

AI Service Development

Service for Watching Video

빅데이터 기반 지능형 서비스 개발

<title>주니어네이버</title>

김민지, 박소연, 신기범, 손지원, 최수민

2022.03.31

Contents

1 팀 소개

- 팀명 소개
- 팀원 소개

2 프로젝트 배경

- 프로젝트 주제
- 선정 이유
- 개요

3 수행 절차 및 방법

- Smart Script
- Eye Detector
- Face Detector

4 수행 결과 및 기대효과

- Smart Script
- Eye Detector
- Face Detector

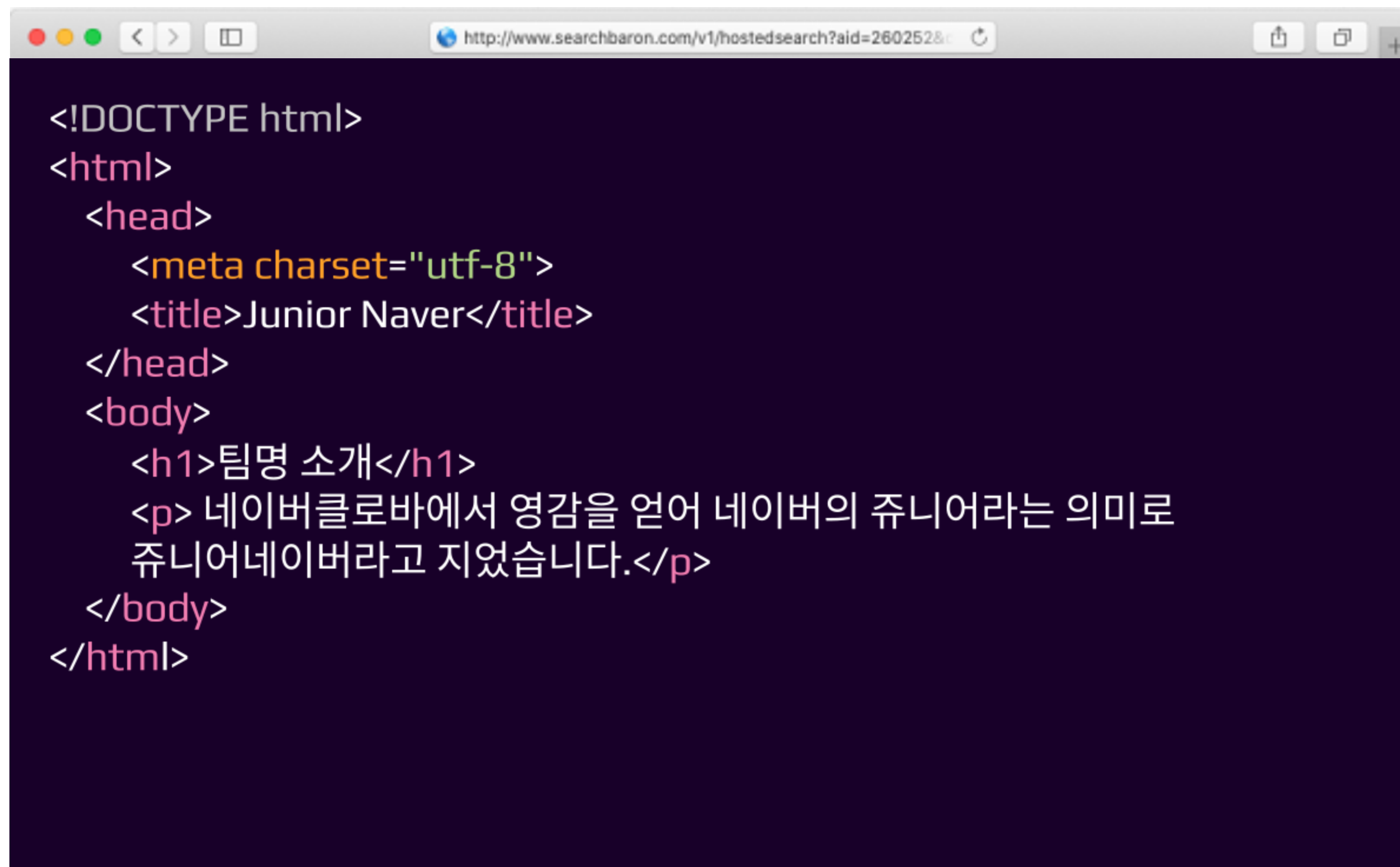
5 후기 및 느낀점

01 팀 소개

팀명 소개
팀원 소개

01. 팀 소개

팀명 소개

A screenshot of a web browser window. The address bar shows the URL 'http://www.searchbaron.com/v1/hostedsearch?aid=260252&c'. The browser window displays a dark-themed code editor with the following HTML code:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Junior Naver</title>
  </head>
  <body>
    <h1>팀명 소개</h1>
    <p> 네이버클로바에서 영감을 얻어 네이버의 주니어라는 의미로
    주니어네이버라고 지었습니다.</p>
  </body>
</html>
```

01. 팀 소개

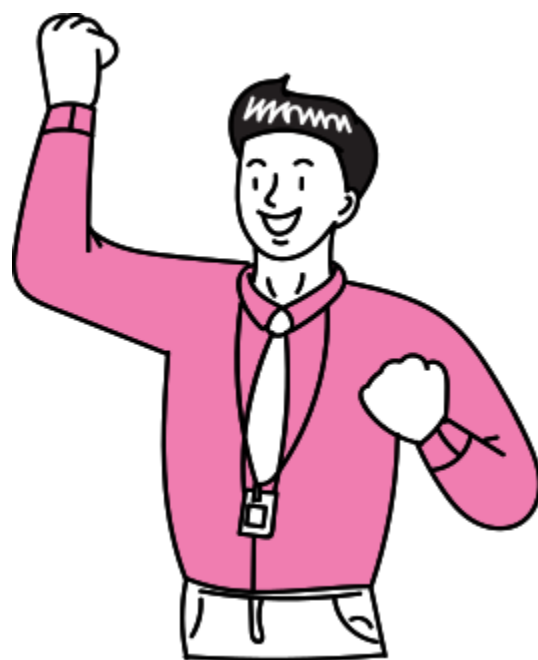
팀원 소개



김민지



박소연



팀장) 신기범



손지원



최수민

Deep Learning, OpenCV, FastAPI, Naver Cloud, HTML

02 프로젝트 배경

프로젝트 주제

선정 이유

개요

프로젝트 배경

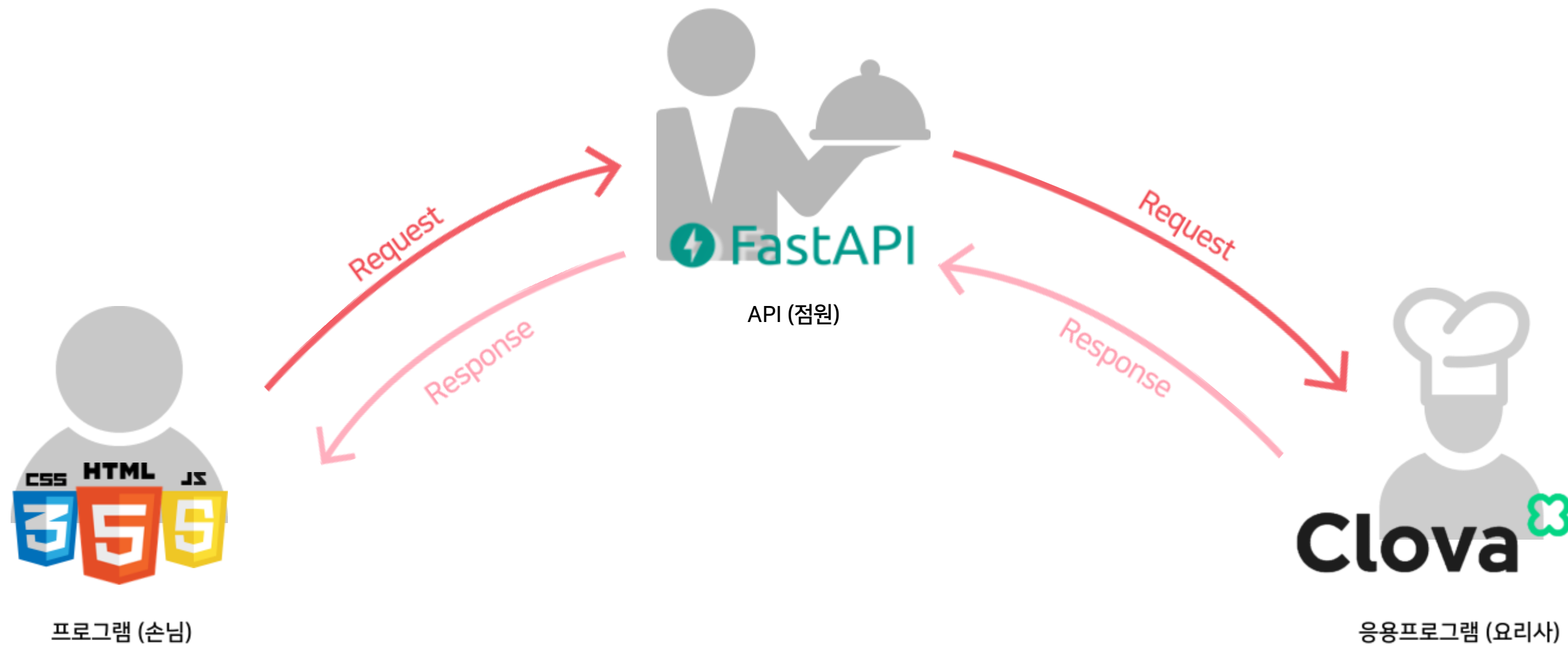


03 수행 절차 및 방법

Smart Script
Eye Detector
Face Detector

03. 수행 절차 및 방법

Smart Script



Smart Script

Smart Text



CLOVA Speech

길고 복잡한 음성을 텍스트로 바꿔주어 다양한 음성 인식 서비스에 활용

Smart Translation



CLOVA Speech

길고 복잡한 음성을 텍스트로 바꿔주어 다양한 음성 인식 서비스에 활용



Papago Translation

입력한 텍스트를 인공지능 기반 번역 알고리즘을 통해 여러 나라의 언어로 자동 번역

Smart Voice



CLOVA Speech

길고 복잡한 음성을 텍스트로 바꿔주어 다양한 음성 인식 서비스에 활용



Papago Translation

입력한 텍스트를 인공지능 기반 번역 알고리즘을 통해 여러 나라의 언어로 자동 번역



CLOVA Voice

고품질 음성 합성 기술로 다양하고 자연스러운 목소리 제공

Smart Script

get : 서버로부터 리소스를 가져올 때 사용하는 read API

- Speech_text.html은 사용자가 파일을 업로드하는 HTML
- FastAPI에서 GET 메소드를 사용하여 경로를 지정하고, return으로 HTML을 설정하면 경로로 이동했을 때 HTML 화면을 볼 수 있다.

```
@app.get('/speech_text', response_class=HTMLResponse)
async def speech_text_file(request: Request):
    return templates.TemplateResponse("speech_text.html", {"request": request})
```



```
%%writefile templates/speech_text.html
<!DOCTYPE html>
<body>
  <h2>JN_Script</h2>
  <form action="/speech/text" method="post" enctype="multipart/form-data">
    <input type="file" name="files">
    <input type="submit" value="Submit">
  </form>

  <p>파일을 업로드하고 'Submit'을 클릭하면 음성파일 텍스트로 반환해 줍니다.</p>

</body>
</html>
```

03. 수행 절차 및 방법

Smart Script

post : 서버에 데이터를 저장할 때 쓰는 create API

- 사용자가 파일을 업로드하고 서비스 요청을 하면 POST 메소드를 통해 데이터가 전송된다.
- 전송받은 데이터는 Naver CLOVA를 통해 작업이 수행된다.
- 가공된 데이터는 return으로 지정한 speech_text_result.html로 보내진다.

```
@app.post('/speech/text', response_class=HTMLResponse)
async def speech_text(request: Request, files: List[UploadFile] = File(...)):
    for file in files:
        contents = await file.read()
        with open(os.path.join('/content/', file.filename), 'wb') as fp:
            fp.write(contents)
        if __name__ == '__main__':

            res = ClovaSpeechClient().req_upload(file=os.path.join('/content/' + file.filename)), completion='sync')

            result = res.json()['text']    response 데이터에서 text부분만 추출

    #return HTMLResponse(content=F"<html>{result}", status_code=200)
    return templates.TemplateResponse("speech_text_result.html", {"request": request, "result": result})
```



```
%%writefile templates/speech_text_result.html
<!DOCTYPE html>
<html>
<body>
    <h2>변환 내용입니다.</h2>
    <p>{{result}}</p>
</body>
</html>
```


03. 수행 절차 및 방법

Smart Script

```
class ClovaSpeechClient:
    # Clova Speech invoke URL
    invoke_url = '+++'
    # Clova Speech secret key
    secret = '+++'

    def req_upload(self, file, completion, callback=None, userdata=None, forbiddens=None, boostings=None,
                   wordAlignment=True, fullText=True, diarization=None):
        request_body = {
            'language': 'enko',
            'completion': completion,
            'callback': callback,
            'userdata': userdata,
            'wordAlignment': wordAlignment,
            'fullText': fullText,
            'forbiddens': forbiddens,
            'boostings': boostings,
            'diarization': diarization,
        }
        headers = {
            'Accept': 'application/json;UTF-8',
            'X-CLOVASPEECH-API-KEY': self.secret
        }
        print(json.dumps(request_body, ensure_ascii=False).encode('UTF-8'))
        files = {
            'media': open(file, 'rb'),
            'params': (None, json.dumps(request_body, ensure_ascii=False).encode('UTF-8'), 'application/json')
        }
        response = requests.post(headers=headers, url=self.invoke_url + '/recognizer/upload', files=files)
        return response

@app.post('/speech/voice')
async def speech_voice(files: List[UploadFile] = File(...)):
    for file in files:
        contents = await file.read()
        with open(os.path.join('/content/', file.filename), 'wb') as fp:
            fp.write(contents)
    if __name__ == '__main__':
        res = ClovaSpeechClient().req_upload(file=os.path.join('/content/' + file.filename), completion='sync')
```

```
result = res.json()['text']
# translation
client_id =
client_secret =
encText = urllib.parse.quote(result)
data = "source=en&target=ko&honorific=True&text=" + encText # 영어 한국어 설정
url = "https://naveropenapi.apigw.ntruss.com/nmt/v1/translation"
request = urllib.request.Request(url)
request.add_header("X-NCP-APIGW-API-KEY-ID", client_id)
request.add_header("X-NCP-APIGW-API-KEY", client_secret)
response = urllib.request.urlopen(request, data=data.encode('utf-8'))
rescode = response.getcode()
if(rescode==200):
    response_body = response.read()
    print(response_body.decode('utf-8'))
else:
    print("Error Code:" + rescode)

mas = response_body.decode('utf-8')
translation = mas.split('')[2]

# voice
encText = urllib.parse.quote(translation)
data = "speaker=nara&volume=0&speed=0&pitch=0&format=mp3&text=" + encText;
url = "https://naveropenapi.apigw.ntruss.com/tts-premium/v1/tts"
request = urllib.request.Request(url)
request.add_header("X-NCP-APIGW-API-KEY-ID", client_id)
request.add_header("X-NCP-APIGW-API-KEY", client_secret)
response = urllib.request.urlopen(request, data=data.encode('utf-8'))
rescode = response.getcode()
if(rescode==200):
    # print("TTS mp3 저장")
    response_body = response.read()
    fn = file.filename.split('.')[0]
    with open('./static/%s_voice.mp3'% fn, 'wb') as f:
        f.write(response_body)
else:
    print("Error Code:" + rescode)
    file_path = os.getcwd() + '/static/%s_voice.mp3'% fn
return FileResponse(path=file_path, media_type='application/octet-stream', filename='%s_voice.mp3'% fn)
```

Eye Detector



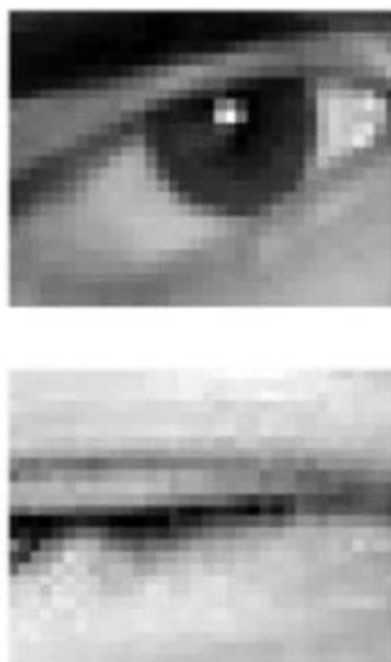
03. 수행 절차 및 방법

Eye Detector



- 눈 이미지만 모아둔 데이터셋을 찾아 학습한다.
- 사진의 수가 부족하다고 판단하여 Data augmentation을 통해 학습량을 증가시킨다.
- Conv2D과 Maxpulling2D으로 차원 축소를 한 뒤, Flatten과 Dense layer를 사용하여 층을 쌓고 Compile 하여 학습한다.

Dataset



Data Augmentation

```
train_datagen = ImageDataGenerator(  
    rescale=1./255,  
    rotation_range=10,  
    width_shift_range=0.2,  
    height_shift_range=0.2,  
    shear_range=0.2  
)  
  
val_datagen = ImageDataGenerator(rescale=1./255)  
  
train_generator = train_datagen.flow(  
    x=x_train, y=y_train,  
    batch_size=32,  
    shuffle=True  
)  
  
val_generator = val_datagen.flow(  
    x=x_val, y=y_val,  
    batch_size=32,  
    shuffle=False  
)
```



Data Compile

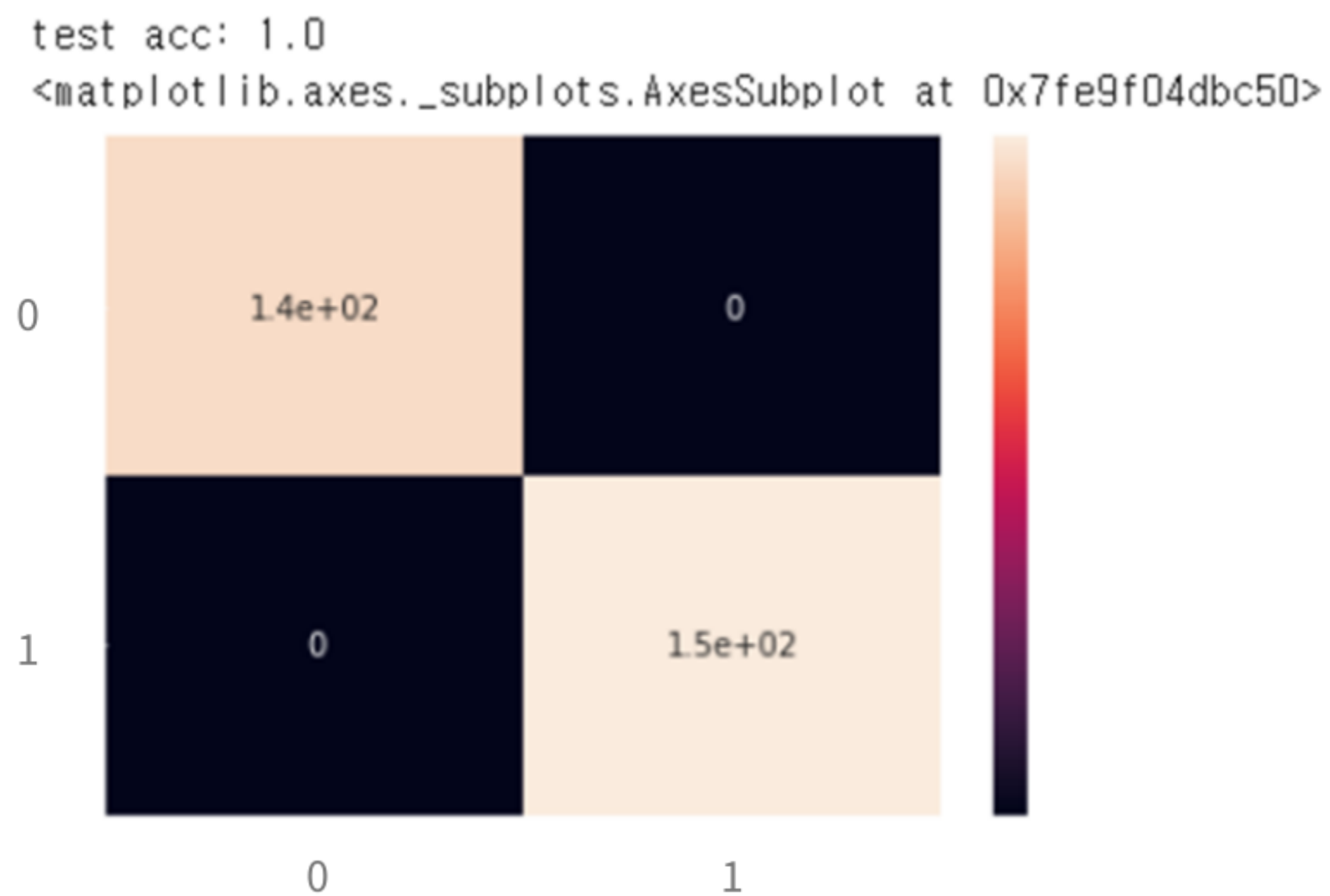
Model: "model"		
Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 26, 34, 1)]	0
conv2d (Conv2D)	(None, 26, 34, 32)	320
max_pooling2d (MaxPooling2D)	(None, 13, 17, 32)	0
conv2d_1 (Conv2D)	(None, 13, 17, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 6, 8, 64)	0
conv2d_2 (Conv2D)	(None, 6, 8, 128)	73856
max_pooling2d_2 (MaxPooling2D)	(None, 3, 4, 128)	0
flatten (Flatten)	(None, 1536)	0
dense (Dense)	(None, 512)	786944
activation (Activation)	(None, 512)	0
dense_1 (Dense)	(None, 1)	513
activation_1 (Activation)	(None, 1)	0
Total params: 880,129		
Trainable params: 880,129		
Non-trainable params: 0		

Eye Detector



성능 평가

- Test acc가 1.0으로 매우 좋은 정확도를 기록한다.
- 0은 눈을 감은 것을 의미하고 1은 눈을 뜬 것으로 인지한다.
- 0일 때는 모두 0으로 인식했고, 1일 때는 모두 1로 인식한 것을 확인할 수 있었다.



Eye Detector



- Face Landmark는 얼굴의 특징점을 나타낸다.
- 우리가 사용한 랜드마크 추출 알고리즘은 총 68개의 특징점을 가져온다.
- 동공의 위치, 입술의 좌표, 눈썹의 좌표 등의 값을 추출하여 x, y 값으로 반환한다.
- 이를 활용하여 얼굴을 감지할 수 있다.



03. 수행 절차 및 방법

Eye Detector



- 사용자의 왼쪽 눈과 오른쪽 눈에 해당하는 각각의 Face Landmark 포인트를 지정한다.
- 0~255로 되어있는 정수를 0.0~1.0까지의 실수로 변환시킨다.
- rectangle()와 putText() 함수를 사용하여 눈 위치에 Bounding Box를 그리고, 눈 감김 정도를 수치화하여 OpenCV 화면에 나타냈다.

```
for face in faces:
    shapes = predictor(gray, face)
    shapes = face_utils.shape_to_np(shapes)

    eye_img_l, eye_rect_l = crop_eye(gray, eye_points=shapes[36:42])
    eye_img_r, eye_rect_r = crop_eye(gray, eye_points=shapes[42:48])

    eye_img_l = cv2.resize(eye_img_l, dsize=IMG_SIZE)
    eye_img_r = cv2.resize(eye_img_r, dsize=IMG_SIZE)
    eye_img_r = cv2.flip(eye_img_r, flipCode=1)

    cv2.imshow('l', eye_img_l)
    cv2.imshow('r', eye_img_r)

    eye_input_l = eye_img_l.copy().reshape((1, IMG_SIZE[1], IMG_SIZE[0], 1)).astype(np.float32) / 255.
    eye_input_r = eye_img_r.copy().reshape((1, IMG_SIZE[1], IMG_SIZE[0], 1)).astype(np.float32) / 255.

    pred_l = model.predict(eye_input_l)
    pred_r = model.predict(eye_input_r)

    # visualize
    state_l = '%.1f' if pred_l > 0.1 else '%.1f'
    state_r = '%.1f' if pred_r > 0.1 else '%.1f'

    state_l = state_l % pred_l
    state_r = state_r % pred_r

    cv2.rectangle(img, pt1=tuple(eye_rect_l[0:2]), pt2=tuple(eye_rect_l[2:4]), color=(255,0,0), thickness=2)
    cv2.rectangle(img, pt1=tuple(eye_rect_r[0:2]), pt2=tuple(eye_rect_r[2:4]), color=(255,0,0), thickness=2)

    cv2.putText(img, state_l, tuple(eye_rect_l[0:2]), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255,255,255), 2)
    cv2.putText(img, state_r, tuple(eye_rect_r[0:2]), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255,255,255), 2)
```

Eye Detector



- 만약 양쪽 눈의 예측 값이 0.1보다 작거나 같으면 count가 시작된다.
- 만약 count가 5보다 크면 "warning"이라는 경고 메시지가 나온다.
- 만약 count가 10보다 크면 영상 재생이 중지된다.

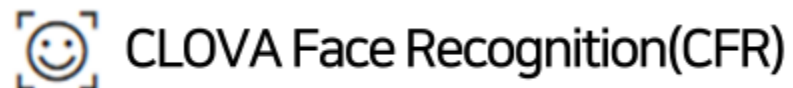
```
if pred_l <= 0.1 and pred_r <= 0.1:
    count += 1
    print(count)
    cv2.putText(img, str(count), (465, 130), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
    time.sleep(1)
    if count > 5:
        org = (400, 400)
        text = "warning"
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(img, text, org, font, 1.5, (0, 0, 255), 2)
        if count > 10:
            print('영상 재생을 중지합니다.')
            break
    else:
        count = 0

cv2.imshow('result', img)
if count > 10:
    break

if (cv2.waitKey(1) == '27'):
    break
```

03. 수행 절차 및 방법

Face Detector



- Naver CLOVA는 이미지 속의 얼굴을 감지하고 인식하여 얻은 다양한 정보를 제공하는 Face Recognition(CFR)을 사용한다.
- Face Detector 역시 HTML 생성 > FastAPI > Naver CLOVA 순으로 수행한다.

```
xxwritefile templates/face.html
<!DOCTYPE html>
<html>
<body>
  <meta charset="utf-8">
  <title>Face_Detector</title>
  <link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-awesome/4.6.2/css/font-awesome.min.css">
  <link rel="stylesheet" href="{{ url_for('static', path='/main2.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', path='/style.css') }}">
</head>
<body>
  <div id="wrap">
    <nav>
      <ul class="nav-items">
        <li><a href="#home">Sign up</a></li>
        <li><a href="#news">Login</a></li>
        <li><a href="#contact">Service center</a></li>
      </ul>
    </nav>
    <header>
      <a class="logo" href="/"></a>
    </header>
    <div>
      <div class="container">
        <ul>
          <li><a href="#">Smart Script<i class="fa fa-angle-down"></i></a>
            <ul>
              <li><a href="/speech_text">Smart Text</a></li>
              <li><a href="/speech_translation">Smart Translation</a></li>
              <li><a href="/speech_voice">Smart Voice</a></li>
            </ul>
          </li>
        </ul>
      </div>
    </div>
  </div>
</body>
</html>
```

```
@app.get('/face', response_class=HTMLResponse)
async def face_age(request: Request):
    return templates.TemplateResponse("face.html", {"request": request})

@app.post('/face/age', response_class=HTMLResponse)
async def face_age_result(request: Request, files: List[UploadFile] = File(...)):
    for file in files:
        contents = await file.read()
        with open(os.path.join('/content/', file.filename), 'wb') as fp:
            fp.write(contents)
        client_id = "s3etavmw8n"
        client_secret = "k4VTQDwCAm3yJMbJCyg0lqlvNAJk35glQKuMzjkr"
        url = "https://naveropenapi.apigw.ntruss.com/vision/v1/face"
        files = {'image': open(os.path.join('/content/', file.filename), 'rb')}
        headers = {'X-NCP-APIGW-API-KEY-ID': client_id, 'X-NCP-APIGW-API-KEY': client_secret}
        response = requests.post(url, files=files, headers=headers)
        rescode = response.status_code
        if(rescode==200):
            print(response.text)
        else:
            print("Error Code:" + rescode)
        res = response.json()
        low = int(res['faces'][0]['age']['value'].split('~')[0])
        high = int(res['faces'][0]['age']['value'].split('~')[1])
        if (low + high) / 2 < 15:
            msg = "시청 불가능한 연령입니다.(15세 미만)"
        else:
            msg = "시청 가능한 연령입니다.(15세 이상)"

    return templates.TemplateResponse("face_result.html", {"request": request, "msg":msg})
```


03. 수행 절차 및 방법

Face Detector



gender : "child"
age : "0~4"
emotion : "neutral"
pose : "frontal_face"



gender : "child"
age : "4~8"
emotion : "neutral"
pose : "frontal_face"



gender : "child"
age : "8~12"
emotion : "smile"
pose : "frontal_face"



gender : "child"
age : "10~14"
emotion : "smile"
pose : "frontal_face"



gender : "female"
age : "16~20"
emotion : "netural"
pose : "frontal_face"

04 수행 결과 및 기대 효과

Smart Script
Eye Detector
Face Detector

04. 수행 결과 및 기대효과

Smart Script - Smart Text



자료 제공 : tvN 드라마 <스물하나 스물다섯>

- Smart Text.html에 영상을 업로드하면 영상 속 음성을 분석해 텍스트로 반환한다.
- 화자 인식 기능을 추가해 화자를 구분하여 나타냈다.
- Copy 기능을 추가하여 사용자가 손쉽게 데이터를 가져갈 수 있도록 하였다.

결과 화면

Smart Text

이 앱은 비디오 파일의 스크립트를 추출하기에 유용합니다.
이 앱은 설치가 필요하지 않고 브라우저에서 작동합니다.

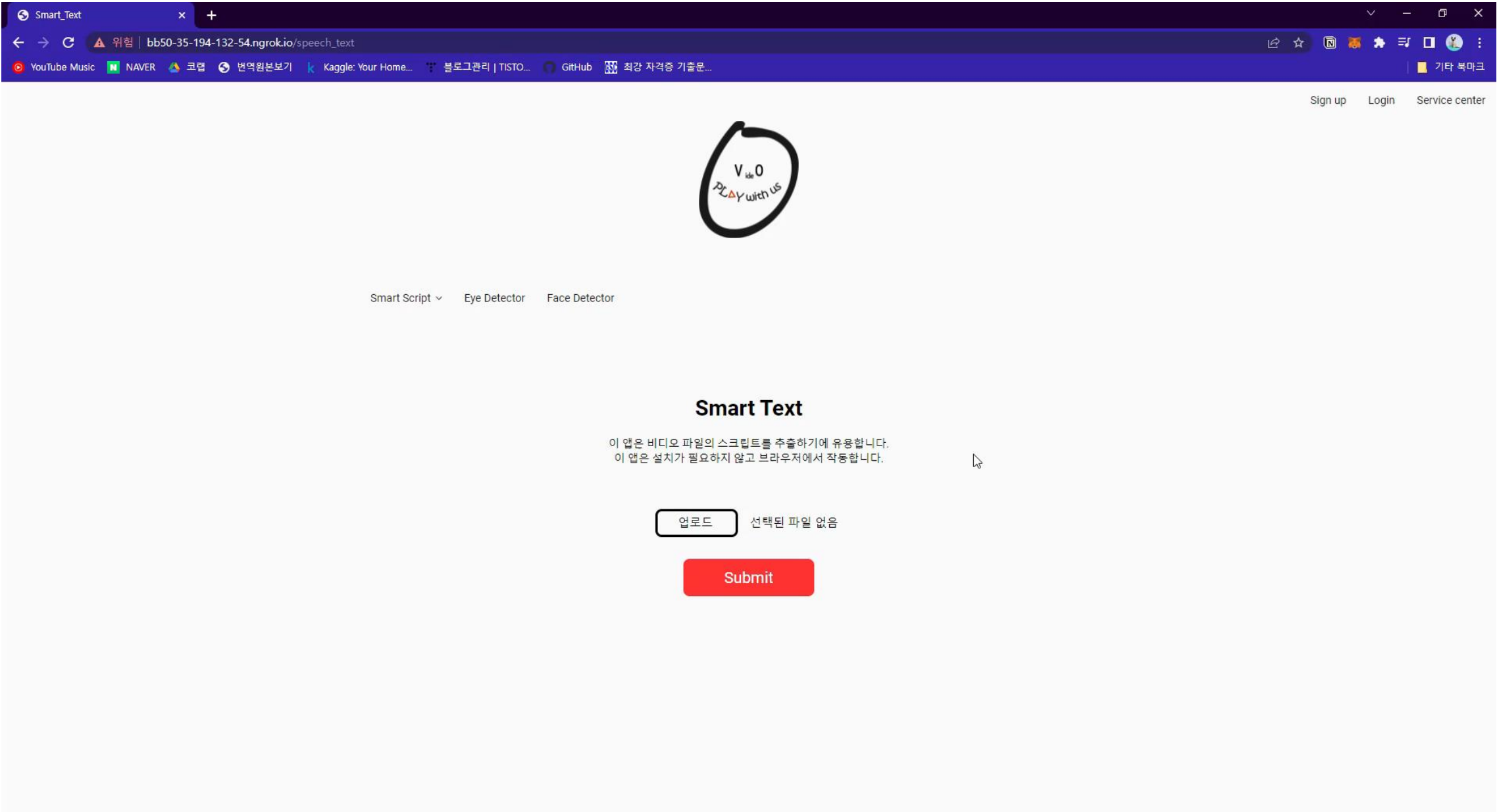
변환 내용입니다.

[[1:나이도], [1:무슨 일이야], [2:엄마가 의자 버렸어 아마가 많은 의자], [2:엄마가 버렸어 없어 없어졌어], [1:찾아보자 어 같이 찾자 울지마 해도야 울지마], [1:의자가 세 개인데 하나는 부러져 있고 못 보셨어요], [3:어 못 봤는데], [4:벌써 수거에 간 것 같아 그만 찾자], [1:괜찮아], [4:원래 안 울었는데 너 봐서 온 거야 너 때문이야], [1:엄마랑 심하게 싸웠어], [4:거의 무슨 무슨 대접 이런 느낌으로 역사에 남을 만하게], [1:신창원 숙보던 날 그날 재경 선배 진짜 멋있었거든 나 방송국 들어와서 처음으로 그런 생각했어], [1:우와 저 사람처럼 되고 싶다 그런데 그런 상황이], [1:너한테 상처였다니깐 좀 복잡해지네 선배 프로정신 뒤편 놀 니 상처가 따라오는 거니까], [4:이상하다], [4:나한테는 상처였지만 널 꿈꾸게 했구나 우리 엄마], [4:그거는], [4:그거대로 좋은데]]

Copy

04. 수행 결과 및 기대효과

Smart Script - Smart Text



04. 수행 결과 및 기대효과

Smart Script - Smart Translation



자료 제공 : Youtube <TED>

- Smart translation.html에 영상을 업로드하면 영상 속 음성을 분석한 뒤, 언어를 번역하여 텍스트로 반환한다.
- Copy 기능을 추가하여 사용자가 손쉽게 데이터를 가져갈 수 있도록 하였다.

결과 화면

Smart Translation

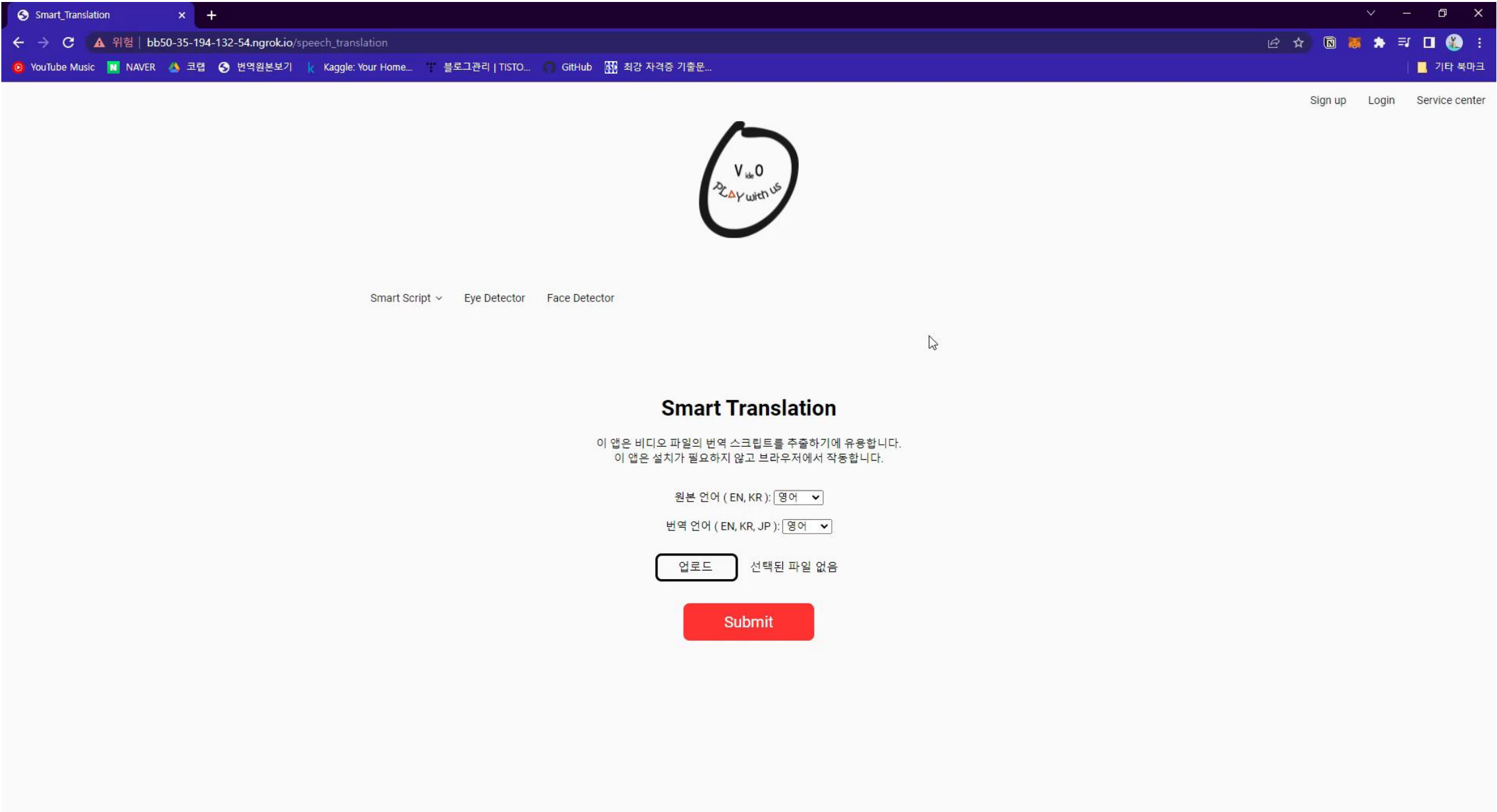
이 앱은 비디오 파일의 번역 스크립트를 추출하기에 유용합니다.
이 앱은 설치가 필요하지 않고 브라우저에서 작동합니다.

변환 내용입니다.

그래서 우리는 어떻게 배우고 왜 그들 중 일부는 다른 사람들보다 더 쉽게 배울 수 있을까? 그래서 나는 의사이고 나는 여기 브리티시컬럼비아 대학의 뇌 연구자이고 이것들은 나를 매료시키는 질문이다. 그래서 뇌 연구는 인간의 생리를 이해하는 데 있어서도 또한 고려의 위대한 프런티어 중 하나이다.우리를 만드는 것의 비결은 뇌 연구원이 되는 것은 놀라운 시기입니다. 그리고 저는 제가 세상에서 가장 흥미로운 직업을 가지고 있다는 것을 말씀드리고 싶습니다. 우리가 뇌에 대해 알고 있는 것은 숨가쁘게 변화하고 있습니다. 그리고 우리가 뇌에 대해 알고 있다고 생각했던 것의 대부분은 사실이 아니거나 현재 이 잘못된 것들 중 일부입니다. 예를 들어, 우리는 어린 시절 이후 뇌가 정말로 변할 수 없다고 생각하곤 했다. 그리고 알고보니 뇌에 대한 또 다른 오해는 당신이 아무때나 그것의 일부만을 사용하고 아무것도 잘 하지 않을 때 침묵한다는 것이다. 이 또한 사실이다.sns는 휴식이 되어 아무것도 생각하지 않을 때에도 뇌는 매우 활동적이기 때문에 모리 등의 테크놀로지의 진보로 인해 이러한 중요한 발견을 할 수 있게 되었습니다.또, 이러한 발견 중 가장 흥미롭고 변혁적인 것은, 새로운 요소 스칼을 배울 때마다, 새로운 스칼을 배울 수 있다는 것입니다.당신의 뇌를 손상시키는 것은 우리가 신경가소성이라고 부르는 것이고, 그래서 그것은 거의 25년 전입니다. 우리는 사춘기 이후 일어난 유일한 변화가 노화와 함께 뇌세포의 손실이며 뇌졸중처럼 손상된 결과라고 생각했고, 그리고 나서 연구는 성인의 뇌와 뇌의 재조직에 주목할 만한 변화를 보여주기 시작했습니다.Summing 연구는 우리의 모든 행동이 우리의 뇌를 변화시킨다는 것을 보여주었습니다. 이러한 변화는 나이에 의해 제한되지 않습니다. 그리고 사실 그것들은 항상 일어나고 있습니다. 그리고 매우 중요한 것은 뇌의 재편이 다시이 소사 통해 회복을 도우니다.가가가 변하는 시점 가소성이기 때문에

04. 수행 결과 및 기대효과

Smart Script - Smart Translation



04. 수행 결과 및 기대효과

Smart Script - Smart Voice



자료 제공 : Youtube <TED>

- Smart Voice.html에 영상을 업로드하면 영상 속 음성을 분석한 뒤, 언어 번역을 하고, 그것을 음성 mp3 파일로 제공한다.
- 사용자가 음성 파일을 다운로드할 수 있도록 기능을 추가하였다.




결과 화면

Smart Voice

파일을 업로드하고 'Submit'을 클릭하면 영어를 한국어로 번역 후 음성으로 녹음한 파일을 지원해 줍니다.
이 앱은 설치가 필요하지 않고 브라우저에서 작동합니다.

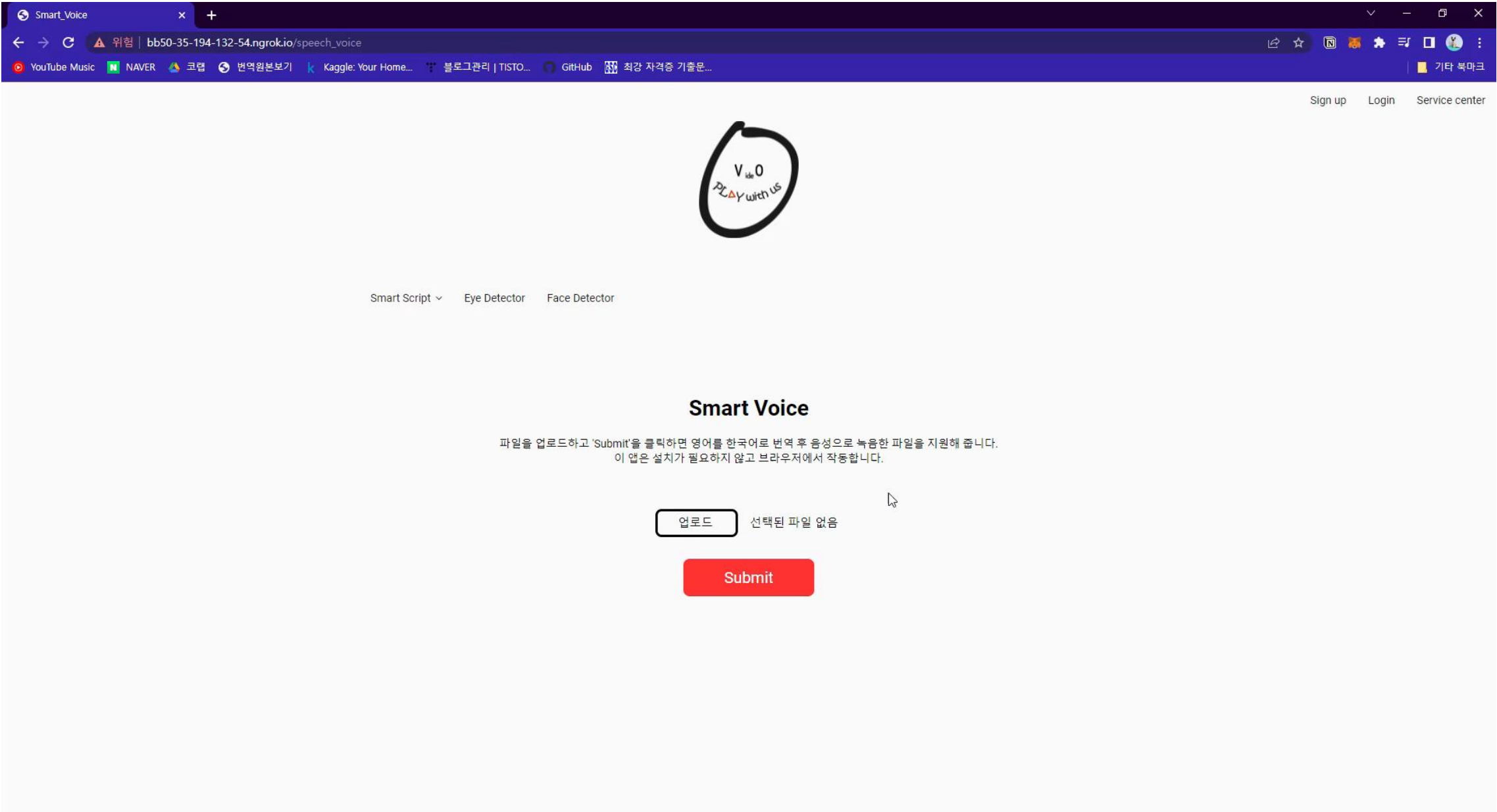
Lara_Boyd_edit.mp4

 Lara_Boyd_edit_v....mp3 ^

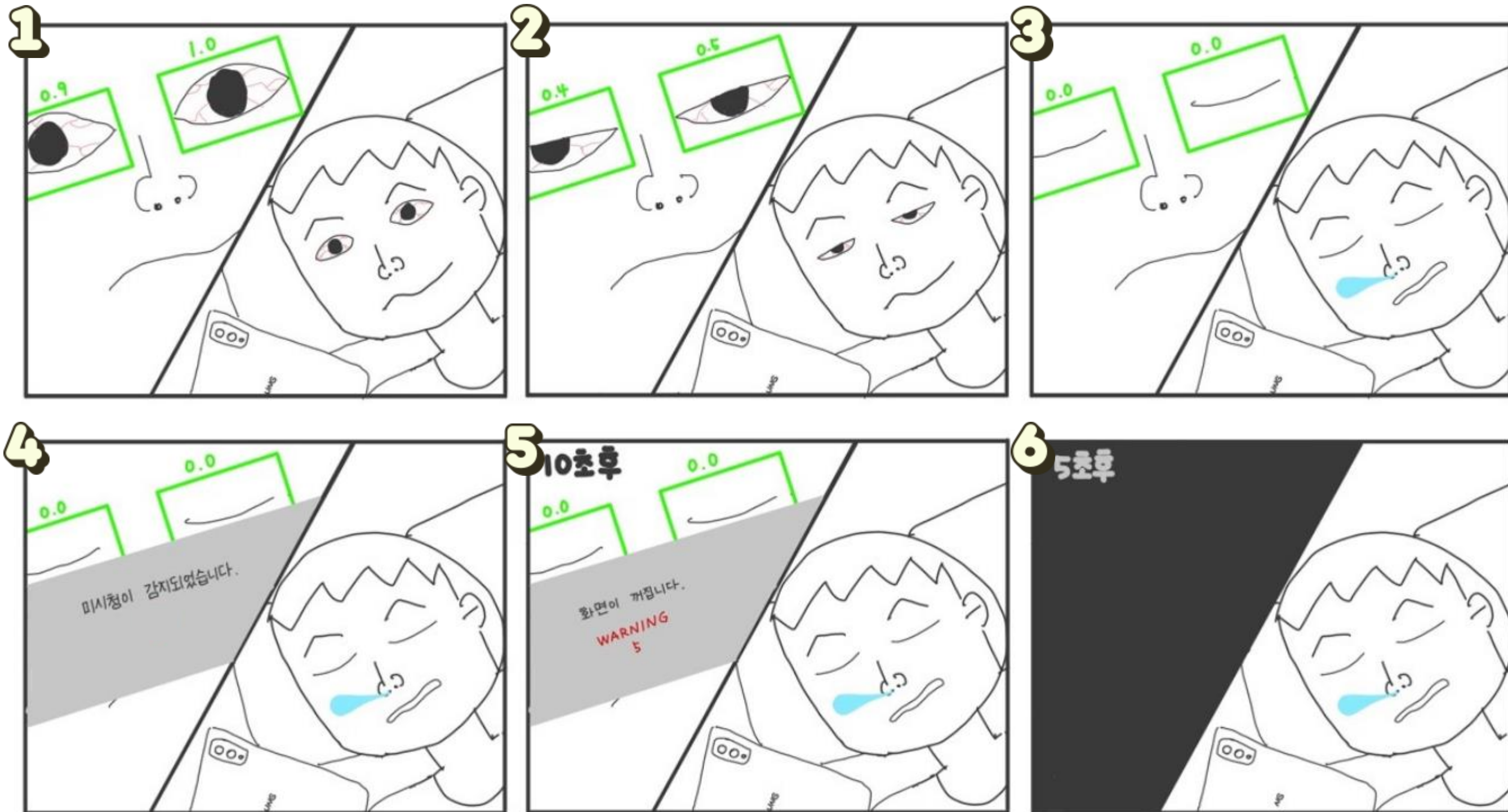
- 음성전환파일 다운로드

04. 수행 결과 및 기대효과

Smart Script - Smart Voice

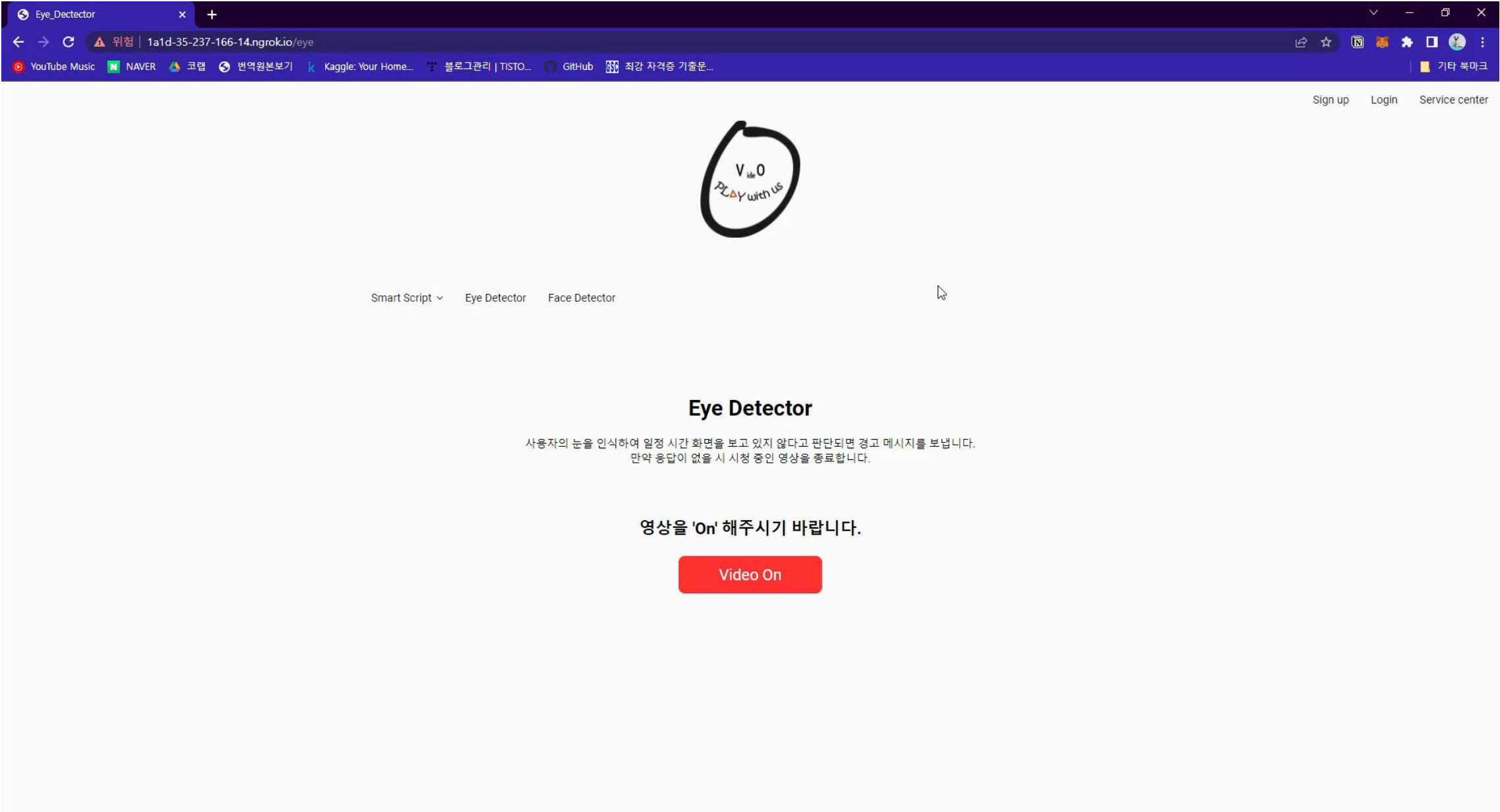


Eye Detector



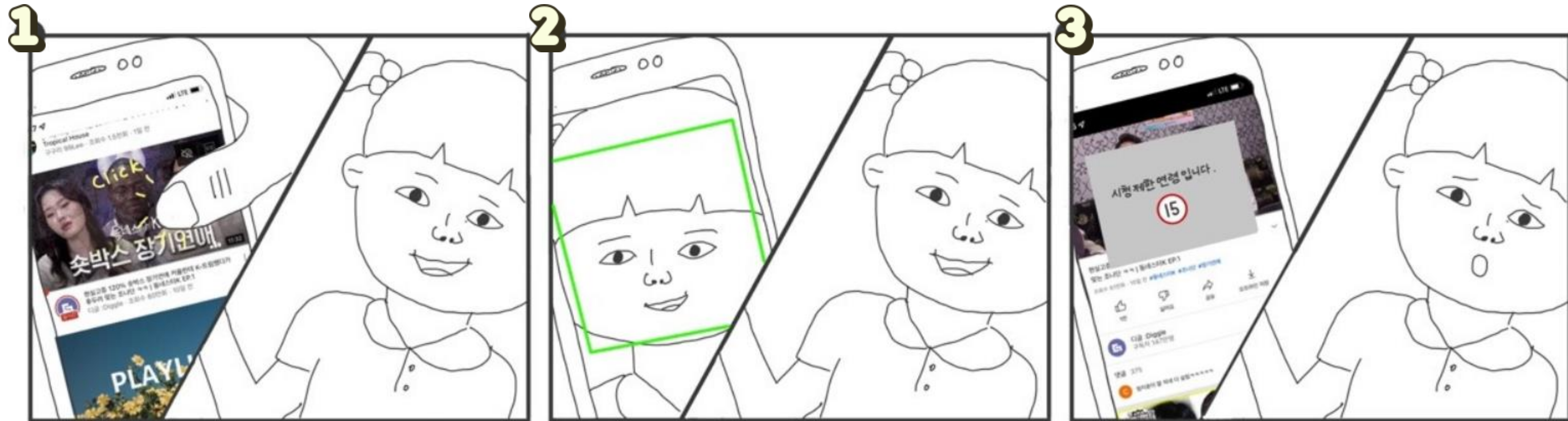
04. 수행 결과 및 기대효과

Eye Detector



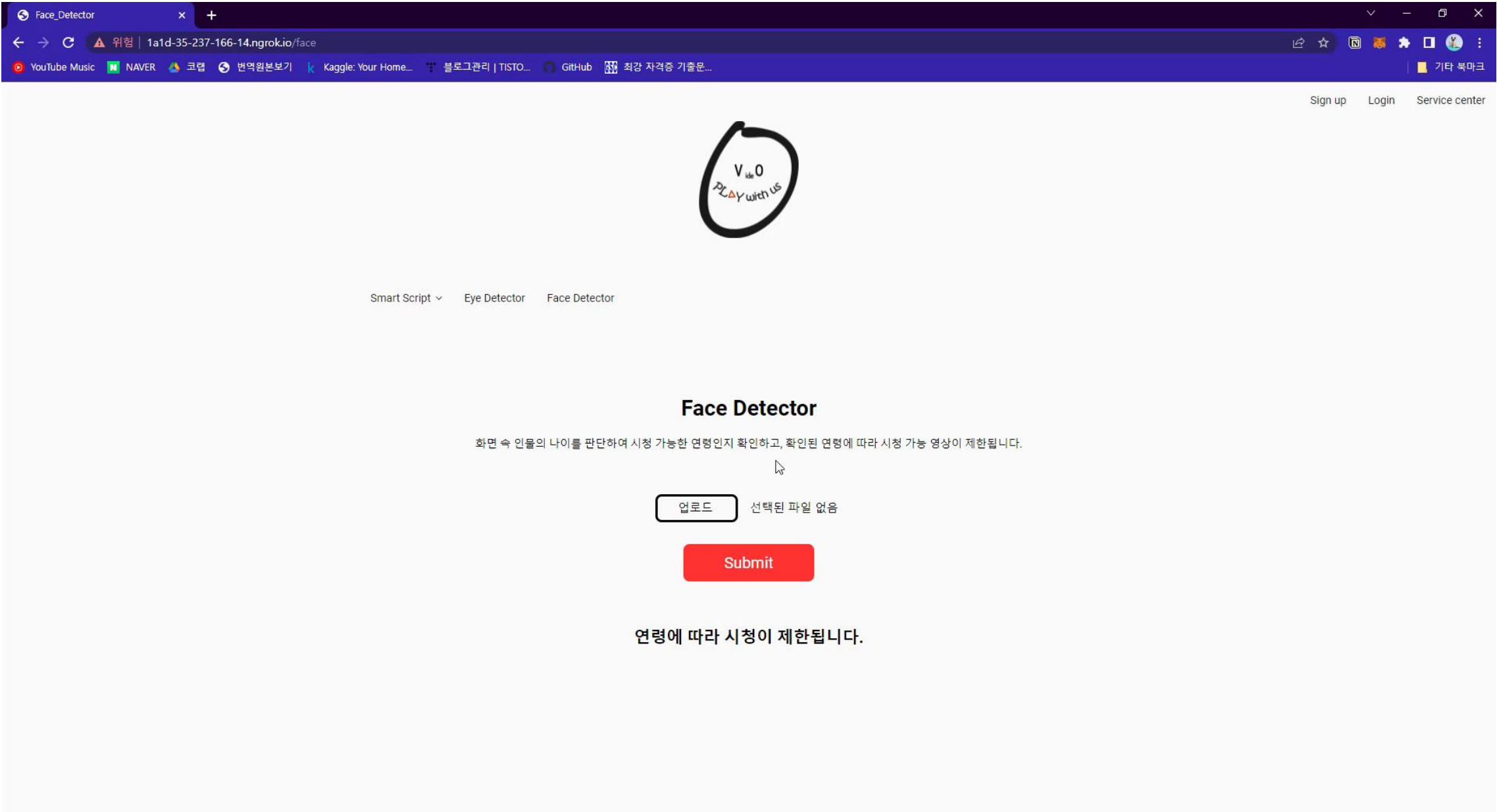
04. 수행 결과 및 기대효과

Face Detector



04. 수행 결과 및 기대효과

Face Detector



기대 효과

Smart Script

- 대본 스크립트 제공 가능
- shadowing 학습법에 적합한 자료 제공
- 원하는 영상의 대본을 한번에 번역하여 자료 제공
- 자막을 볼 수 없는 상황에서 유용하게 사용 가능

Eye Detector

- 자는 동안 화면 켜짐 방지
- 이전 시청 기록 기억
- 편안한 수면 환경 조성

Face Detector

- 우리 아이에게 부적절한 영상 접근 제한
- 나이에 따라 시청 가능 영상의 분류가 가능
(ex 12세, 15세 등)

05 후기 및 느낀점

05. 후기 및 느낀점

후기 및 느낀점



한 번 더 성장하는 계기가 되었다고 생각한다. 훌륭한 팀원들 덕분에 프로젝트도 잘 마무리할 수 있었고, 재밌게 진행해서 기억에 오래 남을 것 같다. 무엇보다 딥러닝뿐만 아니라 백엔드와 프론트엔드 부분 등 다양한 분야를 학습할 수 있어 좋았다. - 신기범



CNN, OpenCV으로 영상 인식을 해보고, FastAPI와 Naver AI 서비스를 활용해 html로 웹페이지까지 구성해볼수있는 기회를 가질수있었다.서로 부족한 부분을 채워줄수있는 조원들이었다고 생각하고, 함께 진행해서 시간내에 마무리할수있었다. - 박소연



html 공부를 간단히 기초적인 것만 배워봤지만 팀원들과 같이 공부하면서 팀워크의 중요성, 협동성을 배웠습니다. 팀원들과 서로 부족한 부분을 설명하고 코드에 대한 이해가 되었습니다. 혼자 작업할 때보단 팀원들과 서로 의견과 아이디어로 프로젝트 수행하면서 결과가 좋게 나와서 좋았습니다. - 김민지

기존에 배웠던 CNN 머신러닝, 새롭게 배운 FastAPI를 통한 서버구축, html, css, javascript을 사용하여 최종적으로 웹사이트를 만들어서 사용자가 서비스를 이용할 수 있게 만들었습니다. 이번 프로젝트로 좋은 팀원들과 함께 노력하여 만족스러운 결과물을 만들어낼 수 있었고, 이번에 웹사이트까지 구현해서 정말 우리가 서비스를 만드는 느낌이 들어서 재미있었습니다. - 최수민



우리가 배운 내용들이 실제로 서비스에 어떻게 적용되고 구현되는지 경험해볼 수 있었고, 프로젝트를 진행하면서 느꼈던 부족한 점을 앞으로 채워나가야 겠다고 생각했다. - 손지원



Thank you