

BUSSING



Team: SSINGSSING

Member: 201600784 김준영

201402750 임광효

201302247 이경원

201600599 김아연

201600637 김예주

201403263 지승환

〈목 차〉

1. 개요.....	5
1.1 프로젝트 개요.....	5
1.2 프로젝트 목적.....	5
2. 시스템 구성 및 설계.....	6
2.1 전체 시스템 구성도.....	6
2.1.1 시스템 구성도와 흐름.....	6
2.1.2 사용 장치.....	7
2.1.3 개발 환경.....	9
2.2 서버 설계.....	10
2.2.1 서버의 클라이언트 핸들러 (쓰레드).....	10
2.3.1 서버 구동.....	12
2.3 버스 설계.....	13
2.3.1 아두이노 - 라즈베리파이.....	13
2.3.2 블루투스 - 라즈베리파이.....	15
2.3.3 버스 - 서버.....	16
2.4 정류장 설계.....	18
2.4.1 정류장.....	18
2.4.2 서버 - 정류장.....	18
2.5 어플리케이션 설계.....	20
2.5.1 어플리케이션 구성도.....	20
2.5.2 안드로이드 - 서버.....	23
3. 기대효과.....	26
4. 팀원 임무.....	27
5. 부록.....	28

〈그 림 목 차〉

[그림1] 전체 시스템 구성도	6
[그림2] 라즈베리파이	7
[그림3] 아두이노	7
[그림4] 초음파 센서 HC-SR04	8
[그림5] 비콘	8
[그림6] 쓰레드 DB 업데이트	10
[그림7] 메시지 형식	10
[그림8] 운행 중인 버스 id 예시	10
[그림9] 쓰레드 DB 업데이트	11
[그림10] 사용자 요청 메시지 형식	11
[그림11] 사용자 답변 메시지 형식	11
[그림12] 최종 데이터 전송 및 DB 업데이트	11
[그림13] 클라이언트 강제 종료 시 수행	12
[그림14] 서버 구동	12
[그림15] 아두이노 - 라즈베리파이 연결	13
[그림16] 아두이노 인원 측정 함수	14
[그림17] 비콘 탐색	15
[그림18] 로컬 DB 지정 및 포트 초기화	16
[그림19] DB 초기화 후, 서버 DB 업데이트	16
[그림20] 버스 운행 중 상태 업데이트	17
[그림21] 버스 정보 요청	18
[그림22] 메시지 파싱	19
[그림23] 버스 정보 출력	19
[그림24] 어플리케이션 실행화면	20
[그림25] 어플리케이션 흐름도	20
[그림26] 이미지 설정	21
[그림27] 로딩 종료 후 메인화면으로 이동	21
[그림28] 1.5초 지연	21
[그림29] button4	21
[그림30] button6	22
[그림31] time_table class	22
[그림32] 버스 시간표	22
[그림33] button_finish	22
[그림34] 클라이언트 소켓 통신 설정	23
[그림35] 4대의 버스 정보를 출력하기 위한 변수 초기화	23
[그림36] 버스 행선로 구분	24
[그림37] 서버로부터 받아오는 메시지 형식	24
[그림38] 버스 위치에 맞는 리스트	24
[그림39] 수신 메시지 파싱 후 저장하는 쓰레드	25

〈표 목 차〉

[표1] 개발 환경.....	9
[표2] 팀원 임무.....	26

1. 개요

1.1 프로젝트 개요

우리 학교 학생이라면, 강의실까지 뛰어가기도 애매하고, 언제 올지도 모르는 버스를 기다리기도 애매해 곤혹스러웠던 경험이 한번쯤은 존재한다. 먼 강의실의 위치 때문에 일찌감치 정류장에서 버스를 기다린다 하더라도, 앞서 승객들을 태워 만석이 된 버스들을 보내다 보면 어김없이 수업 시간에 늦기 마련이다. 우리 팀은 이러한 문제점을 해결하고자 BUSSING 프로젝트를 시작하게 되었다.

BUSSING 프로젝트는 버스 위치 알림 서비스로써, 버스 클라이언트와 서버가 상호 통신을 통해 자동으로 버스 위치 및 혼잡 정보를 업데이트한다. 사용자는 어플리케이션이나 정류장에 마련된 화면을 통해 실시간으로 버스의 위치를 파악하고, 자신의 목적지까지 가는 교통수단을 빠르게 선택할 것이다.

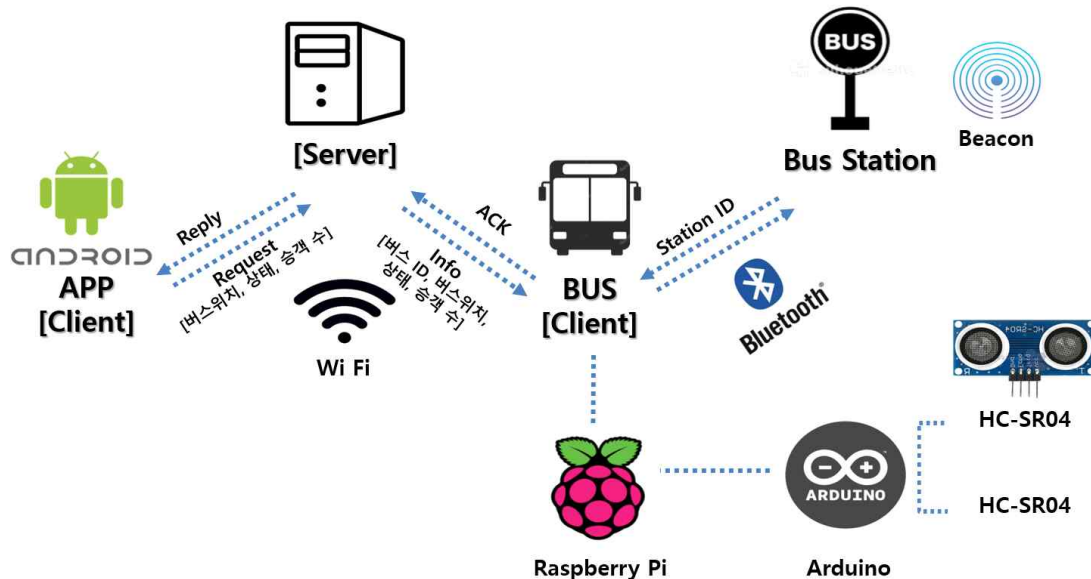
1.2 프로젝트 목적

BUSSING을 사용하게 된다면, 교내에 운영되는 셔틀버스의 위치를 쉽고 빠르게 파악할 수 있고, 막연히 교내 셔틀 버스를 기다리거나, 탑승하지 못하는 셔틀버스를 보내는 상황을 피할 수 있다. 또한 저비용 센서와 장치로 시스템을 구축하여 시중에 존재하는 버스 인원 측정 시스템보다 간단하고 저렴하게 운용할 수 있다.

2. 시스템 구성 및 설계

2.1 전체 시스템 구성도

2.1.1 시스템 구성도와 흐름



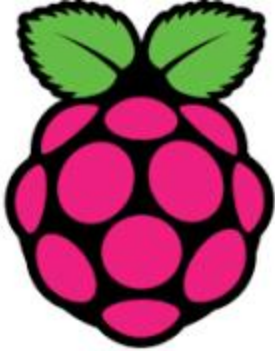

[그림1] 전체 시스템 구성도

사용자가 어플리케이션을 실행하여 버스의 정보를 받기 위해 새로 고침을 하게 되면, 어플리케이션은 서버와 Socket 통신을 통해 고정된 형태의 메시지를 보낸다. 서버는 메시지 형태에 맞는 task를 찾아 정보를 보내주는데, 현재 운행 중인 모든 버스의 정보를 보내준다. 해당 정보를 받으면 사용자의 어플리케이션에서 정보를 알맞게 파싱하여 사용자에게 제공한다.

서버는 DB 운영을 통해 현재 운행 중인 BUS들의 고유 ID기준으로 정보를 관리한다. BUS가 운행을 종료하면 자동으로 DB에서 제거하며, 실시간으로 BUS의 정보가 업데이트 된다.



버스는 아두이노 센서를 사용하여 승객의 수를 세며, 정류장에 있는 비콘과 블루투스 통신을 통해 연결에 성공하면 정차, 끊어지면 출발하는 식으로 버스의 운행 상태를 확인한다. 버스가 출발할 때마다 서버와의 소켓통신으로 버스의 정보를 업데이트 한다.

2.1.2 사용 장치

 <p>Raspberry Pi</p>	
로고	라즈베리 파이 모델

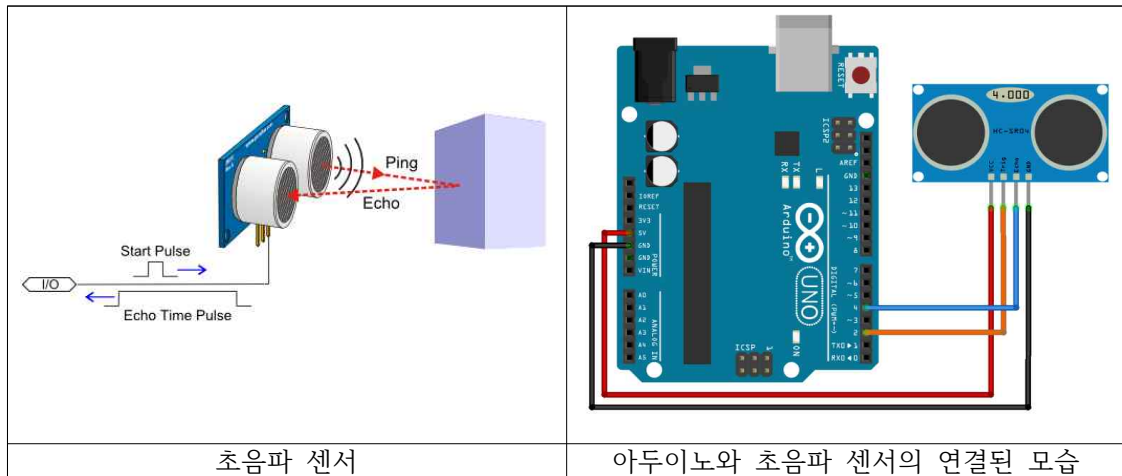
[그림2] 라즈베리파이

라즈베리파이는 영국 라즈베리 파이(Raspberry Pi) 재단에서 교육용 프로젝트의 일환으로 개발된 초소형/초저가 PC이다. 버스에 부착된 라즈베리파이는 초음파 센서와 연결된 아두이노를 연결하여 사용한다.

	
아두이노 로고	모델

[그림3] 아두이노

아두이노는 오픈소스를 기반으로한 마이크로컴퓨터이다. 버스 승객 수를 세기위한 초음파 센서 2개를 연결하여 사용하며 연결된 라즈베리파이와 Serial 통신을 이용하여 통신한다.



[그림4] 초음파 센서 HC-SR04

초음파란 사람의 귀에 들리지 않을 정도로 높은 주파수(약 20kHz 이상)의 소리를 말한다. 이와 같은 특성을 이용한 것이 초음파센서로 음파를 쏘아올리고 반사되어 오는 음파까지의 시간차를 거리로 계산하여 측정하는 방식으로 동작된다. 초음파 거리 센서는 발신부와 수신부로 구성된다. 발신부는 함수 발생기에서 (+)와 전압을 번갈아 압전소자에 가해주면 압전소자의 변형에 의해 진동이 발생하고 진동에 의해 초음파가 발생하는 역압전 현상을 이용한다. 수신부는 발신부에서 발생한 초음파가 물체에 반사되어 돌아오는 파동에 의해 압전소자가 진동하고 진동에 의해 전압이 발생하는 정압전 현상을 이용하여 반사되어 돌아오는 시간을 기초로 거리를 측정한다. 초음파는 파장이 짧아 지향성과 직진성이 높으며 공기 중에서는 340m/s의 일정한 속도로 진행하는 특징을 가지므로 거리 측정을 위한 수단으로 많이 사용된다. 아두이노에 연결되는 초음파 센서는 HC-SR04로, 2cm - 700cm의 거리를 초음파를 사용하여 측정할 수 있으며 5m까지는 안정된 신호를 제공한다. 승객의 수를 측정하기 위해 2개의 센서를 사용하여 방향성을 구현하여 사용한다.



[그림5] 비콘

비콘은 무선통신장치로써 블루투스 4.0 기반의 프로토콜을 사용해 주변에 있는 기기들에게 신호를 전달하는 장치이다. 대여한 비콘 장치가 원활히 작동하지 않아 스마트폰의 블루투스 기능과 Beacon Simulator 어플리케이션을 활용하여 사용하였다.

2.1.3 개발 환경

구분		항목	적용내역
S/W	OS	Windows 7,10, Android, Raspbian, Linux	Windows 7,10: Server, Bus Android: Application Raspbian: Raspberrypi3 Linux: AWS
	IDE	Pycharm, Arduino Sketch, Android Studio	Pycharm: Server Arduino Sketch: 아두이노 Android Studio: Application
	개발도구	Python2 Bluetooth Library	라즈베리파이와 정류장의 비콘 간의 통신을 위해 사용
	개발언어	Python2,3 Java	Python: Server, Bus Java: Android
H/W	디바이스	Arduino Uno	센서로부터 받은 데이터를 정보화
	센서	HC-SR04	승객 수를 세기 위함
	통신	Serial	Raspberrypi와 Serial로 유선 통신
	개발언어	C++	Arduino Sketch(IDE)를 조작하기 위함

[표1] 개발 환경

2.2 서버 설계

2.2.1 서버의 클라이언트 핸들러 (쓰레드)

-서버 - 버스

```
if req_type == 'type:load':

    bus_id = decoded[decoded.find('bus_') : decoded.find('bus_') + 10]
    cnt = decoded[decoded.find('people:') + 7: -2]
    data = cnt
    location = decoded[decoded.find('location:') + 9: decoded.find('\r\nstatus:')]
    print(type(location))
    status = decoded[decoded.find('status:') + 7: decoded.find('\r\npeople:')]
    if bus_id not in bus_list:
        c.execute('CREATE TABLE {} (location text , bus_status text , count integer)'.format(bus_id))
        bus_list.append(bus_id)
        c.execute("""INSERT INTO {} VALUES ('{1}', '{2}', {3})""".format(bus_id, location, status, cnt))
    c.execute("""UPDATE {} SET location = '{1}', bus_status = '{2}', count = {3}""".format(bus_id, location, status, cnt))
    print(decoded, bus_list)
```

[그림6] 쓰레드 DB 업데이트

```
b'type:load\r\nbus_102301\r\nlocation:2\r\nstatus:STOP\r\npeople:0\r\n'
```

[그림7] 메시지 형식

서버는 위 그림과 같은 형식으로 메시지를 수신한다. 서버는 받은 데이터로부터 req_type이 type:load라는 것을 확인하여, 전송받은 데이터로부터 bus_id, location, status, people 정보를 추출하고, 먼저 bus_id가 서버의 bus_list내에 없는 경우(즉, 고유한 id를 가진 새로운 버스가 운영을 시작한 경우) DB파일에 bus_id와 동일한 이름을 갖는 table을 생성하고, 생성된 table에다가 추출했던 데이터들을 저장한다. 또한 같은 id의 bus 정보들을 bus_list 리스트에 추가한다.

```
['bus_101001', 'bus_101940']
```

[그림8] 운행 중인 버스 id 예시

- 서버 - 사용자

```
elif req_type == 'type:get':
    data = ""
    if bus_list == []:
        data = '\n'
    else:
        for bus in bus_list:
            for row in c.execute("SELECT * FROM {} ".format(bus)):
                data += "bus_id:" + bus + "location:" + str(row[0]) + "bus_status:" + str(row[1]) + "count:" + str(row[2]) + "\n"

# for row in c.execute("SELECT * FROM people"):
#     data = str(row[0])
print(data)
```

[그림9] 쓰레드 DB 업데이트

```
b'type:get\r\npeople:\n'
```

[그림10] 사용자 요청 메시지 형식

사용자 어플은 다음과 같은 데이터(정보요청)를 보냅니다.

서버는 받은 데이터로부터 req_type이 type:get이라는 것을 확인하여 만일 bus_list가 비어있으면 (현재 운행 중인 버스가 없으면) "\n"을 사용자 어플리케이션에 전송하여, 운행 중인 버스가 없다는 것을 알리고, 만일 운행 중인 버스가 있다면, 사용자 어플리케이션에 보낼 데이터를 저장한다.

```
bus_id:bus_102301location:6bus_status:GOINGcount:0
```

[그림11] 사용자 답변 메시지 형식

서버는 위의 조건문을 거쳐 req_type = type:load일 경우 버스로부터 전송받은 사람의 수를 다시 버스로 보내주어(ACK) 버스 측에서 데이터가 성공적으로 전송됐다는 것을 인지하도록 해준다. 또한 형식이 type:get일 경우 사용자 어플리케이션에 버스에 대한 모든 정보(버스 식별자, 위치, 버스 운행 상태, 승객 수)를 전송하게 되며 최종적인 데이터를 해당 클라이언트에 전송 후 DB를 업데이트한다.

```
conn.send(data.encode('UTF-8'))
db.commit()
```

[그림12] 최종 데이터 전송 및 DB 업데이트

- 서버 예외 처리

```
except socket.error as e: # socket.error exception
    logging.exception('socket error: {}'.format(e)) # 예외 발생 시 반드시 출력
    if req_type == 'type:load':
        c.execute('DROP TABLE if exists {}'.format(bus_id))
        bus_list.remove(bus_id)
        db.commit()
    break
conn.close() # 함수 종료되면서 쓰레드는 kill 된다.
```

[그림13] 클라이언트 강제 종료 시 수행

클라이언트 측에서 일방적으로 연결을 종료하게 되면(강제 종료), 서버는 exception 핸들러를 통해 에러를 처리한다. 서버는 DB에 해당 쓰레드의 bus_id와 같은 이름의 테이블이 존재하면 제거하고 업데이트하며, bus_list에서도 그 bus_id를 삭제한다. 이후 해당 쓰레드를 종료시킨다.

2.2.2 서버 구동

```
def thread_server(my_port):
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM) # make listening socket
    sock.bind(('', my_port)) # bind it to server port number
    sock.listen(5) # listen, allow 5 pending connects
    logging.info('Server started')
    while True: # do forever (until process killed)
        conn, cli_addr = sock.accept() # wait for next client connect
        logging.info('Connected by {}'.format(cli_addr))
        handler = threading.Thread(target=thread_handler, args=(conn, cli_addr)) # 새로운 쓰레드가 thread handler를 수행한다. func이름과 args를 별도로 줘야한다.
        handler.daemon = True # daemonize this thread. i.e will not wait for it. 새로운 쓰레드가 부모와 상관없이 별도로 돌겠다.
        handler.start() # 새로 생성된 쓰레드가 돌기 시작한다.

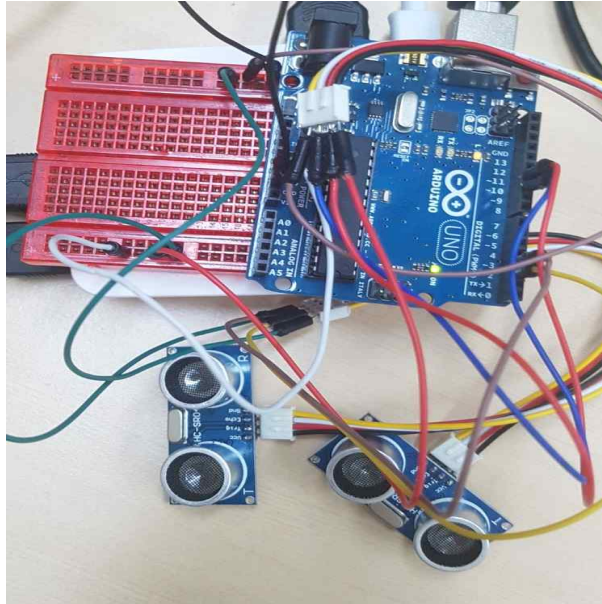
if __name__ == '__main__':
    logging.basicConfig(filename='', level=logging.INFO) # filename = '' : 콘솔로 로그런팅, logging.INFO 이상의 level들의 메시지들이 출력된다.
    thread_server(50007)
```

[그림14] 서버 구동

본 서버는 Daemon threading 서버로 설정한 포트번호(50007)로 서버를 열고, 새로운 연결이 들어올 때마다(새로운 클라이언트들과 연결될 때마다) 각 클라이언트에 대한 핸들러 쓰레드를 생성하여 서비스를 제공한다.

2.3 버스 설계

2.3.1 아두이노 - 라즈베리파이



[그림15] 아두이노 - 라즈베리파이 연결

라즈베리파이와 연결된 아두이노는 2개의 초음파 센서를 가지고 있으며, 카운터를 활용하여 승객의 수를 측정한다.

```

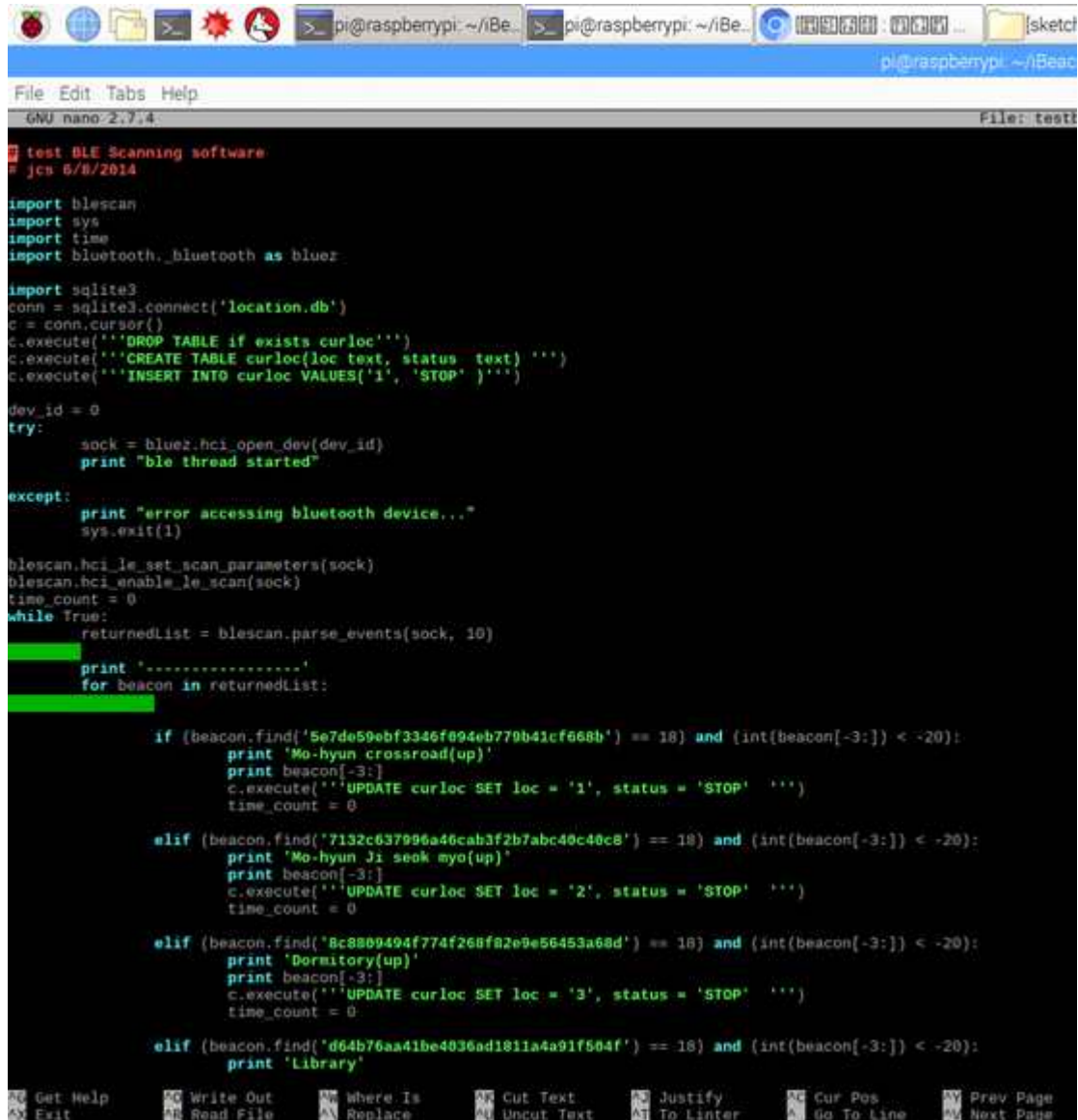
void checkDistance(int distance1, int distance2)
{
  if (distance1 <= 80){
    if (outcount == 0){
      incount = 1;
      delay(700);
    }
    else if (outcount == 1){
      outcount = 0;
      count = count -1;
      Serial.println(count);
      delay(500);
    }
  }
  else if (distance2 <= 80){
    if (incount == 0){
      outcount = 1;
      delay(600);
    }
    else if (incount == 1){
      incount = 0;
      count = count + 1;
      Serial.println(count);
      delay(500);
    }
  }
  else {
    incount = 0;
    outcount = 0;
    delay(50);
  }
}

```

[그림16] 아두이노 인원 측정 함수

버스에서 승차하고 하차하는 승객을 세기 위해서는 센서 한 개로는 구현이 힘들다. 승객의 방향에 따라 승차와 하차가 나뉘는 것을 센서 하나로는 세분화가 힘들기 때문이다. 하지만 센서 2개를 사용하여 구현하였다. 센서는 거리를 주기적으로 측정하며 거리가 가까워지면 사람이 서있다고 감지한다. **센서를 피해 들어오거나 나가는 인원, 버스를 날아가듯이 빨리 타거나 나가는 인원은 없다고 가정하였다.** 센서 2개 중에 바깥 쪽 센서가 먼저 사람을 감지했을 때, 하차 카운터가 0이라면 승객이 승차 중이므로, 승차 카운터를 1로 바꾸어 사람이 승차한다는 것을 준비하고, 하차 카운터가 1이라면 승객이 하차하는 것이므로 승객 수를 1 감소시키며 카운터를 초기화 시킨다. 반대로 버스 안쪽 센서가 먼저 반응했을 때, 승차 카운터가 0이라면 하차하는 사람이 존재하므로 하차 카운터를 1, 1이라면 승객이 승차했으므로 승객수를 1 증가시킨다.

2.3.2 블루투스 - 라즈베리파이



```
test BLE Scanning software
# jcs 6/8/2014

import ble_scan
import sys
import time
import bluetooth._bluetooth as bluez

import sqlite3
conn = sqlite3.connect('location.db')
c = conn.cursor()
c.execute('DROP TABLE if exists curloc')
c.execute('CREATE TABLE curloc(loc text, status text)')
c.execute('INSERT INTO curloc VALUES('1', 'STOP')')

dev_id = 0
try:
    sock = bluez.hci_open_dev(dev_id)
    print "ble thread started"

except:
    print "error accessing bluetooth device..."
    sys.exit(1)

ble_scan.hci_le_set_scan_parameters(sock)
ble_scan.hci_enable_le_scan(sock)
time_count = 0
while True:
    returnedList = ble_scan.parse_events(sock, 10)
    print "-----"
    for beacon in returnedList:

        if (beacon.find('5e7de59ebf3346f094eb779b41cf668b') == 18) and (int(beacon[-3:]) < -20):
            print 'Mo-hyun crossroad(up)'
            print beacon[-3:]
            c.execute('UPDATE curloc SET loc = '1', status = 'STOP'')
            time_count = 0

        elif (beacon.find('7132c637996a46cah3f2b7abc49c40c8') == 18) and (int(beacon[-3:]) < -20):
            print 'Mo-hyun Ji seek myo(up)'
            print beacon[-3:]
            c.execute('UPDATE curloc SET loc = '2', status = 'STOP'')
            time_count = 0

        elif (beacon.find('8c8809494f774f268f02e9e56453a68d') == 18) and (int(beacon[-3:]) < -20):
            print 'Dormitory(up)'
            print beacon[-3:]
            c.execute('UPDATE curloc SET loc = '3', status = 'STOP'')
            time_count = 0

        elif (beacon.find('d64b76aa41be4036ad1811a4a91f504f') == 18) and (int(beacon[-3:]) < -20):
            print 'Library'
```

[그림17] 비콘 탐색

버스는 상시 자신의 위치를 업데이트하기 위해 무한 루프를 사용한다. 라즈베리파이의 블루투스 기능을 사용하여, 정류장에 있는 비콘과 연결을 시도한다. 연결에 성공하면 연결에 성공한 비콘의 고유 UUID를 비교하여 정류장을 파악하고 DB에 위치를 업데이트한다. 또한 라즈베리파이에서는 버스가 정류장에 무사히 정차했다고 판단하여 버스의 상태를 정차로 바꾼다. 만약 연결되어있는 비콘과의 거리가 일정 거리 이상 멀어지거나, 연결이 끊어지게 되면 일정 시간(7초) 대기하며, 잠시 연결이 불안정하여 연결이 끊어질 수도 있는 경우를 방지한다. 연결이 끊어지고 나면, 해당 클라이언트에서는 버스가 출발했다고 판단하고 버스의 상태를 출발로 바꾼다.

2.3.3 버스 - 서버

버스의 정보는 언제나 서버 DB에 저장되어 있으며, 버스의 운행상태가 바뀔 때 마다 서버와 통신하여 버스 정보를 업데이트한다. 또한 전송 시에 아두이노 센서로부터 수신하여 받은 승객의 수 데이터를 기반으로 혼잡도를 판단하여 버스 정보와 함께 송신한다.

```
conn = sqlite3.connect('location.db')
c = conn.cursor()
port = "/dev/ttyACM0"
ser = serial.Serial(port, 57600)
ser.flushInput()
```

[그림18] 로컬 DB 지정 및 포트 초기화

```
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect(server_addr) # connect to server process

# DB 초기화
status = "STOP"
cur_station = 'Nowhere'
now = time.localtime()
cnt_p = '0'
bus_id = "%02d%02d%02d" % (now.tm_hour, now.tm_min, now.tm_sec)
req = "type:load\r\nbus_" + bus_id + "\r\nlocation:" + cur_station + "\r\nstatus:" + status + "\r\npeople:" + cnt_p + "\r\n"
sock.send(req.encode('UTF-8')) # send message to server
data = sock.recv(1024).decode('utf-8') # receive response up to 1KB
print("Data has been arrived successfully.")
```

[그림19] DB 초기화 후, 서버 DB 업데이트

맨 처음 버스가 운행을 위해 프로그램을 실행하면, 버스는 자동으로 해당 로컬 DB를 초기화 하고, 아두이노와의 Serial 통신을 위한 포트를 설정한다. 또한 서버와의 소켓 통신을 위해 변수들을 초기화 한 후, 서버에 메시지를 보내 해당 버스의 정보를 업데이트한다. 이때 버스들을 구분하기 위해 각각 id를 부여받는데, 이때 id는 프로그램이 실행되는 시각(시분 초를 그대로 쓴다. ex: 173024)을 id로 사용한다.


```

while True:
    if status == 'STOP': # 버스가 멈추어야만 인원 세기
        for row in c.execute('''SELECT * FROM curloc'''):
            tmp_station = int(row[0])
            status = row[1]
        if cur_station != tmp_station: # 비콘이 새로 연결되어 업데이트된 장소가 이전 장소와 다를때 -> 새로운 정류장
            cur_station = tmp_station # 새로운 정류장으로 업데이트
            # 정류장 정차 업데이트
            req = "type:load\r\nbus_" + bus_id + "\r\nlocation:" + str(
                cur_station) + "\r\nstatus:" + status + "\r\npeople:" + cnt_p + "\r\n"
            print(req)
            sock.send(req.encode('UTF-8')) # send message to server
            data = sock.recv(1024).decode('utf-8') # receive response up to 1KB
            print("Data has been arrived successfully. People: ", data)

            while True: # 아두이노가 종료 메시지를 보낼때까지 인원을 센다
                input_raw = ser.readline()
                cnt_p = input_raw.decode('utf-8')
                for row in c.execute('''SELECT status FROM curloc'''):
                    status = row[0]
                    if status == 'GOING': break
                req = "type:load\r\nbus_" + bus_id + "\r\nlocation:" + str(cur_station) + "\r\nstatus:" +
                    status + "\r\npeople:" + cnt_p + "\r\n"
                print(req)
                sock.send(req.encode('UTF-8')) # send message to server
                data = sock.recv(1024).decode('utf-8') # receive response up to 1KB
                print("Data has been arrived successfully. People: ", data)
            else: # DB 확인하며 버스 정차 확인
                for row in c.execute('''SELECT status FROM curloc'''):
                    status = row[0]
sock.close() # close socket to send eof to server

```

[그림20] 버스 운행 중 상태 업데이트

버스의 정보를 업데이트 하는 시기는 버스가 정차했을 때만 업데이트하며, 블루투스 통신으로 파악하여 DB에 업데이트한 버스 위치를 불러와서 새로운 정류장일 때만 서버에 DB를 업데이트한다. “type:load\r\n” 형태로 메시지를 보내며, 서버가 ACK를 보낼 때까지 기다려 수신 확인을 한다. 인원 측정은 버스가 정차했다고 여겨질 때만 시행하며, Serial 통신으로 아두이노로부터 인원 측정값을 DB의 버스 상태를 계속 불러와 비콘의 연결이 끝나 버스가 출발했다고 판단할 때까지 받는다. 또한 이 정보도 지속적으로 서버와 통신하여 업데이트한다. 이후 버스의 상태가 출발로 바뀌면 지속적으로 비교하면서 정차를 기다린다.

2.4 정류장 설계

2.4.1 정류장

정류장은 주기적으로 “type:getWrWn” 의 메시지를 서버에 보내주어 버스 정보를 최신화 한다. 그 외적으로 정류장에는 비콘이 설치되어 있어 정류장에 정차하는 버스와 통신한다.

2.4.2 서버 - 정류장

정류장은 주기적으로 서버와 통신하여 정보를 받아 현재 해당 정류장을 기반으로 관련된 버스의 위치를 모니터에 띄워준다.

```
def run_station(server_addr):  
    # make TCP/IP socket obj  
    # 상행선 1 ~ 6  
    # 하행선 7 ~ 14  
    last_sock = time.time()  
    time_delay = 10 # 10초마다 버스 현황 갱신  
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
    sock.connect(server_addr) # connect to server process  
    if station <= 6: # 상행선 하행선 구분  
        up = True  
    else: up = False  
    while True:  
        req = "type:get\r\n" # 정보 요청 메시지 형식  
        sock.send(req.encode('UTF-8')) # send message to server  
        last_sock = time.time()  
        data = sock.recv(1024).decode('utf-8') # receive response up to 1KB  
        bus_info = []
```

[그림21] 버스 정보 요청

10초마다 소켓 통신을 사용해 type:getWrWn 형식의 메시지를 보내고, 받은 버스 현황 정보를 파싱하여 저장하기 위한 리스트를 선언한다.

```

if data != '\n':
    while data != '': # 받은 데이터를 파싱하여 정보화
        temp_id = data[data.find('bus_id:')+7:data.find('location:')]
        temp_lo = data[data.find('location:')+9:data.find('bus_status:')]
        temp_st = data[data.find('bus_status:')+11:data.find('count:')]
        temp_ct = data[data.find('count:')+6:data.find('\n')]
        data = data[data.find('\n') + 1:]
        if int(temp_ct) <= 10: # 승객수에 따른 버스 혼잡도 초기화
            temp_com = 'free'
        elif int(temp_ct) <= 20:
            temp_com = 'normal'
        elif int(temp_ct) <= 30:
            temp_com = 'crowded'
        else: temp_com = 'full'
        bus_info.append([temp_id, temp_lo, temp_st, temp_ct, temp_com])
show_list = []
for x in bus_info:
    if up:
        if int(x[1]) < station:
            show_list.append(x)
    else:
        if (int(x[1]) > 6) and (int(x[1]) < station):
            show_list.append(x)

```

[그림22] 메시지 파싱

받은 정보를 버스id, 위치, 상태, 승객수 순으로 파싱한 후, 승객 수에 따른 혼잡도를 계산하고 이를 버스 정보 리스트에 저장한다. 그 다음 해당 정류장에 알맞은 버스만 골라내기 위해 새로운 리스트에 비교 필터로 걸러내어 새로 저장한다.

```

if show_list:
    for info in show_list:
        station_str = station_map_list[int(info[1]) - 1]
        print("{} | station:{} | {} | {}".format(info[0],station_str,info[2],info[4]))
else:print("There are no bus in service")
while (time.time() - last_sock < time_delay):
    continue

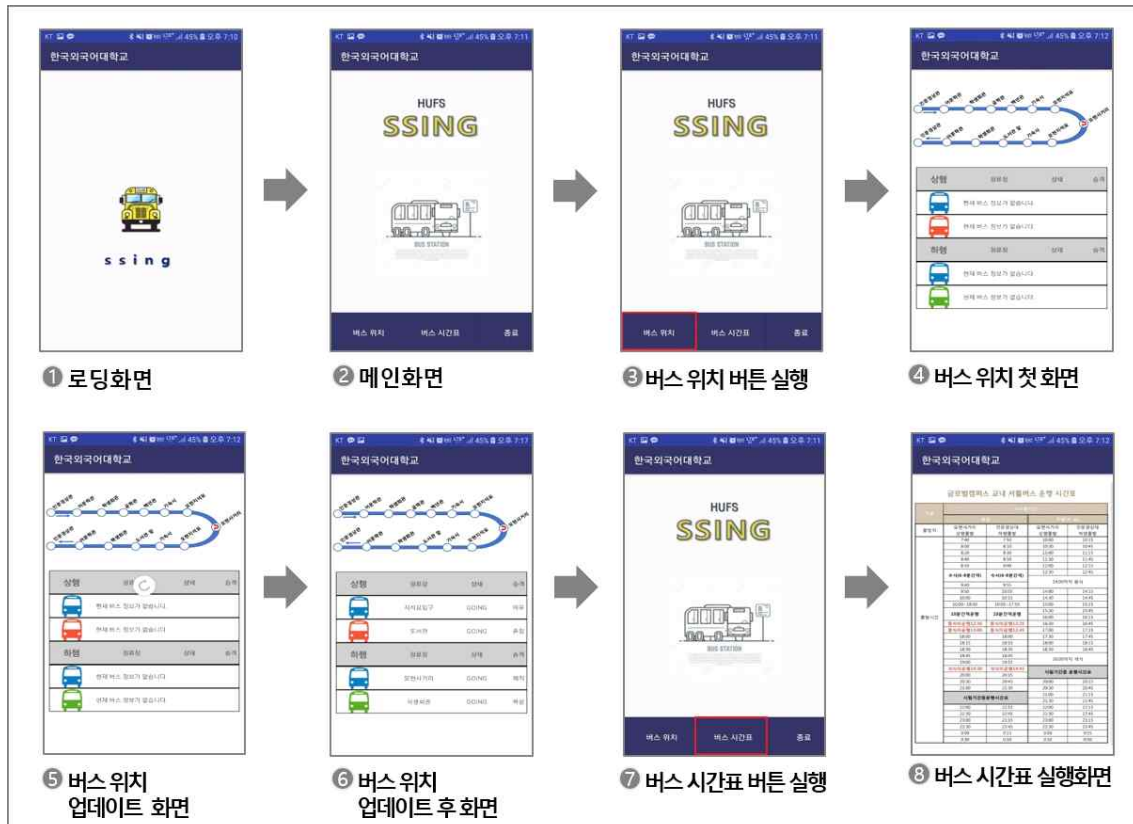
```

[그림23] 버스 정보 출력

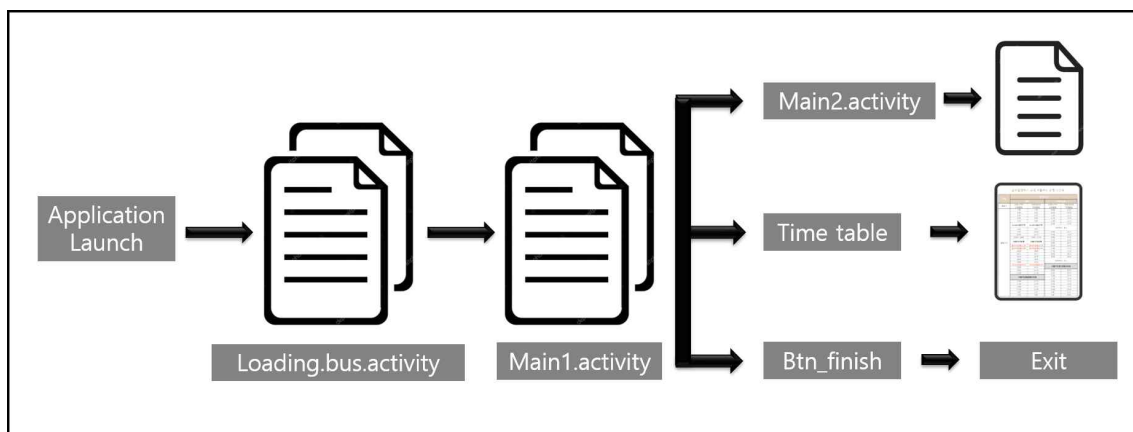
걸러낸 리스트를 형식에 맞게 출력한다. 만약 운행 중인 버스가 존재하지 않거나, 운행 중인 버스가 해당 정류장과 관련이 없다면 버스가 없다고 출력한다.

2.5 어플리케이션 설계

2.5.1 어플리케이션 구성도



[그림24] 어플리케이션 실행화면



[그림25] 어플리케이션 흐름도

-Loading

처음 어플리케이션을 실행했을 때 로딩중임을 나타내는 loading_bus.activity이다. 1.5초 후에 메인 화면으로 이동한다.

```
private void initView() {  
    imgAndroid = (ImageView) findViewById(R.id.img_android);  
    anim = AnimationUtils.loadAnimation(context, this, R.anim.loading);  
    imgAndroid.setAnimation(anim);  
}
```

[그림26] 이미지 설정

```
initView();  
Handler handler = new Handler() {  
    public void handleMessage(Message msg) {  
        super.handleMessage(msg);  
        //startActivity(intent);  
        startActivity(new Intent(packageContext, loading_bus.this, MainActivity.class));  
        finish();  
    }  
}
```

[그림27] 로딩 종료 후 메인화면으로 이동

```
handler.sendEmptyMessageDelayed(what, 0, delayMillis, 1500); //1.5초후 화면전환
```

[그림28] 1.5초 지연

-Main

메인 화면은 3개의 버튼을 지원하며 각각 버스 위치, 버스 시간표, 종료 기능을 수행한다. 메인 화면인 main.activity는 각 3개의 기능을 수행하는 activity로 이동하는 역할을 수행한다.

```
Button button4 = (Button) findViewById(R.id.button4);  
button4.setOnClickListener(new View.OnClickListener() {  
    public void onClick(View view) {  
        Intent intent = new Intent(getApplicationContext(), Main2Activity.class);  
        startActivity(intent);  
    }  
});
```

[그림29] button4

버스 위치 기능인 Main2.activity로 이동하는 setOnClickListener 함수이다. Main2.activity는 다음 절에서 후술한다.

-Time table

```
Button button6 = (Button) findViewById(R.id.button6);
button6.setOnClickListener((view) → {
    Intent intent = new Intent(getApplicationContext(), time_table.class);
    startActivity(intent);
});
```

[그림30] button6

버스 시간표인 time_table.activity로 이동하는 setOnClickListener 함수이다.

```
public class time_table extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_time_table);
    }
}
```

[그림31] time_table class

메인 화면에서 버튼을 클릭하여 버스 시간표 화면으로 왔을 때 실행되는 클래스이다.
버스 시간표 사진을 출력한다.

종별	종별시간	종별시간	종별시간
출발시간	07:40	08:30	09:20
출발시간	08:00	08:30	09:20
출발시간	08:20	08:30	09:20
출발시간	08:40	08:30	09:20
출발시간	09:00	08:30	09:20
출발시간	09:20	08:30	09:20
출발시간	09:40	08:30	09:20
출발시간	10:00	08:30	09:20
출발시간	10:20	08:30	09:20
출발시간	10:40	08:30	09:20
출발시간	11:00	08:30	09:20
출발시간	11:20	08:30	09:20
출발시간	11:40	08:30	09:20
출발시간	12:00	08:30	09:20
출발시간	12:20	08:30	09:20
출발시간	12:40	08:30	09:20
출발시간	13:00	08:30	09:20
출발시간	13:20	08:30	09:20
출발시간	13:40	08:30	09:20
출발시간	14:00	08:30	09:20
출발시간	14:20	08:30	09:20
출발시간	14:40	08:30	09:20
출발시간	15:00	08:30	09:20
출발시간	15:20	08:30	09:20
출발시간	15:40	08:30	09:20
출발시간	16:00	08:30	09:20
출발시간	16:20	08:30	09:20
출발시간	16:40	08:30	09:20
출발시간	17:00	08:30	09:20
출발시간	17:20	08:30	09:20
출발시간	17:40	08:30	09:20
출발시간	18:00	08:30	09:20
출발시간	18:20	08:30	09:20
출발시간	18:40	08:30	09:20
출발시간	19:00	08:30	09:20
출발시간	19:20	08:30	09:20
출발시간	19:40	08:30	09:20
출발시간	20:00	08:30	09:20
출발시간	20:20	08:30	09:20
출발시간	20:40	08:30	09:20
출발시간	21:00	08:30	09:20
출발시간	21:20	08:30	09:20
출발시간	21:40	08:30	09:20
출발시간	22:00	08:30	09:20
출발시간	22:20	08:30	09:20
출발시간	22:40	08:30	09:20
출발시간	23:00	08:30	09:20
출발시간	23:20	08:30	09:20
출발시간	23:40	08:30	09:20
출발시간	24:00	08:30	09:20

[그림32] 버스 시간표

-Quit

```
Button btn_finish = (Button) findViewById(R.id.btn_finish);
btn_finish.setOnClickListener((view) → { finish(); });
```

[그림33] button_finish

어플리케이션을 종료하는 setOnClickListener 함수이다.

2.5.2 안드로이드 - 서버

-Bus Location

```
try {
    clientSocket = new Socket(ip, port);
    socketIn = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
    socketOut = new PrintWriter(clientSocket.getOutputStream(), autoFlush: true);
} catch (Exception e) {
    e.printStackTrace();
}
```

[그림34] 클라이언트 소켓 통신 설정

메인 화면에서 해당 화면으로 이동하게 되면 웹 서버와 소켓 통신 연결을 하게 된다.

```
swipeRefreshLayout.setOnRefreshListener(() → {
    up = 0;
    down = 0;
    tv1.setText(nobus);
    tv2.setText(nobus);
    tv3.setText(nobus);
    tv4.setText(nobus);
    tv1_state.setText("");
    tv2_state.setText("");
    tv3_state.setText("");
    tv4_state.setText("");
    tv1_con.setText("");
    tv2_con.setText("");
    tv3_con.setText("");
    tv4_con.setText("");

    socketOut.println("type:getWr#npeople:");
    //tv.setText("민경관");

    new Handler().postDelayed(() → {
        swipeRefreshLayout.setRefreshing(false);
    }, delayMillis: 1000);
});
```

[그림35] 4대의 버스 정보를 출력하기 위한 변수 초기화

어플리케이션 화면에서 버스 정보를 얻기 위해 스크린을 당겨 새로 고침을 실행하면, Handler().postDelayed() 함수가 실행되고, 1초의 시간을 대기한 후 버스 정보를 받아 오기 위한 새로운 쓰레드를 실행한다.

```

class MyThread extends Thread {
    @Override
    public void run() {
        while (true) {
            try {
                // InputStream의 값을 읽어와서 data에 저장
                String data = socketIn.readLine();
                int loc_idx = data.indexOf("location");
                int status_idx = data.indexOf("bus_status:");
                String loc, status;
                loc = data.substring(loc_idx + 9, status_idx);
                if (Integer.parseInt(loc) < 7){
                    if (up == 0) {
                        Message msg = myHandler1.obtainMessage();
                        msg.obj = data;
                        myHandler1.sendMessage(msg);
                        up++;
                    }
                }
                else{
                    up = 0;
                    Message msg = myHandler2.obtainMessage();
                    msg.obj = data;
                    myHandler2.sendMessage(msg);
                }
            }
            else if (Integer.parseInt(loc) >= 7){
                if (down == 0) {

```

```

                    Message msg = myHandler3.obtainMessage();
                    msg.obj = data;
                    myHandler3.sendMessage(msg);
                    down++;
                }
            }
            else{
                down = 0;
                Message msg = myHandler4.obtainMessage();
                msg.obj = data;
                myHandler4.sendMessage(msg);
            }
        }
        // Message 객체를 생성, 핸들러에 정보를 보낼 땐 이 메시지 객체를 이용
    }
    catch (Exception e) {
        e.printStackTrace();
    }
}

```

[그림36] 버스 행선로 구분

```

bus_id:bus_175646location:14bus_status:STOPcount:38

```

[그림37] 서버로부터 받아오는 메시지 형식

```

private String mlist[] = {"모현 사거리", "지석묘입구", "기숙사", "도서관", "학생회관", "머문관", "인경관", "머문관",
    "학생회관", "공학관", "백년관", "기숙사", "지석묘입구", "모현사거리"};

```

[그림38] 버스 위치에 맞는 리스트

앞서 새로 고침을 통해 새로운 쓰레드를 실행하면, 클라이언트가 소켓 통신을 통해 받은 메시지를 오게 되어, 메시지를 형식에 맞게 분석하기에 앞서 버스의 정보가 상행선인지 하행선인지 버스 위치를 먼저 파악하여 구분한 후 맞는 위치에 있는 쓰레드를 실행하게 된다.


```

class MyHandler3 extends Handler {
    @Override
    public void handleMessage(Message msg) {
        String str = msg.obj.toString();
        String loc, status, con;
        loc = str.substring(str.indexOf("location") + 9, str.indexOf("bus_status:"));
        status = str.substring( str.indexOf("bus_status:") + 11, str.indexOf("count:"));
        con = str.substring( str.indexOf("count:") + 6, str.length());
        if (Integer.parseInt(con) <= 10)
            con = "쾌적";
        else if (Integer.parseInt(con) <= 20)
            con = "머무";
        else if (Integer.parseInt(con) <= 30)
            con = "혼잡";
        else
            con = "꽉참";
        tv3.setText(mList[ Integer.parseInt(loc) - 1]);
        tv3_state.setText(status);
        tv3_con.setText(con);
    }
}

```

[그림39] 수신 메시지 파싱 후 저장하는 쓰레드

앞서 상행선과 하행선으로 나누어져 들어간 버스 정보는 그에 맞는 쓰레드(Myhandler1,2,3,4)가 해당 메시지를 파싱하여 변수에 저장하며, 추가로 버스 승객 수에 따른 혼잡도를 계산하여 추가로 저장한다. 이후 변수들을 setText 함수를 사용하여 String 형태로 맞는 위치에 출력한다.

4. 기대 효과

서비스를 사용하면서 제공되는 정보를 바탕으로 교내 셔틀버스로 인원이 몰리지 않게 되어 인원 분산으로 인한 여유롭고 쾌적한 셔틀버스 이용 가능성을 기대한다. 또한 무리한 탑승으로 인한 문 끼임, 운전 방해 등의 안전사고 위험의 감소를 야기할 수 있다.

버스 탑승여부를 빠르게 선택함으로써 시간을 버리지 않고 효율적으로 사용 가능하게 되며, 또한 라즈베리파이와 아두이노를 이용하여 적은 비용으로 교내 시스템을 구축할 수 있어 효과적이다.

4. 팀원 임무

팀원	담당	상세설명
김준영(T)	총괄	서버와 클라이언트 구현, 어플리케이션 기능 구현 및 통신, 모든 담당 업무 총괄
임광호	장치, 서버, 클라이언트	장치 간 통신 구현 (라즈베리파이, 아두이노, 서버), 어플리케이션 기능 구현 및 통신, 문서 작성
지승환	어플리케이션 통신, 회의록	안드로이드 통신 구축, 문서 작성
이경원	어플리케이션 레이아웃, 디자인	어플리케이션 UI 디자인, 문서 작성
김아연	서버 구축 어플리케이션 레이아웃, 디자인	어플리케이션 레이아웃 구축, 어플리케이션 UI 디자인, PPT, 문서 작성
김예주		어플리케이션 레이아웃 구축, 어플리케이션 UI 디자인, PPT, 문서 작성

[표2] 팀원 임무

4. 부록

〈Server〉

```
###threaded server###
import ...

bus_list = []

def thread_handler(conn, cli_addr):
    bus_id = ''
    while True:
        try:
            data = conn.recv(1024) # recv next message on connected socket
            print(data)
            decoded = data.decode('UTF-8')
            db = sqlite3.connect('people.db')
            print(decoded)
            c = db.cursor()
            if not data: # eof when the socket closed
                logging.info('Client closing: {}'.format(cli_addr))
                # c.execute('DROP TABLE if exists {}'.format(bus_id))
                if req_type == 'type:load':
                    c.execute('DROP TABLE if exists {}'.format(bus_id))
                    bus_list.remove(bus_id)
                print(bus_list)
                break
            req_type = decoded[decoded.find('\n'):decoded.find('\n\n')]
            print(req_type)
            if req_type == 'type:load':

                bus_id = decoded[decoded.find('bus_'): decoded.find('bus_') + 10]
                cnt = decoded[decoded.find('people:') + 7: -2]
                data = cnt
                location = decoded[decoded.find('location:') + 9: decoded.find('\n\nstatus:')]
                print(type(location))
                status = decoded[decoded.find('status:') + 7: decoded.find('\n\npeople:')]
                if bus_id not in bus_list:
                    c.execute('CREATE TABLE {} (location text , bus_status text , count integer)'.format(bus_id))
                    bus_list.append(bus_id)
```

```

        c.execute("""INSERT INTO {0} VALUES ({1}', {2}', {3})""".format(bus_id, location, status, cnt))
        c.execute("""UPDATE {0} SET location = '{1}', bus_status = '{2}', count = {3}""".format(bus_id, location, status, cnt))
        print(decoded, bus_list)
        # conn.send(send_data.encode('UTF-8'))
    elif req_type == 'type:get':
        data = ""
        if bus_list == []:
            data = '\n'
        else:
            for bus in bus_list:
                for row in c.execute("SELECT * FROM {}".format(bus)):
                    data += "bus_id:" + bus + "location:" + str(row[0]) + "bus_status:" + str(row[1]) + "count:" + str(row[2]) + "\n"

            # for row in c.execute("SELECT * FROM people"):
            #     data = str(row[0])
            print(data)
            logging.debug('Received: {}'.format(data)) # 디버깅할 때만 출력이 된다. debug: lowest level

        conn.send(data.encode('UTF-8')) # send a reply to the client
        db.commit()
    except socket.error as e: # socket.error exception
        logging.exception('socket error: {}'.format(e)) # 예외 발생시 반드시 출력
    if req_type == 'type:load':
        c.execute("""DROP TABLE if exists {0}""".format(bus_id))
        bus_list.remove(bus_id)
        db.commit()
        break
    conn.close() # 함수 종료되면서 쓰레드는 killed 된다.

def thread_server(my_port):
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM) # make listening socket
    sock.bind(('', my_port)) # bind it to server port number
    sock.listen(5) # listen, allow 5 pending connects
    logging.info('Server started')
    while True: # do forever (until process killed)

```

```

        conn, cli_addr = sock.accept() # wait for next client connect
        logging.info('Connected by {}'.format(cli_addr))
        handler = threading.Thread(target=thread_handler, args=(conn, cli_addr)) # 새로운 쓰레드가 thread_handler를 수행한다. func 이름과 args를 별도로 줘야한다.
        handler.daemon = True # daemonize this thread, i.e. will not wait for it. 새로운 쓰레드가 부모와 상관없이 별도로 끝난다.
        handler.start() # 새로 생성된 쓰레드가 돌기 시작한다.

if __name__ == '__main__':
    logging.basicConfig(filename='log', level=logging.INFO) # filename = 'log' : 파일로 로그파일될. logging.INFO 이상의 level들의 메시지가 출력된다.
    thread_server(50007)

```

<Bus - SocketClient>

- bus.py

```
import sys, socket
import time, random
import sqlite3
import serial

conn = sqlite3.connect('location.db')
c = conn.cursor()
port = "/dev/ttyACM0"
win_port = 'COM7'
ser = serial.Serial(port, 57600)
ser.flushInput()

def bus_run(server_addr):
    # make TCP/IP socket obj
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.connect(server_addr) # connect to server process

    # DB 초기화
    status = 'STOP'
    cur_station = 'Nowhere'
    now = time.localtime()
    cnt_p = '0'
    bus_id = "X02dX02dX02d" % (now.tm_hour, now.tm_min, now.tm_sec)
    req = "type:load\r\nbus_" + bus_id + "\r\nlocation:" + cur_station + "\r\nstatus:" + status + "\r\npeople:" + cnt_p + "\r\n"
    sock.send(req.encode('UTF-8')) # send message to server
    data = sock.recv(1024).decode('utf-8') # receive response up to 1KB
    print("Data has been arrived successfully. People: ", data)

while True:
    if status == 'STOP': # 버스가 멈추어야만 인원 세기
        for row in c.execute('SELECT * FROM curloc'):
            tmp_station = int(row[0])
            status = row[1]
        if cur_station != tmp_station: # 비콘이 새로 연결되어 업데이트된 장소가 이전 장소와 다를때 -> 새로운 정류장
            cur_station = tmp_station # 새로운 정류장으로 업데이트
            # 정류장 정차 업데이트
            req = "type:load\r\nbus_" + bus_id + "\r\nlocation:" + str(
                cur_station) + "\r\nstatus:" + status + "\r\npeople:" + cnt_p + "\r\n"
            print(req)
            sock.send(req.encode('UTF-8')) # send message to server
            data = sock.recv(1024).decode('utf-8') # receive response up to 1KB
            print("Data has been arrived successfully. People: ", data)

            while True: # 아두이노가 종료 메시지를 보낼때까지 인원을 센다
                input_raw = ser.readline()
                cnt_p = input_raw.decode('utf-8')

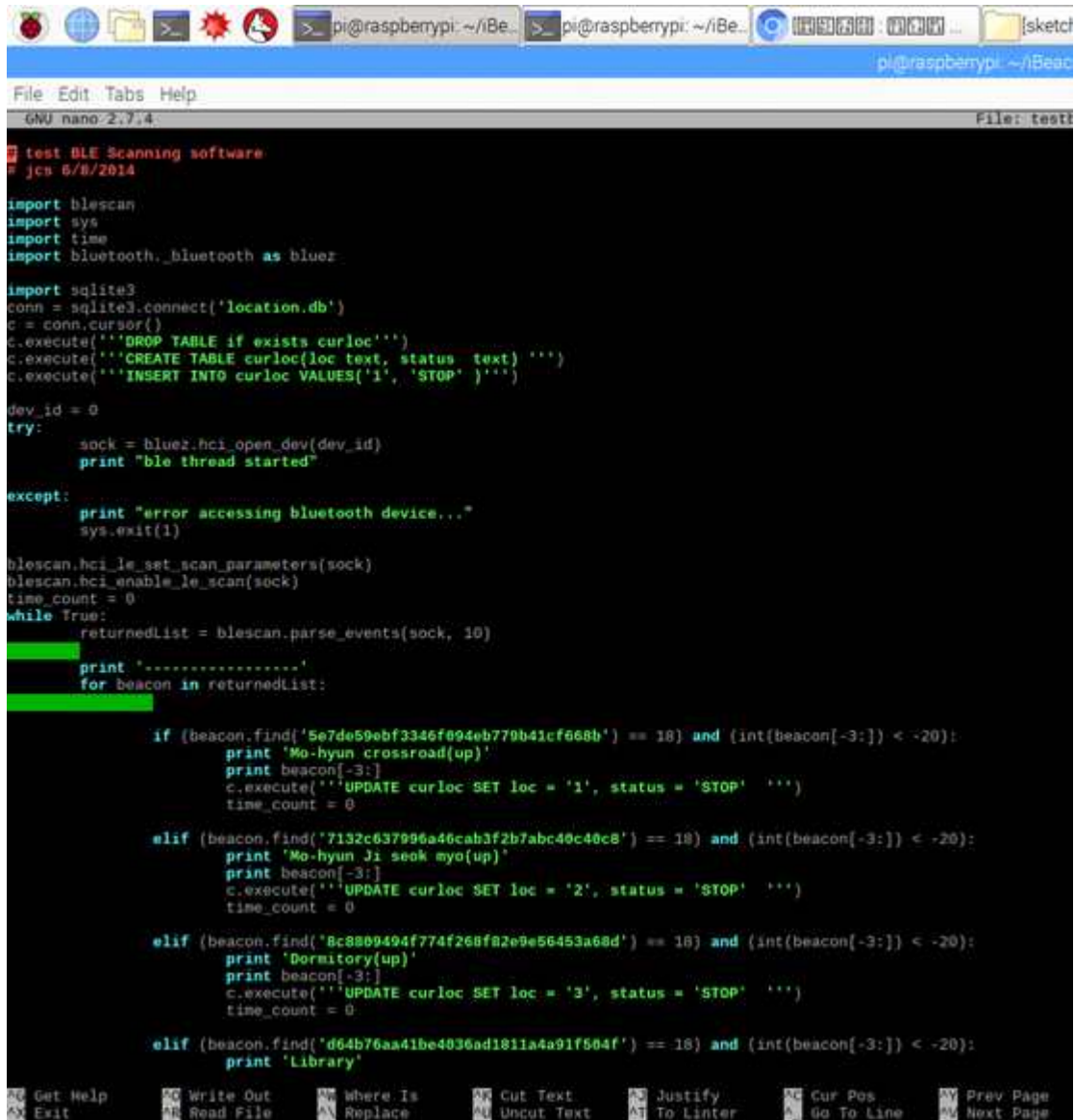
                for row in c.execute('SELECT status FROM curloc'):
                    status = row[0]
                if status == 'GOING': break
                print(cnt_p)

            # else:
            #     status = 'GOING'
            #     # 정류장 출발 업데이트
            #     req = "type:load\r\nbus_" + bus_id + "\r\nlocation:" + str(cur_station) + "\r\nstatus:" + status + "\r\npeople:" + cnt_p + "\r\n"
            #     print(req)
            #     sock.send(req.encode('UTF-8')) # send message to server
            #     data = sock.recv(1024).decode('utf-8') # receive response up to 1KB
            #     print("Data has been arrived successfully. People: ", data)
            # else: # DB 확인하며 버스 정차 확인
            #     for row in c.execute('SELECT status FROM curloc'):
            #         status = row[0]
            #     sock.close() # close socket to send eof to server

if __name__ == '__main__':
    bus_run(('ec2-13-209-49-254.ap-northeast-2.compute.amazonaws.com', 50007))
```

<Bus - BeaconScanner (Python 2.7)>

-testblescan.py



```
File Edit Tabs Help
GNU nano 2.7.4 File: testt

# test BLE Scanning software
# jcs 6/8/2014

import blescan
import sys
import time
import bluetooth._bluetooth as bluez

import sqlite3
conn = sqlite3.connect('location.db')
c = conn.cursor()
c.execute('DROP TABLE if exists curloc')
c.execute('CREATE TABLE curloc(loc text, status text)')
c.execute('INSERT INTO curloc VALUES('1', 'STOP')')

dev_id = 0
try:
    sock = bluez.hci_open_dev(dev_id)
    print "ble thread started"
except:
    print "error accessing bluetooth device..."
    sys.exit(1)

blescan.hci_le_set_scan_parameters(sock)
blescan.hci_enable_le_scan(sock)
time_count = 0
while True:
    returnedList = blescan.parse_events(sock, 10)
    print '-----'
    for beacon in returnedList:

        if (beacon.find('5e7de59ebf3346f094eb779b41cf668b') == 18) and (int(beacon[-3:]) < -20):
            print 'Mo-hyun crossroad(up)'
            print beacon[-3:]
            c.execute('UPDATE curloc SET loc = '1', status = 'STOP' ')
            time_count = 0

        elif (beacon.find('7132c637996a46cah3f2b7abc40c40c8') == 18) and (int(beacon[-3:]) < -20):
            print 'Mo-hyun Ji seek myo(up)'
            print beacon[-3:]
            c.execute('UPDATE curloc SET loc = '2', status = 'STOP' ')
            time_count = 0

        elif (beacon.find('8c8809494f774f268f82e9e56453a68d') == 18) and (int(beacon[-3:]) < -20):
            print 'Dormitory(up)'
            print beacon[-3:]
            c.execute('UPDATE curloc SET loc = '3', status = 'STOP' ')
            time_count = 0

        elif (beacon.find('d64b76aa41be4036ad1811a4a91f504f') == 18) and (int(beacon[-3:]) < -20):
            print 'Library'

Get Help  Write Out  Where Is  Cut Text  Justify  Cur Pos  Prev Page
Exit      Read File  Replace  Uncut Text  To Linter  Go To Line  Next Page
```

```

File Edit Tabs Help
GNU nano 2.7.4 File: tes

        time_count = 0

    elif (beacon.find('9e7de96067ad4c9a94066200009ed53a') == 18) and (int(beacon[-3:]) < -20):
        print 'Student hall(up)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '5', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('da8aaa9ce0304d4f98a7066a4521fc08') == 18) and (int(beacon[-3:]) < -20):
        print 'Lang & Literal Building(up)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '6', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('794bd6571bbf45b8b698f93c42a6e07f') == 18) and (int(beacon[-3:]) < -20):
        print 'Hum & Eco Building'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '7', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('c8e65aebd1d3471eafdad9340063a141') == 18) and (int(beacon[-3:]) < -20):
        print 'Lang & Literal Building(down)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '8', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('a9a95b5651dd4dd0aa4b83712c97165a') == 18) and (int(beacon[-3:]) < -20):
        print 'Student hall(down)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '9', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('8436bcaa619347d08f58120b5722b662') == 18) and (int(beacon[-3:]) < -20):
        print 'Engineering School(down)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '10', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('a2b32e1b5adb4c4f90bf93e4a7f70833') == 18) :
        print 'Baek nyun gwan'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '11', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('f8606bf8d85d4bebb595e1f64900df07') == 18) :
        print 'Dormitory(DOWN)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '12', status = 'STOP' ')

AG Get Help      AG Write Out    AW Where Is     AR Cut Text     AJ Justify      AG Cur Pos      AY Prev Page
AX Exit          AR Read File   AN Replace      AU Uncut Text  AT To Linter   AL Go To Line  AX Next Page

```

```

    elif (beacon.find('ea9bcab3373842bfb9047fe61300d032') == 18) :
        print 'Mo-hyun Ji seok myo(DOWN)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '13', status = 'STOP' ')
        time_count = 0

    elif (beacon.find('115c690c9b6248dd8d68906093ca8e91') == 18) :
        print 'Mo-hyun crossroad(DOWN)'
        print beacon[-3:]
        c.execute('UPDATE curloc SET loc = '14', status = 'STOP' ')
        time_count = 0

    else:
        if time_count > 30:
            c.execute('UPDATE curloc SET status = 'GOING' ')
            time_count = 0

        conn.commit()

        time.sleep(0.1)
        time_count += 1
        print time_count

AG Get Help      AG Write Out    AW Where Is     AR Cut Text     AJ Justify      AG Cur Pos      AY Prev Page
AX Exit          AR Read File   AN Replace      AU Uncut Text  AT To Linter   AL Go To Line  AX Next Page

```


〈Station.py〉

```
import socket
import time

station = 1 # 모현 사거리 상행선 정류장이라고 가정합니다.

station_map_list = ['Mohyun St.(UP)', 'Ji-Seok-Myo(UP)', 'Dormitory(UP)', 'Library', 'Student Hall(UP)',
                    'Lang&Literal Building(UP)', 'Human&Eco Building', 'Lang&Literal Building(DOWN)',
                    , 'Student Hall(DOWN)', 'Engineering Buliding', 'Baek nyun gwan', 'Dormitory(DOWN)', 'Ji-Seok-Myo(DOWN)',
                    'Mohyun crossroad(DOWN)']

station_name = station_map_list[station - 1]

def run_station(server_addr):
    # make TCP/IP socket obj
    # 상행선 1 ~ 6
    # 하행선 7 ~ 14
    last_sock = time.time()
    time_delay = 10 # 10초마다 버스 현황 갱신
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.connect(server_addr) # connect to server process
    if station <= 6: # 상행선 하행선 구분
        up = True
    else: up = False
    while True:
        req = "type:get\r\n" # 정보 요청 메시지 형식
        sock.send(req.encode('UTF-8')) # send message to server
        last_sock = time.time()
        data = sock.recv(1024).decode('utf-8') # receive response up to 1KB
        bus_info = []
        if data != '\n':
            while data != '': # 받은 데이터를 파싱하여 정보화
                temp_id = data[data.find('bus_id:')+7:data.find('location:')]
                temp_lo = data[data.find('location:')+9:data.find('bus_status:')]
                temp_st = data[data.find('bus_status:')+11:data.find('count:')]
                temp_ct = data[data.find('count:')+6:data.find('\n')]
                data = data[data.find('\n') + 1:]
                if int(temp_ct) <= 10: # 승객수에 따른 버스 혼잡도 초기화
                    temp_com = 'free'
                elif int(temp_ct) <= 20:
                    temp_com = 'normal'
                elif int(temp_ct) <= 30:
                    temp_com = 'crowded'
                else: temp_com = 'full'
                bus_info.append([temp_id, temp_lo, temp_st, temp_ct, temp_com])
            show_list = []
            for x in bus_info:
                if up:
                    if int(x[1]) < station:
                        show_list.append(x)
                else:
                    if (int(x[1]) > 6) and (int(x[1]) < station):
                        show_list.append(x)
            if show_list:
                for info in show_list:
                    station_str = station_map_list[int(info[1]) - 1]
                    print("{} | station:{} | {} | {}".format(info[0], station_str, info[2], info[4]))
            else: print("There are no bus in service")
            while (time.time() - last_sock < time_delay):
                continue
        sock.close() # close socket to send eof to server

if __name__ == '__main__':
    run_station(('13.209.49.254', 50007))
    # run_station(('localhost', 50007))
```

〈Arduino〉

```
#define echoPin1 2 // Echo Pin
#define trigPin1 3 // Trigger Pin

#define echoPin2 10 // Echo Pin
#define trigPin2 11 // Trigger Pin

long duration_in, distance_in; // Duration used to calculate distance
long duration_out, distance_out; // Duration used to calculate distance
int count = 0;
int incount=0;
int outcount=0;
int exitcount = 0;

void setup() {
  Serial.begin (57600);
  pinMode(trigPin1, OUTPUT);
  pinMode(echoPin1, INPUT);
  pinMode(trigPin2, OUTPUT);
  pinMode(echoPin2, INPUT);
  // pinMode(LEDPin, OUTPUT); // Use LED indicator (if required)
}

void loop() {
  /* The following trigPin/echoPin cycle is used to determine the
   distance of the nearest object by bouncing soundwaves off of it. */
  digitalWrite(trigPin1, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin1, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin1, LOW);

  duration_in = pulseIn(echoPin1, HIGH);

  digitalWrite(trigPin2, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin2, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin2, LOW);
  duration_out = pulseIn(echoPin2, HIGH);

  //Calculate the distance (in cm) based on the speed of sound.
```

```

distance_in = duration_in/58.2;
distance_out = duration_out/58.2;

checkDistance(distance_in,distance_out);
}

void checkDistance(int distance1, int distance2)
{
  if (distance1 <= 80){
    if (outcount == 0){
      incount = 1;
      delay(700);
    }
    else if (outcount == 1){
      outcount = 0;
      count = count -1;
      Serial.println(count);
      delay(500);
    }
  }
  else if (distance2 <= 80){
    if (incount == 0){
      outcount = 1;
      delay(600);
    }
    else if (incount == 1){
      incount = 0;
      count = count + 1;
      Serial.println(count);
      delay(500);
    }
  }
  else {
    incount = 0;
    outcount = 0;
    delay(50);
  }
}

```

〈Android〉

manifests

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.androidtown.team3">

    <application
        android:allowBackup="true"
        android:icon="@drawable/icon"
        android:label="한국외국어대학교"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity" />

        <activity android:name=".time_table" />
        <activity android:name=".Main2Activity" />
        <activity android:name=".loading_bus"><intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
        </activity>
    </application>

    <uses-permission android:name="android.permission.INTERNET" />

</manifest>
```

Main2Activity

```
package org.androidtown.team3;

import android.os.Handler;
import android.os.Message;
import android.os.StrictMode;
import android.support.v4.widget.SwipeRefreshLayout;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.FrameLayout;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;

import static org.androidtown.team3.R.id.DOWN1;
import static org.androidtown.team3.R.id.DOWN2;
import static org.androidtown.team3.R.id.UP1;
import static org.androidtown.team3.R.id.UP1_state;
import static org.androidtown.team3.R.id.UP2;
import static org.androidtown.team3.R.id.UP2_state;
import static org.androidtown.team3.R.id.DOWN1_state;
import static org.androidtown.team3.R.id.DOWN2_state;
import static org.androidtown.team3.R.id.DOWN1_con;
import static org.androidtown.team3.R.id.DOWN2_con;
import static org.androidtown.team3.R.id.UP1_con;
import static org.androidtown.team3.R.id.UP2_con;
```

```

/*Import static org.androidtown.team3.R.id.Swipe:
import static org.androidtown.team3.R.id.tvSwipe:*/

public class Main2Activity extends AppCompatActivity {

    SwipeRefreshLayout swipeRefreshLayout;
    TextView textView;
    //    FrameLayout frameLayout;

    //Button btn;
    TextView tv1, tv1_state, tv1_con, tv2, tv2_con, tv2_state, tv3_con, tv3, tv3_state, tv4, tv4_con, tv4_state;

    // TCP연결 관련
    private Socket clientSocket;
    private BufferedReader socketIn;
    private PrintWriter socketOut;
    private int port = 50008;
    private final String ip = "13.209.49.254";
    private MyHandler1 myHandler1;
    private MyHandler2 myHandler2;
    private MyHandler3 myHandler3;
    private MyHandler4 myHandler4;
    private MyThread myThread;
    private String temp_id, temp_lo, temp_st, temp_ct;
    private String nobus = "현재 버스 정보가 없습니다.";
    int up = 0;
    int down = 0;
    private String mlist[] = {"모현 사거리", "지석묘입구", "기숙사", "도서관", "학생회관", "여문관", "인경관", "여문관",

```

```
        "학생회관", "공학관", "백년관", "기숙사", "지석묘입구", "모현사거리"});
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main2);

    /*swipeRefreshLayout = (SwipeRefreshLayout) findViewById(R.id.Swipe);
    textView = (TextView) findViewById(tvSwipe);*/

    StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
    StrictMode.setThreadPolicy(policy);

    try {
        clientSocket = new Socket(ip, port);
        socketIn = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
        socketOut = new PrintWriter(clientSocket.getOutputStream(), autoFlush: true);
    } catch (Exception e) {
        e.printStackTrace();
    }

    myHandler1 = new MyHandler1();
    myHandler2 = new MyHandler2();
    myHandler3 = new MyHandler3();
    myHandler4 = new MyHandler4();
    myThread = new MyThread();
    myThread.start();
    swipeRefreshLayout = (SwipeRefreshLayout) findViewById(R.id.Swipe);
```

```

tv1 = (TextView) findViewById(UP1);
tv1_state = (TextView) findViewById(UP1_state);
tv1_con = (TextView) findViewById(UP1_con);
tv2 = (TextView) findViewById(UP2);
tv2_state = (TextView) findViewById(UP2_state);
tv2_con = (TextView) findViewById(UP2_con);
tv3 = (TextView) findViewById(DOWN1);
tv3_state = (TextView) findViewById(DOWN1_state);
tv3_con = (TextView) findViewById(DOWN1_con);
tv4 = (TextView) findViewById(DOWN2);
tv4_state = (TextView) findViewById(DOWN2_state);
tv4_con = (TextView) findViewById(DOWN2_con);
//btn = (Button) findViewById(R.id.btn);
//tv = (TextView) findViewById(R.id.tv);
/*btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        socketOut.println(123);
    }
});*/
swipeRefreshLayout.setOnRefreshListener(() -> {
    up = 0;
    down = 0;
    tv1.setText(nobus);
    tv2.setText(nobus);
    tv3.setText(nobus);
    tv4.setText(nobus);
    tv1_state.setText("");
    tv2_state.setText("");

```



```

        tv3_state.setText("");
        tv4_state.setText("");
        tv1_con.setText("");
        tv2_con.setText("");
        tv3_con.setText("");
        tv4_con.setText("");

        socketOut.println("type:get#\r\npeople:");
        //tv.setText("인경관");

        new Handler().postDelayed(() -> {
            swipeRefreshLayout.setRefreshing(false);
        }, delayMillis: 1000);
    });
}

class MyThread extends Thread {
    @Override
    public void run() {
        while (true) {
            try {
                // InputStream의 값을 읽어와서 data에 저장
                String data = socketIn.readLine();
                int loc_idx = data.indexOf("location");
                int status_idx = data.indexOf("bus_status:");
                String loc, status;
            }
        }
    }
}

```

```

loc = data.substring(loc_idx + 9, status_idx);
if( Integer.parseInt(loc) < 7){
    if(up == 0) {
        Message msg = myHandler1.obtainMessage();
        msg.obj = data;
        myHandler1.sendMessage(msg);
        up++;
    }
    else{
        up = 0;
        Message msg = myHandler2.obtainMessage();
        msg.obj = data;
        myHandler2.sendMessage(msg);
    }
}
else if( Integer.parseInt(loc) >= 7){
    if(down == 0) {
        Message msg = myHandler3.obtainMessage();
        msg.obj = data;
        myHandler3.sendMessage(msg);
        down++;
    }
    else{
        down = 0;
        Message msg = myHandler4.obtainMessage();
        msg.obj = data;
        myHandler4.sendMessage(msg);
    }
}
}

```

```

    }
    catch (Exception e) {
        e.printStackTrace();
    }
}

}

class MyHandler1 extends Handler {
    @Override
    public void handleMessage(Message msg) {
        String str = msg.obj.toString();
        String loc, status, con;
        loc = str.substring(str.indexOf("location") + 9, str.indexOf("bus_status:"));
        status = str.substring( str.indexOf("bus_status:") + 11, str.indexOf("count:"));
        con = str.substring( str.indexOf("count:") + 6, str.length());
        if (Integer.parseInt(con) <= 10)
            con = "쾌적";
        else if (Integer.parseInt(con) <= 20)
            con = "여유";
        else if (Integer.parseInt(con) <= 30)
            con = "혼잡";
        else
            con = "꽉참";
        tv1.setText(mList[ Integer.parseInt(loc) - 1]);
        tv1_state.setText(status);
        tv1_con.setText(con);
    }
}

```

```

}
class MyHandler2 extends Handler {
    @Override
    public void handleMessage(Message msg) {
        String str = msg.obj.toString();
        String loc, status, con;
        loc = str.substring(str.indexOf("location") + 9, str.indexOf("bus_status:"));
        status = str.substring( str.indexOf("bus_status:") + 11, str.indexOf("count:"));
        con = str.substring( str.indexOf("count:") + 6, str.length());
        if (Integer.parseInt(con) <= 10)
            con = "쾌적";
        else if (Integer.parseInt(con) <= 20)
            con = "여유";
        else if (Integer.parseInt(con) <= 30)
            con = "혼잡";
        else
            con = "꽉참";
        tv2.setText(mList[ Integer.parseInt(loc) - 1]);
        tv2_state.setText(status);
        tv2_con.setText(con);
    }
}

class MyHandler3 extends Handler {
    @Override
    public void handleMessage(Message msg) {
        String str = msg.obj.toString();
        String loc, status, con;
        loc = str.substring(str.indexOf("location") + 9, str.indexOf("bus_status:"));

```

```

        status = str.substring( str.indexOf("bus_status:") + 11, str.indexOf("count:"));
        con = str.substring( str.indexOf("count:") + 6, str.length());
        if (Integer.parseInt(con) <= 10)
            con = "쾌적";
        else if (Integer.parseInt(con) <= 20)
            con = "여유";
        else if (Integer.parseInt(con) <= 30)
            con = "혼잡";
        else
            con = "꽉참";
        tv3.setText(mList[Integer.parseInt(loc) - 1]);
        tv3_state.setText(status);
        tv3_con.setText(con);
    }

class MyHandler4 extends Handler {
    @Override
    public void handleMessage(Message msg) {
        String str = msg.obj.toString();
        String loc, status, con;
        loc = str.substring(str.indexOf("location") + 9, str.indexOf("bus_status:"));
        status = str.substring( str.indexOf("bus_status:") + 11, str.indexOf("count:"));
        con = str.substring( str.indexOf("count:") + 6, str.length());
        if (Integer.parseInt(con) <= 10)
            con = "쾌적";
        else if (Integer.parseInt(con) <= 20)
            con = "여유";
        else if (Integer.parseInt(con) <= 30)
            con = "혼잡";
        else
            con = "꽉참";
        tv4.setText(mList[Integer.parseInt(loc) - 1]);
        tv4_state.setText(status);
        tv4_con.setText(con);
    }
}
}

```

MainActivity

```
package org.androidtown.team3;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button button6 = (Button) findViewById(R.id.button6);
        button6.setOnClickListener((view) -> {
            Intent intent = new Intent(getApplicationContext(), time_table.class);
            startActivity(intent);
        });

        Button button4 = (Button) findViewById(R.id.button4);
        button4.setOnClickListener((view) -> {
            Intent intent = new Intent(getApplicationContext(), Main2Activity.class);
            startActivity(intent);
        });

        Button btn_finish = (Button) findViewById(R.id.btn_finish);
        btn_finish.setOnClickListener((view) -> { finish(); });
    }
}
```

time_table

```
package org.androidtown.team3;

import ...

public class time_table extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_time_table);
    }
}
```

res/anim/loading.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <scale
        android:duration="400"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:interpolator="@android:anim/accelerate_decelerate_interpolator"
        android:pivotX="50%"
        android:pivotY="50%"
        android:repeatCount="infinite"
        android:repeatMode="reverse"
        android:toXScale="-1.0"
        android:toYScale="1.0" />
</set>
```

res/drawable/ic_launcher_background.xml

```
<?xml version="1.0" encoding="utf-8"?>
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="108dp"
    android:height="108dp"
    android:viewportHeight="108"
    android:viewportWidth="108">
    <path
        android:fillColor="#26A69A"
        android:pathData="M0,0h108v108h-108z" />
    <path
        android:fillColor="#00000000"
        android:pathData="M9,0L9,108"
        android:strokeColor="#33FFFFFF"
        android:strokeWidth="0.8" />
    <path
        android:fillColor="#00000000"
        android:pathData="M19,0L19,108"
        android:strokeColor="#33FFFFFF"
        android:strokeWidth="0.8" />
    <path
        android:fillColor="#00000000"
        android:pathData="M29,0L29,108"
        android:strokeColor="#33FFFFFF"
        android:strokeWidth="0.8" />
    <path
        android:fillColor="#00000000"
        android:pathData="M39,0L39,108"
        android:strokeColor="#33FFFFFF"
        android:strokeWidth="0.8" />
```



```
<path
    android:fillColor="#00000000"
    android:pathData="M49,0L49,108"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />

<path
    android:fillColor="#00000000"
    android:pathData="M59,0L59,108"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />

<path
    android:fillColor="#00000000"
    android:pathData="M69,0L69,108"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />

<path
    android:fillColor="#00000000"
    android:pathData="M79,0L79,108"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />

<path
    android:fillColor="#00000000"
    android:pathData="M89,0L89,108"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />

<path
    android:fillColor="#00000000"
    android:pathData="M99,0L99,108"
    android:strokeColor="#33FFFFFF"
```

```
        android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,9L108,9"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,19L108,19"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,29L108,29"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,39L108,39"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,49L108,49"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
```

```
<path
    android:fillColor="#00000000"
    android:pathData="M0,59L108,59"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,69L108,69"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,79L108,79"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,89L108,89"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,99L108,99"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
```

```
<path
    android:fillColor="#00000000"
    android:pathData="M19,29L89,29"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,39L89,39"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,49L89,49"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,59L89,59"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,69L89,69"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
```

```

<path
    android:fillColor="#00000000"
    android:pathData="M19,79L89,79"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M29,19L29,89"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M39,19L39,89"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M49,19L49,89"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M59,19L59,89"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />

```

```

<path
    android:fillColor="#00000000"
    android:pathData="M69,19L69,89"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
<path
    android:fillColor="#00000000"
    android:pathData="M79,19L79,89"
    android:strokeColor="#33FFFFFF"
    android:strokeWidth="0.8" />
</vector>

```

res/drawable/ic_launcher_foreground.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:aapt="http://schemas.android.com/aapt"
    android:width="108dp"
    android:height="108dp"
    android:viewportHeight="108"
    android:viewportWidth="108">
    <path
        android:fillType="evenOdd"
        android:pathData="M32,64C32,64 38,39,52,99 44,13,50,95C51,37,48,37 70,14,49,57 70,14,49,57L108,26,87,69L108,109,01L75,97,107,97L32,64Z"
        android:strokeColor="#00000000"
        android:strokeWidth="1">
        <aapt:attr name="android:fillColor">
            <gradient
                android:endX="78.5885"
                android:endY="90.9159"
                android:startX="48.7653"
                android:startY="61.0927"
                android:type="linear">
                <item
                    android:color="#44000000"
                    android:offset="0.0" />
                <item
                    android:color="#00000000"
                    android:offset="1.0" />
            </gradient>
        </aapt:attr>
    </path>
    <path
        android:fillColor="#FFFFFF"
        android:fillType="nonZero"
        android:pathData="M66,94,46,02L66,94,46,02C72,44,50,07,76,56,61,76,64L32,64C32,56,61,35,56,50,11,40,98,46,06L36,18,41,19C35,45,40,45,35,45,
        39,3,35,18,38,56C36,91,37,81,38,05,37,81,38,76,38,56L44,25,44,05C47,18,42,57,50,48,41,71,54,41,71C57,48,41,71,60,78,42,57,63,68,44,05L69,11,
        38,56C69,84,37,81,70,94,37,81,71,71,38,56C72,44,39,3,72,44,40,45,71,71,41,19L66,94,46,02M62,94,56,92C64,08,56,92,65,56,01,65,54,88C65,53,76,64,06,
        52,85,62,94,52,85C61,8,52,85,60,88,53,76,60,88,54,88C60,88,56,01,61,8,56,92,62,94,56,92M45,06,56,92C46,2,56,92,47,13,56,01,47,13,54,88C47,13,53,76,46,2,
        52,85,45,06,52,85C43,92,52,85,43,53,76,43,54,88C43,56,01,43,92,56,92,45,06,56,92"
        android:strokeColor="#00000000"
        android:strokeWidth="1" />
</vector>

```

res/layout/activity_loading_bus.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#F6F6F6"
    tools:context=".loading_bus">
    <ImageView
        android:id="@+id/img_android"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:src="@drawable/bus"
        app:layout_constraintBottom_toTopOf="@+id/imageView2"
        app:layout_constraintHorizontal_bias="0.502"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="1.0" />
```

```
    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="276dp"
        android:layout_height="65dp"
        android:layout_marginBottom="144dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:srcCompat="@drawable/ssing"
        />
</android.support.constraint.ConstraintLayout>
```

res/layout/activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#F6F6F6"
    tools:context=".MainActivity">

    <TableRow
        android:id="@+id/tableRow"
        android:layout_width="0dp"
        android:layout_height="76dp"
        android:layout_marginTop="8dp"
        android:background="#333366"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="1.0">

        <Button
            android:id="@+id/button4"
            android:layout_width="128dp"
            android:layout_height="80dp"
            android:background="#333366"
```



```
android:layout_marginBottom="0.5dp"
android:layout_marginEnd="0.5dp"
android:layout_marginLeft="0.5dp"
android:layout_marginRight="0.5dp"
android:layout_marginStart="0.5dp"
android:layout_marginTop="0.5dp"
android:text="버스 위치"
android:textColor="#FFFFFF"
android:textSize="15sp"
```

```
<Button
    android:id="@+id/button6"
    android:layout_width="128dp"
    android:layout_height="80dp"
    android:background="#333366"
    android:text="버스 시간표"
    android:textColor="#FFFFFF"
    android:textSize="15sp"
    android:layout_marginBottom="0.5dp"
    android:layout_marginEnd="0.5dp"
    android:layout_marginLeft="0.5dp"
    android:layout_marginRight="0.5dp"
    android:layout_marginStart="0.5dp"
    android:layout_marginTop="0.5dp"
/>
```

```

        android:layout_marginBottom="0.5dp"
        android:layout_marginEnd="0.5dp"
        android:layout_marginLeft="0.5dp"
        android:layout_marginRight="0.5dp"
        android:layout_marginStart="0.5dp"
        android:layout_marginTop="0.5dp"
        android:text=" 버스 위치"
        android:textColor="#FFFFFF"
        android:textSize="15sp"
    />

```

```

<Button
    android:id="@+id/button6"
    android:layout_width="128dp"
    android:layout_height="80dp"
    android:background="#333366"
    android:text=" 버스 시간표"
    android:textColor="#FFFFFF"
    android:textSize="15sp"
    android:layout_marginBottom="0.5dp"
    android:layout_marginEnd="0.5dp"
    android:layout_marginLeft="0.5dp"
    android:layout_marginRight="0.5dp"
    android:layout_marginStart="0.5dp"
    android:layout_marginTop="0.5dp"
/>

```

```

<Button
    android:id="@+id/btn_finish"
    android:layout_width="128dp"
    android:layout_height="80dp"
    android:background="#333366"
    android:text=" 종료"
    android:textColor="#FFFFFF"
    android:textSize="15sp"
    android:layout_marginBottom="0.5dp"
    android:layout_marginEnd="0.5dp"
    android:layout_marginLeft="0.5dp"
    android:layout_marginRight="0.5dp"
    android:layout_marginStart="0.5dp"
    android:layout_marginTop="0.5dp"
/>

```

```

</TableRow>

<ImageView
    android:id="@+id/ImageView3"
    android:layout_width="198dp"
    android:layout_height="189dp"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="212dp"
    app:layout_constraintBottom_toTopOf="@+id/tableRow"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.0"
    app:srcCompat="@drawable/bus_background" />

<ImageView
    android:id="@+id/ImageView4"
    android:layout_width="277dp"
    android:layout_height="212dp"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    app:layout_constraintBottom_toTopOf="@+id/ImageView3"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.505"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.0"
    app:srcCompat="@drawable/huffssing" />
</android.support.constraint.ConstraintLayout>

```

res/layout/activity_main2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Main2Activity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_marginEnd="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginTop="8dp"
        android:src="@drawable/download"
        app:layout_constraintBottom_toTopOf="@+id/Swipe"
```

```

        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.019" />

<android.support.v4.widget.SwipeRefreshLayout
    android:id="@+id/Swipe"
    android:layout_width="match_parent"
    android:layout_height="300dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.72">

    <FrameLayout
        android:id="@+id/frame"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"

```

```

app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.72">

<android.support.constraint.Guideline
    android:id="@+id/guideline"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    app:layout_constraintGuide_begin="20dp" />

<TableLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginLeft="8dp"
    android:layout_marginRight="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="@+id/imageView"

```

```
app:layout_constraintVertical_bias="0.318">
```

```
<TableRow
```

```
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#000000"
    android:padding="1dp">
```

```
    <TextView
```

```
        android:layout_width="wrap_content"
        android:layout_height="45dp"
```

```
        android:background="#cccccc"
        android:gravity="center"
        android:text="산행"
        android:textSize="17sp"
        android:textStyle="bold" />
```

```
    <TextView
```

```
        android:layout_width="136dp"
        android:layout_height="match_parent"
        android:text="절류장"
```

```
        android:background="#cccccc"
        android:gravity="center"
        android:textSize="12sp" />
```

```

<TextView
    android:layout_width="73dp"
    android:layout_height="match_parent"
    android:text="상위"

    android:background="#cccccc"
    android:gravity="center"
    android:textSize="12sp" />

<TextView
    android:layout_width="77dp"
    android:layout_height="match_parent"
    android:text="승격"
    android:background="#cccccc"
    android:gravity="center"
    android:textSize="12sp" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#000000"
    android:padding="1dp">

```



```

<ImageView
    android:layout_width="80dp"
    android:layout_height="45dp"
    android:background="#ffffff"
    android:gravity="center"
    android:src="@drawable/bluebus"
    android:text="1"
    android:textSize="12sp" />

<TextView
    android:id="@+id/UP1"
    android:layout_width="136dp"

    android:layout_height="match_parent"
    android:background="#ffffff"

    android:gravity="center"
    android:text="현재 버스 정보가 없습니다."
    android:textSize="12sp" />

<TextView
    android:id="@+id/UP1_state"
    android:layout_width="73dp"
    android:layout_height="match_parent"

```

```

        android:background="#ffffff"
        android:gravity="center"
        android:text=""
        android:textSize="12sp" />

        <TextView
            android:id="@+id/UP1_con"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:background="#ffffff"
            android:gravity="center"
            android:text=""
            android:textSize="12sp" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#000000"
        android:padding="1dp">

        <ImageView
            android:layout_width="80dp"
            android:layout_height="45dp"
            android:background="#ffffff"
            android:gravity="center"
            android:src="@drawable/redbus"

```

```
        android:text="1"  
        android:textSize="12sp" />
```

```
<TextView  
    android:id="@+id/UP2"  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
  
    android:background="#ffffff"  
    android:gravity="center"  
    android:text="현재 버스 정보가 없습니다."  
    android:textSize="12sp" />
```

```
<TextView  
    android:id="@+id/UP2_state"  
    android:layout_width="73dp"  
    android:layout_height="match_parent"  
  
    android:background="#ffffff"  
    android:gravity="center"  
    android:text=""  
    android:textSize="12sp" />
```

```
<TextView  
    android:id="@+id/UP2_con"  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"
```

```

        android:background="#ffffff"
        android:gravity="center"
        android:text=""
        android:textSize="12sp" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#000000"
    android:padding="1dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="45dp"

        android:background="#000000"
        android:gravity="center"
        android:text="하행"
        android:textSize="17sp"
        android:textStyle="bold" />

    <TextView
        android:layout_width="136dp"
        android:layout_height="match_parent"
        android:text="정류장"

```

```

        android:background="#cccccc"
        android:gravity="center"
        android:textSize="12sp" />

        <TextView
            android:layout_width="73dp"
            android:layout_height="match_parent"
            android:text="상태"

            android:background="#cccccc"
            android:gravity="center"
            android:textSize="12sp" />

        <TextView
            android:layout_width="77dp"
            android:layout_height="match_parent"
            android:text="승객"
            android:background="#cccccc"
            android:gravity="center"
            android:textSize="12sp" />
    </TableRow>

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#000000"
        android:padding="1dp">

```

```

<ImageView
    android:layout_width="80dp"
    android:layout_height="45dp"
    android:background="#ffffff"
    android:gravity="center"
    android:src="@drawable/bluebus"
    android:text="1"
    android:textSize="12sp" />

<TextView
    android:layout_width="136dp"
    android:id="@+id/DOWN1"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:gravity="center"
    android:text="현재 버스 정보가 없습니다."
    android:textSize="12sp" />

<TextView
    android:layout_width="73dp"
    android:layout_height="match_parent"
    android:id="@+id/DOWN1_state"
    android:background="#ffffff"
    android:gravity="center"
    android:text=""
    android:textSize="12sp" />

```

```

<TextView
    android:layout_width="73dp"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:gravity="center"
    android:id="@+id/DOWN1_con"
    android:text=""
    android:textSize="12sp" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="51dp"
    android:background="#000000"
    android:padding="1dp">

    <ImageView
        android:layout_width="80dp"
        android:layout_height="45dp"
        android:background="#ffffff"
        android:gravity="center"
        android:src="@drawable/greenbus"
        android:text="1"
        android:textSize="12sp" />

```

```

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:id="@+id/DOWN2"
            android:background="#ffffff"
            android:gravity="center"
            android:text="현재 버스 정보가 없습니다."
            android:textSize="12sp" />

        <TextView
            android:layout_width="73dp"
            android:layout_height="match_parent"
            android:id="@+id/DOWN2_state"
            android:background="#ffffff"
            android:gravity="center"
            android:text=""
            android:textSize="12sp" />

        <TextView
            android:layout_width="73dp"
            android:layout_height="match_parent"
            android:background="#ffffff"
            android:gravity="center"
            android:id="@+id/DOWN2_con"
            android:text=""
            android:textSize="12sp" />

    </TableRow>

</TableLayout>

</FrameLayout>
</android.support.v4.widget.SwipeRefreshLayout>
</android.support.constraint.ConstraintLayout>

```

res/layout/activity_time_table.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".time_table">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <ImageView
            android:id="@+id/imageView5"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:src="@drawable/timetable" />

    </LinearLayout>

    <Switch
        android:id="@+id/switch1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Switch"
        tools:layout_editor_absoluteX="0dp"
        tools:layout_editor_absoluteY="0dp" />

</android.support.constraint.ConstraintLayout>
```

res/mipmap/ic_launcher.xml

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
    <background android:drawable="@drawable/ic_launcher_background" />
    <foreground android:drawable="@drawable/ic_launcher_foreground" />
</adaptive-icon>
```

res/mipmap/ic_launcher_round.xml

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
  <background android:drawable="@drawable/ic_launcher_background" />
  <foreground android:drawable="@drawable/ic_launcher_foreground" />
</adaptive-icon>
```

res/values/colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <color name="colorPrimary">#333366</color>
  <color name="colorPrimaryDark">#303F9F</color>
  <color name="colorAccent">#FF4081</color>
</resources>
```

res/values/styles.xml

```
<resources>

  <!-- Base application theme. -->
  <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
  </style>

  <style name="AppTheme.NoActionBar">
    <item name="windowActionBar">false</item>
    <item name="windowNoTitle">true</item>
  </style>

  <style name="AppTheme.AppBarOverlay" parent="ThemeOverlay.AppCompat.Dark.ActionBar" />

  <style name="AppTheme.PopupOverlay" parent="ThemeOverlay.AppCompat.Light" />

</resources>
```

[참조]

iBeaconScanner(bluetooth_bluetooth), [https://github.com/switchdoclabs/iBeacon-Scanner-Android\(11\)_Python Server & Android Client Thread](https://github.com/switchdoclabs/iBeacon-Scanner-Android(11)_Python_Server_&_Android_Client_Thread), <http://mititch.tistory.com/36>
라즈베리 파이로 iBeacon Scanner로 만들기, <http://codeanalysis.tistory.com/2>
[국내] 장용식, 김관옥, 성낙현 (2011). 안드로이드 앱 개발, infinity Books
[국내] 천민국 (2013). (그림으로 쉽게 설명하는) 안드로이드 프로그래밍, 생능출판사