2024

대학·기업협력형
SW아카데미사업

프로젝트 결과 보고서

K-Software Empowerment BootCamp

K-Software Empowerment BootCamp(KSEB) 3기

2024.08.16

4조 (페메해조 - Face Manager Happy and Joy)

contents

*****

I 서론

프로젝트 소개

AI얼굴 인식 모델을 사용하여 피부 타입을 5가지로 분석하고,
개인 맞춤형 화장품과 영양제 제품 추천 기능을 제공하는 앱 서비스를 제작했습니다.

프로젝트 주제	AI기반 피부 케어 서비스	팀명	페메해조 (Face Manager Happy and Joy)
프로젝트 기간	2024.07.01~2024.08.15	참여인원	6명

구성 및 역할

NO	이름	역할	담당업무	학교
1	노형준	팀장	Flutter기반 인터페이스 설계 및 개발	경기대학교
2	김경영	팀원	백엔드 기능 구현, Flask 서버 관리	경기대학교
3	김민상	팀원	백엔드 설계 및 구현, AWS 서버 구축	경기대학교
4	박진환	팀원	프론트 기능 구현 및 데이터 마이닝	경기대학교
5	신효식	팀원	AI 모델 설계, 개발 및 데이터 마이닝	경기대학교
6	최화영	팀원	백엔드 기능 구현 및 AI모델 설계	경기대학교

<table>
<tr><td>활용기술</td><td>
<table>
<tr><th>NO</th><th>구분</th><th>내용</th></tr>
<tr><td>1</td><td>Languages</td><td>Java, Python, Dart</td></tr>
<tr><td>2</td><td>Framework</td><td>Spring Boot, Flutter, Flask, PyTorch</td></tr>
<tr><td>3</td><td>Database</td><td>MySQL</td></tr>
<tr><td>4</td><td>etc</td><td>Roboflow(학습용)</td></tr>
</table>
</td></tr>
<tr><td>개발환경</td><td>COLAB, AWS, Intelli J, Android Studio, Pycharm</td></tr>
<tr><td>프로젝트
주제선정 배경
및 기획의도</td><td>1. 전 세계 스킨케어 및 화장품 시장이 급성장하고, 소비자들은 더 효과적인 제품을 찾기 위해 노력하고 있습니다. 이에 맞추어 저희는 <u>피부 관리 비용을 절감하고 피부 미용에 대한 접근성을 높이는 서비스를 기획</u>했습니다.

2. 분석 결과를 바탕으로 <u>개인 맞춤형 스킨케어 제품과 영양제를 추천</u>합니다. 또한, <u>커뮤니티 기능</u>을 통해 사용자 간 소통창구를 만들고, 이를 통해 추천 알고리즘을 개선하도록 반영했습니다.

3. 사용자는 간단한 얼굴 사진 촬영으로 자신의 피부 상태를 진단받고, 즉각적으로 제품을 추천받습니다. 이 서비스로 <u>효과적인 피부 관리</u>를 가능하도록 합니다.</td></tr>
<tr><td>개발 주요 사항</td><td>1. Flask와 Flutter의 REST API 통신
REST API를 이용한 Flutter의 통신으로 데이터를 주고받는 환경을 구축했습니다. 이를 통해 Flutter와 Flask 서버 간의 안정적인 데이터 통신이 가능합니다.

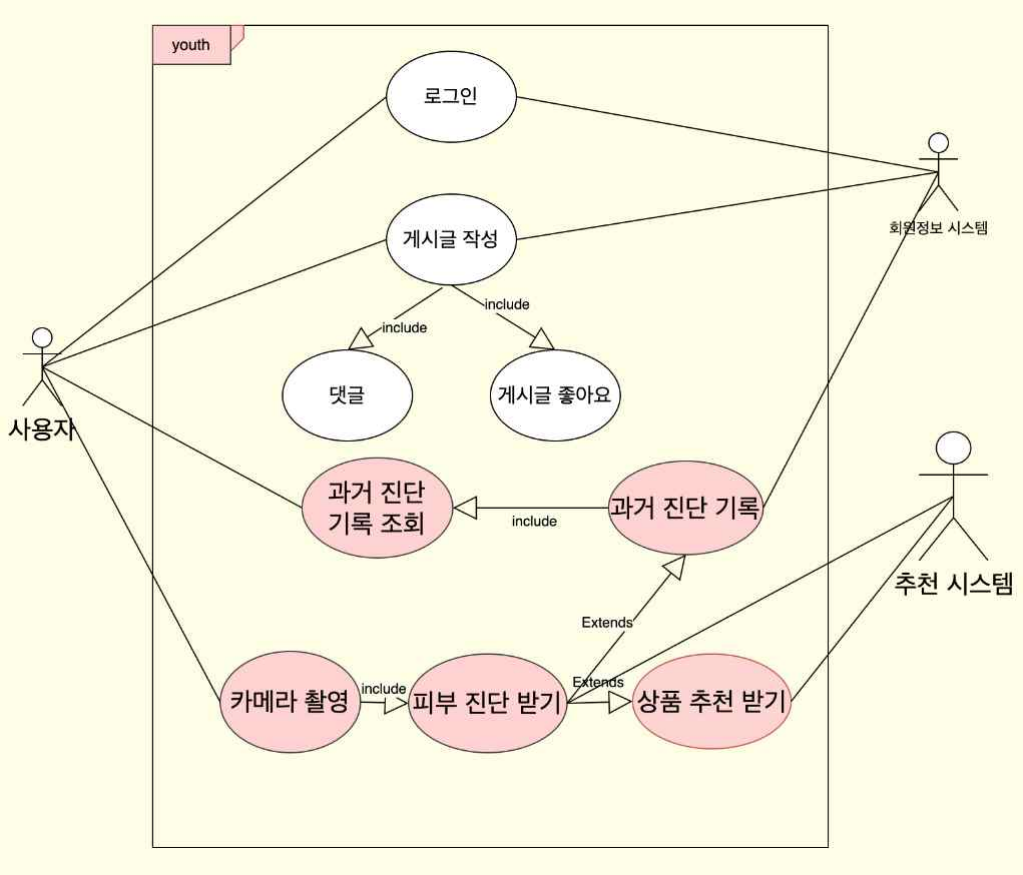
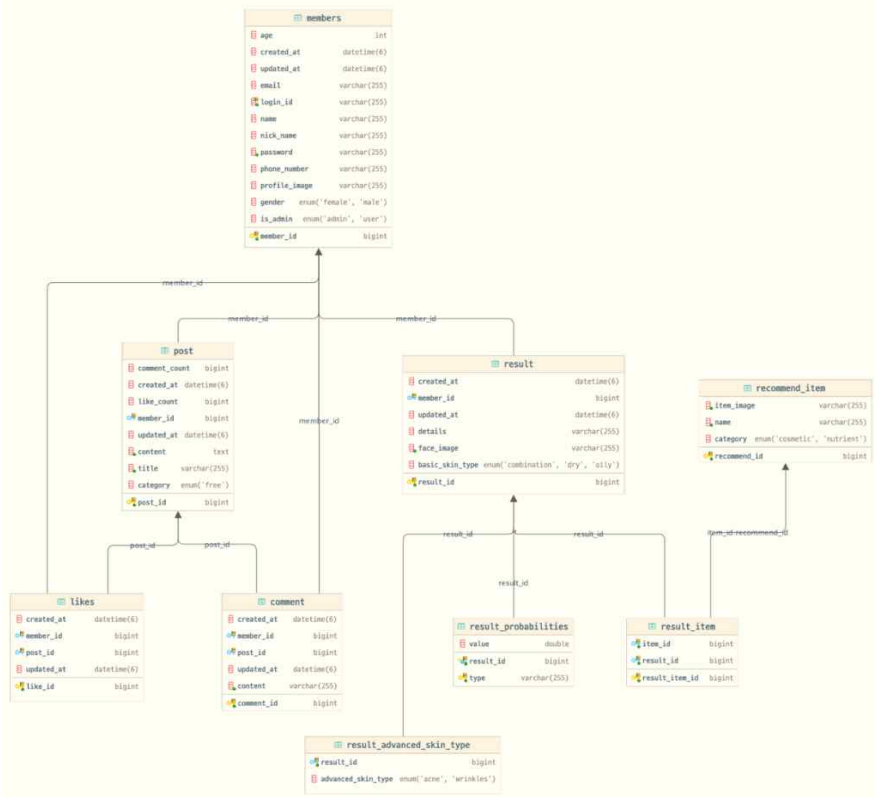
2. Spring Boot와 Flutter의 REST API 통신
Spring Boot를 사용하고 Flutter 클라이언트와의 통신을 구현하였습니다. Spring Boot의 강력한 서비스 계층과 데이터 관리 기능을 활용하여 안정적인 API 서비스와 데이터 처리 로직을 설계하였습니다.

3. 데이터베이스(DB) 설계
JPA를 사용하여 객체 지향 프로그래밍과 관계형 데이터베이스 간의 패러다임 불일치를 해결하고, 데이터 관리를 단순화하였습니다. 또한, 데이터 접근과 쿼리의 효율성을 높여 애플리케이션의 성능을 개선하였습니다.

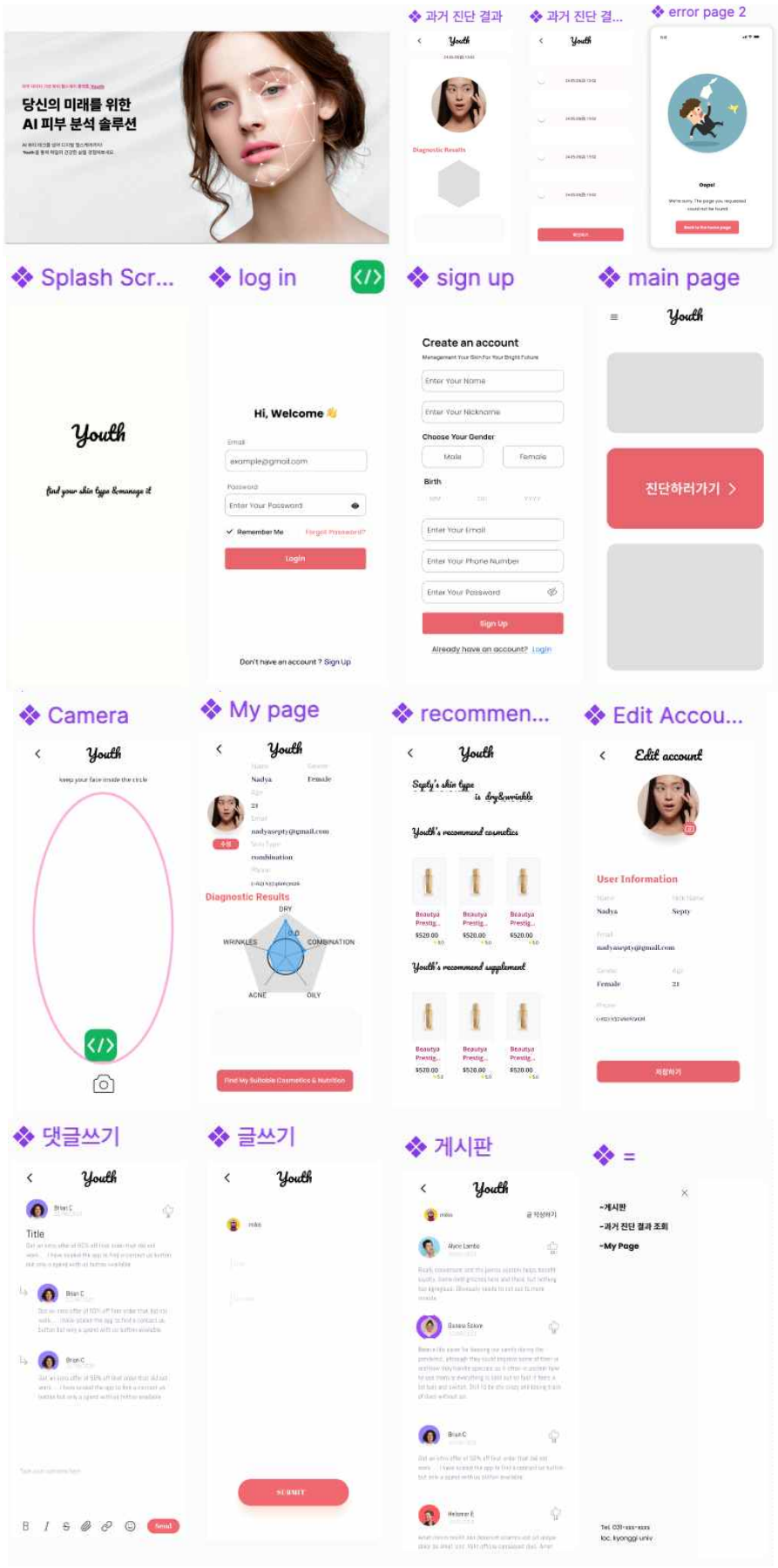
4. 앱 서비스 기능 구현
1) 피부 타입 진단: AI 기반의 얼굴 인식 기술을 활용하여 사용자는 지성, 건성, 복합성, 주름, 여드름으로 크게 5가지 타입으로 진단받을 수 있습니다.
2) 추천 모델: 사용자는 진단받은 피부 타입을 바탕으로 개인 맞춤형 화장품, 영양제를 추천받을 수 있습니다.
3) 과거 진단 결과 보기: 사용자는 이전에 받은 진단 결과를 조회할 수 있습니다.
4) 마이페이지: 사용자 개인 정보를 확인하고 수정 가능합니다, 최근 진단 결과를 확인할 수 있습니다.
5) 커뮤니티: 서비스 사용자들이 커뮤니티 기능을 통해 소통할 수 있도록 게시판, 댓글, 좋아요 기능을 구현하였습니다.</td></tr>
</table>

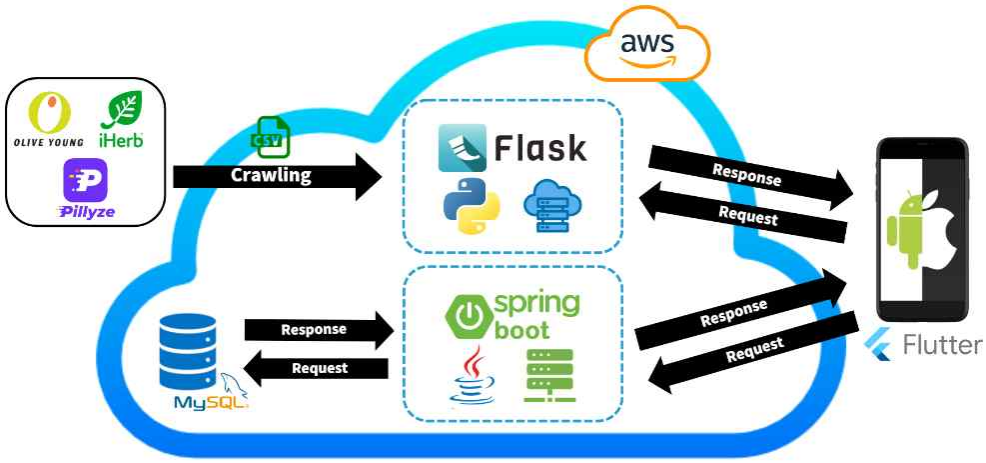
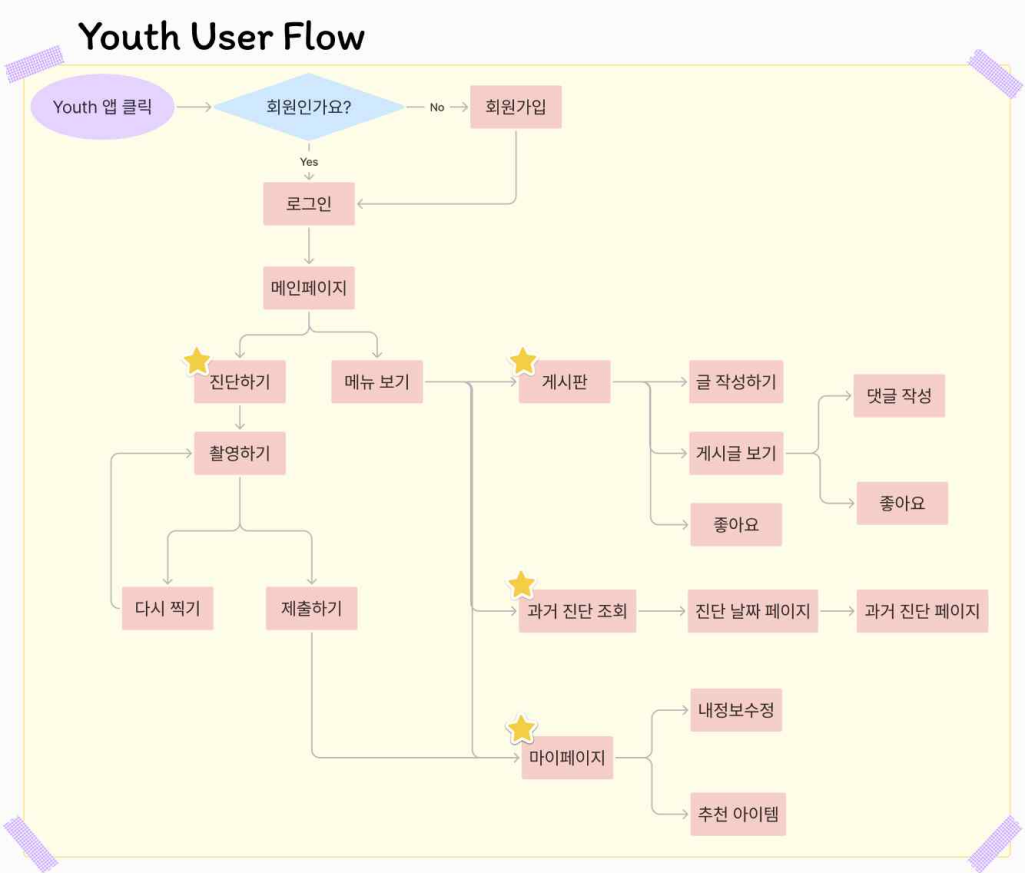
프로젝트 수행절차 및 방법	**1. 사전기획** 1) 시장 조사 및 주제선정 2) 요구사항 명세서, 백로그 작성 3) ERD 설계 및 UI 화면 설계 ➔ **2. 데이터 수집** 1) 피부 관련 데이터 수집과 분석 2) 화장품 관련 데이터 수집과 분석 3) 영양제 관련 데이터 수집과 분석 ➔ **3. 데이터 전처리** 1) 피부 데이터 증강을 통한 전처리 2) 화장품 및 영양제 데이터 크롤링 및 JSON 파싱 **4. 모델링** 1) 기본 피부타입 측정 모델 (pre-trained) 2) 심층 피부타입 측정 모델 (pre-trained) 3) 화장품 추천 모델 1 (contents based-model) 4) 화장품 추천 모델 2 (KNN model) 5) 영양제 추천 모델 ➔ **5. 서비스 구축** AI: Flask, PyTorch FE: Flutter BE: Spring Boot, AWS DB: MySQL API : RESTful-API Crawling: Python ➔ **6. 배포 및 수정** 1) Docker를 통해 일관된 실행환경을 구축 2) 안정화된 AWS 서버 구축 및 CloudWatch를 에러 로그 분석 3) GitHub Actions을 이용한 CI/CD 환경 구축

II 설계

Use Case 설계	
ERD 설계	

UI 설계


당신의 미래를 위한
AI 피부 분석 솔루션
과거 진단 결과
과거 진단 결...
error page 2
Splash Scr...
log in
sign up
main page
Youth
find your skin type & manage it
Hi, Welcome
Email
example@gmail.com
Password
Enter Your Password
Remember Me
Forgot Password?
Login
Don't have an account ? Sign Up
Create an account
Management Your Skin For Your Bright Future
Enter Your Name
Enter Your Nickname
Choose Your Gender
Male
Female
Birth
Enter Your Email
Enter Your Phone Number
Enter Your Password
Sign Up
Already have an account? Login
진단하러가기 >
Camera
My page
recommen...
Edit Accou...
keep your face inside the circle
Diagnostic Results
DRY
COMBINATION
OILY
ACNE
WRINKLES
Find My Suitable Cosmetics & Nutrition
Youth's recommend cosmetics
Youth's recommend supplement
Edit account
User Information
댓글쓰기
글쓰기
게시판
=
Title
Send
SUBMIT
-게시판
-과거 진단 결과 조회
-My Page

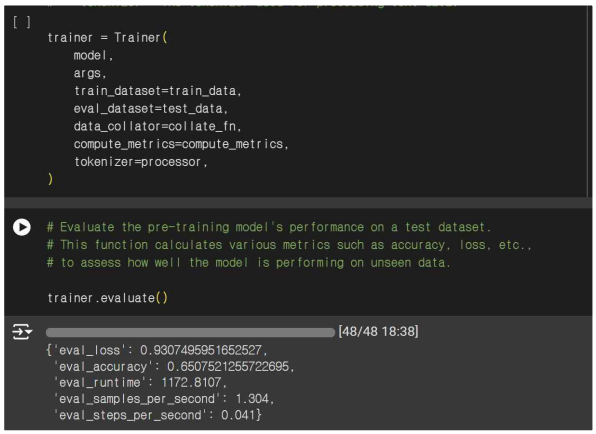
시스템 아키텍처 설계	
User Flow	

III 연구방법

프로젝트 수행 내용	**<Front-End 설계>** **1. UI/UX 설계** - **Figma 화면 설계 툴을 이용**하여 UX와 UI를 고려하여 설계하였습니다. 컬러와 화면 간의 이동 등을 고려해서 화면 이용 시 어색하지 않도록 하였습니다. 또한, Figma에서 제공하는 협업 툴을 이용해서 팀원들의 피드백을 받고 개선해나갔습니다. 예를 들면, 초기에는 전문적인 내용이 많아 읽기 힘든 진단 페이지였습니다. 이를 **사용자가 이해하기 쉬운 Radar Chart와 함께 직관적인 진단 결과 페이지로 수정**하였습니다. (7페이지 참조) **2. 디자인 언어 -** 어플리케이션 전반에 걸쳐 통일된 컬러 팔레트와 타이포그래피가 사용되었습니다. **Flutter의 Material Design 위젯을 활용하여 일관된 디자인을 유지**했습니다. **3. 반응형 UI -** 다양한 화면 크기와 해상도에서 최적의 사용자 경험을 제공하기 위해 반응형 레이아웃을 설계했습니다. **Flutter의 MediaQuery와 LayoutBuilder를 활용해 화면 크기에 따라 UI 요소들이 자동으로 재배치되도록 구현**했습니다. **4. 내비게이션 -** 사용자가 쉽게 탐색할 수 있도록 직관적인 내비게이션 시스템을 도입했습니다. **Flutter의 Navigator와 BottomNavigationBar를 사용해 주요 화면 간 전환이 용이하도록 설계**했습니다. **5. 크로스 플랫폼인 Flutter 사용 -** 하나의 코드로 iOS와 Android에서 모두 사용가능하도록 Flutter 프레임워크를 사용하였습니다. 또한, **플러터의 Custom Widget을 통해 저희 앱의 20개의 페이지들의 유지 보수성을 향상**시켰습니다. 예를 들어, 2번 이상 사용되는 회원 수정 기능에 사용되는 버튼이나 피부 진단 결과의 레이아웃 등을 위젯을 통해 관리하였습니다. **6. DTO와 Freezed 패키지를 활용한 데이터 처리** - **데이터 일관성을 위해 DTO를 사용하여 서버와 교환하는 데이터를 표준화**하였습니다. 이로 인해 초기에 API마다 다른 요청 데이터에 대해 null값으로 처리하던 코드를 엔티티에 영향 주지 않는 DTO를 사용하여 무결성을 유지하였습니다. 또한, Freezed 패키지로 불변 데이터 클래스를 생성하였습니다. 예를 들어, 피부 진단 데이터에 대해 flask 서버에서 받은 결과 그대로 유지하여 신뢰성을 높였습니다.

<AI 설계>

1. 기본 피부타입 측정 모델(시행착오) - HuggingFace의 pre-trained model에 해당합니다. Huggingface의 transformer 라이브러리의 기능인 ViTImageProcessor와 ViTForImageClassification을 활용해서 기본 피부타입 측정 모델을 구축하려했으나, **데이터셋을 학습하는 과정에서 라벨링된 데이터셋을 학습시키는 것이 아니라, 라벨링 이름을 가진 폴더를 가지고 이미지를 분류시킨 상태에서 학습시키는 모델인 것을 발견**했습니다. 그래서 이미지들이 기준에 맞게 제대로 분류되지 않았다고 판단했고, 심지어 **학습 결과 정확도도 높지 않아서 다른 모델을 사용하기로 결정**했습니다.



<그림1. HuggingFace 기본 피부타입 측정 모델 평가 결과>

1-1. 기본 피부타입 측정 모델 - Kaggle의 최적화된 피부 타입 분류 pre-trained 모델을 확보했습니다. MobileNetV2 모델을 사용하여 경량화 및 모바일 시스템에서 효율적으로 동작하도록 설계, Depthwise Separable Convolutions와 Inverted Residuals with Linear Bottlenecks를 통해 연산 효율성을 높였습니다. 사전 학습된 가중치 ImageNet을 사용하여 다양한 비전 작업에 대해 모델의 성능을 향상시켰습니다. **모델을 저희가 원하는 결과를 출력하게 하기 위해서 모델의 특정 layer 이후의 layer들을 재학습**을 했습니다.
Flatten을 사용하여 출력 데이터의 벡터를 변환하고 Dense를 사용하여 256 및 128 유닛을 가진 두 개의 은닉 층을 추가했습니다. 그리고 마지막으로 3개의 클래스에 대한 예측을 위해 softmax 활성화 함수를 사용하는 층을 추가했습니다. **최종적으로 피부 상태에 따른 dry, oily, combination을 탐지하는 모델을 구현**했습니다.

<그림2. MobileNetV2 기본 피부타입 측정 모델 학습 결과>

2. 심층 피부타입 측정 모델(시행착오) - 기존에는 여드름, 아토피, 주름을 심층 진단으로 탐지하려고 했으나 **아토피는 얼굴이 아닌 몸에 있는 데이터가 많아 모델을 학습 시킬 데이터를 얻기 힘들다고 판단하여 심층 진단으로는 여드름과 주름만 탐지하는 걸로 방향성을 잡았습니다.**

첫 번째 시도) AI허브에 있던 심층진단 모델을 사용했습니다. 하지만, 이 모델은 여드름을 탐지하지 않아 적합하지 않다고 판단하여 보류하였습니다.

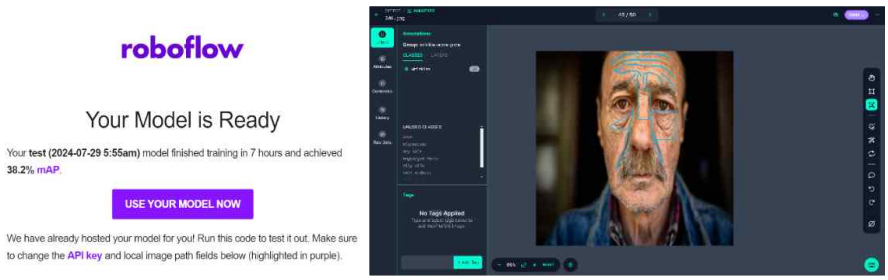
두 번째 시도) AI 허브에서는 데이터셋만을 가져와서 로보플로우로 학습시킨 모델을 받도록 했습니다. 로보플로우에서는 cocojson 형식의 파일만을 지원합니다. 따라서 json 형식인 데이터 셋의 라벨링 파일은 변환하는 작업이 필요했습니다. 파이썬을 통해 이를 해결하고, 로보플로우에 원천데이터와 변환한 라벨링 데이터를 업로드하여 pre-trained model을 얻었습니다. 모델을 돌려본 결과 mAP가 0.34로 AP의 평균이 낮은 편에 속하여 이 모델을 보류한 뒤, 다른 모델을 찾게되었습니다.

결과) 로보플로우에서 공개된 데이터셋을 사용하여 다시 학습시켰습니다. **데이터셋의 라벨링 과정을 재검토하여 잘못된 부분을 수정하였으나, 얻은 모델의 mAP 값이 여전히 낮았습니다.** 이후 **추가적인 데이터셋과 알고리즘을 테스트하며 최적의 모델을 찾기 위해 노력**하였습니다.

```
2024-07-11 02:45:20,817
Sub_0585_Equ_01_Angle_R15_Area_4(wrinkle)==> Pred: 0.281 /
Gt: 0.320 ==> MAE: 0.039
2024-07-11 02:45:20,899
Sub_0585_Equ_01_Angle_R15_Area_5(moisture)==> Pred: 0.178
/ Gt: 0.553 ==> MAE: 0.376
2024-07-11 02:45:20,903
Sub_0585_Equ_01_Angle_R15_Area_5(elasticity)==> Pred: 0.184
/ Gt: 0.418 ==> MAE: 0.234
2024-07-11 02:45:20,906
Sub_0585_Equ_01_Angle_R15_Area_5(pore)==> Pred: 0.133 /
Gt: 0.332 ==> MAE: 0.199
2024-07-11 02:45:20,988
Sub_0585_Equ_01_Angle_R15_Area_6(moisture)==> Pred:
0.484 / Gt: 0.523 ==> MAE: 0.039
2024-07-11 02:45:20,991
Sub_0585_Equ_01_Angle_R15_Area_6(elasticity)==> Pred:
0.276 / Gt: 0.480 ==> MAE: 0.204
2024-07-11 02:45:20,994
Sub_0585_Equ_01_Angle_R15_Area_6(pore)==> Pred: 0.490 /
Gt: 0.310 ==> MAE: 0.180
2024-07-11 02:45:21,118
Sub_0585_Equ_01_Angle_R15_Area_8(moisture)==> Pred:
0.535 / Gt: 0.457 ==> MAE: 0.078
2024-07-11 02:45:21,123
Sub_0585_Equ_01_Angle_R15_Area_8(elasticity)==> Pred:
0.556 / Gt: 0.384 ==> MAE: 0.172
2024-07-11 02:45:21,127 count: 0.4505 // moisture: 0.21 //
wrinkle: 0.0902 // elasticity: 0.3032 // pore: 0.4835
2024-07-11 02:45:21,129 [1 / 1393] Total Average MAE => 0.307
2024-07-11 02:45:21,131
```

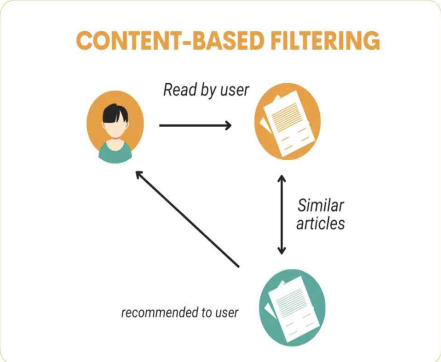
<그림3. AIhub 심층 피부타입 측정 모델 평가 결과>

2-1. 심층 피부타입 측정 모델 - **Roboflow를 활용해서 YOLOv10 모델을 시행착오 과정에서 수집한 데이터로 직접 학습시켰습니다.** 여러 개의 모델을 사용해보면서 수집한 데이터셋을 Roboflow에서 지원하는 COCO JSON 형식으로 변환 및 전처리를 한 후에 학습을 진행했습니다.



<그림4. Roboflow 자료화면>

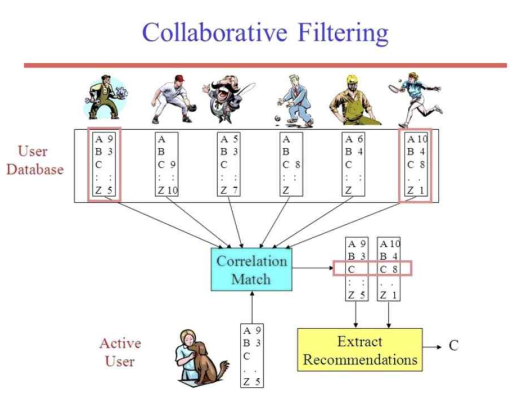
3. 화장품 추천 모델1 - 웹 크롤링 후 정제한 화장품 데이터셋을 CSV파일로 저장한 후, **콘텐츠 기반 추천 알고리즘으로 제품을 추천**합니다. 사용자의 진단받은 기본 피부 타입과 심층 피부 타입이 화장품의 타입과 일치하면 score값을 높게 주고 이 score값이 높은 제품의 이름과 사진을 보여줍니다. 이 추천방법은 사용자가 처음 사용한다고 가정했을 때, 사용자의 선호도를 충분히 파악하지 못한다고 판단하여 아이템 특성 기반으로 추천을 하기 위해 이 방법을 사용했습니다. **이 방법은 특성을 추출하기 어렵다는 단점이 있으나, 피부타입이라는 명확한 기준이 있기 때문에 오히려 우리 서비스에 적합할 것이라 판단**했습니다.

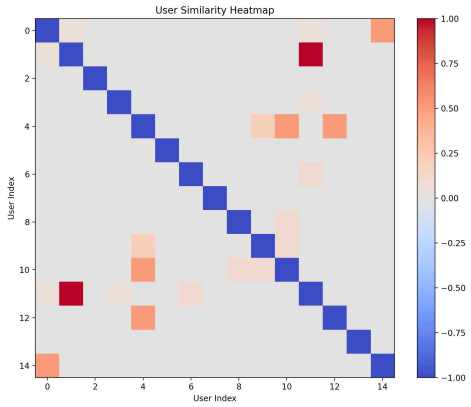



```
def content_based_recommendation(product_df, user_profile, top_n=10):
    product_df['score'] = product_df.apply(
        lambda x: (x['skin_type_result'] == user_profile['skinType']) +
                  sum(trouble in user_profile['skinTrouble'] for trouble in x['skin_trouble_result'].split(',')), axis=1)

    # 스코어에 따라 정렬 후 무작위로 섞기
    recommended_products = product_df.sort_values(by='score', ascending=False)
    recommended_products = recommended_products.head(2 * top_n)  # 상위 2 * top_n개 아이템 중 무작위 선택

    # 만약 제품이 2 * top_n개 미만이라면, 제품의 전체 목록을 사용
    if len(recommended_products) < 2 * top_n:
        recommended_products = product_df

    recommended_products = recommended_products.sample(frac=1).head(top_n)  # 무작위로 섞은 후 상위 top_n개 선택

    recommend_items = recommended_products[['title', 'imgurl']].to_dict(orient='records')

    return recommend_items
```

<그림5. 화장품추천모델1 자료화면>

4. 화장품 추천 모델2 - 웹 크롤링 후 정제한 화장품 데이터셋과 사용자의 평점을 포함한 구매이력 데이터셋을 CSV파일로 저장한 후, **협업필터링 기반 추천 알고리즘**으로 제품을 추천합니다. 사용자가 제품을 충분히 구매했고, 이에 대한 평점이 있다는 가정 하에 이 방법을 사용합니다. **KNN모델을 사용해서 유사 사용자 군집 안에서 추천을 하는 방법을 사용**합니다. 유사 사용자를 찾기 위해서 유사도를 이용하는데, 피부타입이 같은 사용자들을 유사 사용자로 지정하기 위한 유사도 함수를 만들었습니다. **각 군집 안에서 평점이 높은 제품을 유사 사용자에게 추천**해줍니다.



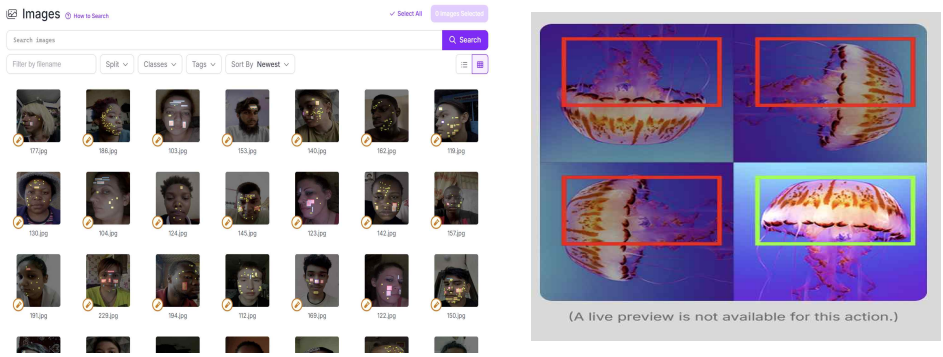


<그림6. 화장품추천모델2 자료화면>

5. 영양제 추천 모델 - 웹 크롤링 후 정제한 영양제 데이터셋을 CSV파일로 저장한 후, **콘텐츠 기반 추천 알고리즘으로 제품을 추천**합니다. 진단받은 기본 피부 타입과 심층 피부 타입을 가진 사용자가 사용하면 좋은 구성성분이 영양제의 구성성분과 일치하면 일치하는 제품들 중에서 추천해줍니다.

6. 사용자 피부 타입 그래프 이미지 출력 모델 – flutter의 fl_chart package에 있는 기능을 이용해서 **pentagon 형식의 이미지를 출력함으로써 사용자의 편의성을 증가**시켜주었다.

7. 피부 데이터 전처리 - rescale, horizontal_flip, rotation, contrast stretching 등의 기술을 사용하여 피부 데이터를 전처리했습니다. 이를 통해 모델 학습에 적합한 데이터셋을 구성했습니다.



<그림7. 피부 데이터 전처리 자료화면>

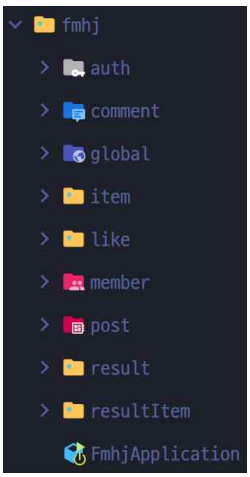
8. 화장품, 영양제 데이터 전처리 - 웹 크롤링, HTML 파싱, selenium 라이브러리 등을 사용하여 화장품 및 영양제 데이터를 수집하고 정제했습니다. 수집한 데이터를 CSV 파일로 저장하여 모델 학습 및 추천 시스템에 활용했습니다.

<그림8. 화장품,영양제 데이터 전처리 자료화면>

<Back-End 설계>

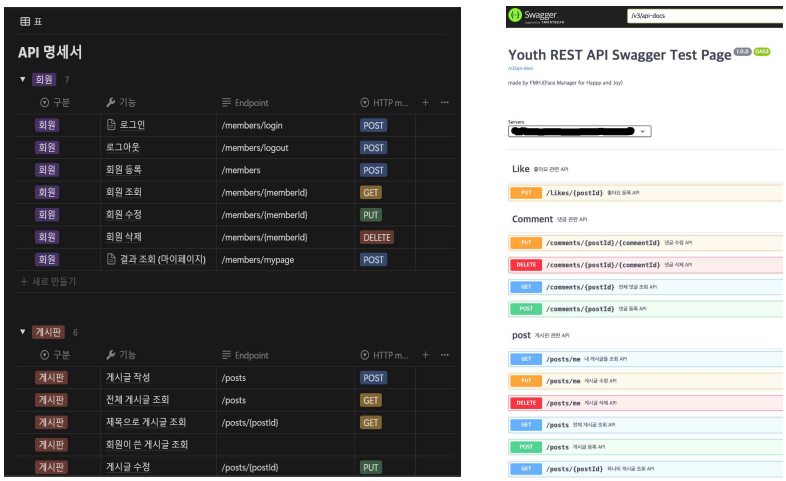
1. ERD 설계 – 처음에 생각한 것은 크게 회원, 게시판, 피부 진단 결과였습니다. 3개의 큰 엔티티를 바탕으로 관계형 DB를 설계했습니다. 회원은 인증 및 인가, 게시판은 좋아요, 댓글을 포함하여 사용자 친화적으로 설계, 피부 진단과 관련하여 통신 방법 등을 고민하였습니다. 하였습니다. 초기 설계 시 **제품 엔티티와 진단 결과 엔티티가 다대다 관계라서 고민**이 많았습니다. 이는 성능 저하와 일반적으로 사용하지 않는 방식이기 때문입니다. 따라서 **브릿지 테이블을 만들어서 정규화를 해주었고 데이터 중복을 제거**했습니다. 또한, 저희가 사용한 JPA 기술에서도 조회 부분에서 복잡한 쿼리문도 단순하게 해소하였습니다.

2. 도메인형 디렉토리 구조 – 디렉토리 구조를 도메인형으로 채택하여 협업 시에 **다른 팀원과의 코드 충돌을 최소화**하였습니다. 이는 유지보수와 확장에도 큰 도움을 주었고 깃허브 merge 과정에서도 수월하게 협업할 수 있도록 하였습니다. 또한, 토큰과 관련된 **인증과 인가는 auth 폴더에서 관리**하였습니다. **공통 응답 코드와 에러 처리, CORS 접근, 보안 설정은 global 폴더에서 관리**하였습니다.



<그림1. 도메인형 디렉토리 구조>

3. API 명세서 작성 및 문서화 – **Notion 작성 툴을 이용하여 REST API를 사용하여 도메인 별로 자원을 할당해 작성**하였습니다. 처음에는 개인적인 API를 접근할 때 {memberId}를 사용하였으나, JWT 토큰을 도입하여 인증된 사용자를 유지하였기 때문에 며칠 뒤 me를 사용한 API로 수정하였습니다. 또한, **Swagger API 문서화 툴을 이용**하여 자동으로 해당 API에 대해 json 형식으로 작성해주어서 편리하게 기능 테스트를 하였습니다.

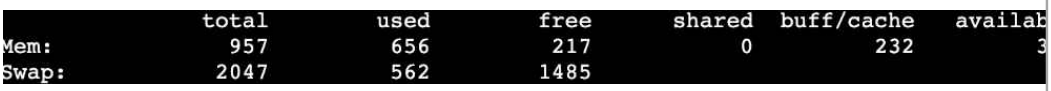


<그림2. API 명세서 화면>

4. AWS 서버 구축 및 배포 - AWS EC2 인스턴스를 설정하여 안정적인 운영을 하였습니다, **RSA 키페어를 따로 설정하고 탄력적 IP를 설정하여 보안과 연결성을 강화**했습니다. 또한, **트래픽이 증가될 때 인스턴스가 자동으로 확장되도록 구성**했습니다. 또한, AWS CLoudWatch를 이용하여 에러 로그를 모니터링 하기 쉽게 하였습니다. 초기에는 EC2 인스턴스에 SSH 키를 사용하여 접속하고, 프로젝트를 클론 후 빌드하여 스프링 부트 서버를 실행했습니다. 코드 변경 시 불편함을 느껴, 나중에는 **GitHub Actions를 사용하여 CI/CD 파이프라인을 구축**했습니다. 이를 통해 코드가 변경될 때마다 자동으로 테스트하고, 배포하는 환경을 만들었습니다.

<그림3. CI/CD 자료화면>

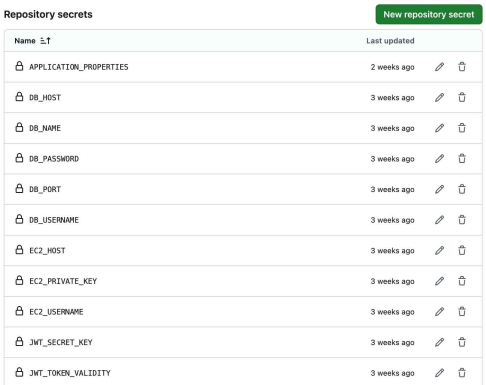
5. 서버 메모리 확장 – Docker에 설정된 개발 환경들을 받아오는 과정에서만 메모리를 많이 사용하는 것을 확인하였습니다. 이때만을 위해 인스턴스를 확장하는 것은 비용적으로 비효율적이라고 생각하였습니다. 따라서 안쓰는 디스크 용량을 메모리로 할당하여 메모리를 확장하였습니다. 즉, **swap memory를 할당하여 기존의 메모리에서 3배까지 늘려 사용**했습니다. 이렇게 기존의 인스턴스를 이용하고 메모리를 늘려, 큰 비용 절감을 할 수 있었습니다.



	total	used	free	shared	buff/cache	availab
Mem:	957	656	217	0	232	3
Swap:	2047	562	1485			

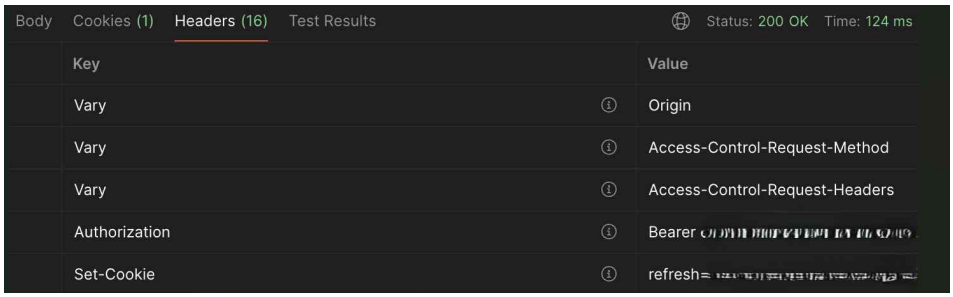
<그림4. swap 메모리 할당 자료>

6. 환경변수 관리 - **민감한 정보를 안전하게 관리**하기 위해 여러 환경변수를 따로 **Github Secrets로 관리**하였습니다. 해당 환경변수들이 사용되는 yml 파일에서는 이름 환경변수 이름으로 삽입되도록 설정하였습니다.



<그림5. 환경변수 관리 자료화면>

7. JWT 기반 인증 시스템 - 로그인을 통해 스프링 서버에서 Access Token과 Refresh Token을 발급받습니다. Secure Storage에 Access Token을 저장하여 토큰 탈취에 대한 위험을 고려하였습니다. 앞으로의 **API 요청 시 Access Token을 헤더의 Authorization에 삽입하여 인증된 상태를 유지**하였습니다. **토큰이 만료되면 Cookie Manager를 통해 관리된 Refresh Token을 통해 만료된 Access Token을 갱신**하는 데 사용됩니다.



<그림6. JWT 토큰 자료화면>

8. 사용자 간의 소통 창구 개설 - 게시판, 댓글, 좋아요 기능을 개발하여 **사용자 간의 소통을 통해 관련 주제에 대해 자유롭게 의견을 나눌 수 있도록 만들었습니다.** 또한, 사용자는 이를 통해 다양한 상호작용을 하며 사용자 경험을 풍부하게 하고 유대감을 향상시킵니다. 동일한 회원이 하나의 게시글에 여러 개의 좋아요를 등록할 수 없도록 좋아요 엔티티를 따록 분리해두었습니다.

9. 서버 간의 통신 경로 고안 – AI 모델로 사용자의 피부 타입을 진단할 수 있는 flask 서버, 실질적 DB 접근을 하는 springboot 서버, 클라이언트인 flutter 간 통신에 대한 고민이 많았습니다.

첫 번째 기획) flask 서버에서 flutter로 이미지를 보내는 방식을 선택하였습니다. 이는 파일 크기 문제와 보내는 정보가 많아, 중간에 짤리는 오류가 발생했습니다.

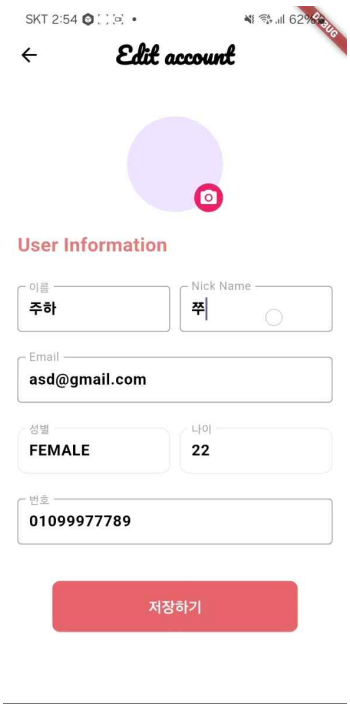
두 번째 기획) flask 서버에서 DB 접근이 가능하도록 기획했습니다. 하지만, 기존의 springboot에서 JPA를 통해 엔티티 매핑, pk 자동관리, 조인 관계 등 flask에서 접근하기에는 무결성을 위반할 요소들이 다분하여 폐기하였습니다.

세 번째 기획) flask에서 진단하고 pandas로 생성하던 이미지를 flutter의 라이브러리를 이용하여 이미지를 생성하는 방식을 기획했습니다. flask에서는 5가지 피부 타입에 대해 확률을 생성하고 flutter에서는 확률에 대해 오각형 이미지 생성을 합니다. springboot에서는 생성된 이미지를 DB에 저장하였습니다.

결과) DB에는 하나의 서버만 접근하여 무결성을 유지하고, **flask의 AI 모델을 사용하여 피부 진단에 대한 신뢰성**을 높였습니다. 또한, **flutter 자체의 라이브러리로 이미지를 생성하여 응답 속도를 증가**시켰습니다.

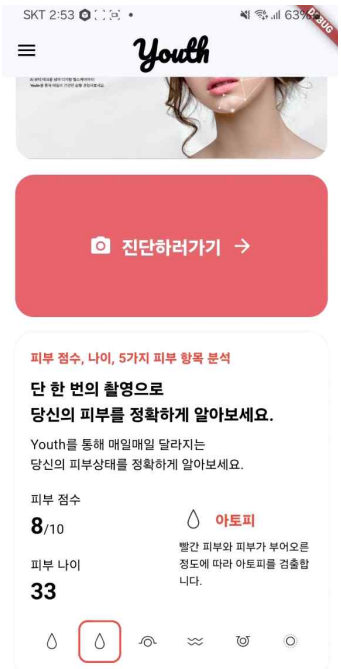
IV 결과

구현 화면

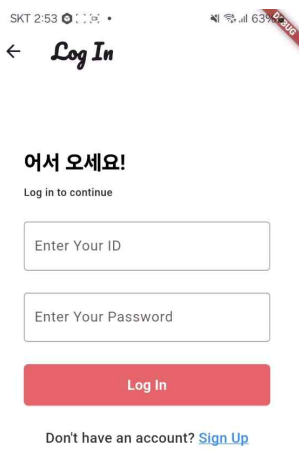

Edit account
User Information
이름
주하
Nick Name
쭈
Email
asd@gmail.com
성별
FEMALE
나이
22
번호
01099977789
저장하기


Youth
쭈
글 작성하기
뷰티퀸수진
16/8/2024
4 likes
피부 관리 비법
피부박사철수
16/8/2024
5 likes
여드름 치료법
화장왕민호
16/8/2024
1 likes
건성 피부 케어
뷰티지니지영
16/8/2024
2 likes
화장품 추천


Youth
주하's skin type is DRY
Youth가 추천하는 화장품들
메디힐 마데카 소사이드 흔...
에르쯔틴 리제너레이티브 ...
Youth가 추천하는 영양제들
비타민C 1000
혈행건강 프로메가 알티지 ...
베이직 뉴트리언트 투퍼데이


Youth
진단하러가기
피부 점수, 나이, 5가지 피부 항목 분석
단 한 번의 촬영으로
당신의 피부를 정확하게 알아보세요.
Youth를 통해 매일매일 달라지는
당신의 피부상태를 정확하게 알아보세요.
피부 점수
8/10
피부 나이
33
아토피
빨간 피부와 피부가 부어오른 정도에 따라 아토피를 검출합니다.


Youth
이름: 주하
성별: FEMALE
나이: 22
Email: asd@gmail.com
피부 타입: DRY
번호: 01099977789
수정
진단 결과
DRY
WRINKLES
COMBINATION
ACNE
OILY
your skintype is BaseType.DRY
나에게 알맞은 제품 추천받기


Log In
어서 오세요!
Log in to continue
Enter Your ID
Enter Your Password
Log In
Don't have an account? Sign Up

성과	1. AI 기술을 사용하여 피부 타입, 지성, 건성, 복합성, 주름, 여드름에 대한 **피부 상태를 분석하는 시스템을 구축**하였습니다. 2. 분석된 피부 상태를 바탕으로 사용자에게 가장 **적합한 스킨케어 제품과 영양제를 추천하는 시스템을 개발**하였습니다. 3. **사용자 간의 피드백 제공 기능을 구현**하여, 이를 통해 추천 알고리즘을 지속적으로 개선할 수 있는 구조를 마련하였습니다.
기대효과	1. 간단한 촬영만으로 피부 상태를 분석하고 즉각적인 제품 추천을 통해 **접근성이 향상되고 비용 절감 효과를 기대할 수 있습니다.** 2. 커뮤니티 피드백을 통해 지속적으로 **추천 알고리즘을 개선하여, 더 정확하고 신뢰성 있는 추천을 제공**할 수 있습니다. 3. 커뮤니티 기능을 통해 사용자들이 서로의 경험을 공유하고 피드백을 주고받으면서, **커뮤니티가 활성화되고 플랫폼의 유용성이 증대**됩니다.
자체 평가	**1. AI 모델 성능 평가** 피부타입을 측정하는 AI 모델의 성능이 mAP50가 약 0.4정도로 측정되었습니다. 이는 모델이 피부 타입을 정확하게 분류하는 데 있어 **제한적인 성능**을 보여주고 있으며, 이는 사용자에게 제공되는 분석 결과의 신뢰도에 영향을 미칠 수 있습니다. 이러한 성능 저하는 **주로 AI 모델을 학습시키기 위한 정확하고 다양한 피부 데이터셋을 확보하는 데 어려움이 있었기 때문입니다.** 피부 타입에 따라 정확한 진단을 내리기 위해서는 다양한 연령대, 인종, 피부 상태를 아우르는 **고품질 데이터셋이 필요하지만, 이를 구축하는 데 제약이 있어 모델 성능이 제한적일 수밖에 없었습니다.** **2. 사용자 인터페이스 평가** 사용자 피드백을 바탕으로 분석한 결과, **과거 진단 메뉴의 인터페이스가 직관적이지 않다는 문제가 확인되었습니다.** 현재 각 기록의 리스트만을 보여주는 방식은 사용자에게 혼란을 줄 수 있으며, 진단 기록을 효율적으로 관리하기 어려운 것으로 나타났습니다. 이에 따라, **각 기록을 리스트 형태로 제공하기보다는 과거 진단 페이지 자체를 보여주는 방식으로 변경하는 것이 더 나은 사용자 경험을 제공할 수 있을 것으로 판단됩니다.** **3. 개인화된 추천 시스템 평가** 비록 AI 모델의 성능이 제한적이지만, 피부 타입에 기반한 개인화된 추천은 비교적 효과적으로 이루어지고 있습니다. **사용자는 자신의 피부 상태에 맞춘 화장품과 영양제 추천을 통해 만족스러운 경험을 얻고 있으며, 이는 어플리케이션의 중요한 강점으로 작용하고 있습니다.** 이를 통해 사용자는 보다 개인화된 피부 관리 솔루션을 제공받아 만족도가 높아지고 있습니다.

	4. 개선 가능성 및 발전 방향 어플리케이션의 기능을 강화하기 위한 개선 가능성도 적극 검토되었습니다. 구체적으로 다음과 같은 기능들이 추가적으로 고려될 수 있습니다: 1) **사용자 주변의 피부과를 지도를 통해 시각적으로 제공**하여 사용자가 쉽게 접근할 수 있도록 합니다. 2) **챗봇을 통해 사용자가 원하는 피부과에 예약을 간편하게 진행**할 수 있도록 지원합니다. 3) **사용자의 피부타입에 맞는 화장법 및 피부 관리 방법을 담은 유튜브 영상을 추천**하여 보다 종합적인 피부 관리 솔루션을 제공합니다. 4) **redis를 통한 인메모리 접근으로 빠른 데이터 접근으로 응답 시간 최소화** 5) **Nginx를 통한 동시 접근 안전성 보완 및 서버 부하를 분산** 이러한 기능들은 사용자 경험을 더욱 풍부하게 만들며, 어플리케이션의 가치를 높이는 데 기여할 것입니다. **5. 경쟁력 분석** 기존 어플리케이션의 단점을 보완하기 위해 AI 성능 및 사용자 인터페이스 개선이 지속적으로 이루어지고 있습니다. **특히, 데이터셋의 질을 개선하고, 직관적인 UI/UX를 통해 사용자 만족도를 높이고자 하는 노력이 어플리케이션의 경쟁력을 강화할 것으로 보입니다.** 이러한 개선 노력은 시장 내에서 어플리케이션이 독보적인 위치를 차지하는 데 중요한 역할을 할 것입니다. 종합적으로, 본 어플리케이션은 피부타입 측정의 정확성, 사용자 경험의 편리성, 개인화된 추천 기능의 유용성을 강화함으로써, 사용자에게 실질적인 가치를 제공하고 있습니다. 향후 계획된 개선 사항들을 반영하여 지속적으로 발전해 나간다면, 더욱 높은 성과를 달성할 수 있을 것으로 기대됩니다.
Github	https://github.com/FeMeHaeJo