SK네트웍스 Family AI과정 7기

모델배포 개발된 LLM 연동 웹 애플리케이션

□ 개요

- 산출물 단계 : 모델배포
- 평가 산출물 : 개발된 LLM 연동 웹 애플리케이션
- 제출 일자 : 2025.04.22
- 깃허브 경로 : SKNETWORKS-FAMILY-AICAMP/SKN07-FINAL-2Team
- 작성 팀원 : 김성근

개요	● 목표 웹캠 또는 사진으로 전송된 사용자의 얼굴을 분석하여 얼굴형을 판별하고 이에 어울리는 안경을 추천하고, 제출된 얼굴 사진에 선택된 안경을 합성한 이미지를 생성한다. ● 주요기능 ○ 웹캠을 이용하여 얼굴 사진을 찍고 이를 이용하여 얼굴형을 분석한다. ○ 사진을 전송하여 얼굴 사진을 찍고 이를 이용하여 얼굴형을 분석한다. ○ 사진을 전송하기전 동의 여부를 묻고 사진 전송을 진행한다. ○ 웹캠을 이용하여 얼굴 사진을 찍을 경우 웹캠이 꺼져있는 경우 웹캠을 키는 명령을 전송한다. ○ 분석된 얼굴형을 기반으로 DB에서 안경 목록을 검색하여 제공한다. ○ 안경 목록 중 특정 안경을 선택 시 해당 안경의 정보와 상세 정보를 확인 할 수 있는 링크를 제공한다. ○ 분석된 얼굴형과 전송된 얼굴 사진을 이용하여 합성 사진을 제작한다. ○ 사진을 찍고 파일을 전송하는 등의 업무 진행을 사용자와 대화를 통해 진행하며, 인사말 과 DB에 기반한 안경에 대한 이야기 이외에 대화는 거부 하도록 RAG 기반 LLM을 이용한 채팅을 구현한다. ● 기술스텍 ○ Frond-End: ■ 개발 언어: HTML, CSS, Javascript ■ 프레임워크: Jquery, ■ 얼굴 인식 : MediaPipe ○ Back-End: ■ 개발언어: Python ■ 프레임워크: FastApi, Tortoise ORM ■ ASGI Server: Uvicorn ■ DB : MariaDB, ChromaDB ○ 얼굴 추론 : ■ 프레임워크 : Tensorflow ■ 랜드마크 처리 : MTCNN ■ 이미지 전처리 : OpenCV ■ 추론 모델 : VGG16 ○ 이미지 합성: ■ 랜드마크 처리 : DLIB

<table>
<tr>
<td></td>
<td>
■ 이미지 합성 : OpenCV
○ 배포:
■ 개발 (Docker) :
● WEB:
○ Rocky Linux
○ NginX
● WAS:
○ Rocky Linux
○ Uvicorn
● DB:
○ Rocky Linux
○ MariaDB
■ 서비스 (AWS) :
● WEB:
○ EC2
○ Ubuntu
○ NginX
● WAS:
○ EC2
○ Ubuntu
○ Uvicorn
● DB:
○ RDS
○ MariaDB
</td>
</tr>
<tr>
<td>설치 및 설정</td>
<td>
● 개발:
○ WEB:
_____
1. Rocky Linux Container 만들기 및 실행
> docker run -d --privileged --name web -p 10022:22 -p 53:53 -p80:80 -p 443:443 rockylinux/rockylinux init

2. package upgrade
> dnf upgrade -y

3. 원격 접속을 위한 tool 설치
> dnf install -y net-tools openssh-server passwd procps

4. ssh 의 root 원격 접속 허용을 위한 sshd_config 파일의 내용 수정
> vi /etc/ssh/sshd_config
</td>
</tr>
</table>

	---------- PermitRootLogin prohibit-password 를 다음과 같이 변경. PermitRootLogin yes PasswordAuthentication no 를 다음과 같이 변경. PasswordAuthentication yes ---------- 5. root password 설정 > passwd 6. ssh 6.1. 실행 > systemctl start sshd 6.2. ssh enable 설정 > systemctl enable sshd 7. 계정 7.1. 생성 > useradd firefly 7.2. 암호 설정 > passwd firefly 8. sudo 권한 생성 8.1. sudo 설치 > dnf install sudo 8.2. config 파일 수정, > visudo --- ## Read drop-in files from /etc/sudoers.d (the # here does not mean a comment) #includedir /etc/sudoers.d firefly ALL=(ALL) ALL --- 9. Nginx 설치 9.1. package install > dnf install nginx 9.2. 작업 디렉토리 생성

> mkdir -p /home/facefit/html
> mkdir -p /home/facefit/images
> chown -R firefly:firefly /home/facefit
> chown -R firefly:firefly /home/images

9.3. service 실행
> systemctl start nginx

9.4 enable
> systemctl enable nginx

9.5. ssl 인증서 설치
9.5.1. key 파일 복사
/etc/pki/nginx/private/facefit.key
9.5.2. crt 파일 복사
/etc/pki/nginx/facefit.crt
9.5.3. nginx.conf 파일 수정
> vi /etc/nginx/nginx.conf

```
sendfile            on;
tcp_nopush          on;
tcp_nodelay         on;
keepalive_timeout   65;
types_hash_max_size 2048;

client_max_body_size 5M;


    include             /etc/nginx/mime.types;
    default_type        application/octet-stream;


server {
        listen       443 ssl http2 default_server;
        listen       [::]:443 ssl http2 default_server;
        server_name  facefit.halowing.com;
        root         /home/facefit/html;

        ssl_certificate "/etc/pki/nginx/facefit.crt";
```

```
        ssl_certificate_key "/etc/pki/nginx/private/facefit.key";
        ssl_session_cache shared:SSL:1m;
        ssl_session_timeout  10m;
        ssl_ciphers PROFILE=SYSTEM;
        ssl_prefer_server_ciphers on;

        # Load configuration files for the default server block.
        include /etc/nginx/default.d/*.conf;

         location / {
                return 301 https://$host$request_uri;
        }



        error_page 404 /404.html;
                location = /40x.html {
        }

        error_page 500 502 503 504 /50x.html;
                location = /50x.html {
        }
}
# Settings for a TLS enabled server.
#
server {
        listen       443 ssl http2 default_server;
        listen       [::]:443 ssl http2 default_server;
        server_name  facefit.halowing.com;
        root         /home/facefit/html;

        ssl_certificate "/etc/pki/nginx/facefit.crt";
        ssl_certificate_key "/etc/pki/nginx/private/facefit.key";
        ssl_session_cache shared:SSL:1m;
        ssl_session_timeout  10m;
        ssl_ciphers PROFILE=SYSTEM;
        ssl_prefer_server_ciphers on;

        # Load configuration files for the default server block.
```

```
include /etc/nginx/default.d/*.conf;

location / {
}

location /images/ {
        alias /home/facefit/images/;
        autoindex on; # 선택 사항: 디렉토리 목록 표시
}

location /welcome/ {
        proxy_pass http://172.17.0.3:8000/welcome/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
}

location /chat/ {
        proxy_pass http://172.17.0.3:8000/chat/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
}
 location /file/ {
        proxy_pass http://172.17.0.3:8000/file/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
}

location /webcam/ {
        proxy_pass http://172.17.0.3:8000/webcam/;
        proxy_http_version 1.1;
```

```
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location /glasses/ {
        proxy_pass http://172.17.0.3:8000/glasses/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location /docs {
        proxy_pass http://172.17.0.3:8000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location /openapi.json {
        proxy_pass http://172.17.0.3:8000/openapi.json;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    error_page 404 /404.html;
        location = /40x.html {
    }

    error_page 500 502 503 504 /50x.html;
        location = /50x.html {
```

```
        }
}
---
9.5. nginx 재시작
> systemctl restart nginx

10. 정적 데이터 복사
10.1. 안경 이미지
/home/facefit/images 디렉토리에 glasses.tar 파일 복사
> cd /home/facefit/images
> tar xvf glasses.tar

10.2. 얼굴형 이미지
/home/facefit/images 디렉토리에 face_shape.tar 파일 복사
> cd /home/facefit/images
> tar xvf face_shape.tar

10.3. Application
/home/facefit/images 디렉토리에 html.tar 파일 복사
> cd /home/facefit/
> tar xvf html.tar

11. SystemD 등록
11.1. 서비스 파일 생성
vi /etc/systemd/system/facefit.service
---
[Unit]
Description=FaceFit
After=network.target

[Service]
User=firefly
Group=firefly
WorkingDirectory=/home/facefit/app
ExecStart=/home/firefly/.local/bin/uvicorn --host 0.0.0.0 --port 8000 --env-file
/home/facefit/.env --app-dir /home/facefit/app  main:app
Type=simple
Restart=on-failure
RestartSec=1
```

Environment=PATH=/home/firefly/.local/bin:/usr/local/bin:$PATH
Environment=LD_LIBRARY_PATH=/usr/local/lib:/home/firefly/.local/lib/python3.12/site-packages:$LD_LIBRARY_PATH

TimeoutSec=30

[Install]
WantedBy=multi-user.target

11.2. 서비스 등록
> sudo systemctl daemon-reload
> sudo systemctl enable facefit

11.3. 서비스 실행
> sudo systemctl status facefit
> sudo systemctl start facefit

- WAS:

---

1. Rocky Linux Container 만들기 및 실행
> docker run -d --privileged --name api -p 50022:22 -p 58000:8000 rockylinux/rockylinux init

2. package upgrade
> dnf upgrade -y

3. 원격 접속을 위한 tool 설치
> dnf install -y net-tools openssh-server passwd procps

4. ssh 의 root 원격 접속 허용을 위한 sshd_config 파일의 내용 수정
> vi /etc/ssh/sshd_config

PermitRootLogin prohibit-password 를 다음과 같이 변경. PermitRootLogin yes
PasswordAuthentication no 를 다음과 같이 변경. PasswordAuthentication yes
---

5. root password 설정
> passwd

6. ssh
6.1. 실행
> systemctl start sshd

6.2. ssh enable 설정
> systemctl enable sshd

7. 계정
7.1. 생성
useradd firefly
7.2. 암호 설정
> passwd firefly

8. sudo 권한 생성
8.1. sudo 설치
> dnf install sudo

8.2. config 파일 수정,
> visudo
—
Read drop-in files from /etc/sudoers.d (the # here does not mean a comment)
#includedir /etc/sudoers.d
> firefly ALL=(ALL) ALL

9. Application 설치
9.1. 디렉토리 생성
> mkdir /home/facefit/
9.2. app.tar 파일을 /home/facefit/ 폴더에 upload

9.3. app.tar 파일 압축 해제
> tar xvf app.tar

9.4 env 파일 생성
> vi ./app/.env
—
OPENAI_API_KEY='your key'
PYTHONPATH=$PYTHONPATH:/home/facefit/app

db_url=mysql://{id}:{passwd}@{db_server}:3306/{db_schema}
—
9.5. CMAKE 설치
> sudo yum install python3.12-devel gcc gcc-c++ cmake

9.6. SQLite 3.35 설치
9.6.1. 소스 다운로드
> sudo yum install wget
> wget https://sqlite.org/2025/sqlite-autoconf-3490100.tar.gz
> tar xzf sqlite-autoconf-3490100.tar.gz

9.6.2. 소스 컴파일
> cd sqlite-autoconf-3490100
> ./configure --enable-fts5
> sudo make
> sudo make install

9.6.3. LD_LIBRARY_PATH 설정:
> vi ~/.bashrc_profile
—
export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
—

9.7. OPENCV Library 설치
> sudo yum install mesa-libGL-devel

10. pip package 설치
/home/facefit 디렉토리에 requirements.txt 파일을 복사한다.
> cd /home/facefit/
> pip3 install -r requirements.txt

11. dilb package 설치
dlib 를 user 권한으로 설치하면 컴파일 된 라이브러리 설치시 권한이 없어서 에러 난다.
> sudo pip3 install dlib

12. 라이브러리 설치
12.1. 안면 인식 라이브러리 설치
/home/facefit/ml/model/ 디렉토리에 shape_predictor_68_face_landmarks.dat 파일을 복사한다.

	12.2. 안면인식 학습 모델 설치 /home/facefit/ml/model/ 디렉토리에 vgg16.keras 파일을 복사한다. 13. ChromaDB 초기화 /home/firfly/.bash_profile 파일에 PYTHONPATH 설정 --- export PYTHONPATH=$PYTHONPATH:/home/facefit/app --- 13.1. 명령어 Document 입력 > cd /home/facefit/ > python ./app/admin/init_command.py 13.2. 안경 Document 입력 > cd /home/facefit/ > python ./app/admin/init_glasses.py _____ ● DB: _____ 1. Docker Image 설치 및 실행 > docker pull mariadb/latest > docker run --restart=always --name mariadb -p 3306:3306 ₩ --mount source=mdb_volume,target=/var/lib/mysql -e MARIADB_ROOT_PASSWORD=dreamiscomming -e MARIADB_DATABASE=facefit -d mariadb:latest 2. 계정 설정 > use mysel; > create user firefly@'%' identified by '{password}; > grant all privileges on facefit.* to firefly@'%'; > flush privileges; 3. Table 생성 및 Data 초기화 > source DDL-create_tables.sql > source DATA-INIR.sql
기본 사용법	● 사설 root CA 인증서 설치 ○ 인증서 다운로드 : https://facefit.halowing.com/static/js/ca.crt

	○ Ca.crt 파일을 더블클릭하여 설치 진행 ● 서비스 접속 크롬이나 마이크로소프트 엣지 브라우저를 이용하여 다음 URL 접속 https://facefit.halowing.com/ ● 웹캠으로 캡쳐 user: 사진 찍어줘 ai : 웹캠을 이용한 사진 찍기를 선택하셨습니다. sys : 카메라로 사진을 찍는다. ai : 사진이 준비 되었습니다. 사진을 전송하시겠습니까? 1.예, 2.아니오 user: 네 ai : 얼굴을 분석 하였습니다. ... ● 사진 올리기 user: 사진을 업로드 할게 sys: 사진을 업로드할 UI 표시 user: 사진을 선택한다. ai : 사진이 준비 되었습니다. 사진을 전송하시겠습니까? 1.예, 2.아니오 user: 네 ai : 얼굴을 분석 하였습니다. ... ● 안경 목록 요청 user : 안경 목록 보여줘 ai : 요청하신 안경 목록은 다음과 같습니다 sys: 안경 목록을 좌측 사진 옆에 전시 ● 안경 합성 사진 요청 user: 안경 목록에서 안경을 선택 user: 안경 피팅해줘 ai : 요청하신 안경을 사진에 합성하였습니다.
확장 및 커스터마이징	● 확장: ○ Back-End에서 안경 사진 합성을 완료 후, 실시간 안경 착용 모습을 볼 수 있으면 좋겠다는 의견이 있어서 Front-End에서 이를 구현함.
결론	● 성과 ○ 사용자가 안경점을 가기 전에 컴퓨터를 이용하여 간편하게 안경 쓴 모습을 미리 확인 할 수 있어서 선택의 시간을 줄여줌. ● 프로젝트 기여 ○ 김서진: 팀장. 프로젝트 관리, 시나리오 개발 ○ 김성근 : 시스템 설계, Back-End program, 서버 설치

	○ 김태희 : 안경 데이터 수집 및 이미지 처리. 얼굴형 분석 AI 모델 학습 및 테스트 ○ 유수현 : GUI 디자인, Front-End program 및 서비스 시나리오 개발 ○ 정승연 : 얼굴형 분석 AI 모델 개발, 학습 및 테스트 ● 향후 발전 방향 ○ 실제 상거래 사이트를 섭외하여 판매 가능한 서비스를 개발 할 수 있을 것으로 생각됩니다.. ● 한계 및 개선방안 ○ 안경과 얼굴의 합성시 안경의 위치를 정확히 예측하기 어려움. 향후 안경 랜드마크 프로그램을 개발하면 안경 합성에 좀더 나은 성과가 있을 것으로 생강 됩니다. ○ 가용한 얼굴형 데이터의 한계로 아시안 핏에 대한 얼굻형 분석이 이루어 지지 않았고, 얼굴형을 단순히 5가지 형태로 분류하는데 그침. 향후 더 많은 학습 모델을 확보한다면 좀더 나은 안경 추천이 이루어 질 수 있을 것으로 보인다. ○ OpenAI API Agent 프로그램의 미숙으로 TOOL을 이용한 안경 상세 검색 기능을 구현하지 못함. 이 부분에 개선이 가능하다고 생각됩니다.