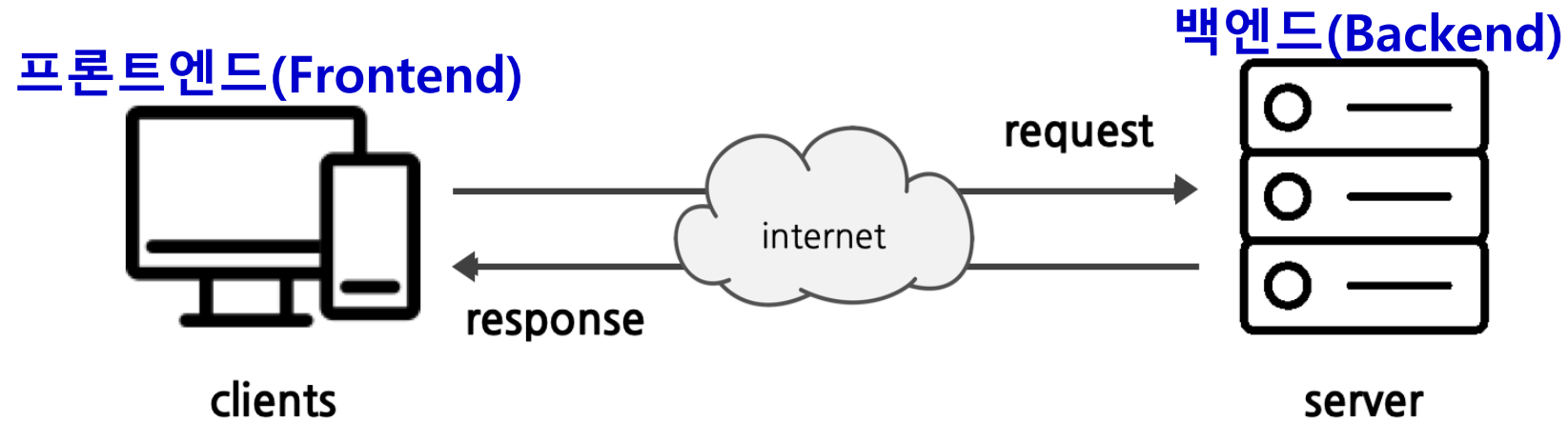


Vision AI 웹서비스 개발



웹서비스 구조



GET

client가 server에 특정 data를 요청하여 받음

client



Request
URL ? data

Response

server



Data에 변화없이 해당
data 전달

VS

POST

client가 server에 data를 전달하여 저장하고 그 값을 받음

client



Request
form submit()

Response

server



Data에 저장 후
data 전달

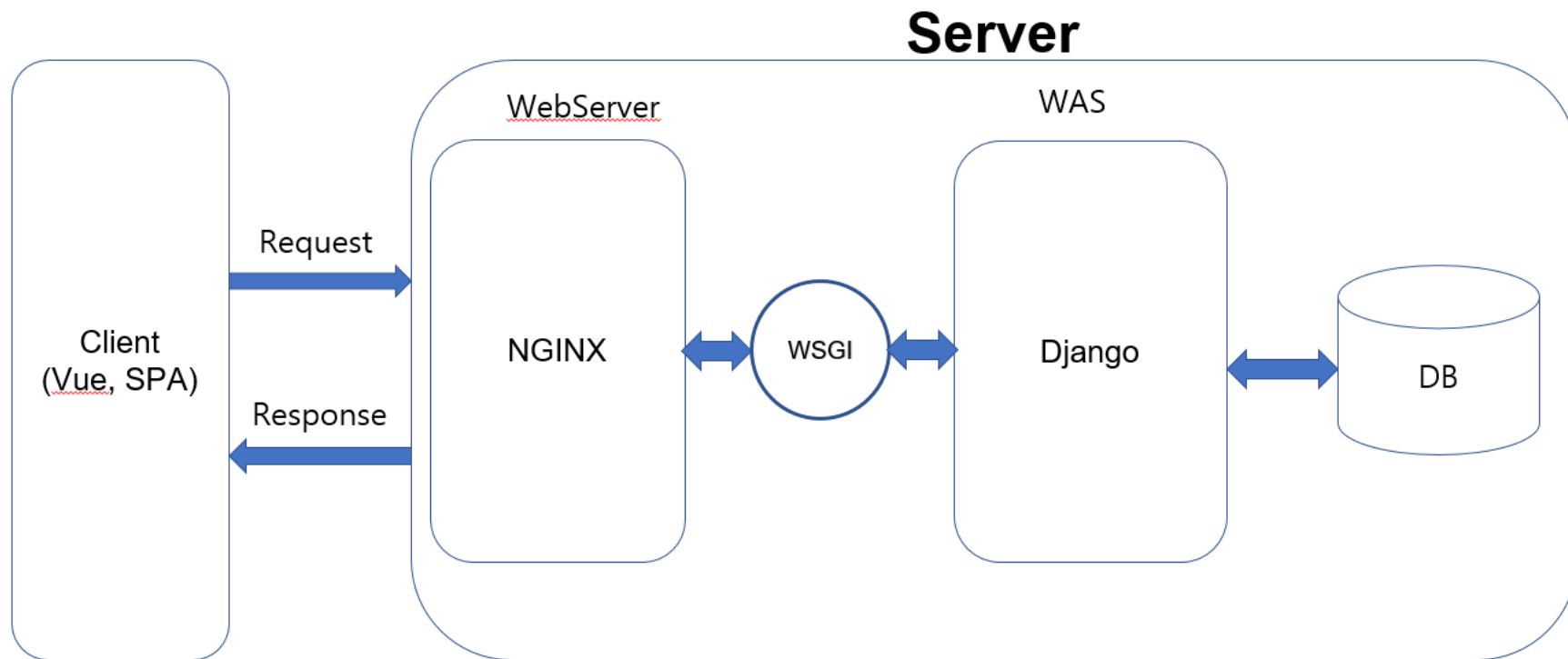
웹서버와 웹애플리케이션 서버

■ 웹서버(WEB Server)

- 클라이언트로부터 http 요청을 받아 HTML, CSS, JS, IMAGE 같은 정적 페이지를 반환
- WEB Server : Apache, NGINX

■ 웹애플리케이션 서버(WAS)

- 동적인 콘텐츠를 반환
- DB를 조회해서 데이터를 넘겨 주거나 다양한 서버 로직들을 처리해 반환
- WAS : Flask, Django



웹 개발 프레임워크 : Flask, Django

■ Flask

- 오픈소스, Django보단 상대적으로 사용자가 적음
- Django의 1/10수준으로 가벼움
- 매우 가볍고 심플한 Framework를 지향하는 점이 특징(Micro framework)
- Flask는 기본 기능 제공에 다양한 확장 모듈을 이용할 수 있는 구조여서 자유도가 높음
- Flask에는 DB ORM 구조가 따로 존재 하지 않음. 개발자가 원한다면 ORM 지원 패키지를 선택해서 사용하면 된다. (보통 SQLAlchemy 를 사용한다)
- REST API 서버처럼 요청과 응답이 매우 확정적인 경우에는 가볍고 군더더기 없는 Flask 개발이 더 효율적



■ Django

- 오픈소스, flask보다 사용자가 더 많아서 자료 찾기가 편리함
- flask보다 약 10배 많은 코드 라인으로 개발해서 더 무거움(full stack web framework)
- 프레임워크가 복잡하지만 틀에 맞추면 쉽게 큰 프로젝트도 가능(자유도가 적음)
- RDBMS와의 상호 작용을 완전히 지원하는 기본 내장 ORM과 함께 제공
ORM은 마이그레이션 생성 및 관리도 지원
- 유지 보수하기편함
- Django는 자동으로 관리자 화면을 구성 (Admin 페이지)



Flask 설치

■ Windows

- `md myproject`
- `cd myproject`
- `python -m venv venv`
- `venv\Scripts\activate`
- `pip install Flask`

■ macOS/Linux

- `mkdir myproject`
- `cd myproject`
- `python3 -m venv venv`
- `source venv/bin/activate`
- `pip install Flask`

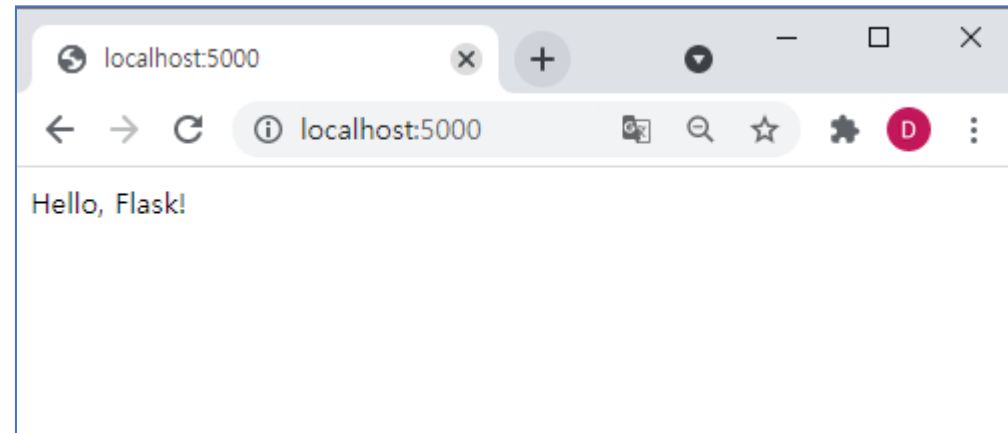
Flask - 실행

■ hello.py

```
1 from flask import Flask
2
3 app = Flask(__name__)
4
5 @app.route('/index')
6 @app.route('/')
7 def hello():
8     return 'Hello, Flask!'
```

■ Windows

- set FLASK_APP=hello.py
- set FLASK_ENV=development
- set FLASK_ENV=production
- flask run
- 웹브라우저에서 <http://127.0.0.1:5000/> 접속



■ macOS/Linux

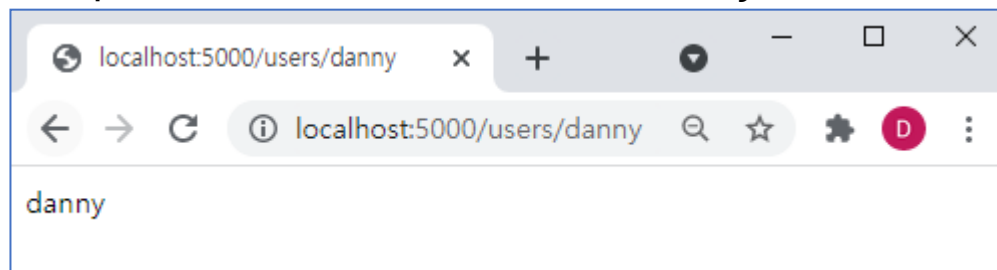
- export FLASK_APP=hello.py
- export FLASK_ENV=development
- set FLASK_ENV=production
- flask run
- 웹브라우저에서 <http://127.0.0.1:5000/> 접속

Flask - 파라미터 (parameter)

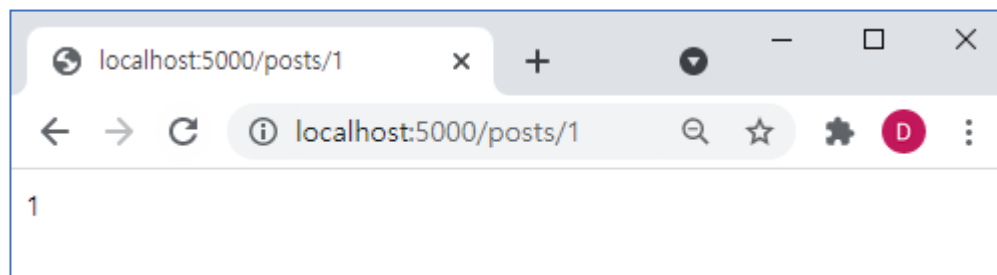
hello.py

```
1 from flask import Flask
2
3 app = Flask(__name__)
4
5 @app.route('/index')
6 @app.route('/')
7 def hello():
8     return 'Hello, Flask!'
9
10 @app.route('/users/<username>')
11 def get_user(username):
12     return username
13
14 @app.route('/posts/<int:post_id>')
15 def get_post(post_id):
16     return str(post_id)
17
18 @app.route('/uuid/<uuid:uuid>')
19 def get_uuid(uuid):
20     return str(uuid)
```

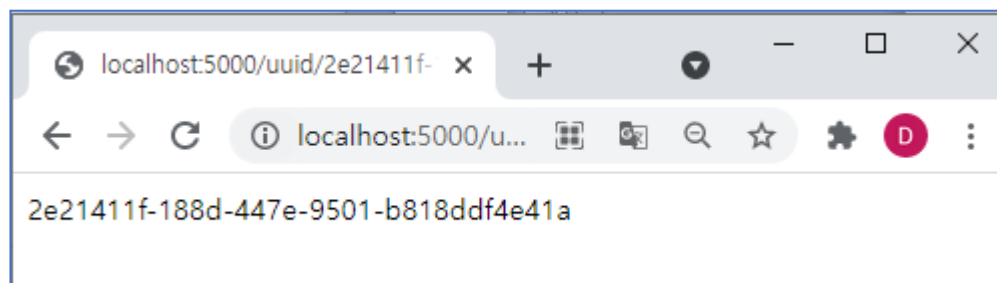
http://localhost:5000/users/danny



http://localhost:5000/posts/1



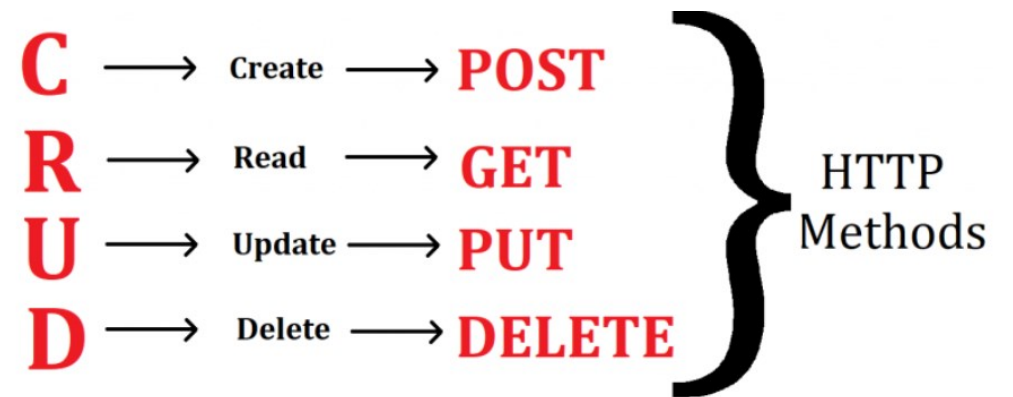
http://localhost:5000/uuid/2e21411f-188d-447e-9501-b818ddf4e41a



Flask - 메서드(method)

login.py

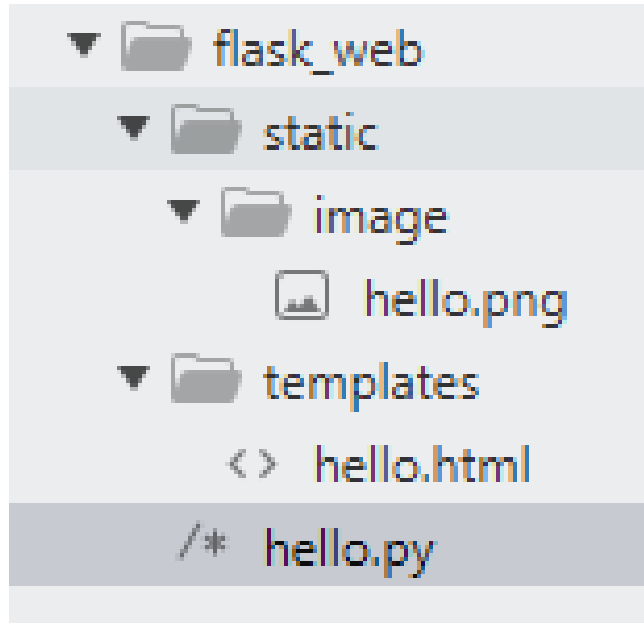
```
1 from flask import Flask, request
2
3 app = Flask(__name__)
4
5 @app.route('/login', methods=['GET', 'POST'])
6 def login():
7     if request.method == 'POST':
8         return do_login()
9     else:
10        return show_login_form()
```



HTTP Method

Flask - 정적파일과 템플릿

■ 디렉토리 구조

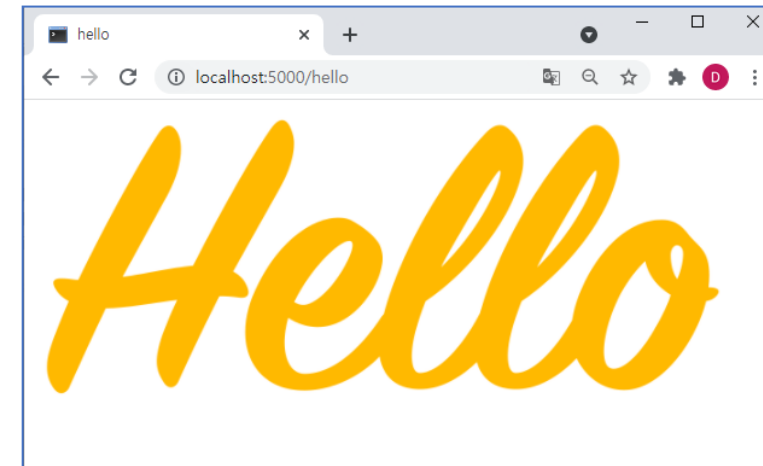


■ hello.py

```
1 from flask import Flask, render_template
2
3 app = Flask(__name__, static_folder='static', template_folder='templates')
4
5 @app.route('/hello')
6 def hello():
7     return render_template('hello.html')
```

■ hello.html

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>hello</title>
5 </head>
6 <body>
7     
8 </body>
9 </html>
```



Flask - 템플릿엔진

■ 디렉토리 구조

```
▼ flask_web_te
  ► __pycache__
  ▼ static
    ► image
  ▼ templates
    ▼ layout
      <> base.html
      <> hello.html
    /* hello.py
```

■ hello.py

```
1 from flask import Flask, render_template
2
3 app = Flask(__name__, static_folder='static', template_folder='templates')
4
5 @app.route('/hello')
6 @app.route('/hello/<name>')
7 def hello(name=None):
8     return render_template('hello.html', name=name)
```

■ base.html

```
1 <!DOCTYPE html>
2 <html>
3 ▼ <head>
4     {% block head %}
5     {% endblock %}
6 </head>
7 ▼ <body>
8     {% block body %}
9     {% endblock %}
10 </body>
11 </html>
```

■ hello.html

```
1 {% extends 'layout/base.html' %}
2
3 {% block head %}
4     <title>hello</title>
5 {% endblock %}
6
7 {% block body %}
8     {% if name %}
9         <h1>Hello, {{ name }}!</h1>
10     {% else %}
11         <h1>Hello, World!</h1>
12     {% endif %}
13 {% endblock %}
```

TesseractOCR 설치

■ Windows

- 설치파일 다운로드 : <https://github.com/UB-Mannheim/tesseract/wiki>
- 참조 : <https://turtle-dennis.tistory.com/29>

■ macOS

- brew install tesseract
- brew install tesseract-lang
- 참조 : <https://tariat.tistory.com/703>

■ Linux

- apt-get install tesseract-ocr
- 참조 : <https://linuxhint.com/install-tesseract-ocr-linux/>

■ kor.traineddata 파일 다운로드

- 다운로드 : <https://github.com/tesseract-ocr/tessdata/> 에서 kor.traineddata 다운로드
<https://github.com/tesseract-ocr/tessdata/blob/main/kor.traineddata>
- Windows : C:\Program Files\Tesseract-OCR\tessdata\ 디렉토리에 복사
- MacOS/Linux : /usr/local/Cellar/tesseract/4.1.1/share/tessdata/ 디렉토리에 복사
cp Downloads/kor.traineddata /usr/local/Cellar/tesseract/4.1.1/share/tessdata/.

백엔드 개발 (Django 웹개발프레임워크)

1. 파이썬 설치 : <https://www.python.org/downloads/>

2. 웹개발프레임워크 Django 및 패키지 설치

- pip install django
- pip install djangorestframework
- pip install drf-yasg
- pip install django-import-export
- pip install django-cors-headers
- pip install tensorflow

3. 프로젝트 생성 : <https://docs.djangoproject.com/ko/4.0/intro/tutorial01/>

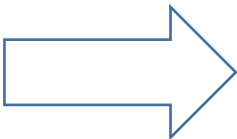
- django-admin startproject server

4. 데이터베이스 생성

- cd server
- python manage.py migrate
- python manage.py createsuperuser

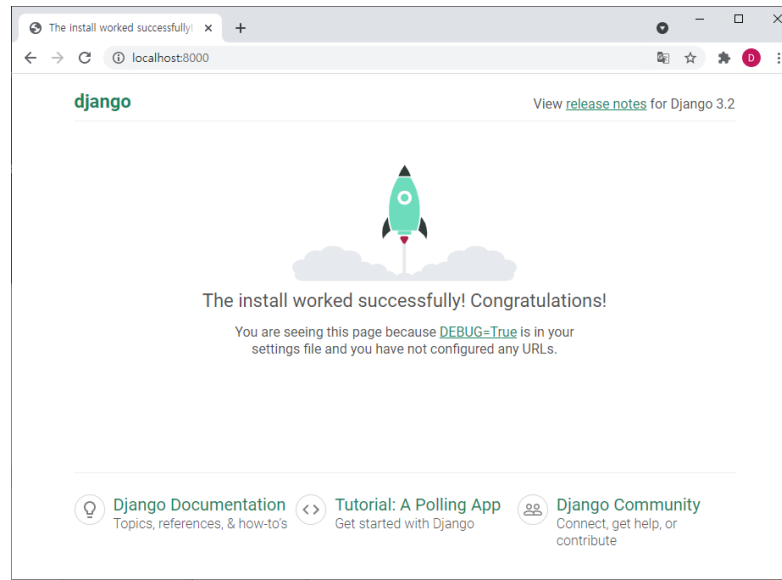
5. 서버 프로그램 실행

- python manage.py runserver
- <http://localhost:8000/> 접속 확인



장고걸스 튜토리얼 (Django Girls Tutorial)

<https://tutorial.djangogirls.org/ko/> 참고



프론트엔드 개발 (Vue.JS)

1. Node.js 설치 : <https://nodejs.org/ko/download/>

- node -v
- npm -v

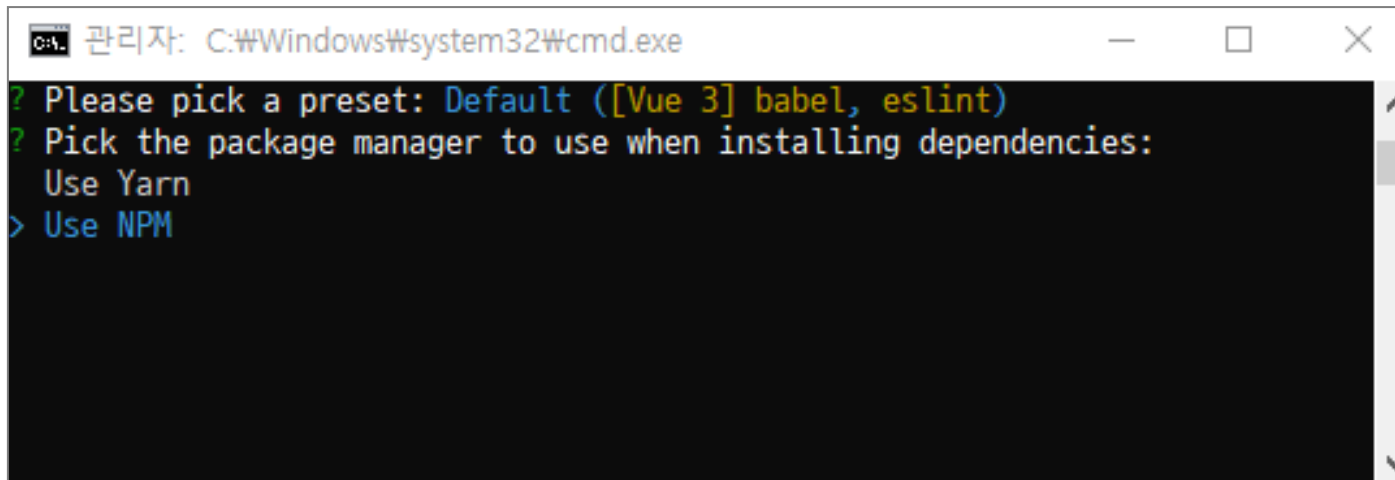
2. Vue.js(프론트엔드 개발 프레임워크) 설치 : <https://kr.vuejs.org/v2/guide/index.html>

- npm install -g @vue/cli

3. Vue 프로젝트 생성

- vue create frontend

Use NPM 선택



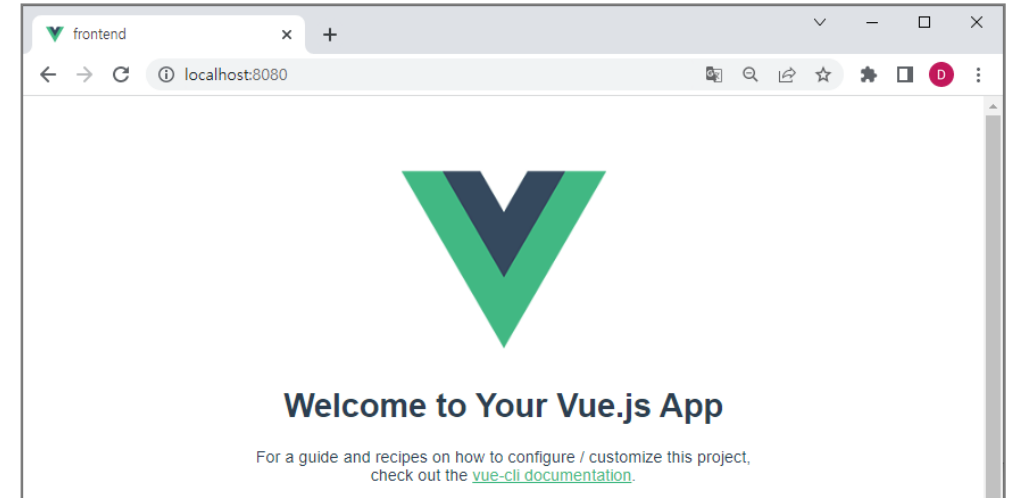
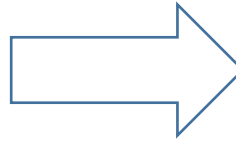
```
C:\> 관리자: C:\Windows\system32\cmd.exe

? Please pick a preset: Default ([Vue 3] babel, eslint)
? Pick the package manager to use when installing dependencies:
  Use Yarn
> Use NPM
```

프론트엔드 개발 (Vue.js)

4. 프론트엔드 실행

- cd frontend
- npm run serve
- <http://localhost:8080/> 접속확인



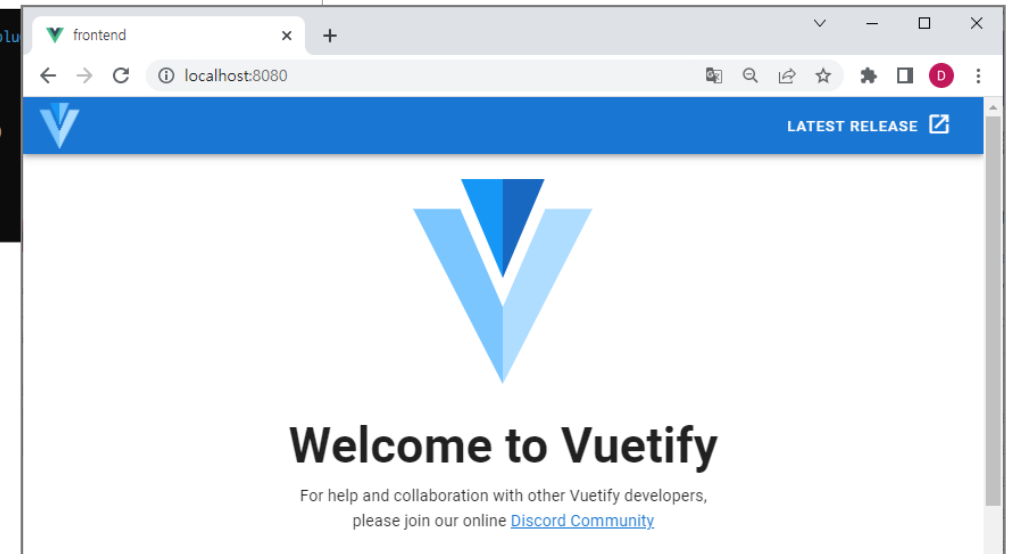
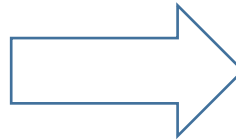
5. Vuetify(Vue UI 라이브러리) 설치/실행 : <https://vuetifyjs.com/en/getting-started/installation>

- vue add vuetify

Vuetify 2- Vue CLI(recommended) 선택

- npm install vuetify
- npm run serve
- <http://localhost:8080/> 접속 확인

```
npm install vue-cli-plugin-vuetify
Run 'npm audit' for details.
Successfully installed plugin: vue-cli-plu
Choose a preset: (Use arrow keys)
Vuetify 2 - Configure Vue CLI (advanced)
> Vuetify 2 - Vue CLI (recommended)
Vuetify 2 - Prototype (rapid development)
Vuetify 3 - Vite (preview)
Vuetify 3 - Vue CLI (preview)
```



■ Javascript 패키지 설치

- npm install babel-eslint vue-router@3 axios vue-sweetalert2 vue-echarts material-design-icons-iconfont

Vision AI 웹서비스 개발

1. Git 프로그램 설치

- <https://git-scm.com/downloads>

2. 백엔드 서버 소스 설치

- `git clone https://github.com/kgpark88/mserver`

3. 파이썬 가상환경 생성 및 실행

- `python -m venv venv`
- `venv\Scripts\activate` (windows) `source venv/bin/activate` (Linux, macOS)

4. 파이썬 패키지 설치

- `pip install django`
- `pip install pandas`
- `pip install matplotlib`
- `pip install tensorflow`
- `pip install django-import-export`
- `pip install django-cors-headers`
- `pip install djangorestframework`
- `pip install django-rest-swagger`
- `pip install drf-yasg`
- `pip install django-crispy-forms`
- `pip install opencv-python`
- `pip install Pillow`
- `pip install pytesseract`
- `pip install git+https://github.com/haven-jeon/PyKoSpacing.git`
- `pip install git+https://github.com/ssut/py-hanspell.git`
- `pip install tensorflow`
- `pip install tensorflow_hub`

Vision AI 웹서비스 개발

5. 테이블 생성

- `cd mserver`
- `python manage.py makemigrations ocr`
- `python manage.py migrate`

6. 데이터베이스 관리자 계정 생성

- `python manage.py createsuperuser`

7. 백엔드 실행

- `python manage.py runserver`

8. 프론트엔드 소스 설치

- `git clone https://github.com/kgpark88/mfrontend`
- `cd mfrontend`

9. Javascript 패키지 설치

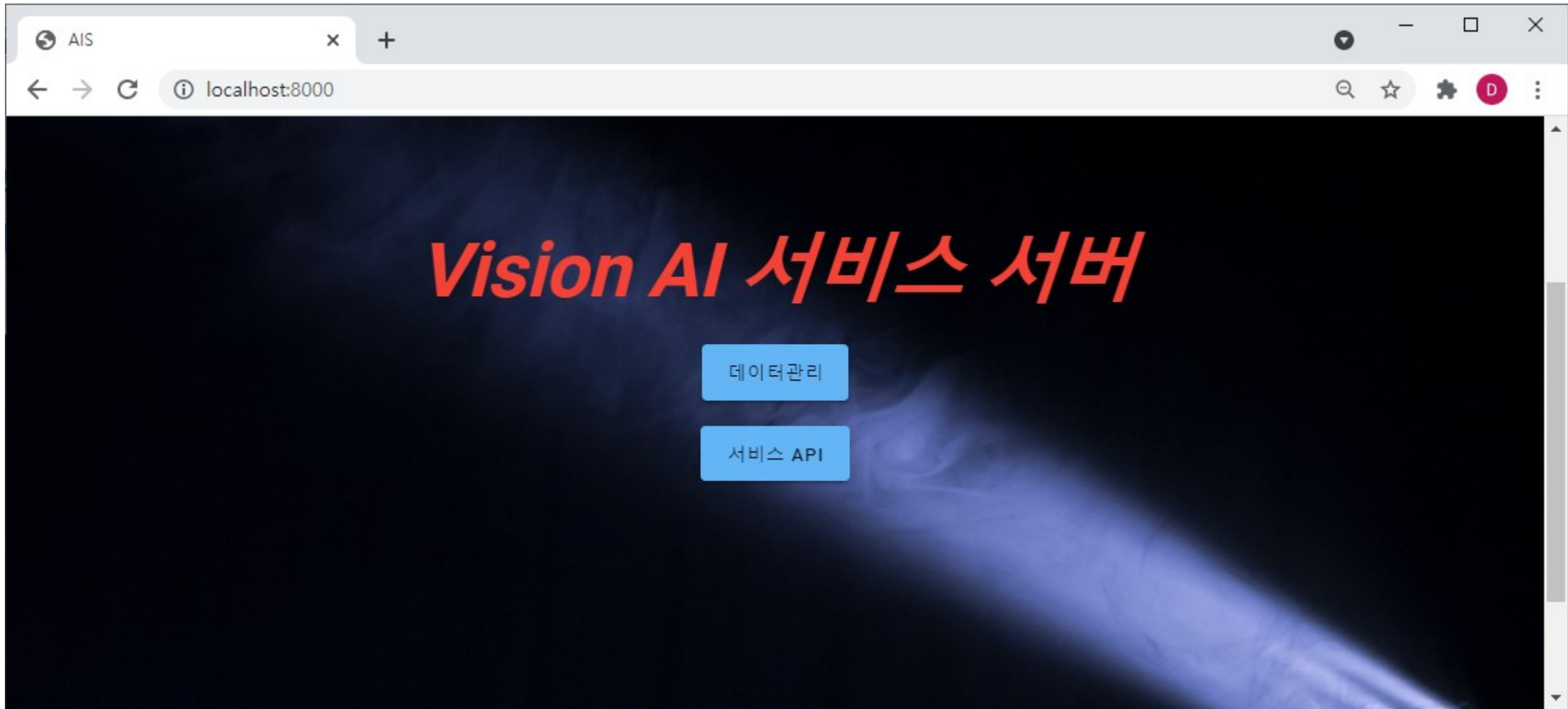
- `npm install`

10. 프론트엔드 실행

- `npm run serve`

Vision AI 웹서비스 서버

<http://localhost:8000>



Vision AI 웹서비스 데이터 관리

http://localhost:8000/api

변경할 텍스트 인식 선택 | AIS P x +

localhost:8000/admin/ocr/ocrtext/

Admin

환영합니다, AIS. [사이트 보기](#) / [비밀번호 변경](#) / [로그아웃](#)

홈 > Optical Character Recognition > 텍스트 인식

OPTICAL CHARACTER RECOGNITION

텍스트 인식 + 추가

사이트

사이트들 + 추가

인증 및 권한

그룹 + 추가

사용자(들) + 추가

인증 토큰

Tokens + 추가

변경할 텍스트 인식 선택

가져오기 내보내기 텍스트 인식 추가 +

Q

검색

액션:

 실행 2 중 아무것도 선택되지 않았습니다.

<input type="checkbox"/>	ID	이미지 파일	텍스트 인식결과	생성일시	수정일시
<input type="checkbox"/>	2	english_x7Lm9uQ.png	A Dream Within A Dream by Edgar Allan Poe Take this kiss upon the brow! And, in parting from you now, 'Thus much let me avow-- 'You are not wrong, who deem That my days have been a dream; 'Yet if hope has flown away In a night, or in a day, In a vision, or in none, Is it therefore the less gone? All that we see or seem Is but a dream within a dream. ♫	2021년 10월 24일 6:15 오후	2021년 10월 24일 6:15 오후
<input type="checkbox"/>	1	hanguL_Mwutuq0.png	소년 여기 저기 서 HES BS 슬픈 가을이 SH 떨어진다. 단풍 잎 떨어져 온 자리 마다 SS 마려해 놓고 나뭇가지 위에 하늘이 펼쳐 있다. 가만히 하늘을 올려다 보려면 눈썹에 파란 물감이 든다. 두 손으로 따뜻한 SS 쓸어 보면 손바닥에도 파란 Sz 이 묻어 난다. 다시 손바닥을 올려다 본다. 손금에는 맑은 강물이 흐르고, 맑은 강물이 흐르고, 강물 속에는	2021년 10월 24일 6:14 오후	2021년 10월 24일 6:14 오후

Vision AI 웹서비스 API

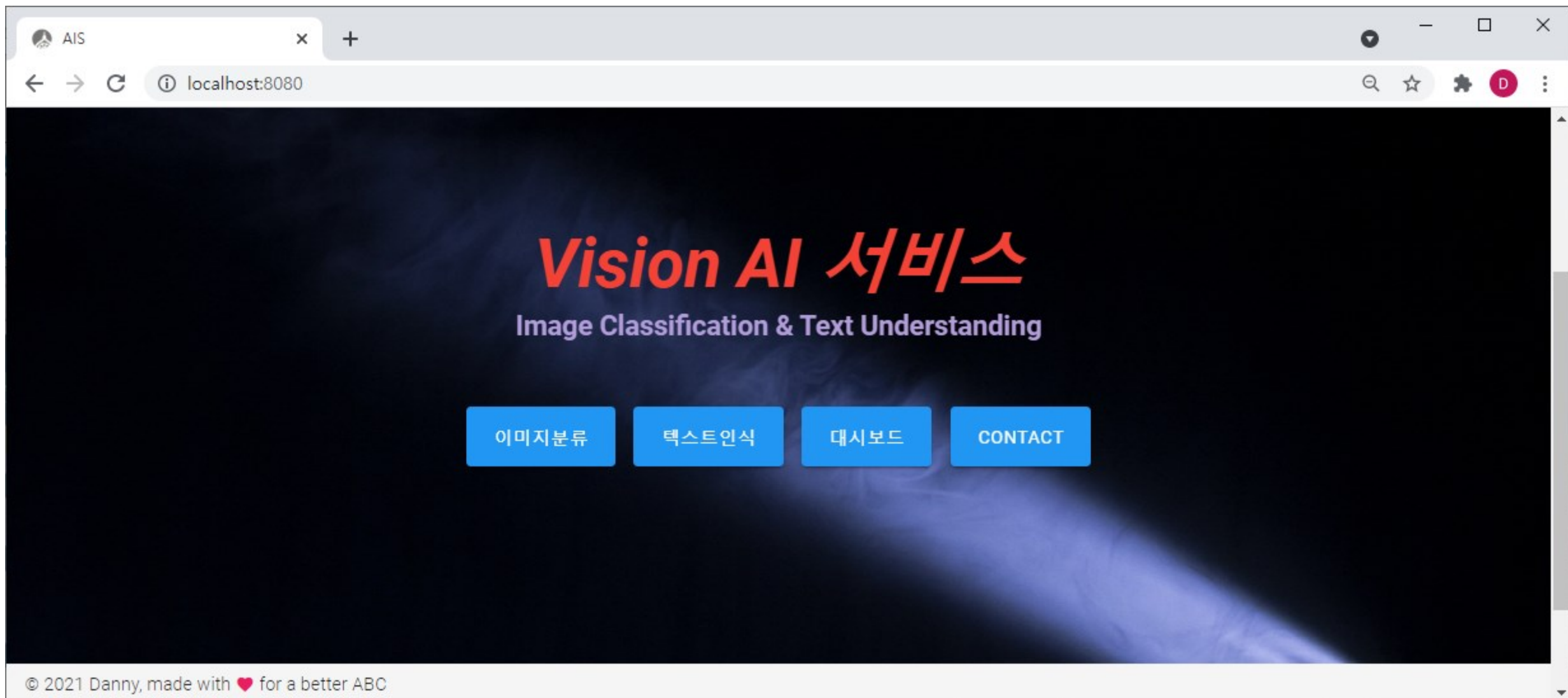
<http://localhost:8000/api>

The screenshot shows a web browser window with the title "AIS API". The address bar displays "localhost:8000/api". The main content area features a "Filter by tag" input field. Below it, the "ocr" category is selected, indicated by a dropdown arrow. The selected endpoint is "POST /ocr/" with a lock icon and the label "ocr_create". Under the "Parameters" section, it states "No parameters" and includes a red "Cancel" button. A large blue "Execute" button is positioned below the parameters. The "Responses" section shows a dropdown menu for "Response content type" set to "application/json". At the bottom, a table lists the response codes:

Code	Description
201	

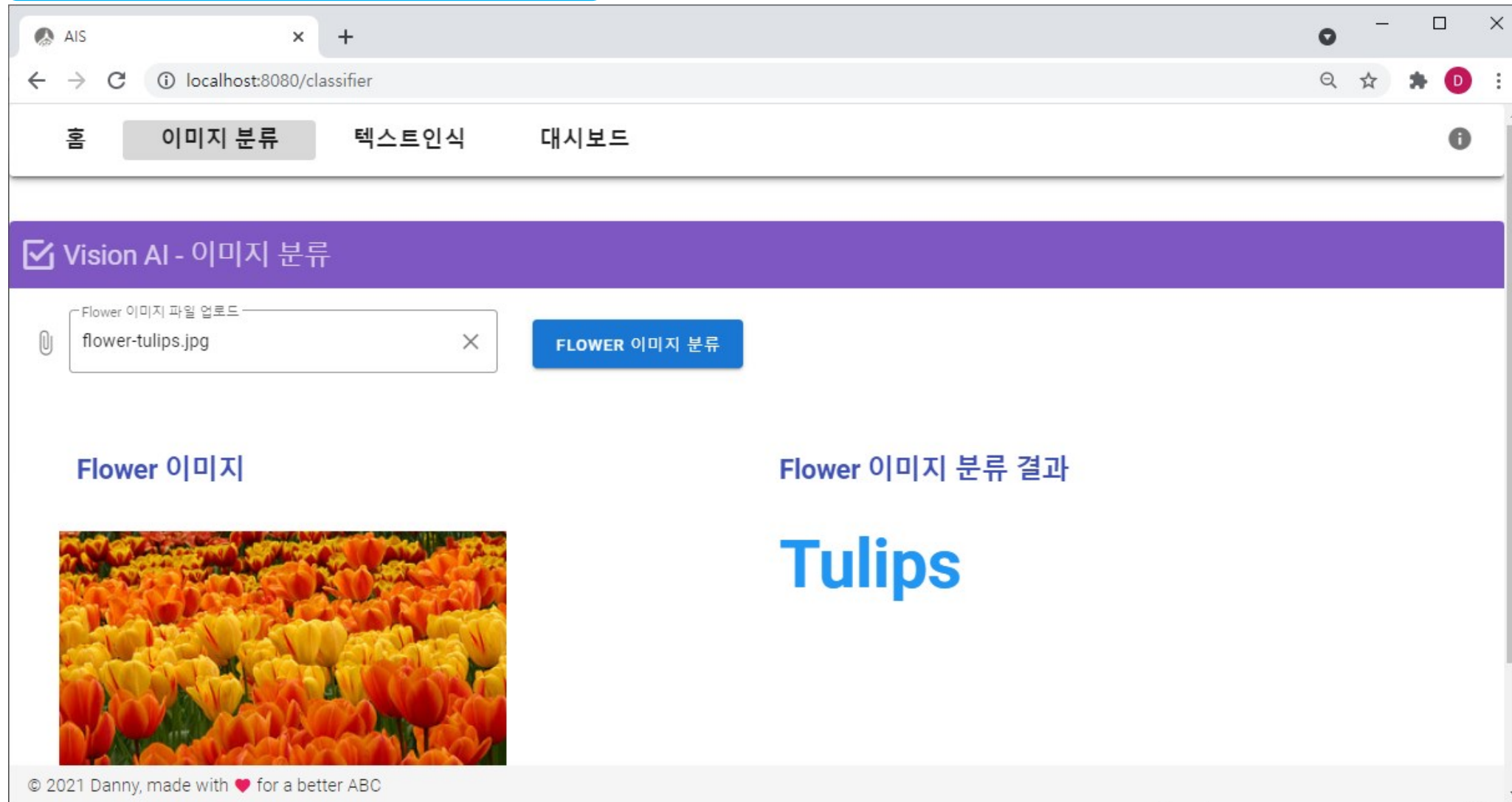
Vision AI 프론트엔드

http://localhost:8080



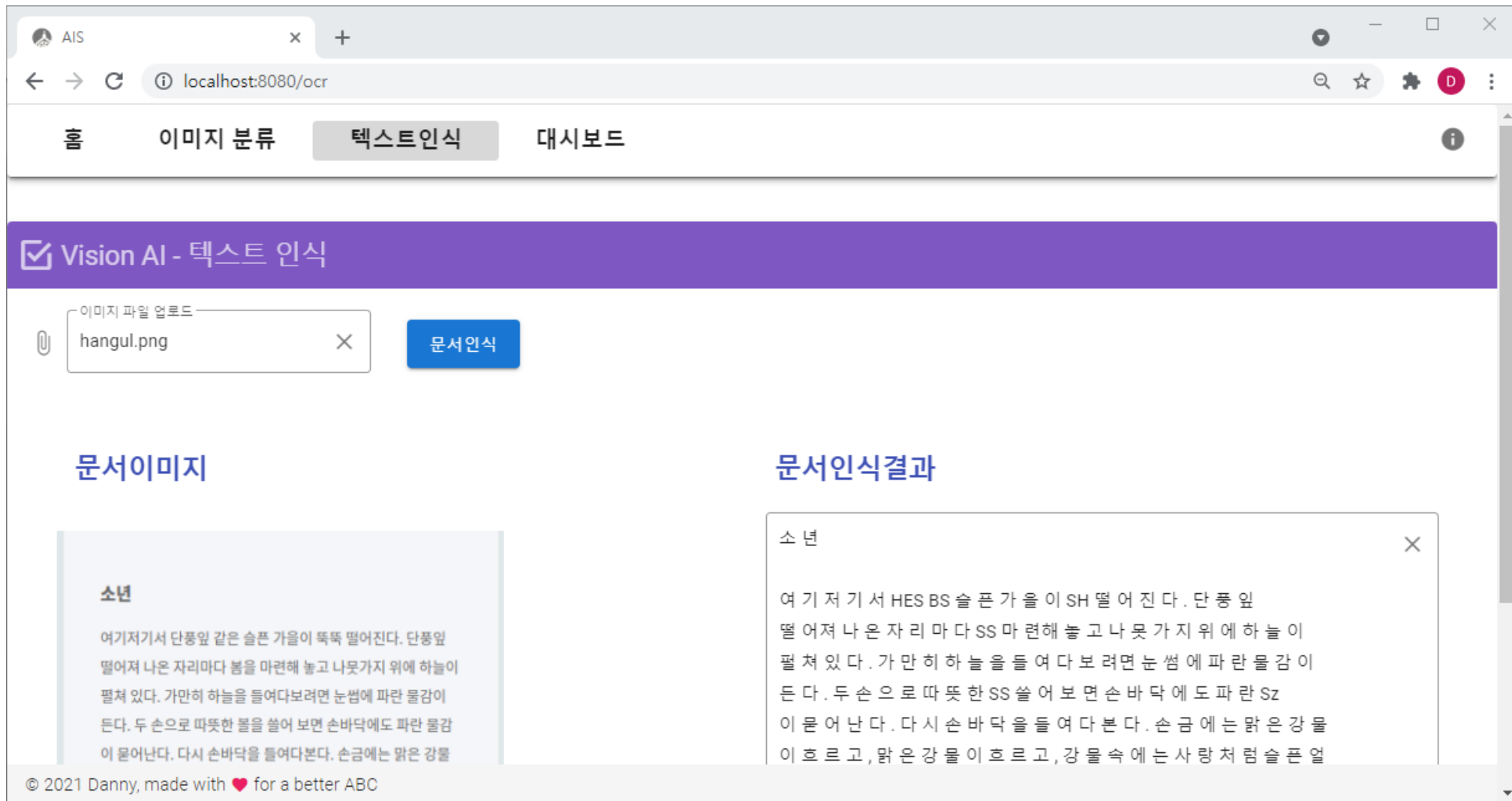
Vision AI 프론트엔드 - 이미지분류

http://localhost:8080



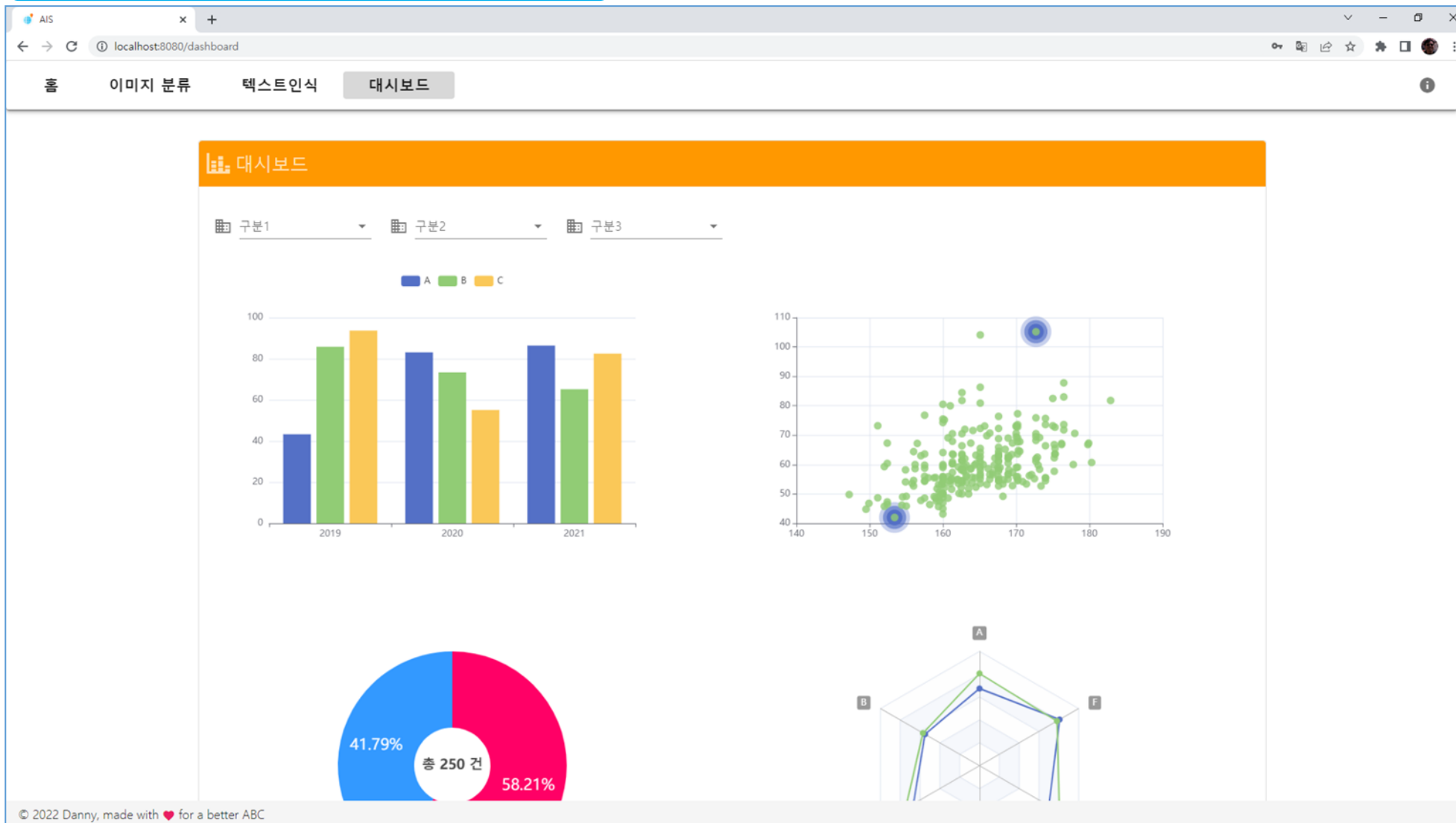
Vision AI 프론트엔드 - 텍스트인식

http://localhost:8080



Vision AI 프론트엔드 - 대시보드

<http://localhost:8080>



참고

Google Cloud Vision AI

<https://cloud.google.com/vision/docs/ocr?hl=ko>

Amazon Textract

<https://aws.amazon.com/ko/textract/>

 CLOVA OCR











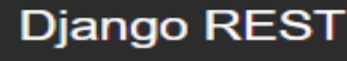








<https://www.ncloud.com/product/aiService/ocr>

kakao Vision API

<https://vision-api.kakao.com/#ocr>



개발 프레임워크

프론트엔드(UI)	     <p>프론트엔드 개발 프레임워크</p> <p>차트 라이브러리</p>
AI 모델링	   <p>딥러닝 라이브러리</p> <p>머신러닝 라이브러리</p>
데이터 분석	   <p>수치연산 라이브러리</p> <p>과학 라이브러리</p> <p>데이터분석 라이브러리</p>
API 태스크관리	   <p>REST API</p> <p>분산 태스크 큐</p> <p>메시지 브로커</p>
Web 서버 WAS DB	   <p>웹서버</p> <p>WSGI HTTP 서버</p> <p>데이터베이스</p>
개발언어	   <p>개발언어</p> <p>웹개발 프레임워크</p>





수고하셨습니다. 감사합니다.