Hufstudy

(TEAM 5)

Ver. 1.0

2018. 06. 25.

한국외국어대학교

정보통신공학과

HUFSTUDY 팀

201601838 안세영

201202192 유정훈

201401075 문명기

201403586 허성윤

201600763 김주영

201603590 최초로

문서 정보

구분	소속	성명	날짜
작성자	한국외국어대학교 정보통신공학과	안세영	2018.06.22
	한국외국어대학교 정보통신공학과	유정훈	
	한국외국어대학교 정보통신공학과	문명기	
	한국외국어대학교 정보통신공학과	허성윤	
	한국외국어대학교 정보통신공학과	김주영	
	한국외국어대학교 정보통신공학과	최초로	
검토자	한국외국어대학교 정보통신공학과	안세영	2018.06.22
	한국외국어대학교 정보통신공학과	유정훈	
	한국외국어대학교 정보통신공학과	문명기	
	한국외국어대학교 정보통신공학과	허성윤	
	한국외국어대학교 정보통신공학과	김주영	
	한국외국어대학교 정보통신공학과	최초로	
승인자	한국외국어대학교 정보통신공학과	홍진표	2018.06.25

개정 이력

버전	작성자	개정일자	개정내역
1.0	안 세 영 유 정 훈 문 명 기 허 성 윤 김 주 영 최 초 로	2018.05.17	아이디어 회의 및 선정
	검토자	안세영, 유정훈, 문명기, 허성윤, 김주영, 최초로	
1.1	안 세 영 유 정 훈 문 명 기 허 성 윤 김 주 영 최 초 로	2018.05.21	Django 서버 구축작업 시작
	검토자	안세영, 유정훈, 문명기, 허성윤, 김주영, 최초로	
1.2	안 세 영 유 정 훈 문 명 기 허 성 윤 김 주 영 최 초 로	2018.05.24	구성도 빛 제안서 작성
	검토자	안세영, 유정훈, 문명기, 허성윤, 김주영, 최초로	
1.3	안 세 영 유 정 훈 문 명 기 허 성 윤 김 주 영 최 초 로	2018.05.28	아두이노와 라즈베리파이 연동
	검토자	안세영, 유정훈, 문명기, 허성윤, 김주영, 최초로	
1.5	안 세 영 유 정 훈 문 명 기 허 성 윤 김 주 영 최 초 로	2018.05.31	키패드 활성화 Django 서버 구축 70% 완성
	검토자	안세영, 유정훈, 문명기, 허성윤, 김주영, 최초로	
1.6	안 세 영 유 정 훈 문 명 기 허 성 윤 김 주 영 최 초 로	2018.06.05	데이터 베이스 구축
	검토자	안세영, 유정훈, 문명기, 허성윤, 김주영, 최초로	
2.0	안 세 영 유 정 훈 문 명 기 허 성 윤 김 주 영 최 초 로	2018.06.22	Django 서버 구축 완성 최종보고서 작성 및 검토
	검토자	안세영, 유정훈, 문명기, 허성윤, 김주영, 최초로	

머리말

본 문서는 4200 만 명이 사용하는 카카오톡의 플러스 친구를 이용한 예약 서비스를 기술한다. 사용자는 따로 앱을 설치하거나 오픈채팅방을 이용해 관리자에게 직접 예약하는 것이 아닌, 카카오톡 플러스친구를 추가하여 원하는 장소, 시간대를 선택해 예약하는 방법을 통해 스터디룸, 숙박업체 등을 예약할 수 있다. 기존의 오픈채팅방을 이용하던 방식보다 빠르고 편리하게 예약서비스를 이용할 수 있다.

목차

1. 개요 및 프로젝트 소개 6

1.1 개요 6

1.2 필요성과 목적 6

2. 프로젝트 구성 7

2.1 전체 시스템 구성도 7

2.1.1 유저와 서버 관점 8

2.1.2 서버와 스터디룸 관점 8

2.2 시퀀스 다이어그램 9

3. 시나리오 10

3.1 사용자 관점 시나리오 11

3.2 관리자 관점 시나리오 13

4. 기대효과 및 발전방향 **16**

5. 프로젝트 세부일정 **17**

6. 팀원 담당업무 **18**

7. [부록] Hufstudy 코드 **19**

8. 참고문헌 **59**

1. 개요 및 프로젝트 소개

1.1 개요

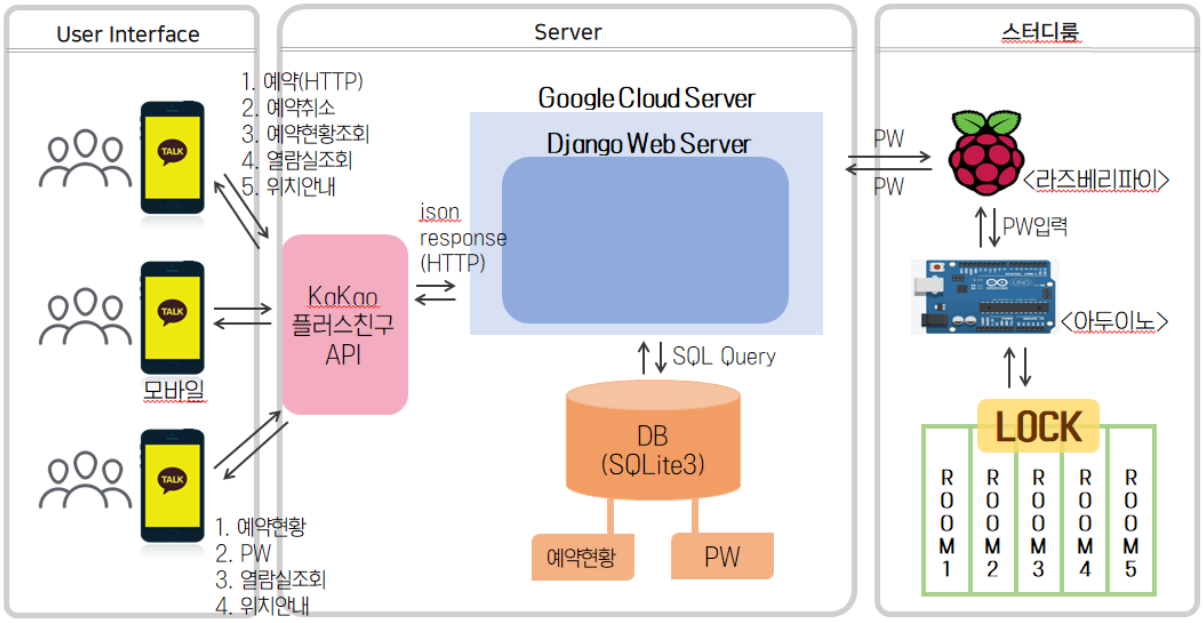
Hufstudy는 스터디룸 예약서비스이다. 관리자와 사용자의 스터디룸 예약에 관한 불편함을 해결하고자 고안해낸 Hufstudy는 카카오톡 플러스 친구를 이용하여 스터디룸의 현황을 파악해 사용하고자 하는 시간대에 맞춰 쉽고 빠르게 스터디룸을 예약할 수 있도록 도와준다.

1.2 필요성과 목적

현재 스터디룸을 이용하는데 있어서 사람들은 많은 불편함을 겪고 있다. 스터디룸을 이용하려면 관리자에게 스터디룸 키를 직접 수령해야 하며 이 과정에서 관리자가 자리를 비우고 있으면 사용자는 큰 불편함을 겪는다. 또한 정보통신공학과 스터디룸은 시간에 따라 다른 예약시스템을 가지고 있다. 오후 5시 이전에는 과 사무실을 통해 예약하고, 5시 이후에는 오픈 채팅방을 통해 예약을 하도록 되어있다. 그리고 공휴일에 이용이 불가하다는 점은 많은 사용자에게 불편함을 준다. 이러한 문제를 해결하기 위해 Hufstudy는 모든 사람들이 쉽게 이용할 수 있는 카카오톡 플러스친구 API를 사용하여 간단하게 스터디룸을 예약할 수 있게 한다

2. 프로젝트 구성

2.1 전체 시스템 구성도



2.1.1 유저와 서버 관점

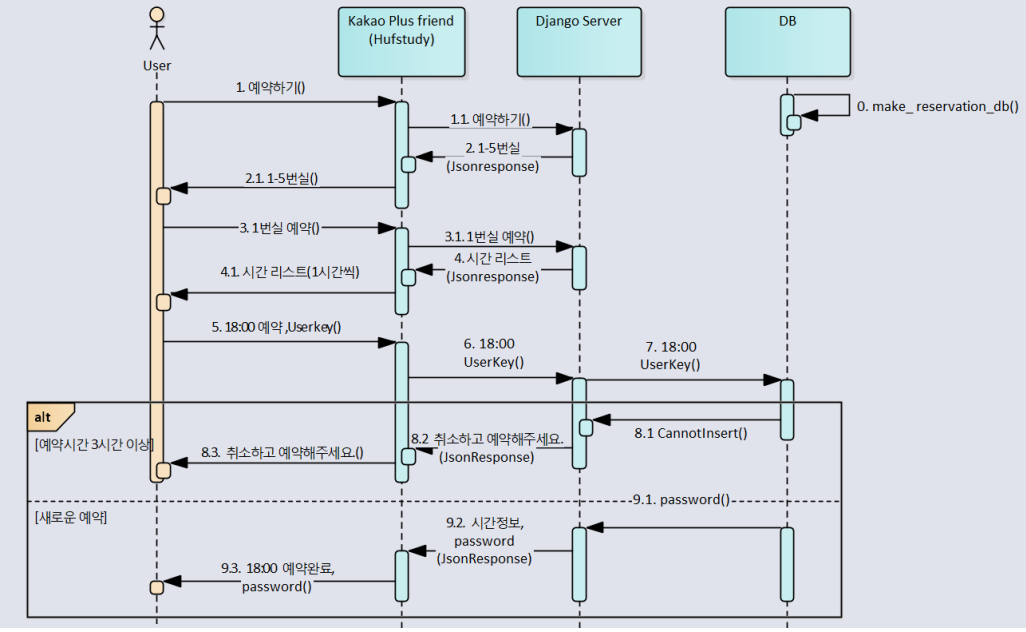
사용자가 누르는 버튼에 대한 값(Httprequest)은 django web server 에 전해지고 text나 그 다음 버튼 값을 Jsonresponse로 받게 된다. (HTTP 통신을 통해서 이는 초기에 Hufstudy API 설정 서버 주소와 포트 번호를 적게 된다.) 사용자가 오후 4시에 예약을 했을 때(초기에는 column[16:00(start_time),17:00(end_time),user_key(student), 예약시간(reservation_time), password]가 있다. Password는 매일 24시에 네자리수로 한번 random으로 만든다.) DB에서 16:00에 Password를 가져와서 사용자에게 보내준다.

2.1.2 서버와 스터디룸 관점

라즈베리파이는 django web server에게 매일 24시에 요청(request)해서 받은 시간당 비밀번호 전체를 txt 파일에 write 한 후 저장한다. txt file 을 read해서 시간에 맞춰 아두이노에게 시리얼통신으로 보낸다. 아두이노에 연결된 도어락의 비밀번호는 시리얼통신으로 받은 비밀번호가 된다.

구분	이름	설명
하드웨어	Raspberry Pi	리눅스 커널 기반 운영 체제를 사용하는 초소형 pc이며 Raspbian이라는 Raspberry Pi에 최적화된 데비안 계열의 자유 운영 체제가 현재로서는 가장 권장되는 시스템
	Arduino	오픈소스를 기반으로 한 단일 보드 마이크로 컨트롤러로 완성된 보드(상품)와 관련 개발 도구 및 환경
	서버용 컴퓨터	서버를 올리기 위한 서버용 컴퓨터
소프트웨어	linux ubuntu	서버 컴퓨터의 OS
	python3	시스템의 전체 통합적인 로직을 담당하는 프로그래밍 언어
	jupyter notebook	팀 repository 및 디버깅 환경을 제공
	pycharm	파이썬 개발 툴
	python anaconda	파이썬 개발 툴
	django	Django는 Python으로 만들어진 무료 오픈소스 웹 애플리케이션 프레임워크(web application framework)
	sqlite3	mysql이나 postgreSQL와 같은 dbms 지만 서버가 아니라 응용프로그램에 넣어 사용하는 비교적 가벼운 데이터베이스
	javascript	웹 언어
	bs4	HTML 및 XML 파일에서 데이터를 가져오는 python 라이브러리
	Google cloud platform	클라우드 플랫폼 제공자들은 단순 웹사이트에서부터 복잡한 애플리케이션에 이르는 일련의 프로그램들을 빌드하기 위한 개발자 제품들을 제공
	카카오톡 API	앱 개발 환경 제공

2.2 시퀀스 다이어그램



0. DB 속의 Password는 24시간이 지나면 업데이트된다.

1. 사용자가 예약하기 버튼을 누른다.

2. hufstudy가 1-5번실 버튼을 띄운다.

3. 사용자가 (예를 들어)1번실을 예약한다.

4. 01:00- 24:00까지 시간 리스트 버튼을 띄운다.

5. 사용자가 18시(18-19시)에 예약한다..

6. hufstudy가 18시와 userkey를 서버에 보내준다.

7. 18시와 userkey를 현재시간과 함께 DB 에 넣는다.

8. 1-3. 만약에 같은 userkey 사용자가 3시간이상 예약되어 있으면 예약되어 있다고 jsonresponse로 보내준다.

9.1. 3시간 이하 예약 대상자 이면 해당 시간대에 password를 같은 방식으로 보내준다.

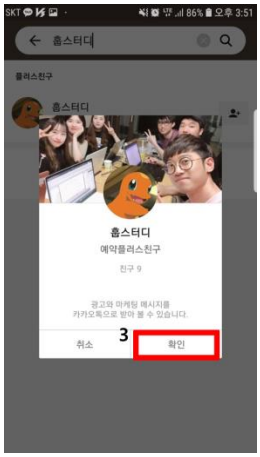
9.2-3. 예약완료 text 와 함께 password를 알려준다.

3. 시나리오

3.1 사용자 관점 시나리오

① 카카오톡 앱을 실행한 뒤, 상단의 검색창에 [홉스터디]를 검색한다.

② 검색된 플러스친구의 오른쪽에 플러스 버튼을 누르면 사용자의 플러스 친구에 추가된다.



③ '홉스터디'이름을 클릭하면 다음과 같은 '홉스터디' 정보와 버튼이 뜬다. 예약하려면 확인버튼을 눌러준다.

④ 확인버튼을 누르면 다음과 같이 채팅방이 실행되고 자동응답 메시지에서 스터디룸 사용방법 및 규칙이 보내 진다.

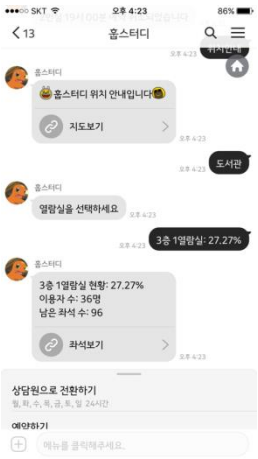
⑤ 메시지를 읽은 후, 하단에 보면 메뉴 바들이 있다. [예약하기], [예약취소], [나의 예약현황], [위치안내], [도서관]

⑥ [예약하기]버튼을 누르면 예약하고 싶은 스터디룸을 선택할 수 있다.

⑦ [2번실]을 선택했다면 예약할 수 있는 시간대가 다음과 같이 나타난다.

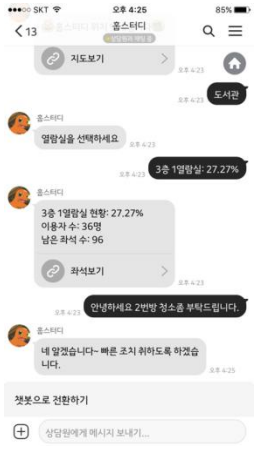
⑧ 스터디룸과 시간대까지 선택을 마치면 다음과 같은 예약정보가 암호와 함께 자동응답 메시지로 전송된다.

⑨ [나의 예약현황] 버튼을 클릭하면 사용자가 예약한 스터디룸과 시간대가 암호와 함께 나타난다. 만약 사정이 생겨 예약을 취소하거나 시간을 바꾸려 하는 경우, [예약취소]버튼을 클릭한다. '예약취소 하시겠습니까?' 라는 메시지가 도착하고 [예] 버튼을 클릭하면 사용자가 예약한 룸과 예약시간 정보가 취소된다.



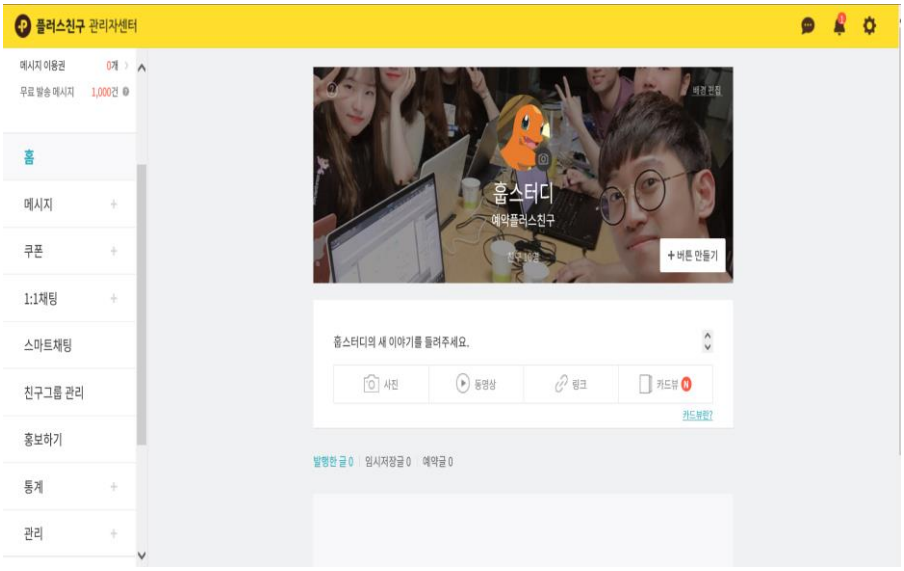
⑩ 사용자가 예약하고자 하는 스터디룸의 위치를 알고 싶으면 [위치안내] 버튼을 누른다. URL이 보내지면 스터디룸의 구글 맵 정보가 뜬다. [도서관] 버튼을 클릭하면 현재 학교 도서관의 열람실 현황이 뜬다. 또한 좌석보기 링크를 통해 어떤 자리에 이용 중인지 실시간으로 확인할 수 있다.

⑪ 메뉴바의 가장 상단에 위치한 [상담원으로 전환하기] 를 클릭하면 상담원과 직접 상담을 할 수 있다.

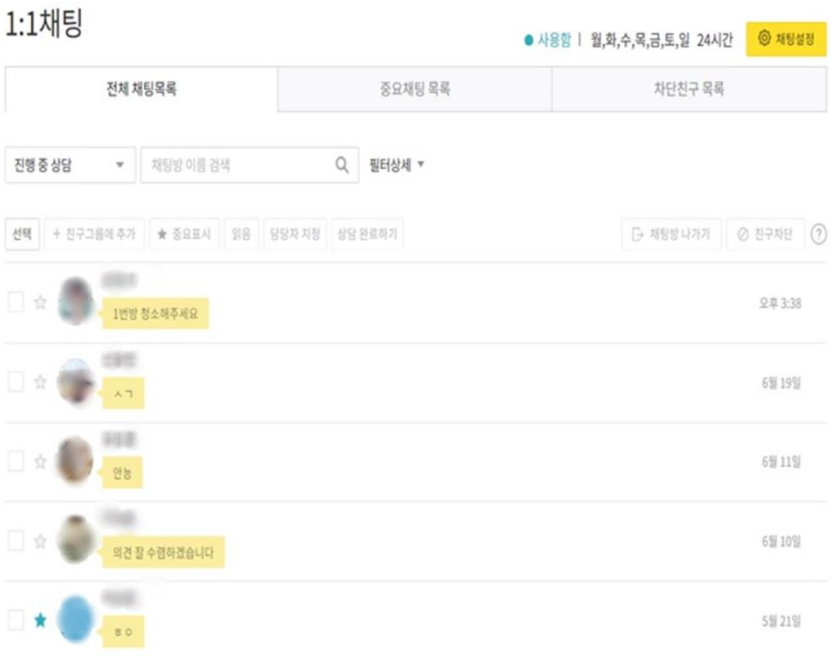


⑫ 건의사항이나 문의사항 등 상담원과 자유롭게 대화한 뒤, 챗봇으로 돌아가려면 [챗봇으로 전환하기] 버튼을 클릭하면 된다.

3.2 관리자 관점 시나리오

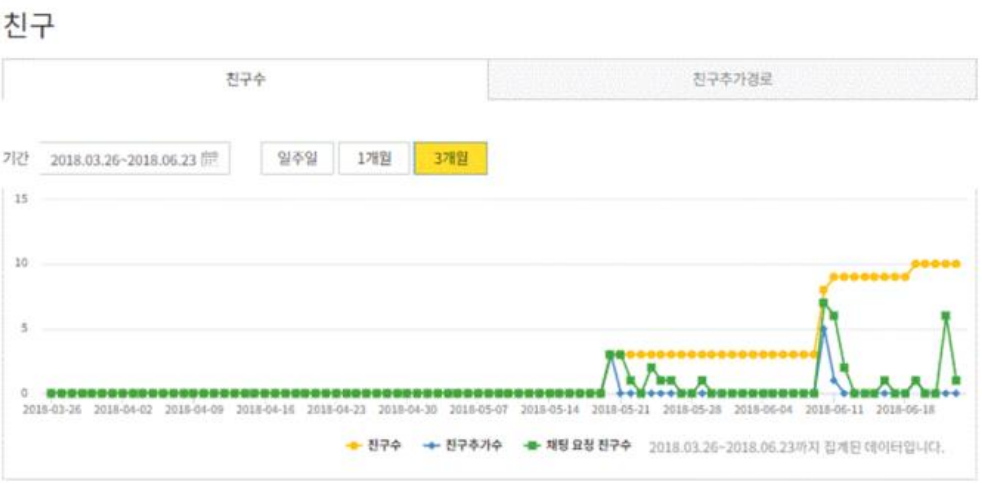


① 관리자는 플러스친구 관리자 센터 페이지에서 예약서비스에 대한 플러스친구를 관리할 수 있다.

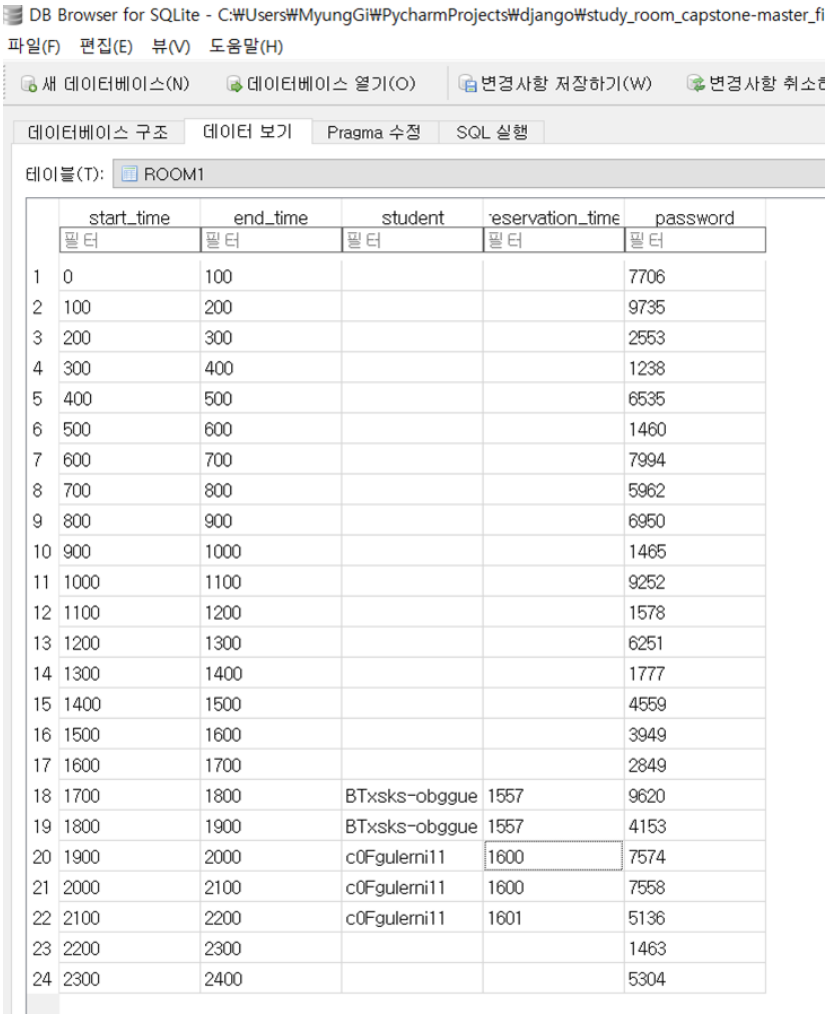


② 관리자는 사용자A에게 1:1 채팅 문의가 올 때마다 24시간 실시간 채팅으로 사용자들과의 상담을 통해 사용자들의 불편한 사항에 대해 빠른 조치를 취했다.

③ 또한 관리자는 플러스친구 관리자 페이지에서 친구추가 한 사용자의 통계를 보며 얼마나 많은 사용자들이 홉스터디를 이용하고 있는지 확인하며 더 나은 서비스 제공을 위해 힘쓴다.



④ 일주일, 1개월, 3개월 간격으로 홉스터디를 친구 추가한 수, 채팅요청을 한 친구 수, 총 사용자 수를 그래프로 확인하며 사용자관리 한다.



	start_time	end_time	student	reservation_time	password
	필터	필터	필터	필터	필터
1	0	100			7706
2	100	200			9735
3	200	300			2553
4	300	400			1238
5	400	500			6535
6	500	600			1460
7	600	700			7994
8	700	800			5962
9	800	900			6950
10	900	1000			1465
11	1000	1100			9252
12	1100	1200			1578
13	1200	1300			6251
14	1300	1400			1777
15	1400	1500			4559
16	1500	1600			3949
17	1600	1700			2849
18	1700	1800	BTxsks-obggue	1557	9620
19	1800	1900	BTxsks-obggue	1557	4153
20	1900	2000	c0Fgulerni11	1600	7574
21	2000	2100	c0Fgulerni11	1600	7558
22	2100	2200	c0Fgulerni11	1601	5136
23	2200	2300			1463
24	2300	2400			5304

⑤ 관리자는 디비에 어떠한 값들이 들어오는지 확인 할 수 있고 24시간마다 바뀌는 비밀번호들을 다 확인하고 관리한다. 또한 카카오톡에서 개인마다 부여되는 user key를 통해 사용자들이 어떠한 시간에 예약을 했고 어떠한 시간이 비어있는지 확인한다. 이렇게 관리자 입장에서는 관리자 페이지를 통해 총 사용자 수와 사용자와의 1:1채팅, db관리를 통해 예약서비스 플러스친구를 관리할 수 있게 된다.

4. 기대효과 및 발전방향

본 제품은 기존에 있던 스터디룸 예약 서비스에서 간단하게 이용할 수 있는 카카오톡을 통한 예약서비스를 제공함으로써 관리자와 사용자 모두 편리한 서비스를 이용할 수 있게 된다. 접근성이 좋은 카카오톡 플러스친구를 통한 예약방식으로, 예약 매시마다 관리자가 필요하지 않아 인건비가 크게 절감되는 효과를 가지고 있다. 또한 무인 시스템이기 때문에 평일 공휴일에 제약 받지 않고 24시간 이용가능 하게 된다. 이전에는 관리자에게 직접 키를 수령하여 사용했지만 본 제품을 사용하게 되면 키를 직접 받으러 가지 않아도 되는 편리함과 시간 효율성이 제공되어 기존 관리자와 사용자 모두에게 긍정적인 효과를 얻을 수 있다.

오늘날 첨단 시스템이 상용화되면서 식당, 편의점 등등에 무인시스템이 도입되고 있다. 우리 서비스도 같은 무인 시스템이지만 스마트폰만을 이용한다는 점에서 큰 장점을 갖고 있다. 따라서 병원진료, 고속버스, 열차 등 예약 서비스에도 적용이 가능하다. 더 나아가 사업자등록자와 계약이 체결된다면 현재 보유한 기술력으로 무인 숙박 예약시스템도 바로 사용할 수 있다.

5. 프로젝트 세부일정

기간 / 개발내용	개발 단계			
	5월		6월	
	5.17 ~ 5.31		6.1 ~ 6.24	
아이디어 회의 및 결정				
제안서 작성 및 검토				
Raspberry Pi, Arduino 연결				
Kakao API				
Django				
Connection				
Database				
최종보고서 작성				
최종보고서 검토				

6. 팀원 담당업무

담당자	상세내용
안세영	DB / 카카오톡 플러스친구
유정훈	Raspberry Pi, Arduino
문명기	Django 서버 / 보고서
허성윤	Django 서버 / 보고서
김주영	카카오톡 플러스친구 / 보고서
최초로	Raspberry Pi, Arduino / PPT

7. [부록] Hufstudy 코드

7.1 Manage.py

```
#!/usr/bin/env python
import os
import sys


if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE", "study_room_capstone.settings")
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)
```

7.2 Auto_rasp.py

```
import requests
import json


room_number = 2


room_path = "./DB/room"


data = {'room': room_number,
                }
```

```
data_json = json.dumps(data)


r = requests.get('http://35.200.7.249:8000/all_pw', data=data_json)
#r = requests.get('http://127.0.0.1:8000/all_password', data=payload)
pw_dict = r.json()


with open('password.json', 'w', encoding="utf-8") as f:
    json.dump(pw_dict, f)
   print(r.json())
```

7.3 db_func.py

```
import sqlite3
import os
import datetime
import random
import requests
import json
from django.http import JsonResponse


"""
#make_reservation_db : DB폴더없을때 생성해줌
#reservation : room DB연결하여 예약생성
#reserve_main :


"""



def make_reservation_db():
    """
```

3시간 간격으로 DB 컬럼 채운다.

start_time | end_time | student | reservation_time 으로 구성되어있으며

start_time과 end_time은 개발자가 지정(0930~2130), student는 예약자 이름,

reservation_text는 사용자 예약당시의 시간이다.

```
"""
path = "../DB"
os.makedirs(path, exist_ok=True)
os.remove("../DB/room")
con = sqlite3.connect('../DB/room')

cur = con.cursor()

for i in range(1, 6):
    cur.execute("CREATE TABLE ROOM" + str(
        i) + "(start_time text, end_time text, student text, reservation_time text, password text)")

    time_res = 100
    for j in range(0, 2400, time_res):
        randpassword = random.randrange(1000, 10000)
        cur.execute("INSERT INTO ROOM" + str(
            i) + "
Values(:start_time, :end_time, :student, :reservation_time, :password);",
                    {"start_time": j, "end_time": j + time_res, "student": '',
"reservation_time": '',
                     "password": str(randpassword)})
```

```
        con.commit()
        con.close()


def reservation(request=None):
        # NO reservation_time student start_time
        # 정상이면 0 리턴, 비정상 종료면 1리턴, 기존예약이 있으면 5 리턴
        # 시간지났으면 7리턴
        con = sqlite3.connect('./DB/room')
        cur = con.cursor()
        select = []

        for j in range(1, 6):
            user_select_room = cur.execute("SELECT start_time FROM ROOM{} WHERE
student = {!r}"
                                           .format(str(j), request['student']))
            for i in user_select_room.fetchall():
                select.append(i)
        print(select)

        time = datetime.datetime.now().strftime("%H%M")
        if int(request['start_time']) < int(time):
            return {"code": 7}
        # 안비어으면 예약안되게 한다.
        # 예약한 스터디룸이 3  개를 넘어선다 ==> 예약 더이상 안되게 한다.
        if len(select) > 2:
            print(select)
```

```
            con.close()
            return {"code": 5}

        now_room = cur.execute("SELECT reservation_time FROM ROOM{0} WHERE start_time = {1};"
                               .format(str(request['NO']), str(request['start_time'])))

        now_room = now_room.fetchall()
        # now_room 이 비어있으면 UPDATE 없으면 그대로 리턴
        if now_room == [('',)]:
            cur.execute("UPDATE ROOM{} SET reservation_time = {}, student = {!r} WHERE start_time = {};"
                        .format(str(request['NO']), str(request['reservation_time']), str(request['student']),
                                str(request['start_time'])))
            # sqlite3 포매팅에 !r을 붙이지않으면 unrecognized token 오류가 난다.
        else:
            print(now_room)
            con.close()
            return {"code": 1}

        room_password = cur.execute("SELECT password FROM ROOM{} WHERE start_time = {};"
                                    .format(str(request['NO']), str(request['start_time'])))
        room_password = room_password.fetchall()
        con.commit()
        con.close()
        print(room_password[0][0])
        return {"code": 0, "password": room_password[0][0]}
```

```
    # code 정상종료 확인코드, password 스터디룸 비밀번호



def search_reservation(request=None):
    con = sqlite3.connect('./DB/room')
    cur = con.cursor()

    # now_time = datetime.datetime.now()
    # now_time = int(now_time.strftime("%H%M%S"))

    available_list = {}

    for i in range(1, 6):
        available_list['ROOM' + str(i)] = []
        # 스터디룸 갯수별로 딕셔너리에 리스트 타입으로 만들고 append
        cur.execute("SELECT start_time FROM ROOM{} WHERE reservation_time = '';"
                    .format(str(i)))
        # reservation_time 이 비어있으며, 해당 시간대가 현재 시간보다 나중일 경우만 선택
        # 시간 나중인것은 후에 구현하기로 함. 2018-06-02 21:31
        for cell in cur.fetchall():
            available_list['ROOM' + str(i)].append(cell[0])
            # {'ROOM1': ['1230', '1530', '1830'], 'ROOM2': ['930', '1230', '1530', '1830'], 'ROOM3': ['1530', '1830'],
            # 'ROOM4': ['930', '1230', '1530', '1830'], 'ROOM5': ['930', '1230', '1530', '1830']}

    con.close()
    # print(available_list)
```

```
    return available_list


def my_reservation(request=None):
    # student키
    # 나의 예약현황
    reservation_list = []
    # reservation_list 에 [방번호, 사용할 시간, 예약당시 시간]
    con = sqlite3.connect("./DB/room")
    cur = con.cursor()

    for i in range(1, 6):
        cur.execute("SELECT start_time, reservation_time, password, end_time FROM
ROOM{} WHERE student = {!r};"
                    .format(str(i), request['student']))
        for j in cur.fetchall():
            time = datetime.datetime.now().strftime("%H%M")
            if int(j[3]) > int(time):
                reservation_list.append(['ROOM{}'.format(i), j[0], j[1], j[2]])
            else:
                pass
    print(reservation_list)

    return reservation_list
    # [['ROOM1', '930', '110', PASSWORD]]
```

```
def cancel_reservation(request=None):
    # 리턴 1은 비정상적 종료,
    # student키
    con = sqlite3.connect("./DB/room")
    cur = con.cursor()

    prev_cancel = my_reservation({"student": request['student']})
    print("예약 ㅣ : ", prev_cancel)
    if prev_cancel == []:
        return "예약한 방이 없습니다"
    try:
        for i in range(len(prev_cancel)):
            cur.execute("UPDATE ROOM{} SET student = '', reservation_time = '' WHERE start_time = {};"
                        .format(prev_cancel[i][0][4:], prev_cancel[i][1]))

    except Exception as e:
        con.close()
        print(e)
        return "예약한 방이 없습니다"
    con.commit()
    con.close()

    text = ""
    for i in range(len(prev_cancel)):
        time = prev_cancel[i][1]

        if len(time) == 1:
```

```
            time = "{}시 00분".format(time)
        elif len(time) == 3:
            time = "{}시 {}분".format(time[:1], time[1:])
        elif len(time) == 4:
            time = "{}시 {}분".format(time[:2], time[2:])

        text += "{}번실 {} 예약 취소되었습니다₩n" ₩
            .format(prev_cancel[i][0][4:], time)
    return text


def rasp_password(request):
    # {'room': 방번호, 'start_time': 시작시간
    # {'room': 1, 'start_time': 200}
    # 1번실 오전2시

    # print(request.body)
    con = sqlite3.connect("./DB/room")
    cur = con.cursor()

    json_reqeust = (request.body).decode('utf-8')
    json_res = json.loads(str(json_reqeust))
    print(json_res)
    # json_res = {'room': 3, 'start_time': 1300}

    cur.execute("SELECT password FROM ROOM{} WHERE start_time = {}"
                .format(str(json_res['room']), str(json_res['start_time'])))
```

```
        password = cur.fetchall()
        print(password)
        return JsonResponse({
            'password': (password[0])
        })


def all_rasp_password(request):
    """

    :param request: request = {"room": 2}
    :return:
    """

    json_request = (request.body).decode('utf-8')
    json_res = json.loads(json_request)

    # json_res = {'room': 3}

    all_password_dict = {}
    try:
        con = sqlite3.connect("./DB/room")
        cur = con.cursor()

        cur.execute("SELECT password, start_time FROM ROOM{}"
                    .format(str(json_res['room'])))

        all_password = cur.fetchall()
```

```
            # print(all_password)


        except Exception as e:
            print("ERROR all_rasp_password execute")
            print(e)
            pass
        for pw, st in all_password:
            all_password_dict[str(st)] = pw


        # print(all_password_dict)


        return JsonResponse(all_password_dict)



if __name__ == "__main__":
    """
    cron

    0 6 * * * /~~~/study_room/db_func.py
    """
    make_reservation_db()
        # search_reservation()
```

7.4 app_urls.py

```
from django.contrib import admin
from django.conf.urls import url
from django.urls import include
from study_room import views
```

```
from .db_func import rasp_password
from .db_func import all_rasp_password


urlpatterns = [
    url(r'keyboard', views.keyboard),
    url(r'message', views.message),
    url(r'password', rasp_password),
    url(r'all_pw', all_rasp_password)
   ]
```

7.5 Library_crawl.py

```
import urllib.request
from bs4 import BeautifulSoup



def glo_library(name):

    try:
        req = urllib.request.urlopen('http://203.232.237.8/domian5/2/domian5.asp')
    except:
        return 555


    soup = BeautifulSoup(req, 'lxml', from_encoding="utf-8")


    my_titles = soup.select(
        'tr'
    )
    data = []
```

```
    for title in my_titles:
        data.append(title.text)

    # data[3]은 3층 1열람실 내용
    # [3~7]까지의 (6)
    # #return (data[num+2].split()
    if name == "도서관":
        return {'3-1': data[3].split()[6], '3-2': data[4].split()[6], '4-3A': data[5].split()[6], '4-3B': data[6].split()[6]}
    else:
        # #print(data[name+2].split())
        return {'%': data[name+2].split()[6], '이용자': data[name+2].split()[4], '남은 좌석': data[name+2].split()[5]}


def seo_library(name):

    try:
        req = urllib.request.urlopen('http://203.232.237.8/domian5/domian5.asp')
    except:
        return 555

    soup = BeautifulSoup(req, 'lxml', from_encoding="utf-8")

    my_titles = soup.select(
        'tr'
    )
```

```
    data = []


    for title in my_titles:
        data.append(title.text)


    # #return data[num+2].split()
    if name == '도서관':
        return {'4-1A': data[3].split()[6], '4-1B': data[4].split()[6], '4-2': data[5].split()[6], '5-3A': data[6].split()[6],
                '5-3B': data[7].split()[6], '5-4': data[8].split()[6]}
    else:
        # #print(data[name+2].split())
            return {'%': data[name+2].split()[6], '이용자': data[name+2].split()[4], '남은좌석': data[name+2].split()[5]}
```

7.6 Views.py

```
from .db_func import make_reservation_db
from .db_func import reservation
from .db_func import search_reservation
from .db_func import cancel_reservation
from .db_func import my_reservation
from . import library_crawl


import urllib.request
from bs4 import BeautifulSoup
import json
import os
import sqlite3
import datetime
```

```
import random
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt


def keyboard(request):


    print(request.body)
    menus = ['예약하기', '예약취소', '나의 예약현황', '위치안내', '도서관']


    return JsonResponse({
        'type': "buttons",
        "buttons": menus,
    })


@csrf_exempt
def message(request):
    menus = ['예약하기', '예약취소', '나의 예약현황', '위치안내', '도서관']
    study_room_menu = ['1번실', '2번실', '3번실', '4번실', '5번실']
    emoti = '(멘붕)', '(깜짝)', '(허걱)', '(부르르)', '(훌쩍)', '(심각)', '(헉)', '(열받아)', '(빠직)', '(짜증)', '(정색)', '(힘듦)'
    emoti2 = '(찡긋)', '(아잉)', '(뿌듯)', '(쑥스)', '(뽀뽀)', '(감동)', '(굿)', '(꺄아)', '(좋아)','(수줍)', '(컴온)', '(발그레)', '(하하)', '(우와)', '(하트뿅)', '(씨익)', '(신나)'


    req = (request.body).decode('utf-8')
    received_json = json.loads(req)


    content_name = received_json['content']
    user_name = received_json['user_key']
```

```
now_date = datetime.datetime.now()
now_date = now_date.strftime("%H%M")
## 183700 > 164700 + 30000
# 18시37분00초

if not os.path.exists("./DB"):
    make_reservation_db()  # 생성하게끔
    # DB폴더없으면

if content_name == '나의 예약현황':
    reservation_MyStatus = my_reservation({"student": user_name})
    # 리턴형식 : [방번호, 사용할 시간, 예약당시 시간, 패스워드]
    text = "사용자의 예약현황₩n"
    for cell in reservation_MyStatus:
        if len(cell[1]) == 3:
            time = "{}시 {}분".format(cell[1][:1], cell[1][1:])
        else:
            time = "{}시 {}분".format(cell[1][:2], cell[1][2:])

        text += "{} - {}₩n비밀번호 : {}₩n".format(cell[0], time, cell[3])

    return JsonResponse({
        "message": {
            "text": text
        },
        "keyboard": {
            "type": "buttons",
```

```
                "buttons": menus
            }
        })

    elif content_name == '예약하기':
        # '예약하기' => '번실' => '번실 - '
        me = ['취소', '1번실', '2번실', '3번실', '4번실', '5번실']
        return JsonResponse({
            "message": {
                "text": "스터디룸을 선택해주세요"
            },
            "keyboard": {
                "type": "buttons",
                "buttons": me
            }
        })

    elif content_name == "위치안내":

        return JsonResponse({
            'message': {
                'text': random.choice(emoti2) + '홉스터디 위치 안내입니다' + random.choice(emoti2),
                'message_button': {
                    'label': '지도보기',
                    'url': 'https://www.google.co.kr/maps/place/%ED%95%9C%EA%B5%AD%EC%99%B8%EA%B5%AD%E'
```

'C%96%B4%EB%8C%80%ED%95%99%EA%B5%90+%EA%B8%80%EB%A1%9C%EB%B2%8C%EC%BA%A0%ED%8'

'D%BC%EC%8A%A4/@37.3383934,127.2672934,17z/data=!4m5!3m4!1s0x357b5562395fd07d:'

'0x7ef3f3135ee53b77!8m2!3d37.3383892!4d127.2694874'

}

},

'keyboard': {

'type': 'buttons',

'buttons': menus

}

})

elif '번실 - ' in content_name and len(content_name) < 20:

time_sen = str(content_name[6:])

time_sen = time_sen.replace('시', '').replace('분','').replace(' ','')

reserve_data = {

"NO": content_name[:1],

"reservation_time": now_date,

"student": user_name,

"start_time": time_sen,

}

result = reservation(reserve_data)

정상 0, 비정상종료는 1, 예약한 곳이 있으면 5 리턴

```
    if result["code"] == 0:
        return JsonResponse({
            "message": {
                "text": "{} 예약되었습니다₩n비밀번호 : {}"
                    .format(content_name, result["password"])
            },
            "keyboard": {
                "type": "buttons",
                "buttons": menus
            }
        })
    elif result["code"] == 1:
        return JsonResponse({
            "message": {
                "text": "예약중 오류가 발생했습니다₩n개발자에게 말씀해주세요"
            },
            "keyboard": {
                "type": "buttons",
                "buttons": menus
            }
        })
    elif result["code"] == 5:
        return JsonResponse({
            "message": {
                "text": "이미 3시간을 모두 소진하였습니다₩n취소후 다시
예약해주세요"
            },
            "keyboard": {
```

```
                    "type": "buttons",
                    "buttons": menus
                }
            })
        elif result["code"] == 7:
            return JsonResponse({
                "message": {
                    "text": "이미 지난 시간입니다."
                },
                "keyboard": {
                    "type": "buttons",
                    "buttons": menus
                }
            })
        elif '취소' == content_name:
            return JsonResponse({
                "message": {
                    "text": "취소를 합니다₩n메인으로 이동합니다"
                },
                "keyboard": {
                    "type": "buttons",
                    "buttons": '취소'
                }
            })

    elif '번실' in content_name and len(content_name) < 20:
        available_list = search_reservation()
        room_no = content_name[:1]
```

```
can_select = ['취소']
print(available_list)
time = datetime.datetime.now().strftime("%H%M")
for can_time in available_list['ROOM' + room_no]:
    if int(can_time) < int(time):
        continue
        #시간지나면 예약화면 안뜨게
    print(can_time)
    if len(can_time) == 1:
        can_time = "{}시 {}분".format(can_time[0:1], "00")
    elif len(can_time) == 3:
        can_time = "{}시 {}분".format(can_time[0:1], can_time[1:])
    elif len(can_time) == 4:
        can_time = "{}시 {}분".format(can_time[0:2], can_time[2:])

    # can_select는 가능한 호실들 response하기위한 목록
    # can_time은 available_list 해당호실의 예약가능한 시간들
    can_select.append("{} - {}".format(content_name, can_time))
        # '2번실 - 9시 30분'
        # 분류는 '번실 - ' 로 하면된다.
#print(can_select)
#print('₩n₩n')
print(can_select)

mes_text = "예약 화면"

if can_select == ['취소']:
    mes_text = "예약 가능한 방이 없습니다" + random.choice(emoti)
```

```
            can_select = menus
        elif can_select == '취소':
            can_select = menus

        return JsonResponse({
            "message": {
                "text": mes_text
            },
            "keyboard": {
                "type": "buttons",
                "buttons": can_select
            }
        })

    elif '예' == content_name:
        res = cancel_reservation({"student": user_name})
        return JsonResponse({
            "message": {
                "text": "{}".format(res)
            },
            "keyboard": {
                "type": "buttons",
                "buttons": menus
            }
        })

    elif '아니오' == content_name:
        return JsonResponse({
```

```
            "message": {
                "text": "예약취소를 하지않습니다₩n메인으로 이동합니다"
            },
            "keyboard": {
                "type": "buttons",
                "buttons": menus
            }
        })

    elif content_name == '예약취소':
        return JsonResponse({
            "message": {
                "text": "예약취소하시겠습니까?"
            },
            "keyboard": {
                "type": "buttons",
                "buttons": ['예', '아니오']
            }
        })

    elif content_name == '도서관':
        try:
            req = urllib.request.urlopen('http://203.232.237.8/domian5/2/domian5.asp')
        except:
            pass

        soup = BeautifulSoup(req, 'lxml', from_encoding="utf-8")
```

```
my_titles = soup.select(
    'tr'
)
data = []
for title in my_titles:
    data.append(title.text)

# data[3]은 3층 1열람실 내용
# [3~7]까지의 (6)

if content_name == "도서관":
    lib_data = {'3-1': data[3].split()[6], '3-2': data[4].split()[6], '4-3A': data[5].split()[6],
                '4-3B': data[6].split()[6]}
else:
    # #print(data[name+2].split())
    lib_data  = {'%': data[name + 2].split()[6], '이용자': data[name + 2].split()[4],
                '남은 좌석': data[name + 2].split()[5]}

if lib_data == 555:
    return JsonResponse({
        'message': {
            'text': "도서관 좌석을 불러 올 수 없습니다.₩n다시 이용해주세요",
        },
        'keyboard': {
            'type': 'buttons',
            'buttons': button_info
        }
```

```
            })

        buttons = ['3층 1열람실: ' + str(lib_data['3-1']) + '%',
                   '3층 2열람실: ' + str(lib_data['3-2']) + '%',
                   '4층 3열람실A: ' + str(lib_data['4-3A']) + '%',
                   '4층 3열람실B: ' + str(lib_data['4-3B']) + '%',
                   '취소'
                   ]

        return JsonResponse({
            'message': {
                'text': '열람실을 선택하세요'
            },
            'keyboard': {
                'type': 'buttons',
                'buttons': buttons
            }
        })


    elif '열람실' in content_name:

        if '3층 1열람실: ' in content_name:

            name = "3층 1열람실 현황: "
            lib_num = 1
            room_no = 8
```

```
        elif '3층 2열람실: ' in content_name:

            name = "3층 2열람실 현황: "
            lib_num = 2
            room_no = 9

        elif '4층 3열람실A: ' in content_name:

            name = "4층 3열람실A 현황: "
            lib_num = 3
            room_no = 10

        elif '4층 3열람실B: ' in content_name:

            name = "4층 3열람실B 현황: "
            lib_num = 4
            room_no = 11

        lib_data = library_crawl.glo_library(lib_num)

        return JsonResponse({
            'message': {
                'text': name + str(lib_data['%']) + '%' + '₩n이용자 수: ' +
str(lib_data['이용자']) +
                        '명₩n남은 좌석 수: ' + str(lib_data['남은 좌석']),
                'message_button': {
                    'label': '좌석보기',
                    'url': 'http://203.232.237.8/domian5/roomview5.asp?room_no=' +
```

```
str(room_no)
                    }
                },
                'keyboard': {
                    'type': 'buttons',
                    'buttons': menus
                }
            })

    else:
        return JsonResponse({
            "message": {
                "text": "메인으로 이동합니다"
            },
            "keyboard": {
                "type": "buttons",
                "buttons": menus
            }
        })
```

7.7 Apps.py

```
from django.apps import AppConfig


class StudyRoomConfig(AppConfig):
    name = 'study_room'
```

7.8 Urls.py

```
from django.contrib import admin
from django.conf.urls import url
```

```
from django.urls import include


urlpatterns = [
    url('admin/', admin.site.urls),
    url('', include('study_room.app_urls'))
  ]
```

7.9 Settings.py

```
"""
Django settings for study_room_capstone project.

Generated by 'django-admin startproject' using Django 2.0.2.

For more information on this file, see
https://docs.djangoproject.com/en/2.0/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/2.0/ref/settings/
"""


import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))



# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/2.0/howto/deployment/checklist/
```

```
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '9gmdsukdoaz7@kp%i6xd4isi%l=**w7k^2upm2h%b%xgxsry*e'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['*']


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'study_room',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
```

```
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'study_room_capstone.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')]
        ,
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'study_room_capstone.wsgi.application'


# Database
# https://docs.djangoproject.com/en/2.0/ref/settings/#databases
```

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}


# Password validation
# https://docs.djangoproject.com/en/2.0/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]
```

```
# Internationalization
# https://docs.djangoproject.com/en/2.0/topics/i18n/

LANGUAGE_CODE = 'ko-kr'

TIME_ZONE = 'Asia/Seoul'

USE_I18N = True

USE_L10N = True

USE_TZ = True



# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.0/howto/static-files/

    STATIC_URL = '/static/'
```

7.10 wsgi.py

```
import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault("DJANGO_SETTINGS_MODULE", "study_room_capstone.settings")

application = get_wsgi_application()
```

7.11 라즈베리파이 코드

```
import serial
import time
import requests

import datetime
import json
now_time = datetime.datetime.now()
str_time = now_time.strftime("%H") +"00"

data =   {"room":2, "start_time": int(str_time)}
data_json = json.dumps(data)
r = requests.get("http://35.200.7.249:8000/all_pw", data=data_json)

with open('pwdata.txt', 'w') as outfile:
 json.dump(r, outfile)



import serial
import time
import requests

import datetime
import operator
now_time = datetime.datetime.now()
str_time = now_time.strftime("%H") +"00"
import json

with open('pwdata.txt', 'r') as outfile:
```

```
json.dump(outfile)


ser = serial.Serial('/dev/ttyACM0', 9600)


password=outfile.json()
pw_list = []
y=time.strftime("%H")


for i in range(0,24):
        if(y==str(i)):
        print("y:" + y)
        if (i == 0):
                time = 0
            else:
                time = str(i) + "00"


        x = u'{!r}'.format(int(time)) #pasing for dict
        print("present time: "+x)


        for i in range(0,5):
                ser.write('{!r}'.format(password[x])[2:6])
```

7.12 아두이노 코드

```
#include <Keypad.h>
#include <LiquidCrystal.h>
#include <Servo.h>
Servo myservo;
```

```
LiquidCrystal lcd(A0, A1, A2, A3, A4, A5);

#define Password_Lenght 5 // Give enough room for  four chars + NULL char

char ch[4];
int pos = 0;      // variable to store the servo position
int i_ch =0;
char Data[Password_Lenght]; // 4 is the number of chars it can hold + the null char = 5
char Master[Password_Lenght];
byte data_count = 0, master_count = 0;
bool Pass_is_good;

char customKey;

const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3', 'A'},
  {'4','5','6', 'B'},
  {'7','8','9', 'C'},
  {'*','0','#', 'D'}
};
bool door = true;

byte rowPins[ROWS] = {8, 7, 6, 5}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {4, 3, 2, 1}; //connect to the column pinouts of the keypad
```

```
Keypad customKeypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS); //initialize an instance of class NewKeypad


void setup()
{
   myservo.attach(9);
   ServoClose();
   lcd.begin(16, 2);
   lcd.print(" Arduino Door");
   lcd.setCursor(0, 1);
   lcd.print("--Look project--");
   delay(3000);
   lcd.clear();
   Serial.begin(9600);
}


void loop()
{
        initial();
        Serial.print("pw:");
        Serial.print(Master[0]);
        Serial.print(Master[1]);
        Serial.print(Master[2]);
        Serial.print(Master[3]);
        Serial.print(" ");
        delay(2000);


   if (door == 0)
```

```
  {
    customKey = customKeypad.getKey();
//    Serial.println(customKey);
    delay(300);
    if (customKey == '#')
    {
      analogWrite(12,120);
      lcd.clear();
      ServoClose();
      lcd.print("   Door is close");
      delay(3000);
      door = 1;
    }
  }

  else Open();
}

void clearData()
{
  while (data_count != 0)
  { // This can be used for any array size,
    Data[data_count--] = 0; //clear array for new data
  }
  return;
}

void ServoOpen()
```

```
{
  for (pos = 180; pos >= 30; pos -= 5) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);                  // tell servo to go to position in variable 'pos'
    delay(15);                           // waits 15ms for the servo to reach the position
  }
}

void ServoClose()
{
  for (pos = 0; pos <= 180; pos += 5) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);                  // tell servo to go to position in variable 'pos'
    delay(15);                           // waits 15ms for the servo to reach the position
  }
}
void initial(){
  int i;
  delay(2000);
  ch[0] =Serial.read();
  ch[1] =Serial.read();
  ch[2] =Serial.read();
  ch[3] =Serial.read();
  for(i=0;i<4;i++){
  Master[i]= ch[i];

}
}
```

```
void Open()
{
  lcd.setCursor(0, 0);
  lcd.print(" Enter Password");

  customKey = customKeypad.getKey();
//   Serial.println(customKey);
  delay(300);
  if (customKey) // makes sure a key is actually pressed, equal to (customKey != NO_KEY)
  {
    Data[data_count] = customKey; // store char into data array
    lcd.setCursor(data_count, 1); // move cursor to show each new char
    lcd.print(Data[data_count]); // print char at said cursor
    data_count++; // increment data array by 1 to store new char, also keep track of the number of chars entered
  }

  if (data_count == Password_Lenght - 1) // if the array index is equal to the number of expected chars, compare data to master
  {
    if (!strcmp(Data, Master)) // equal to (strcmp(Data, Master) == 0)
    {
      lcd.clear();
      ServoOpen();
      lcd.print("   Door is Open");
      door = 0;
    }
    else
```

```
    {
      lcd.clear();
      lcd.print("    Wrong Password");
      delay(1000);
      door = 1;
    }
    clearData();
  }
   }
```

8. 참고 문헌

Title	URL
Kakao Talk API	https://github.com/plusfriend/auto_reply#specification
SQLite3	https://soooprmx.com/archives/4056
Auduino_lock	https://www.youtube.com/watch?v=r3z5GIRbzCY
Raspberry_lock	https://github.com/c-base/raspberrylock
Google cloud platform	https://cloud.google.com
결제 시스템	https://github.com/iamport/iamport-manual/tree/master/SMS%EB%B3%B8%EC%9D%B8%EC%9D%B8%EC%A6%9D