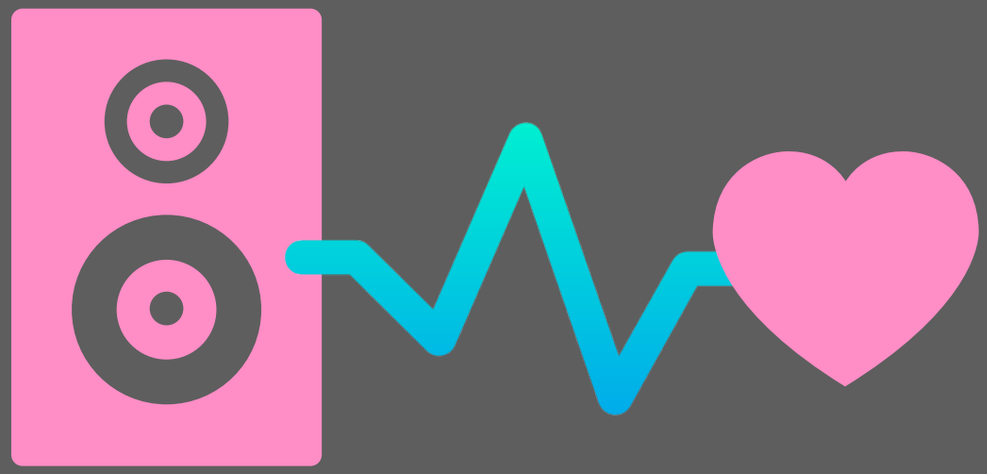


인공지능 주크박스
너의 기분은

2018.06.09 (토)

창의ICT공학설계입문 4조 **TMC**

조태환, 박종혁, 이재준, 이진욱, 이창윤, 장민혁

목차

Contents

1. 프로젝트 개요

Project Introduction

현대 사회의 고질병

Chronic Diseases of Modern Society

우울증

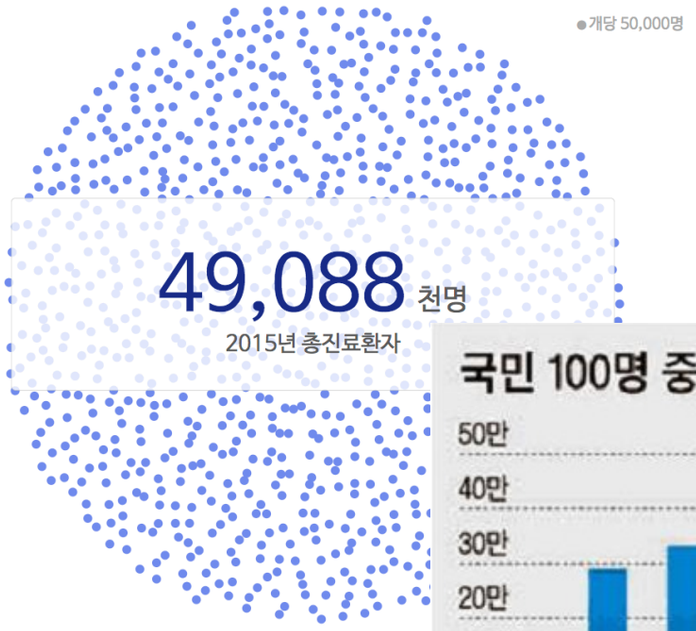
우울증 대한민국

총 진료인원

2015년 한 해 동안 병원에서 진료를 받은 전체 인원은 약 4908만 8000명이다.

이는 대한민국 약 5100만 인구(2015년 인구총조사 기준)의 96%에 이르는 수치다.

출처 | 건강보험심사평가원 의료통계정보

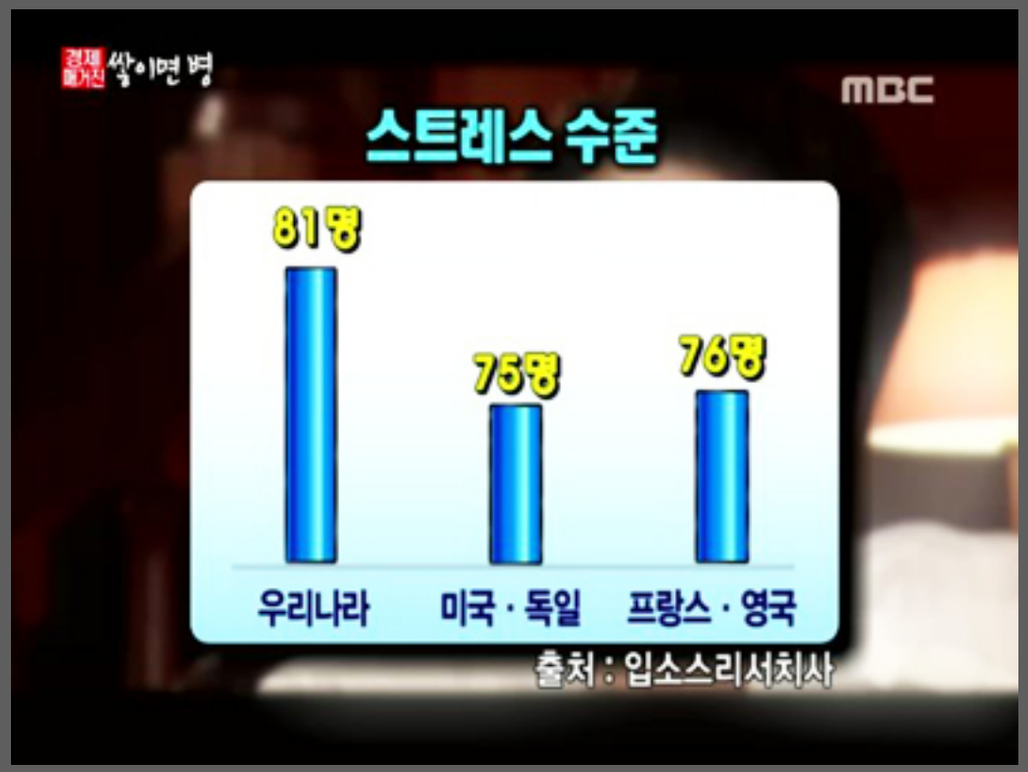


우울증 환자의 수는 해마다 점진적으로 증가하는 추세

현대 사회의 고질병

Chronic Diseases of Modern Society

스트레스성 질환



스트레스가 만성질환이 된 현대사회
(사회의 복합적인 문제들)

인공지능 주크박스

Artificial Intelligence Jukebox

현대인의 우울증과 스트레스를 치유해줄 수 있는 방법이 있을까?

"너의 기분은?"

ADS

Application Definition Statement

사람들의 현재 정서와 기분을 파악하여 맞춤형 노래를 틀어주는 스피커

5개의 감정

Five Emotions

기쁨
HAPPY

슬픔
SAD

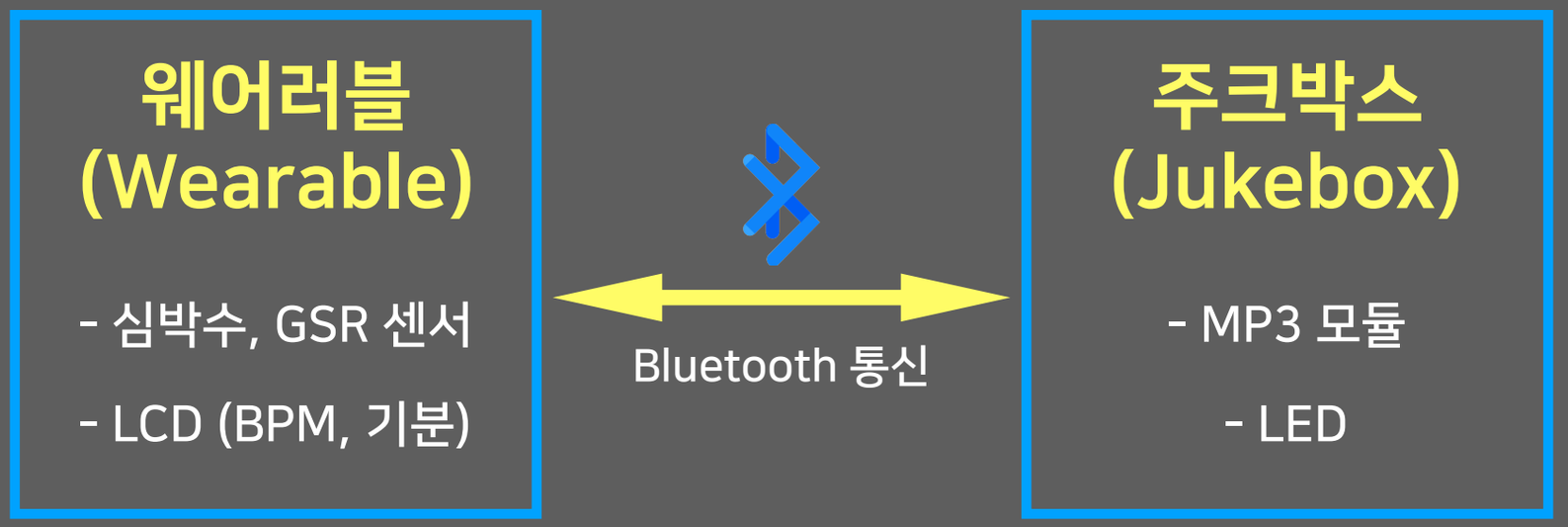
화남
ANGRY

평상
NORMAL

우울
GLOOMY

구동 방식

Working Method



사용 방식

Using Method

아두이노 나노

기분 측정

Bluetooth 통신

아두이노 우노

맞춤형
노래 재생

주크박스 기능

VOL 조정 | 재생 일시정지 | 다음곡 | 재측정

팀원 역할분담

Role Division

조태환 — LED 센서 구동 및 코드 최적화 — 5

박종혁 — 주크박스 및 웨어러블 디자인 — 5

이재준 — MP3 모듈 구동 및 코드 최적화 — 5

이진욱 — 핵심 센서 연구 — 5

이창윤 — 웨어러블 회로설계 — 5

장민혁 — 블루투스 통신 및 주크박스 회로설계 — 5

II. 하드웨어

Hardware

핵심 센서 (1) 심박수센서

Core Sensors (1) Heartrate Sensor

3.2 심박수 데이터 분석

측정한 데이터에 대하여 평상시, 슬플 때 및 기쁠 때의 평균 BPM은 각각 64.66 BPM, 65.90 BPM, 65.32 BPM으로 나타났다. 측정 결과에 따르면 세 분위기 모두 BPM 평균이 비슷하다는 사실을 알 수 있었다. 하지만 BPM 그래프를 살펴보면 비교적 일정한 평상시 및 슬플 때와 달리 기쁠 때의 그래프가 변화가 큼을 알 수 있다.

이에 표준 편차를 통해 각 분위기별 BPM 변화 정도를 분석해 본 결과 평상시 3.89, 슬플 때 4.29, 기쁠 때 9.08 값을 얻을 수 있었다. 기쁜 경우에는 평상시와 슬픈 경우보다 확실히 BPM 값 변화 정도가 큼을 알 수 있었지만, 슬픈 경우에는 평상시와 값의 차이를 별반 느끼지 못하였으며, 좀 더 다양한 분석 및 실험을 통하여 보완할 필요성이 있다.

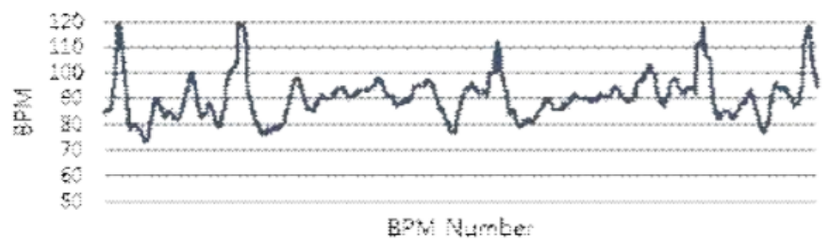


그림 5. 기쁠 때 심박변이도
Fig. 5. Heart rate variability of Joy

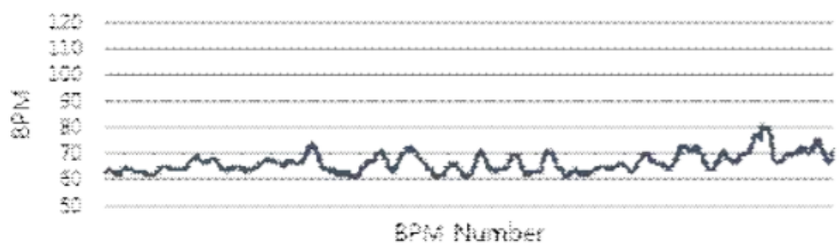


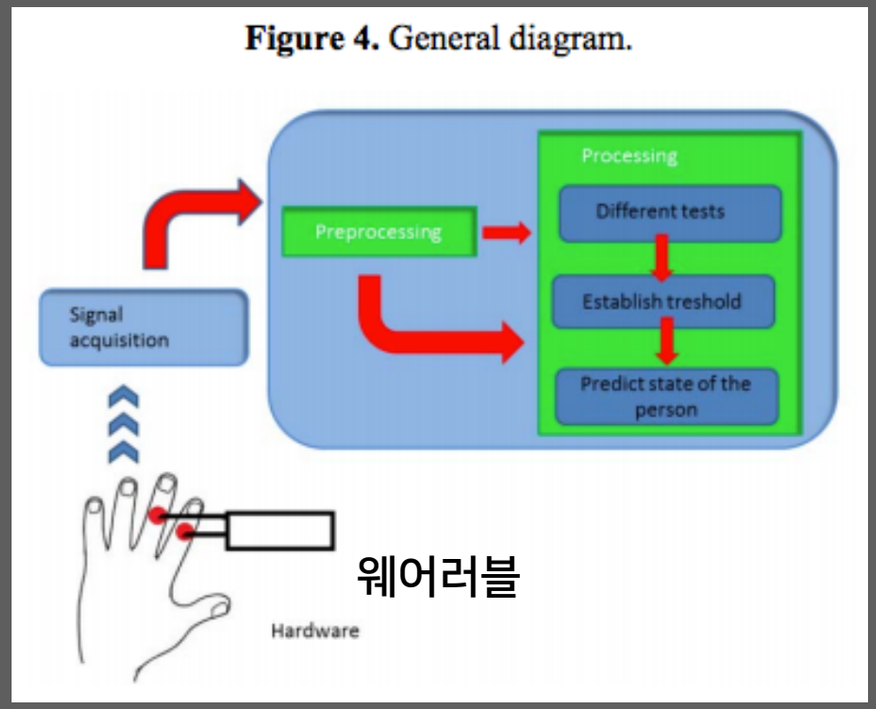
그림 6. 슬플 때 심박변이도
Fig. 6. Heart rate variability of Sadness

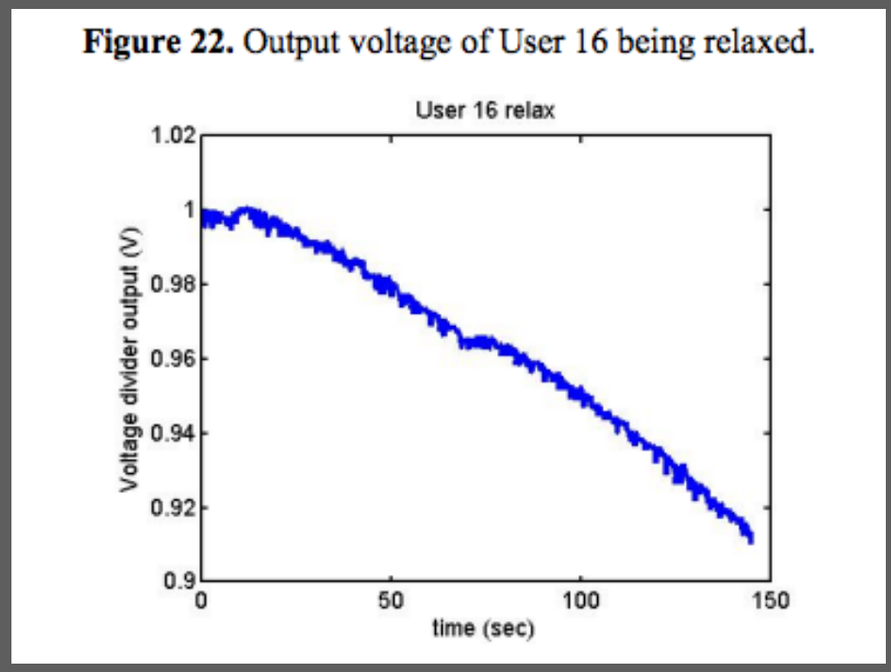
사람마다 심박수의 평균정도가 다르다는 문제점

핵심 센서 (2) GSR센서

Core Sensors (2) Galvanic Skin Sensor

- 땀 배출량과 체온 변화 측정 => 피부 긴장도를 파악
- 악몽을 꾸거나, 스트레스를 받거나, 예상치 못한 상황을 겪게 되면 피부의 컨덕턴스가 변화

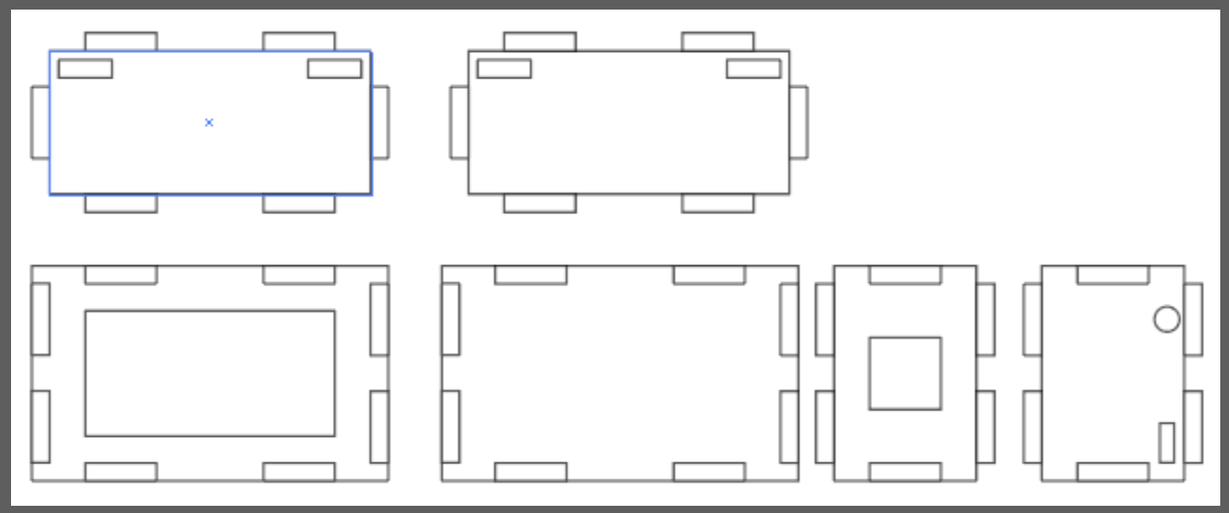




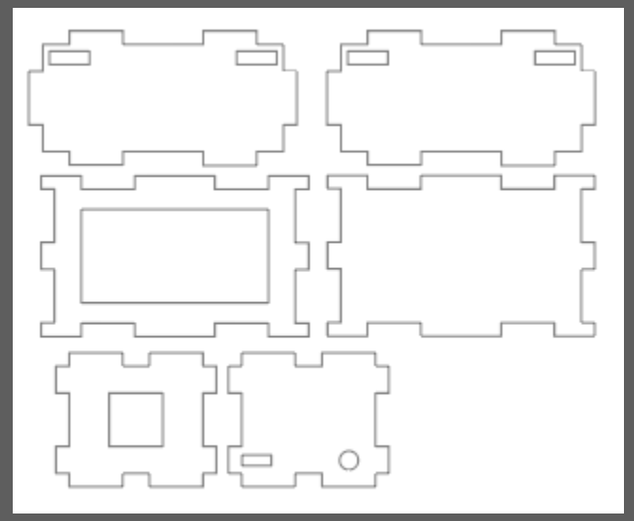
사람마다 다른 심박수 편차를 GSR 센서가 보완해주는 방식 선정

웨어러블 설계도

Blueprint of Wearable



초기 설계도



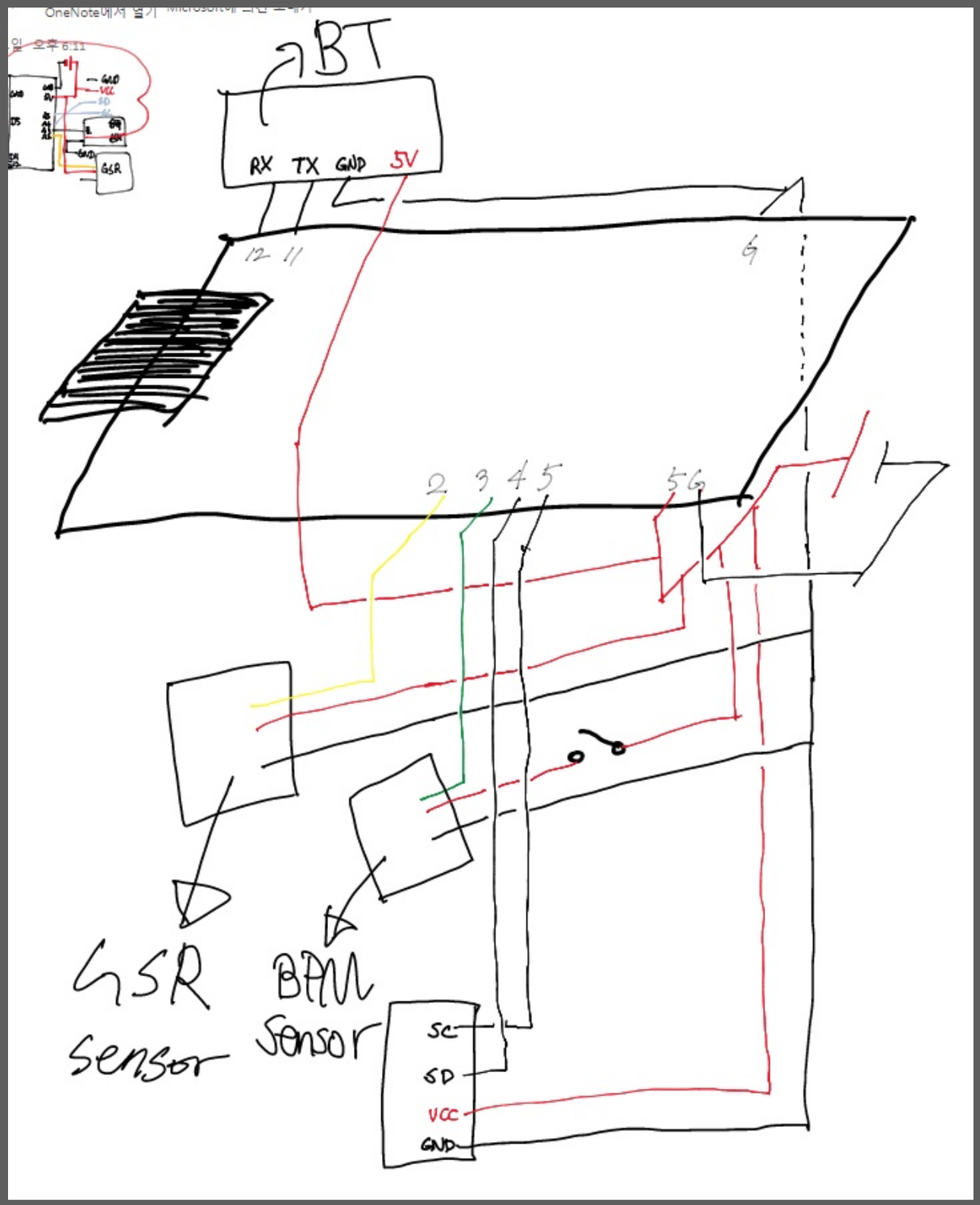
최종 설계도

웨어러블 회로도

Circuit Diagram of Wearable

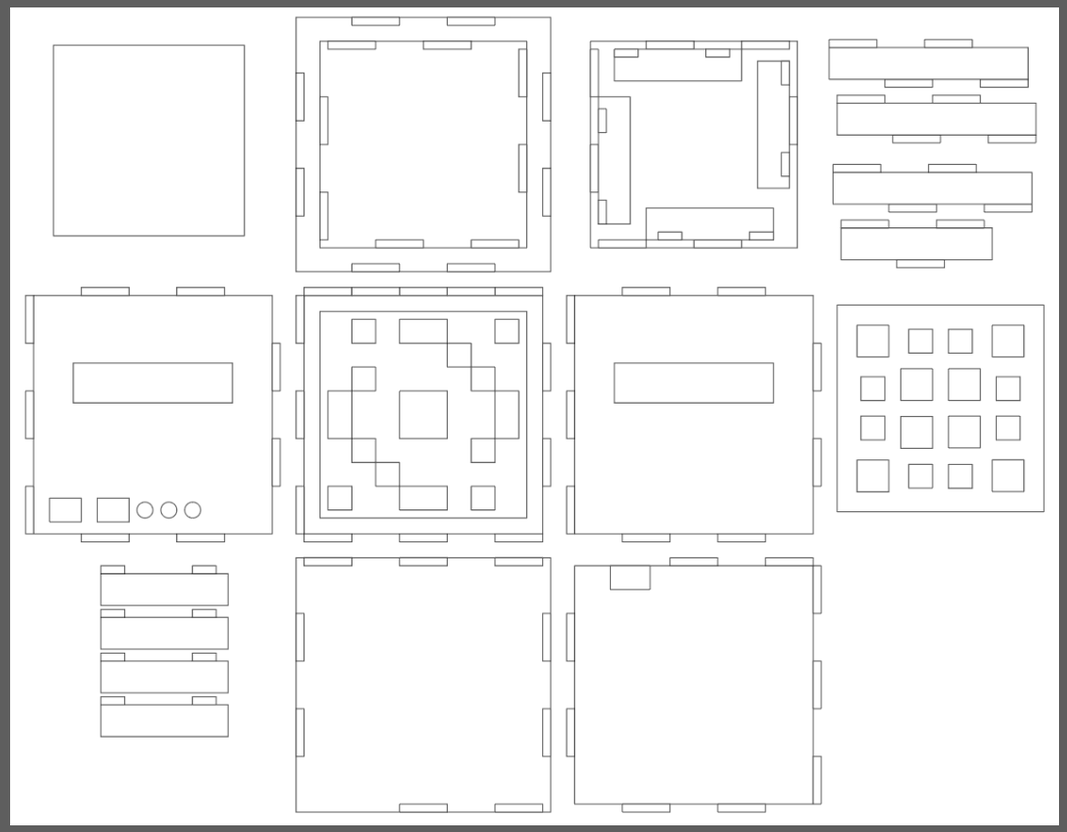
아두이노 나노
핀 번호 설정

```
#define BPMSENSORPIN  A3
#define GSRSENSORPIN  A2
#define BT_TXD  12
#define BT_RXD  11
```

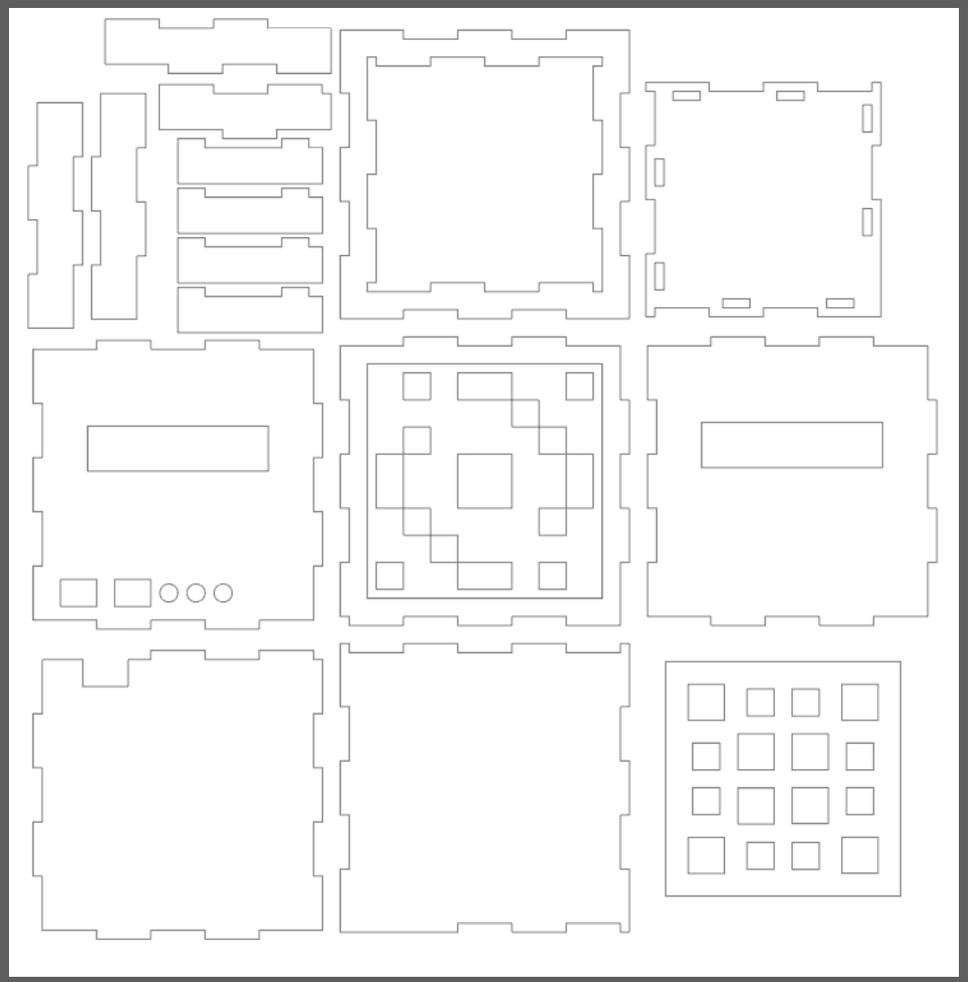


주크박스 설계도

Blueprint of Jukebox



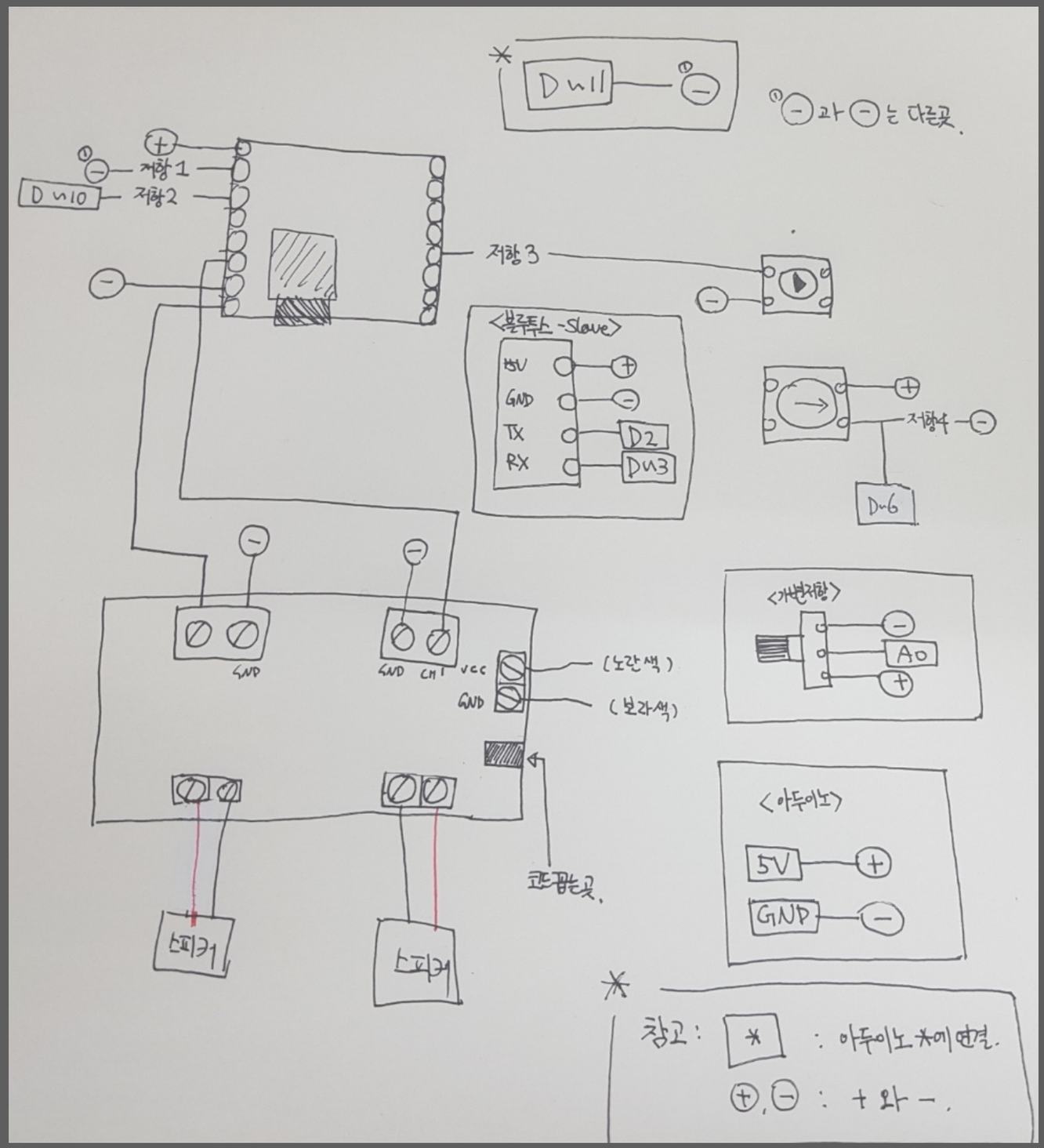
초기 설계도



최종 설계도

주크박스 회로도

Circuit Diagram of Jukebox



아두이노 우노 핀 번호 설정

```
//Pin definition
#define BT_TXD 2
#define BT_RXD 3
#define NEOPIN 9
#define MP3_TXD 10
#define MP3_RXD 11
#define VOLVRREST A1
#define LEDVRREST A2
#define RE_MEASURE 12
```

III. 소프트웨어

Software

라이브러리

Libraries

TMCHeartBeat.cpp

getHRState

```
byte getHRState(uint8_t BPM){//심박수에 따라 state_hrt 분류
        byte state_hrt;

        if(BPM <= 55)
                state_hrt = 1;
        else if(BPM <= 76)
                state_hrt = 2;
        else if(BPM <= 150)
                state_hrt = 3;
        else
                state_hrt = 4;

        return state_hrt;
}
```

심박수센서로부터 “BPM”값 수신
=> state_hrt 분류

getGRState

```
byte getGRState(uint8_t GSR, uint8_t IGSR){
        int Value = abs((int)GSR - (int)IGSR);
        byte state_gsr = 0;
        if(Value < DEVIATION)
                state_gsr = 5;
        else
                state_gsr = 6;

        return state_gsr;
}
```

GSR센서로부터 “Value”값 수신
=> state_gsr 분류

라이브러리

Libraries

TMCHeartBeat.cpp

state_hrt

+

state_gsr

getStateValue

```
byte getStateValue(byte state_hrt, byte state_gsr){
        byte state = -1;

        /* <기분에 따른 반환값>
        *  7 우울 gloomy
        *  8 슬픔 sad
        *  9 평범/평소 normal
        *  10 기쁨 happy
        *  11 화남 angry
        *  -1 ERROR
        */
        if(state_gsr == 5){
                if(state_hrt == 2)
                        state = NORMAL;
                else if(state_hrt == 3)
                        state = HAPPY;
                else
                        state = ERROR;
        }
        else if(state_gsr == 6){
                if(state_hrt == 1)
                        state = GLOOMY;
                else if(state_hrt == 2)
                        state = SAD;
                else if(state_hrt == 3 || state_hrt == 4)
                        state = ANGRY;
                else
                        state = -1;
        }
        else
                state = -1;
        return state;
}
```

웨어러블 코드

Source Code of Wearable

```
while(!isGetState){
  BPM = getBPM(BPMSENSORPIN, &lcd);
  states[0] = getHRState(BPM);
  Serial.print("\nBPM: ");
  Serial.println(BPM);
  Serial.print("BPMState: ");
  Serial.println(states[0]);
  GSR = getGSR(GSRSENSORPIN);
  states[1] = getGRState(GSR, initialGSR);
  Serial.print("\nGSR: ");
  Serial.println(GSR);
  Serial.print("GSRState: ");
  Serial.println(states[1]);
  states[2] = getStateValue(states[0], states[1]);
  Serial.print("Final State: ");
  Serial.println(states[2]);
  if(states[2] != (byte)-1){
    isGetState = true;
  }
  Serial.println(isGetState);
  Serial.println("ALL State Calculated!");
}
```

NanoPart.ino

1. 심박수, GSR 센서값 받기

임의로 계속 변하는 BPM값 해결 방법 (라이브러리 코드)

목적

신뢰할만한 값을 받아왔는지 판단

방법

임시값을 저장하기 위한 배열 생성
4회 같으면 신뢰하는 것으로 판단

웨어러블 코드

Source Code of Wearable

2. 웨어러블 LCD 화면에 상태 출력 및 블루투스 통신

아두이노 우노로 states[2] (기분 상태) 전송 Using btSerial.write()

```
btSerial.write(states[2]);
delay(100);
btSerial.write(BPM);
Serial.println("Sent");
```

```
lcd.setCursor(0, 0);
lcd.print("Heartrate: ");
lcd.setCursor(10, 0);
int temp = ((int)BPM % 1000) / 100;
if(temp)
  lcd.print(temp);
else
  lcd.print(" ");
temp = ((int)BPM % 100) / 10;
lcd.print(temp);
temp = (int)BPM % 10;
lcd.print(temp);
lcd.setCursor(13, 0);
lcd.print("bpm");
lcd.setCursor(0, 1);
lcd.print("Feeling: ");
lcd.setCursor(10,1);
switch(states[2]){
  case GLOOMY:
    lcd.print("GLOOMY");
    break;
  case SAD:
    lcd.print("   SAD");
    break;
  case NORMAL:
    lcd.print("NORMAL");
    break;
  case HAPPY:
    lcd.print(" HAPPY");
    break;
  case ANGRY:
    lcd.print(" ANGRY");
    break;
  default:
    lcd.print(" ERROR");
}
```

주크박스 코드

Source Code of Jukebox

UnoPart.ino

1. MP3 모듈 제어

```
//void loop() -- MP3   아날로그 값(1023) -> 볼륨 값(30) 비율 조정
int sensorValue;
sensorValue = analogRead(VOLVRREST);
modifiedValue = (int)(30 * sensorValue / 1023);

if(previous_modifiedValue != modifiedValue) {
  mp3_set_volume(modifiedValue);
  previous_modifiedValue = modifiedValue;
}
```

재생/일시정지, 다음곡버튼은 모듈 내부 제어
아두이노 우노에서 볼륨 비율 조정 제어

기분에 따라 노래 재생
startMusic() 함수

```
void startMusic(){
  if(feelingState == HAPPY){
    uint8_t feelA = random(1,50);
    mp3_play(feelA);
  }
  else if(feelingState == SAD){
    uint8_t feelB = random(51,80);
    mp3_play(feelB);
  }
  else if(feelingState == NORMAL){
    uint8_t feelC = random(81,110);
    mp3_play(feelC);
  }
  else if(feelingState == GLOOMY){
    uint8_t feelD = random(111,140);
    mp3_play(feelD);
  }
  else if(feelingState == ANGRY){
    uint8_t feelE = random(141,170);
    mp3_play(feelE);
  }
  else ;
}
```

주크박스 코드

Source Code of Jukebox

2. LED 제어

```
//void loop() -- LED부분
targetChange();
colors[0] = getLEDBright(target[0]);
colors[1] = getLEDBright(target[1]);
colors[2] = getLEDBright(target[2]);
for(int i=0; i<NUMPIXELS; i++) {
  pixels.setPixelColor(i, colors[0], colors[1], colors[2]);
  pixels.show();
}
```

LED 숨쉬기 효과 구현
(BPM에 따라 숨쉬기 주기 변화)

블루투스 통신으로 아두이노 나노에서 받아온 feelingState(기분)값

```
void targetChange(){
  if(feelingState == NORMAL){
    target[0] = colorSEO[0];
    target[1] = colorSEO[1];
    target[2] = colorSEO[2];
  }
  else if(feelingState == SAD){
    target[0] = colorDAL[0];
    target[1] = colorDAL[1];
    target[2] = colorDAL[2];
  }
  else if(feelingState == HAPPY){
    target[0] = colorOMM[0];
    target[1] = colorOMM[1];
    target[2] = colorOMM[2];
  }
  else if(feelingState == GLOOMY){
    target[0] = colorGLA[0];
    target[1] = colorGLA[1];
    target[2] = colorGLA[2];
  }
  else if(feelingState == ANGRY){
    target[0] = colorYOU[0];
    target[1] = colorYOU[1];
    target[2] = colorYOU[2];
  }
  else ;
}
```

NORMAL
노랑

SAD
파랑

HAPPY
초록

GLOOMY
보라

ANGRY
빨강

기분 별 LED 색변경 구현

IV. 보완점과 기대효과

Improvement & Expectation

보완점 계획

Things to Improve

1 스피커 업그레이드

2 노래추천 알고리즘 확장

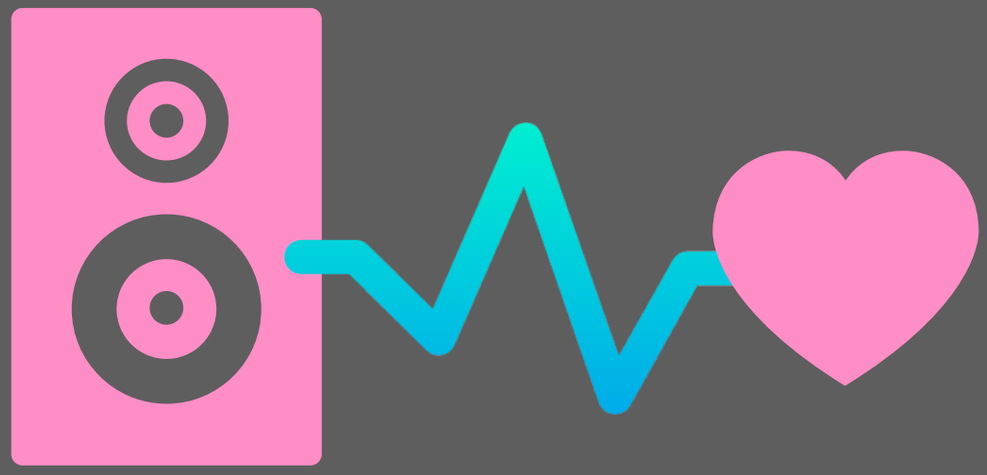
3 웨어러블 소형화

4 Github 오픈소스 Public 전환

https://github.com/OMMANT/TMC_MoodSpeaker

기대효과

Our Expectations



기존에 출시된 인공지능 스피커와 달리

사람의 감정도 보살펴주는 감성적인 스피커

V. 시연

Hands-On

감사합니다.

Presentation used for 'Creative ICT Engineering Design' class, June 9th, 2018. Created with Keynote by Jaejun Lee.

AI_Mood_Speaker
- TMC(Too Much Coding)