

띵동 - 음성인식기반 엘리베이터 연동 앱



IT기술로 세상을 “잇(IT)다”



201931043
오영선



201931044
민지수



201931068
이혜민



201931073
윤미나

언택트 방식으로



엘리베이터 이용하자 !

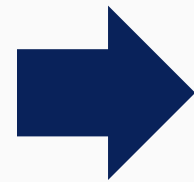
음성인식 기반 엘리베이터 연동 앱 “띵동”



기술배경

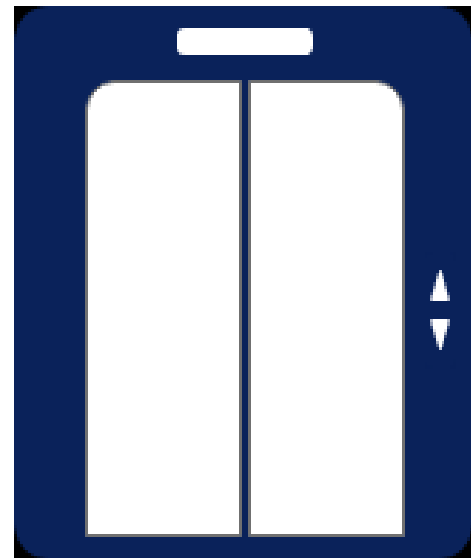
4차 산업 혁명의 핵심 기술인 사물 인터넷(IoT)을
우리의 일상생활에 접목하는 경우 많아짐

누구나 이용하는 이동수단인 엘리베이터의 경우에도
IoT기술을 활용하려는 시도를 보이고 있음



코로나 19로 인해 언택트 시대가 도입한
지금 엘리베이터 버튼을 **손으로 누르지 않고 이용할 수 있는 기술**이 필요

딩동 흐름도



엘리베이터

① 무선통신(Bluetooth)를 통해
층 수 값 전송하여 엘리베이터 제어



① 무게 센서를 이용해 무게 측정 후,
블루투스로 무게 값 전송
② LCD를 통한 층수 확인

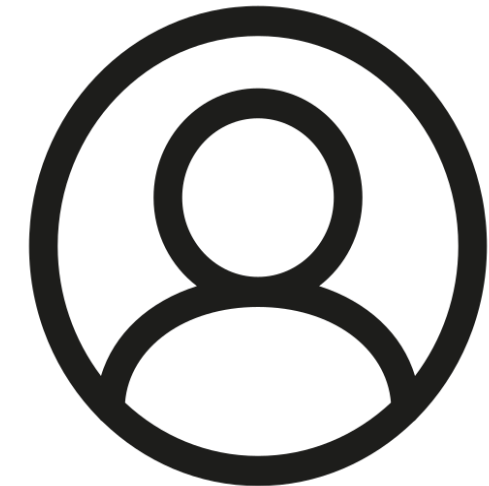


딩동 앱

① 도착 층 수
음성 입력 or 직접 입력



① 혼잡도 4단계 안내
② 음성을 통한 설명서 제공



앱 사용자

| 기능



음성입력

음성을 통한
엘리베이터 제어



직접입력

직접 입력을 통한
엘리베이터 제어



혼잡도

엘리베이터 무게 측정과
혼잡도 제공

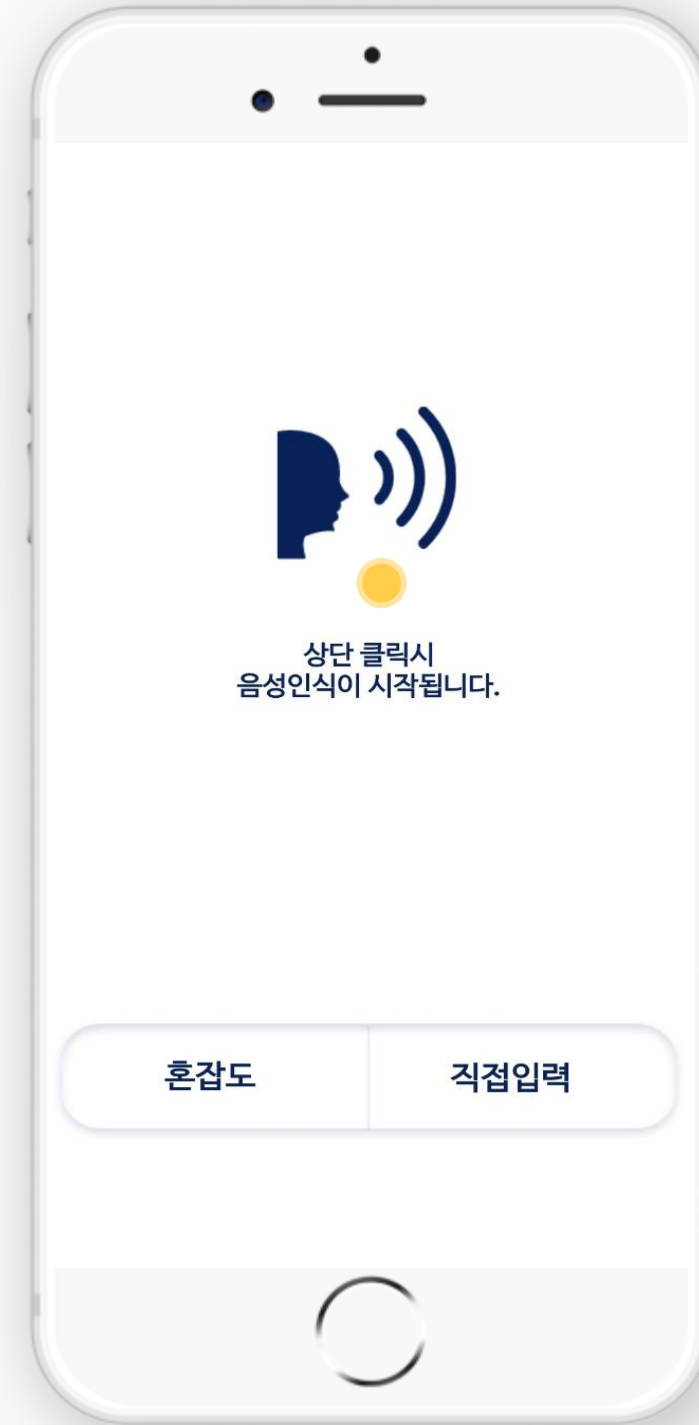


음성안내

앱 설명서 음성 안내

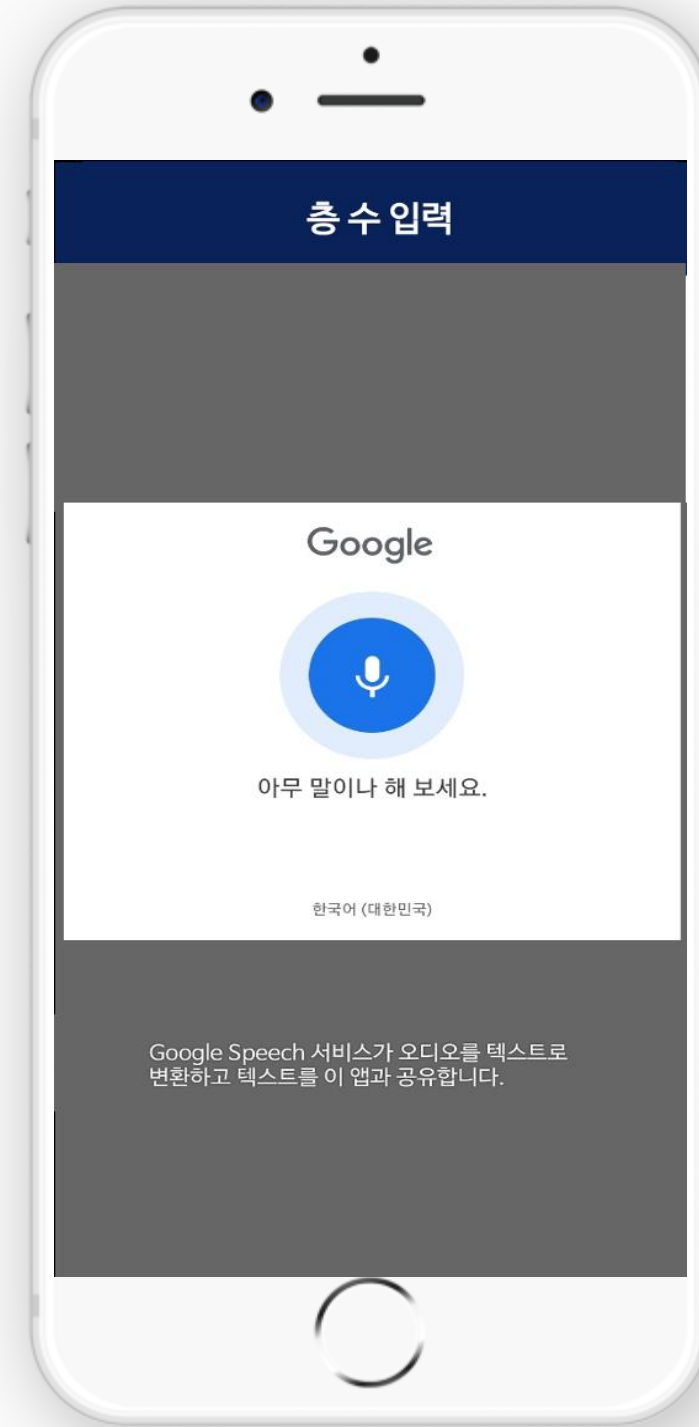
음성입력

화면 상단 클릭 시 음성입력 사용 가능



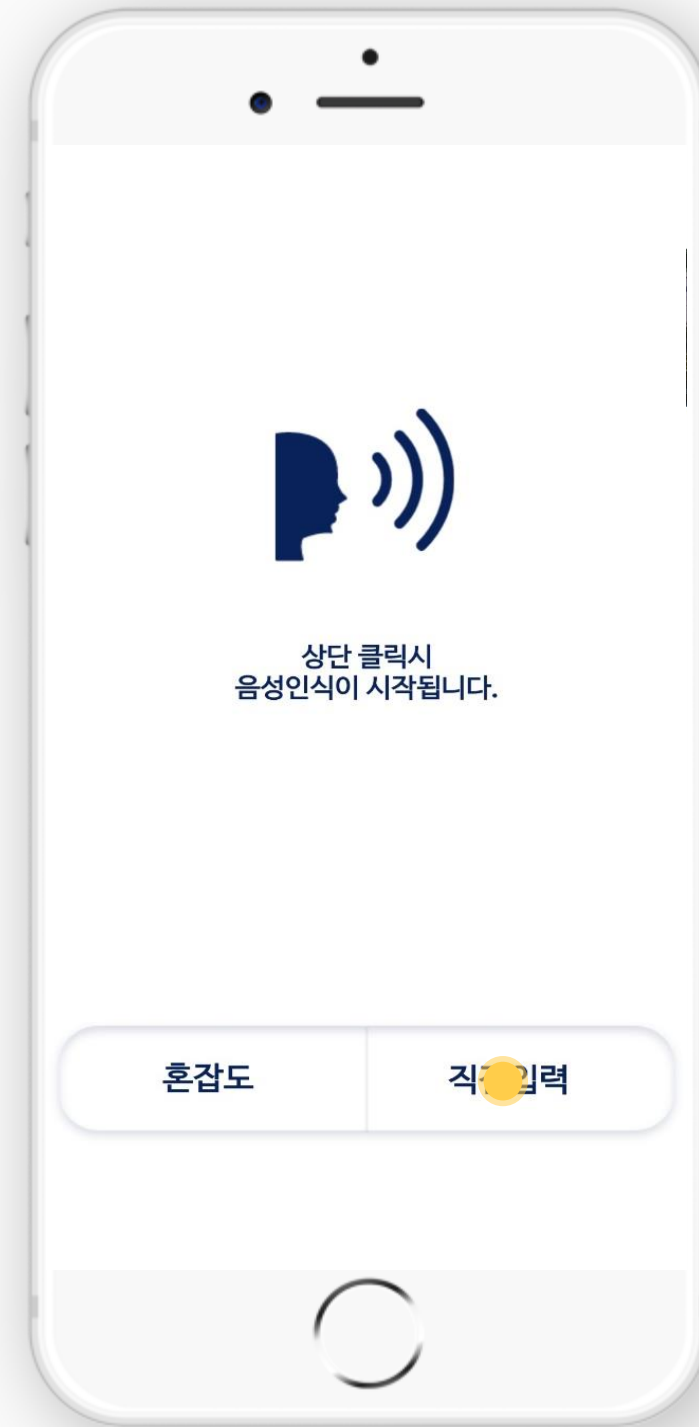
음성입력

“원하시는 층수를 말해주세요.”

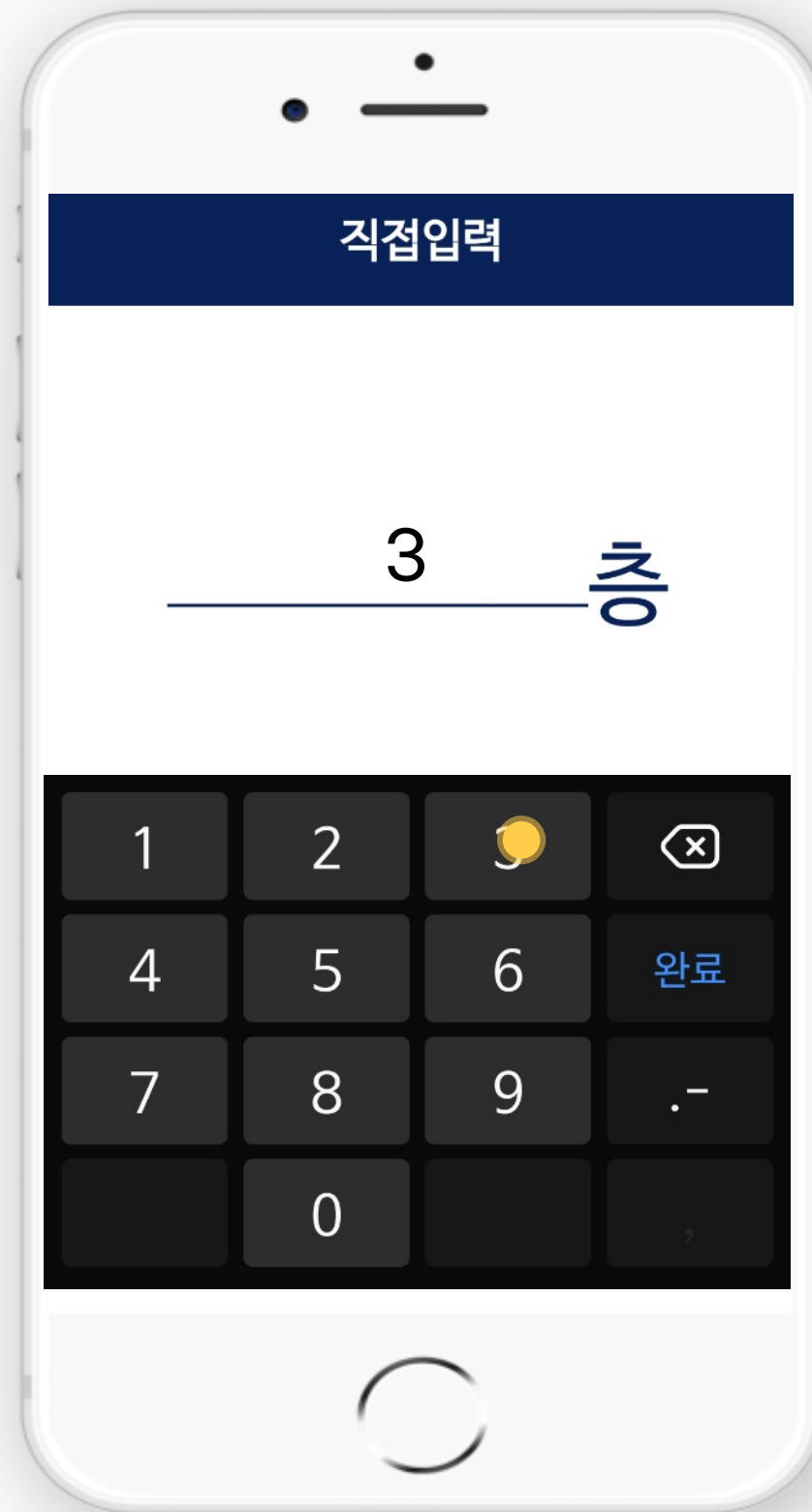


직접입력

화면 우측 하단 클릭 시 직접입력 사용 가능

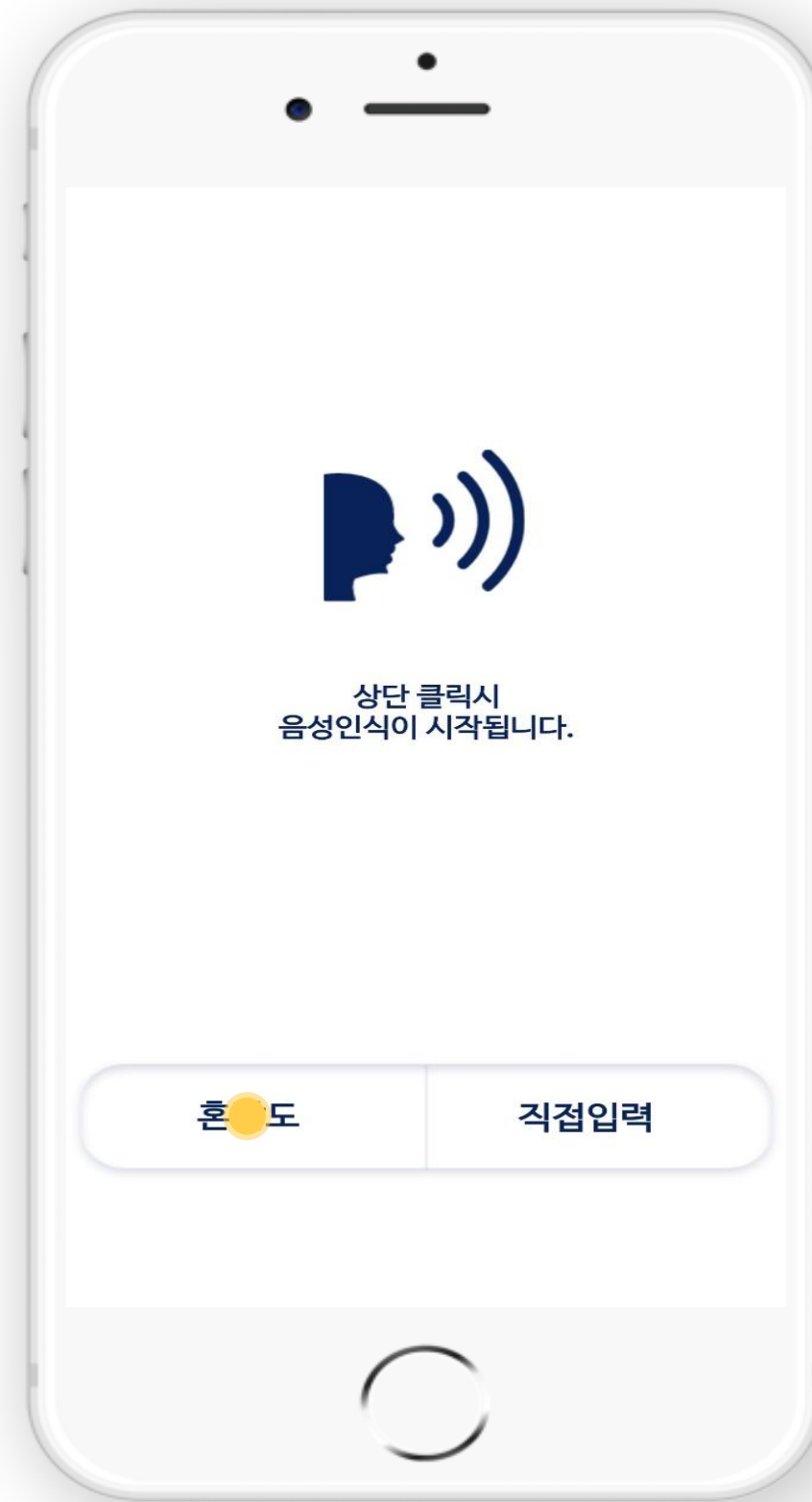


직접입력



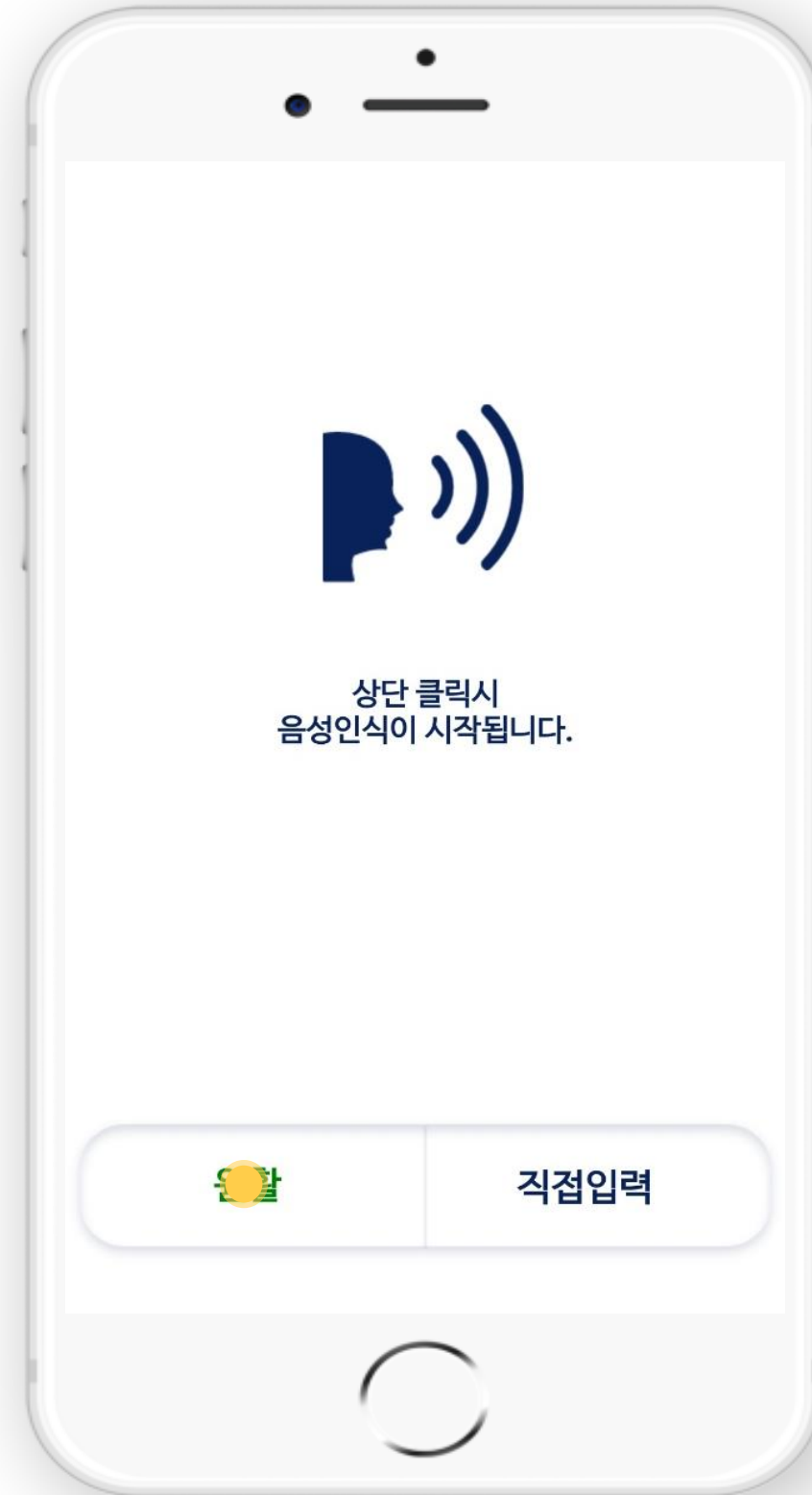
혼잡도

화면 좌측 하단 클릭 시 혼잡도 안내 음성 출력



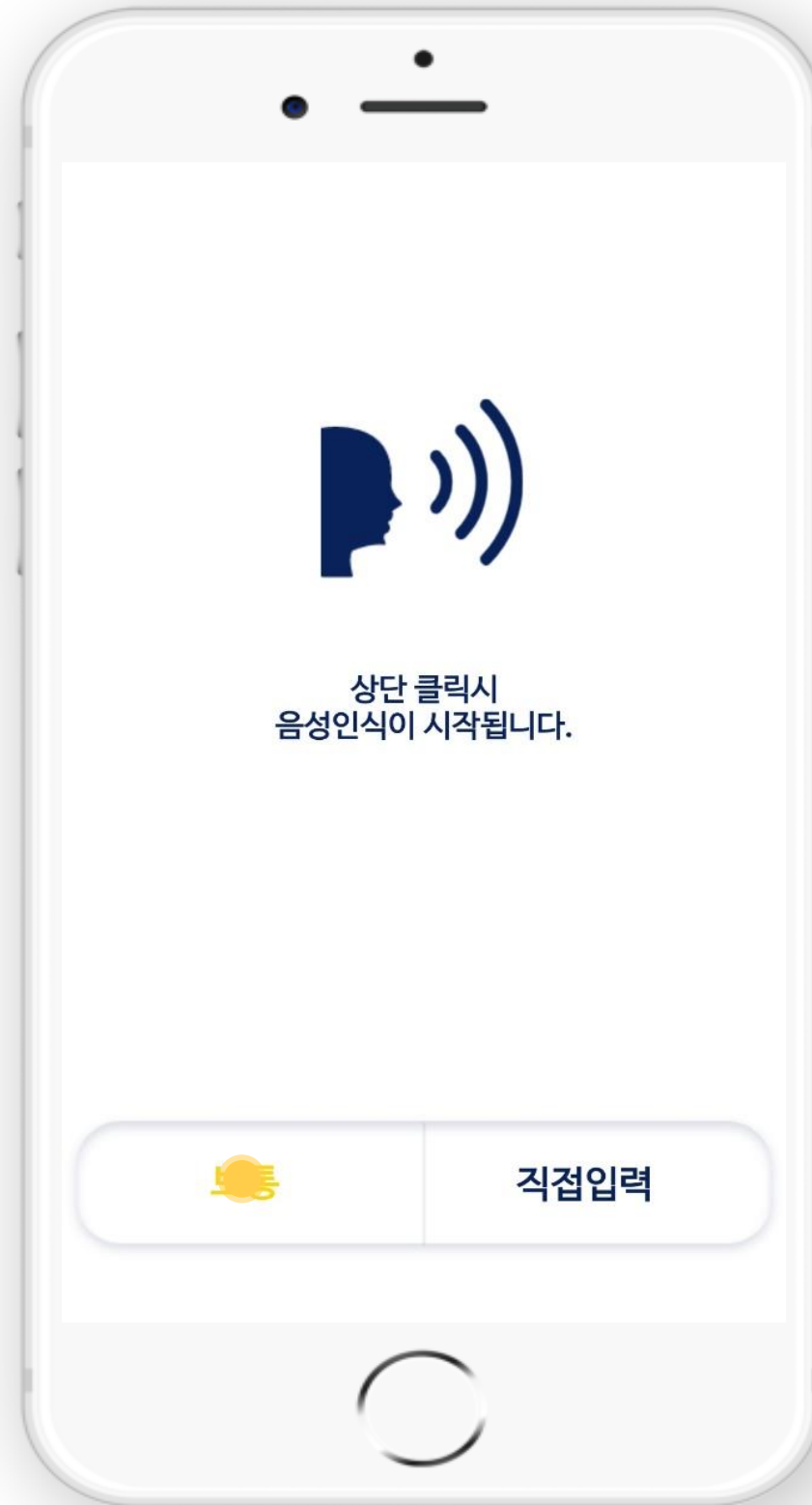
혼잡도

“원활상태 입니다.”



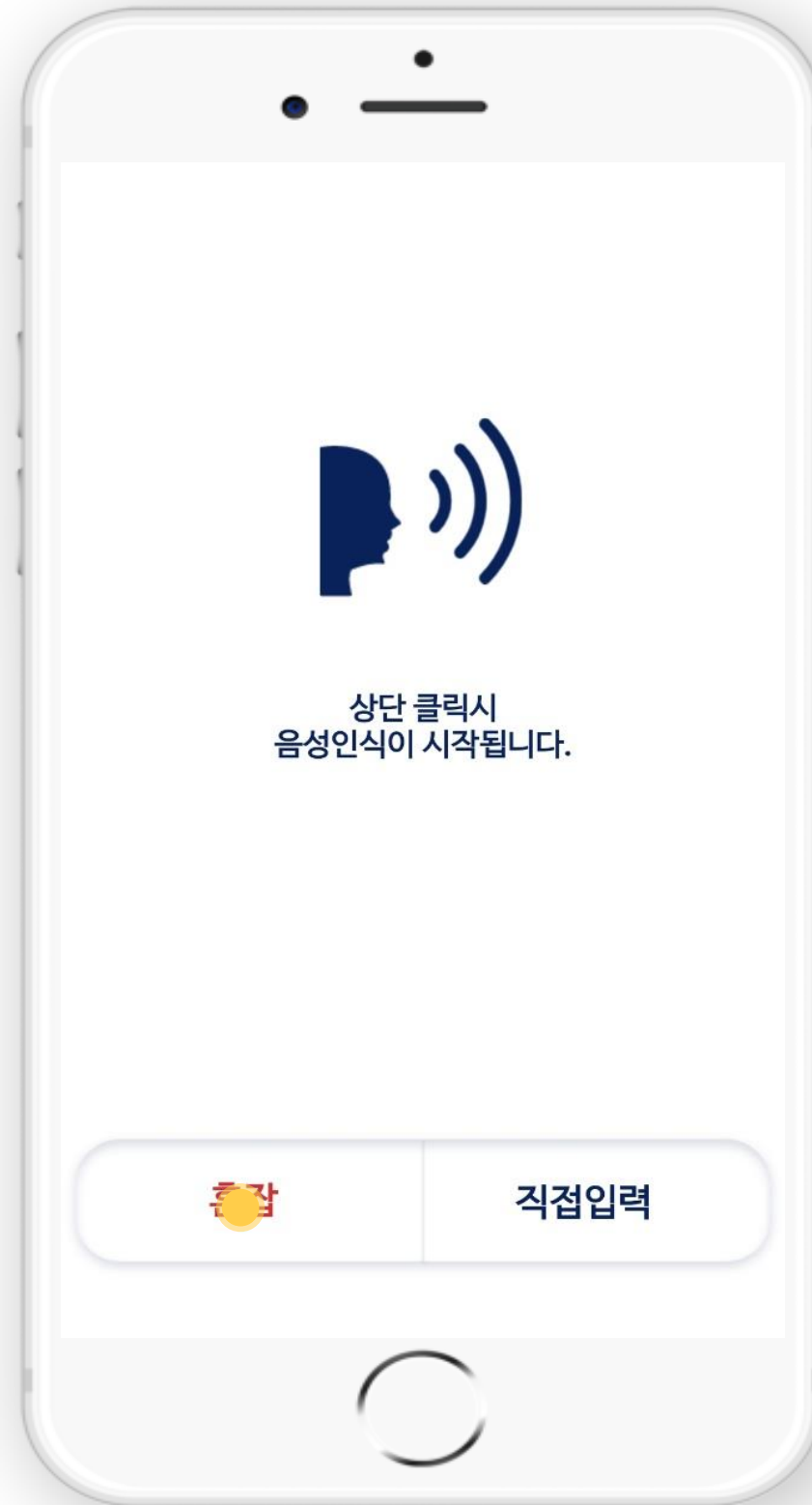
혼잡도

“보통상태 입니다.”



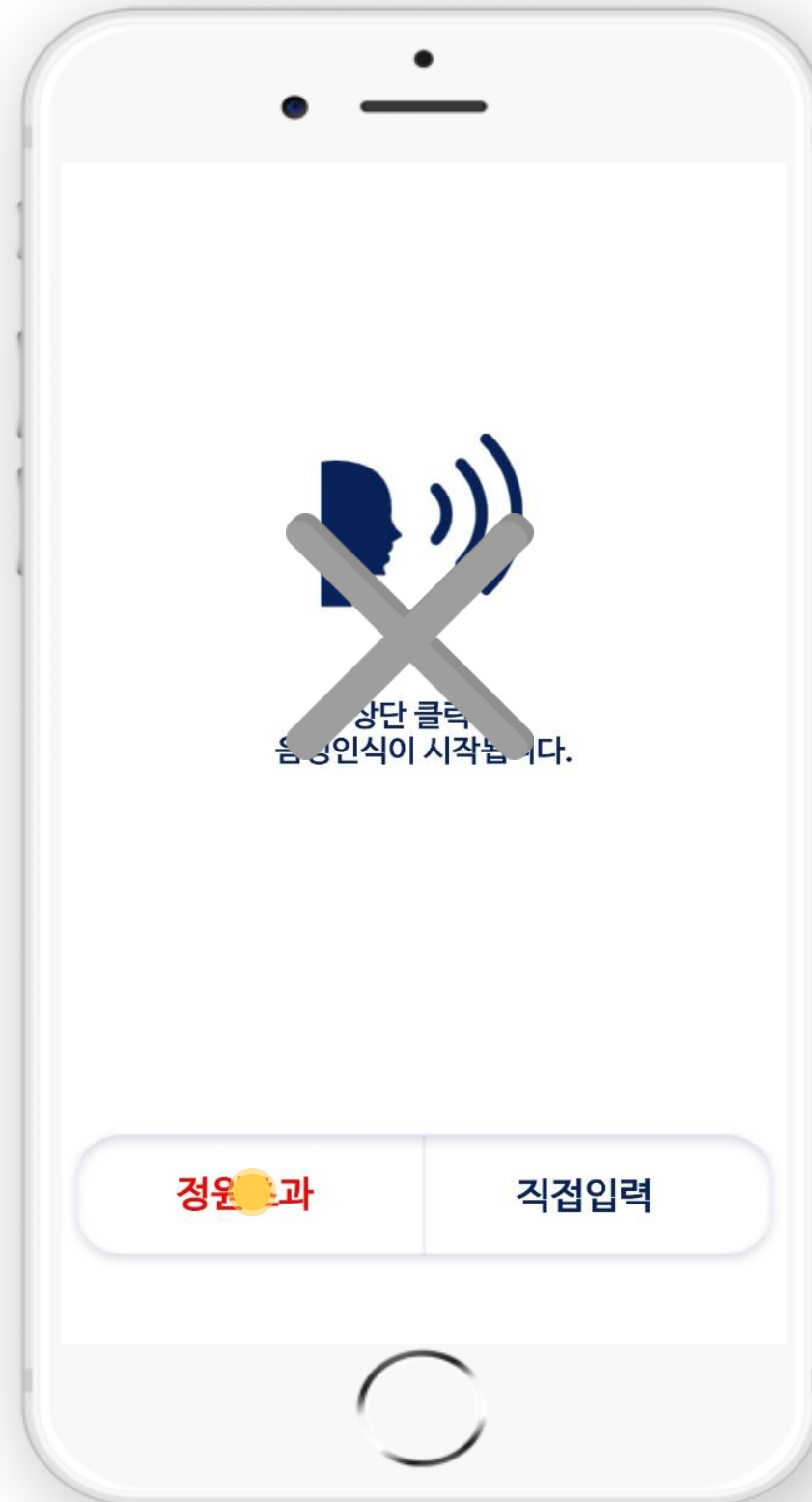
혼잡도

“혼잡상태 입니다. ”



혼잡도

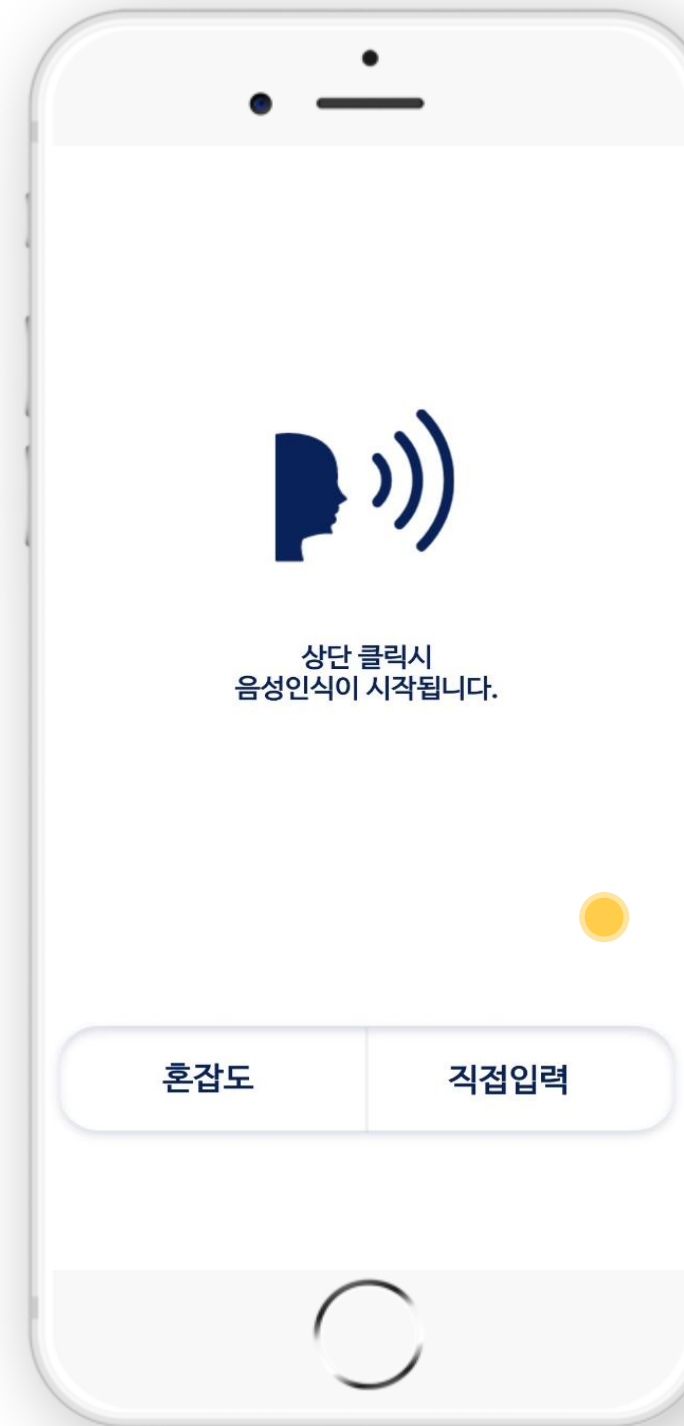
“정원초과 입니다.”



음성입력, 직접입력 기능 사용 불가

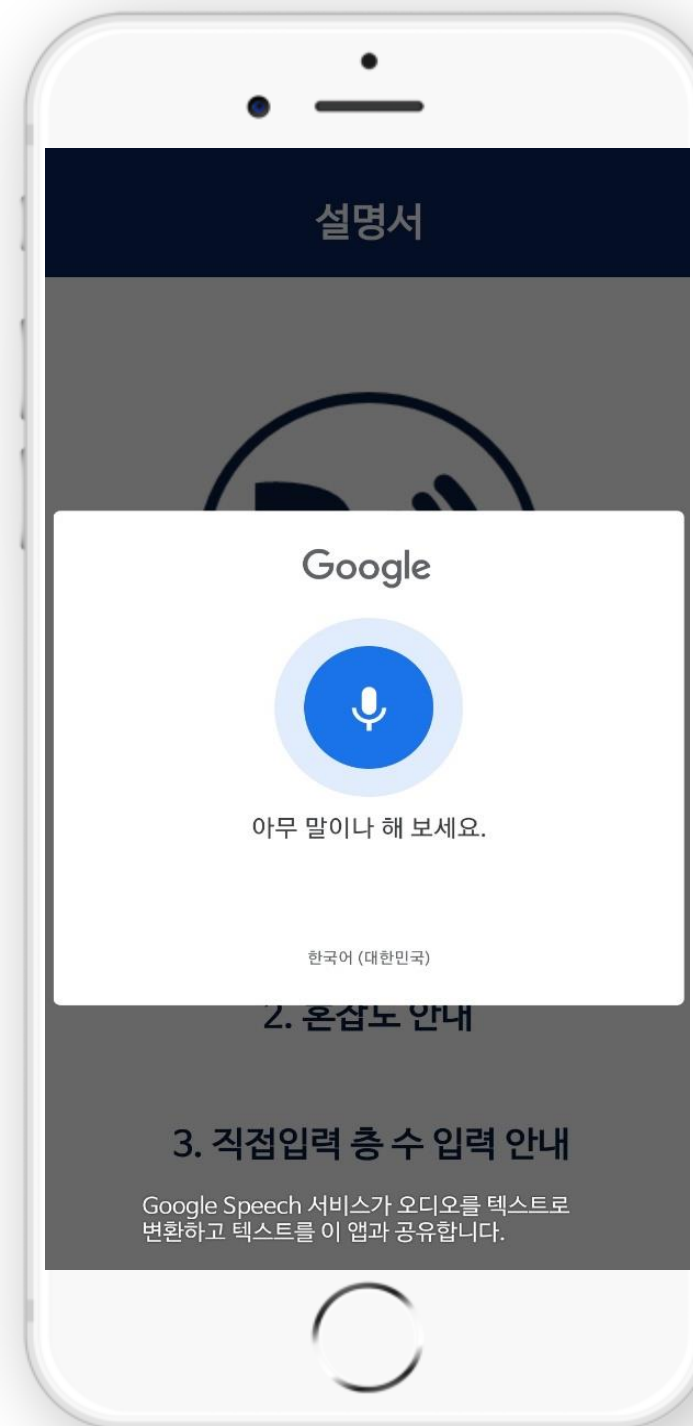
음성안내

화면을 왼쪽으로 넘길 시 음성 안내 듣기 가능

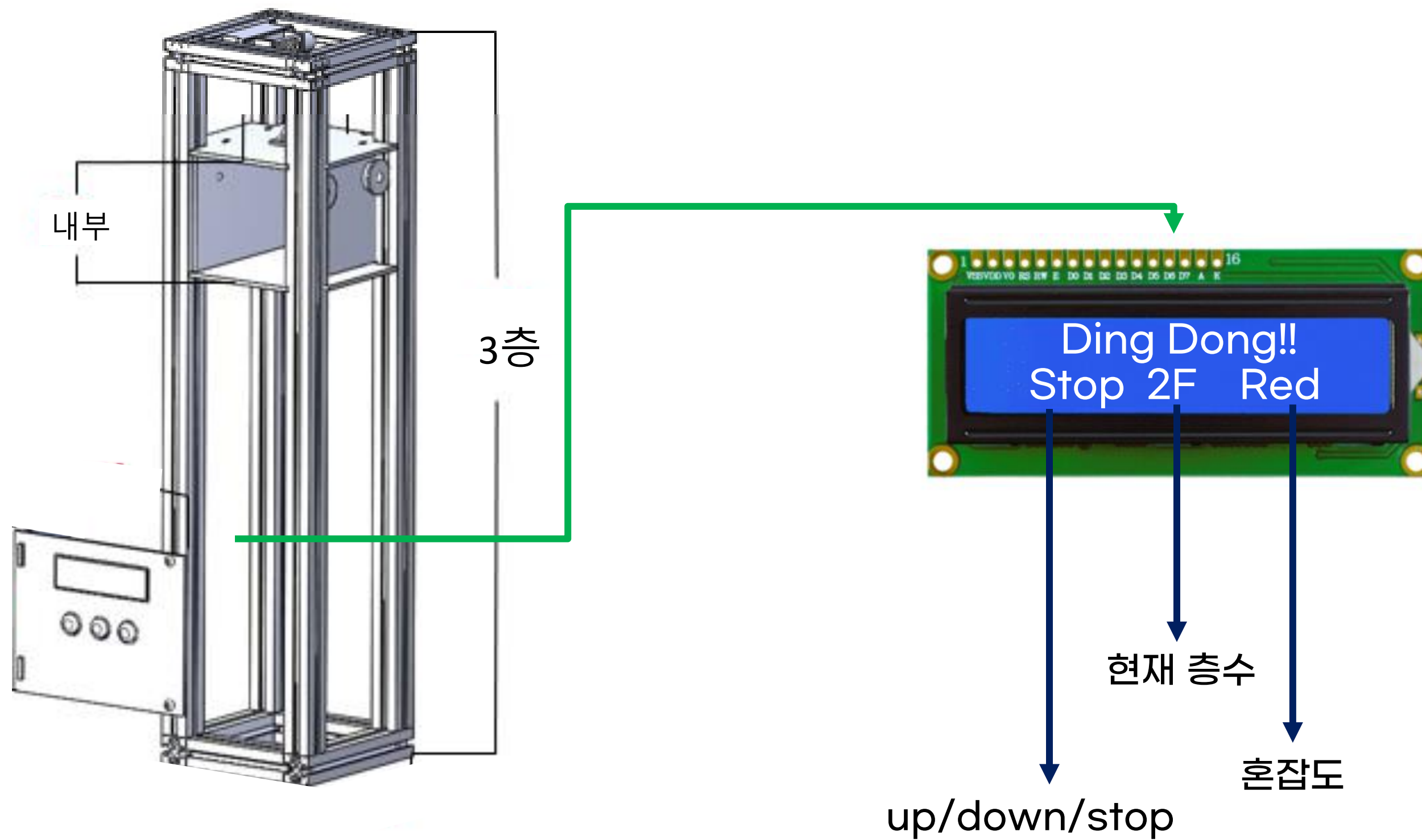


음성안내

“듣고 싶은 설명의 번호를 말해주세요”



엘리베이터 모형



| 개발환경



음성인식기술



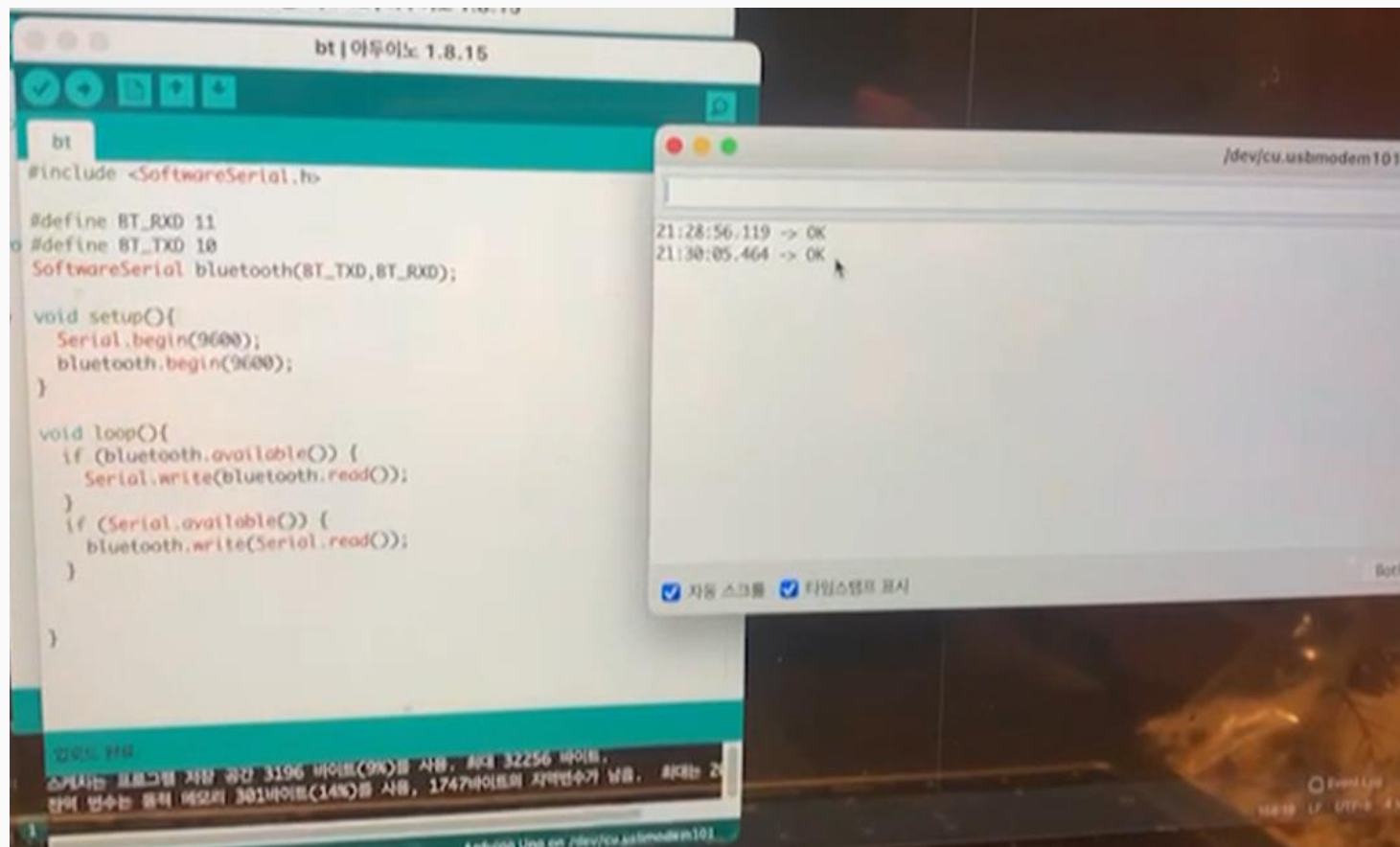
Google Cloud
Speech API

구글 음성인식 API 이용

```
new android.os.Handler().postDelayed(new Runnable() {  
    @Override  
    public void run() {  
        if(isConnected()){  
            Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);  
            intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,  
                RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);  
            startActivityForResult(intent, REQUEST_CODE);  
        }  
        else{  
            Toast.makeText(getApplicationContext(), text: "Plese Connect to Internet", Toast.LENGTH_LONG).show();  
        }  
    }  
} //run  
}, delayMillis: 2700);
```

구글 음성인식 API를 이용하여 앱 사용자의 음성을 입력 받습니다.

블루투스 통신



```
// 블루투스 활성화하기
BluetoothAdapter bluetoothAdapter = BluetoothAdapter.getDefaultAdapter(); // 블루투스 어댑터를 디폴트 어댑터로 설정
if (bluetoothAdapter == null) { // 디바이스가 블루투스를 지원하지 않을 때
    Toast.makeText(getApplicationContext(), text: "Plese Connect to Bluetooth", Toast.LENGTH_LONG).show();
} else { // 디바이스가 블루투스를 지원 할 때
    if (bluetoothAdapter.isEnabled()) { // 블루투스가 활성화 상태 (기기에 블루투스가 켜져있음)
        selectBluetoothDevice(); // 블루투스 디바이스 선택 함수 호출
    } else { // 블루투스가 비 활성화 상태 (기기에 블루투스가 꺼져있음)
        // 블루투스를 활성화 하기 위한 다이얼로그 출력
        Intent intent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        // 선택한 값이 onActivityResult 함수에서 콜백된다.
        startActivityForResult(intent, REQUEST_ENABLE_BT);
    }
}
```

```
// UUID 생성
UUID uuid = java.util.UUID.fromString("00001101-0000-1000-8000-00805f9b34fb");
// Rfcomm 채널을 통해 블루투스 디바이스와 통신하는 소켓 생성
try {
    BluetoothSocket bluetoothSocket = bluetoothDevice.createRfcommSocketToServiceRecord(uuid);
    bluetoothSocket.connect();
    // 데이터 송,수신 스트림을 얻어옵니다.
    OutputStream outputStream = bluetoothSocket.getOutputStream();
    InputStream inputStream = bluetoothSocket.getInputStream();
    // 데이터 수신 함수 호출
    receiveData();
} catch (IOException e) {
    e.printStackTrace();
}
```

블루투스 통신을 통해 앱에 필요한 데이터를 송수신합니다.

Arduino

층 수 입력

```
itda_project
}
} else {
  Serial.println("HX711 not found.");
}
delay(100); // 0.1초 딜레이

blue = bluetooth.read(); // 블루투스 데이터 읽기

//1층,2층,3층
if(blue == '1') btt = 1;
if(blue == '2') btt = 2;
if(blue == '3') btt = 3;

itda_project
{ // 모터 동작
  if(digitalRead(5) == LOW) sen = 1;
  if(digitalRead(6) == LOW) sen = 2;
  if(digitalRead(7) == LOW) sen = 3;

  // 층 버튼 입력값과 센서 값이 같거나, 층 버튼 입력이 없을 경우
  if((btt == sen) || (btt == 0) || (a1 >= 5500)) {
    updown = 0;
    btt = 0;
    // 층 버튼 입력이 있으면서 센서 값이 없을 경우 하강
  } else if((btt != 0) && (sen == 0)) {
    updown = 2;
    // 층 버튼값이 센서 값 보다 큰 경우 상승
  } else if (btt > sen) {
    updown = 1;
    // 층 버튼값이 센서 값 보다 작은 경우 하강
  } else if (btt < sen) {
    updown = 2;
  }

  if (updown == 0) { // 멈춤
    digitalWrite(8, LOW);
    digitalWrite(9, LOW);
  } else if (updown == 1) { // 상승
    digitalWrite(8, LOW);
    digitalWrite(9, HIGH);
  } else if (updown == 2) { // 하강
    digitalWrite(8, HIGH);
    digitalWrite(9, LOW);
  }
}

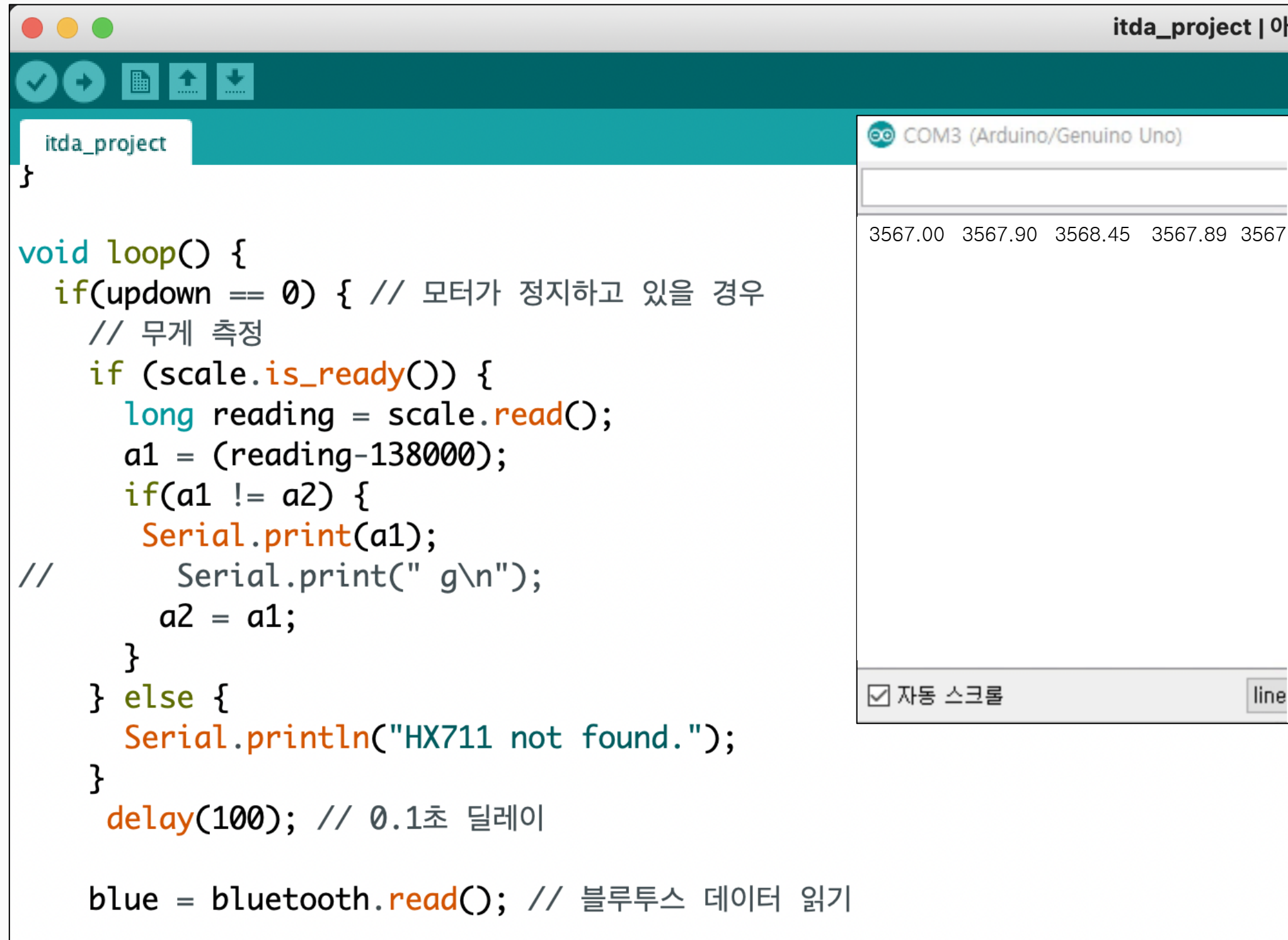
itda_project
// 층 버튼값이 센서 값 보다 작은 경우 하강
} else if (btt < sen) {
  updown = 2;
}

if (updown == 0) { // 멈춤
  digitalWrite(8, LOW);
  digitalWrite(9, LOW);
}
```

어플에서 음성, 직접 입력하여 보낸 층 수 값을 아두이노에서 받아 엘리베이터를 작동시킵니다.

Arduino

혼잡도 안내



```
itda_project
}

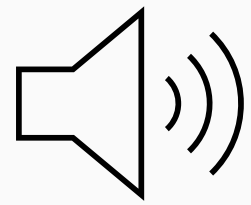
void loop() {
  if(updown == 0) { // 모터가 정지하고 있을 경우
    // 무게 측정
    if (scale.is_ready()) {
      long reading = scale.read();
      a1 = (reading-138000);
      if(a1 != a2) {
        Serial.print(a1);
        //      Serial.print(" g\n");
        a2 = a1;
      }
    } else {
      Serial.println("HX711 not found.");
    }
    delay(100); // 0.1초 딜레이

    blue = bluetooth.read(); // 블루투스 데이터 읽기
```

```
if (a1 < 3500 ) {
  lcd.print(" Green");
  bluetooth.print("green");
  bluetooth.print("\n");
  digitalWrite(A3, LOW);
} else if (a1 < 6000 ) {
  lcd.print("Yellow");
  bluetooth.print("yellow");
  bluetooth.print("\n");
  digitalWrite(A3, LOW);
} else if (a1 < 7500 ) {
  bluetooth.print("red");
  bluetooth.print("\n");
  lcd.print(" Red ");
  digitalWrite(A3, LOW);
} else if(a1>=8500){
  bluetooth.print("danger");
  bluetooth.print("\n");
  lcd.print(" Full ");
  tone(A3, 1000, 100);
}
}
```

무게 센서로 무게 값을 측정한 후 기준 값에 따라 혼잡도를 핑동 어플로 블루투스 통신합니다.

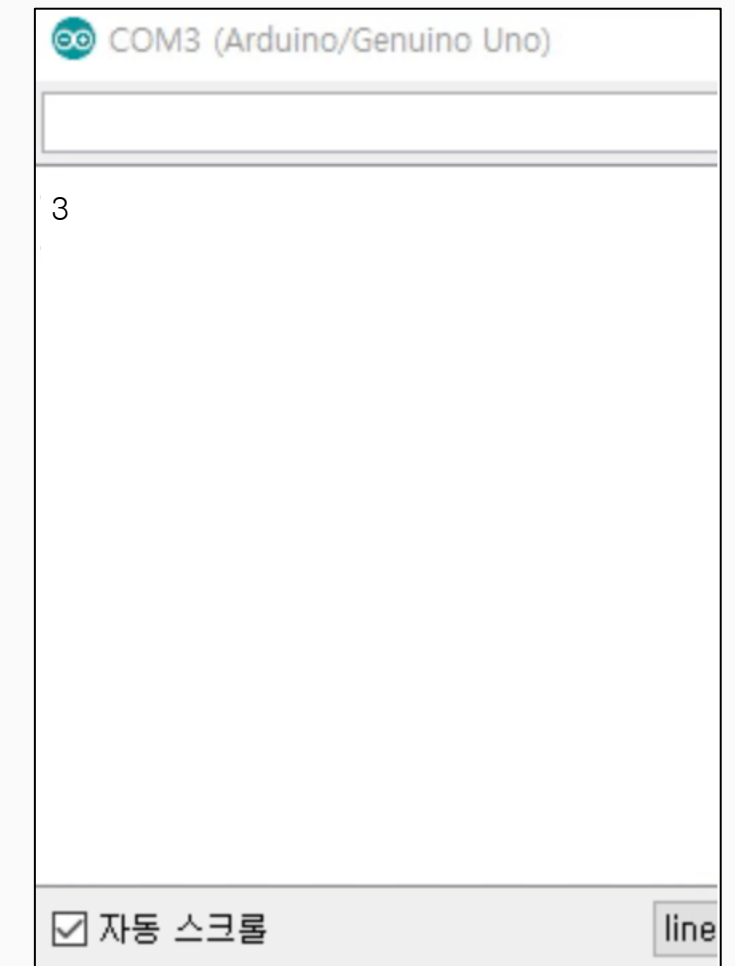
층수 이동 (음성입력 , 직접입력)



3층



```
public void sendData(String text) {  
    // 문자열에 개행문자("\n")를 추가해줍니다.  
    text += "\n";  
    try {  
        // 데이터 송신  
        outputStream.write(text.getBytes());  
        outputStream.close();  
        System.out.println(text);  
    } catch (Exception e) {  
        System.out.println("error");  
        e.printStackTrace();  
    }  
}
```

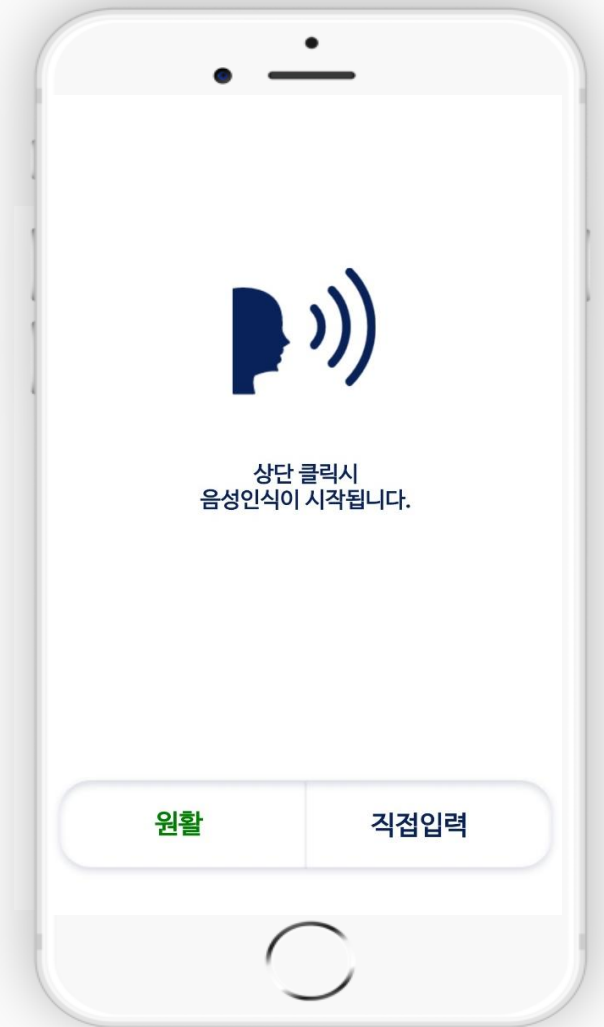


텍스트나 음성으로 층을 입력하면, 아두이노로 값을 보내 엘리베이터가 작동됩니다.

혼잡도



```
workerThread = new Thread(new Runnable()  
{  
    @Override  
    public void run() {  
        while (!Thread.currentThread().isInterrupted()) {  
            try {  
                // 데이터를 수신했는지 확인합니다.  
                int byteAvailable = inputStream.available();  
                // 데이터가 수신 된 경우  
                if (byteAvailable > 0) {  
                    // 입력 스트림에서 바이트 단위로 읽어 옵니다.  
                    byte[] bytes = new byte[byteAvailable];  
                    inputStream.read(bytes);  
                    // 입력 스트림 바이트를 한 바이트씩 읽어 옵니다.  
                    for (int i = 0; i < byteAvailable; i++) {  
                        byte tempByte = bytes[i];  
                        // 개행문자를 기준으로 받음(한줄)  
                        if (tempByte == '\n') {  
                            // readBuffer 배열을 encodedBytes로 복사  
                            byte[] encodedBytes = new byte[readBufferPosition];  
                            System.arraycopy(readBuffer, 0, encodedBytes, 0, encodedBytes.length);  
                            // 인코딩 된 바이트 배열을 문자열로 변환  
                            final String text = new String(encodedBytes, "US-ASCII");  
                            readBufferPosition = 0;  
                        }  
                    }  
                }  
            }  
        }  
    }  
});
```



무게 측정 값을 안드로이드 스튜디오로 송신한 후, 메인 화면에서 내부 혼잡도를 보여줍니다.

DB 연동 (음성설명서)

```
DatabaseHelper DatabaseHelper
DatabaseHelper.java input.java IntroActivity.java MainActivity.java activity_main.xml ttsCompleteActivity.java ArsGuide.java
import ...

public class DatabaseHelper extends SQLiteOpenHelper
{
    static final String TABLE_NAME = "guide";
    public DatabaseHelper(Context context, String name, SQLiteDatabase
        super(context, name, factory, version);
        Log.d(TAG, msg: "DataBaseHelper 생성자 호출");
    }

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        Log.d(TAG, msg: "Table Create");
        String createQuery = "create table if not exists "+TABLE_NAME
            "( ID INTEGER PRIMARY KEY AUTOINCREMENT, " +
            "TITLE TEXT NOT NULL, " +
            "CONTENT TEXT NOT NULL);";
        sqLiteDatabase.execSQL(createQuery);
    }

    public void dbInsert(String tableName, String title, String content) {
        String query = "select id from " + tableName
            + " where title" + "=" + title + "'";
        Cursor cursor = db.rawQuery(query, selectionArgs: null);
        cursor.moveToFirst(); // Cursor를 제일 첫행으로 이동
        if(cursor.getCount() == 0) { // 중복이 없으면 저장하라.
            ContentValues contentValues = new ContentValues();
            contentValues.put("TITLE", title);
            contentValues.put("CONTENT", content);
            // 리턴값: 생성된 데이터의 id
            long id = db.insert(tableName, nullColumnHack: null, contentValues);
            Log.d(TAG, msg: "id : " + id);
        }
    }
} //테이블 삽입
```

안드로이드 스튜디오에서 SQLite를 이용하여 음성 안내를 위한 설명서 DB 생성한 후

DB 연동 (음성설명서)

```
DatabaseHelper.java × input.java × IntroActivity.java × MainActivity.java × activity_main.xml × ttsCompleteActivity.java × ArsGuide.java × ttsActiv
118
119 @Override
120 protected void onActivityResult(int requestCode, int resultCode, Intent data) {
121     int error=0;
122     if (requestCode == REQUEST_CODE && resultCode == RESULT_OK)
123     {
124         text = null;
125         text = data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
126         String text2=text.get(0);
127         if (text2.equals("1번")||text2.equals("일번")||text2.equals("1")||text2.equals("일")){
128             content = getContent( tableName: "guide", id: 1);
129         }
130     }
131     else if (text2.equals("2번")||text2.equals("이번")||text2.equals("2")||text2.equals("이")) {
132         content = getContent( tableName: "guide", id: 2);
133     } else if (text2.equals("3번")||text2.equals("삼번")||text2.equals("3")||text2.equals("삼")) {
134         content = getContent( tableName: "guide", id: 3);
135     }
136     else {
137         content = "입력오류, 1 2 3번중에 다시한번 말씀해주세요.";
138         error=1;//입력오류
139     }
140     Log.d(TAG, msg: "content:"+content);
141     System.out.println(content);
142     tts.speak(content,TextToSpeech.QUEUE_FLUSH, params: null);
143 }
```

번호 값에 따라 해당되는 DB의 Content 값을 가져와 음성으로 출력합니다. (TTS)

엔티티관계도

논리 ERD

guide		
id	title	content
1	음성입력	음성입력 안내입니다. 화면 상단부를 클릭하면 음성인식이 시작됩니다. 도착할 층수를 말해주세요.
2	혼잡도	혼잡도 안내입니다. 화면 좌측 하단을 클릭하면 혼잡도 안내 음성이 나옵니다
3	직접입력	직접입력 안내입니다. 화면 우측 하단을 클릭하면 층수를 직접 입력할 수 있습니다.

물리 ERD

guide
id : integer
title : text
content : text

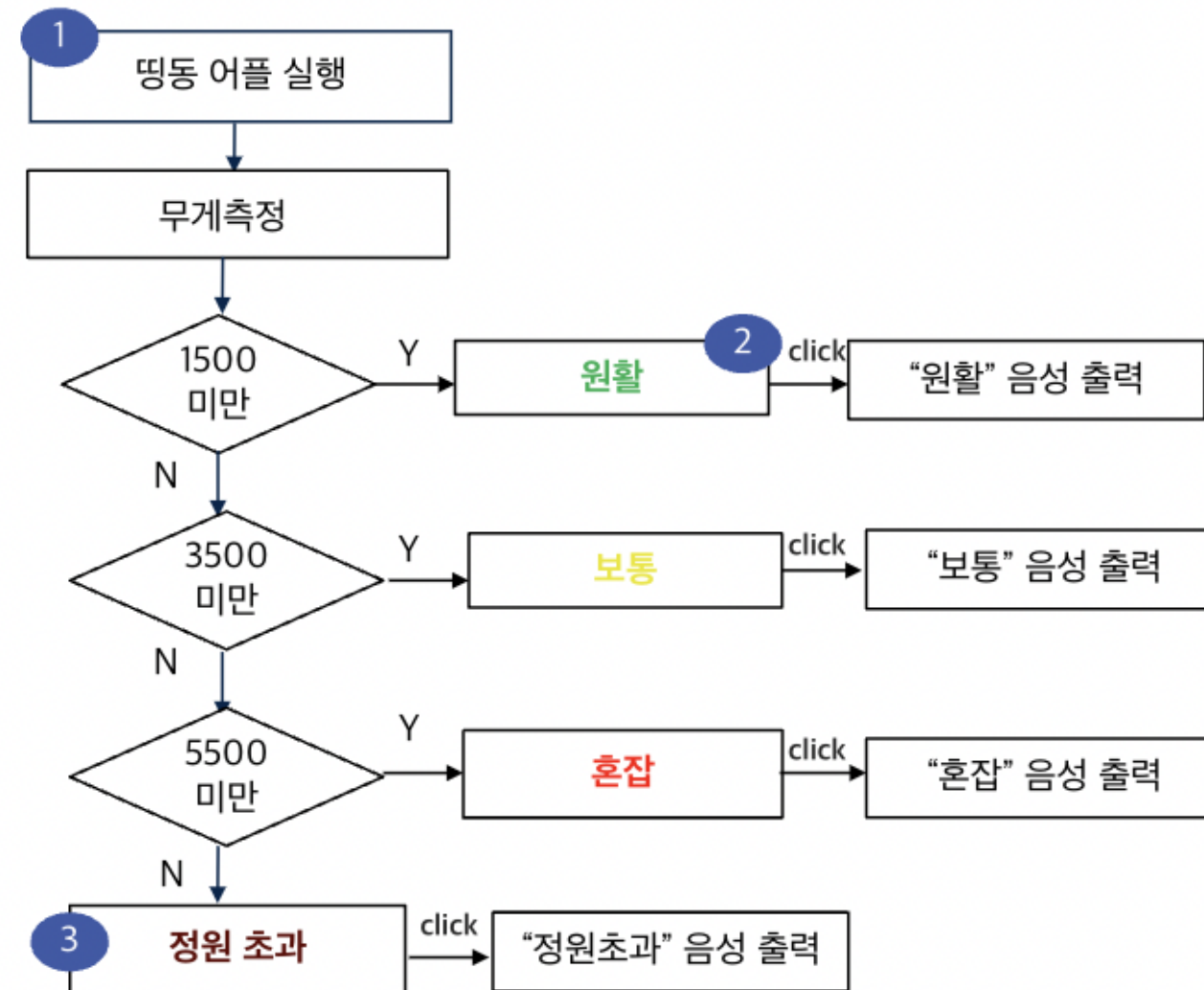
엔티티관계도

엔티티명		가이드				
테이블명		guide				
테이블 설명		가이드 문구 정보를 관리한다.				
번호	컬럼명	속성명	데이터타입	NULL 여부	기본값	KEY
1	id	번호	integer	NOT NULL		PRIMARY KEY, AUTOINCREME NT
2	title	기능	text	NOT NULL		
3	content	설명	text	NOT NULL		

기능 처리도 - 혼잡도

[기능 처리도(능 흐름도)]

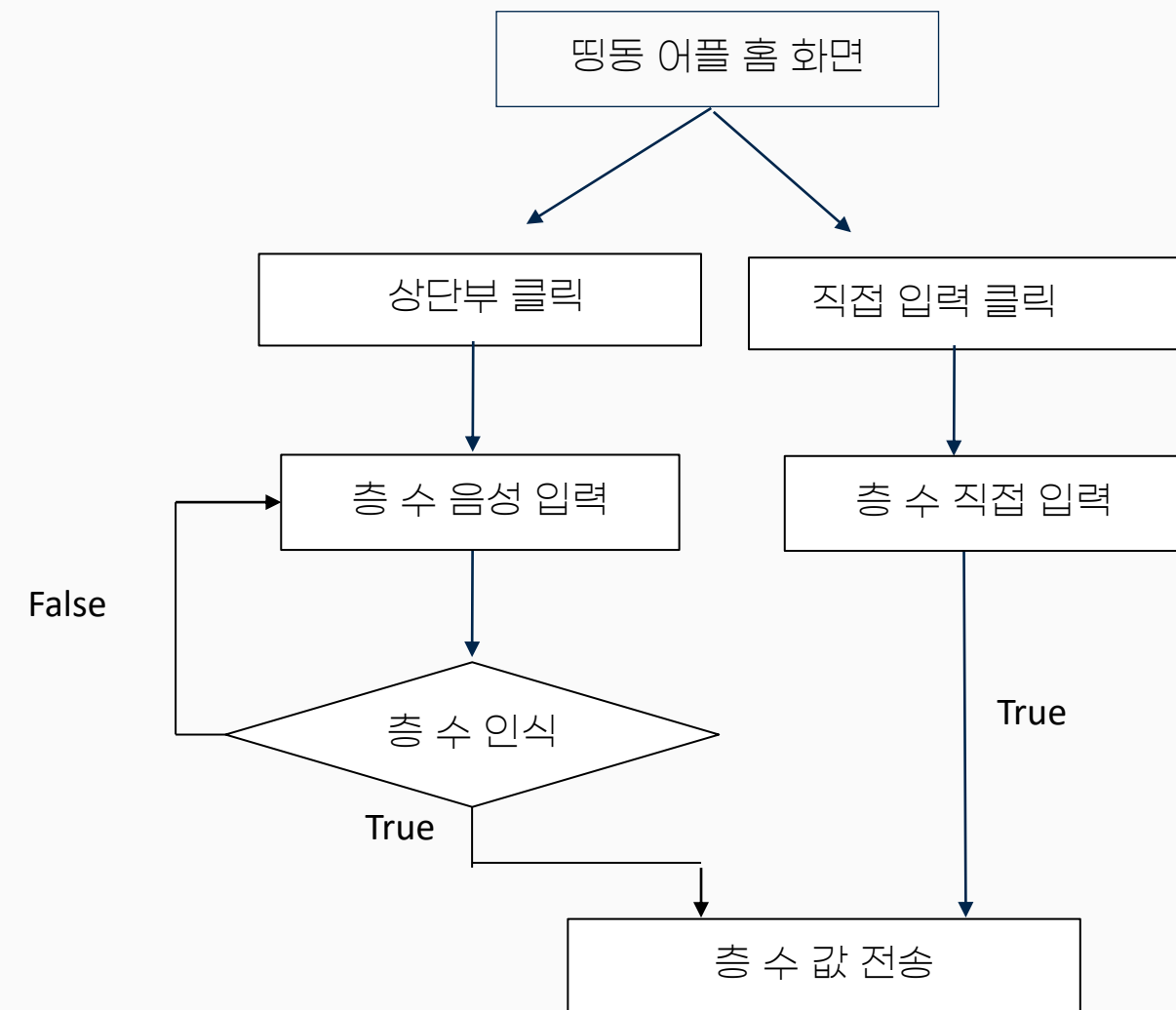
프로그램 ID	A001	프로그램 명	혼잡도 표시	작성일	2021. 08. 27.	Page	1
개요	무게센서(Load Cell)이용하여 무게를 측정하여 데이터를 앱에 전송 후 혼잡, 보통, 원활, 정원초과로 나누어서 분류 후 앱 시작 화면에 혼잡은 빨강, 보통은 노랑, 원활은 초록,정원 초과는 어두운 빨강으로 표시하며 화면 클릭 시 음성으로 안내해준다.					작성자	



- 1 무게센서를 이용하여 엘리베이터 내부 무게 계산하여 기준 값에 따라 무게값 변환 후 땡땡 어플로 전송
- 2 혼잡도 화면 클릭 시 현재 엘리베이터의 혼잡도 현황을 원활, 보통, 혼잡, 정원 초과로 나누어 음성으로 안내
- 3 무게값이 5500이상일 시 정원 초과로 엘리베이터가 작동하지 않으며 피에조 부저 울림

기능 처리도 - 층 수 입력

						실무 산출물 형식	
프로그램 ID	A002	프로그램 명	층 수 입력	작성일	2021 . 08. 27.	Page	2
개요	앱을 통해서 음성이나, 직접 입력으로 층수를 입력 받은 후 층 수 값을 넘겨서 엘리베이터 모형을 제어한다.					작성자	
기능 흐름도							



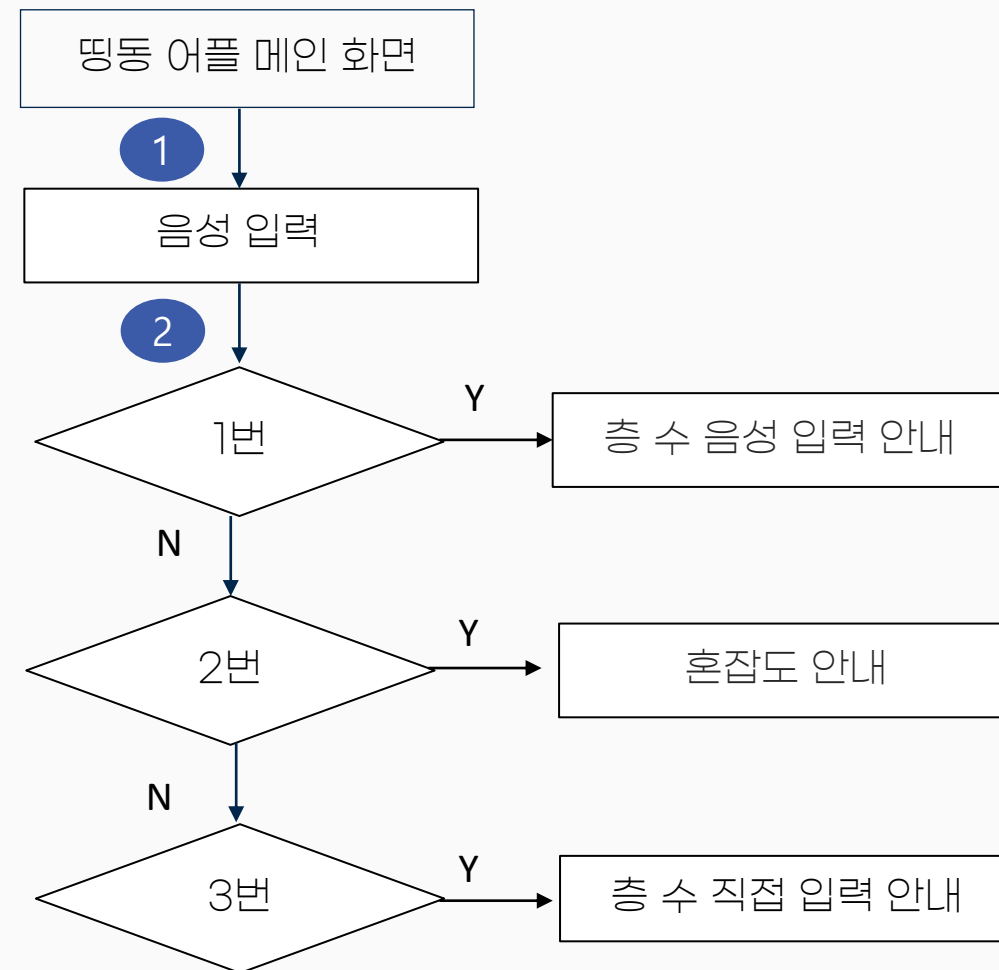
알고리즘 - 음성 설명서

| 기능 처리도(기능 흐름도)

실무 산출물 형식

프로그램 ID	A003	프로그램 명	음성 설명서	작성일	2021. 08. 27.	Page	3
개요	메인 화면에서 오른쪽에서 왼쪽으로 화면을 넘기는 제스처를 취할 시에 음성 설명서 페이지로 넘어간다. 음성 안내 ARS를 들은 후 설명을 듣고자 하는 번호를 말하면(Speech-To-Text) 해당하는 설명을 음성으로 출력(Text-To-Speech)한다.					작성자	

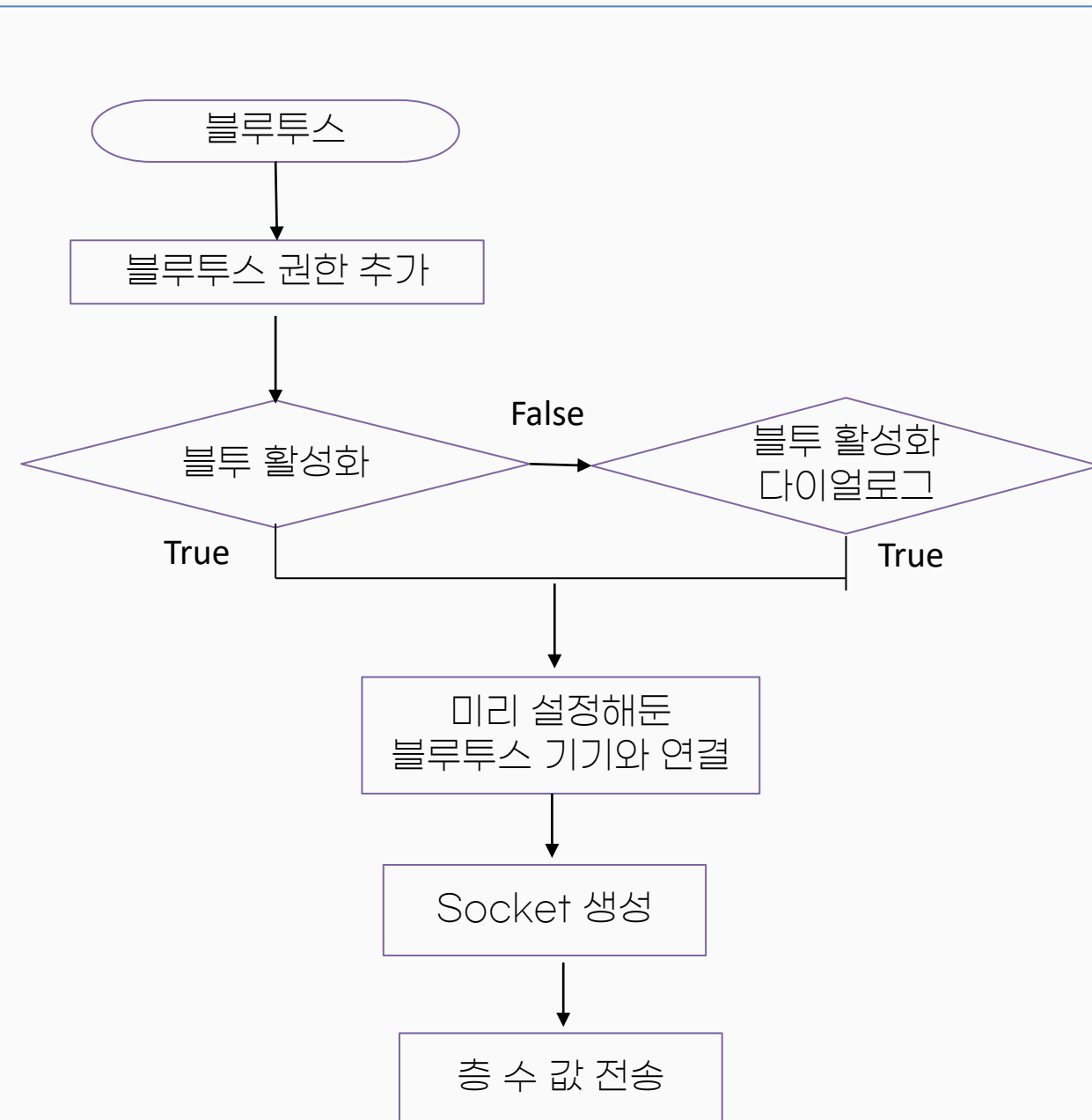
기능 흐름도



- 1 땡동 어플 메인 화면에서 화면을 오른쪽에서 왼쪽으로 넘기는 제스처를 취할 시, 설명서를 듣기 위한 안내 멘트 출력 후 음성인식
- 2 Guide테이블에서 음성 인식된 번호에 해당하는 id 값의 content를 가져와서 음성으로 출력한다.

알고리즘 - 블루투스

| 알고리즘 명세서



① 안드로이드 스튜디오에서 블루투스를 사용하기 위해 manifests에 권한을 추가해준다.

② 기기가 블루투스가 활성화 되어있으면 블루투스를 선택하기 위한 selectBluetoothDevice() 메소드로 이동한다.

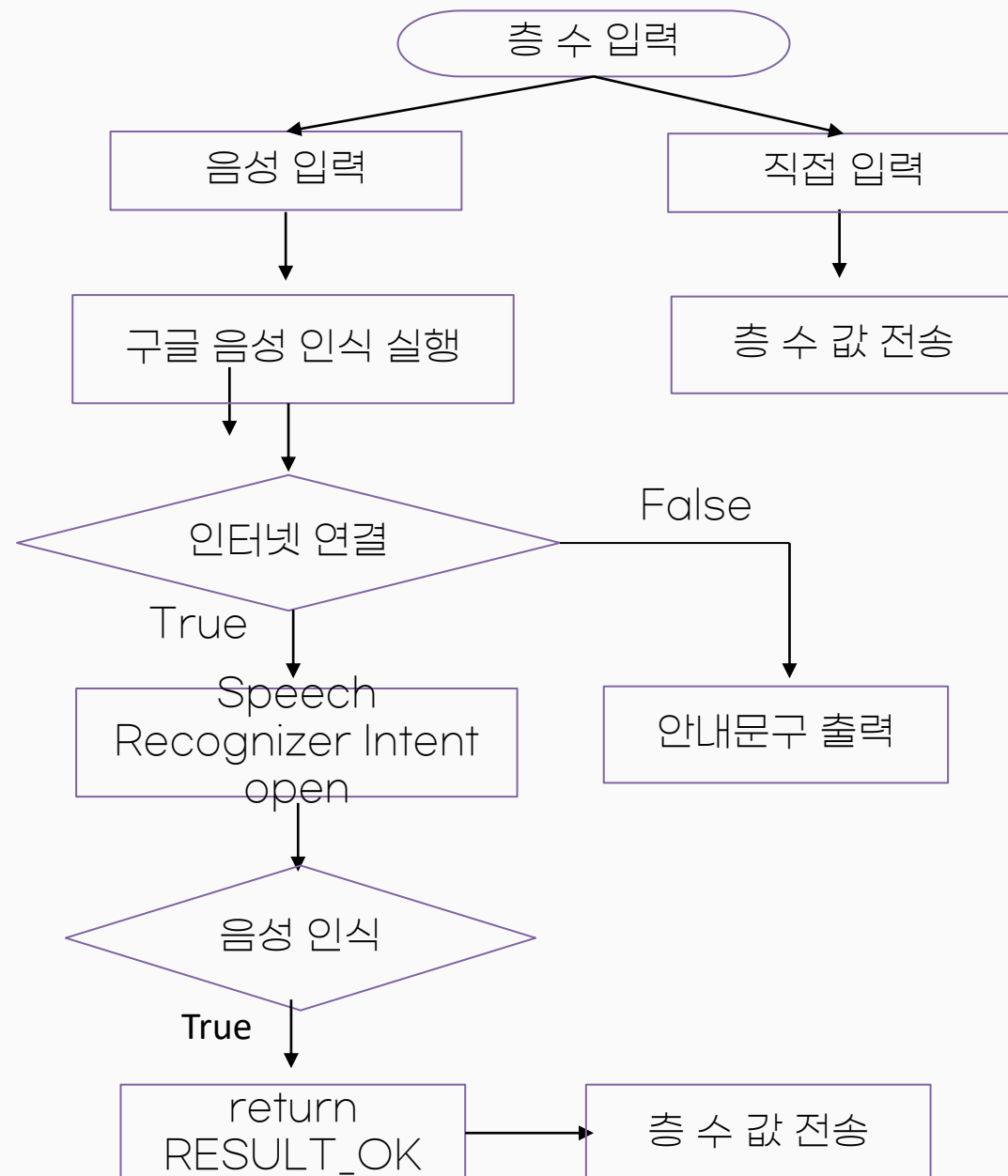
③ 기기가 블루투스가 활성화 되어 있지 않으면 블루투스 활성화 다이얼로그를 출력한다. 이때 사용을 눌렀을 시 RESULT_OK가 반환되면서 위한 selectBluetoothDevice() 메소드로 이동한다.

④ 페어링 되어 있는 블루투스 기기 중에서 미리 설정한 기기를 찾는다.

⑤ 기기를 찾은 후 통신할 수 있는 소켓을 생성하여 송,수신 스트림을 얻어온다.

⑥ 음성이나 직접 입력으로 받은 총 수 값을 sendData() 메소드 내에서 송신 스트림을 통해 아두이노로 전송한다.

알고리즘 - 층 수 입력



① 메인 화면에서 층 수를 음성 입력할 지 텍스트로 직접 입력할 지 선택

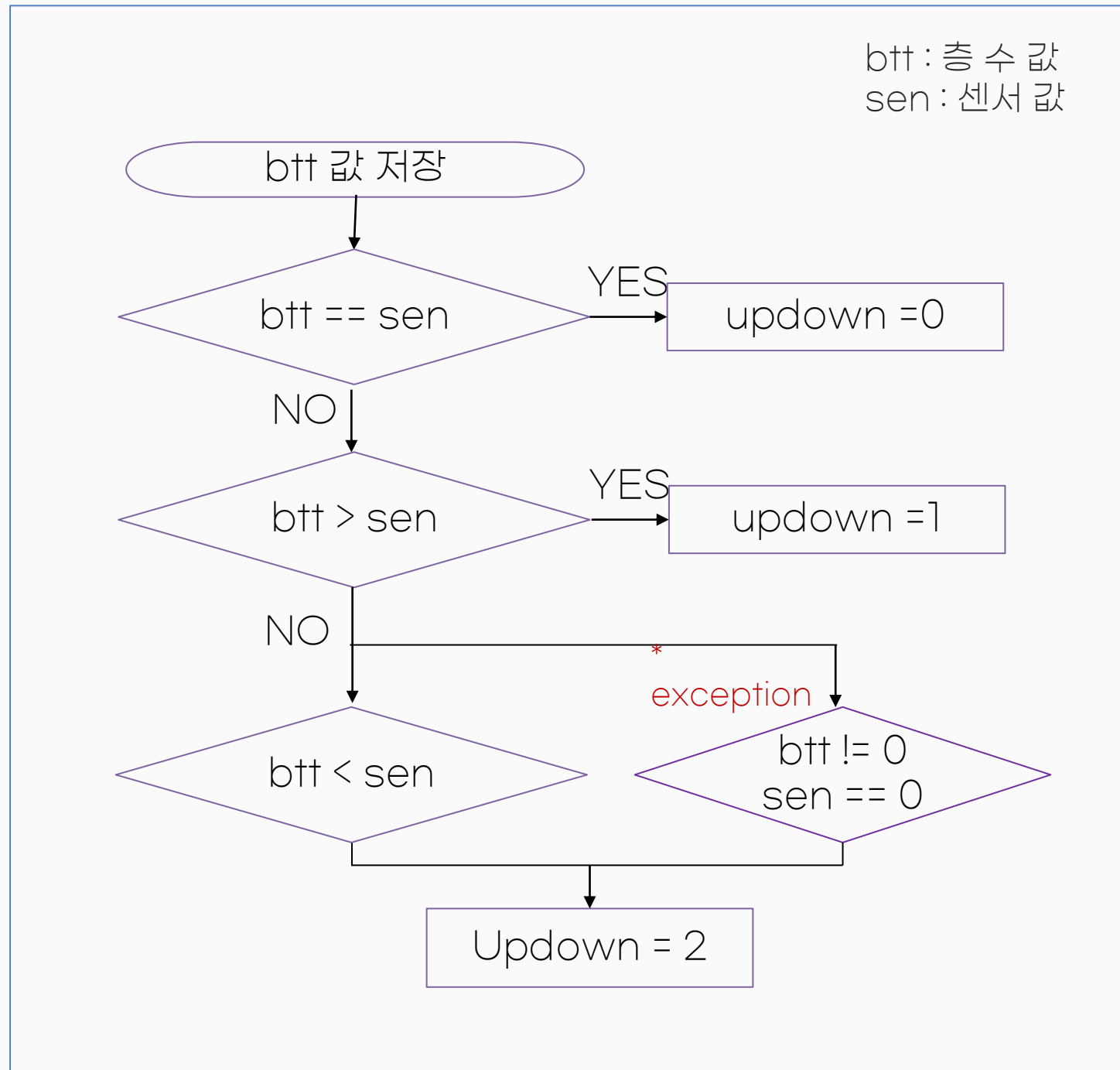
② 음성 입력 화면 클릭 시, 구글 음성 인식을 실행한다.

③ 인터넷이 연결 되어있지 않을 시, 인터넷에 연결해달라는 토스트 메시지를 출력한다.

④ 인터넷에 연결되어 있을 시, Speech Recognizer Intent를 open하여 음성인식을 실행한다.

⑤ 사용자의 음성이 인식 되면 RESULT_OK값을 반환하고 층 수 값을 문자형으로 변환하여 아두이노에 값을 넘긴다.

알고리즘 - 아두이노



① 땡동 앱에서 음성인식 / 직접 입력으로 층 수를 입력한 후 블루투스를 통해 아두이노로 값을 전송한다.

② 문자형으로 넘어온 층 수 값을 저장한다.

③ 아래의 경우, 엘리베이터는 작동하지 않는다.

3-1. 층 수 값과 센서 값이 동일할 경우

3-2. 아무런 입력이 없을 경우

3-2. 무게 값이 5500 이상일 경우

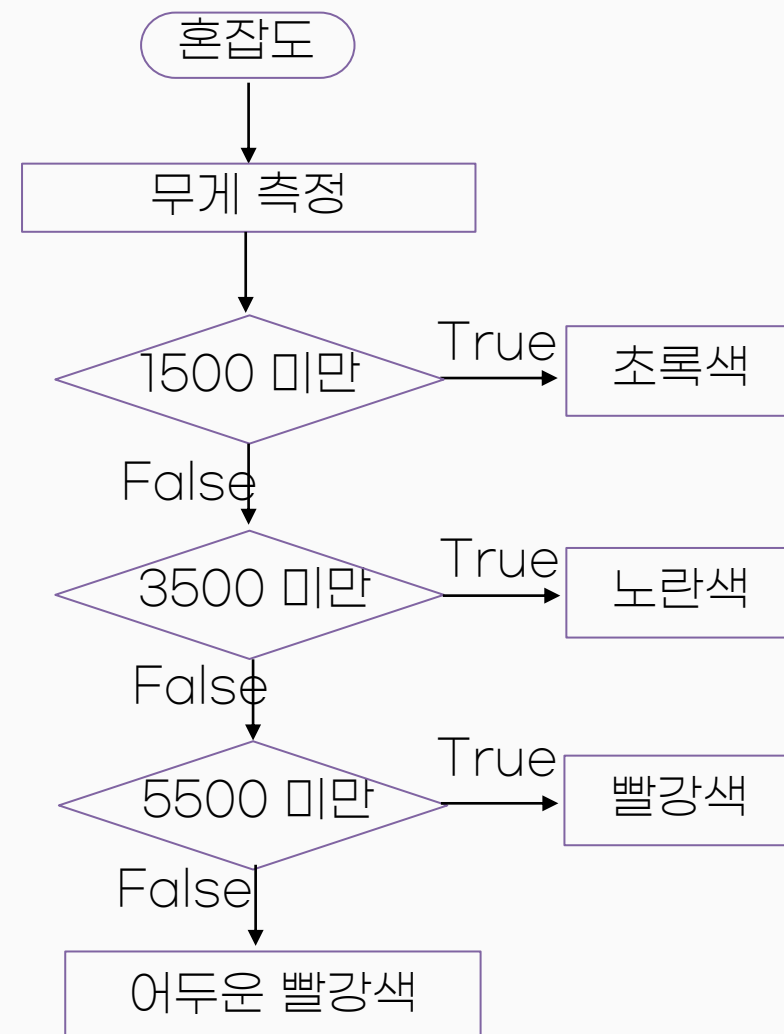
④ 층 수 값이 센서 값보다 클 경우 엘리베이터 상승한다.

⑤ 입력한 층 수 값이 센서 값보다 작을 경우, 엘리베이터 하강한다.

- 센서 값 : 현재 위치하고 있는 엘리베이터의 층수 값
- 층수 값 : 앱에서 블루투스로 넘어온 층 수 값 / 스위치 값
- updown : 엘리베이터를 작동하기 위한 변수
(0 : 멈춤, 1: 상승, 2: 하강)

* 예외 사항을 막기 위해 층 수 값이 입력되었으나 센서 값이 없을 경우 하강하도록 한다. → 센서 값 인식 후 조건 재확인

알고리즘 - 혼잡도



① 무게센서를 이용하여 엘리베이터 내부 무게 계산하여 기준 값에 따라 무게값 변환 후 땡땡 어플로 전송

② 1500 미만 시 혼잡도는 원활, 메인 화면에 “**초록색**” 을 띄운다.

③ 3500 미만 시 혼잡도는 보통, 메인 화면에 “**노란색**” 을 띄운다.

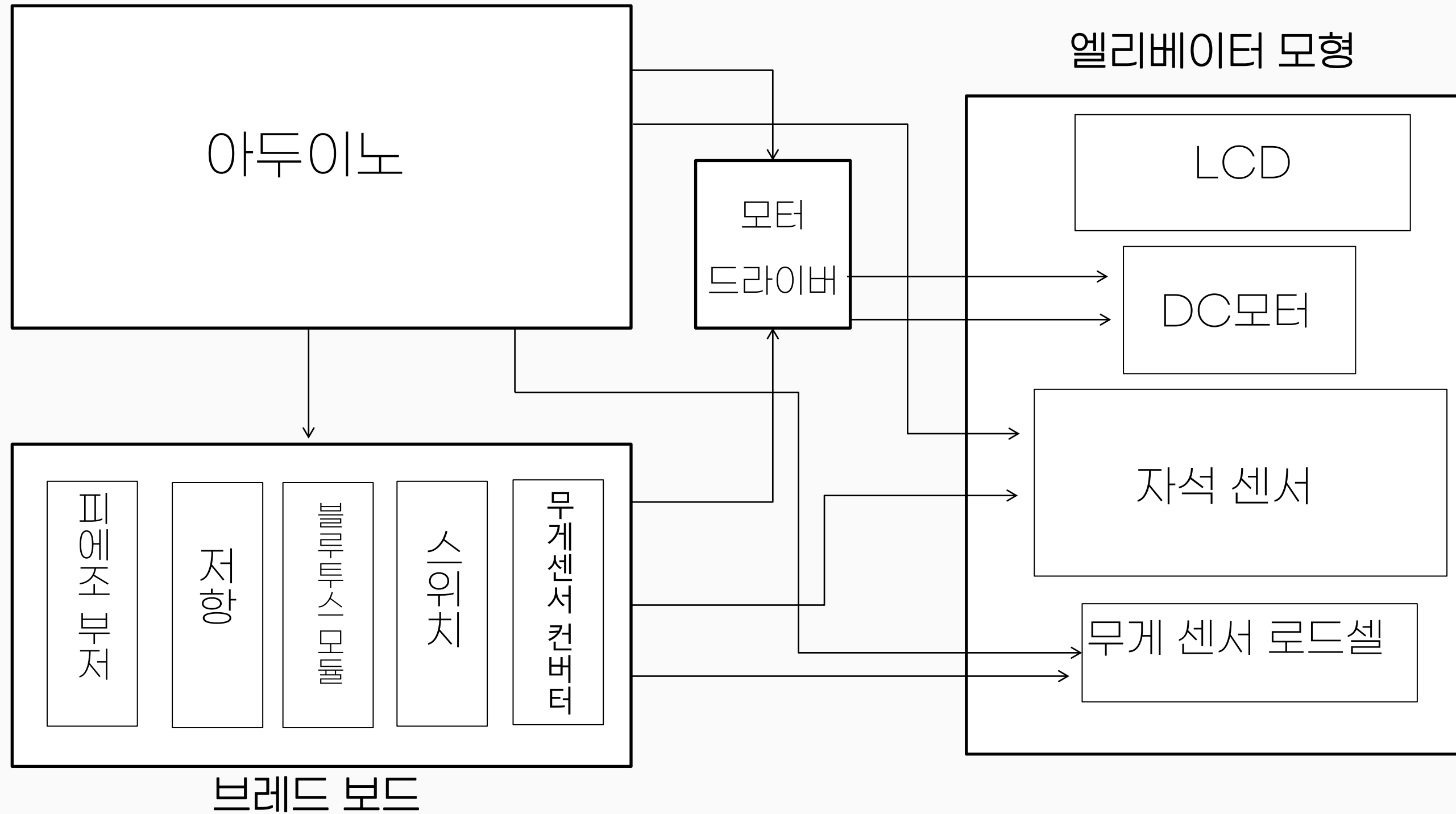
④ 5500 미만 시 혼잡도는 혼잡, 메인 화면에 “**빨간색**” 을 띄운다.

⑤ 5500 이상 시 혼잡도는 정원 초과, 메인 화면에 “**어두운 빨강색**” 을 띄운다.

* 실제 엘리베이터를 대체한 모형이기 때문에 무게를 최소로 측정하여 진행한다.

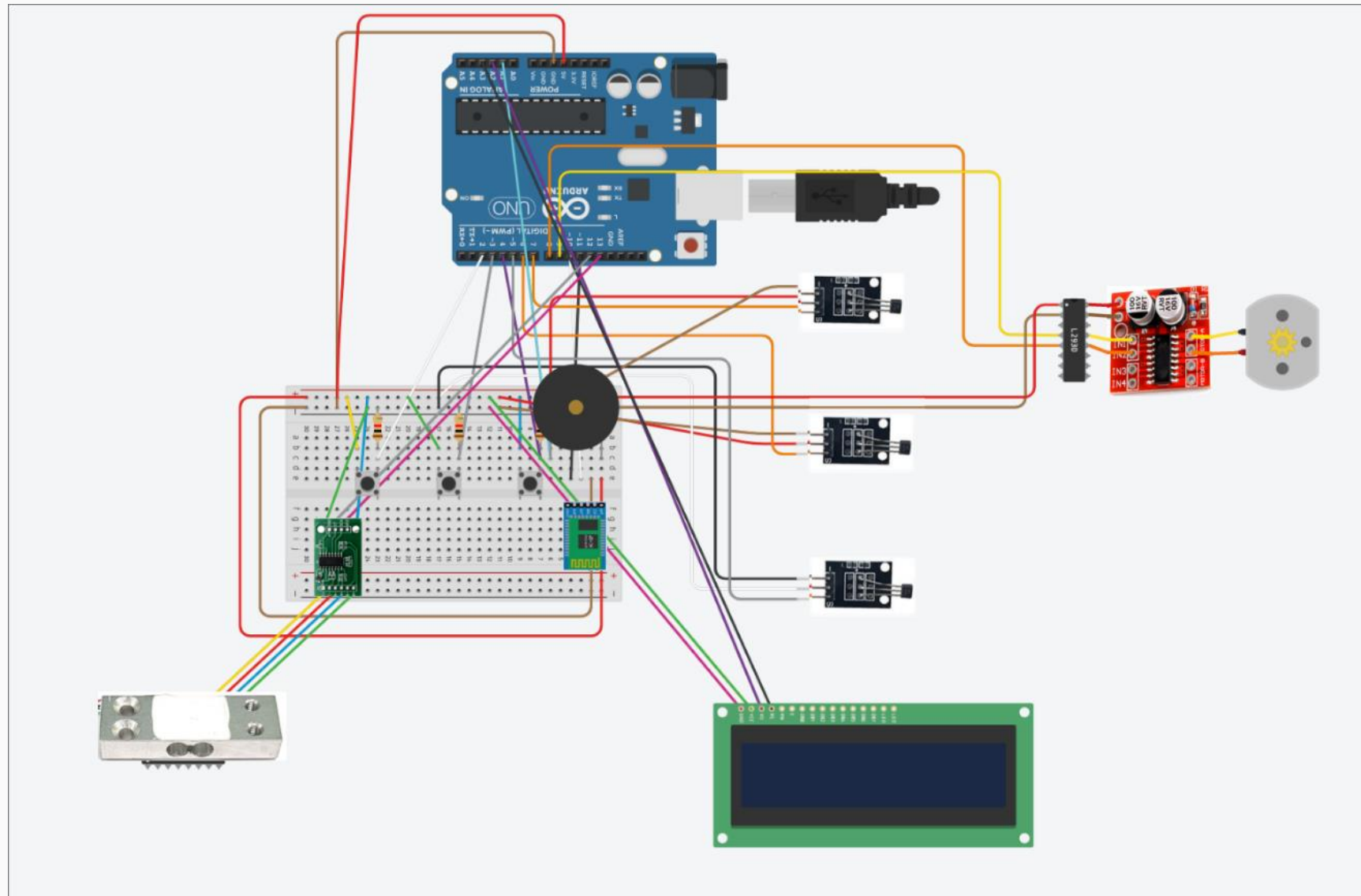
하드웨어 설계도

| 하드웨어/센서 구성도



하드웨어 센서 구성도

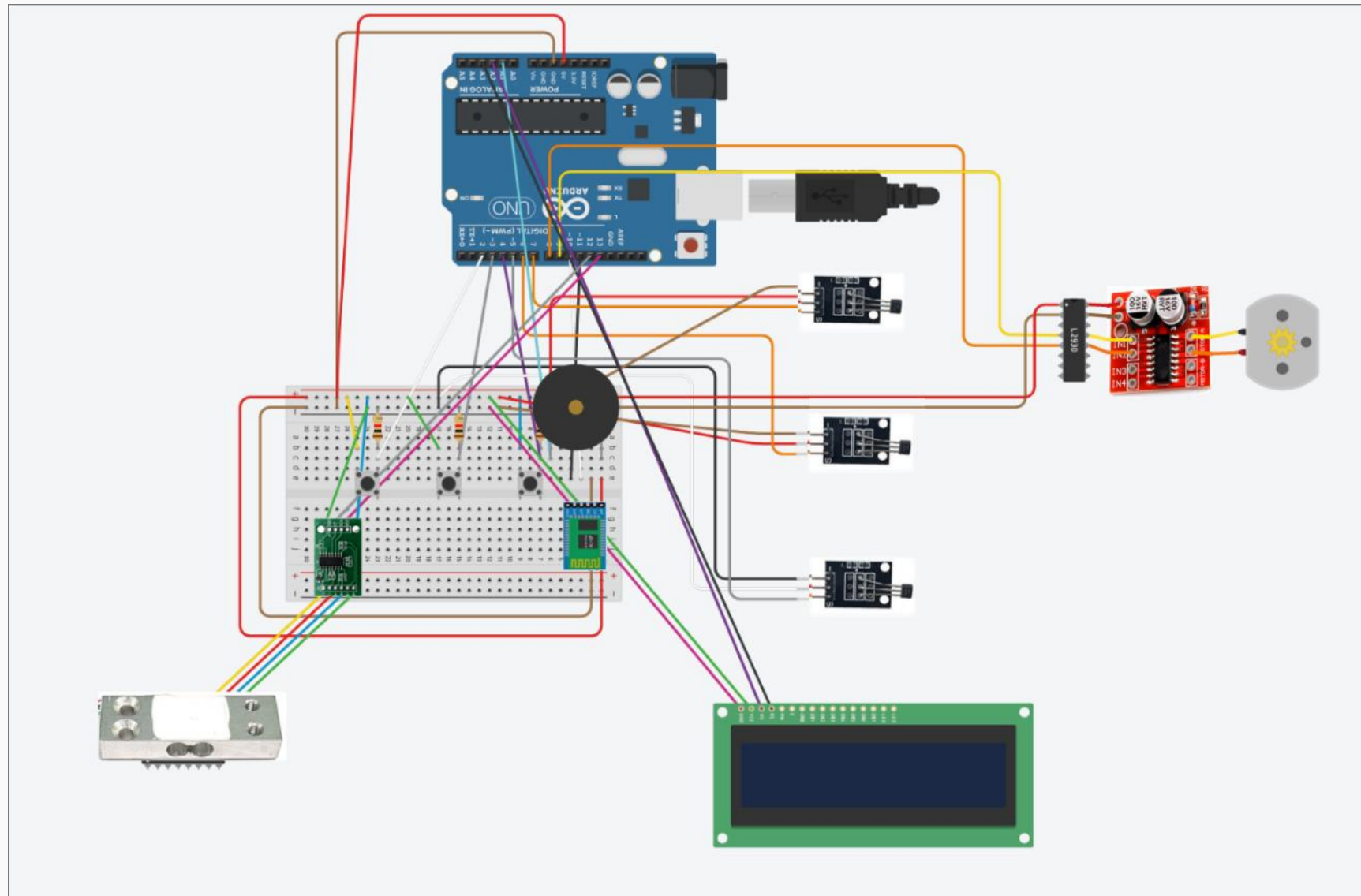
| 하드웨어/센서 구성도



센서 종류	연결 핀	설명
HC-06	RXD	아두이노 11번 핀에 연결
	TXD	아두이노 10번 핀에 연결
	GND	아두이노 GND에 연결
	VCC	아두이노 5v에 연결
H- bridge motor driver	IN1	아두이노의 9번 핀에 연결
	IN2	아두이노의 8번 핀에 연결
	motor-A	DC모터와 연결
DC모터	단자1	모터 드라이버의 motor-A out 1와 연결
	단자2	모터 드라이버의 motor-A out 2와 연결

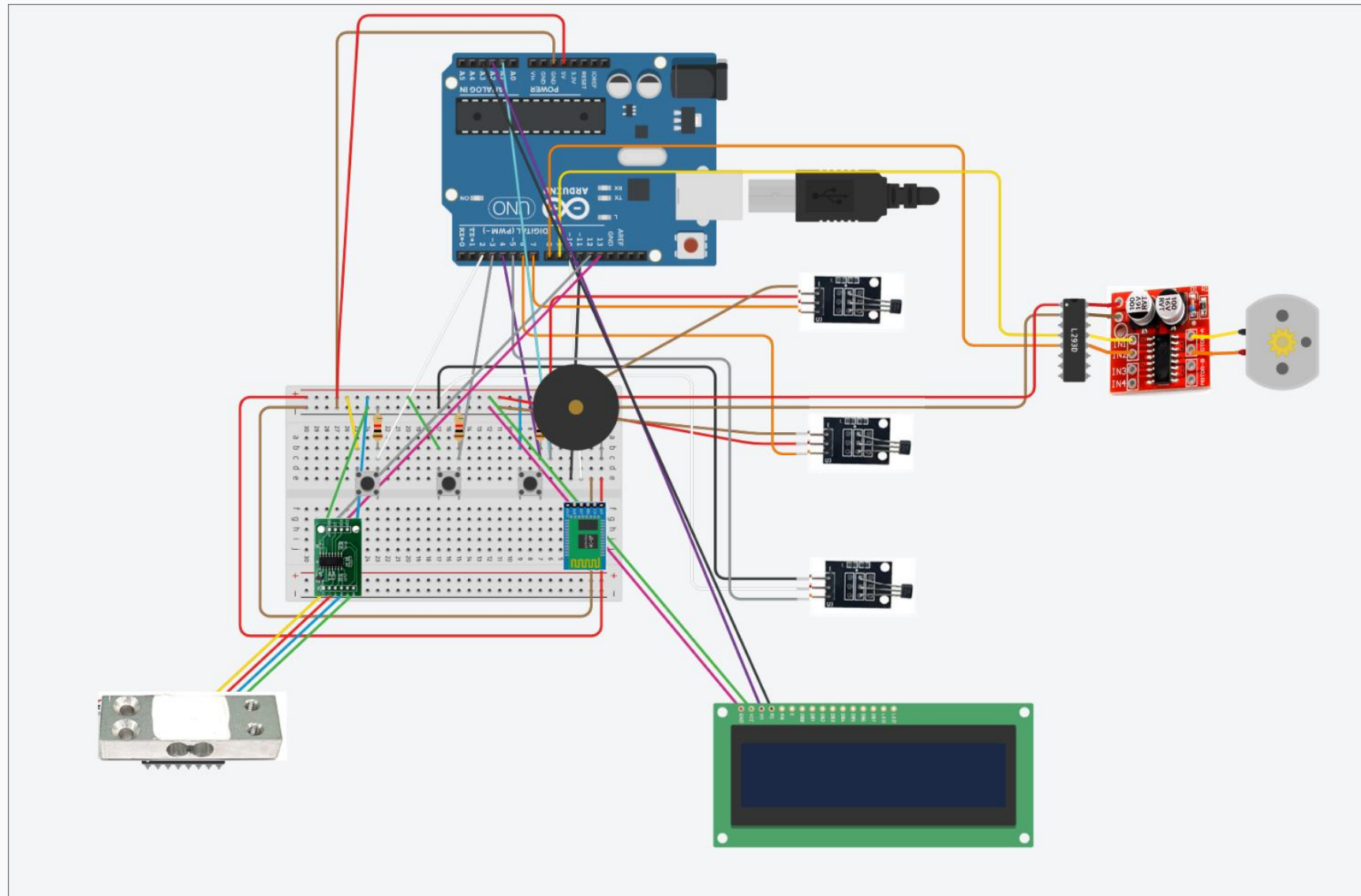
하드웨어 센서 구성도

| 하드웨어/센서 구성도

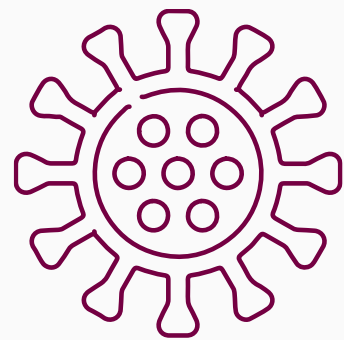


센서 종류	연결 핀	설명
무게센서 컨버터	DT	아두이노 12에 연결
	SCK	아두이노 13번에 연결
	GND	아두이노 GND에 연결
	VCC	아두이노 5v에 연결
무게센서 로드셀	Black	컨버터 E+에 연결
	Red	컨버터 E-에 연결
	White	컨버터 A+에 연결
	Green	컨버터 A-에 연결

하드웨어 센서 구성도



센서 종류	연결 핀	설명
LCD	SCL	아두이노 A2에 연결
	SDA	아두이노 A3에 연결
	GND	아두이노 GND에 연결
	VCC	아두이노 5v에 연결
피에조 부저	양극	아두이노 A1에 연결



COVID-19 방역

엘리베이터 버튼 직접 접촉 최소화
감염 불안요소 감소와 동시에 방역 효과



복지 분야

장애인 복지 센터나 요양원 병원 등 다양한
복지시설에서 노약자, 장애인분들에게 유용하게 사용

기대효과



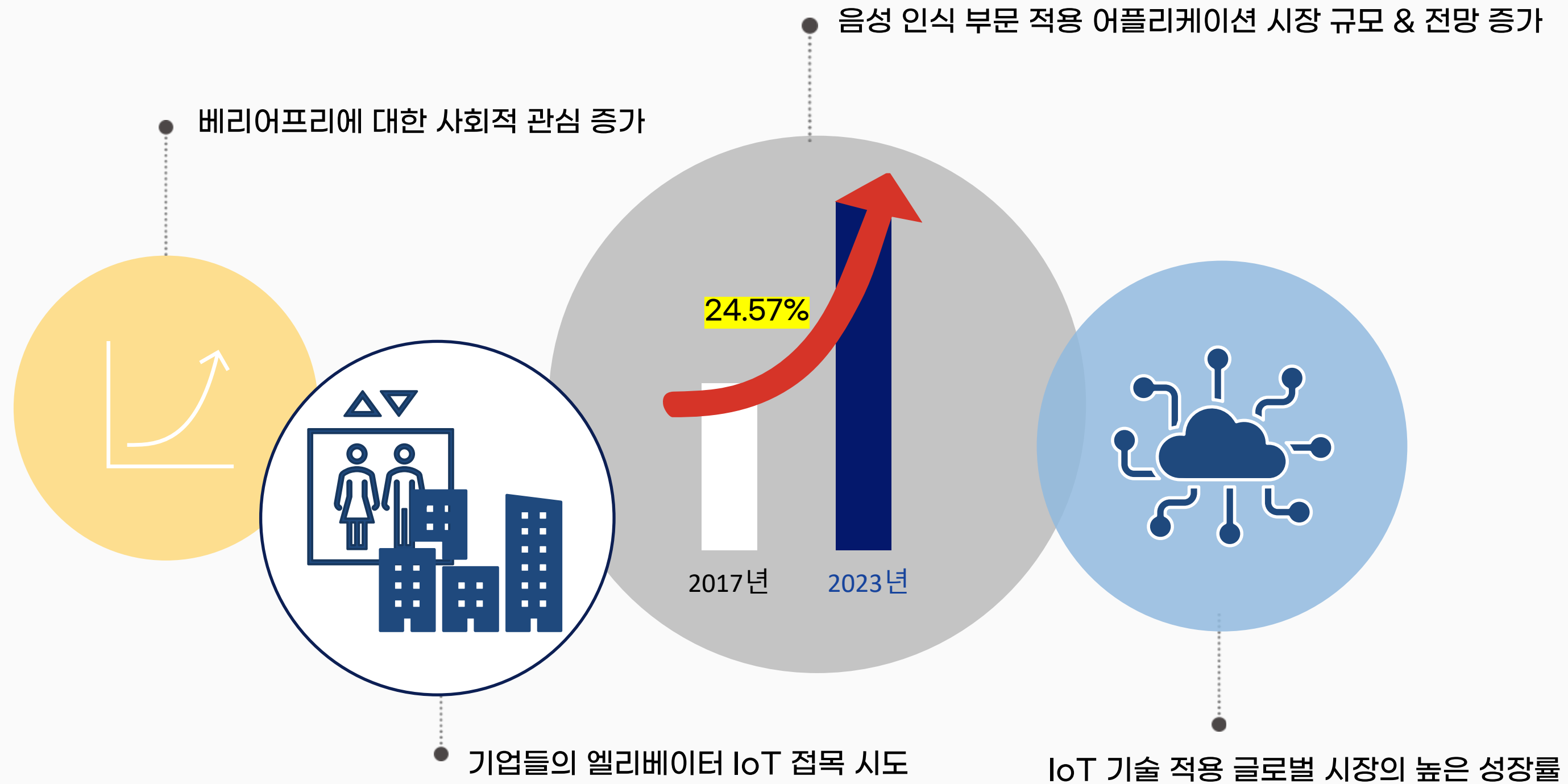
- 향균 필름으로 인해 점자 버튼 인식하기 어려운 경우
- 휠체어를 타서 층 버튼을 누르기 힘든 경우
- 눈이 보이지 않아 승하차시 충돌 위험성이 높은 경우



- 짐이 너무 많아서 버튼을 누르기 힘든 경우
- 엘리베이터 내부에 사람이 많아 버튼을 누르기 힘든 경우
- 높이 있는 층 버튼을 누르기 힘든 경우

#스마트한 #편리한

시장성



핑동 포인트

01 경제성

- ▼ 블루투스 장치와 스마트폰 어플을 이용하여 기존 음성인식 엘리베이터 보다 경제적

02 접근성

- ▼ 이미 블루투스가 내장된 엘리베이터가 있어 확대성이 좋으며
스마트폰을 소지한 모든 사용자가 대상이 될 수 있음

03 베리어프리

- ▼ 누구나 불편함 없이 사용할 수 있는 앱의 목적에 맞게 시각장애인도 쉽게 앱 작동 가능
제스처를 통한 음성 설명서와 단순한 메뉴 구성 , 앱 사용법 음성 안내

딩동플랜

희망 협업체 | 추가 발전 기술

현대엘리베이터

안면, 모션 인식을 통한 엘리베이터 층수 지정(저장)후 자동 층수 입력

블루앤*

스마트폰 블루투스 센서를 통한 출입구 및 엘리베이터 자동 개방

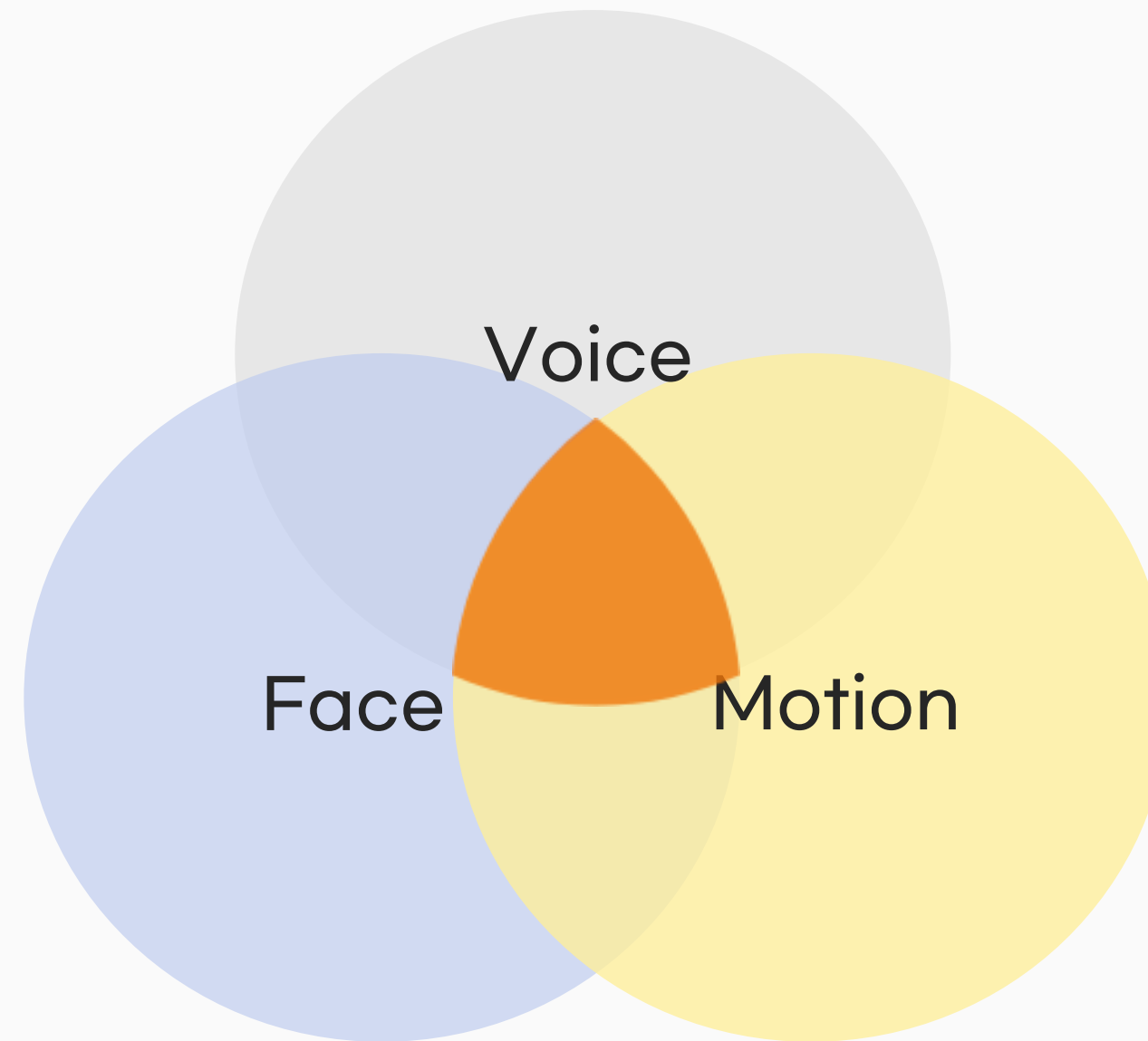
코맥스* & 카카오

AI 스피커를 통해 집 안에서 음성으로 엘리베이터 호출 (“엘리베이터 호출해줘~”)

* 블루앤 : 스마트폰, 블루투스 하이패스 출입 솔루션 업체

* 코맥스 : 주거 공간에서의 네트워크 음성 제어 기술과 인공지능 플랫폼 개발 기업

최종목표



All-in-One 엘리베이터 앱 “땡땡”



땡땡과 함께
언택트 시대를 **스마트** 하게