

# 눈치보이조

# 8조 눈치보이조

지도교수  
한재일 교수님

팀장  
김도훈

팀원  
곽지훈  
김상원  
명석현  
홍령기  
소가위



# 목차

01 02 03 04

기존 계획

문제 사항

진행 상황

앞으로의 계획

# 1. 기존 계획

1. 이메일과 비밀번호를 이용한 로그인, 산모들이 모두 발급 받는 국민 행복카드를 이용하여 산모 인증
2. Firestore에 각 노선의 시간표별로 문서를 만들어 좌석 예약을 가능하게 함
3. 아두이노를 이용하여 하드웨어를 개발. 압력 센서를 이용하여 사람이 있는지를 판별하여 Firestore에 저장
4. 아두이노를 블루투스 모듈을 사용하여 BLE 비컨으로 만들어 사용자가 접근하면 LED 점등

## 2. 문제 사항

1. 아두이노 기판이 계속 고장나는 문제 발생
2. 아두이노 wemos 보드의 경우 기존 보드와 핀 번호와 함수가 달라 자료 발견의 어려움
3. 하드웨어 개수의 한계와 ODSay API로부터 얻을 수 있는 지하철의 정보로는 각 노선별 시간표마다의 열차를 특정해낼 수가 없음
4. 실제 카드와 산모의 리스트가 없으므로 카드 인증이 불가능



- 1, 2번 문제의 경우 하드웨어를 라즈베리 파이로 교체하여 해결
- 3번 문제의 경우 하드웨어와 직접 연동할 demo firestore document와 고정 값을 저장한 static firestore document로 분리하여 DB 설계.
- 시연 시에는 특정 노선을 지원하는 demo document만을 이용
- 4번 문제의 경우 프로토타입의 한계상 DB에 임의의 이름과 카드번호를 저장하여 비교하는 인증 방식

### 3. 진행 사항 (Application)

The image displays three sequential mobile app screens for a service named '눈치보이조' (Nunchiboyjo). The first screen is the login page, featuring a logo with a car and the text '눈치보이조?' and 'GG'. It includes input fields for '이메일' (Email) and '비밀번호' (Password), and buttons for '로그인' (Login) and '회원가입' (Sign Up). The second screen is the registration page, titled '회원가입', with input fields for '이메일' (Email), '비밀번호' (Password), and '비밀번호 확인' (Confirm Password), and a '회원가입' (Sign Up) button. The third screen is the card creation page, titled '카드번호(- 패고 작성)' (Card Number (- Pego Writing)), with an input field for the card number and a '등록' (Register) button.

- 회원가입과 로그인, 산모 인증이 구현되었습니다.
- 산모 인증은 자체적으로 이루어집니다.
- 현재는 이메일과 비밀번호를 이용한 회원가입만을 지원합니다.

### 3. 진행 사항 (Application)



- 지하철 경로 찾기가 구현 완료 되었습니다.
- Open source인 ODSay API의 지하철 경로검색 조회 레퍼런스를 사용하였습니다.
- 출발역과 도착역을 입력함으로써 경로가 출력됩니다.
- 당장은 해당 레이아웃에 필요한 데이터들을 API로부터 파스만 해놓은 상태지만, 추후 이 데이터들을 기반으로 구체적인 레이아웃을 구축할 예정입니다.

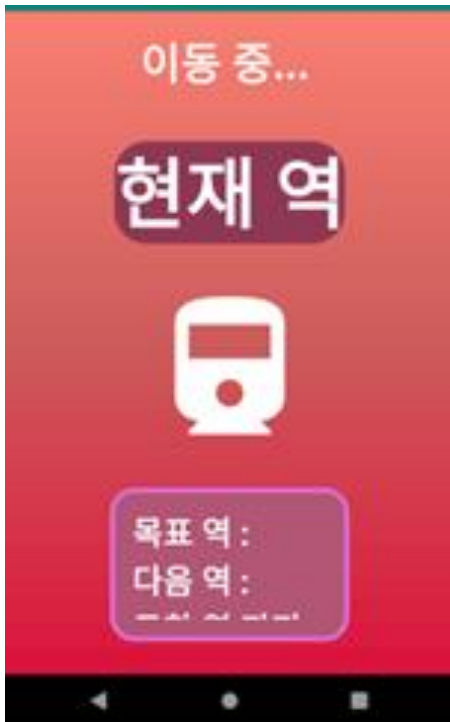
### 3. 진행 사항 (Application)



- 경로를 찾고 나면 하단의 바를 드래그하여 탑승역의 예약 가능한 열차 목록을 볼 수 있는 창을 구성했습니다. 이 창은 현재는 실제 열차 목록을 보여주지는 않습니다.
- 열차를 선택하면 해당 열차의 좌석현황을 볼 수 있습니다. 현재는 대략적인 레이아웃 구성만 해 놓은 상태로, 실제 자리 현황을 불러와 띄우지는 않는 상태입니다.

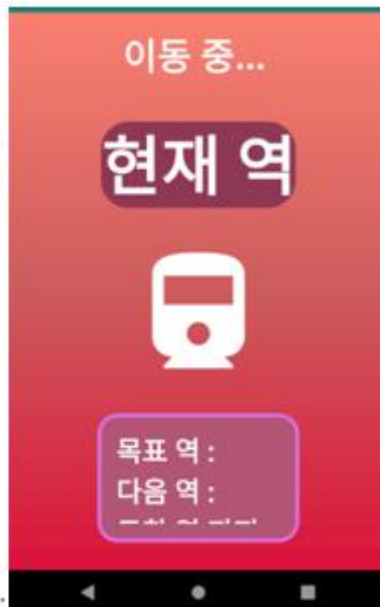


### 3. 진행 사항 (Application)



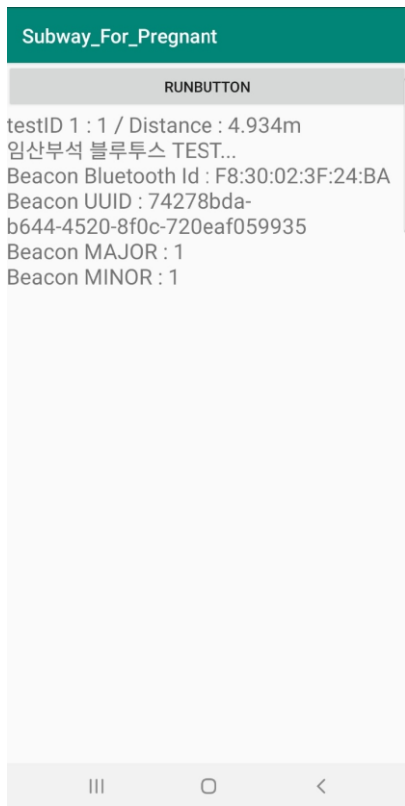
- 좌석 선택 후, 목표 역에 도착할 때까지 띄워지는 대기 화면 레이아웃을 구성했습니다.
- 마찬가지로 실제로 현재 역과 목표 역 등을 나타내지는 않으며, 불러온 데이터를 표시할 위치만 정해두었습니다.

### 3. 진행 사항 (Application)



- 경로를 찾고 나면 하단의 바를 드래그하여 탑승역의 예약 가능한 열차 목록을 볼 수 있는 창을 구성했습니다. 이 창은 현재는 실제 열차 목록을 보여주지는 않습니다.
- 열차를 선택하면 해당 열차의 좌석현황을 볼 수 있습니다. 현재는 대략적인 레이아웃 구성만 해 놓은 상태로, 실제 자리 현황을 불러와 띄우지는 않는 상태입니다.
- 좌석 선택 후, 목표 역에 도착할 때까지 띄워지는 대기 화면 레이아웃을 구성했습니다. 마찬가지로 실제로 현재 역과 목표 역 등을 나타내지는 않으며, 불러온 데이터를 표시할 위치만 정해두었습니다.

### 3. 진행 사항 (Application)

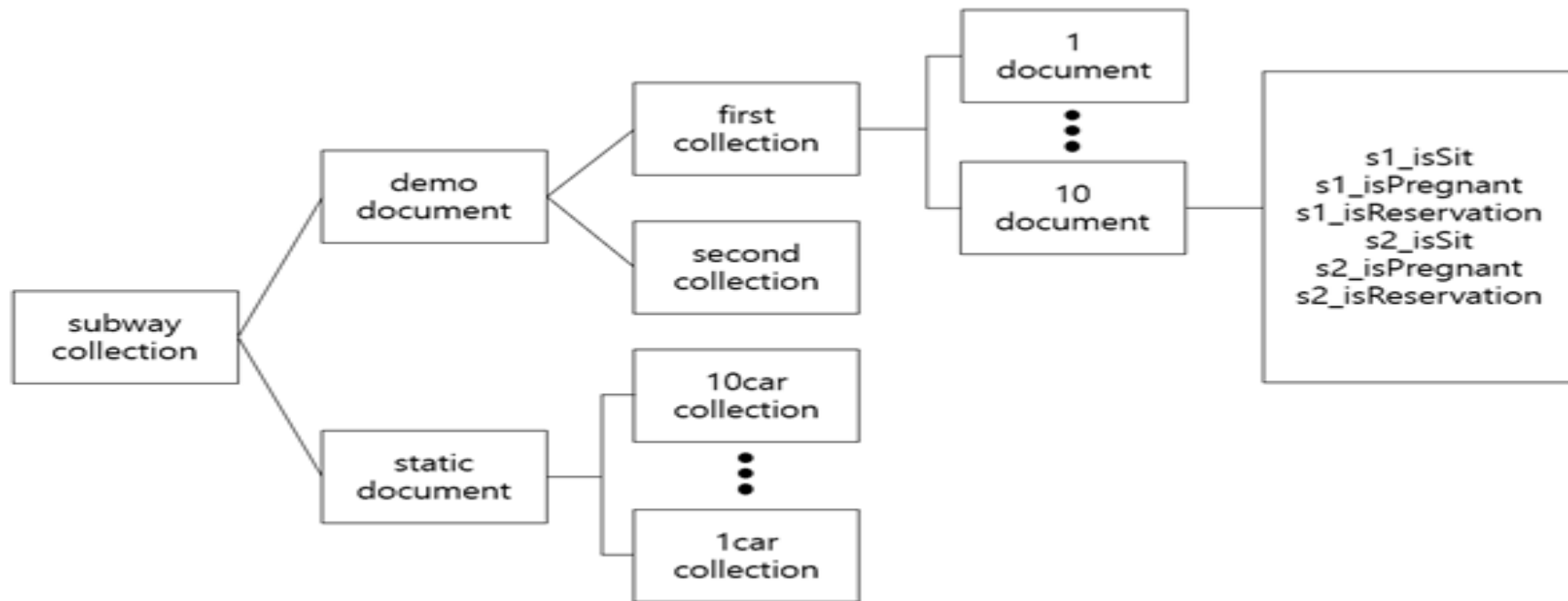


- 어플에서 Beacon을 감지할 수 있도록 하였습니다. 현재는 Beacon에서 전송해주는 Bluetooth Address, UUID, major, minor, TXpower(거리) 값을 받아와서 출력하도록 하였습니다.
- 추후에는 사용자의 스마트폰에서 어플을 실행 시, 자동으로 임산부 좌석의 라즈베리 파이, 즉 Beacon을 감지해 해당 Major, Minor의 Unique한 값을 이용해 일정 거리 이내의 좌석을 자동으로 예약할 수 있게끔 할 것입니다.

### 3. 진행 사항 (Server 및 DB)

- Firebase Authentication을 이용해 회원가입 및 로그인이 진행됩니다.
- Firestore를 통해 데이터를 저장합니다.
- 라즈베리 파이의 센서 값을 MQTT 브로커를 통해 Firestore에 저장하기 위하여 Node.js express 서버를 추가로 구축하였습니다.

### 3. 진행 사항 (Server 및 DB)

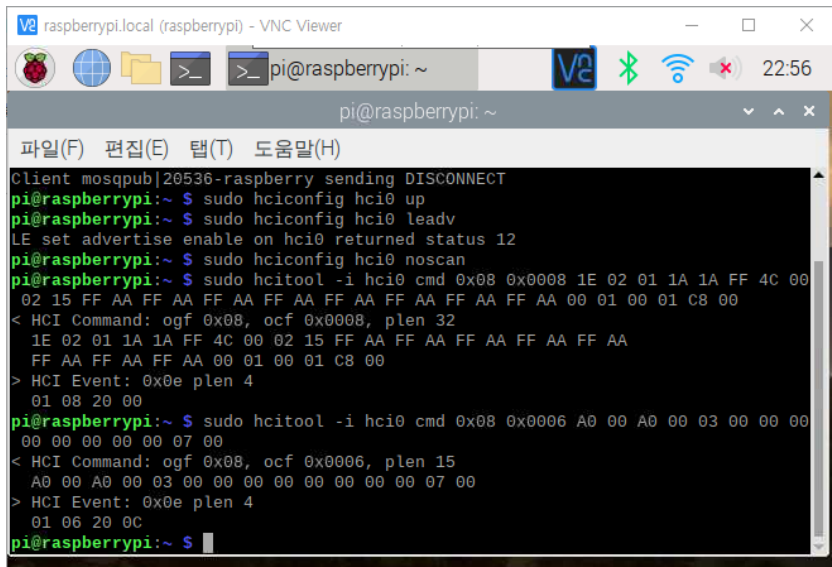


### 3. 진행 사항 (Server 및 DB)

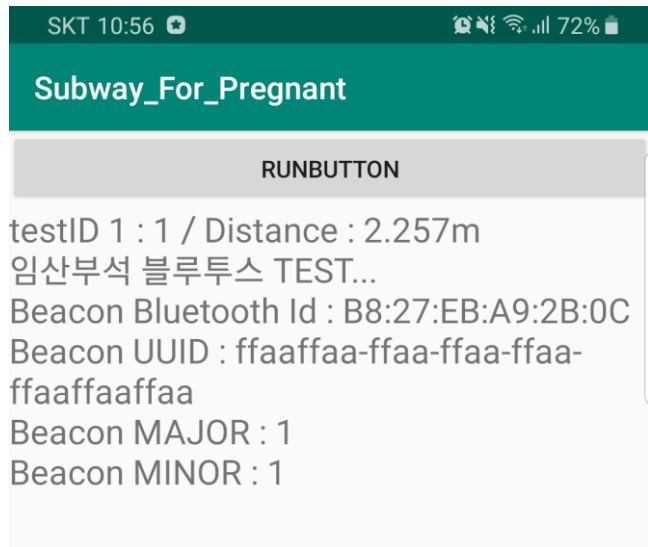
- 최상위 subway collection 아래에 demo용 db인 document와 고정값 db인 static document이 있습니다.
- demo document는 직접 하드웨어와 연동하며 특정 노선에 대해서만 이용합니다. 환승하는 경우를 위해 first, second로 나누었으며 해당 컬렉션 안에는 각 칸의 document와 field가 저장됩니다.
- isSit은 로드셀 센서의 값을 통해 착석 여부를 판별하는 boolean type field입니다.
- isPregnant는 BLE를 통해 저장되는 산모 여부를 판별하는 boolean type field입니다.
- isReservation은 어플을 통해 예약 여부를 판별하는 boolean type field입니다.
- static document 하위 DB 또한 같습니다.

### 3. 진행 사항 (Hardware)

- BLE Beacon을 세팅해서 커스텀한 UUID, Major, Minor 값으로 브로드캐스팅 하고 저희가 만든 어플리케이션에서 Major 값을 필터링 해서 해당 라즈베리 파이 만 검색되는 것을 확인했습니다. (Major 0x 00 01 / Minor 0x 00 01)

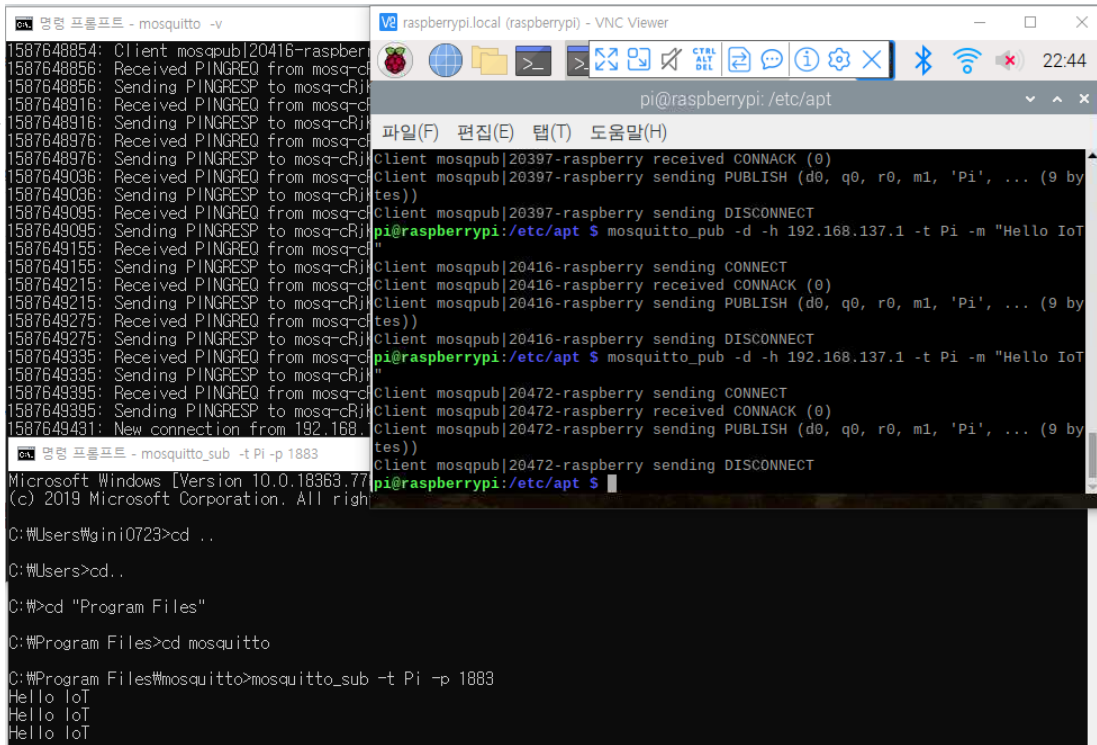


```
Client mosqpub|20536-raspberry sending DISCONNECT
pi@raspberrypi:~ $ sudo hciconfig hci0 up
pi@raspberrypi:~ $ sudo hciconfig hci0 leadv
LE set advertise enable on hci0 returned status 12
pi@raspberrypi:~ $ sudo hciconfig hci0 noscan
pi@raspberrypi:~ $ sudo hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00
02 15 FF AA FF AA FF AA FF AA FF AA FF AA FF AA 00 01 00 01 C8 00
< HCI Command: ogf 0x08, ocf 0x0008, plen 32
1E 02 01 1A 1A FF 4C 00 02 15 FF AA FF AA FF AA FF AA
FF AA FF AA FF AA 00 01 00 01 C8 00
> HCI Event: 0x0e plen 4
01 08 20 00
pi@raspberrypi:~ $ sudo hcitool -i hci0 cmd 0x08 0x0006 A0 00 A0 00 03 00 00 00
00 00 00 00 00 07 00
< HCI Command: ogf 0x08, ocf 0x0006, plen 15
A0 00 A0 00 03 00 00 00 00 00 00 00 00 00 07 00
> HCI Event: 0x0e plen 4
01 06 20 0C
pi@raspberrypi:~ $
```



### 3. 진행 사항 (Hardware)

- 컴퓨터에 MQTT 브로커를 Pi 라는 토픽에 수신하도록 실행하고 라즈베리 에서 “Hello IoT” 라는 메시지를 송신하여 브로커에서 수신함을 확인하였습니다.



```
명령 프롬프트 - mosquitto -v
1587648854: Client mosqpub|20416-raspber
1587648856: Received PINGREQ from mosq-c
1587648856: Sending PINGRESP to mosq-cRj
15876488916: Received PINGREQ from mosq-c
15876488916: Sending PINGRESP to mosq-cRj
15876488976: Received PINGREQ from mosq-c
15876488976: Sending PINGRESP to mosq-cRj
1587649036: Received PINGREQ from mosq-c
1587649036: Sending PINGRESP to mosq-cRj
1587649095: Received PINGREQ from mosq-c
1587649095: Sending PINGRESP to mosq-cRj
1587649155: Received PINGREQ from mosq-c
1587649155: Sending PINGRESP to mosq-cRj
1587649215: Received PINGREQ from mosq-c
1587649215: Sending PINGRESP to mosq-cRj
1587649275: Received PINGREQ from mosq-c
1587649275: Sending PINGRESP to mosq-cRj
1587649335: Received PINGREQ from mosq-c
1587649335: Sending PINGRESP to mosq-cRj
1587649395: Received PINGREQ from mosq-c
1587649395: Sending PINGRESP to mosq-cRj
1587649431: New connection from 192.168.

raspberrypi.local (raspberrypi) - VNC Viewer
pi@raspberrypi: /etc/apt
파일(F) 편집(E) 탭(T) 도움말(H)
Client mosqpub|20397-raspberry received CONNACK (0)
Client mosqpub|20397-raspberry sending PUBLISH (d0, q0, r0, m1, 'Pi', ... (9 by
tes))
Client mosqpub|20397-raspberry sending DISCONNECT
pi@raspberrypi:/etc/apt $ mosquitto_pub -d -h 192.168.137.1 -t Pi -m "Hello IoT"
Client mosqpub|20416-raspberry sending CONNECT
Client mosqpub|20416-raspberry received CONNACK (0)
Client mosqpub|20416-raspberry sending PUBLISH (d0, q0, r0, m1, 'Pi', ... (9 by
tes))
Client mosqpub|20416-raspberry sending DISCONNECT
pi@raspberrypi:/etc/apt $ mosquitto_pub -d -h 192.168.137.1 -t Pi -m "Hello IoT"
Client mosqpub|20472-raspberry sending CONNECT
Client mosqpub|20472-raspberry received CONNACK (0)
Client mosqpub|20472-raspberry sending PUBLISH (d0, q0, r0, m1, 'Pi', ... (9 by
tes))
Client mosqpub|20472-raspberry sending DISCONNECT
pi@raspberrypi:/etc/apt $

명령 프롬프트 - mosquitto_sub -t Pi -p 1883
Microsoft Windows [Version 10.0.18363.77]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\gini0723>cd ..
C:\Users>cd..
C:\>cd "Program Files"
C:\Program Files>cd mosquitto
C:\Program Files\mosquitto>mosquitto_sub -t Pi -p 1883
Hello IoT
Hello IoT
Hello IoT
```



# 3. 진행 사항 demo

APP



Hardware



## 4. 앞으로의 계획

1. 라즈베리 파이와 DB를 모두 연동하여 좌석 현황을 제대로 보여주고 예약을 가능하게 합니다.
2. BLE를 이용하여 산모가 좌석 일정 거리에 접근하게 되면 LED가 점등되고 산모가 앉아있음을 DB에 저장하게 합니다.
3. UI 디자인을 통일 시키고 보기 좋게 업데이트 합니다.
4. 사용자의 편의성을 증가시키기 위한 추가 기능들(빠른 환승 정보, 역무원 호출 등)을 구현합니다.
5. 노약자석의 좌석현황을 볼 수 있게 합니다.

감사합니다.