페이스 인식 기능을 활용한 감정분석 기능

[상세설계서]

2023.10.10

인하대학교 컴퓨터공학과

캡스톤 설계 - 005분반

지도 교수 : 신병석

팀명:**CODETRIO**

팀원: 전예준 12181676, 김성아 12181581,최선아 12191714

버전관리

버전	개정 내용	작성자	적용 날짜
1.0	최초 생성	김성아	10.09

목차

4. 프로젝트 관리

1. 서론

1.1. 목적

본 문서는 페이스 인식 기능을 활용한 감정분석 기능에 대한 상세 설계서이다.

인하대학교 컴퓨터 공학과 'CODETRIO' 팀의 프로젝트를 위한 것으로, 이를 바탕으로 시스템을 설계 및 구현한다.

1.2. 개요

본 연구에서는 FER 기술을 활용하여 얼굴 표정에서 감정이 어떻게 드러나는지에 대한 연구를 진행하고자 한다. 특히, 어떠한 상황에서 표정이 더 정확하게 감정을 전달하는지, 그리고 실제로 느끼는 감정과 얼굴 표정을 통해 인식된 감정 간의 일치도를 조사함으로써, 감정 인식 기술의 정확성과 유효성에 대한 통찰을 얻고자 한다.

본 문서의 구성은 다음과 같다. 1장에서는 프로젝트 목적과 전체 개발 환경에 대해 소개하고 2장에서는 시스템을 이루는 다양한 모듈에 대한 상세 설명과 모듈간 상호작용, 사용할 데이터셋에 대한 설명이 이루어진다. 3장에서는 시스템이 사용자에게 보여지는 인터페이스에 대한 시각적 자료를 이용하여 시스템의 기능 및 인터페이스의 연결 관계를 표현한다.

1.3. 개발 환경

- OS : macOS 13.4, window11

- 개발 도구 : Google Colab, VS code, IntelliJ

- 개발 언어: Python (Pytorch), React, Typescript, Java
- 데이터베이스: MySQL
- 개발 기간 : 3개월
- 개발 인원 : 3명

1.4. 주요 기능

1. 사진에서 얼굴 영역 크롭
2. 얼굴 감정 분석을 통해 해당 이미지의 나타난 감정 확인
3. 촬영한 얼굴을 통해 분석한 감정과 실제로 자신이 느낀 감정을 비교

1.5. 개발 범위

1. 데이터셋 파일 범위

이번 컴퓨터 공학 졸업 설계 프로젝트에서는 image 데이터셋의 종류를 jpeg파일 형식으로 한정한다.

2. 데이터셋 규모

데이터셋은 총 50만장의 이미지로, 약 3.8TB 분량이다. 그러나 용량의 한계로 데이터를 전부 사용하지 않고 감정별 최대 2만장의 이미지를 8:1:1비율로 나누어 학습, 검증, 테스트 데이터 셋으로 사용하고자 한다.

1.6. 참고 자료

- http://ki-it.com/_common/do.php?a=full&b=12&bidx=3223&aidx=35957
- https://seokii.tistory.com/112

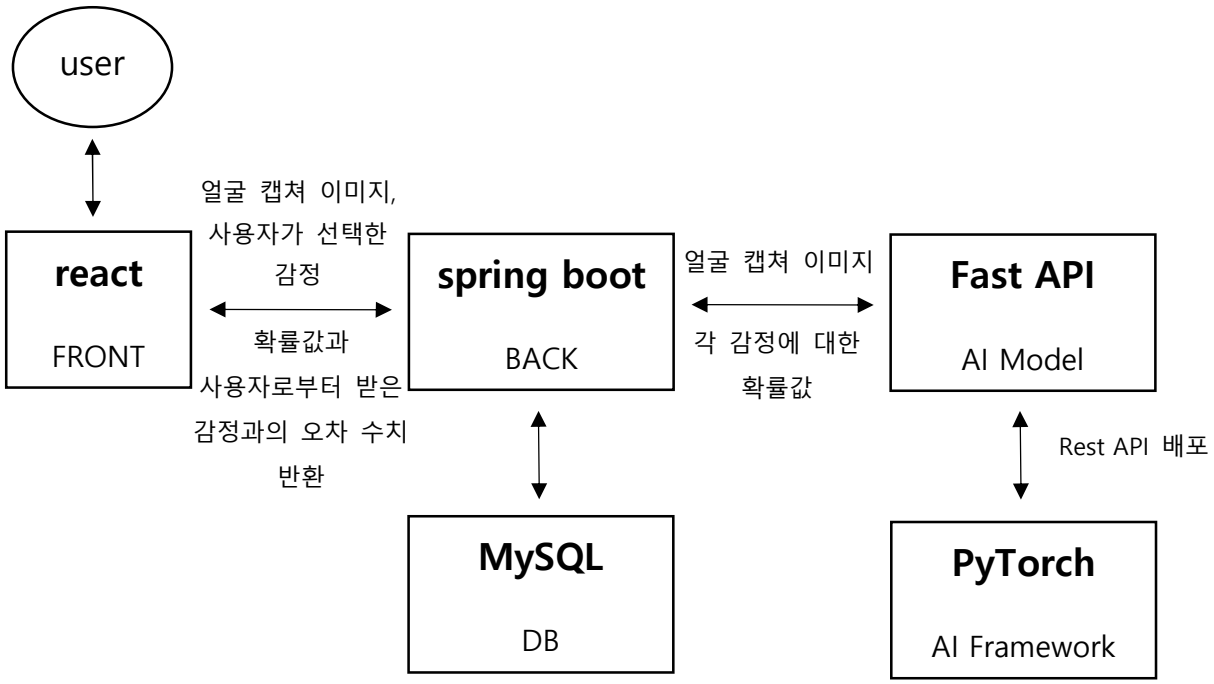
- https://arxiv.org/abs/1409.1556

- https://stickode.tistory.com/691

- https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL

- https://recharts.org/en-US/api/BarChart

2. System 설계



2.1. ai 설계

2.1.1. face detection model

library
yaml
glob
IPython.display
os

model	description

yolov5s	yolov5모델을 활용하여 사람만 인식하도록 학습한다.

yolov5 modify	**description**
data.yaml	데이터 정보를 담고있는 파일
data['nc']=1	클래스 수를 하나로 변경(nc=number of class)
data['names']=['person']	클래스 이름을 담고있는 배열에 person만 남긴다
train(이미지,라벨) 폴더	라벨이 사람인 train데이터셋만 남김
detect.py	모델 로드 및 데이터셋 결과출력 함수
largest_box_index= torch.argmax(torch.tensor([(box[2] - box[0]) * (box[3] - box[1]) for box in det]))	이미지마다 인식된 사람이 여러명 있을 수 있으므로 그 중 가장 크게 인식된 얼굴의 바운딩 박스 인덱스를 찾기 위해 사용
if idx != largest_box_index: continue	해당 인덱스가 가장 큰 바운딩 박스일 때만 저장하기 위해 사용

설계

1. yolov5모델을 클론한 후 사람만 인식하도록 클래스 수와 클래스 이름을 변경한다.

2. 모든 train데이터셋과 valid데이터셋에 대해 라벨이 'person'인 데이터셋만 남기고 나머지는 삭제한다.

3. yolov5의 train.py를 이용해 batch=16, epochs=5으로 하여 학습시킨다.

2.1.2. face emotion recognition model

2.1.2.1. face emotion recognition model info

library
torch

torch.nn
torchvision

model
resnet-50

resnet modify	**description**
resnet50.fc	fully connected layer 변경
nn.Dropout(0.5)	과적합을 방지하기 위해 사용
nn.Linear(resnet50.fc.in_features,1024)	fully connected 레이어 추가
nn.Linear(1024, num_classes)	감정 라벨 수와 동일하게 출력 클래스 수를 변경한다 (num_classes: 출력 클래스 수)

2.1.2.2. face emotion recognition model train

1) image crop

variable	**description**
label=['기쁨','당황','분노','불안','상처','슬픔','중립']	감정 라벨을 저장 하는 배열, 라벨 별로 데이터셋이 저장되어있기 때문에 파일 경로를 지정할 때 사용
folder_path	이미지 데이터셋을 저장하고 있는 폴더
crop_folder_path	크롭된 이미지 데이터셋을 저장하고 있는 폴더

설계

1. 감정 라벨을 label 배열에 저장한다.

2. 각 라벨별 데이터셋의 경로를 가져와 folder_path변수에 할당하고 yolov5.py의 detect.py를 이용하여 이미지를 크롭하고 저장한다.

2) face emotion recognition model train

variable	description
LABELS = {'기쁨':np.array([1,0,0,0,0,0,0]), '당황':np.array([0,1,0,0,0,0,0]), '분노':np.array([0,0,1,0,0,0,0]), '불안':np.array([0,0,0,1,0,0,0]), '상처':np.array([0,0,0,0,1,0,0]), '슬픔':np.array([0,0,0,0,0,1,0]), '중립':np.array([0,0,0,0,0,0,1])}	라벨을 numpy형 array로 만들기 위해 사용
dataset = labeled_data	이미지와 라벨을 튜플 형식으로 저장
batch_size	배치 사이즈 정의
dataloader = DataLoader(dataset, batch_size=batch_size, shuffle=True)	dataset을 미니배치 단위로 만들어 저장
num_class = 7	감정 클래스 수 정의
model = CustomResNet50	사용할 모델 정의
criterion = nn.CrossEntropyLoss	교차 엔트로피 손실함수 사용
optimizer = Adam(model.parameters(), lr=0.1)	Adam 옵티마이저 함수 사용
num_epochs	epoch 수 정의
device = torch.deviece('cuda')	gpu 사용
total_loss	epoch마다 총 오차를 저장
output	모델 output값 저장
average_loss	total_loss/dataloader 수

설계

1. num_epoch만큼 CustomResNet50모델 학습한다.

2. optimizeer.step()함수를 통해 가중치를 업데이트 한다.

3. 평균 오차를 구한다.

3) face emotion recognition model valid

variable	description
correct_predictions	예측값과 라벨이 일치하는 데이터셋 수 저장
total_samples	검증 데이터셋 수 저장
val_accuracy	correct_predictions/total_samples

설계

1. 데이터 셋을 모델에 입력 후 출력 값을 클래스별 확률로 받는다.

2. 확률이 가장 큰 값을 predicted 변수에 저장한다.

3. predicted와 라벨이 일치하는 데이터셋 수를 correct_predictions변수에 저장한다.

4. correct_predictions변수를 valid데이터셋 수로 나눈 후 val_accuracy 변수에 저장 후 출력한다.

2.1.2.3. ResNet-50 Model 개요

ResNet은 "Residual Networks"의 줄임말로, 딥 러닝 모델의 깊이를 증가시키면서도 그라디언트 소실 문제를 해결하기 위해 도입된 혁신적인 아키텍처이다. ResNet-50은 ResNet 시리즈 중 하나로, 50개의 층을 가지고 있으며 이미지 인식 및 다양한 컴퓨터 비전 태스크에 사용된다.

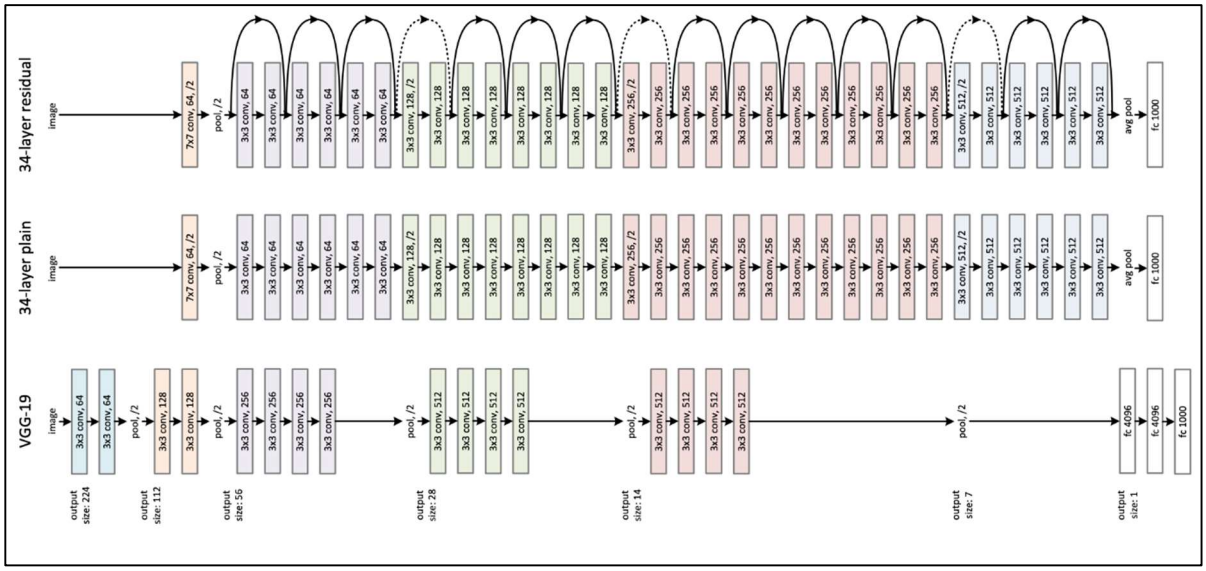
주요특징

- 깊은 네트워크 구조: ResNet-50은 깊은 네트워크로 구성되어 있어서 복잡한 패턴과 특징을 학습할 수 있다.

- 잔여 연결 (Residual Connection): ResNet의 핵심 아이디어는 잔여 연결이다. 각 레이어의 입력은 해당 레이어에서 학습된 변화만큼만 추가된다. 이러한 잔여 연결은

그라디언트 소실 문제를 줄여주고 더 깊은 네트워크를 효과적으로 학습할 수 있게 해준다.

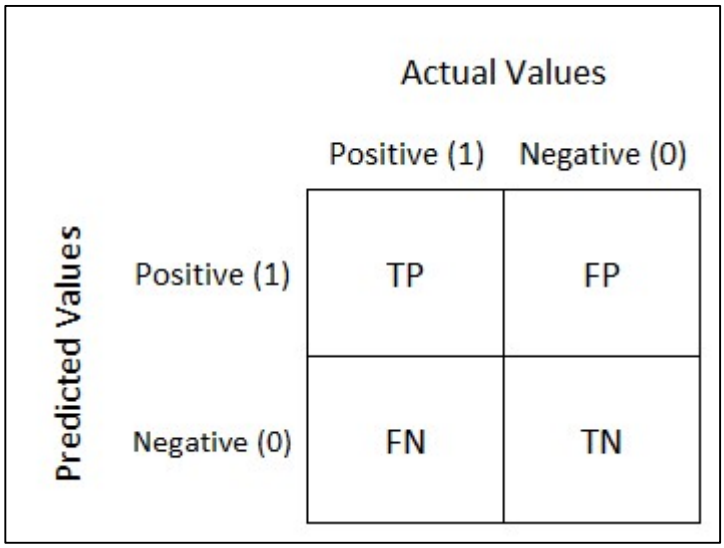
- pretrained weight: ResNet-50 모델은 대규모 이미지 데이터셋에서 미리 학습된 가중치를 가지고 있어서, 전이 학습(transfer learning)을 통해 다양한 작업에 활용될 수 있다.



https://huggingface.co/microsoft/resnet-50

2.1.2.4. 평가지표: confusion matrix - accuracy

confusion matrix: 모델의 성능을 평가할 때 사용되는 지표로 예측값이 실제 관측값을 얼마나 정확히 예측했는지 보여주는 행렬이다.



https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

accuracy: 모델이 입력된 데이터에 대해 얼마나 정확하게 예측하는지를 나타낸다.

정확도 = 예측값결과와 실제값이 동일한 건수 / 전체 데이터수

= (TP+TN) / (TP+TN+FN+FP)

2.1.3. REST API

2.1.3.1. detect_crop_face 함수

library
torch
pathlib
PIL

variable	**description**
detection_model	yolov5s 모델
result	detection_model 출력값 저장
faces	예측 결과에서 얼굴 정보 추출
cropped_faces	이미지에서 얼굴 영역만 크롭후 저장

detect_crop_face 설계

1. yolo model을 사용하여 주어진 이미지에서 사람의 얼굴을 탐지하고 결과를 result에 저장한다.

2. result에 저장된 바운딩 박스 좌표를 정수로 변환하고 이미지에서 해당 좌표 영역만 잘라내어 cropped_faces에 저장한다.

3. cropped_faces를 반환한다.

2.1.3.2. classify_face 함수

library
torch
torch.nn
torchvision

variable	description
fer_model	resnet50 모델
preprocess	이미지 전처리를 위한 변환 정의
output	fer_model 출력값 저장

classify_face 함수 설계

1. 이미지 resize, 텐서 변환 등 이미지 전처리 과정을 preprocess에 저장한다.

2. 입력 이미지에 대해 모델을 통과시켜 예측값을 계산한 후 각 클래스에 대한 확률값을 output에 저장한다.

3. output을 반환한다.

2.1.3.3. detect_classify 함수

library
flask
io
PIL

variable	description
image_file	클라이언트로부터 받은 이미지파일 저장
image_byte	이미지를 바이트 형식으로 저장
image	Image.open을 사용하여 jpg 형식으로 저장
faces	detect_crop_face 모델의 출력값 저장
predictions	classify_face 모델의 출력값 저장

detect_classify 함수 설계

1. 클라이언트로부터 전송된 이미지 파일을 받아 image_file에 저장한다.

2. 이미지 파일에서 바이트 데이터를 읽어와 image_bytes에 저장한다.

3. PIL을 사용하여 바이트 데이터를 이미지로 연다.

4. detect_crop_face 함수를 호출하여 이미지 데이터에서 얼굴 이미지를 추출한다.

5. classify_face 함수를 호출하여 얼굴을 분류하고 예측값을 얻는다.

6. 결과를 JSON형식으로 반환한다.

2.2. Dataset 소개

한국인 감정인식을 위한 복합영상

본 데이터셋은 한국인의 '안면이미지' 를 통해 '감정인식' AI 학습용 모델 구축의 목적으로 구성되며 특정 장소/배경, 특정 감정이 나타난 '1인 셀프카메라(Selfie)' 50만장으로 구축되어 있다. AI 학습 모델의 정확도 증대를 위해, 특정 감정을 잘 표현할 수 있는 '전문인' 을 별도 모집하여 25만장을 수집, 자연스러운 일반인의

표정을 묘사할 수 있는 '일반인' 을 따로 모집하여 25만장을 수집하였으며, 이는 데이터의 포괄성과 품질에 대한 신뢰성을 확보할 수 있도록 구성되어 있다고 할 수 있다. 7가지로 감정을 분류하였는데 감정체계를 분류하는 기준은 Susan David 의 감정분류체계에 따랐으며, 현재 세계적인 감정 이미지를 활용한 학습용 데이터 구축 흐름에 발맞추어 '중립(무표정)' 감정을 별도 추가하였다.

Angry	Sad	Anxious	Hurt	Embarrassed	Happy
Grumpy	Disappointed	Afraid	Jealous	Isolated	Thankful
Frustrated	Mournful	Stressed	Betrayed	Self-conscious	Trusting
Annoyed	Regretful	Vulnerable	Isolated	Lonely	Comfortable
Defensive	Depressed	Confused	Shocked	Inferior	Content
Spiteful	Paralyzed	Bewildered	Deprived	Guilty	Excited
Impatient	Pessimistic	Skeptical	Victimized	Ashamed	Relaxed
Disgusted	Tearful	Worried	Aggrieved	Repugnant	Relieved
Offended	Dismayed	Cautious	Tormented	Pathetic	Elated
Irritated	Disillusioned	Nervous	Abandoned	Confused	Confident

<출처: AI Hub 한국인 감정인식을 위한 복합 영상 데이터 설명서>

표 | 감정 분류체계 7종

No.	대감정	세부감정
1	분노	툴툴대는, 좌절한, 짜증내는, 방어적인, 악의적인, 안달하는, 구역질 나는, 노여워하는, 성가신
2	슬픔	실망한, 비통한, 후회되는, 우울한, 마비된, 염세적인, 눈물나는, 낙패한, 환멸을 느끼는
3	불안	두려운, 스트레스 받는, 취약한, 헷갈리는, 당혹스러운, 회의적인, 걱정스러운, 조심스러운, 신경쓰이는
4	상처	질투하는, 배신당한, 격리된, 충격 받은, 궁핍한, 희생된, 억울한, 괴로워하는, 버려진
5	당황	격리된, 시선 의식하는, 외로운, 열등한, 죄책감의, 부끄러운, 혐오스러운, 한심한, 헷갈리는
6	기쁨	감사하는, 믿는, 편안한, 만족한, 흥분한, 느긋한, 안도하는, 신이 난, 자신하는
7	중립	무표정, 감정이 0인 상태

<출처: AI Hub 한국인 감정인식을 위한 복합 영상 데이터 설명서>

카테고리(중)	이미지 개수(개)	분포(%)
기쁨	70,735	14.47%
당황	70,457	14.41%
분노	68,835	14.08%
불안	69,965	14.31%
상처	70,103	14.34%
슬픔	70,508	14.42%
중립	68,173	13.94%
평균	69,825	14.28

<출처: AI Hub 한국인 감정인식을 위한 복합 영상 데이터 설명서>

데이터 셋 예시

jpg, jpeg 형태 이미지 약 50만 장

(1) 인물의 감정 상태 포함

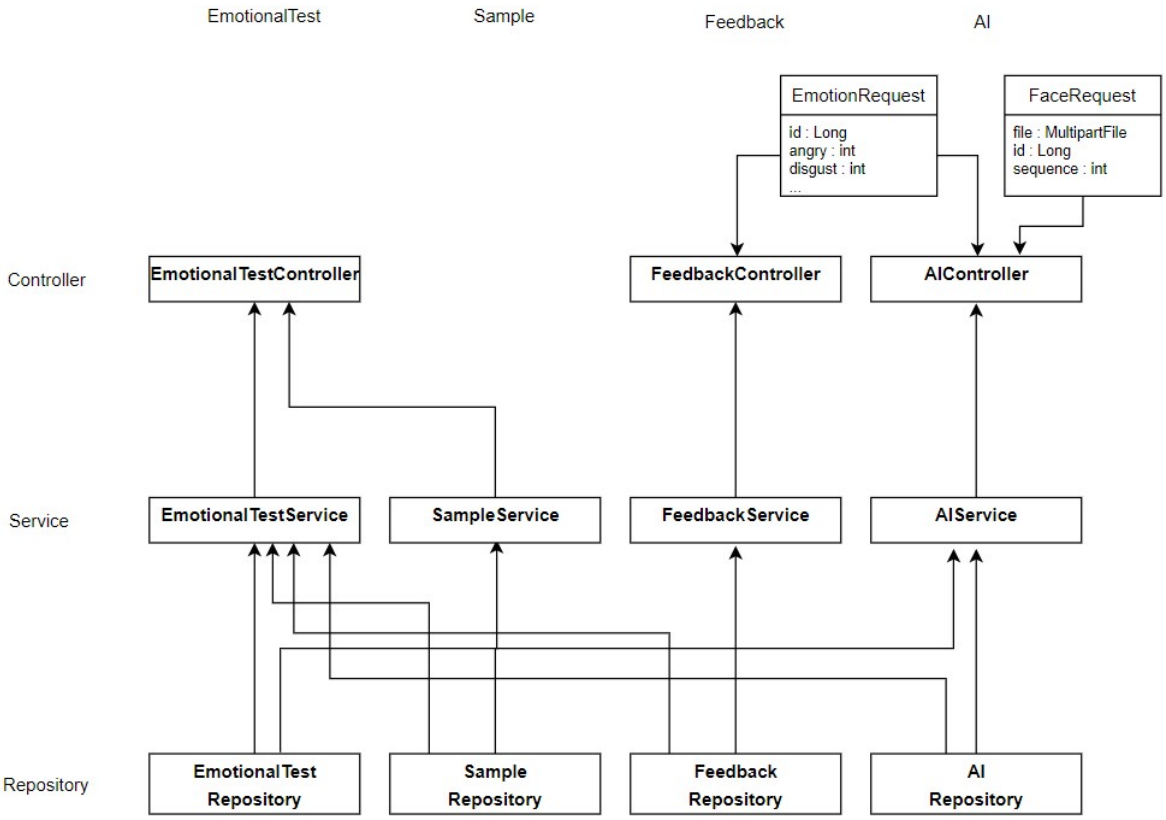
(2) 다양한 장소/배경 포함

2.3. 백엔드 설계

2.3.1. Module 설계

2.3.1.1. 전체적인 Module 구성도

- Controller-Service-Repository로 구성된 3-layered architecture model을 채용



2.3.1.2. 세부 Module 설계

1) EmotionTest Package

전반적인 테스트 과정을 진행하는 작업을 수행

EmotionTestController

프론트단에서 받은 전반적인 테스트 관련 요청을 처리	
emotionTestService	객체를 생성자를 통해 주입
sampleService	
emotionTestStart()	감정 분석 정보를 관리하기 위한 새로운 테스트를 등록함
sendTest()	테스트 데이터 전송 요청을 받아 샘플을 보내는 작업을 수행
sendResult()	테스트 결과를 종합해서 프론트단으로 보냄

EmotionTestService 테스트 진행 관련 핵심 비즈니스 로직을 처리	
emotionTestRepository	객체를 생성자를 통해 주입
sampleRepository	
feedbackRepository	
aiRepository	
generateTest()	테스트에 사용할 샘플을 무작위로 선정해서 전송
assembleResult()	테스트 결과를 종합해서 JSON 형식으로 보냄
calculateError()	분석 결과와 사용자 피드백 간의 정확도를 계산함 편차^2의 합을 이용하여 계산

EmotionTestRepository 테스트 정보를 담는 DB에 접근	
registerTest()	새로운 테스트 정보를 DB에 추가 후 부여한 테스트 번호를 반환
findTest(id)	테스트 번호를 이용해 전반적인 테스트 정보를 반환
addSample()	해당 테스트를 저장한 DB에 사용한 샘플을 등록
addPicture()	해당 테스트를 저장한 DB에 감정분석 대표 사진 uri를 등록

2) AI Package

테스트 과정에서 AI 측과 통신하는 데이터를 관리하는 작업을 수행

AIController 표정 분석 관련 요청을 처리	
aiService	객체를 생성자를 통해 주입
receivePhoto()	웹에서 받은 사진을 저장하고 AI 측으로 전송함
receiveResult()	감정 분석 수치를 받아서 데이터를 처리함

AIService 표정 분석 관련 핵심 비즈니스 로직을 처리	
aiRepository	객체를 생성자를 통해 주입
emotionTestRepository	
savePhoto()	웹캠으로 촬영한 BASE64로 인코딩 된 5장의 사진 파일을 서버 내부 저장소에 저장
postPhoto()	촬영한 5장의 사진을 그대로 분석에 사용할 AI 서버로 전송
calculateAIResult()	촬영한 5장의 사진에 대해 각 감정 값의 평균을 계산하여 DB에 결과값 저장
selectRepresentivePhoto()	촬영한 5장의 사진에 대해 감정 값의 편차 제곱합의 수치가 가장 적은 사진의 uri를 DB에 추가

AIRepository 표정 분석 정보를 담는 DB에 접근	
saveAIResult()	AI 분석 결과를 DB에 저장
findAIResult()	테스트 번호(id)를 이용하여 감정 분석 결과를 반환

3) Sample Package

테스트에 사용할 샘플을 관리하는 작업을 수행

SampleService 감정 추출용 샘플에 대한 비즈니스 로직을 처리	
sampleRepository	객체를 생성자를 통해 주입
randomSample()	테스트에 사용할 샘플을 무작위로 결정함

SampleRepository 샘플 정보를 담는 DB에 접근	
FindSample()	샘플의 id값을 이용하여 해당 샘플의 데이터를 반환

4) Feedback Package

테스트 도중 받는 사용자의 피드백을 관리하는 작업을 수행

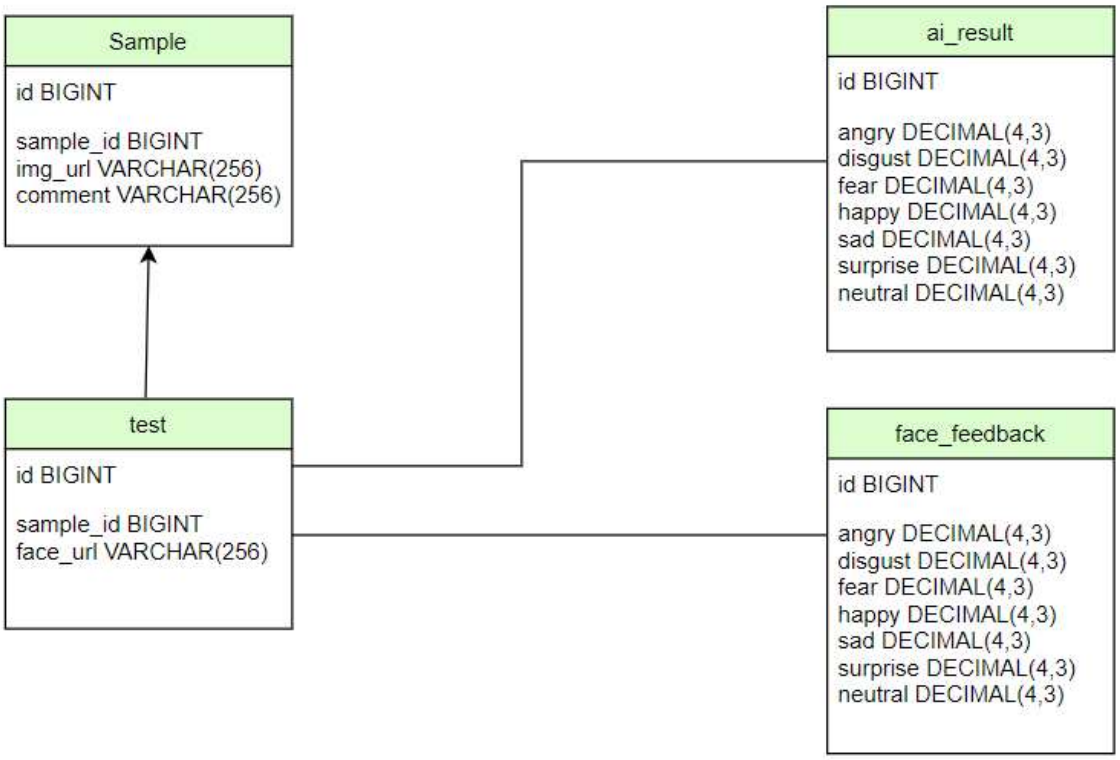
FeedbackController 사용자 피드백에 대한 요청을 처리	
feedbackService	객체를 생성자를 통해 주입
saveFeedback()	해당 테스트 id에 대한 사용자의 감정 피드백 값을 확률값으로 변환 후, DB에 저장

FeedbackService 사용자 피드백에 대한 비즈니스 로직을 처리	
feedbackRepository	객체를 생성자를 통해 주입
calculateFeedback()	사용자가 보낸 Neutral을 제외한 6가지의 감정 수치를 가지고 Neutral까지 계산하여 7개의 BigDecimal 수치로 변환 6개의 응답 수치(0~5) 중 최대값에 따라서 1~0을 Neutral에 부여하고 나머지는 수치에 비례해서 배분

FeedbackRepository 사용자 피드백 정보를 담는 DB에 접근	
saveFeedback()	계산된 피드백 수치를 DB에 저장
findFeedback()	테스트 번호를 이용하여 해당 감정 피드백을 반환

2.3.2 Database 설계

2.3.2.1 ER Diagram



2.3.2.2. DB Table

1) sample

사용자의 감정이 포함된 사진을 추출하기 위한 샘플의 데이터를 저장할 테이블

sample	
Primary key	id BIGINT
	img_uri VARCHAR(256) comment VARCHAR(256) group INT

id : Primary Key 역할, 샘플 식별 번호
img_uri : 샘플로 사용할 사진 파일의 경로
comment : 샘플과 같이 띄울 문구.
group : 샘플을 그룹별로 분류하기 위한 변수

2) test

테스트 데이터를 관리하기 위한 테이블

test	
Primary key	id BIGINT
Foreign Key	sample_id BIGINT face_uri VARCHAR(256)

id : Primary Key 역할, 테스트 번호
sample_id : 얼굴 사진 추출에 사용된 샘플 번호, 1)의 id를 참조함.
face_uri : 샘플을 활용하여 촬영한 얼굴 사진이 찍힌 파일의 정보

3) face_result

페이스 분석으로 나온 테스트 결과물을 저장할 테이블

face_result	
Primary key	id BIGINT
Foreign Key	angry DECIMAL(4,3) disgust DECIMAL(4,3) fear DECIMAL(4,3) happy DECIMAL(4,3) sad DECIMAL(4,3) surprise DECIMAL(4,3) neutral DECIMAL(4,3)

id : Primary Key 역할, 2)의 id 참조

angry, disgust, fear, happy, sad, surprise, neutral : AI로부터 받은 7가지 감정 수치,
0~1 사이의 확률 값으로 표현됨

4) face_feedback

사용자의 피드백을 통해 받은 감정값

face_feedback	
Primary key	id BIGINT
Foreign Key	angry DECIMAL(4,3) disgust DECIMAL(4,3) fear DECIMAL(4,3) happy DECIMAL(4,3) sad DECIMAL(4,3) surprise DECIMAL(4,3) neutral DECIMAL(4,3)

id : Primary Key 역할 , 2)의 id를 참조함angry, disgust, fear, happy, sad, surprise, neutral :
피드백을 통해 전달받은 7가지 감정 수치.
0~1 사이의 확률 값으로 표현됨

2.4. Frontend 설계

2.4.1. 디자인 패턴

view와 logic을 분리하기 위해 custom Hooks 디자인 패턴을 사용한다. custom

Hooks 디자인 패턴이란 로직을 hooks로 분리하여 관리하는 디자인패턴이다. 로직을 캡슐화하여 코드의 가독성과 유지보수성을 향상시키고, 여러 컴포넌트에서 동일한 로직을 재사용할 수 있게 해준다.

2.4.2. route

path	element
/	Homepage
/test/prepare	TestPreparePage
/test/progress	TestProgressPage
/test/result	TestResultPage

2.4.3 page

page	description
Homepage	프로그램 진입 페이지이다.
TestPreparePage	테스트를 준비하는 페이지이다.
TestProgressPage	테스트를 진행하는 페이지이다.
TestResultPage	테스트 결과를 나타내는 페이지이다.

2.4.4 component

2.4.4.1 공통 component

component	description
Header	프로그램 이름을 나타낸다.
Button	클릭 시 이벤트가 발생하는 버튼을 나타낸다.
Radio	요소들 중 하나만 선택 가능한 라디오를 나타낸다.
TestSample	샘플(사진과 문구)을 나타낸다.

2.4.4.2 페이지 별 component

page	
component	**description**

Homepage	
Header(공통)	프로그램 이름을 나타낸다.
BackgroundVideo	배경 비디오를 나타낸다.
ProgramIntroduction	프로그램 소개 문구를 나타낸다.
ProgramStartButton	프로그램 시작 버튼을 나타낸다.

TestPreparePage	
Header(공통)	프로그램 이름을 나타낸다.
Webcam	웹캠 화면을 나타낸다.
TestNotification	프로그램 이용 방법과 주의 사항 문구를 나타낸다.
TestStartButton	테스트 시작 버튼을 나타낸다.

TestProgressPage	
Header(공통)	프로그램 이름을 나타낸다.
TestSample(공통)	샘플(사진과 문구)을 나타낸다.
EmotionTF	감정 유무 질문과 버튼을 나타낸다.
EmotionChoice	감정 선택지를 나타낸다.
NextTestButton	다음 샘플에 대한 테스트 또는 테스트 결과 페이지로 넘어갈 수 있는 버튼을 나타낸다.

TestResultPage	
Header(공통)	프로그램 이름을 나타낸다.

TestSample(공통)	샘플(사진과 문구)을 나타낸다.
UserImage	샘플에 대한 사용자의 반응 사진을 나타낸다.
ErrorComment	ai 분석 결과와 사용자의 실제 감정의 오차 문구를 나타낸다.
Graph	ai 분석 결과와 사용자의 실제 감정 수치를 그래프로 나타낸다.
NextResultButton	다음 샘플에 대한 분석 결과를 띄우는 버튼을 나타낸다.
PrevResultButton	이전 샘플에 대한 분석 결과를 띄우는 버튼을 나타낸다.
DetailResult	분석 결과를 글로 나타낸다.

2.4.5 module

page - component

1) Hoepage

HomePage - Header		
handleToClickLogo	MouseEvent<HTMLElement>	로고를 클릭하면 Homepage로 이동한다.

HomePage - BackgroundVideo
-

HomePage - ProgramIntroduction
-

HomePage - ProgramStartButton		
handleToClickStartButton	MouseEvent<HTMLElement>	시작하기 버튼을 클릭하면 TestPrepare 페이지로 이동한다.

2) TestPreparePage

TestPreparePage - Header		
handleToClickLogo	MouseEvent<HTMLElement>	로고를 클릭하면 Homepage로 이동한다.

TestPreparePage - Webcam		
const videoRef = useRef(null)	<HTMLVideoElement>	video의 id를 나타내어 ref를 설정한 video 요소에 웹캠이 띄워질 수 있게 한다.
startWebcam()	-	웹캠을 시작하고 화면에

		띄운다.

TestPreparePage - TestNotification
-

TestPreparePage - TestStartButton		
handleToClickStartButton	MouseEvent<HTMLElement>	테스트 시작 버튼을 클릭하면 TestProgress 페이지로 이동한다.

3) TestProgressPage

TestProgressPage		
const [samples, setSamples] = useState()	{ sampleUrl: string, sampleComment: string }[]	서버에서 받아온 샘플 데이터들(샘플 이미지, 샘플 문구) 저장
const [currentStep, setCurrentStep] = useState(1)	number	몇 번 째 샘플에 대한 테스트인지 저장하여 step에 대한 sample이 뜰 수 있도록 한다.
cosnt [userImages, setUserImages] = useState([])	string[]	웹캠으로 캡처한 사용자의 얼굴 사진을 저장한다. 사진이 5장이 되면 캡처를 그만하고 감정 선택지를 띄워준다.
const [userEmotionTF,	string	사용자가 감정을 느꼈는지

setUserEmotionTF] = useState()		아닌지 '감정 있음', '감정 없음'으로 저장한다. '감정 있음' 이면 EmotionChoice 컴포넌트를 띄운다. '감정 없음' 이면 EmotionChoice 컴포넌트를 없애고 releaseSelectedEmotions()를 호출한다.

const [userEmotion, setUserEmotion] = useState()	{
	sad: number,
	angry: number,
	surprise: number,
	happy: number,
	fear: number,
	disgust: number
	}
		사용자가 선택한 감정을(아주 조금:1, 조금:2, 보통:3, 많이:4, 매우 많이:5)저장한다.
		
		초기값: 모두 0(감정 없음)
		
		감정 선택지 중 적어도 하나가 선택돼있거나 '감정 없음'이 선택돼있으면 NextTestButton 컴포넌트를 띄운다.

getSamples()	-	서버로부터 감정 유발 샘플들을 받아서 samples에 저장한다.
releaseSelectedEmotions()	-	useEmotion의 모든 감정들을 0으로 바꾼다.

TestProgressPage - Header		
handleToClickLogo	MouseEvent<HTMLElement>	로고를 클릭하면 Homepage로 이동한다.

TestProgressPage - TestSample
-

TestProgressPage - Webcam		
const videoRef = useRef(null)	<HTMLVideoElement>	video의 id를 나타내어 ref를 설정한 video 요소에 웹캠이 띄워질 수 있게 한다.
const canvasRef = useRef(null)	<HTMLCanvasElement>	canvas의 id를 나타내어 ref를 설정한 canvas 요소에 그림이 띄워질 수 있게 한다.
startWebcam()	-	웹캠을 시작하고 화면에 띄운다.
setTImer()	-	3초동안 5번 captureWebcam()을 호출한다.
captureWebcam()	-	웹캠 화면을 캡처하고 setUserImages()로 userImages에 저장한다.

TestProgressPage - EmotionTF		
handleToChangeEmotio	ChangeEvent<HTMLInputEleme	setUserEmotionTF()

nTF	nt>	를 통해 userEmotionTF값을 변경한다.

TestProgressPage - EmotionChoice		
handleToChangeEmotionDetails	ChangeEvent<HTMLInputElement>	setUserEmotion()를 통해 userEmotion값을 변경한다.

TestProgressPage - NextTestButton		
handleToClickNextTestButton	MouseEvent<HTMLElement>	setcurrentStep(step+1)를 통해 currentstep값을 변경하여 다음 sample에 대한 테스트로 넘어갈 수 있도록 한다. postUserInfo를 호출한다. 마지막 step이었으면 TestResult 페이지로 이동한다.
postUserInfo	-	5장의 사용자 사진이 담겨있는 useImages와 감정 7개에 대한 값이 담겨 있는 userEmotion을 서버에

		보낸다.

4) TestResultPage

TestResultPage		

const
[analysisResults,
setAnalysisResults] =
useState()
	{
	sampleUrl: string,
	sampleComment: string,
	userUrl: string,
	ai: {
	sad: number,
	angry: number,
	surprise: number,
	happy: number,
	fear: number,
	disgust: number,
	neutral: number,
	},
	user: {
	sad: number,
	angry: number,
	surprise: number,
	happy: number,
	fear: number,
	disgust: number,
	neutral: number,
	},
	error: number,
	}[]
		서버에서 받아온 sample 사진/문구, 사용자 사진 저장,
		
		서버에서 받아온, 7개의 감정에 대한 ai 분석 결과 수치를 저장,
		
		서버에서 받아온, 7개의 감정에 대한 사용자 선택 결과 수치를 저장,
		
		서버에서 받아온 ai 분석 결과의 오차값 저장

const [currentResult, setCurrentResult] = useState(1)	number	몇 번 째 샘플에 대한 분석 결과인지 저장하여 해당하는 분석 결과(그래프, 글)이 뜰 수 있도록 한다.
getAnalysisResult()	-	서버로 부터 모든 샘플에 대한 분석 결과를 받아서 setAnalysisResults()를 이용하여 analysisResult에 저장한다.

TestResultPage - Header		
handleToClickLogo	MouseEvent<HTMLElement>	로고를 클릭하면 Homepage로 이동한다.

TestResultPage - ResultTab		
handleToChangeResultTab	ChangeEvent<HTMLInputElement>	요소를 클릭하면 currentResult 값이 클릭한 요소에 대해 몇 번째 sample인지의 값으로 바뀐다.

TestProgressPage - TestSample
-

TestResultPage - UserImage
-

TestResultPage - ErrorComment
-

TestResultPage - Graph
-

TestResultPage - NextResultButton		
handleToClickNextResultButton	MouseEvent<HTMLElement>	currentResult의 값을 1 증가 시켜 다음 샘플에 대한 분석 결과가 화면에 뜰 수 있도록 한다.

TestResultPage - PrevResultButton		
handleToClickPrevtResultButton	MouseEvent<HTMLElement>	currentResult의 값을 1 감소 시켜 이전 샘플에 대한 분석 결과가 화면에 뜰 수 있도록 한다.

2.4.6 기능

1) 웹캠 화면 띄우기

- navigator Web API를 사용한다.

- navigator.mediaDevices.getUserMedia()의 인자로 비디오를 사용하겠다는 {video:true}객체를 넣는다.

- 함수 실행 결과를 stream 변수에 저장한다.

- 비디오 태그의 srcObject 속성으로 stream값을 지정해준다.

2) 웹캠 화면 캡처

- canvas를 생성하여 canvas의 너비와 높이를 video태그의 videoWidth 속성과 videoHeight 속성을 이용하여 웹캠의 너비와 높이와 같게 설정해준다.

- canvas.getContext('2d')를 호출하여 그리기 메서드와 속성을 가지는 컨텍스트 객체를 변수 context에 받는다.

- context의 메소드 drawImage(video, 0, 0, canvas.width, canvas.height)를 이용해 웹캠 화면을 canvas에 그린다.

- canvas 그림을 toDataURL('image/jepg') 메소드를 이용하여 jepg 형식의 이미지 파일로 변환한다.

3) 3초 안에 5번 캡처

- requestAnimationFrame을 사용하여 캡처 수가 5번 이전일 때 까지 반복한다.

- requestAnimationFrame을 사용하여 지연시간이 3000/5 값보다 크거나 같으면 캡처를 한다.

4) 그래프 나타내기

- recharts의 BarChart를 이용한다.

- data 객체에 각 감정에 대한 감정 이름, ai 분석 결과, 사용자 의견 값을 아래 예시와 같이 저장한다.

```
const data = [
  {
    "name": "기쁨",
    "ai 분석 결과": 0.5,
    "사용자 의견": 0.2
  }, ...
]
```

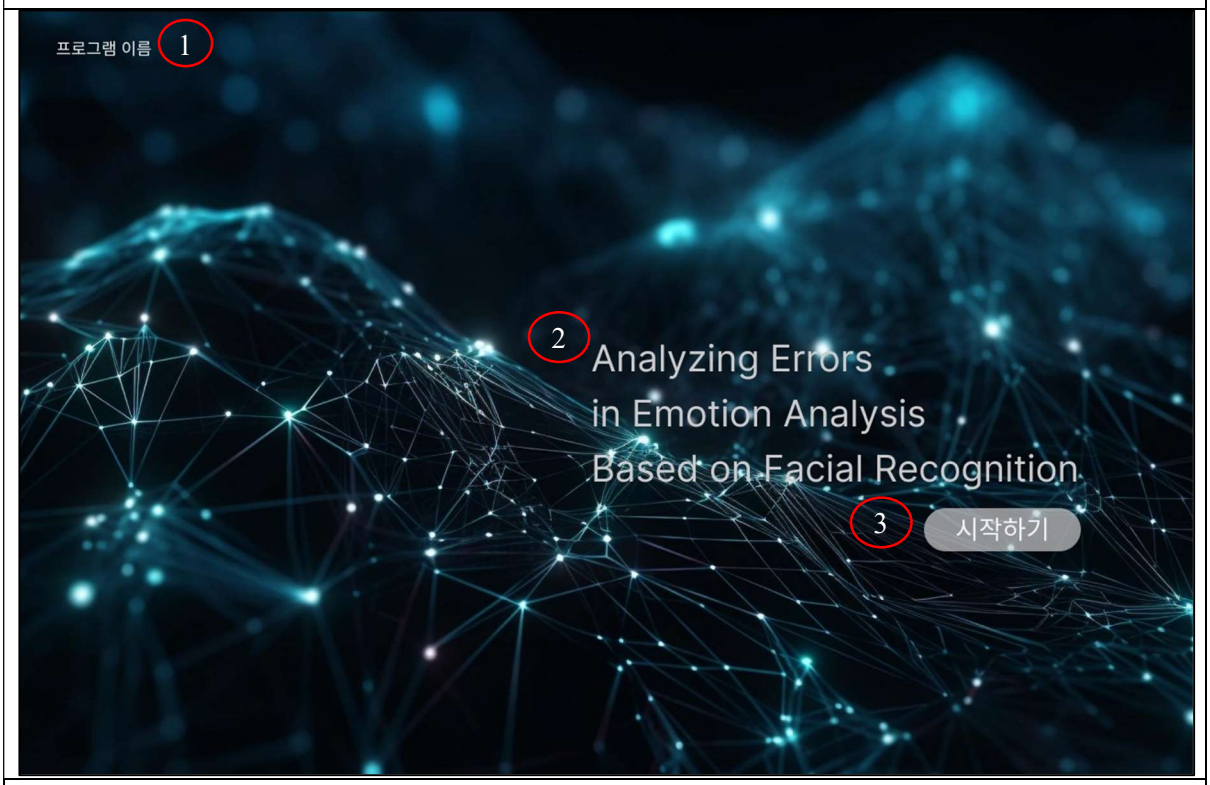
- data 객체를 BarChart의 data props로 넣는다.

```
<BarChart width={730} height={250} data={data}>
  <CartesianGrid strokeDasharray="3 3" />
  <XAxis dataKey="name" />
```

```
    <YAxis />
    <Tooltip />
    <Legend />
    <Bar dataKey="ai 분석 결과" fill="#8884d8" />
    <Bar dataKey="사용자 의견" fill="#82ca9d" />
</BarChart>
```

3. Interface 설계

Home



■ 웹 사이트를 들어왔을 때 처음 나타나는 화면이다.

■ 배경으로는 영상이 들어가고 배경 위 아래에 검정색 그림자를 준다.

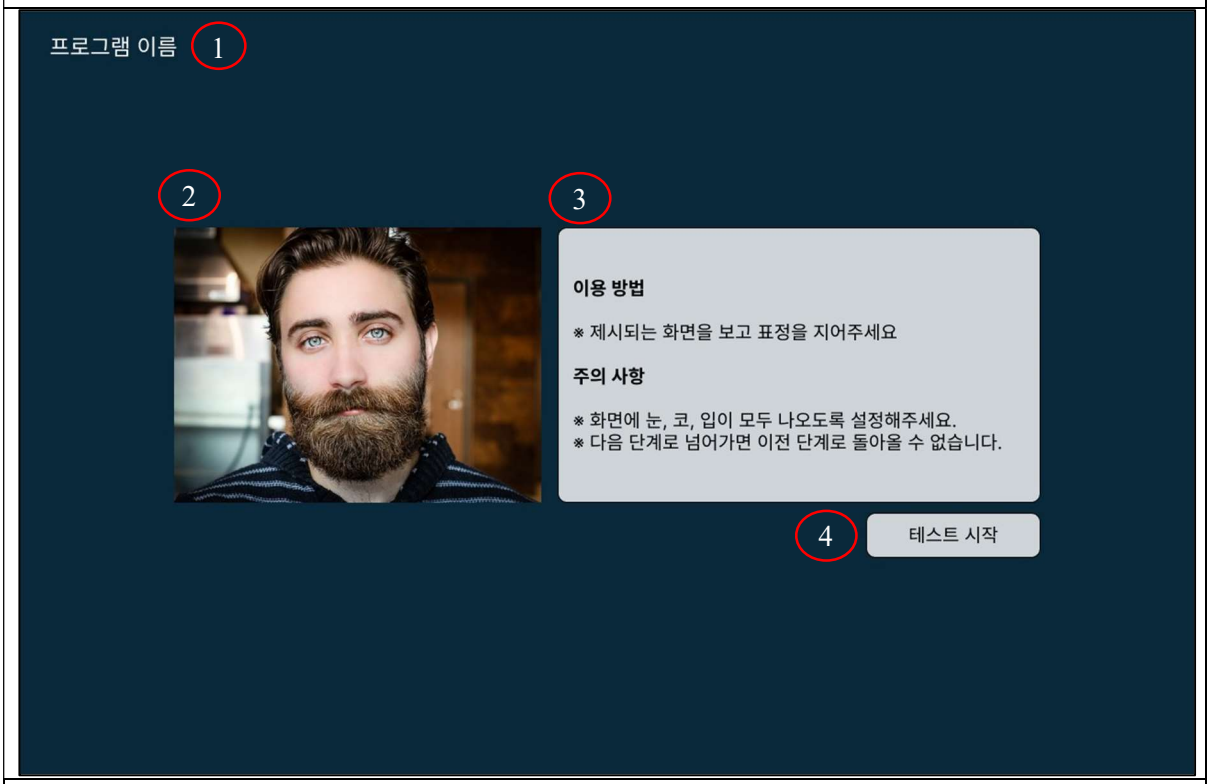
① 사이트 로고를 클릭하면 Home 화면이 새로고침 된다.

② 프로그램에 대한 간단한 설명을 포함한다.

③ 테스트 준비 페이지로 이동하는 '시작하기' 버튼이 있다.
'시작하기' 버튼을 클릭하면 '테스트 준비 페이지'로 이동한다.

이미지 출처: 작가vecstock 출처 Freepik

테스트 준비 페이지



① 사이트 로고를 클릭하면 Home 화면으로 이동한다.

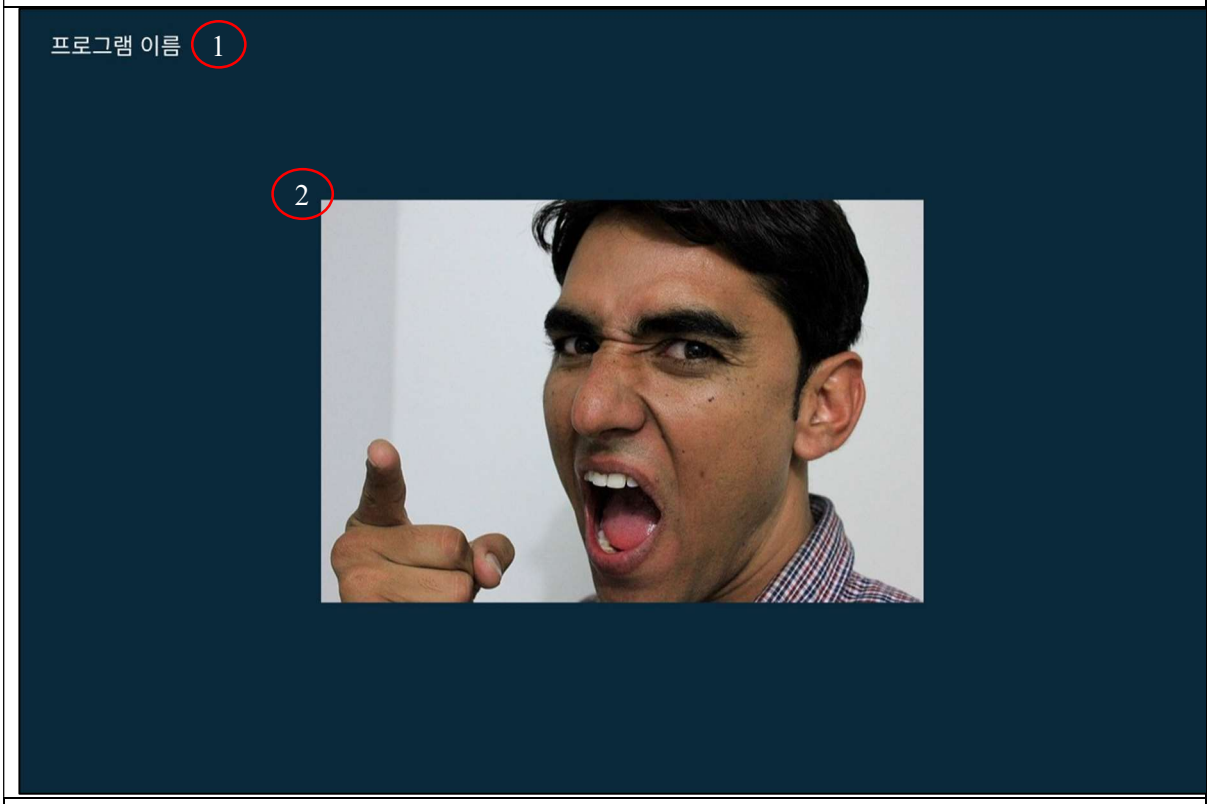
② 웹캠 화면이 들어간다.

③ 프로그램 이용 방법과 주의 사항이 들어간다.

④ '테스트 시작' 버튼을 클릭하면 '테스트 페이지'로 이동한다.

이미지 출처: pixabay

테스트 페이지 - 1



■ 사용자가 감정 유발 사진에 집중할 수 있도록 화면에 샘플만을 띄운다.

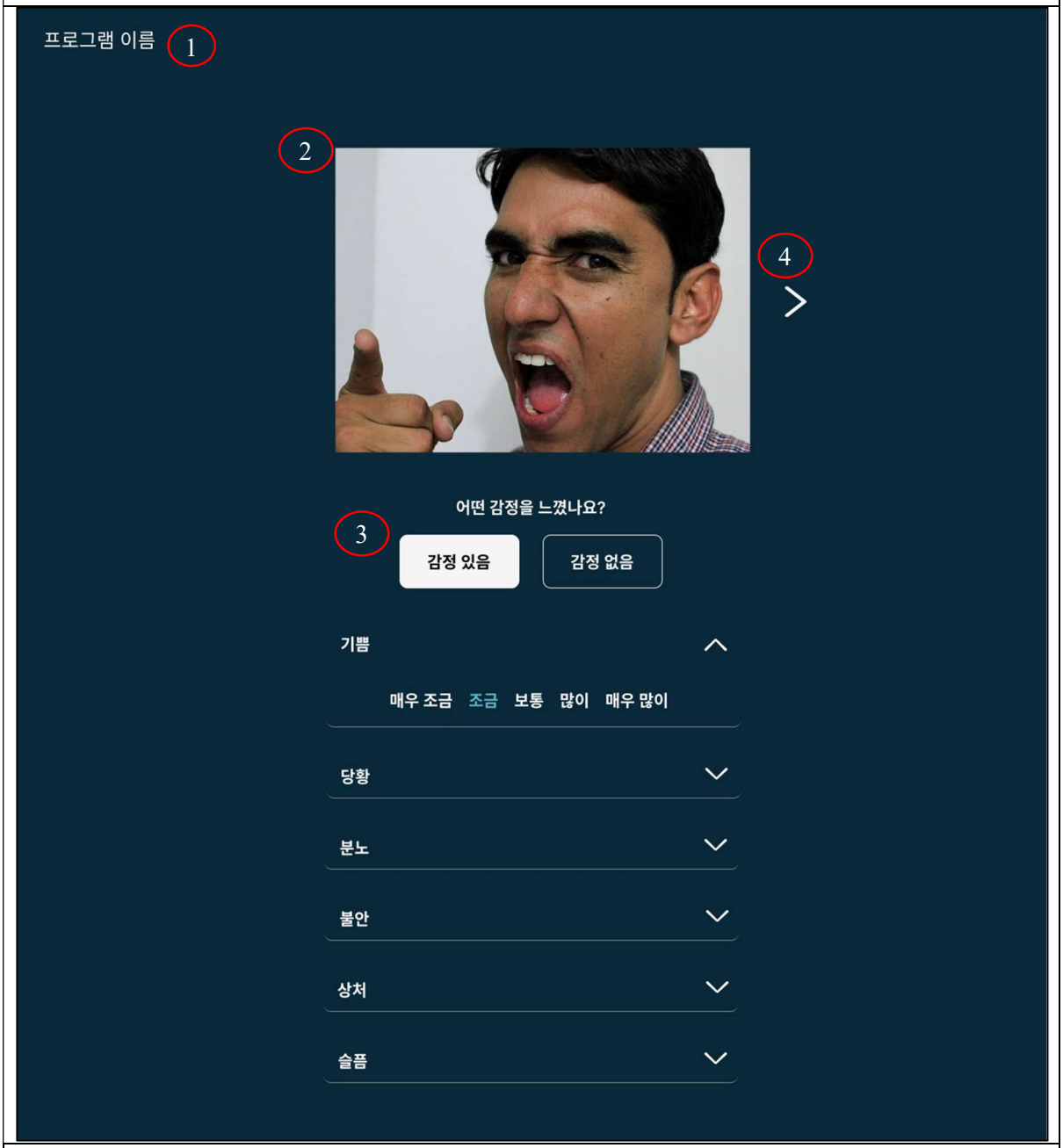
■ 샘플을 띄운 후 3초가 지나면 '테스트 페이지 - 2'가 된다.

① 사이트 로고를 클릭하면 Home 화면으로 이동한다.

② 감정 유발 샘플이 들어간다.

이미지 출처: pixabay

테스트 페이지 - 2



① 사이트 로고를 클릭하면 Home 화면으로 이동한다.

② 감정 유발 샘플(사진, 문구)이 들어간다.

③ 해당 샘플에 대해 사용자가 감정을 느꼈는지 선택지가 들어간다.

'감정 있음' 버튼을 클릭하면 각 감정에 대한 세부적인 선택지가 들어간다.

드롭다운 버튼을 통해 세부 선택지를 보거나 보지 않을 수 있다.

④ 감정 없음을 선택하거나 감정 있음 선택 후 감정을 하나라도 선택하면 다음 테스트로 넘어가는 버튼이 활성화된다. 클릭 시 다음 테스트로 넘어간다.

만약 마지막 테스트였다면 클릭 시 '테스트 결과 페이지' 로 이동한다.

이미지 출처: pixabay

테스트 결과 페이지

프로그램 이름 1

2

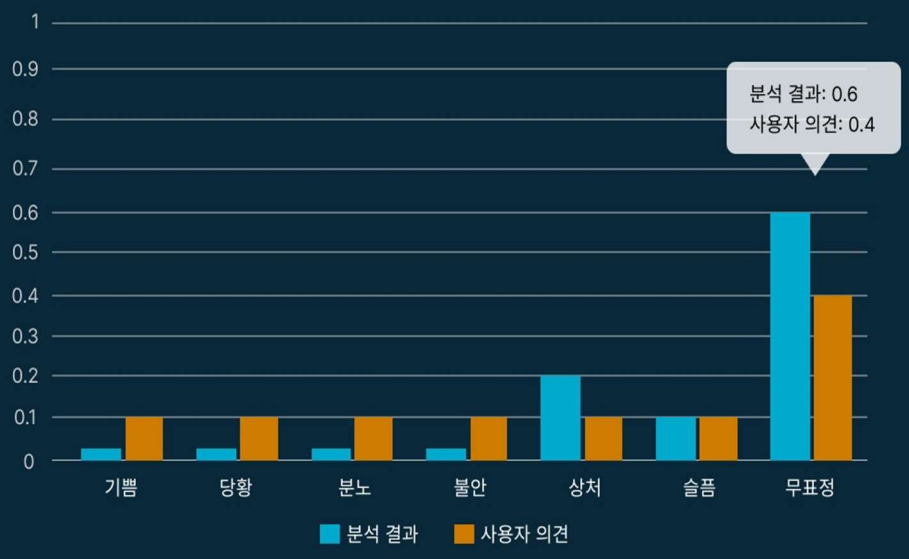
sample1 | sample2 | sample3

3 감정 유발 사진

4 나의 반응

5

페이스 인식 기반 감정 분석기의 오차: XXX



6

7

'기쁨'에 대한 AI 분석 결과는 XX, 사용자 의견은 XX으로 비율 상 XX차이가 난다.
'당황'에 대한 AI 분석 결과는 XX, 사용자 의견은 XX으로 비율 상 XX차이가 난다.
'분노'에 대한 AI 분석 결과는 XX, 사용자 의견은 XX으로 비율 상 XX차이가 난다.
'불안'에 대한 AI 분석 결과는 XX, 사용자 의견은 XX으로 비율 상 XX차이가 난다.
'상처'에 대한 AI 분석 결과는 XX, 사용자 의견은 XX으로 비율 상 XX차이가 난다.
'슬픔'에 대한 AI 분석 결과는 XX, 사용자 의견은 XX으로 비율 상 XX차이가 난다.
'무표정'에 대한 AI 분석 결과는 XX, 사용자 의견은 XX으로 비율 상 XX차이가 난다.

이를 바탕으로 페이스 인식 기반 감정 분석기의 오차를 구하면 XX이다.

① 사이트 로고를 클릭하면 Home 화면으로 이동한다.

② 샘플 번호를 클릭하여 해당 샘플에 대한 분석 결과를 볼 수 있다.

③ 샘플 데이터(사진, 문구)가 들어간다.

④ 해당 샘플 데이터에 대한 나의 표정 사진이 들어간다.

⑤ 구한 오차가 화면에 들어간다.

ai 분석 결과와 사용자 의견에 대한 수치를 막대그래프로 나타낸다.

막대 그래프의 특정 영역을 마우스 호버하면 그 영역의 감정에 대한 ai 분석 결과,

사용자 의견을 수치로 화면에 나타낸다.

⑥ 이전 버튼을 클릭하면 이전 샘플에 대한 분석 결과를, 다음 버튼을 클릭하면

다음 샘플에 대한 분석결과를 화면에 나타낸다.

⑦ 오차 도출 과정을 상세히 글로 나타낸다.

이미지 출처: pixabay

4. 프로젝트 관리

- 주차별 관리 계획

설명	W7	W8	W9	W10	W11	W12	W13	W14	W15	W16
상세설계서 작성	●	●								
AI - 데이터셋 얼굴 크롭 및 이미지 저장			●							
AI - 모델 구현 및 학습				●						
분석결과 및 피드백 관리 기능 구현				●						
AI - RestAPI 구현					●					
사진 파일 송수신 기능 구현						●	●			
분석 결과 출력 기능 구현						●				
Front - 개발 환경 설정			●							
Front - Route 설정			●							
Front - HomePage UI , 기능 완성			●							
Fromt - TestPreparePage UI , 기능 완성			●							
Front - TestProcessPage UI, 기능 완성				●						
Front - TestResultPage UI, 기능 완성				●						
Front - 유닛테스트			●	●						
Front - 리팩토링			●	●	●	●	●	●		
Front - 인터렉션 추가					●					
Front - api 연결 초기 설정					●					
Front - api 연결						●				
Front - 렌더링 최적화							●	●		
Front - 사용자 경험 최적화							●	●		
성능 최적화								●		
테스트									●	●
최종 발표 준비									●	●