 문제도 없었고 다들 성실하게 자신의 일을 했기 때문에 이만한 결과가 나올 수 있었다고 생각한다. 이번 프로젝트는 만들면서 상당히 애착이 생겨서 많은 욕심을 부렸던 것 같다. 그만큼 시간에 쫓겨 다른 일들을 포기하는 일도 생겨났지만 덕분에 상상했던 것 이상의 결과물이 나올 수 있었다. 이렇게 타이트하게 프로젝트를 준비했던 경험은 없었기 때문에 이번 프로젝트는 나에게 꽤나 많은 도움이 되었다.</p>
문태진	<p>이번 창의적 공학설계를 통하여 프로젝트를 진행하기 위해서는 꼼꼼한 기초작업이 필요하다는것은 뼈저리게 느꼈다. 초기 모델을 구상할 때 우리가 할 수 있는것과 할 수 없는것을 구분하는것 뿐만 아니라 재료들의 세심한 부분까지 신경써야한다. 우리팀은 이번 프로젝트를 성공적으로 마친것은 맞는것 같지만 초반 설계부분의 세심함이 조금 떨어져 제작을 할 때 어려움을 겪었다. 이번 프로젝트를 통하여 다음 프로젝트 부터는 그러한 세심한 부분까지 챙길수 있는 꼼꼼함과 세심함을 얻은 것 같다.</p>

<p>손용락</p>	<p>이번 창의적 공학설계 프로젝트를 진행하면서 처음으로 물품 디자인을 해 보고 물품을 설계했다. 실제로 물품을 디자인하고, 그것을 토대로 만드는 것은 처음이라 굉장히 걱정됐지만, 팀원들이 나의 설계를 무리 없이 구현한 덕분에 무사히 제품이 완성도 높게 나올 수 있었다. 중간 과정이 너무 좋아서 끝까지 욕심을 가지고 열심히 한 덕분에 좋은 결과물을 만들 수 있었다고 생각한다. 친구들과 함께 이런 커다란 프로젝트를 진행한 것이 굉장히 재밌고 새로운 경험이었다. 중간에 이탈자 없이 무사히 좋은 결과물을 만들 수 있게 정말 노력한 팀원들에게 감사하다.</p>
<p>조세현</p>	<p>이번 창의적 공학설계 프로젝트를 시작하면서 아두이노, LCD 등 하드웨어 요소가 들어간 부품들을 처음 사용하게 되었다. 제대로 작동하지 않는 라이브러리, 이유를 모른 채 생기는 회로문제는 매우 당황스러웠고, 컴파일도 되다가도 오류가 뜨고, 다시 되던 일들도 잊을 수 없다. 얼마 배운 것도 없는 상태인 1학년 2학기에 프로젝트를 시작했다는 것은 많이 아쉬웠다. 좀 더 아는 것이 많은 때 프로젝트를 진행했다면, 더 좋은 결과를 만들 수 있었을 거라고 생각한다.</p>