

종합설계 프로젝트 수행 보고서

프로젝트명	청각장애인을 위한 스마트워치 드라이브스루 주문 시스템
팀번호	S3-8
문서제목	수행계획서() 2차발표 중간보고서() 3차발표 중간보고서() 최종결과보고서(O)

2020.12.03

팀원 : 강현빈(팀장)

백승아

손지수

유해미

지도교수 : 전광일 교수 (인)

문서 수정 내역

작성일	대표작성자	버전(Revision)	수정내용	
2020.1.20	강현빈(팀장)	1.0	수행계획서	최초작성
2020.2.18	유해미	1.1	수행계획서	수정
2020.3.02	강현빈(팀장)	2.0	2차발표자료	설계서추가
2020.3.25	손지수	2.1	2차발표자료	설계서추가
2020.4.28	강현빈(팀장)	3.0	3차발표자료	시험결과 추가
2020.6.27	유해미	3.1	3차발표자료	시험결과 수정
2020.12.03	백승아	4.0	최종결과보고서	결론 추가

문서 구성

진행단계	프로젝트 계획서 발표	중간발표1 (2월)	중간발표2 (4월)	학기말발표 (6월)	최종발표 (10월)
기본양식	계획서 양식	계획서 양식	계획서 양식	계획서 양식	계획서 양식
포함되는 내용	I. 서론 (1~6) II. 본론 (1~3) 참고자료	I. 서론 (1~6) II. 본론 (1~4) 참고자료	I. 서론 (1~6) II. 본론 (1~5) 참고자료	I. 서론 (1~6) II. 본론 (1~7) 참고자료	I II III

이 문서는 한국산업기술대학교 컴퓨터공학부의
 “종합설계” 교과목에서 프로젝트 “청각장애인을 위한
 스마트워치 주문알림 시스템”을 수행하는
 S3-8(강현빈, 백승아, 손지수, 유해미)들이 작성한 것으로 사용하기
 위해서는 팀원들의 허락이 필요합니다.

목 차

I. 서론

1. 작품선정 배경 및 필요성	4-5
2. 기존 연구/기술동향 분석	6
3. 개발 목표	7
4. 팀 역할 분담	8
5. 개발 일정	9
6. 개발 환경	9

II. 본론

1. 개발 내용	10
2. 문제 및 해결방안	12
3. 시험시나리오	13
4. 상세 설계	19
5. Prototype 구현	25
6. 시험/ 테스트 결과	29
7. Coding & DEMO	34

III. 결론

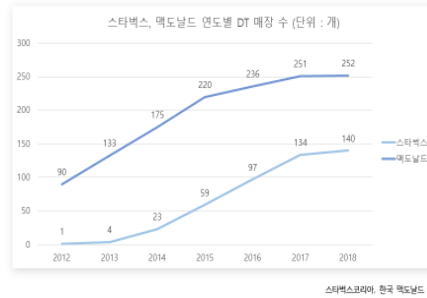
1. 연구 결과	37
2. 작품제작 소요재료 목록	37

참고자료	38
------------	----

I. 서론

1. 작품선정 배경 및 필요성

<p>웨어러블 기기 수요의 증가</p>	<div data-bbox="684 445 1197 801" data-label="Figure"> <table border="1"> <caption>스마트워치 최종 사용자 지출</caption> <thead> <tr> <th>연도</th> <th>지출액</th> </tr> </thead> <tbody> <tr> <td>2018</td> <td>12,412</td> </tr> <tr> <td>2019</td> <td>17,047</td> </tr> <tr> <td>2020</td> <td>22,803</td> </tr> </tbody> </table> <p>IT 자문기관 가트너(Gartner)</p> </div> <p>IT 자문기관 가트너(Gartner)는 2020년, 전 세계 사용자들이 웨어러블 디바이스인 스마트워치에 총 220억 달러를 지출할 예정이라고 발표했다.</p> <p>증가하는 웨어러블 기기의 수요로 인해 청각장애인은 활용에 따른 수준 격차로 인해 사회적 의사소통 단절, 정보 불평등으로 인해 제한된 삶이 될 우려가 크다. 따라서 웨어러블 기기인 스마트워치를 통해 청각장애인에게 다양한 서비스를 제공하고자 한다.</p>	연도	지출액	2018	12,412	2019	17,047	2020	22,803				
연도	지출액												
2018	12,412												
2019	17,047												
2020	22,803												
<p>청각장애인 소통의 어려움</p>	<p>청각장애인은 일상생활에서 소리를 인식하지 못하여 불편함을 겪는 경우가 많으며, 주문이나 예약같이 전화나 구어로 해야 하면 직원이 소통의 답답함을 느껴 불친절한 태도를 보여 차별하는 때도 있다.</p> <p>또한 대부분 긴급경보는 소리를 이용하여 알리기 때문에 위급 상황에서도 대처하기가 어려운 경우가 많아 대부분 시각경보기를 통하여 시각적으로 경고상황을 알려주지만 생각보다 보급된 곳이 많지 않아 사고가 났음에도 불구하고 확인이 불가능한 경우도 많다.</p> <p>따라서 스마트워치를 통해 시각적 정보와 음성인식 기법을 통해 청각장애인의 소통을 돕고자 한다.</p>												
<p>청각장애인 드라이브 스루 접근성</p>	<div data-bbox="699 1673 1144 1966" data-label="Figure"> <table border="1"> <caption>주요 브랜드 드라이브스루(DT) 매장</caption> <thead> <tr> <th>브랜드</th> <th>매장 수</th> </tr> </thead> <tbody> <tr> <td>스타벅스</td> <td>140개</td> </tr> <tr> <td>맥도날드</td> <td>252개</td> </tr> <tr> <td>롯데리아</td> <td>56개</td> </tr> <tr> <td>버거킹</td> <td>51개</td> </tr> <tr> <td>KFC</td> <td>12개</td> </tr> </tbody> </table> <p>스타벅스코리아, 한국 맥도날드</p> </div>	브랜드	매장 수	스타벅스	140개	맥도날드	252개	롯데리아	56개	버거킹	51개	KFC	12개
브랜드	매장 수												
스타벅스	140개												
맥도날드	252개												
롯데리아	56개												
버거킹	51개												
KFC	12개												



최근에 드라이브 스루가 활성화되면서 드라이브 스루를 하는 매장이 많아지고 있다. 하지만 보통 주문받는 곳은 사람이 없고 마이크를 통하여 음성인식으로 주문을 받기 때문에 청각장애인들은 이용하는 것이 쉽지 않다. 그래서 청각장애인들을 위해 터치스크린을 설치하거나 계산대에서 바로 주문할 수 있게 도와주는 곳도 있지만 그렇지 않은 매장이 대부분이어서 청각장애인들이 많은 불편함을 겪고 있다.

불편함뿐만 아니라 청각장애인을 차별하는 사례도 있었다. 2019년 8월 31일 캠벨 해밀턴 애비뉴에 위치한 패스트푸드점 ‘잭 인더박스’ 직원이 청각 장애 고객을 조롱하고 서비스를 거부한 사건이 있었다. 이 패스트푸드점 직원은 청각 장애를 가진 고객이 드라이브 스루 스피커를 지나쳐 카운터로 와서 햄버거를 주문하자 스피커로 다시 돌아가 주문하라며 흥분한 채 소리를 쳤던 사례가 있다.

이처럼 영상이나 음성을 통해 주문하는 시스템이 대다수인 드라이브 스루는 청각장애인 접근성 측면에서 떨어지는 것을 확인할 수 있다.

2. 기존 연구/기술동향 분석

- 최근 증가하고 있는 드라이브 스루 사례들로, 이에 대해 혜택을 받지 못하는 청각장애인에 대한 서비스 고려가 필요한 상황

My DT Pass	스타벅스에서 2018년 6월 5일 하이패스처럼 빠른 결제가 가능한 My DT Pass 서비스를 도입했다. 이는 고객의 차량정보를 스타벅스 선불식 충전 카드와 연동시켜, 스타벅스 드라이브스루 이용 시 별도의 결제수단을 제시하지 않고, 자동 결제를 통해 주문한 메뉴를 바로 받을 수 있는 드라이브 스루 전용 서비스이다.
드라이브 스루 환전	우리은행은 2019년 11월 14일 신세계면세점 방문 고객을 대상으로 드라이브 스루(Drive Thru) 환전서비스 제공을 위한 업무협약(MOU)을 체결했다. 드라이브 스루 환전은 모바일로 환전을 미리 신청하면, 고객이 차량을 이용해 드라이브 스루 환전소에 방문하여 본인인증을 거친 뒤 환전 그리고 현금 인출이 가능한 서비스다.

- 청각장애인을 위한 안전 시스템 사례

BE-J11	블루엔터프라이즈사에서 개발한, 다양한 상황에서 발생하는 소리의 진동을 분석하여 사용자에게 그림으로 보여주는 스마트워치이다. 알림 기능만 제공하는 스마트워치 기기를 따로 구매해야 한다는 단점이 있다.
누구나 넥 밴드	주변 소음보다 큰 소리가 발생하였을 때 소리가 발생한 곳의 방향을 넥 밴드의 진동으로 알려준다. 사용자의 주변 소음보다 높은 고음에만 반응한다.
H-bell	불필요한 소리 대신 빛을 내는 사용자 맞춤형 알림으로 빛으로 소리의 기능을 대체하여 난청 노인, 청각장애인에게 효과가 크다. 기능이 한정적이다.

3. 개발 목표

- 청각장애인 웨어러블 기기 사용성 증가, 청각장애인 원활한 소통 도움, 청각장애인 드라이브스루 접근성 향상

청각장애인 웨어러블 기기 사용성 증가	<p>청각장애인은 소리 감각의 발달이 조금 달라, 시각이나 촉각의 신경이 발달하였다. 따라서 시각적 이미지로 인식하거나, 진동을 통해 알림을 받는 경우가 대다수이며 이에 따른 보조기기나 의사소통 기기들이 생겨나고 있다. 인공와우나 보청기 등 소리를 증폭시켜주거나 들을 수 있도록 해주는 기기들이 많지만 방수등급을 받은 보조기기는 다양하지 않아 일상생활 이외에 스포츠 활동이나 물에 직접 닿는 활동들에선 착용할 수 없으며, 큰 모양새 때문에 청각장애인임을 드러내고 싶지 않은 사람들에게도 걸로 티가 많이 나는 모습이다.</p> <p>이에 반해 스마트 워치는 일반인들도 알림을 위하여 착용하는 부분에서 일반인과 청각장애인이 착용하는 것에 대해 보편적으로 보급되어있기 때문에 걸모습으로 차별받지 않을 수 있고, 스마트 워치는 손목에 착용하기 때문에 촉각적으로 진동 알림을 받을 수 있으며, 또한 스마트 워치의 디스플레이를 통하여 알림을 표시할 수 있으므로 시각적 표현으로 나타내기 좋은 기기이다. 그렇기 때문에 사람이 많은 음식점이나, 카페 등 사람들 사이에서 청각장애인용 보조기기를 착용하지 않고, 애플리케이션이 보급된 곳에서는 청각장애인이란 것에 대해 차별받지 않고 다양한 서비스를 제공하는 것이 목표이다.</p>
청각장애인 원활한 소통 도움	<p>최근 의사소통하지 않고도 소통할 수 있는 기기들이 무수히 많이 발달하고 있다. 소통하지 않고 주문할 수 있는 애플리케이션이나, 소통하지 않고도 결제가 되는 시스템 등 편리함을 주고 있다.</p> <p>하지만 청각장애인에게 원활한 소통의 기회를 주고자 하였고, 조금의 소통으로도 편리함을 느낄 수 있는 서비스를 제공하고 이 시스템을 기획하게 되었다.</p>
청각장애인 드라이브스루 접근성 향상	<p>기존 드라이브 스루의 대부분은 화면을 통하거나, 음성을 통해 종업원과 대화를 통해 주문하는 방식이다. 하지만 소리 감각이 조금 다른 청각장애인에게는 어려운 방식이다.</p> <p>따라서 기존의 방식에서 시각적인 정보와 음성인식을 통한 정보를 제공함으로써, 편리한 주문 방식 시스템을 제공하는 것이 목표이다.</p>

4. 팀 역할 분담

	강현빈	유해미	백승아	손지수
자료수집	<ul style="list-style-type: none"> - 딥러닝 및 파이썬 학습 - 사용자 요구사항 조사 - 팀원들의 업무 분담과 계획관리 	<ul style="list-style-type: none"> - 딥러닝 및 파이썬 학습 - 개발환경 및 개발 방법 조사 	<ul style="list-style-type: none"> - 스마트워치와 애플리케이션 연동방법 조사 - 하드웨어 부품 선정 	<ul style="list-style-type: none"> - 관련 연구 및 사례 조사 - 전처리 알고리즘 조사
설계	<ul style="list-style-type: none"> - 딥러닝 알고리즘 설계 - 애플리케이션과 파이어베이스 간 연결 	<ul style="list-style-type: none"> - 딥러닝 알고리즘 설계 - 모듈 설계 및 클래스 간의 관계 설계 및 분석 	<ul style="list-style-type: none"> - 모듈 사이에 제공하는 인터페이스 분석 및 설계 - Firebase를 이용한 앱 설계 	<ul style="list-style-type: none"> - 각 모듈 간에 전달되는 데이터 흐름 분석 - Firebase를 이용한 앱 설계
구현	<ul style="list-style-type: none"> - 딥러닝 알고리즘 구현 - 안드로이드 UI 개발 - 데이터베이스 구현 	<ul style="list-style-type: none"> - 딥러닝 알고리즘 구현 - 안드로이드 UI 개발 - STT API 구현 	<ul style="list-style-type: none"> - Firebase를 이용한 앱 개발 - 스마트워치 UI 개발 - 스마트워치 연동 	<ul style="list-style-type: none"> - Firebase를 이용한 앱 개발 - 안드로이드 UI 개발 - STT API 구현
테스트	<ul style="list-style-type: none"> - 애플리케이션 작동 및 제어 테스트 - 통합 테스트 및 유지보수 			

5. 개발 일정

개발 일정 1.0	주요 업무	2019년	2020년							
		12	1	2	3	4	5	6	7	8
	주제 선정 및 계획									
	자료 조사									
	요구 사항 분석									
	시스템 설계									
	시스템 구현									
	테스트									
	유지 보수									
	최종 발표									

개발 일정 1.1	주요업무	2019	2020							
		12월	1월	2월	3월	4월	5월	6월	7월	8월
	주제선정 및 계획									
	자료조사									
	요구사항분석									
	시스템설계									
	시스템구현									
	테스트									
	유지보수									

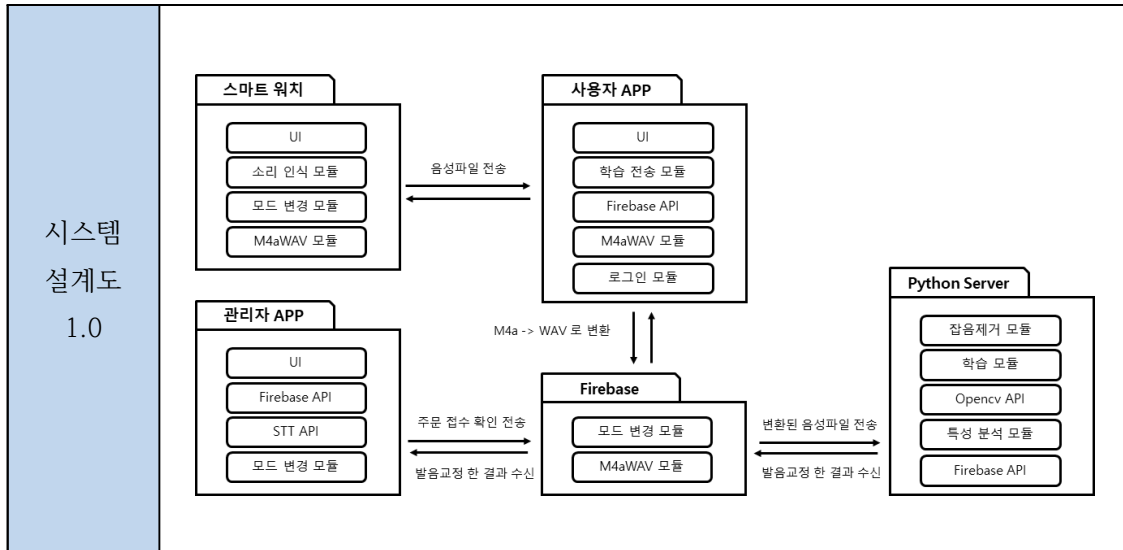
6. 개발 환경

개발 언어	Java
	Kotlin
	Python
개발방법론	Agile
개발환경	Android studio
	TensorFlow
	Firebase
하드웨어	Galaxy Gear S
	Galaxy S6

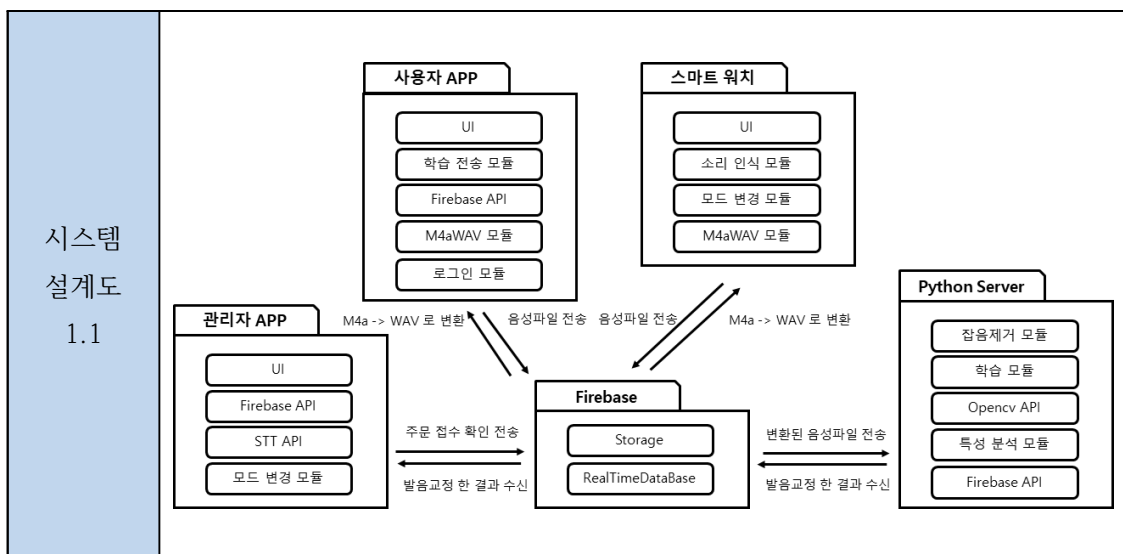
II. 본론

1. 개발 내용

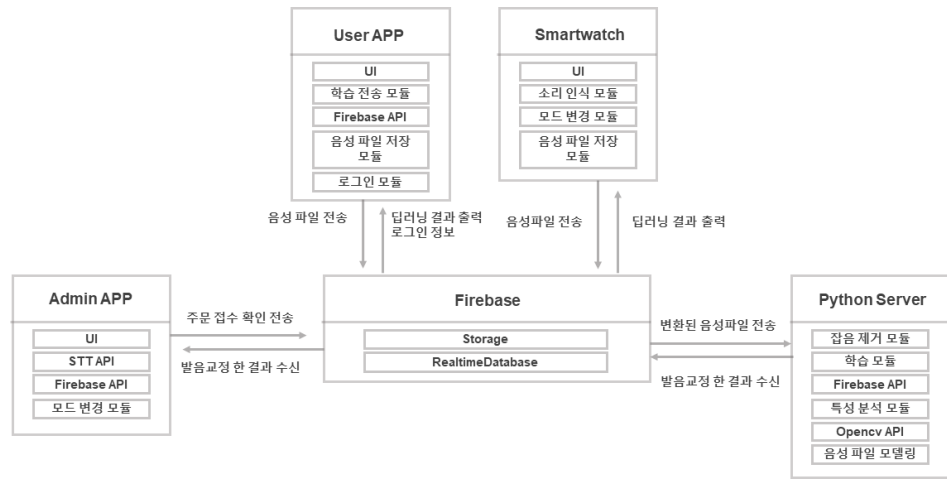
- 관리자 애플리케이션과 사용자 애플리케이션은 Firebase를 통해 연결
- 파이썬을 통해 발음 교정된 단어는 다시 관리자와 사용자 애플리케이션에 전달



- 사용자가 사용하는 스마트폰 애플리케이션과 스마트워치 애플리케이션의 발음 교정 기능은 동일
- 사용자가 인식한 음성은 Firebase storage에 전달되고, 이는 파이썬 서버에서 분석하여 다시 관리자 애플리케이션, 사용자 애플리케이션, 스마트워치 애플리케이션으로 전달



시스템 설계도 2.1

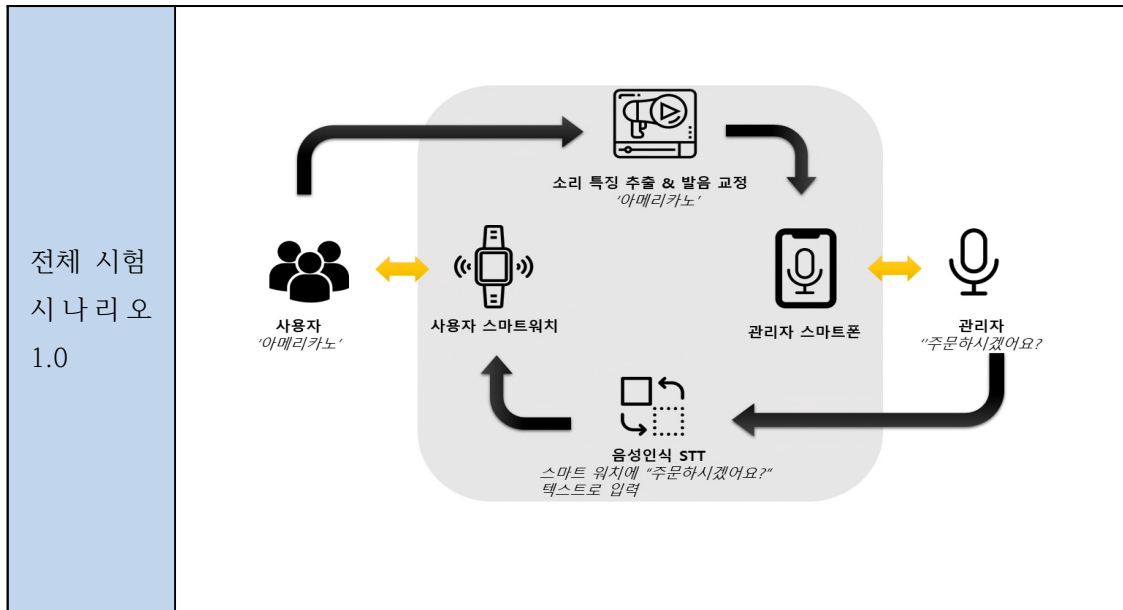


2. 문제 및 해결방안

문제	해결방안
<ul style="list-style-type: none"> • 드라이브 스루 시스템 특성상 점원과 면대면 소통이 아닌, 음성으로 주문이 진행되는 경우가 많아 청각장애인의 주문이 어려움 최근 카페, 음식점뿐만 아니라 다양한 시설에서 드라이브 스루 시스템을 도입중이다. 하지만 음성으로 주문이 진행되는 경우가 많아 청각장애인이 주문을 시작하는 것이 어렵다.	<ul style="list-style-type: none"> • 주문 시작 단계부터 음성인식 시스템을 도입
<ul style="list-style-type: none"> • 제품의 준비가 완료되어, 직원이 “30번 손님 메뉴 나왔습니다.” 라고 말 할 때, 이를 알려주는 기기가 없을 경우, 청각장애인 확인 어려움 	<ul style="list-style-type: none"> • 스마트폰과 스마트워치 애플리케이션을 활용 • 간단한 주문완료 버튼을 통한 알림으로 쉽게 확인 가능 직원(관리자)와 손님(청각장애인) 서로 어플을 통신 하여 스마트 워치에 알림으로 전송하여 소통
<ul style="list-style-type: none"> • 청각장애인 손님 중 음성인식 서비스와 알림 서비스를 받기를 원하는 손님을 구분 필요 직원이 청각장애인 손님이 왔는지 관리 및 확인이 불가함	<ul style="list-style-type: none"> • 관리자가 서비스를 받고자 하는 손님 인지 구분하기 위해, 로그인 서비스를 통해 관리 GPS를 통한 위치인식, 로그인을 통하여 회원관리 서비스를 이용함
<ul style="list-style-type: none"> • 직원이 주문내역을 확인하거나 주문내역에 대해 문의할 때, 청각장애인이 이를 확인하기 어려움 	<ul style="list-style-type: none"> • 애플리케이션에 ‘주문 확인’을 구성하여 이를 쉽게 확인 직원이 말하는 내용을 청각장애인의 스마트 워치로 전송하여 소통

3. 시험 시나리오

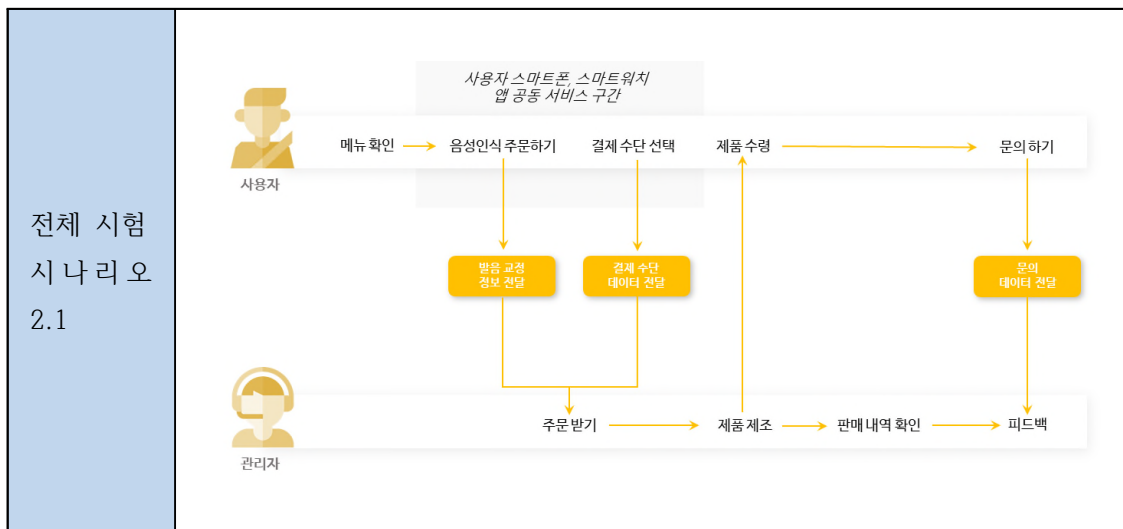
- 사용자가 어눌한 발음으로 '아메리카노' 언급
- 사용자의 음성을 분석하여 올바른 발음으로 관리자에게 전송
- 이를 확인한 관리자는 메뉴 주문을 받고, 확인되면 즉시 사용자에게 알림



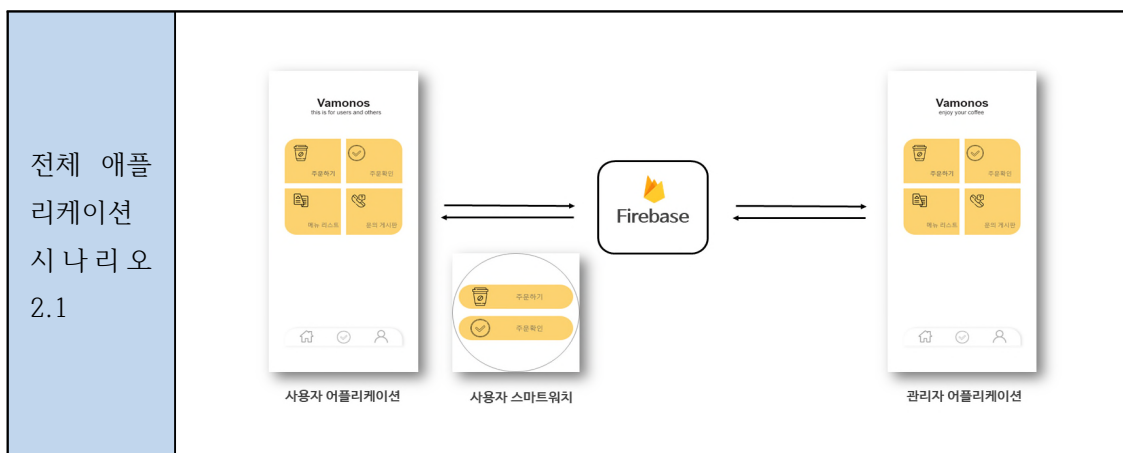
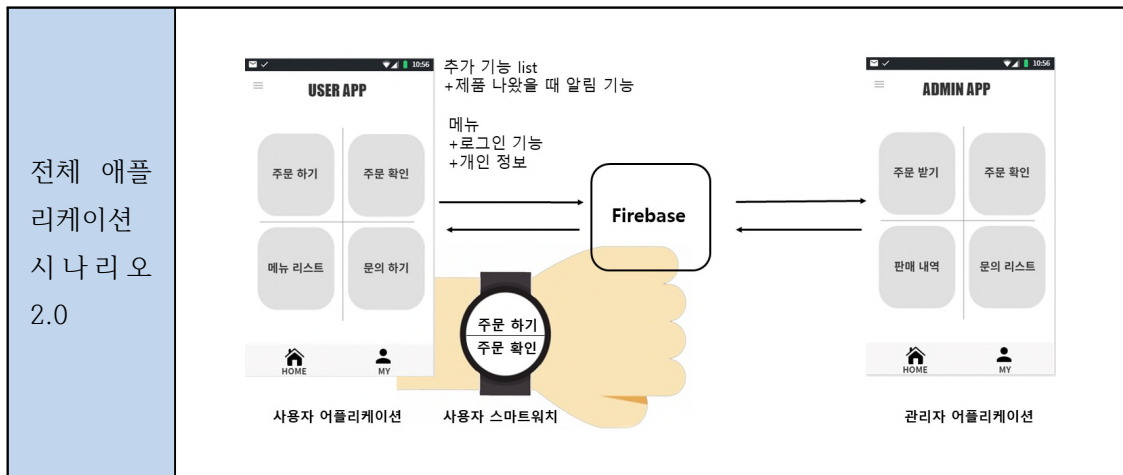
- 드라이브 스루에 도착한 사용자가 어눌한 발음으로 '아메리카노'라고 언급하면, 이를 분석하여 올바른 발음으로 관리자에게 전송
- 결제 수단 선택은 버튼 선택으로 진행
- 이를 확인한 관리자는 STT를 통해 사용자에게 메뉴 주문 확인
- 확인되면 즉시 사용자에게 알림과 동시에 상품 전달



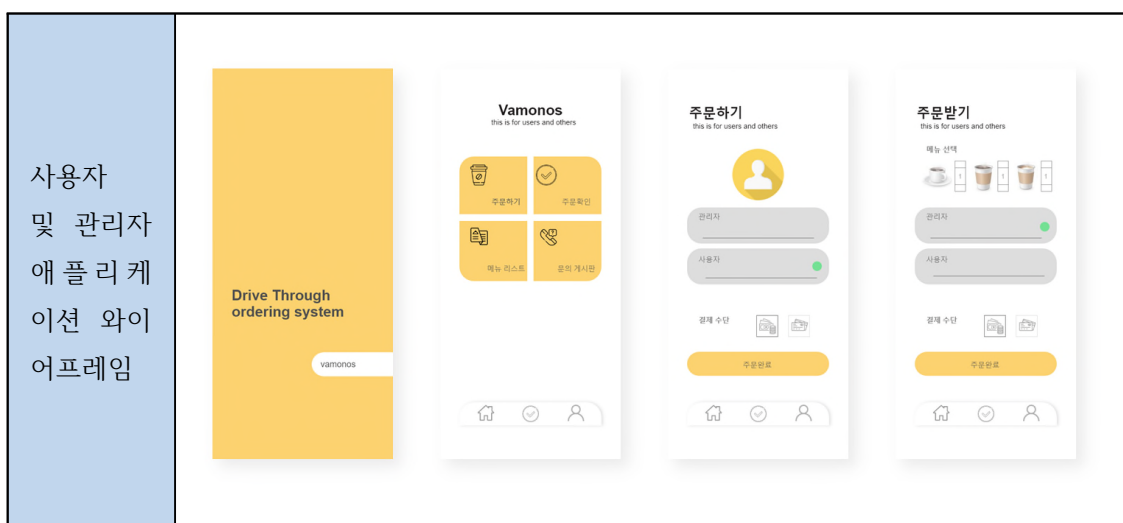
- 사용자 스마트폰과 스마트워치로 음성인식 주문하기, 결제 수단 선택 서비스 사용 가능
- 사용자는 메뉴 확인한 후 음성인식을 통해 주문을 하고, 발음 교정이 된 정보를 관리자 앱에 전달



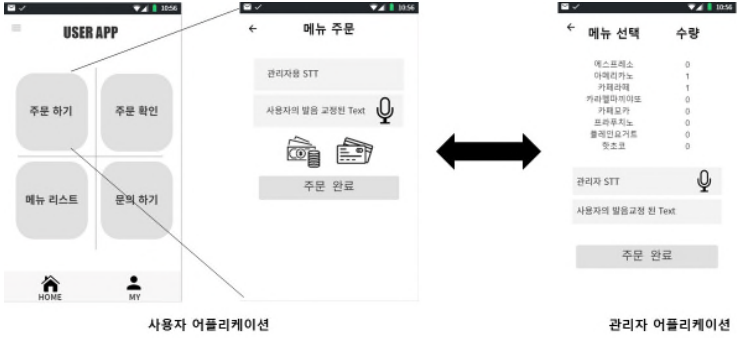



- 사용자 애플리케이션, 사용자 스마트워치 애플리케이션, 관리자 애플리케이션으로 구성
- 세 애플리케이션은 Firebase를 통해 연결

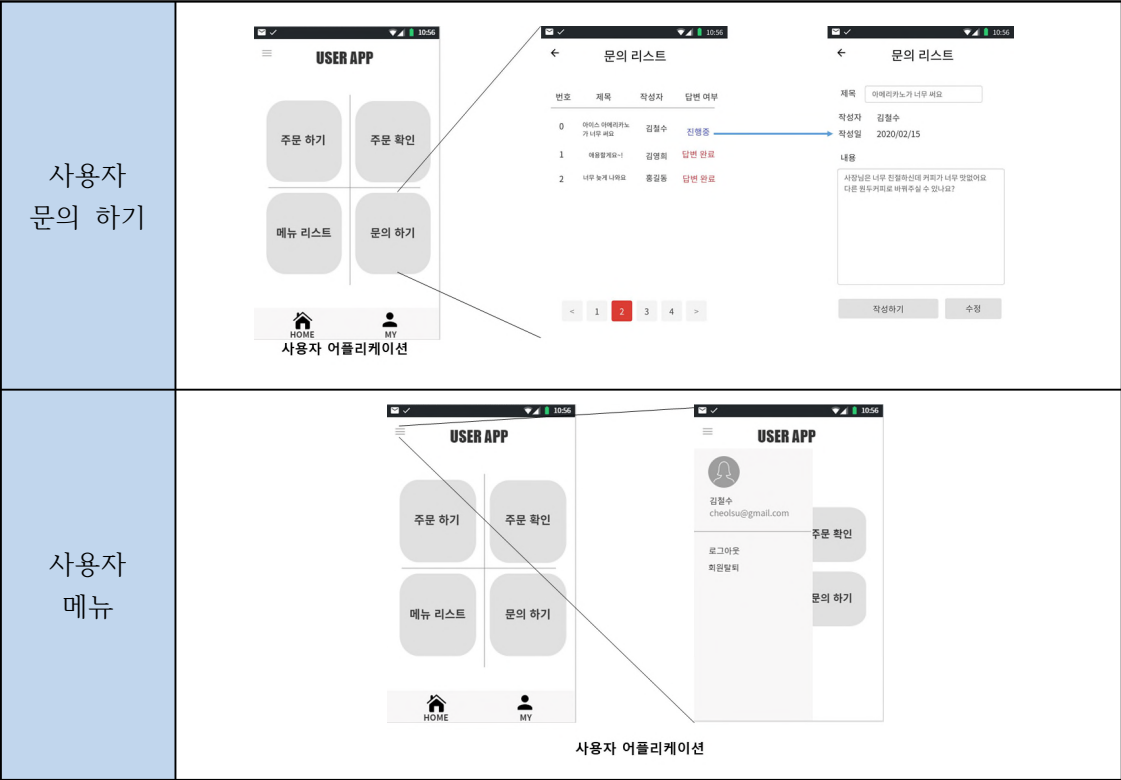


- 사용자, 관리자 애플리케이션 와이어프레임(2.1)

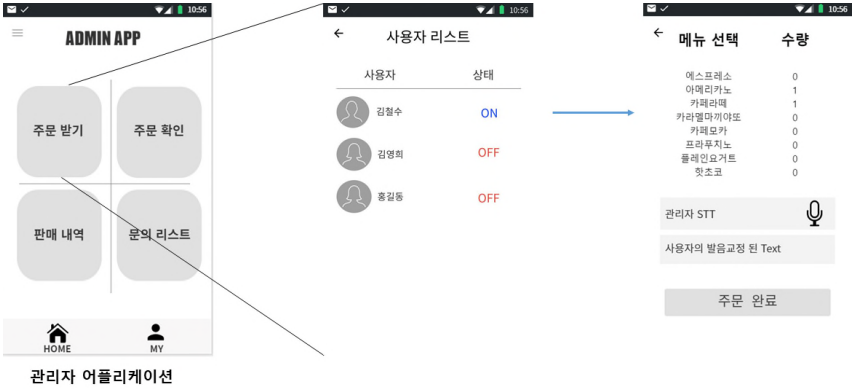





• 사용자 애플리케이션 상세 시스템 시나리오(2.0)

<p>사용자 스마트폰 주문하기</p>	 <p>사용자 어플리케이션</p> <p>관리자 어플리케이션</p>
<p>사용자 스마트워치 주문하기</p>	 <p>사용자 스마트워치</p> <p>관리자 어플리케이션</p>
<p>사용자 주문 확인</p>	 <p>사용자 어플리케이션</p> <p>사용자 스마트워치</p>
<p>사용자 메뉴 리스트</p>	 <p>사용자 어플리케이션</p>

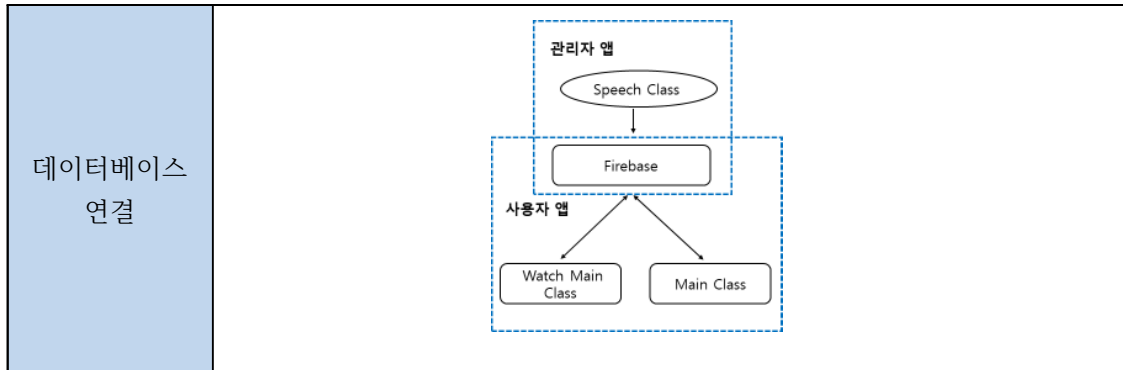


• 관리자 애플리케이션 상세 시스템 시나리오(2.0)

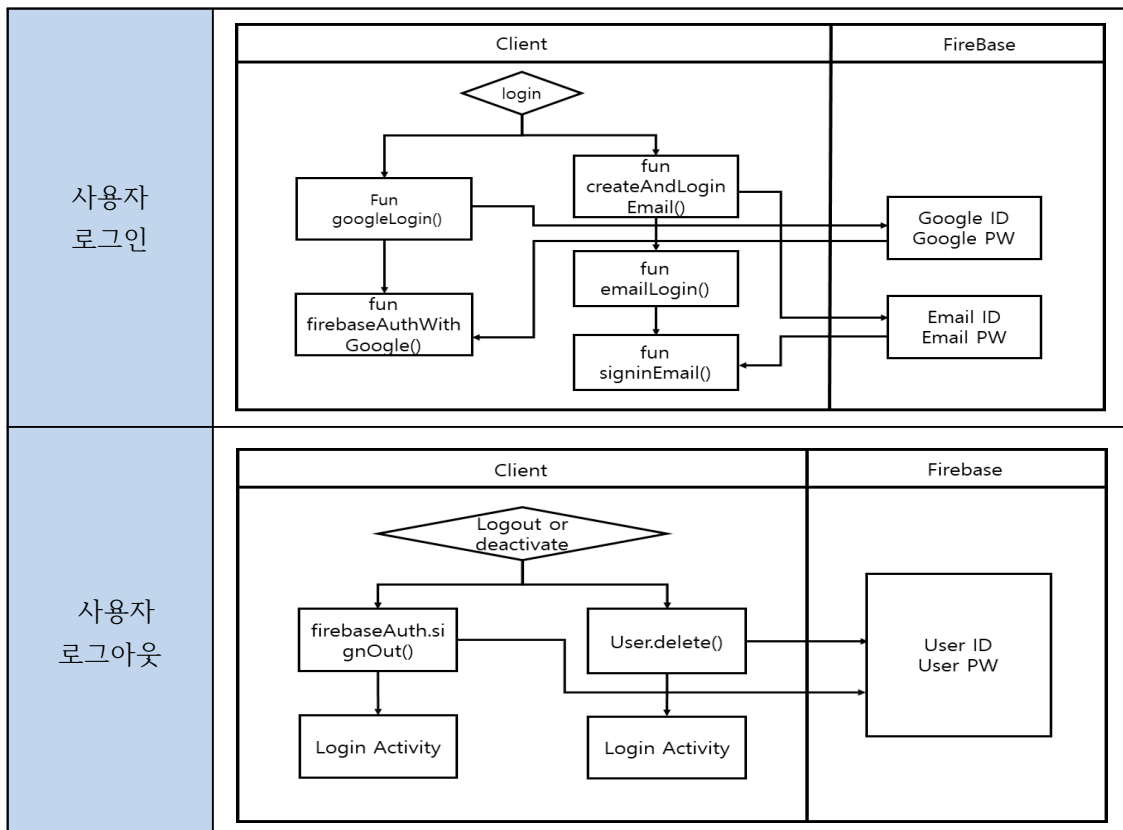
<p>관리자 주문 받기</p>	 <p>관리자 어플리케이션</p>
<p>관리자 주문확인</p>	 <p>관리자 어플리케이션</p>
<p>관리자 판매 내역</p>	 <p>관리자 어플리케이션</p>
<p>관리자 문의 리스트</p>	 <p>관리자 어플리케이션</p>

4. 상세 설계

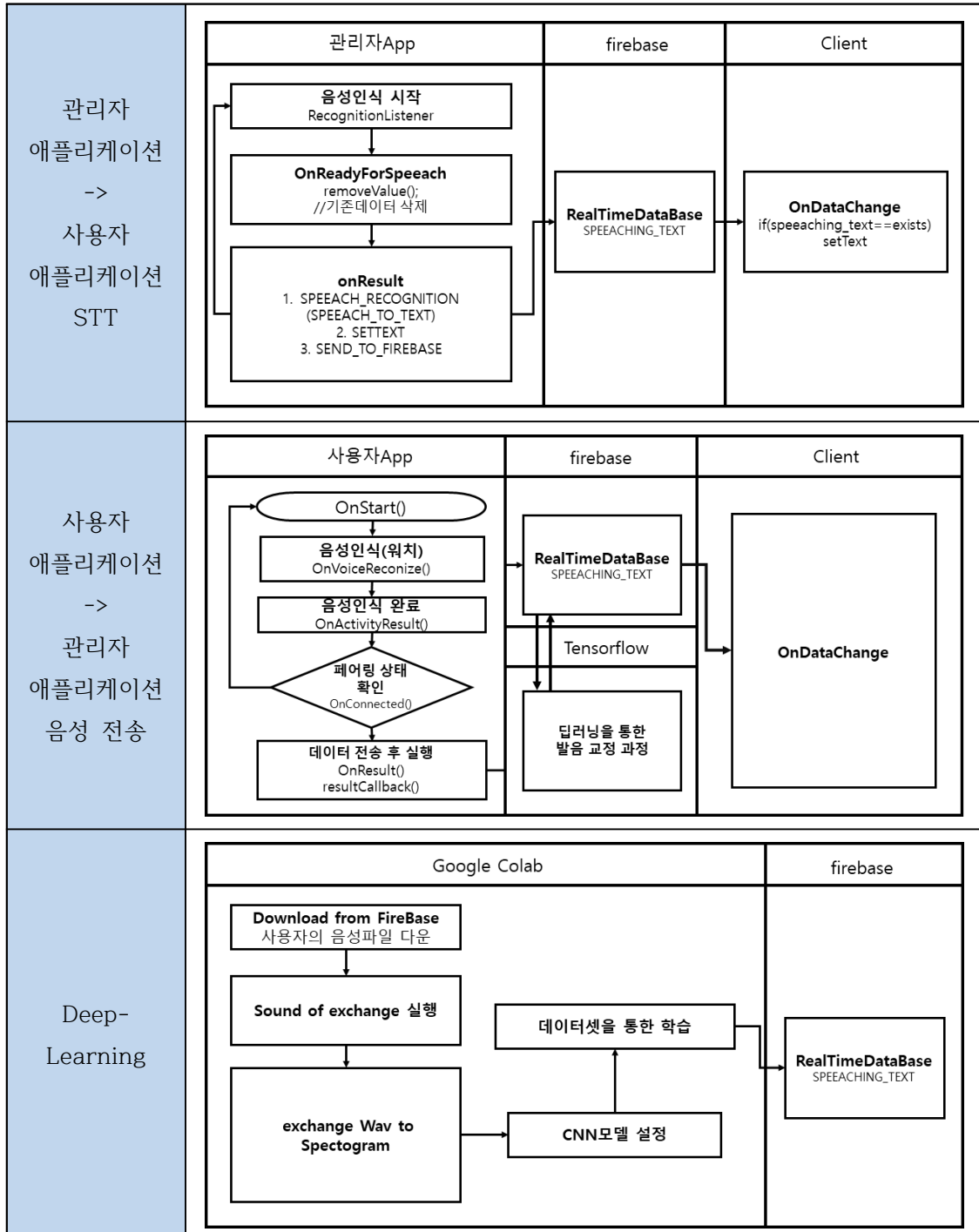
- 데이터베이스 연결은 **Firestore**로 진행
- **Speech Class**는 **SpeechRecognizer**를 이용하여 음성인식 사용, 그 결과값을 **databaseReference.push()**를 통해 **Firestore**에 저장
- **Firestore**는 사용자의 발음을 교정하고, 관리자의 음성을 관리하는 데이터베이스



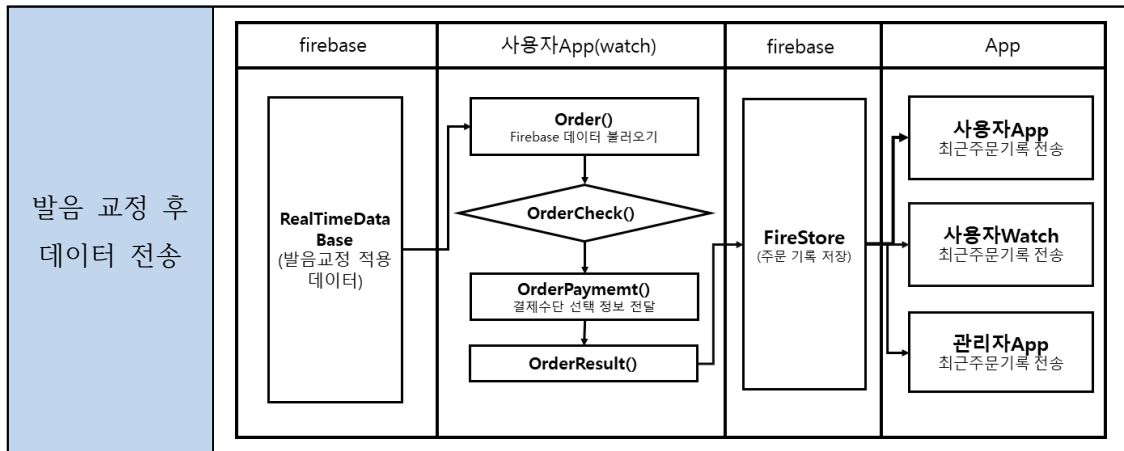
- 로그인과 로그아웃은 모두 **Firestore**에서 관리
- 로그인은 구글 ID와 이메일 ID로 가능



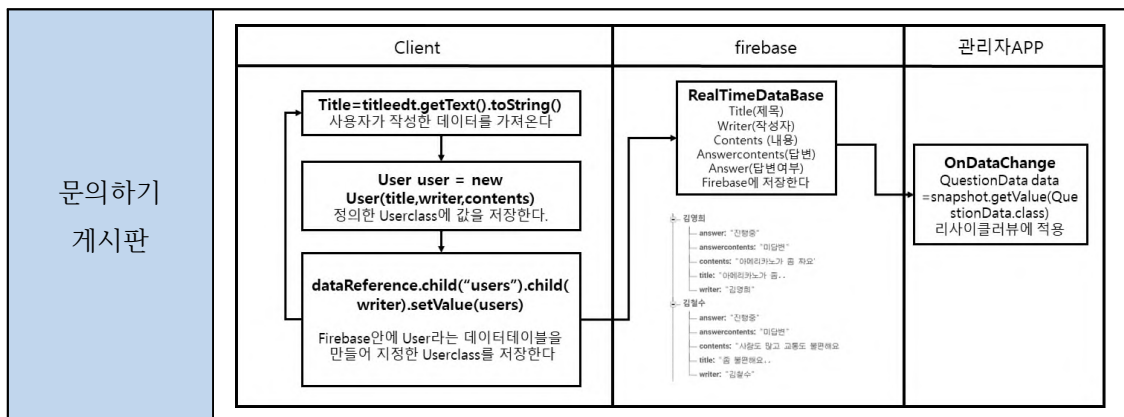
- 사용자의 주문하기와 관리자의 주문 받기
- 관리자의 음성 인식은 Google API 사용
- 사용자의 발음 교정은 CNN 알고리즘 사용



- 사용자와 관리자의 주문확인
- RealTimeDataBase를 사용하여 Firebase에 저장된 데이터로 주문 확인 가능



- 사용자와 관리자의 문의 게시판
- 사용자 애플리케이션에서 작성하여 Firebase를 통해 관리자 애플리케이션에서도 확인 가능



답리닝 상세 설계

- `firebase_admin`이라는 패키지를 통해 앱과 연결된 firebase에 접속

파이썬에서 firebase 연결	<pre> import firebase_admin from firebase_admin import credentials from firebase_admin import storage from firebase_admin import db json_key = 'foruser-86c2a-firebase-adminsdk-o9ghu-8e76c66a0c.json' db_url = 'https://foruser-86c2a.firebaseio.com', bk_name = 'foruser-86c2a.appspot.com' fb_dir = './firebase' </pre>
-------------------------	--

- firebase storage에 저장된 사용자의 음성 wav 파일을 디렉토리에 저장

firebase에서 가져온 wav 파일 디렉토리에 저장	<pre> def download_from_fb(uid, dname): dir_path = uid + "/" + dname blobs = bucket.list_blobs(prefix=dir_path, delimiter=None) for blob in blobs: blob=bucket.blob(blob.name) fname = blob.name.split('/')[2] fdir = fb_dir + '/kDdNvE4mcfQsdXVmUrtJ9TAMUZm2' if not os.path.isdir(fdir): os.mkdir(fdir) fdir = fb_dir + '/' + 'kDdNvE4mcfQsdXVmUrtJ9TAMUZm2' + '/' + dname os.mkdir(fdir) fpath = fdir + '/' +fname blob.download_to_filename(fpath) </pre>
--	---

- firebase에서 가져 온 wav 파일을 이미지 파일로 변환

wav에서 이미지 파일로 변환	<pre> def convert_wav_file(uid, dname): print(uid) print(dname) os.system('wts.sh firebase/%s/%s' % (uid, dname)) </pre>
------------------------	--

- 파이썬 내에서 사용 불가능한 sox api를 shell script를 이용하여 사용
- sox api를 통해 wav파일을 이미지 파일로 변환

Shell script	<pre>#!/bin/bash for file in ./\$1/*.wav do outfile=\${file%.*} sox "\${outfile}".wav -n spectrogram -x 100 -y 100 -r -o "\${outfile}".png done</pre>
--------------	---

- 학습 할 데이터 구분을 위해 라벨링 작업

라벨링 통해 데이터 구분	<pre>train_path_label = [] for (path, dir, files) in os.walk(train_dir): for filename in files: ext = os.path.splitext(filename)[-1] if ext.lower() == '.png': path_file = "%s/%s" % (path, filename) label = filename.split('_')[0] # print(path_file, " ", label) train_path_label.append((path_file, label))</pre>
------------------	---

- 라벨링 작업 후 학습을 통해 데이터 생성

학습 데이터 생성	<pre>for i in range(l_times + 1): batch_X, batch_Y = next_batch(spect_test_imgs, spect_test_labels, i, b_size) a,c, summary = sess.run([accuracy, cross_entropy, merged_summary_op], feed_dict={X: batch_X, Y_: batch_Y, dkeep: 1.0, step: i}) print("training : ", i, ' accuracy = ', '{:7.4f}'.format(a), ' loss = ', c) sess.run(train_step, {X: batch_X, Y_: batch_Y, dkeep: 0.75, step: i}) model_path = './' + fb_path + 'model/sred_model' if i % 100 == 0: saver.save(sess, model_path, global_step=i)</pre>
--------------	--

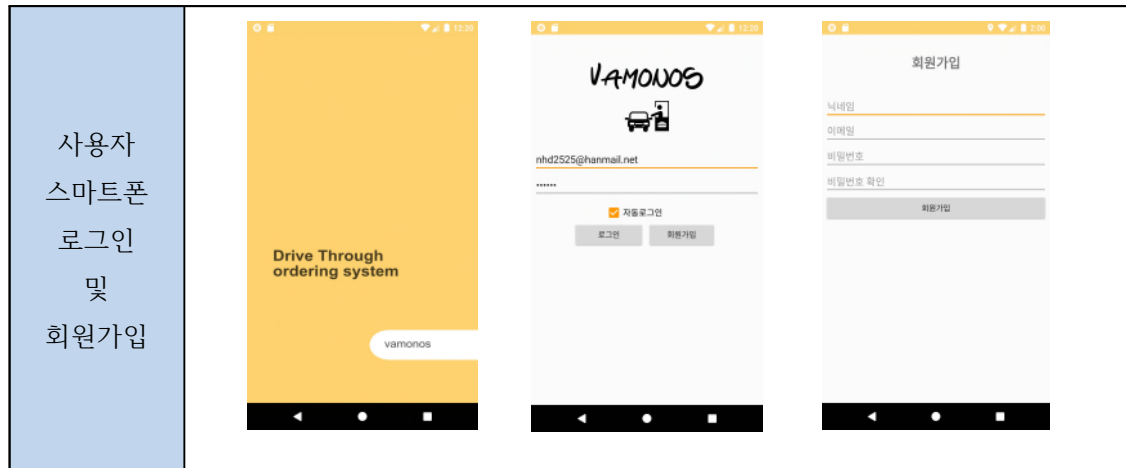
- 학습한 데이터로 예측 모델 생성

<p>예측 모델 생성</p>	<pre>def predict_spect(uid): input_image_name = "./firebase/-M5yk1BDXigI44Jaa7Xd/using/using.png" model_path = './firebase/-M5yk1BDXigI44Jaa7Xd/model/' model_name = './firebase/-M5yk1BDXigI44Jaa7Xd/model/sred_model-500.meta' saver = tf.train.import_meta_graph(model_name) with tf.Session() as sess: saver.restore(sess, tf.train.latest_checkpoint(model_path)) graph = tf.get_default_graph() X = graph.get_tensor_by_name('X:0') Y = graph.get_tensor_by_name('Y:0') dkeep = graph.get_tensor_by_name('dkeep:0') img_ndarr = read_one_spect_png(input_image_name) predict_result = sess.run(Y, feed_dict={X: img_ndarr, dkeep: 1.0}) prediction_list = predict_result[0].tolist() print('') for index, score in enumerate(prediction_list): print('%s %s %0.3f' % (index, label_list[index], score)) max_score_index = prediction_list.index(max(prediction_list)) print(max_score_index) print('-----') print('prediction = %s' % (label_list[max_score_index])) return label_list[max_score_index]</pre>
---------------------	---

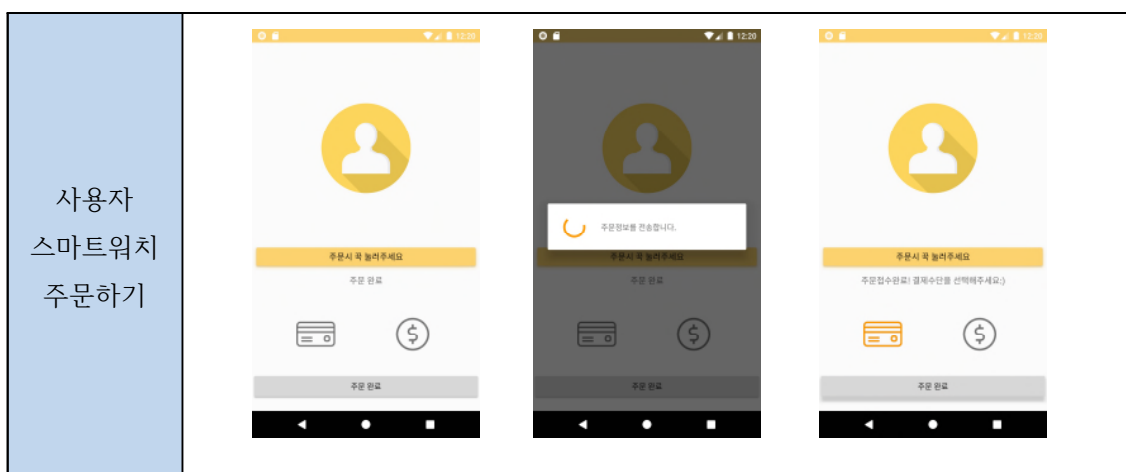
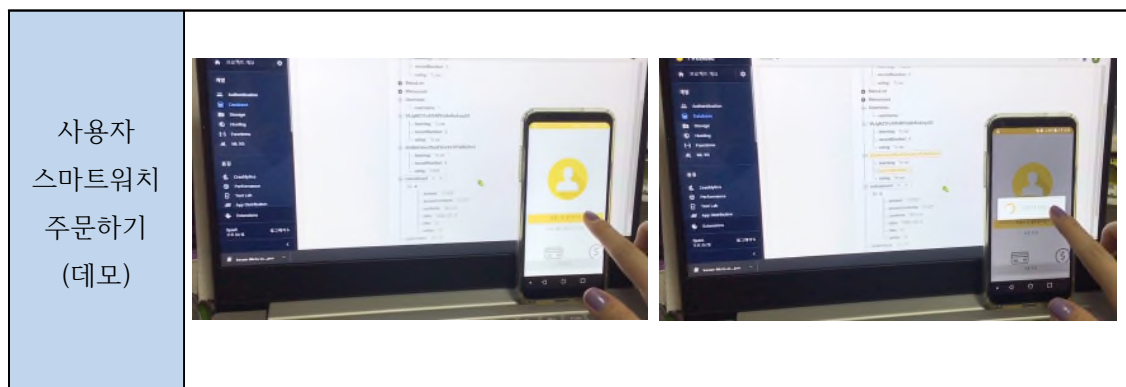
5. Prototype 구현

사용자 애플리케이션 프로토타입(3.0)

- 사용자 로그인과 회원가입 가능



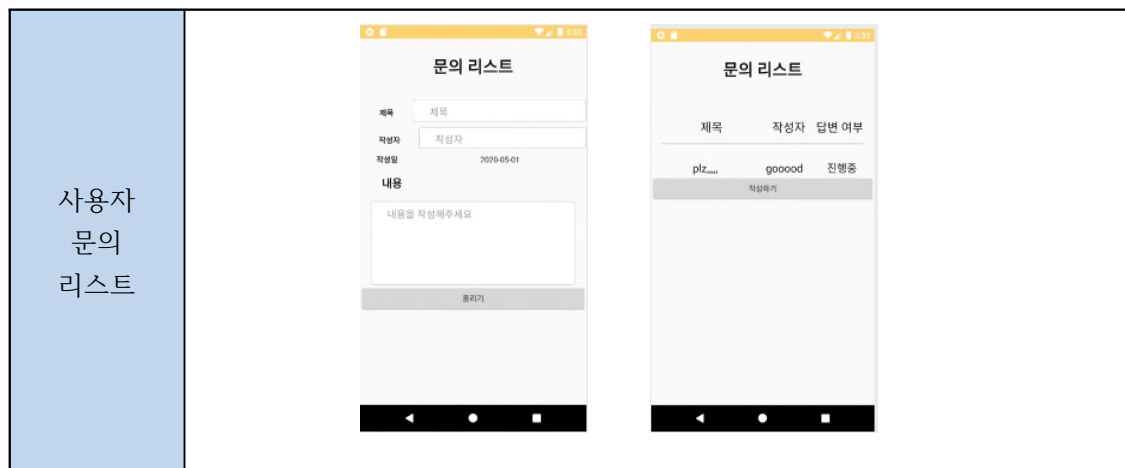
- 사용자가 음성 녹음을 하면 파이어베이스 실시간 데이터베이스의 숫자가 바뀌고, 음성 녹음 파일은 파이어베이스 스토리지에 저장



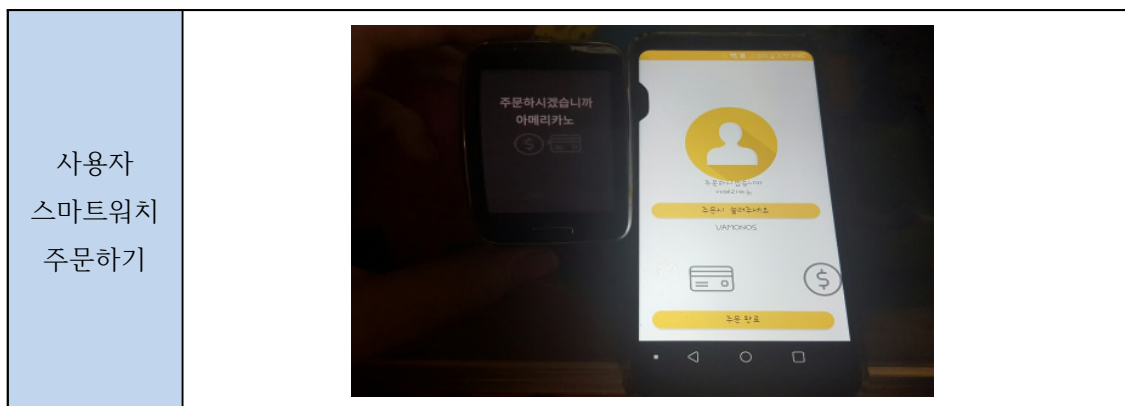
- 주문 할 메뉴 리스트 확인과 주문한 메뉴 확인 가능



- 사용자가 문의하고자 하는 글을 작성 가능

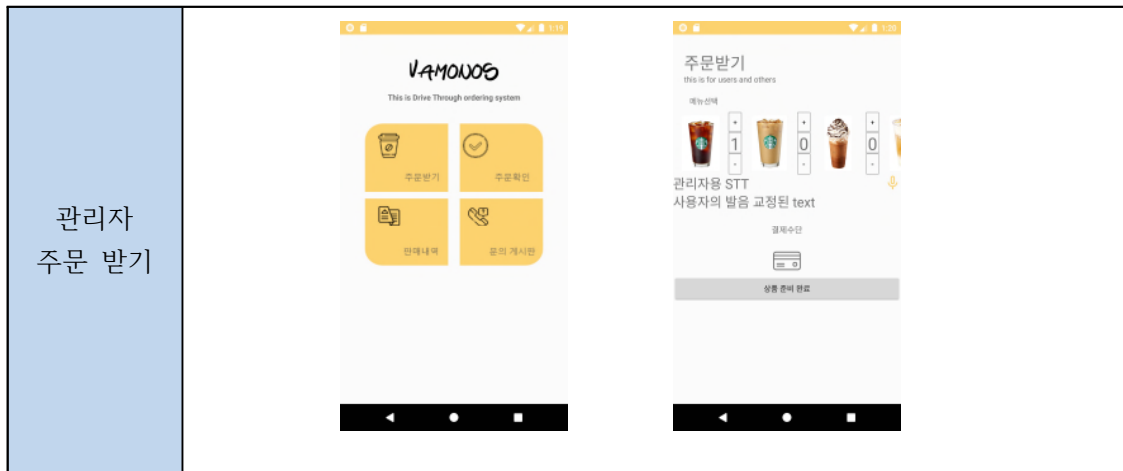


- 사용자 스마트워치에 음성 데이터 출력
- 주문한 제품이 나왔을 때, 알림 기능



관리자 애플리케이션 프로토타입(3.0)

- 관리자의 주문 받기에서는 사용자가 선택한 결제 수단 출력과 음료 수량은 직접 선택



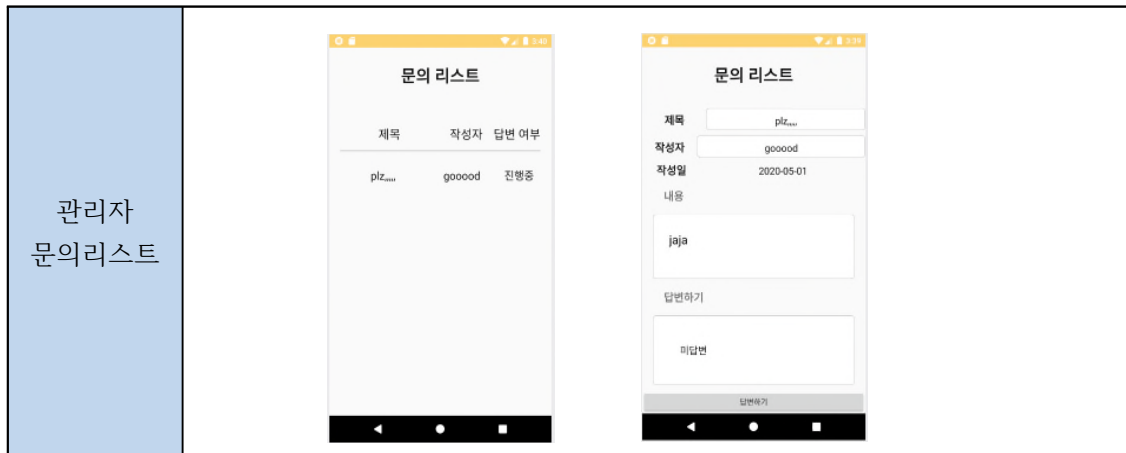
- 관리자의 주문 받기에서는 사용자가 선택한 결제 수단 출력과
- 사용자가 원하는 음료 수량은 직접 선택



- 주문 받기에서 선택한 음료들을 준비 완료를 통해 제거



- 사용자가 작성한 문의 리스트 확인 가능
- 관리자의 답변을 통해 사용자 앱에서 답변 확인 가능



답리닝 프로토타입(3.0)

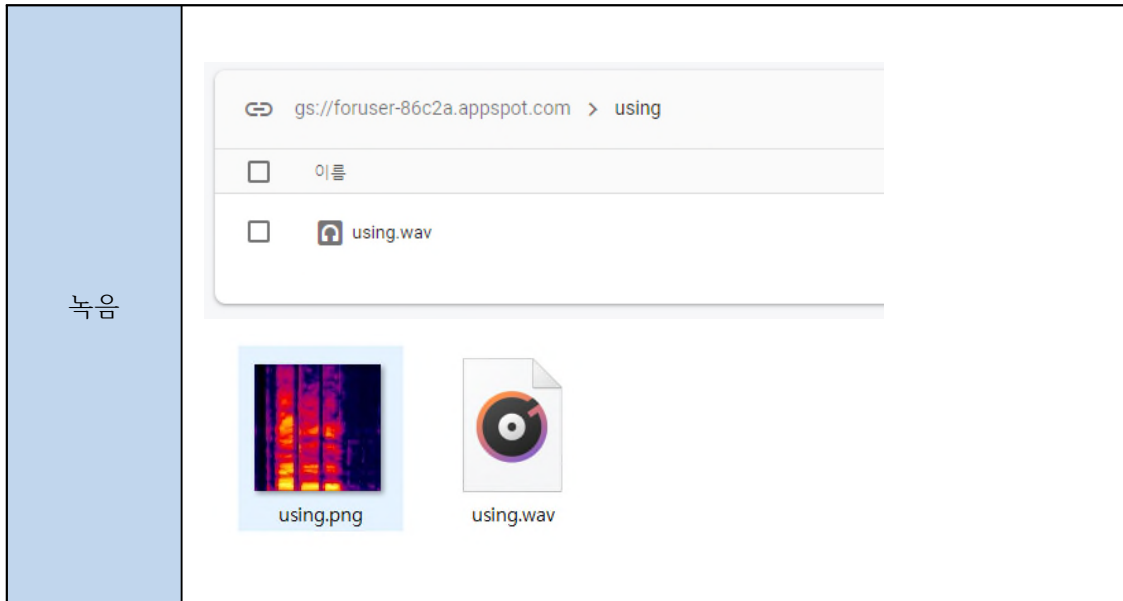
- 파이썬에서 CNN모델로 음성 학습



6. 시험/테스트 결과

딥러닝 테스트




- Application에서 녹음된 파일을 firebase storage 에 업로드
- python server에서 firebase storage에서 다운로드하여 디렉토리에 저장 후 spectrogram으로 변환



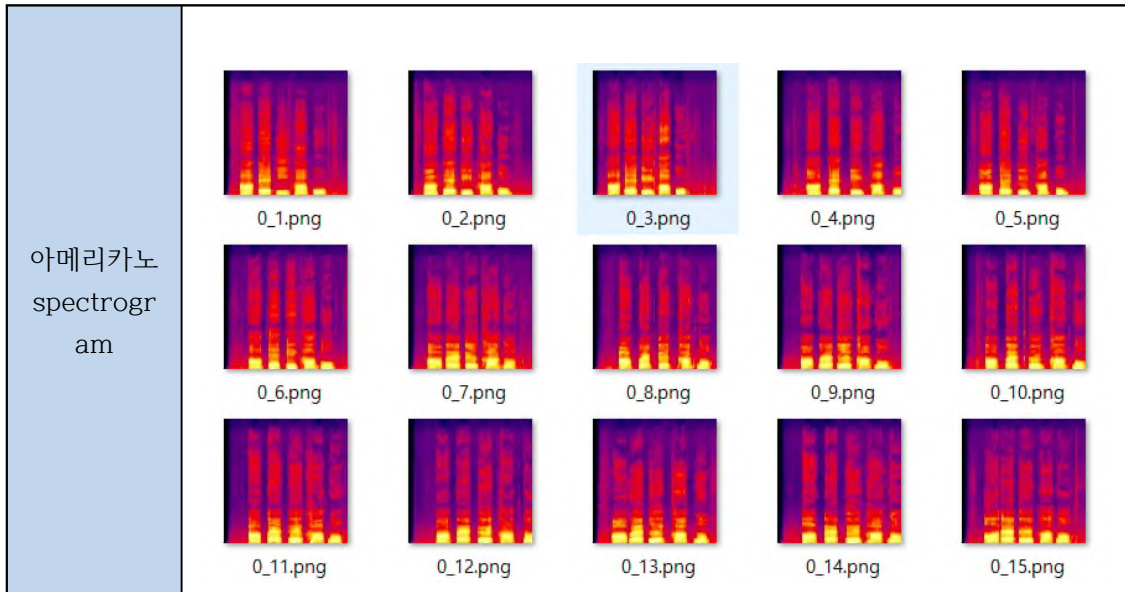
라벨링 테스트

- 파일명을 기준으로 라벨링 하여 학습을 통해 예측을 위한 메타 데이터 생성

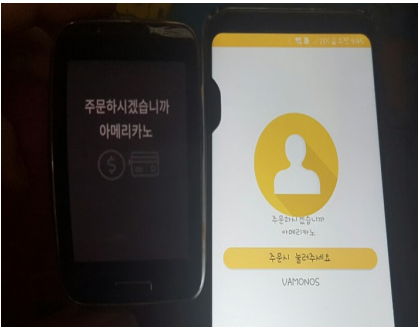


메타 데이터 생성	 vamonos_model-500.data-00000-of-000...	2020-06-27 오후 12:58
	 vamonos_model-500.index	2020-06-27 오후 12:58
	 vamonos_model-500.meta	2020-06-27 오후 12:58

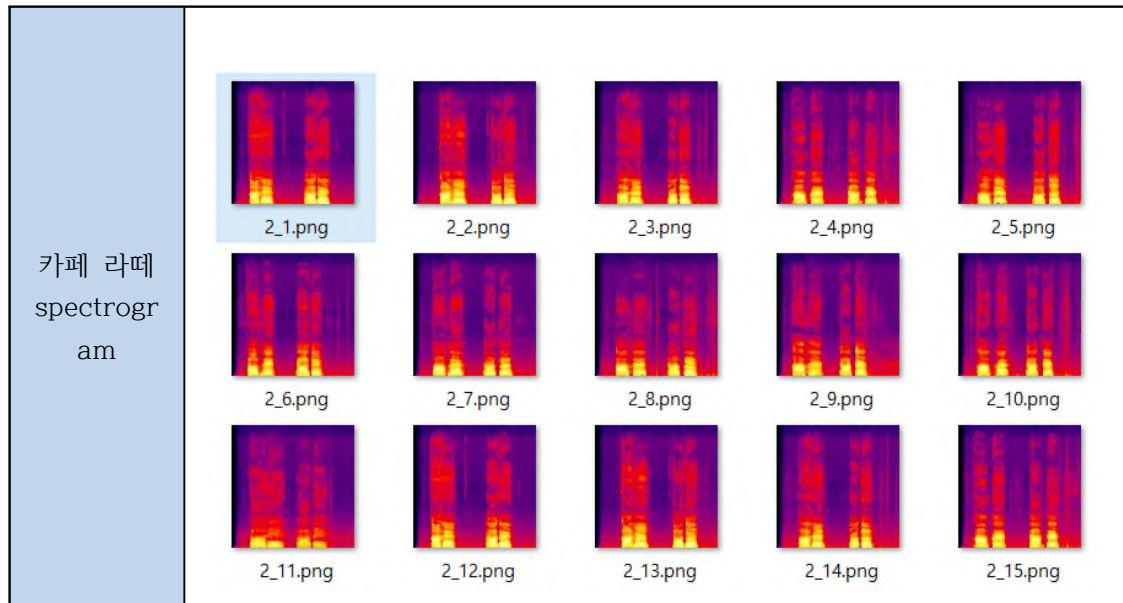
- 아메리카노 녹음 후 spectrogram으로 변환 한 결과



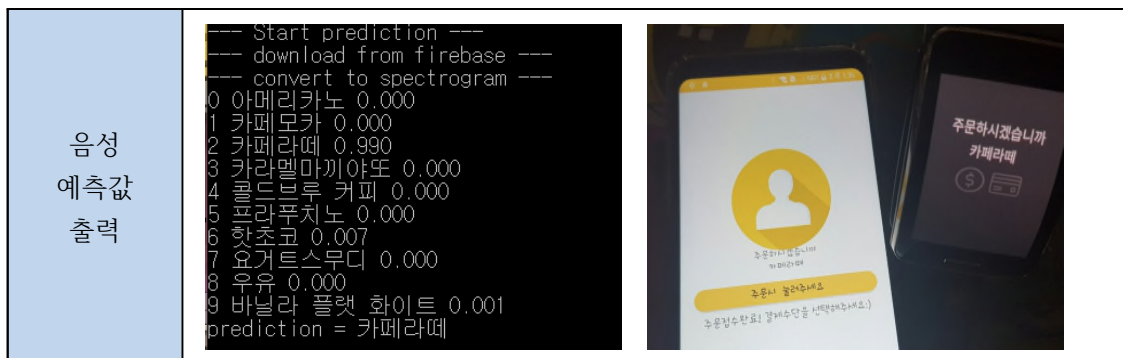
- 아메리카노를 테스트 했을 시, 99.4%의 정확도 출력

아메리카노 예측	<pre> 0 아메리카노 0.994 1 카페모카 0.000 2 카페라떼 0.000 3 카라멜마끼아또 0.001 4 콜드브루 커피 0.000 5 프라푸치노 0.001 6 핫초코 0.000 7 요거트스무디 0.000 8 우유 0.000 9 바닐라 플랫 화이트 0.003 prediction = 아메리카노 </pre>	
-------------	--	--

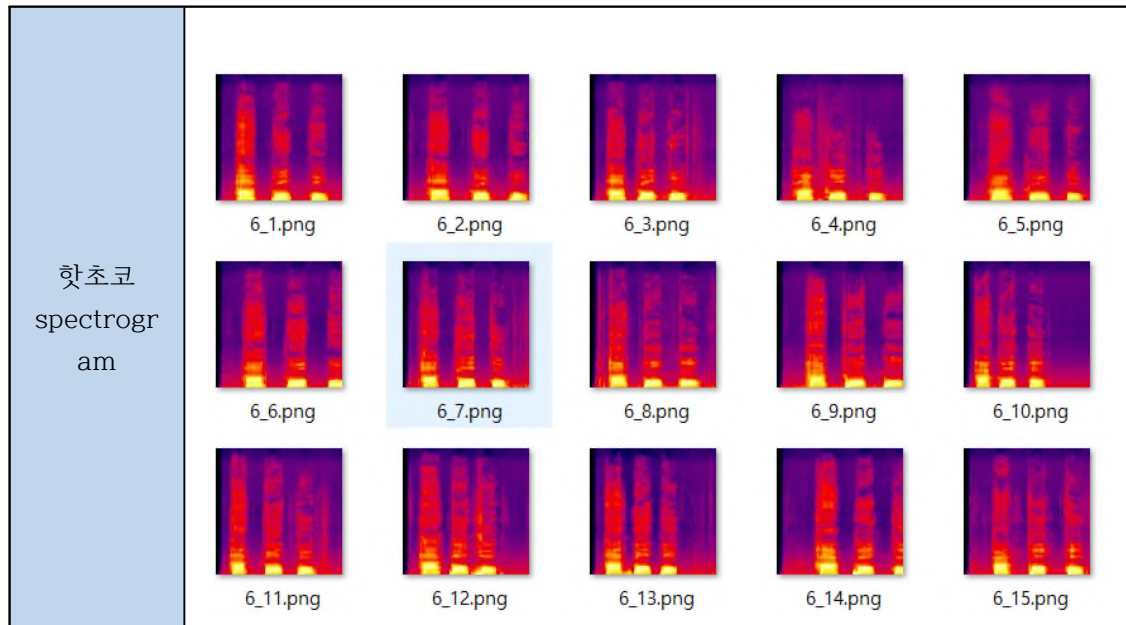
- 카페 라떼 녹음 후 spectrogram으로 변환 한 결과



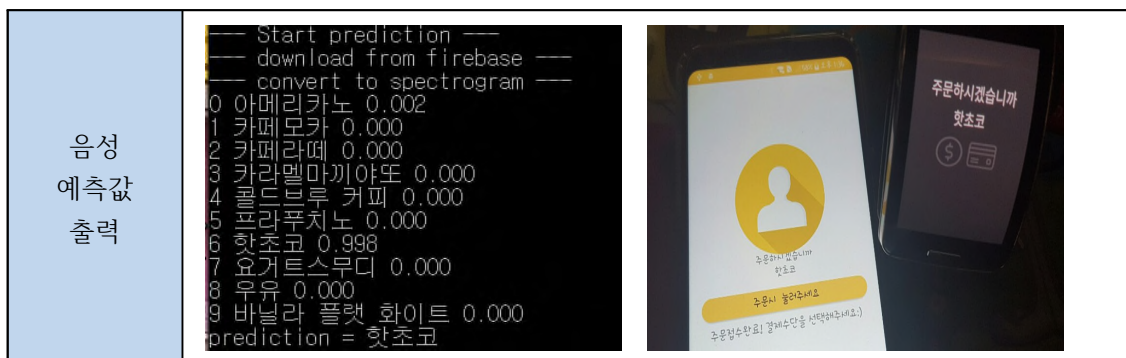
- 카페 라떼를 테스트 했을 시, 99%의 정확도 출력



- 핫초코 녹음 후 spectrogram으로 변환 한 결과



- 핫초코를 테스트 했을 시, 99.8%의 정확도 출력



Accuracy, Loss 측정

- 학습을 통해 정확도와 손실 값 측정
- 초반 학습은 0.4065 , 0.5122 수준의 정확도를 보이며, loss의 값은 상당히 큼

초반 학습	training : 1	accuracy = 0.3750	loss = 3249.1558
	training : 2	accuracy = 0.4062	loss = 1308.9009
	training : 3	accuracy = 0.4688	loss = 171.49625
	training : 4	accuracy = 0.4062	loss = 592.94183
	training : 5	accuracy = 0.3438	loss = 522.15546
	training : 6	accuracy = 0.0938	loss = 433.77313
	training : 7	accuracy = 0.1562	loss = 279.57812
	training : 8	accuracy = 0.0938	loss = 203.90692
	training : 9	accuracy = 0.4688	loss = 170.28143
	training : 10	accuracy = 0.3750	loss = 207.80368
	test* : 10	accuracy = 0.4065	loss = 210.87633
	training : 11	accuracy = 0.4062	loss = 204.27649
	training : 12	accuracy = 0.4688	loss = 153.99573
	training : 13	accuracy = 0.4062	loss = 155.04626
	training : 14	accuracy = 0.4375	loss = 162.03366
	training : 15	accuracy = 0.5625	loss = 132.62791
	training : 16	accuracy = 0.4375	loss = 136.86748
	training : 17	accuracy = 0.4688	loss = 138.83075
	training : 18	accuracy = 0.5625	loss = 120.66698
	training : 19	accuracy = 0.4062	loss = 130.93802
	training : 20	accuracy = 0.5000	loss = 127.231575
	test* : 20	accuracy = 0.5122	loss = 134.79301

- 500번의 학습 후에 정확도는 0.9024 정도의 수준으로 측정

학습 후	training : 481	accuracy = 0.9375	loss = 11.525591
	training : 482	accuracy = 0.9062	loss = 26.239431
	training : 483	accuracy = 0.9375	loss = 18.103119
	training : 484	accuracy = 0.9375	loss = 11.370681
	training : 485	accuracy = 0.9062	loss = 26.320251
	training : 486	accuracy = 0.9375	loss = 17.95859
	training : 487	accuracy = 0.9375	loss = 11.292225
	training : 488	accuracy = 0.9062	loss = 26.456945
	training : 489	accuracy = 0.9375	loss = 17.718704
	training : 490	accuracy = 0.9375	loss = 11.27639
	test* : 490	accuracy = 0.9024	loss = 25.24131
	training : 491	accuracy = 0.9062	loss = 26.575745
	training : 492	accuracy = 0.9375	loss = 17.25733
	training : 493	accuracy = 0.9375	loss = 11.373822
	training : 494	accuracy = 0.9062	loss = 26.53873
	training : 495	accuracy = 0.9375	loss = 16.954336
	training : 496	accuracy = 0.9375	loss = 11.373445
	training : 497	accuracy = 0.9062	loss = 26.603615
	training : 498	accuracy = 0.9375	loss = 16.786224
	training : 499	accuracy = 0.9375	loss = 11.405806
	training : 500	accuracy = 0.9062	loss = 26.603565
	test* : 500	accuracy = 0.9024	loss = 24.199339

7. Coding & Demo

음성 데이터 딥러닝

- 입력 데이터는 음성 데이터를 입력받아 이미지 형태의 spectrogram으로 변환된, 100*100사이즈로의 이미지 입력
- 출력 데이터는 아메리카노, 카페라떼 등 메뉴로 구성된 라벨 10개로 분류

입력 데이터, 출력 데이터 설정	<pre> X = tf.placeholder(tf.float32, [None, 100, 100, 3], name='X') # correct answers will go here Y_ = tf.placeholder(tf.float32, [None, 10], name='Y_') step = tf.placeholder(tf.int32) dkeep = tf.placeholder(tf.float32, name='dkeep') </pre>
-------------------------------	---

- 전체 5개의 계층으로 구성되어 있고, 가중치(W)와 편향(b) 설정
- 각 계층은 convolutional 3계층과 Affine-ReLU로 이루어진 완전 연결층, 그리고 분류를 할 마지막 Affine-softmax 계층으로 구성

가중치, 편향 설정	<pre> W1 = tf.Variable(tf.truncated_normal([5, 5, 3, K], stddev=0.1)) B1 = tf.Variable(tf.ones([K]) / 10) W2 = tf.Variable(tf.truncated_normal([5, 5, K, L], stddev=0.1)) B2 = tf.Variable(tf.ones([L]) / 10) W3 = tf.Variable(tf.truncated_normal([4, 4, L, M], stddev=0.1)) B3 = tf.Variable(tf.ones([M]) / 10) W4 = tf.Variable(tf.truncated_normal([13 * 13 * M, N], stddev=0.1)) B4 = tf.Variable(tf.ones([N]) / 10) W5 = tf.Variable(tf.truncated_normal([N, 10], stddev=0.1)) B5 = tf.Variable(tf.ones([10]) / 10) </pre>
---------------	--

- 활성화 함수는 Relu 사용
- 변화하는 가중치와 편향에 따라 Relu 함수 적용
- 오버피팅 방지를 위해 dropout 적용, dkeep은 75%로 설정

모델 구성	<pre> stride = 1 Y1 = tf.nn.relu(tf.nn.conv2d(X, W1, strides=[1, stride, stride, 1], padding='SAME') + B1) # conv 100 >> 100 Y1d = tf.nn.dropout(Y1, dkeep) stride = 2 Y2 = tf.nn.relu(tf.nn.conv2d(Y1d, W2, strides=[1, stride, stride, 1], padding='SAME') + B2) # conv 100 >> 50 Y2d = tf.nn.dropout(Y2, dkeep) </pre>
-------	--

	<pre> Y2p = tf.nn.max_pool(Y2d, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME') # pool 50 >> 26 stride = 2 Y3 = tf.nn.relu(tf.nn.conv2d(Y2p, W3, strides=[1, stride, stride, 1], padding='SAME') + B3) # conv 26 >> 13 Y3d = tf.nn.dropout(Y3, dkeep) YY = tf.reshape(Y3d, shape=[-1, 13 * 13 * M]) # 최종 이미지 크기에 맞춰 재설정 Y4 = tf.nn.relu(tf.matmul(YY, W4) + B4) Ylogits = tf.matmul(Y4, W5) + B5 Y = tf.nn.softmax(Ylogits, name='Y') </pre>
--	---

- 학습률은 0.0001으로 설정
- 빠르게 학습되어 최적의 장소에 도착하지 못하는 것을 방지하기 위해, 학습 속도를 줄이는 exponential decay 사용
- 하이퍼 파라미터 최적화를 위해 AdamOptimizer로 학습 진행

학습률 설정	<pre> lr = 0.0001 + tf.train.exponential_decay(0.003, step, l_times, 1 / math.e) train_step = tf.train.AdamOptimizer(lr).minimize(cross_entropy) </pre>
-----------	---

- tf.global_variables_initializer 함수로 초기화 진행
- 배치 사이즈를 지정하여 학습
- 매 iteration마다 학습 정확도와 손실 출력, 10번에 한번씩 test를 통해 정확도와 손실 출력

학습 진행	<pre> init = tf.global_variables_initializer() sess = tf.Session() sess.run(init) ##시작 부분 for i in range(l_times + 1): batch_X, batch_Y = next_batch(spect_test_imgs, spect_test_labels, i, b_size) a, c, summary = sess.run([accuracy, cross_entropy, merged_summary_op], feed_dict={X: batch_X, Y_: batch_Y, dkeep: </pre>
-------	--

```

1.0, step: i})
print("training : ", i, ' accuracy = ', '{:7.4f}'.format(a), ' loss = ', c)
summary_writer.add_summary(summary, i)

if i % 10 == 0:
    a, c = sess.run([accuracy, cross_entropy], feed_dict={X:
spect_test_imgs, Y_: spect_test_labels, dkeep: 1.0})
    print("test* : ", i, ' accuracy = ', '{:7.4f}'.format(a), ' loss = ', c)

sess.run(train_step, {X: batch_X, Y_: batch_Y, dkeep: 0.75, step: i})

model_path = './' + fb_path + 'model/sred_model'
if i % 100 == 0:
    saver.save(sess, model_path, global_step=i)

```

Ⅲ. 결론

1.연구 결과

본 연구는 청력의 손실로 인해 청각장애인의 대부분이 겪고 있는 의사소통의 어려움을 딥러닝 분석을 통해 해소될 수 있도록 구현하였다.

또한 스마트워치를 통해 앱을 사용하고, 알림 기능을 통해 주문을 확인 할 수 있음으로 청각장애인의 웨어러블 기기의 활용도를 증진시켰다.

현재는 딥러닝 학습 시간의 문제로 분류할 수 있는 라벨은 10개로 지정하여 청각장애인이 주문할 수 있는 메뉴의 한계가 있다. 그러나 학습을 통해 메뉴와 문장의 데이터셋을 추가시켜 충분히 활용도를 높힐 수 있다.

앞으로 문장을 추가시키게 되면 단순한 메뉴를 주문하는 것에서 원활한 소통을 가능하게 할 수 있는 발전 가능성도 가지고 있다.

또한 웨어러블 기기를 통해 기술적으로 음성 인식 과정은 어려움이 있지만, 기술적인 문제에 대해 더 연구하면 보완할 수 있다.

2.작품제작 소요재료 목록

핸드폰	갤럭시S6(SM-G920S), 갤럭시노트9 (SM-N960N), LG Q6 (LGM-X600)
스마트워치	갤럭시 워치 기어 S(SM-R750K)

참고 자료

- 장애인 지침 모음서 中 모바일 애플리케이션 콘텐츠 접근성 지침 2.0
- 네이버 웹툰 ‘나는 귀머거리다.’
- 홍기형 (2016). 블루투스 기반 청각장애인을 위한 의사소통 앱의 효용성 연구. 보완대체의 사소통연구, 4(2), 59-76
- 김민지, 김예담, 권효은 (2019). 외부 소음 시각화 시스템을 이용한 청각장애인 운전자 보조 장치 개발 연구. Proceedings of KIIT
- 서승욱, 서혜원, 유호진, 선우연, 박성준 (2017). 청각장애인을 위한 딥 러닝 기반 위험 소리 분류 시스템. 한국정보과학회 학술발표
- 허다경, 이병권, 이소정, 남양지, 권준한, 송병섭 (2015). 청각장애인을 위한 소리정보전달 보조기기 개발. 한국재활복지공학회 학술대회 논문집, 18-20
- 청각장애인을 위한 긴급상황 경고 픽토그램 시스템 개발연구 : 재난 상황 및 안전 사고 상황을 중심으로 서울과학기술대학교 손정섭,이은실
- 청각장애인 위험감지·경고시스템 특성화 필요성.에이블뉴스, 2019-12-04
- 이대섭, 2017년 한국수어 사용 실태 조사 연구, 국립국어원, 2017
- 차주민, “합성곱 신경망을 적용한 결함 위치 식별”, 성균관대학교 일반대학원, 2018
- Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet classification with deep convolutional neural networks.", Advances in neural information processing systems, 2012.