B E E
: Be your Eyes and Ears

팀장 허 훈
팀원 고용규 고도현 김서연 송무경 이윤주

Contents

1.
프로젝트 추진 개요

BEE

프로젝트 추진 개요

배경

시청각중복장애인 및
시각/청각장애인들의 의사소통 불편

값비싼 의사소통 보조기기 가격

보다 효율적인 의사소통 환경
구축 필요

스마트 디바이스 및 프로그램과
연결을 통한 기능적 확장 요구

Main issues

1. 점자 정보 입출력을 위한
BEE Device 하드웨어 구성

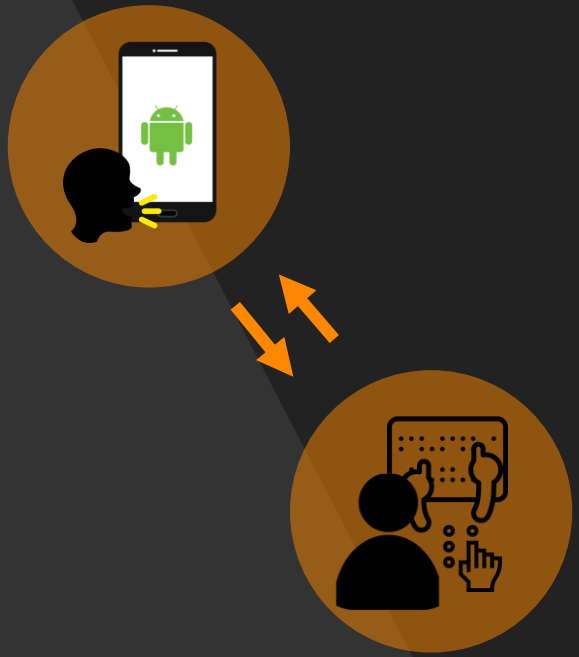
2. 음성 정보 입출력을 위한
BEE Mobile Application
작동과 UI 구현

3. 점자 정보 변환을 위한
API와 로직 구성

목적

시청각장애인의 원활한
의사소통 환경 구축을 통한
삶의 질 개선

프로젝트 목표



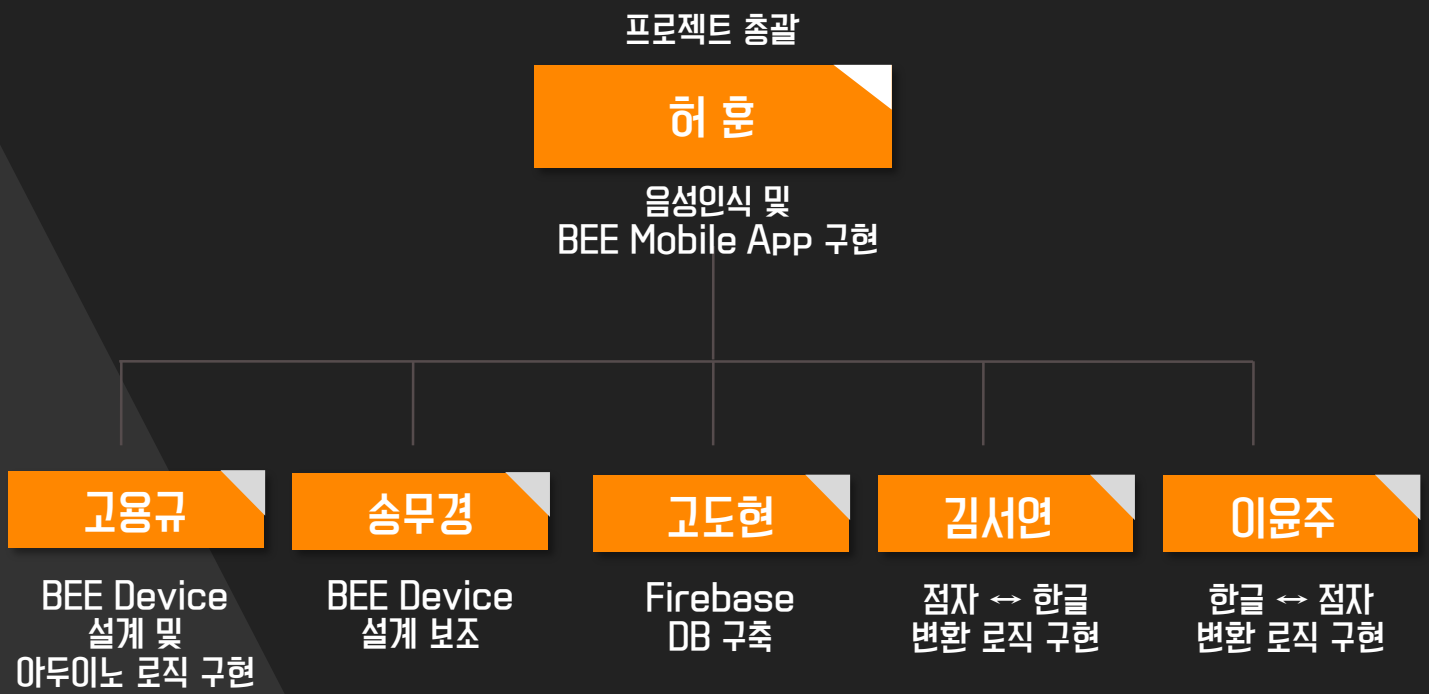
BEE Device 와 BEE Mobile Application을
사용한 의사소통 보조기구 제작

장애인과 비장애인 구분 없이 이용 가능한
범용적 통신 시스템 구축

유사 제품 · 서비스 비교

구분	제품명	장점	단점
Device	Selvas Healthcare 한소네 시리즈	· 점자기기 산업에 대한 전문성 · 웹서핑, 신문 기사 읽기 등 다양한 고급 기술 확보	· 높은 가격과 수리비 · 시청각 중복장애인을 위한 제품 부족
Application	점자 스캔 (한글판) [iOS]	· 점자를 카메라로 촬영, 화상 인식을 통해 한글로 변환	· 한글 → 점자 변환 기능 부재 · 주된 사용자가 장애인이 아닌 일반인들을 위한 어플 · 2$ 비용 발생
	㈜도서출판 점자 점자사전 앱 [Android]	· 단어를 검색하면 점자로 변환되어 사전형식으로 나타남 · 무료	· 개발자가 버전 업데이트를 통해 어플리케이션 내에 단어를 직접 추가해야 함

프로젝트 수행 조직


프로젝트 총괄
허 훈
음성인식 및
BEE Mobile App 구현
고용규
BEE Device
설계 및
아두이노 로직 구현
송무경
BEE Device
설계 보조
고도현
Firebase
DB 구축
김서연
점자 ↔ 한글
변환 로직 구현
이윤주
한글 ↔ 점자
변환 로직 구현

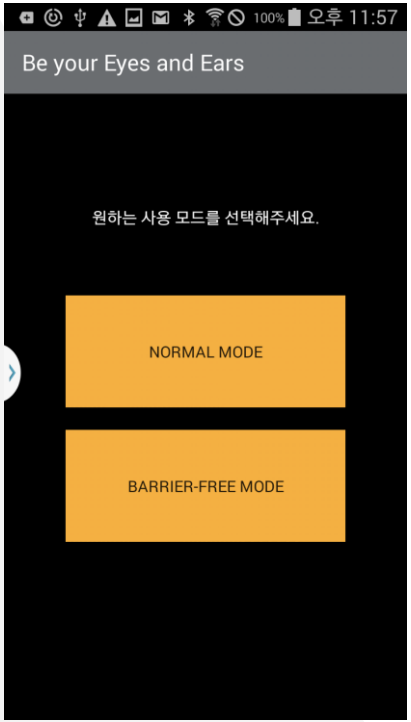
Time schedule

주차 / 내용	4 월				5 월				6 월		
	1	2	3	4	1	2	3	4	1	2	3
아이디어 회의 및 기술 동향 조사	■										
사업계획서 작성	■										
개발장비 조사	■	■	■								
요구사항 정의서 작성		■	■								
서버 생성			■	■	■						
안드로이드 앱 UI 설계			■	■	■						
안드로이드 앱 UI Activity 설계					■	■					
안드로이드와 Google Speech API 연동					■	■					
서버와 안드로이드 앱 연동					■	■					
Arduino 스케치 작성					■	■	■				
Arduino 점자 입출력부 제작					■	■	■				
Arduino 와 안드로이드 앱 Bluetooth 연동						■	■				
웹 서버 내 점자-텍스트 변환 로직 구현						■	■	■			
안드로이드 앱 세부 기능 구현								■	■		
통합 테스트 및 오류 수정								■	■	■	
최종 보고서 작성										■	■

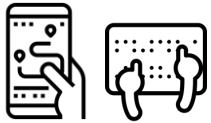
2.
사용법 소개

BEE

User Mode



비장애인: Normal mode 사용



장애인: Barrier-free mode 사용

Communication

근거리 통신

- 블루투스 모듈을 활용한 Application과 Device간 근거리 통신

원거리 통신 및 인터넷 서비스

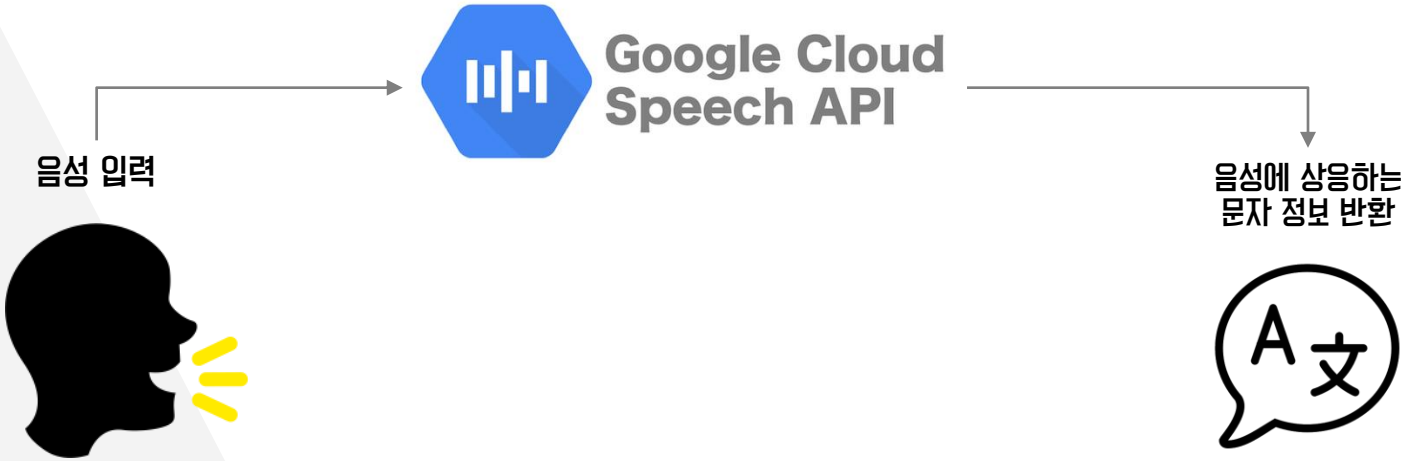
- 근거리 통신의 제약 사항을 개선한 채팅 어플리케이션 기능 구현
- 시청각장애인들이 보다 자유롭게 인터넷 기술을 활용할 수 있는 편의 기능 개발

3.
핵심 기술

BEE

음성 인식 기술

- Google Speech API를 이용한 음성 인식 기능

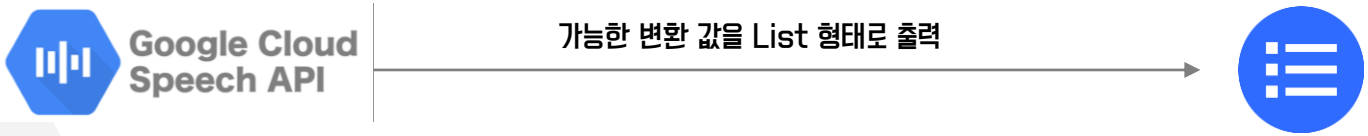


① 사용자가 입력한 음성이 Google Cloud Platform에 업로드

② Google Cloud Speech API를 거쳐 오디오에 상응하는 문자 반환

음성 인식 기술

- 음성인식 인스턴스 문자열 반환 문제

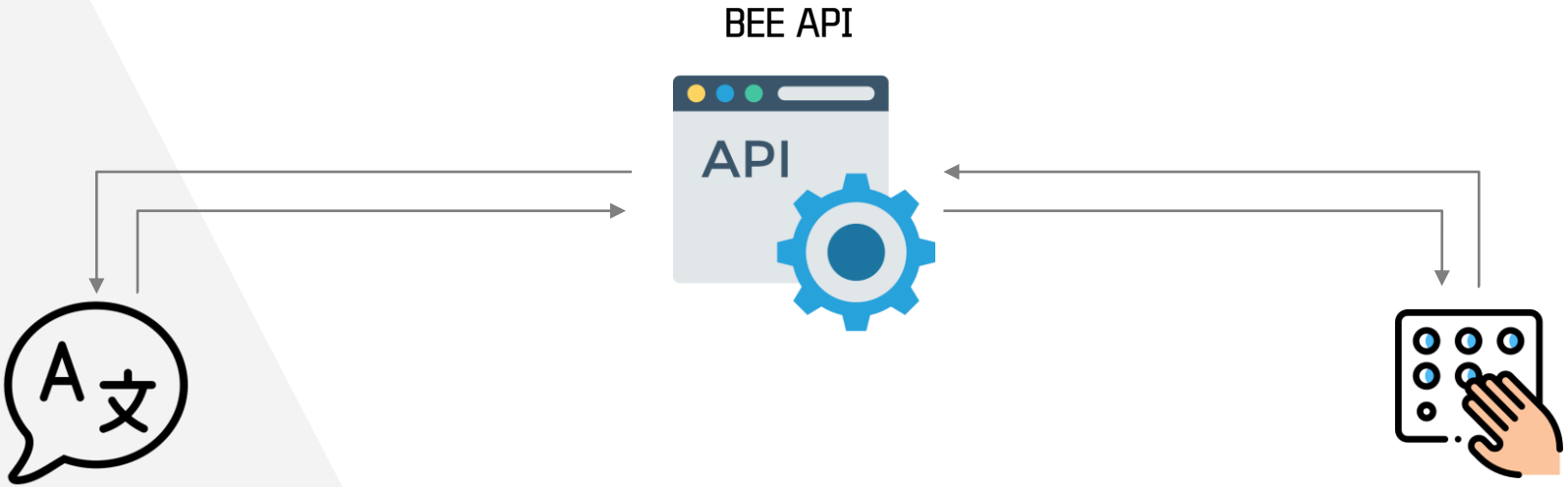



```
I/System.out: 종합설계
I/System.out: 종합 설계
I/System.out: 종합 새벽에
I/System.out: 종업 새벽에
I/System.out: 종업 설계
```

```
// 음성인식 결과를 다음 액티비티에 전달
@Override
public void onResults(Bundle results) {
    ArrayList<String> matches =
            results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION);
    Intent intent = new Intent(getApplicationContext(), CheckActivity.class);
    intent.putExtra( name: "sentence", matches.get(0)); // 1번 후보 문장 채택
    startActivity(intent);
}
```

➢ 해결 방법: 해당 List의 index 순서는 변환 확률 우선 순위이기 때문에 0번 째 index의 문장을 추출해서 사용

점자 변환



- BEE API를 통해 어플리케이션 사용자의 문자를 점자 정보로 변환하고,
  디바이스 사용자의 점자 정보를 문자 정보로 변환

점자 변환

- 문자 → 점자 변환 로직

- 한글 점자 체계에서는 약자가 존재하여
  글자의 변환 뿐만 아니라 단어의 변화 또한
  고려해야 함

➢ 단어를 글자로 나누기 전에
  2종약자에 해당하는지 확인

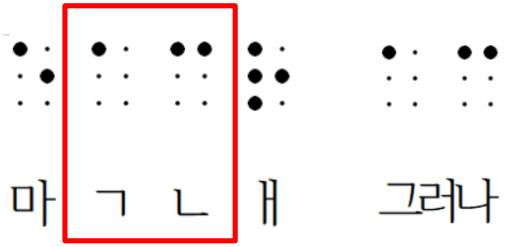
```
def fastest_handler(fast_text):
    abbreviation = {'그래서': '3228',
                    '그러나': '3236',
                    '그러면': '3218',
                    '그러므로': '3217',
                    '그런데': '3246',
                    '그리고': '3241',
                    '그리하여': '3235',
                    '것': '0728'
    }

    if fast_text in abbreviation:
        return abbreviation[fast_text]
    else:
        return False

# 이종약자 규칙 먼저 걸러줌
for elem in text:
    # 이종약자 규칙 함수 호출
    fast_result = fastest_handler(elem)

    # 이종약자 규칙에 걸렸으면 해당 반환값 전체 result에 추가
    if fast_result != False:
        result += fast_result

    # 이종약자 규칙에 안걸렸으면 문자 단위 분석 수행
    else:
        for hangul_letter in elem:
            result+= letter(hangul_letter)
```

점자 변환

- 점자 → 문자 변환 로직

- 모든 2종약자는 [종성(ㄱ) + 초성]으로 이루어지기 때문에,
  디바이스에서 입력한 점자정보가
  [종성+새로운 글자]의 초성인지,
  2종약자인지 구분을 해야함

➢ 글자와 글자 사이에 구분자 느낌표(!)를 둠으로써
새로운 글자의 시작인지 판별



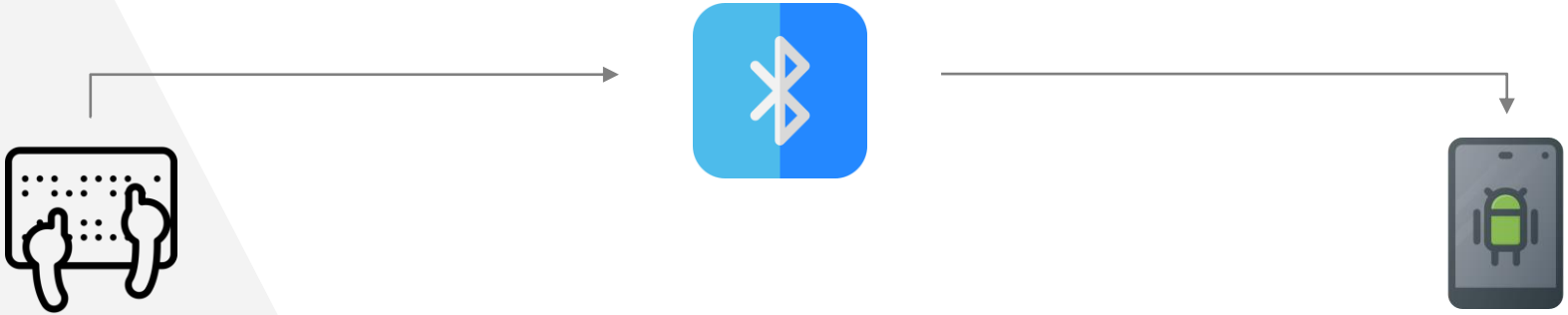

```
// 현재 만들어놓은 점자를 송신할 점자 정보에 입력
else{
  information.concat(text);
  information.concat('!');
  text.remove(0);
  delay(100);
}
```

[Arduino 점자정보 전송 시]

```
# !를 기준으로 점자정보 쪼개기
braille=braille.split('!')
```

[API 점자정보 분석 시]

점자 입력



➢ 디바이스의 입력부를 통해 점자정보를 입력하고,
입력된 정보를 블루투스를 통해 BEE Application로 전달

점자 입력

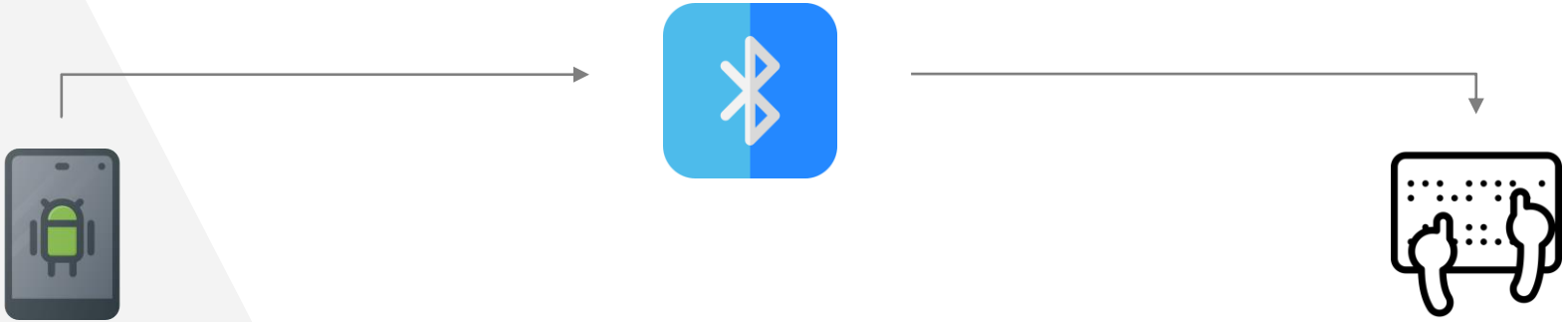
- 각각의 점자를 의미하는 6개의 버튼
- Enter: 한 철자를 완성했음을 의미
- Send: 글자 완성 및 전송
- Backspace: 현재 입력 중인 점자 삭제

➢ 총 9개의 버튼으로 구성

```
// 점자 입력부에서 입력된 버튼을 저장
b1+=!digitalRead(button1);
b2+=!digitalRead(button2);
b3+=!digitalRead(button3);
b4+=!digitalRead(button4);
b5+=!digitalRead(button5);
b6+=!digitalRead(button6);
// enter 버튼을 누르는 경우의 동작
if(digitalRead(enter)==LOW){
  sum=0;
  if(b1>0)
    b1=1;
  if(b2>0)
    b2=1;
  if(b3>0)
    b3=1;
  if(b4>0)
    b4=1;
  if(b5>0)
    b5=1;
  if(b6>0)
    b6=1;
  // 눌러진 버튼의 총합을 10진수로 바꿔줌
  sum=b1+(b2*2)+(b3*4)+(b4*8)+(b5*16)+(b6*32);
```

```
// send 버튼을 누를 때의 동작
if(digitalRead(send_msg)==LOW){
  // 점자입력 없이 send 버튼을 눌렀을 때
  if(text.length()==0){
    // 현재 입력된 점자를 어플리케이션으로 전송
    BEE.println(information);
    information.remove(0);
    delay(500);
  }
  // 현재 만들어놓은 점자를 송신할 점자 정보에 입력
  else{
    information.concat(text);
    information.concat('!');
    text.remove(0);
    delay(100);
  }
}
```

점자 출력



- 어플리케이션에서 송신한 점자정보를 디바이스의 출력부를 통해 확인

점자 출력: 관련 이슈

① 점자 출력 시 솔레노이드가 계속해서 지속되지 않음

- 점자출력부는 점자가 계속해서 돌출되어 있어야 사용자가 점자를 읽을 수 있음
- 하지만, 위의 경우처럼 개별적으로 작동시키면 솔레노이드 점자 하나가 약 0.1초 정도 튀어나왔다가 다시 들어가게 됨

```
while(start==1){
  for(int s=0; s<6; s++){
    // 점자셀 중 현재의 상태와 표현하고자 하는 부분을 계산하여 동작
    if(cell[s]!=stat[s]){
      digitalWrite(SOL[s],LOW);
      delay(50);
      digitalWrite(SOL[s],HIGH);
      // 현재 상태 수정
      if(stat[s]==0){
        stat[s]=1;
      }
      else{
        stat[s]=0;
      }
    }
  }
  start=0; // 무한루프를 막아줌
}
```

➢ 해결방법: Retractable의 원리를 활용하여 솔레노이드가 동작 시 점자부를 돌출 시키고, 이후 다시 동작할 때 점자부가 들어갈 수 있게 설계함

점자 출력: 관련 이슈

② 출력부 구동 시 솔레노이드가 한 번에 출력되지 않음

- BEE Device에 솔레노이드는 하나당 정격전압 12V를 사용
- 솔레노이드 6개가 병렬로 연결되어 있어 동시에 작동하기 위해서는 6(EA) * 12(V) = 최대 72V 가 필요
- 72V 외부전압을 사용하기 위해 9V 건전지 8개를 직렬로 연결해야 하므로 휴대용에 적합하지 않음

➢ 해결방법: 6개를 병렬로 연결하되, 한 번에 작동하는 것이 아니라
개별적으로 작동시키도록 하여 낮은 전압에서도 작동할 수 있도록 설계함

실시간 채팅

- Google Firebase를 이용한 실시간 채팅 기능

➢ Google Firebase의 Realtime Database를 활용한 실시간 채팅 기능의 구현

실시간 채팅

- Google Firebase를 이용한 실시간 채팅 기능

```
btnSend.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String stText = etText.getText().toString();

        // 사용자가 메시지를 입력하지 않았을 때,
        if (stText.equals("") || stText.isEmpty()){
            Toast.makeText( context: ChatActivity.this,  text: "내용을 입력해주세요", Toast.LENGTH_SHORT).show();
        }
        // 사용자가 메시지를 입력했을 때,
        else {
            String formattedDate = df.format(c.getTime());
            DatabaseReference myRef = database.getReference( s: "users").child(chatId).
                    child("chat").child(formattedDate);

            Hashtable<String, String> chat = new Hashtable<~>();
            chat.put("email", email);
            chat.put("text", stText);
            myRef.setValue(chat);
            etText.setText("");
        }
    }
});
```

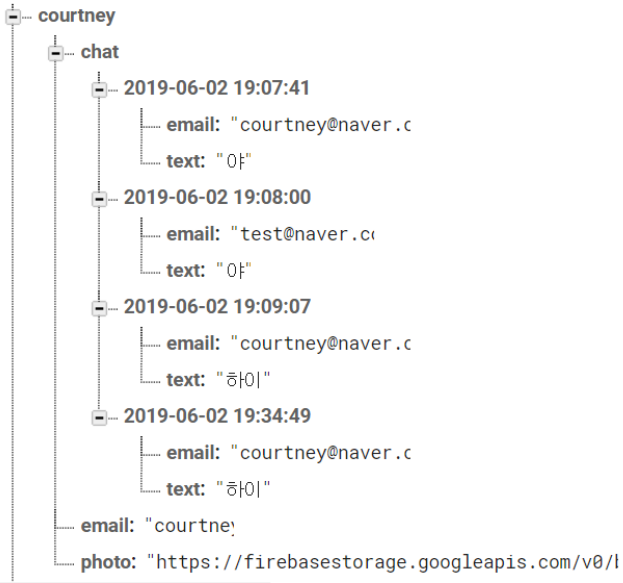
Firebase 로드

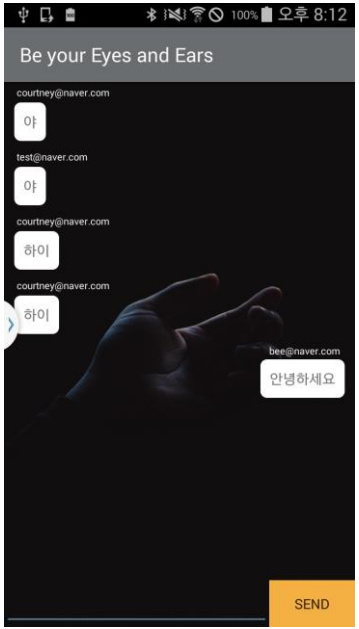
HashTable 형태로
채팅 내역 저장

- Firebase Realtime Database에 채팅 기록을 NoSQL 형태로 저장

실시간 채팅

- Google Firebase를 이용한 실시간 채팅 기능

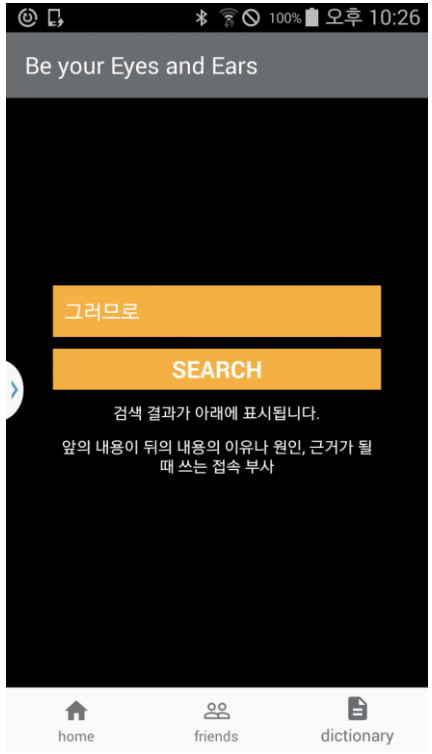




➢ Firebase에 NoSQL 형태로 저장된 채팅 내역을 실시간으로 받아오는 실시간 채팅

사전 검색

```python
#-*- coding: utf-8 -*-
import requests
from bs4 import BeautifulSoup

print("검색하려는 단어를 입력하세요.")
word = input()

url = "https://stdict.korean.go.kr/search/searchResult.do?&searchKeyword=" + word

response = requests.get(url)
soup = BeautifulSoup(response.content, "lxml")

result = ""
try:
    result += soup.find('ul', {'class': 'result'}).find('dt').find('font', {'class': 'dataLine'}).get_text()
except:
    result = "사전에 등재되어 있지 않습니다."
print(result)
```



- Web Crawling을 통한 사전 검색 결과를 점자로 반환

4.
경쟁력 및 사업성

BEE

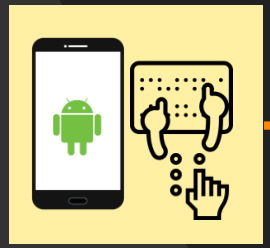
구매 비용

상품명	가격(원)
브레드보드 830핀	6,600
테스트 소켓 점퍼 케이블	770
솔레노이드 액추에이터	49,500
5V 릴레이 모듈 8채널	5,500
12mm 사각 택트버튼	1,860
40핀 커넥터 M-M	990
40핀 커넥터 F-M	990

상품명	가격(원)
아두이노 블루투스 모듈 슬레이브	4,100
아두이노 우노 R3 호환보드	5,500
USB2.0 A-B케이블	990
3mm LED RGB	1,320
막대저항 R1	550
브레드보드 400핀	1,760
4핀 택트 버튼 스위치	320

= 합계 **82,450원** 의 비용 발생 VS 한소네 포켓(U2 미니) **4,495,000원**

경쟁력 및 사업성



- 원가 절감을 통한 가격 경쟁력 확보
- 장애인과 비장애인 모두 이용 가능한 범용적 시스템 구축
- 타 Device와의 연동을 통한 기능적 확장

기대 효과

기존 현황

- 의사소통 보조 기기 및 서비스의 부족으로 인한 시청각장애인의 제한적 의사소통

- 실시간 소통이 가능한 저가형 양방향 의사소통 보조 시스템
- 시청각장애인의 근본적인 불편 해소와 원활한 의사소통 실현
- 구매력이 낮은 시청각장애인의 생활 편의 향상

5.
결론

BEE

프로젝트 특장점

구분	특징
장점	**저렴한 가격:** 기존 보조기기보다 가격을 대폭 낮춰 시청각장애인들의 접근성 향상
	사용 용이성: 점자를 숙지하고 있지 않은 사람도 손쉽게 사용 가능
	확장 가능성: 의사소통 뿐만 아니라 다양한 기능을 Mobile Application에 추가하여 인터넷 편의 서비스 이용 가능케 할 수 있음
단점	스마트 디바이스에 접목시키기 위해 Device 제품의 소형화 필요

문제점 및 해결 방안

① 점자 셀을 하나 밖에 구현하지 못함

– 해결 방안

· 시청각장애인의 모바일 환경을 고려하여 휴대가 간편한 18셀의 최소형 점자 정보 단말기 제작

② 음성인식 성능 개선 필요

– 해결 방안

· 원거리 음성 인식 성능 향상을 위한 기술 개발 (e.g. Kaldi Framework)

· 잡음 환경에 효과적인 음성 인식을 위한 음성 향상 기법 도입

차후 방향 : 활용 가능성

- ✓ 기존에 시청각장애인이 사용하기 어려웠던 스마트 디바이스(노트북, 스마트폰 등)에 BEE 기술을 추가적으로 탑재하여 접근성 향상

- ✓ 모바일 어플리케이션에 인터넷 서비스를 이용할 수 있는 편의 기능 추가

  e.g) 날씨 알림, 뉴스 검색

개발 언어 및 환경

- 개발 환경: OS (Ubuntu)
- 개발 도구: Android Studio, PyCharm
- 개발 언어: C, Java, Python
- 데이터베이스: Firebase
- 시뮬레이터: 삼성 갤럭시 ALPHA (android 8.0)

감사합니다