

기획력과 리더십을 겸비한 개발자, 권순형입니다.

최종학력

서울시립대학교 전자전기컴퓨터공학부

자격증

SQLD

git

<https://github.com/SunHyongKwon>

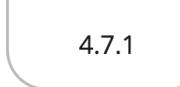
notion

<https://neuroo.notion.site/>

목 차

1 사용한 프로그램	3	5 App에 사용한 Backend	17
2 App 프로젝트	4	5.1 Spring Boot	18
2.1 솜솜 하우스 - 부동산 정보 및 예측 어플	5	5.2 Flask	21
2.2 성연(星演) - 소상공인을 위한 어플	7	6 App에 사용한 Open API	22
2.3 가계부자 - 간편하게 사용 가능한 가계부 어플	9	7 App에 사용한 AI 모델	26
3 App UI	11	7.1 머신러닝 지도학습 회귀 모델	27
3.1 Flutter	12	7.2 머신러닝 지도학습 분류 모델	29
4 App에 사용한 DB	14	8 기타 기술 스택	31
4.1 RDBMS	15	8.1 데이터 모델링	32
		8.2 Scraping	33

사용한 프로그램

App Tool	Backend Tool	Database Tool	데이터 분석 & AI	Cooperation Tool
Application Framework  3.3.10	언어  2.18.7	framework  3.0.2 / 2.7.5	server  9.0.68	RDBMS  8.0.31
IDE  14.2	언어  9.0.68	Flask  17 / 11	NoSQL  Firebase	언어  4.2.2
			TinyDB  3.40.1	소프트웨어  2022.12.0+353
		 3.10.9		Web Service  Github
				DVCS  Git
				Project Design  Notion
				Interface Design  Figma
				Miro 

App 프로젝트

목 차 | 1 솜솜 하우스 - 부동산 정보 및 예측 어플

1.1 [프로젝트 개요](#) 5

1.2 [System Flow Diagram](#) 6

2 성연(星演) - 소상공인을 위한 어플

2.1 [프로젝트 개요](#) 7

2.2 [System Flow Diagram](#) 8

3 가계부자 - 간편하게 사용가능한 가계부 어플

3.1 [프로젝트 개요](#) 9

3.2 [System Flow Diagram](#) 10

솜솜 하우스

프로젝트 개요



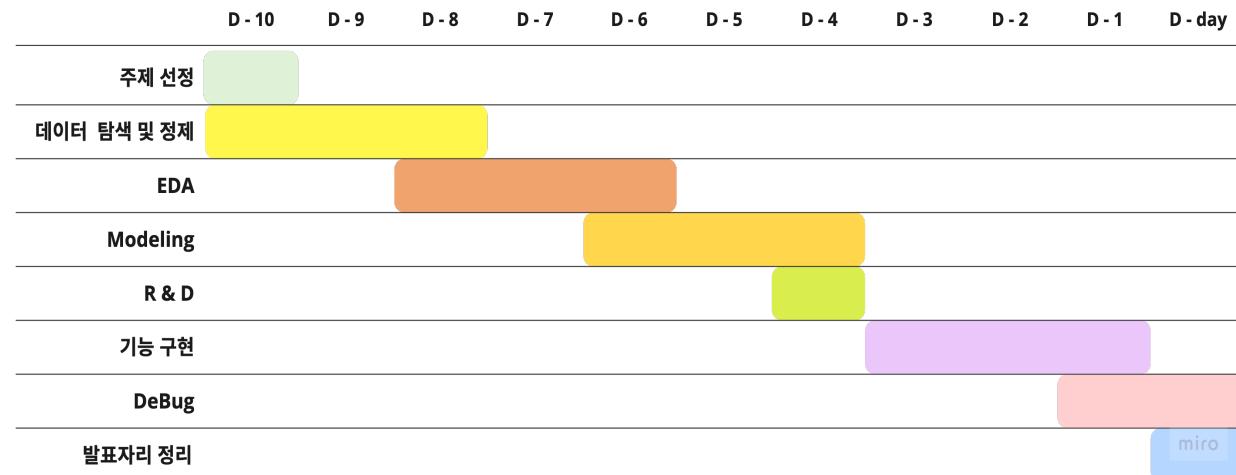
주제

서울 전월세 정보 조회 및
전세 가격 예측 서비스 제공

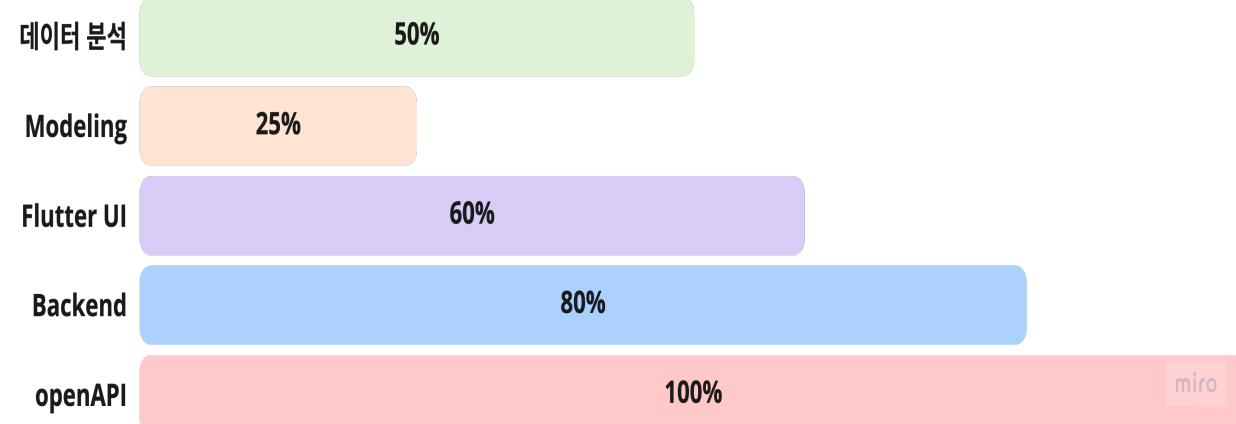
주제 선정 이유

치솟는 부동산 가격으로 인한 주거안정에 대한 불안감과
부동산 문제가 인생 전반의 계획에 큰 부분을 차지 하기에
부동산 전세 가격 예측과 전월세 정보를 제공함으로써
주거안정에 대한 불안감 해소와
인생 계획에 도움이 되고자 주제를 선정했습니다.

개발 일정



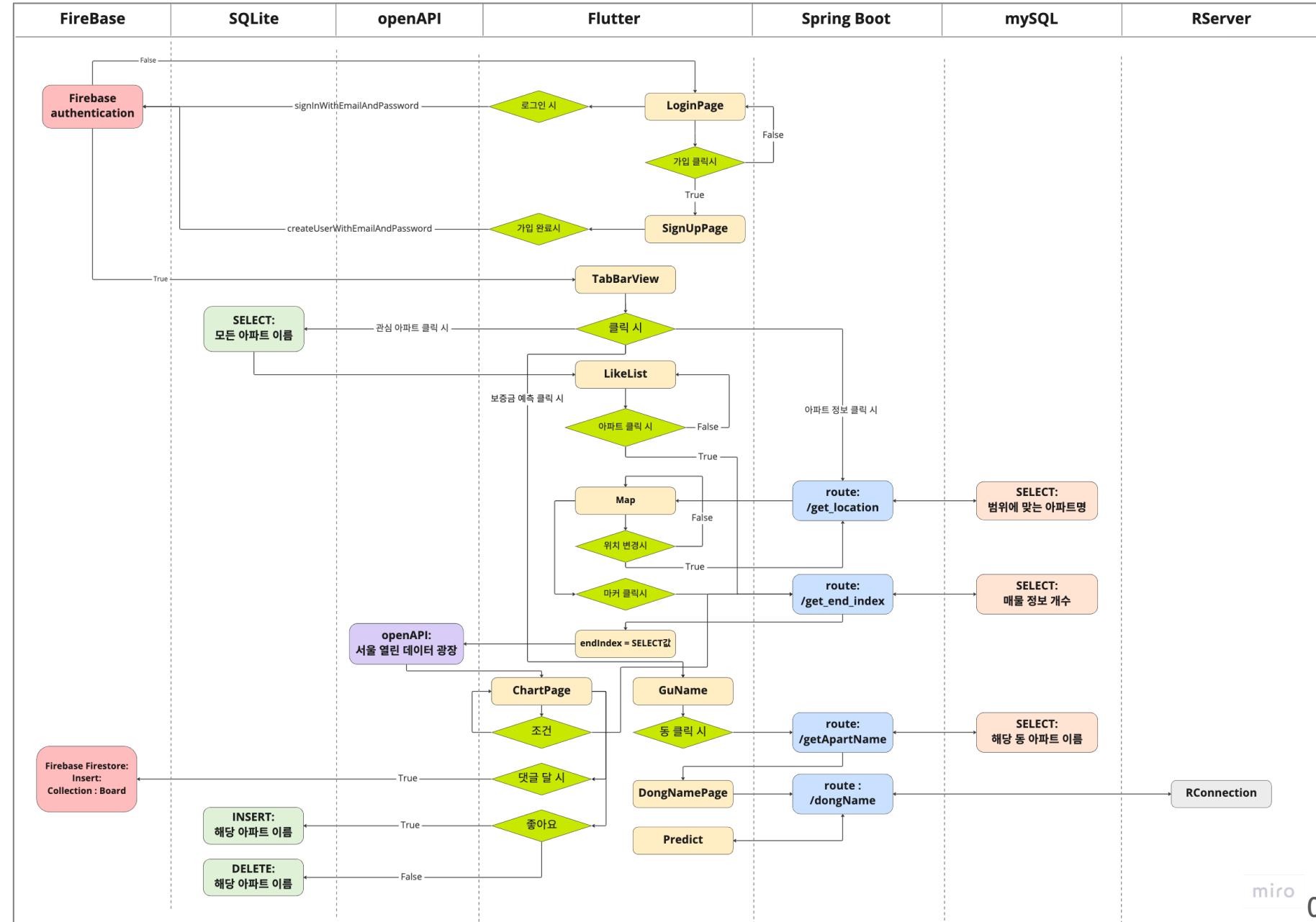
담당 역할



솜솜 하우스

System Flow Diagram

- Flutter 이용한 앱 UI 디자인
- SQLite를 이용하여
Device 내부 DB 구현
- Spring Boot를 이용하여
mySQL DB 연동과 RServer
연결하는 Backend 구성
- Firebase 이용하여
Login ,RealTime DB 구축
- openAPI 이용하여
필요한 데이터 가져오기



성연 (星演)

프로젝트 개요



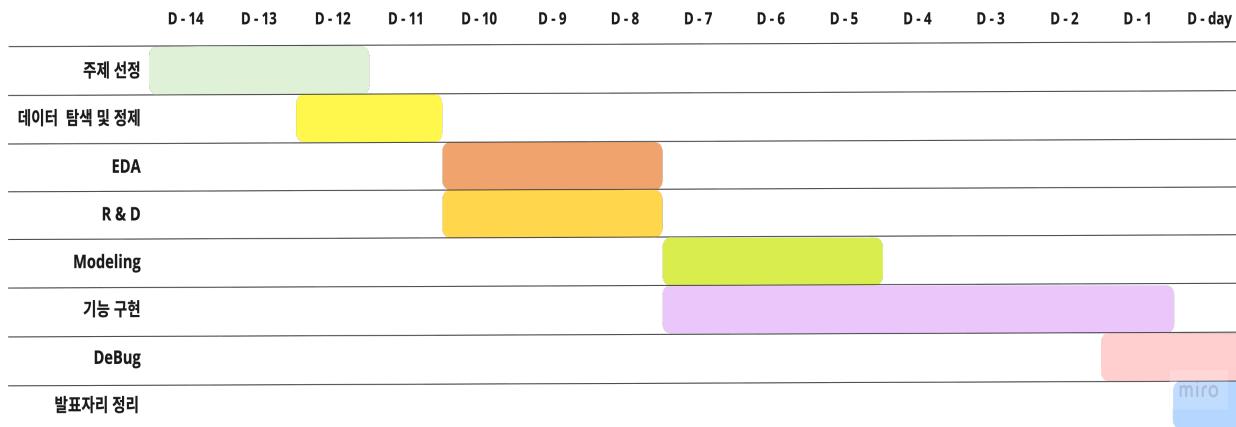
주제

예비 자영업자를 위한
상권 분석과 매출 예측 서비스 제공

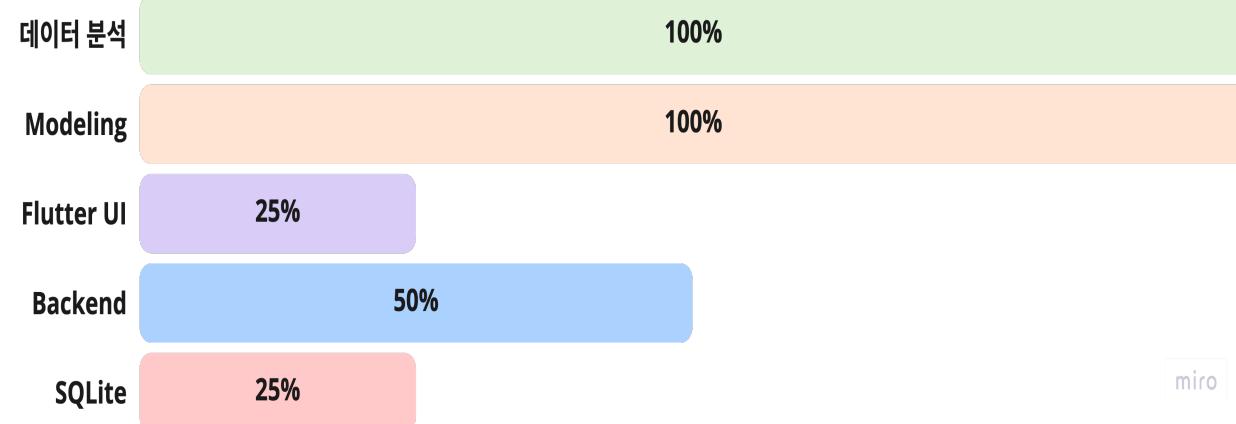
주제 선정 이유

전체 취업자 중 자영업자의 비율이 25% 정도입니다.
위와 같은 이유로 많은 사용자를 타겟으로 할 수 있고,
자영업의 경우, 상권에 따른 매출의 차이가 크다고 알고 있습니다.
이를 분석하여 정보를 제공함으로써 예비 자영업자한테
의미 있는 지표로 사용할 수 있을 것 같습니다.

개발 일정



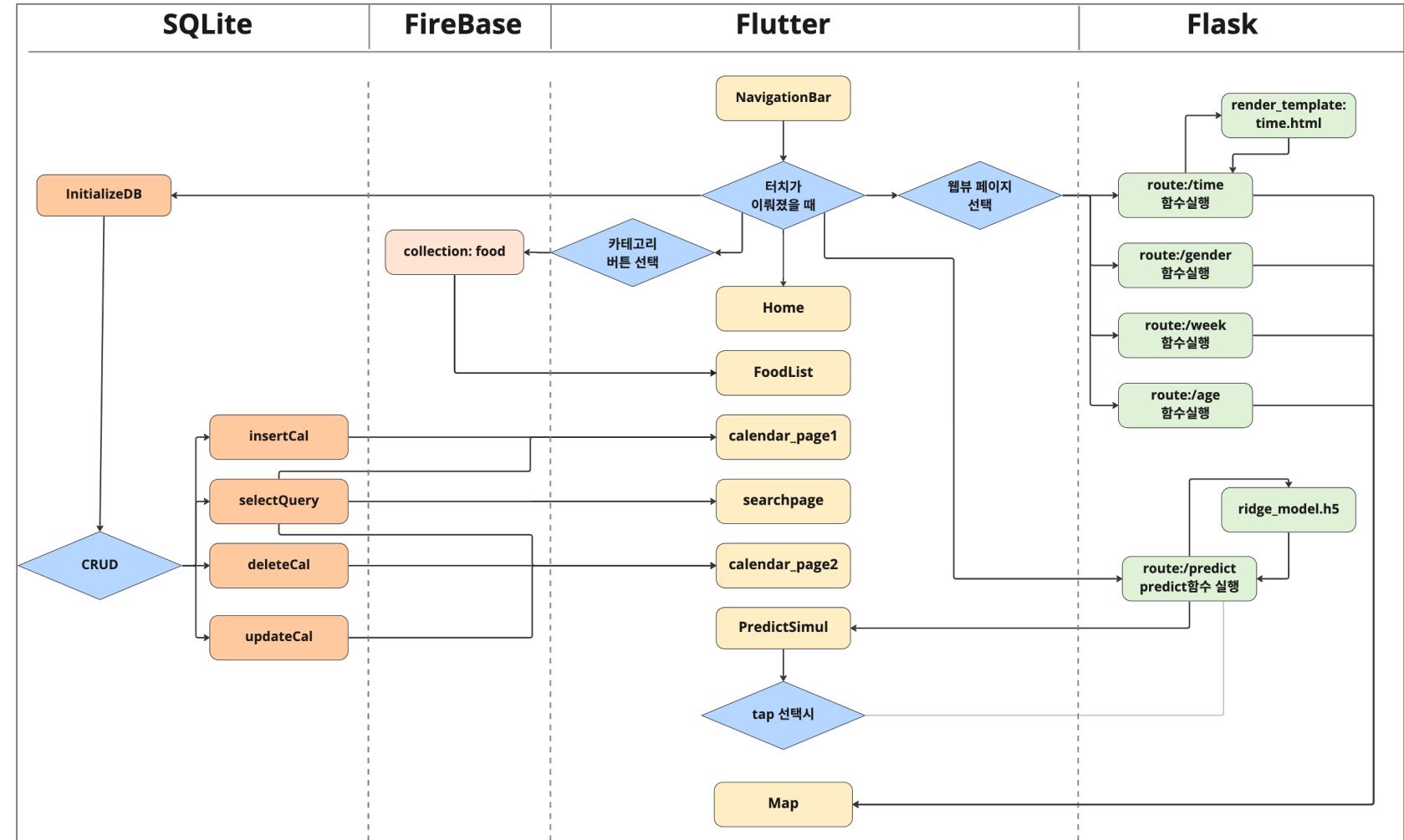
담당 역할



성연 (星演)

System Flow Diagram

- Flutter를 이용하여 앱 UI 디자인
- SQLite를 이용하여
Device 내부 DB 구현
- Flask를 이용해 AI 모델과 연동하여
매출액 예측 금액을 Flutter UI에 표시
- Firebase를 이용하여 crawling한
데이터를 Flutter 화면에 표현



가계부자 - app store 배포 완료

프로젝트 개요



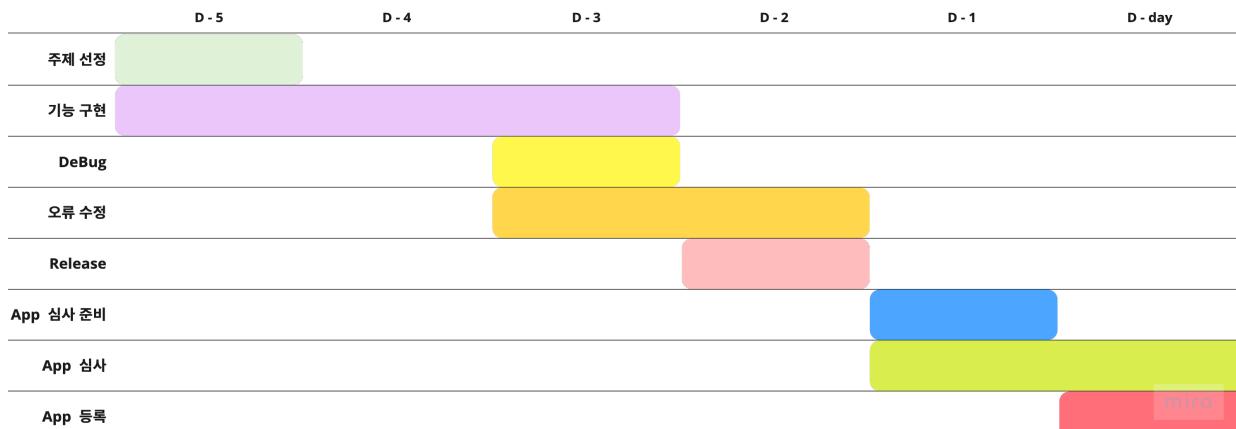
주제

회원가입없이 간편하게
사용할 수 있는 가계부

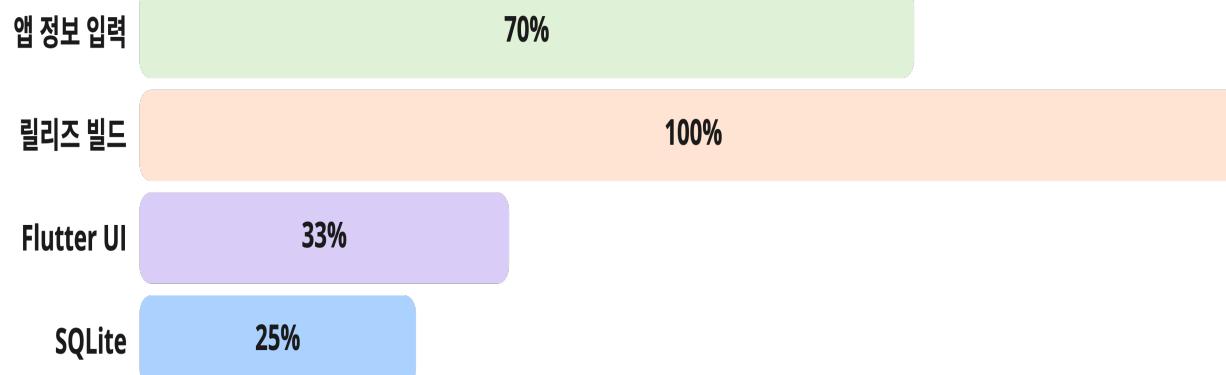
주제 선정 이유

성연 프로젝트에서 사용한 가계부가 SQLite만을 사용하여서 이를 개인이 사용할 수 있도록 카테고리 항목 추가와 일별, 카테고리별, 제목 별 검색 기능을 추가하여 app store에 배포하는 프로젝트를 추가적으로 진행하였습니다.

개발 일정



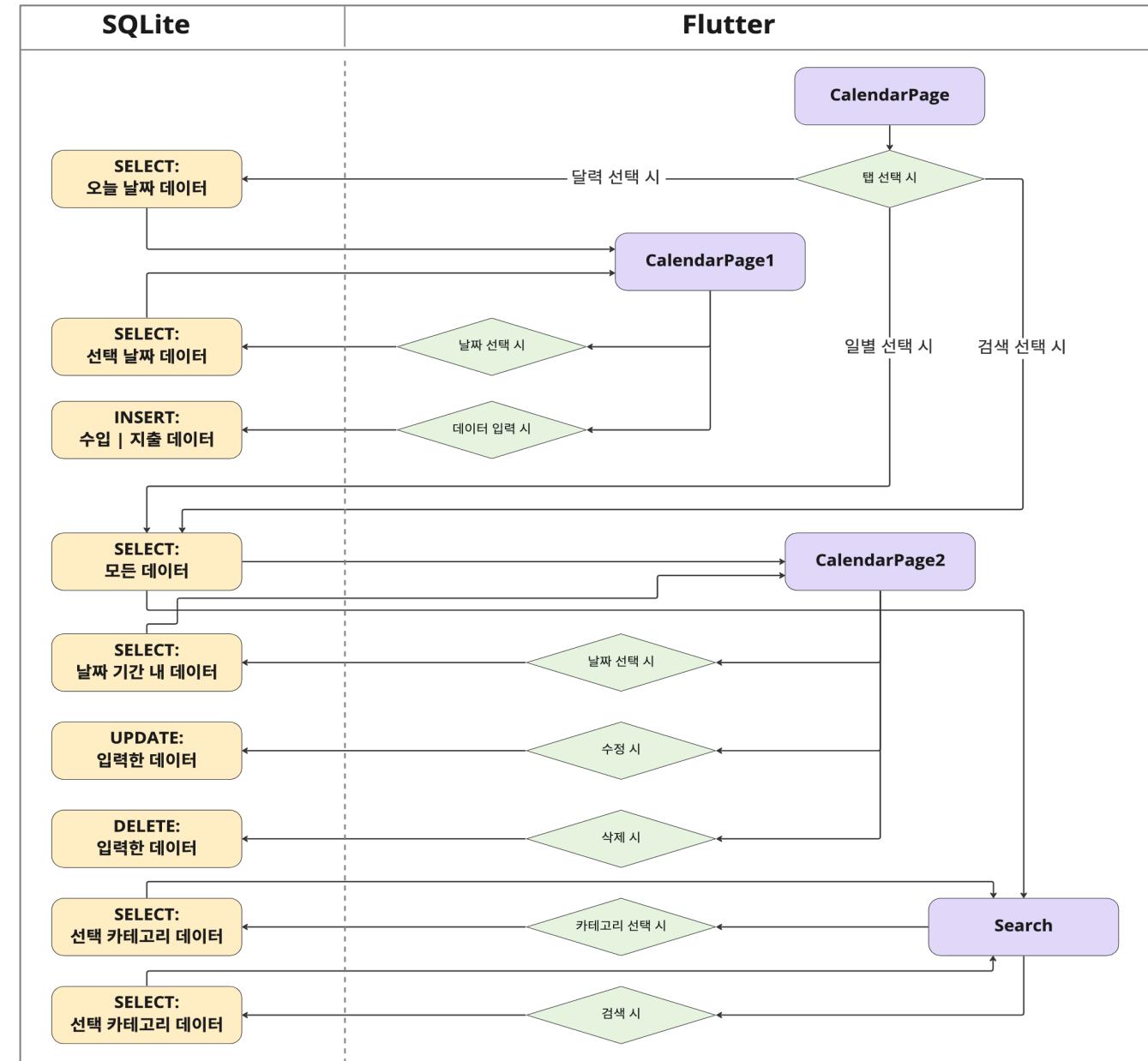
담당 역할



가계부자 - app store 배포 완료

System Flow Diagram

- Flutter를 이용하여 앱 UI 디자인
- SQLite를 이용하여 Device 내부 DB 구현



App UI

목 차 | 1 Flutter

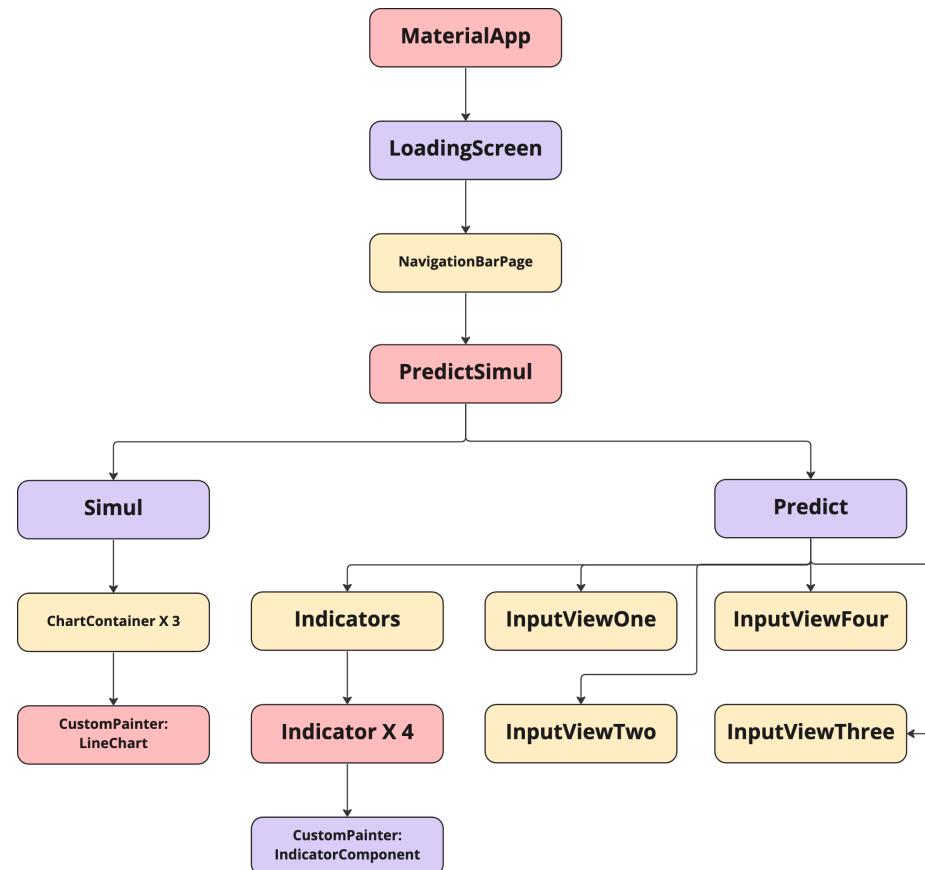
1.1 [위젯 분리](#) 12

1.2 [자체 위젯 만들기](#) 13

기능 구현 : 다른 클래스 간의 데이터 주고 받기

StreamController를 이용하여 다른 클래스 간에 데이터를 주고 받을 수 있는 기능을 구현

Widget Tree

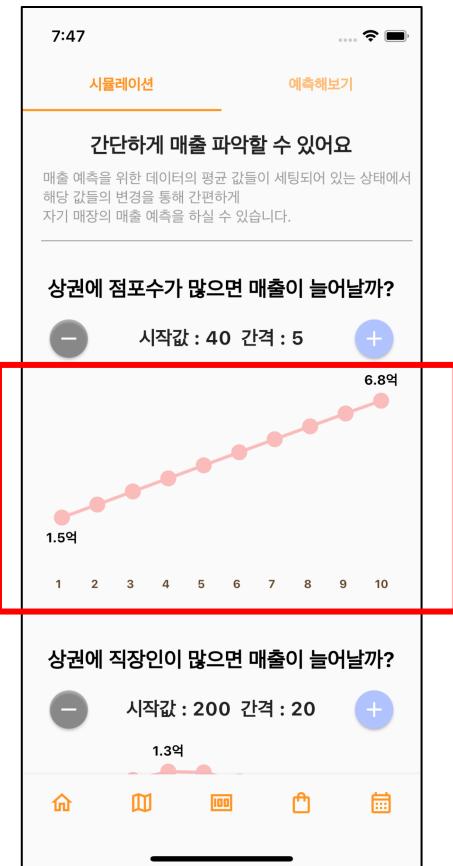


- 위젯 간의 분리를 통해 유지 보수 시 어느 부분에서 오류가 났는지 쉽게 파악할 수 있다.
- 한 화면을 여러 위젯으로 분리를 함으로써 한 화면 사이에서 데이터 교환이 필요할 때 클래스 간의 데이터가 필요한 경우가 발생
- 그 때에는 StreamController를 이용하여 controller의 `listen` 함수와 `controller.stream`의 `add` 함수를 통해 변화가 생기는 위젯에는 `listen` 함수를 변화를 주는 위젯에는 `add` 함수를 줘서 다른 클래스여도 데이터를 주고 받을 수 있게 하였다.

자체 위젯 만들기

CustomPainter를 이용한 차트 그리기 및 진행 상태 바 만들기

차트 그리기



- flutter CustomPainter를 이용한 차트 그리기
 - 크기가 있는 값을 리스트로 만든다.
 - 크기는 MinMaxScaler 공식을 이용하여 0 – 1 사이로 변환시킨다.
 - 리스트의 index와 값을 이용하여 Offset 변수를 만든다.
 - 이를 Path class의 moveTo, lineTo 를 이용하여 선을 만든다.
 - canvas.drawPath를 이용하여 차트로 그린다.

진행 상태 바 만들기



- CustomPainter를 이용하여 상태 바를 만들고 stack으로 4개의 화면을 구성하여 Visibility 의 visible 값을 action이 일어날 때마다 변하도록 list를 만든다.
- 이 값에 맞춰서 상태바의 color 값도 변하도록 리스트를 구성했다.

DB

목 차 | 1 RDBMS

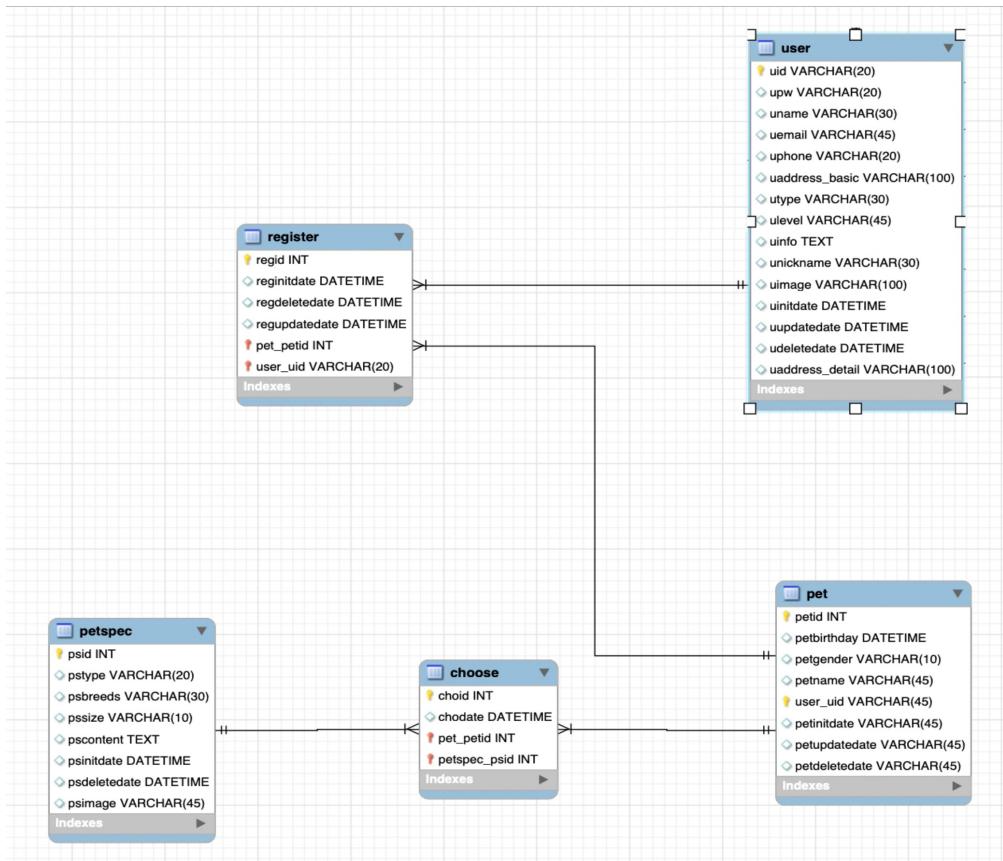
1.1 [MySQL](#) 15

1.2 [SQLite](#) 16

MySQL - 회원가입

회원 정보를 저장할 테이블 정의 (DDL) , 회원가입 / 정보 수정 / 아이디 비밀번호 찾기 시에 필요한 DML

EER / DDL



DML

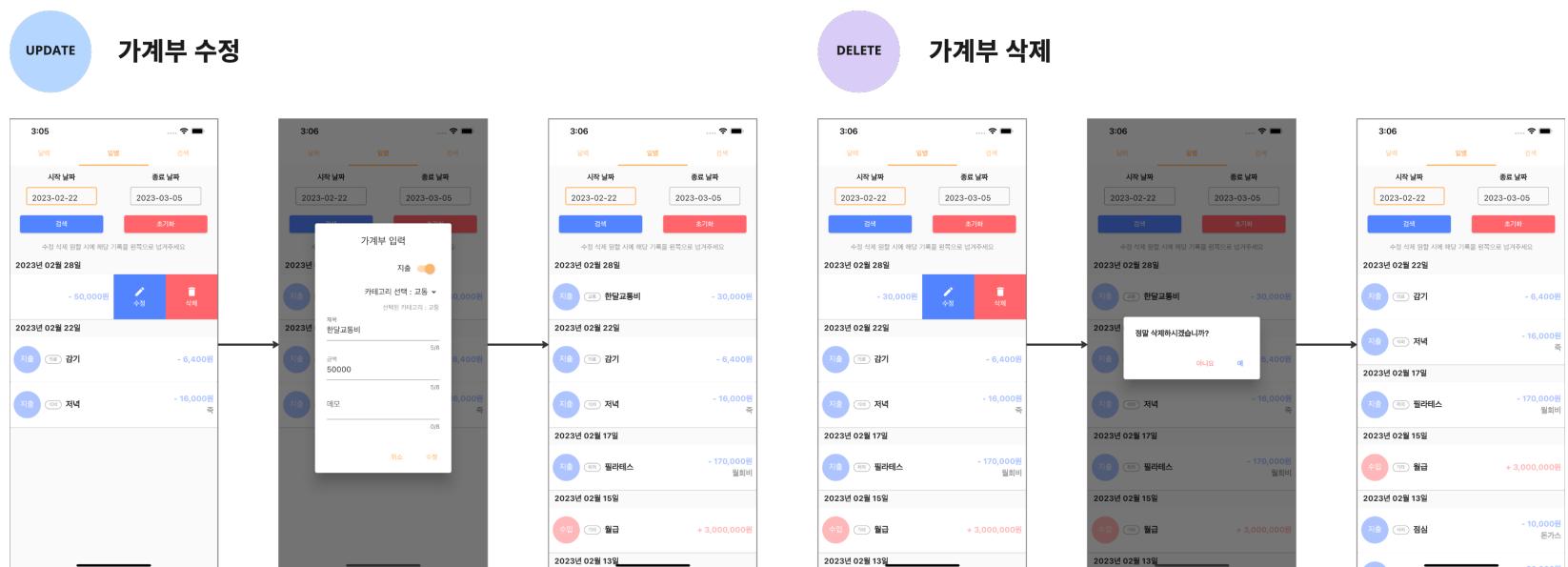
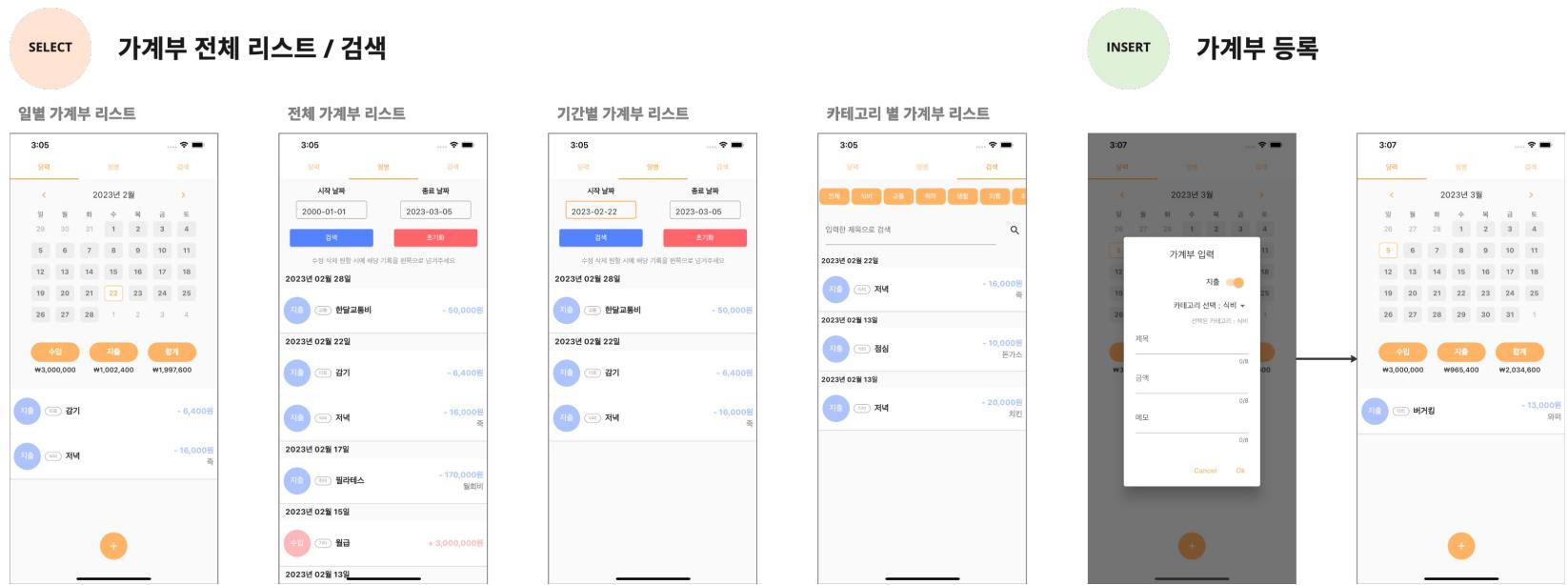
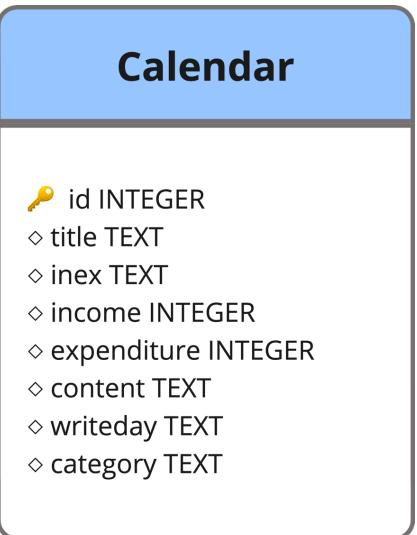


SQLite - 가계부

가계부를 저장할 테이블 정의 (DDL),
가계부 추가 / 수정 / 삭제 / 검색
기능이 가능하게 하는 DML

EER / DDL

DML



Backend

목 차 | 1 Spring Boot

1.1 [Spring Boot + DB](#) 18

1.2 [Spring Boot + AI](#) 19

1.3 [기타 기능 구현](#) 20

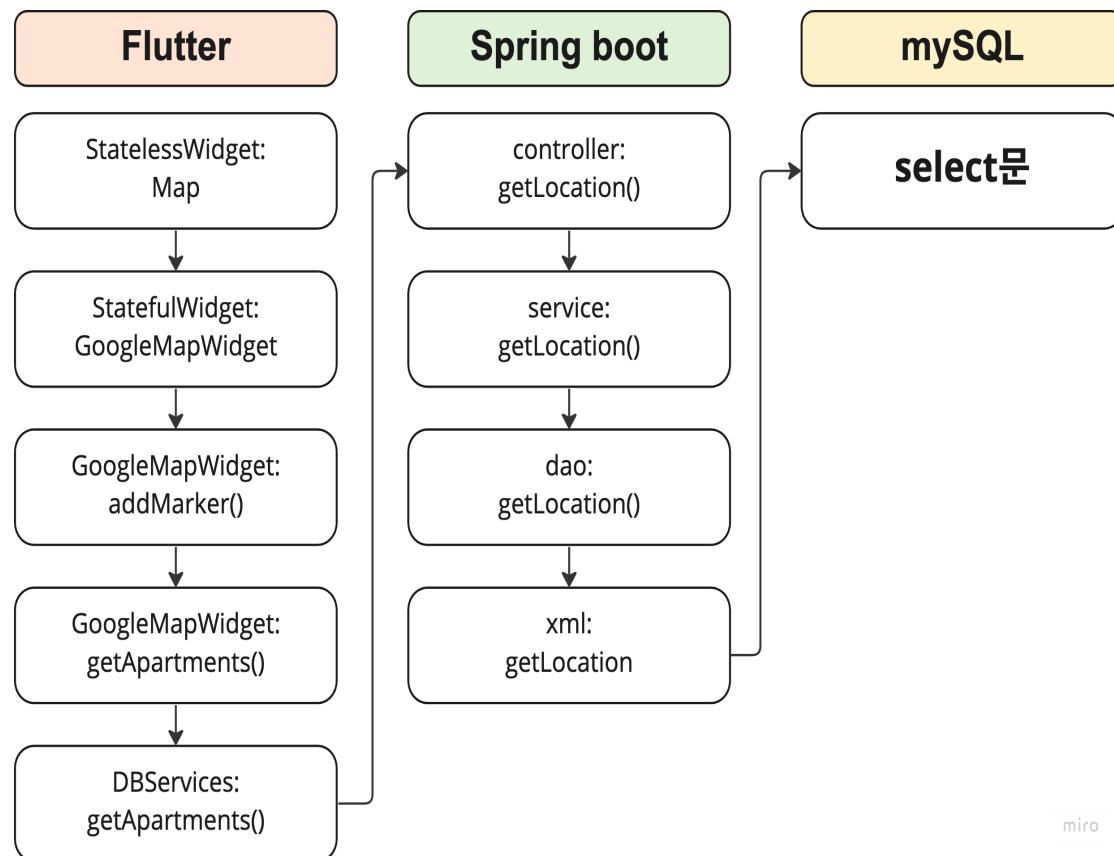
2 Flask

2.1 [Flask + AI](#) 21

Spring Boot +DB by.MySQL

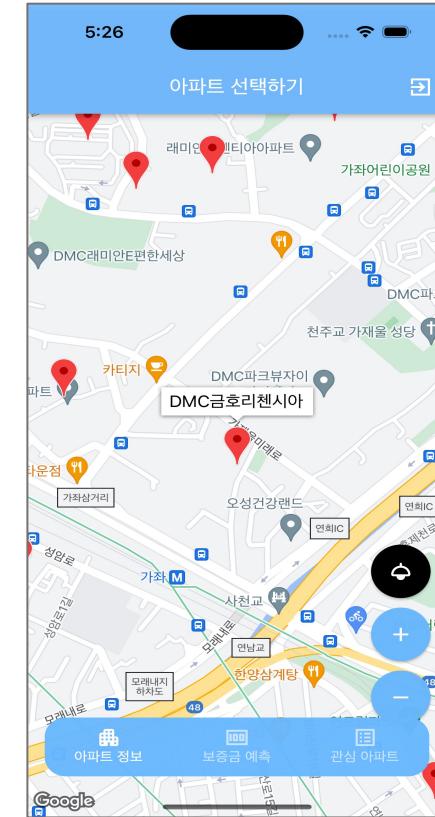
Spring Boot로 Rest API 구현 하여 DB의 아파트별 경도 위도 데이터를 App UI에 지도의 마커로 표시

Flow Diagram



구현 화면

iPhone



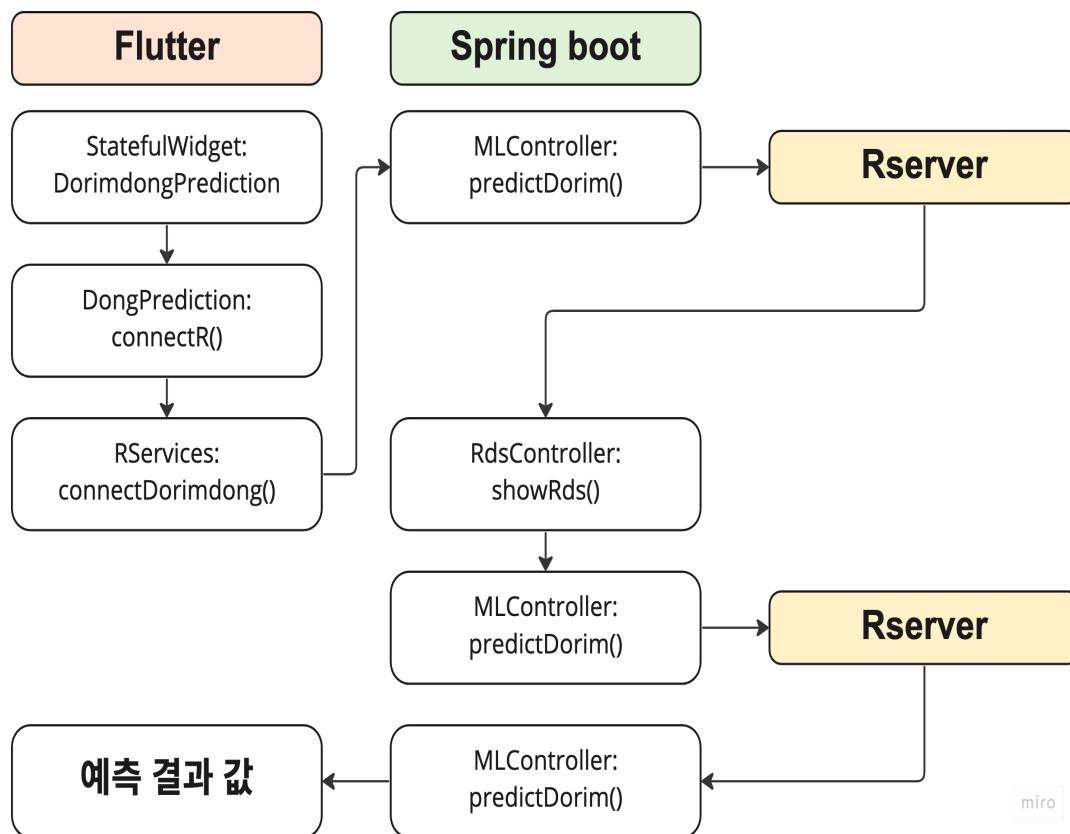
Android



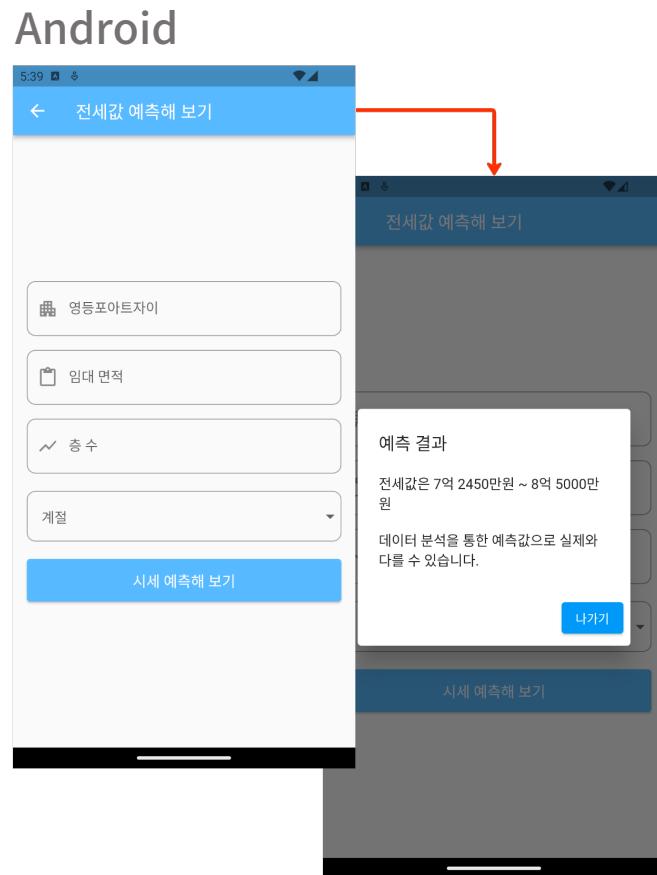
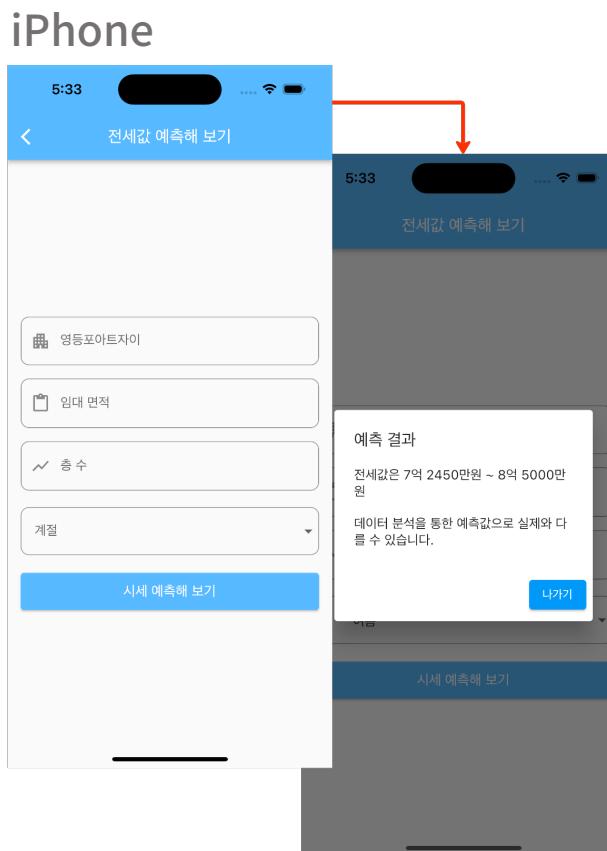
Spring Boot + AI by.R

Spring Boot로 Rserver와 연결하여 모델링한 AI에 입력값을 넣어 예측 결과 값을 UI에 보여주는 기능 구현

Flow Diagram



구현 화면



Spring Boot 기타 기능 구현

Spring Boot로 사용자에게 받은 파일을 웹 서버에 저장, 이메일로 인증 번호 발송

다중 파일 업로드

MultiPartHttpServletRequest를 이용한
Web Server에 다중 파일 업로드 기능

게시물 작성

제목

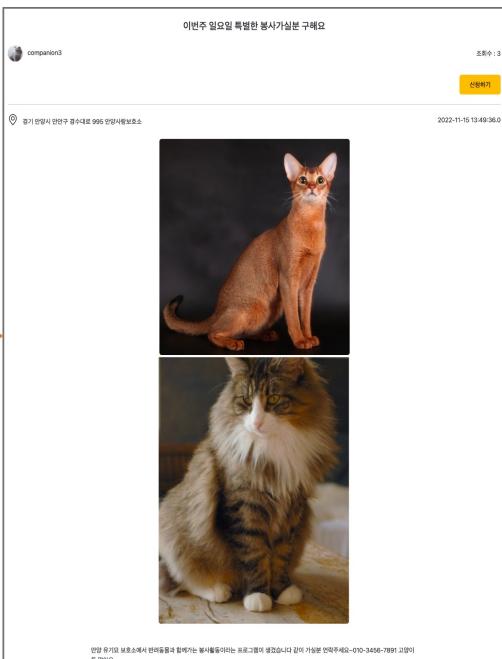
카테고리
 찾아주세요

내용

주소
 주소를 입력해 주세요

상세 주소를 입력해 주세요

사진 업로드
 Choose Files No file chosen



이메일 인증

Google SMTP 이용하여 이메일로 인증번호를 발송 하는 기능

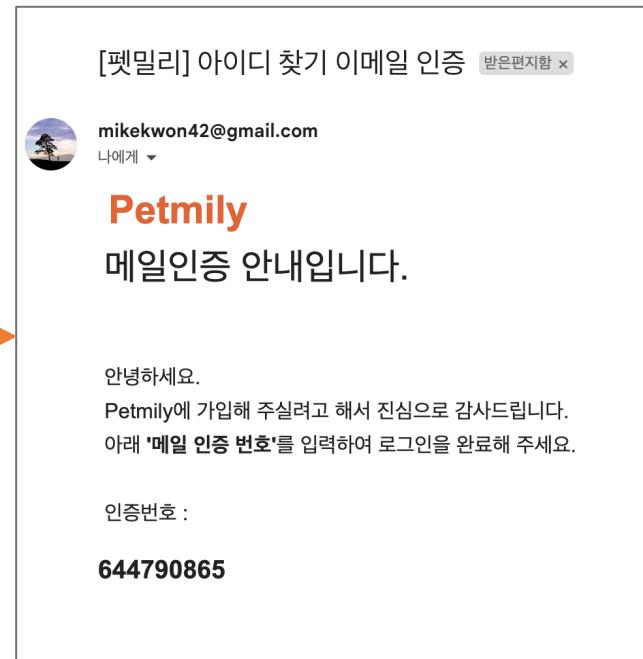
아이디 찾기

회원정보에 입력한 정보를 작성해 주세요

이름
 이름을 입력해 주세요

이메일
 이메일을 입력해 주세요

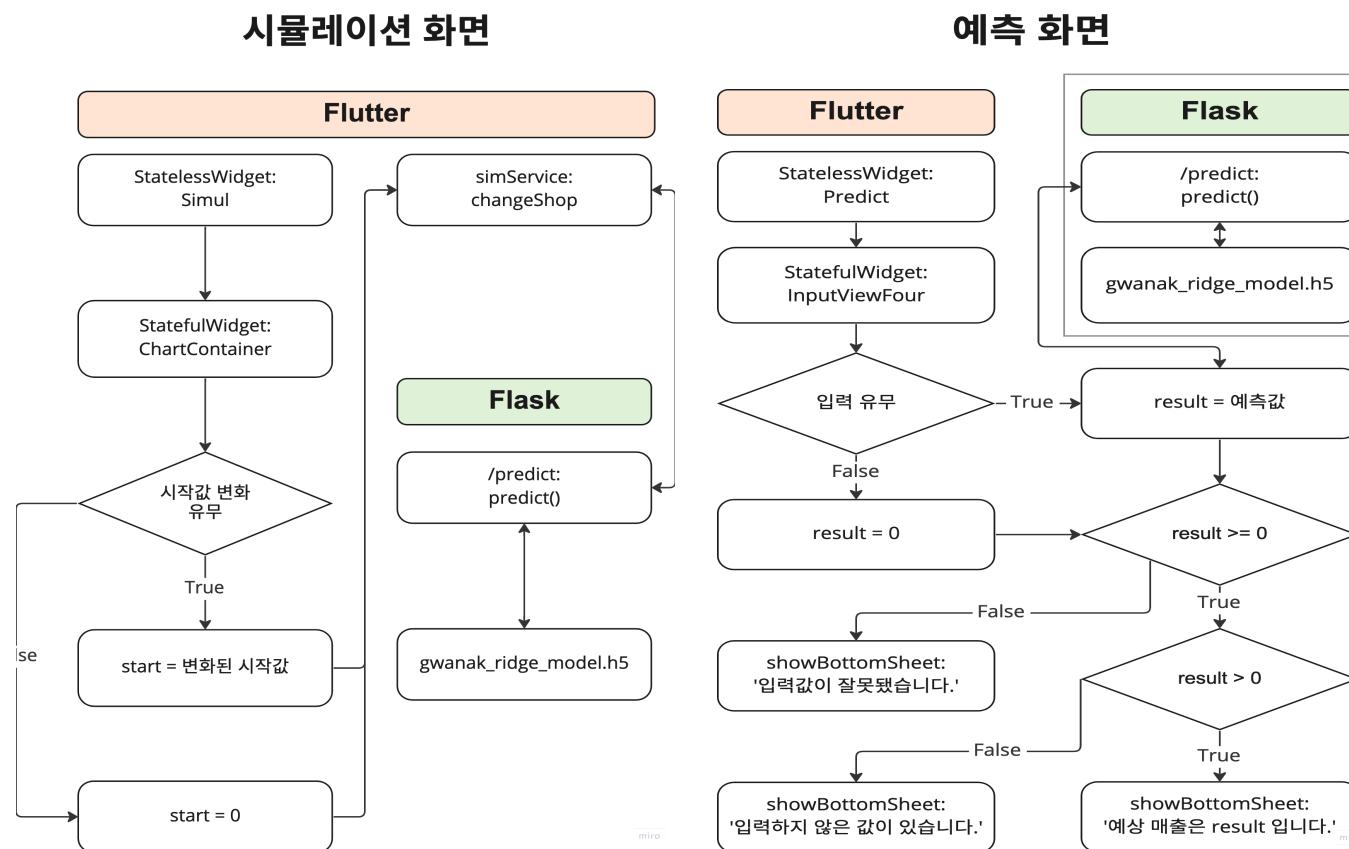
이메일 인증번호
 인증번호를 입력해 주세요



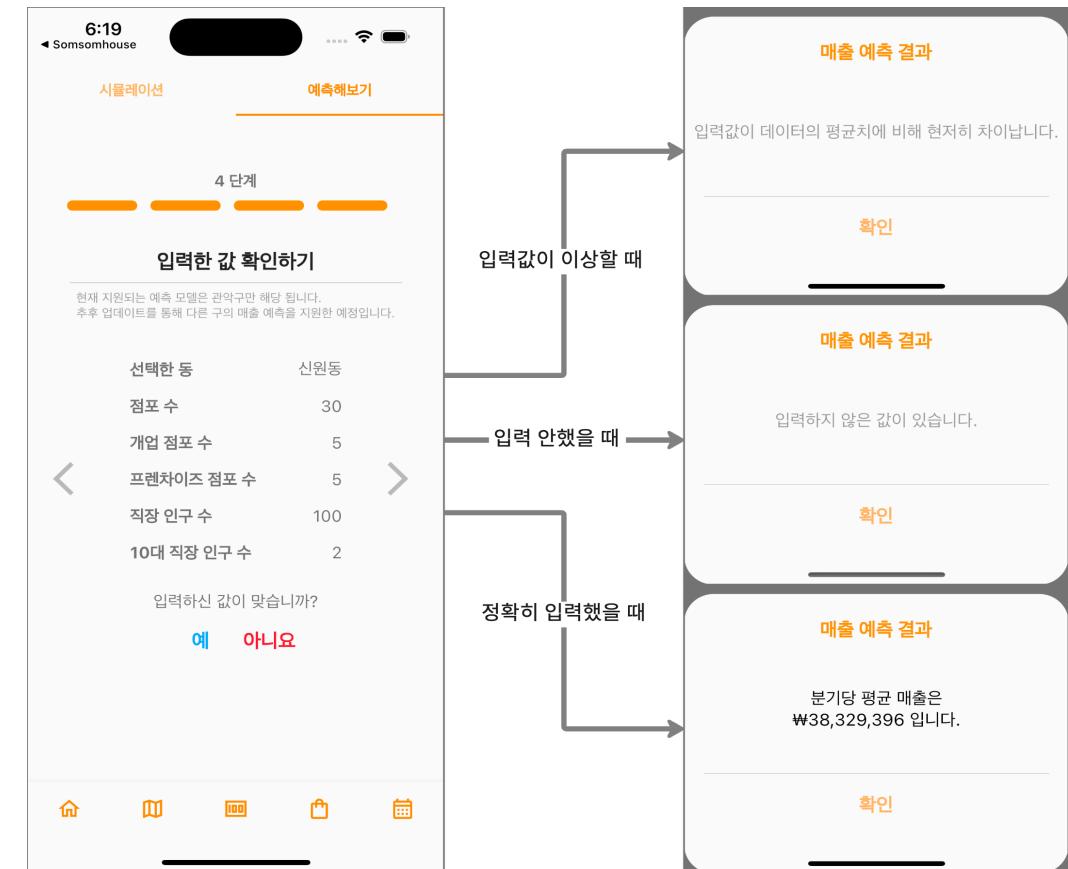
Flask + AI by.python(scikit-learn)

parameter 값을 받아 Web Server의 h5 파일을 이용해서 예측결과를 flutter 화면으로 불러오는 기능

Flow Diagram



구현 화면 - 예측 화면



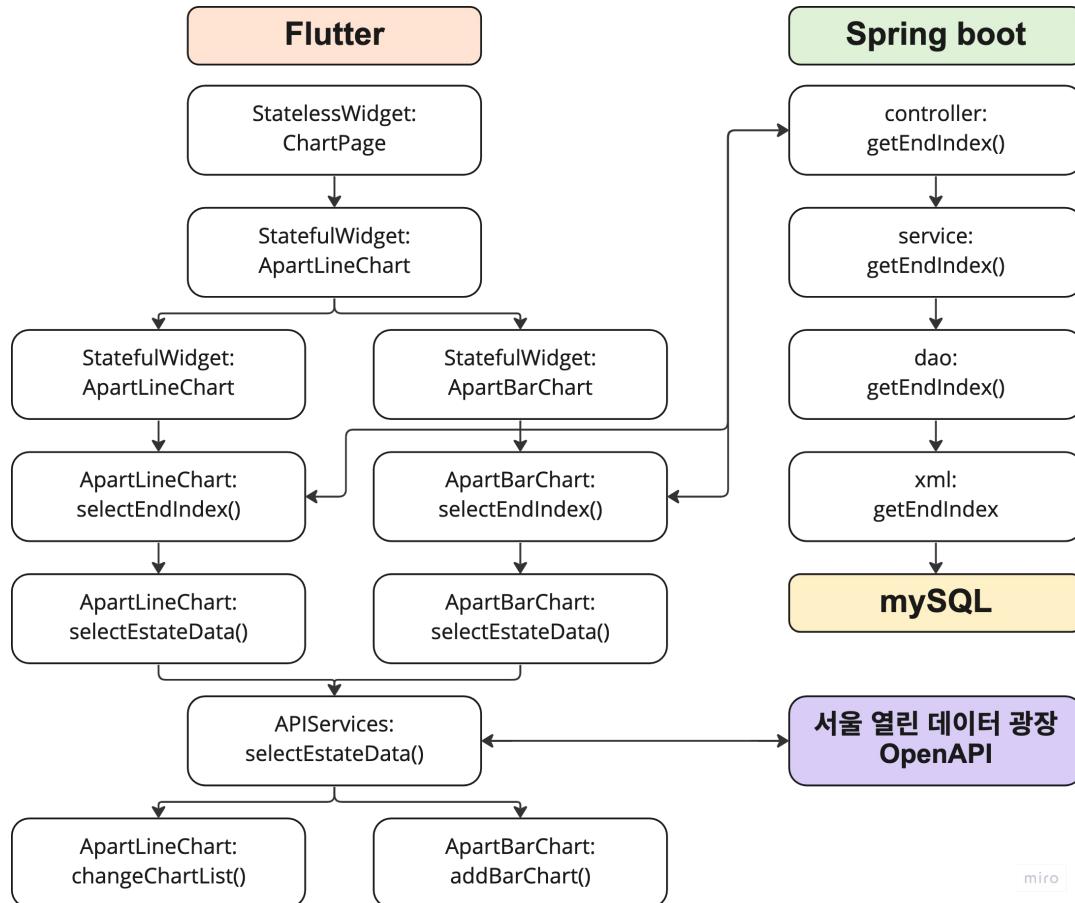
Open API

목 차 	1	<u>서울 열린 데이터 광장</u>	23
	2	<u>Google Map API</u>	24
	3	<u>Kakao Login API</u>	25

서울 열린 데이터 광장 - 서울시 부동산 전월세가 정보

서울 열린 데이터 광장 openAPI로 JSON 구조의 데이터를 받아서 fl-chart 패키지 이용하여 차트로 표현

Flow Diagram



구현 화면

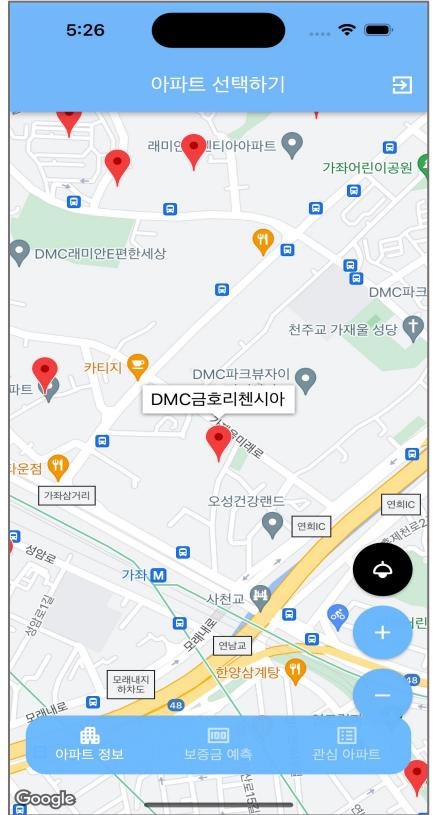


Google Map API

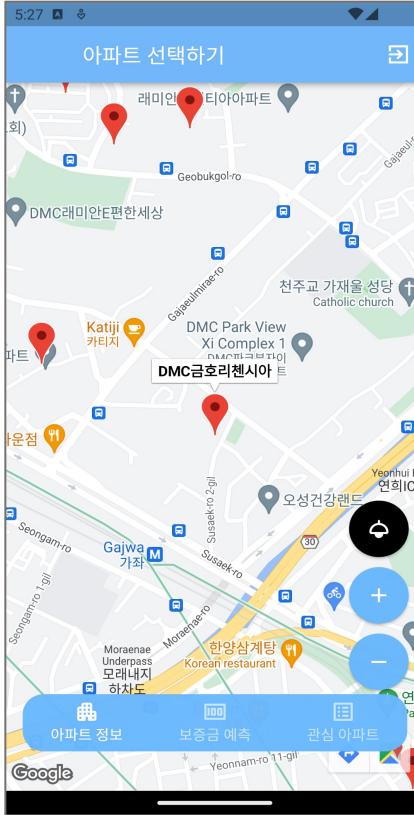
App에 위치 데이터를 마커 표시 위한 Maps SDK, 주소로 경도 위도를 찾기 위한 Geocoding API

Maps SDK for IOS , Android - App

iPhone

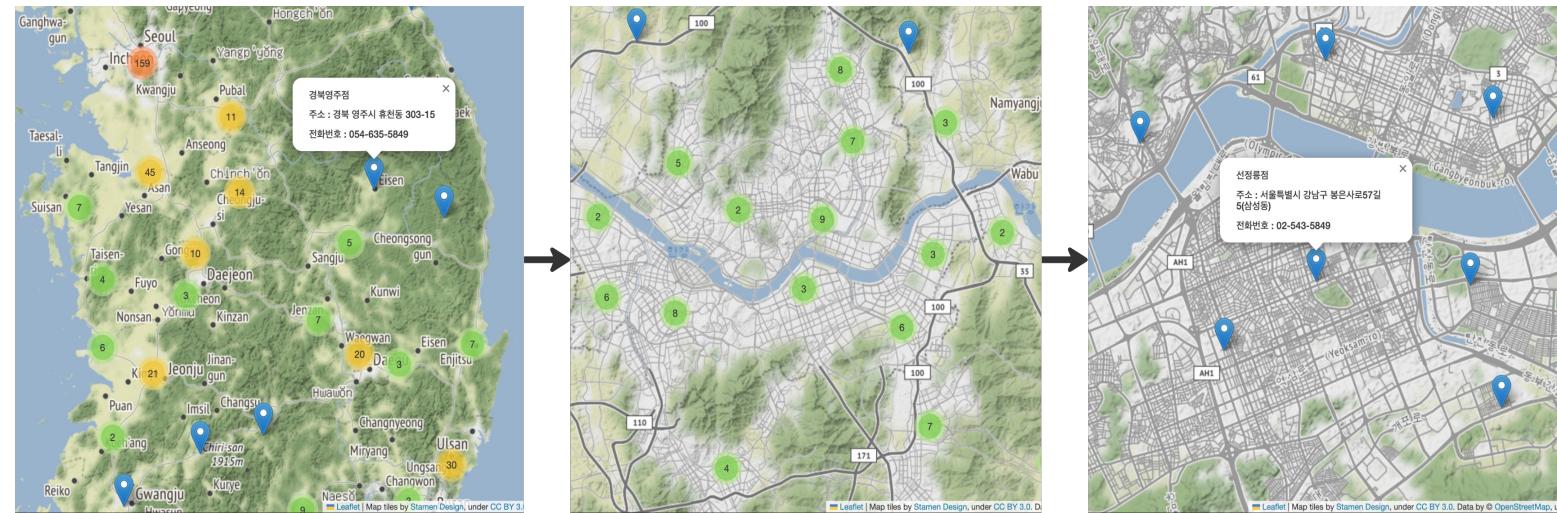


Android



Geocoding API - python

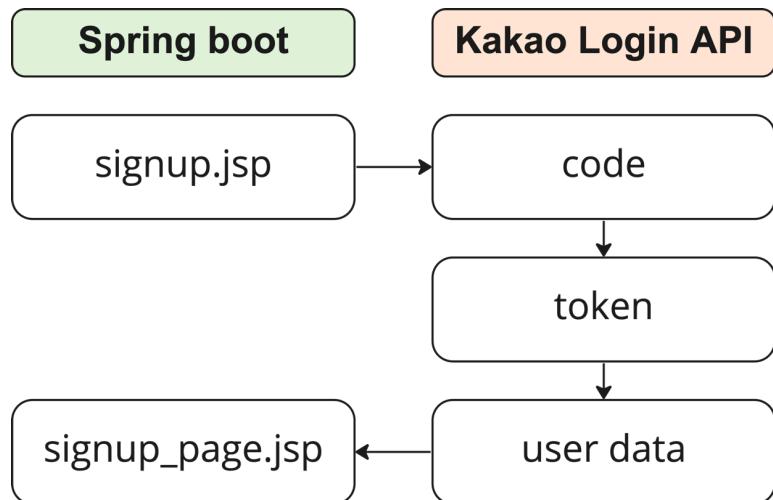
Scraping 한 데이터의 주소를 Geocoding API 이용하여
경도 위도를 찾아서 이를 python folium의 marker로 시각화하고
marker cluster를 이용하여 비슷한 범위의 마커들을 합쳐서 개수로 표시
클릭 시에 세부적인 영역으로 나누게 시각화 하였다.



Kakao Login API

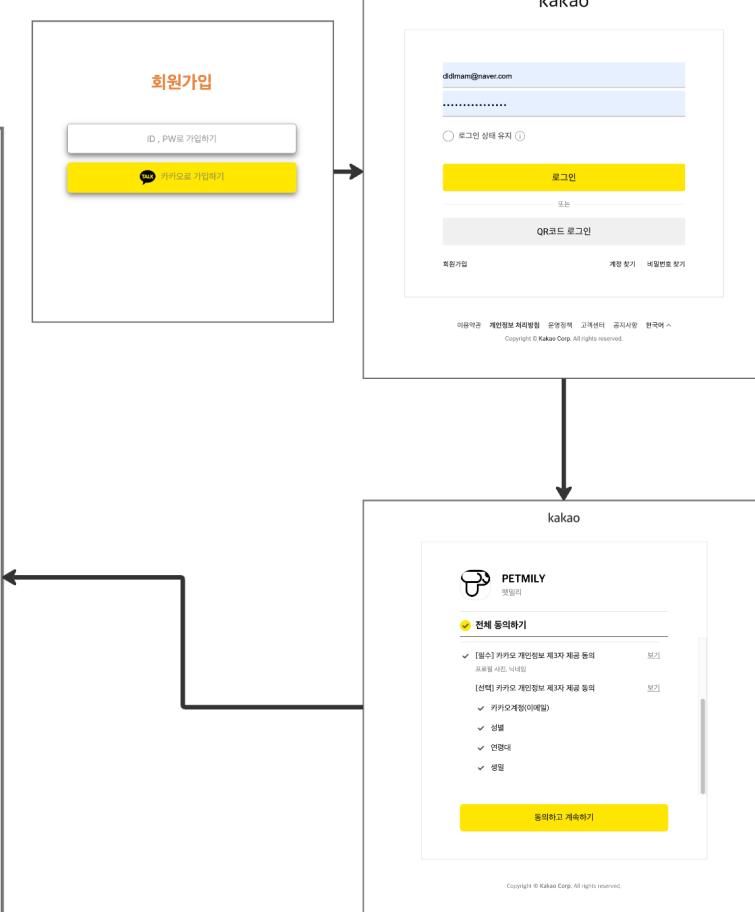
Kakao Login API를 이용해서 회원가입 시 필요한 정보 가져오기

Flow Diagram



구현 화면

This screenshot shows the implementation of the sign-up form. It includes fields for '비밀번호 확인' (Password Confirmation), '이름' (Name), '닉네임' (Nickname), '이메일' (Email) with placeholder 'petmily22@gmail.com 또는 petmily22@naver.com', '이메일 인증번호' (Email Verification Number), '휴대폰 번호' (Phone Number), and '주소' (Address). A yellow '인증' (Verify) button is positioned next to the email field. The entire form is contained within a '회원가입' (Sign Up) box.



AI 모델링

목 차 | 1 머신 러닝 지도학습 회귀 모델

주제 | 서울시 상권 분기당 매출 금액 예측 모델 - python

1.1 EDA	27
-------------------------	----

1.2 Modeling	28
------------------------------	----

2 머신 러닝 지도학습 분류 모델

주제 | 서울시 전세 금액 예측 모델 - R

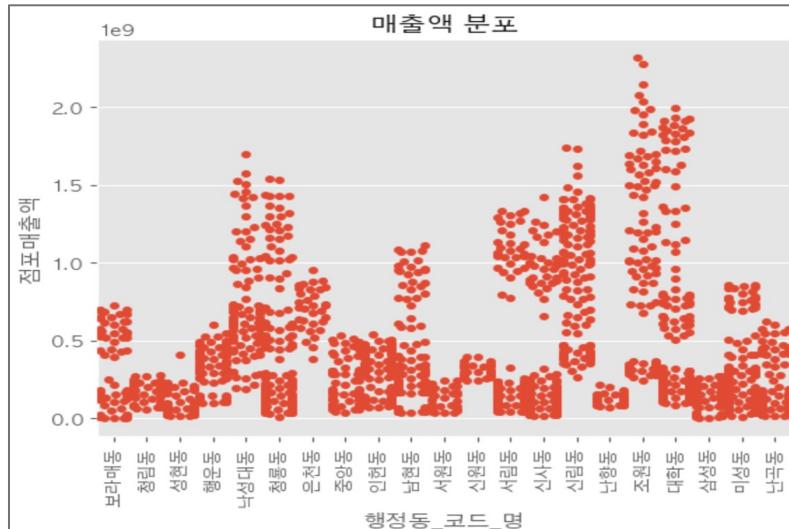
2.1 EDA	29
-------------------------	----

2.2 Modeling	30
------------------------------	----

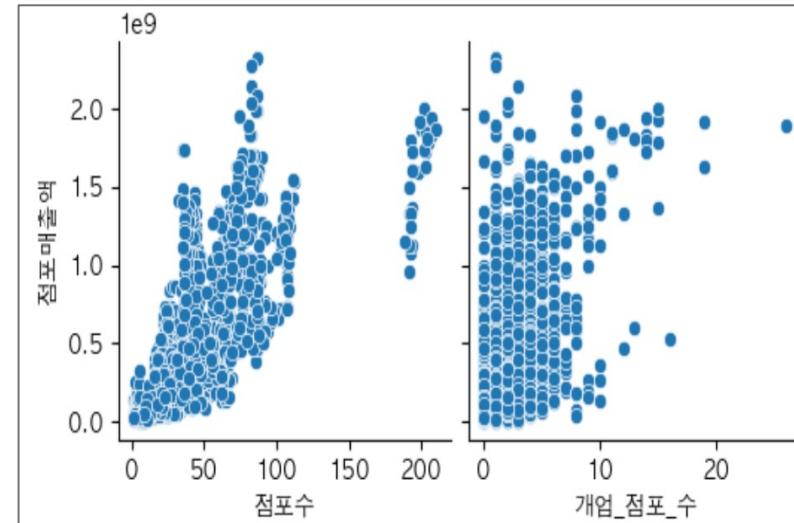
EDA

서울시 상권 데이터 중 머신 러닝 지도 학습의 회귀 모델의 feature로 사용 가능한 column 찾기

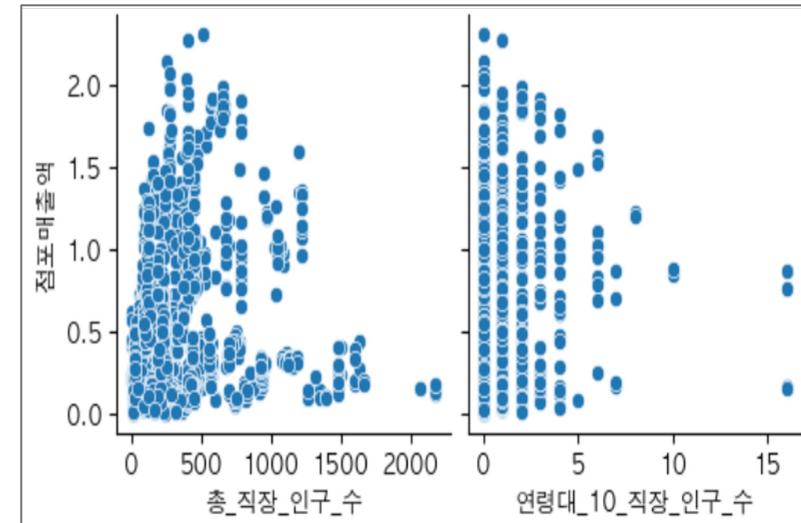
동 이름



점포 수 관련



직장 인구 수 관련



분석

동 별로 매출액의 분포가 다르다는 것을 확인할 수 있습니다.
점포 수, 개업 점포 수, 직장 인구 수, 10대 직장 인구 수가
매출액과 선형적인 관계를 가지는 것을 확인 할 수 있다.

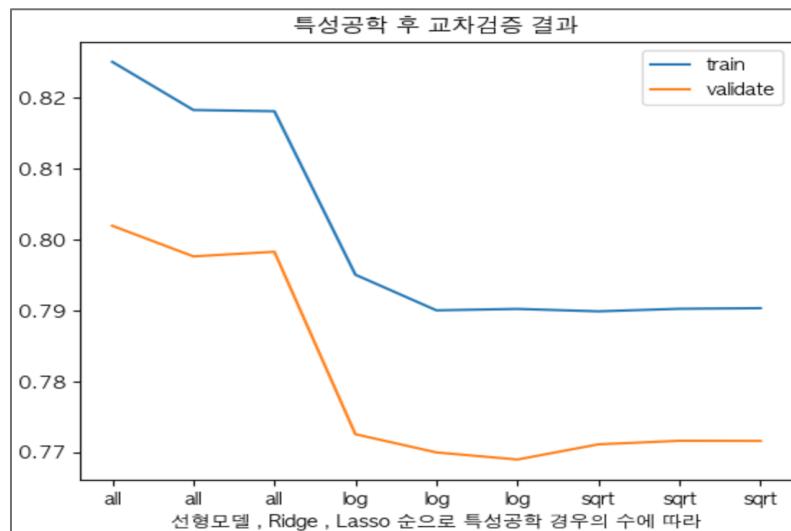
결론

선형적인 관계를 가지는 column별 매출액과의 분포가 다르고,
동 이름별로도 매출액의 분포가 다르기 때문에
위 5개의 column을 feature로 사용할 수 있다고 판단했습니다.

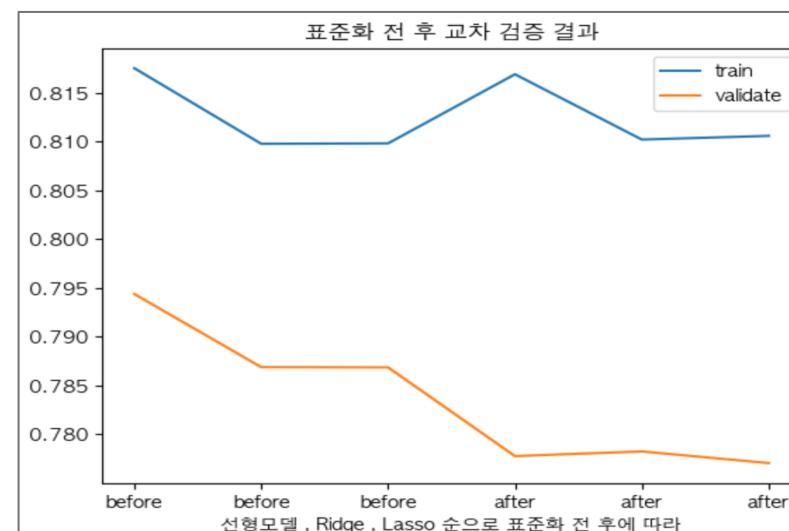
Modeling

표준화, 특성공학, 교차 검증을 통해 예측률을 높이고 grid search를 통해 일반화된 머신 러닝 모델 만들기

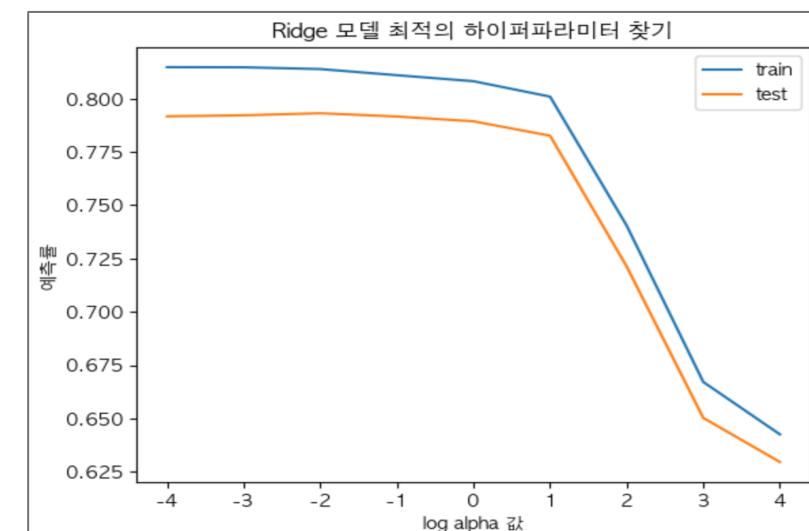
특성공학 후 교차 검증 결과



표준화 전 후 교차 검증 결과



최적의 hyper parameter값 찾기



분석

특성공학 한 것을 모두 사용했을 때 교차 검증결과가 가장 높습니다.
표준화 전후의 train 데이터의 예측률은 비슷하지만,
표준화 전의 validate 예측률이 더 높습니다.

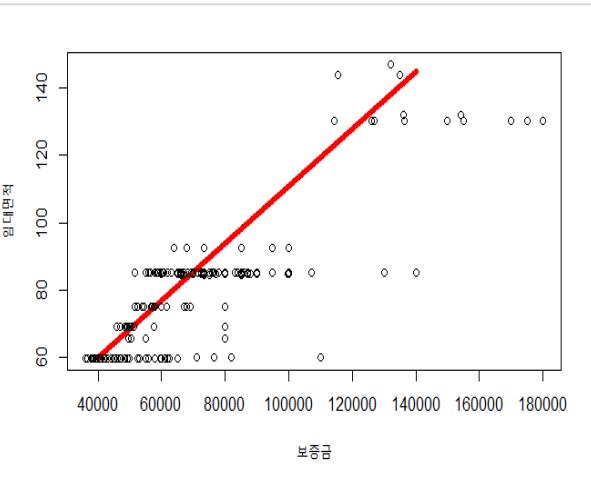
결론

특성공학 한 것을 모두 사용하고, 표준화 하지 않았을 때의 데이터를 가지고
Ridge 모델의 alpha 값을 10으로 했을 때가 예측률이 가장 높은 모델이고,
이를 매출 예상 머신 러닝 모델로 사용하기로 했습니다.

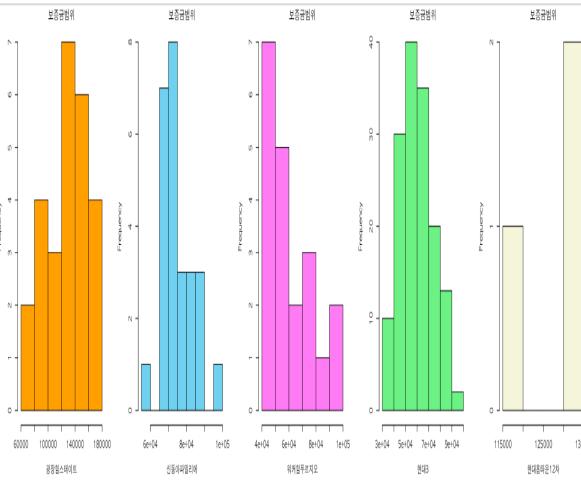
EDA

서울시 전세 데이터 중 머신 러닝 지도 학습의 분류 모델의 feature로 사용 가능한 column 찾기

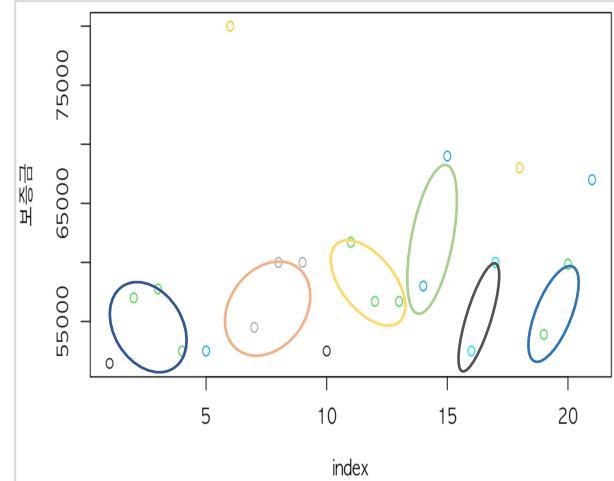
임대 면적



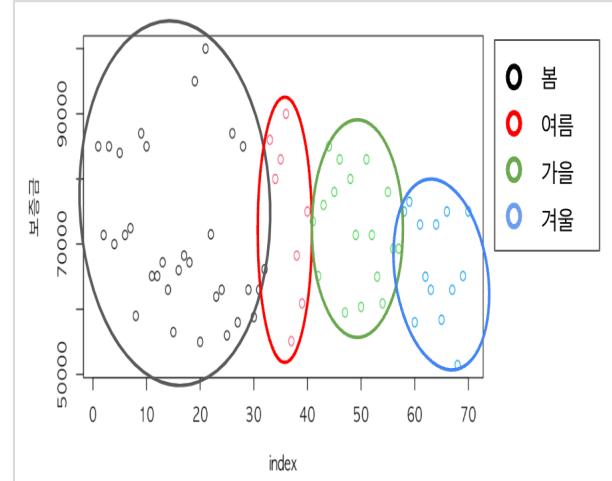
아파트 명



층 수



계약 계절



분석

보증금과 각 부제목과의 관계를 시각화 시킨 것입니다.

임대면적은 보증금과 선형적인 관계를 가진다는 것을 확인 할 수 있습니다.

나머지는 값들 별로 보증금의 분포가 다르다는 것을 확인 할 수 있습니다.

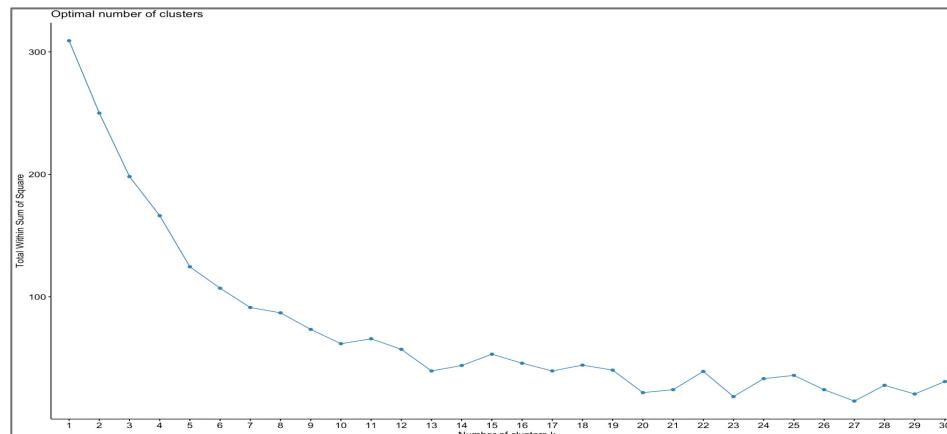
결론

위 4개의 column이 feature로서 적합하다고 판단했습니다.

Modeling

4개의 feature를 clustering 하여 label별 보증금 금액의 범위를 target으로 하여 모델링

Clustering



결론

elbow method를 통해 군집의 개수를 20으로 하기로 결정

Modeling

예측률	정규화 전(%)	정규화 후(%)
인공신경망	47.6	98.9
랜덤포레스트	97.8	98.5
SVM	93	93.4

결론

feature를 정규화 한 후

인공신경망 모델을 사용할 경우의 예측률이 제일 높으므로
해당 모델을 머신 러닝 모델로 채택하였습니다.

기타 기술 스택

목 차 | 1 데이터 모델링

1.1 [ERD](#) 32

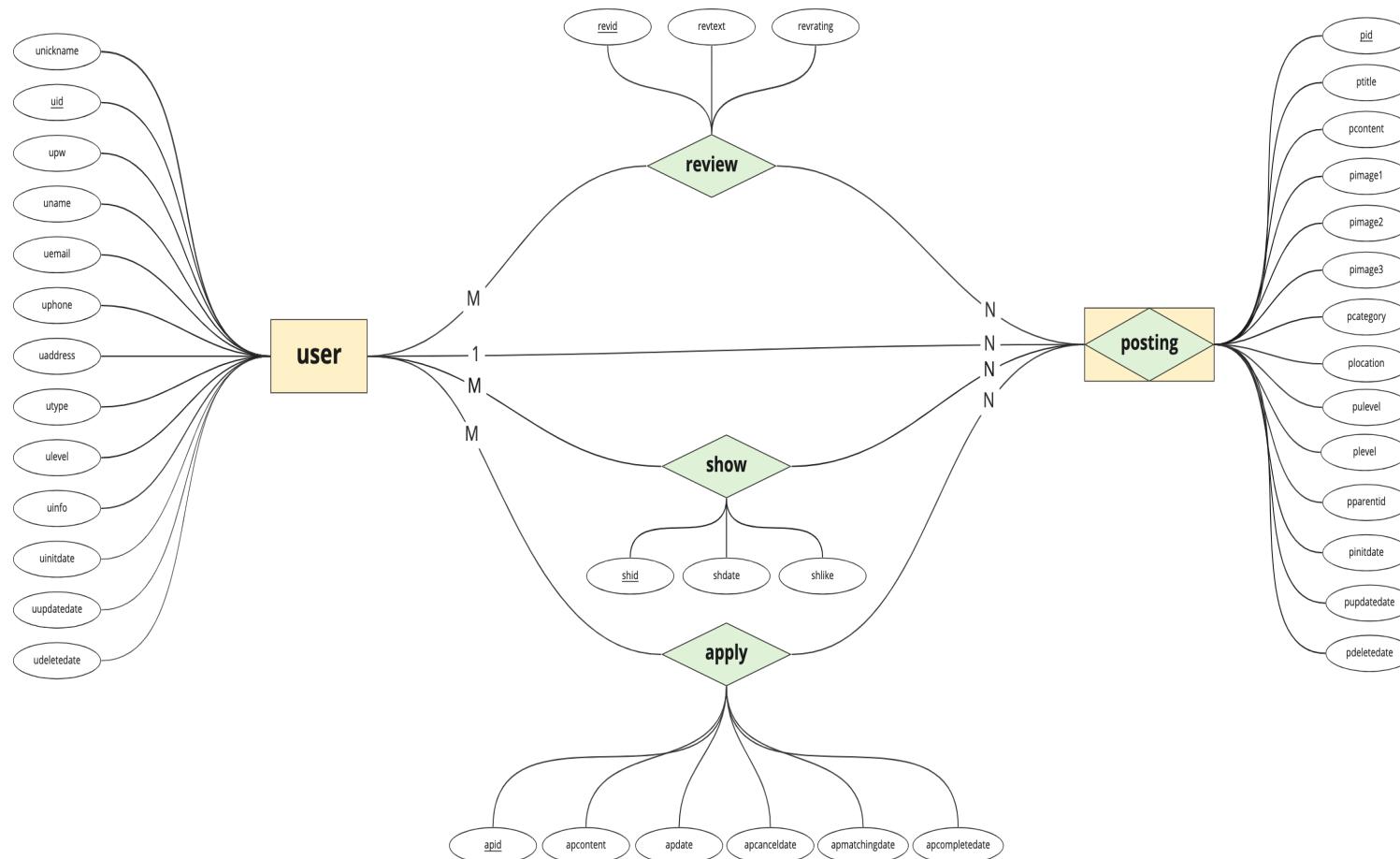
2 Scrapyng

2.1 [selenium을 이용한 Scrapyng](#) 33

2.2 [multi processing](#) 34

ERD

전체 ERD 중 Project의 핵심인 유저와 게시글 Entity 사이의 Relation Diagram



Entity

- **user** : 유저
- **posting** : 게시글

Relation

- **posting**
유저가 게시글을 작성한다는 관계를 가지기에 entity 이자 relation으로 표시
- **review / show / apply**
후기글과 평점 / 조회수와 좋아요 / 신청글은 유저와 게시글 사이의 관계

Attribute

- 게시글 삭제시 DELETE 문 사용하지 않고, UPDATE 문으로 delete date를 UPDATE한다.

Selenium & BeautifulSoup

올리브영 전국 매장 이름과 주소 scraping

1

```
scroll_bar = event.find_element(By.CSS_SELECTOR, 'div[class="sroll_store scrbar mCustomScrollbar _mCS_2"]')
```

scroll 하고 reload를 하기 위해서 PAGE_DOWN action 실행될 div 설정

2

```
while True:
    before_height=event.execute_script("return document.querySelector('div[class='mCSB_container']").scrollHeight")
    event.execute_script("document.querySelector('div[class='mCSB_container'].style.top = '-' + str(before_height) + 'px'')")
    actions = webdriver.ActionChains(event).send_keys_to_element(scroll_bar).send_keys(Keys.PAGE_DOWN)
    actions.perform()
    time.sleep(2)

    now_height=event.execute_script("return document.querySelector('div[class='mCSB_container'].scrollHeight")

    if now_height == before_height:
        break
```

css selector의 top을 div의 최대 높이로 설정하여 scroll 하고, reload를 위해 PAGE_DOWN 액션을 준다.

scroll 되기 전 높이와 된 후의 높이를 비교하여 마지막 까지 scroll이 됐는지 확인 할 수 있다.

3

```
html = event.page_source
soup = BeautifulSoup(html, 'html.parser')
```

```
shops=soup.select('#wordStoreList a')
addresses=soup.select('#wordStoreList p')
```

BeautifulSoup을 이용하여 마지막 까지 scroll이 된 page를 parsing해서 내가 원하는 정보인 가게 이름과 가게 주소를 선택하여 변수에 저장한다.

Python MultiProcessing Scrapy

가장 맛있는 족발 가게명, 주소, 전화번호, 이미지 저장하여 csv 파일로 저장

1

```
if __name__ == "__main__":
# recursionlimit 늘려주기
sys.setrecursionlimit(10000)

p = Pool(os.cpu_count - 2)

# 가게 몇 개인지 판단하기 위해서
url='https://gajok.kr/sub/store.php'

res = req.urlopen(url)

soup = BeautifulSoup(res,"html.parser")

shops = soup.select('dt')

shops = [shop.text for shop in shops]
shops.remove('')

# multi processing
ret1 = p.apply_async(get_data,(1,100,len(shops)+1))
ret2 = p.apply_async(get_data,(100,200,len(shops)+1))
ret3 = p.apply_async(get_data,(200,250,len(shops)+1))
ret4 = p.apply_async(get_data,(350,len(shops)+1,len(shops)+1))
ret5 = p.apply_async(get_data,(250,300,len(shops)+1))
ret6 = p.apply_async(get_data,(300,350,len(shops)+1))

names = ret1.get()[0]+ret2.get()[0]+ret3.get()[0]+ret4.get()[0]+ret5.get()[0]+ret6.get()[0]
addresses = ret1.get()[1]+ret2.get()[1]+ret3.get()[1]+ret4.get()[1]+ret5.get()[1]+ret6.get()[1]
phones = ret1.get()[2]+ret2.get()[2]+ret3.get()[2]+ret4.get()[2]+ret5.get()[2]+ret6.get()[2]
images = ret1.get()[3]+ret2.get()[3]+ret3.get()[3]+ret4.get()[3]+ret5.get()[3]+ret6.get()[3]

df=pd.DataFrame(data=zip(names,addresses,phones,images), columns=['Name','Address','Phone','Image'])
df.to_csv('./Data/gajok2.csv',encoding='utf-8',index=False)

p.close()
p.join()
```

multi process로 실행되기 위해
cpu의 코어 개수 파악

가져와야 될 데이터의 개수 파악
하여 이를 해당 되는 core 개수에
맞게 개수를 분할

작업 완료 후 하나의 data-frame으로 만들어 csv로 저장

image는 따로 저장하고 이의 url을 문자열로 변수로 저장해 두었다.

```

get_data(start,end,len):
    chrom_options = webdriver.ChromeOptions()
    driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()),options=chrom_options)
    driver.get("https://gaok.kr/sub/store.php")

    time.sleep(20)

    names = []
    addresses = []
    phones = []
    images = []

    for i in range(start,end):
        try:
            xpath = f'//div[@id="result_store"]//div[1]/ul/li[{i}]/a'
            driver.find_element(By.XPATH,xpath).click()

            time.sleep(1)

            xpath = f'{i}/a/div[6]/div[1]/div/div[6]/div[1]/{len}]/div/div[2]/a'
            driver.find_element(By.XPATH,xpath).click()

            time.sleep(1)

            # 웹툰 이미지
            # selenium을 Beautiful soup 이용해서 책속 정보 가져오기
            html = driver.page_source
            soup = BeautifulSoup(html, 'html.parser')

            # 가격
            names.append(soup.select_one('h1').text)
            print(f"제작일 : ",soup.select_one('h1').text)

            # 주소
            address = soup.select_one('div.info_l1')
            addresses.append(address.text.replace('책판주소',''))
            print(f"책판주소 : ",address.text.replace('책판주소',''))

            # 전화번호
            phone=soup.select_one('div.info_a')
            phones.append(phone.text)
            print(f"전화번호 : ",phone.text)

            # ISBN
            ISBN()
            image = driver.find_element(By.XPATH,'//div[@id="detail"]//div/div/div[1]/div')
            url="https://gaok.kr"+image.get_attribute('style').split('}')[-2][23:-2]

            # 사진 확장자 바꾸기
            url=url.replace("%","%20")

            # 파일 찾기
            url_korre.compile('(-+|-+|-+)[-\\W]+').findall(url)

            if not url_ko is None:
                for ko in url_ko:
                    enc_korre.quote_plus(ko)
                    url_korre.replace(ko,enc_ko)

            images.append(url)
            print(f"이미지 : ",url)

            saveName = f'{i}/Image/{i}.jpg'
            res.urlretrieve(url, saveName)

            xpath = f'//div[@id="detail"]//div/div/a'
            driver.find_element(By.XPATH,xpath).click()
            time.sleep(1)

        except:
            print("error")

    return names,addresses , phones , images

```

multi process로 실행될 함수