Fast campus × upstage

Upstage AI Lab

MLOps project | 2025. 06. 10(화)

- 3조 -

목차

01

팀 소개

팀장/팀원 소개

협업 방식

이름	이승민	문국현	문진숙	홍정민	조선미
역할	팀장	팀원	팀원	팀원	팀원
전공	비고	인테리어디자인	법학, 경영학	영어 영문	교육학
관심사	데이터 엔지니어	MCP, LLM, CV MLops	데이터 분석, AI를 활용한 영상제작	해외 취업	에듀테크

프로젝트 협업 방식

: MLOps Project

경진대회 이후 새로운 협업 방식 : 진도가 빠른 팀원의 리드로 진행 상황 체크해 가면서 협업함

역할 분담 : 깃허브에 브랜치를 나눠서 팀원 각자의 역할 분담과 협업
데이터 전처리, 모델 학습, 모델 서빙, 배포, 시각화, 테스트

애자일 방법론 & Fail Fast 활용 : 프로젝트를 작은 단위로 나누어 반복적으로 진행하고 테스트 결과를 반영하여 개선하는 방식을 사용 & 하루만에 간략한 버전의 파이프라인을 만들어 돌려보고 데이터 및 기능을 추가해가면서 고도화.

모든 팀원의 역량 향상을 위한 지원과 협업 : MLOps를 처음 접하는 팀원들을 위해 MLOps 학습 파이프라인 Docker 실행 가이드 등 상세한 튜토리얼을 작성해서 노션에 공유하고, 팀원이 작성한 학습 블로그(개념 정리) 공유하여 전체 팀원이 프로젝트에 참여할 수 있도록 함.

협업 진행 횟수 및 일정 : 매일 1회이상 정기적인 줌 미팅 및 슬랙, 노션 활용함

문제 해결 방법 : 문제 해결을 위해 협의하는 시간을 많이 가졌고, 멘토님과의 상담 시간을 통해 해결안을 찾음

< 협업을 위한 상세한 튜토리얼 작성 (예시) >

1. MLOps 팀원을 위한 Docker 이미지 사용 튜토리얼

1. Docker Hub 계정 만들기
1. https://hub.docker.com/ 접속
2. Sign Up (회원가입) 클릭
3. 아이디, 비밀번호 입력 → 가입 완료

2. Docker 설치
Mac 사용자:
- https://www.docker.com/products/docker-desktop/ 접속
- "Mac with Apple chip" 또는 "Mac with Intel chip" 중 선택하여 다운로드 및 설치

Windows 사용자:
- 같은 사이트에서 "Windows" 선택 → 다운로드
- 설치 후 컴퓨터 재시작 필요할 수 있음

3. Docker Desktop 실행 확인
- 설치 후 "Docker Desktop" 실행
- 상단 메뉴바에 고래 아이콘이 보이면 정상 동작 중

4. 터미널에서 Docker 명령어 실행 환경 준비
- **Mac:** `터미널` 앱 열기
- **Windows:** `PowerShell` 또는 `WSL` 혹은 `CMD` 창 열기

5. Docker Hub 로그인 (터미널)

```
docker login
```

- Username: 본인의 Docker ID
- Password: Docker 비밀번호 또는 Access Token (보안 강화를 위해 Access Token 사용 권장)

6. API 서버 이미지 다운로드 및 실행
1) 이미지 다운로드:

```
docker pull welovecherry/weather-api:v1
# 약 700MB 이미지가 내려받아집니다
```

2) 컨테이너 실행:
8000 포트가 비어있을 경우:

02

프로젝트 개요

목표 수립

기술 스택 / 아키텍처 설계

프로젝트 목표 수립

: MLOps Project | 목표 및 주요 작업

주제

공공 기상 데이터를 활용한 서울 지역 날씨 7일 예측 + 의상 추천 자동화 서비스

목표

목표

* 기상청의 공공데이터를 활용하여 서울 지역의 시간별 **날씨(기온, 습도 등)를 예측**
* **MLOps 파이프라인을 구축**하여 데이터 수집->모델 학습->예측 결과 저장까지 전 과정을 자동화
* 예측된 날씨 데이터를 바탕으로 **사용자 맞춤형 옷차림을 추천하는 API 서비스 구현**

주요 작업

* 기상청 날씨 데이터를 수집하고 모델 학습에 맞게 전처리
* 딥러닝 LSTM (Pytorch)을 활용해 (과거 데이터 사용) 7일 예측
* MLflow로 실험 기록을 관리하고, Airflow를 통해 데이터 파이프라인 자동화
* 예측 결과를 기반으로 옷차림 추천 앱, API 엔드포인트를 구현

개요

소개 및 배경 설명

* 기온을 예측하여 옷차림을 추천해주는 앱 개발의 자동화 프로젝트

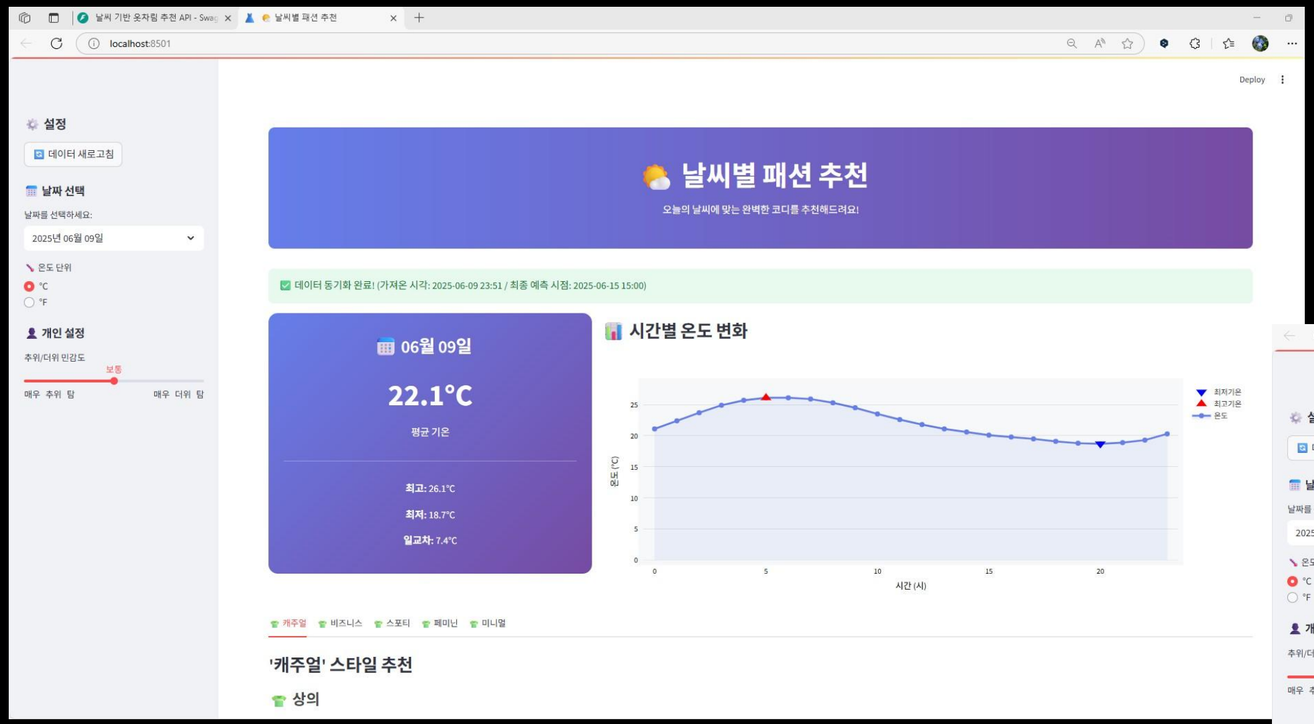
기간

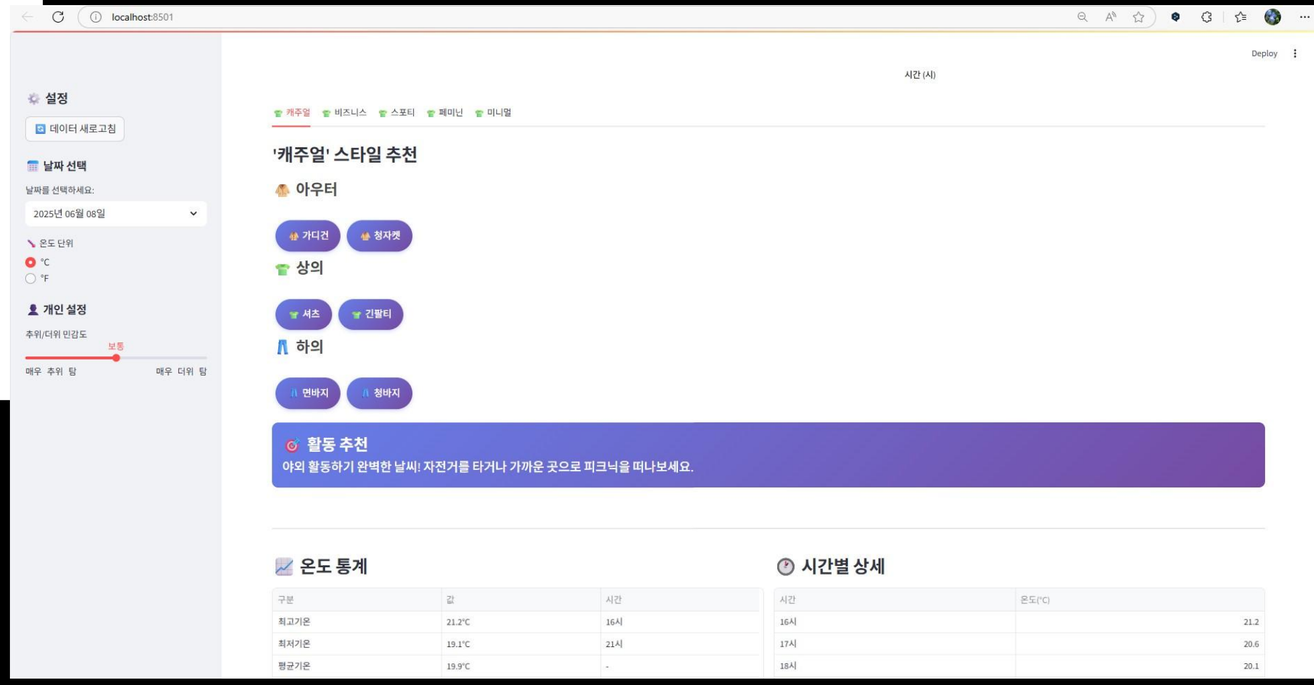
*2025. 5. 26 ~ 2025. 6.10

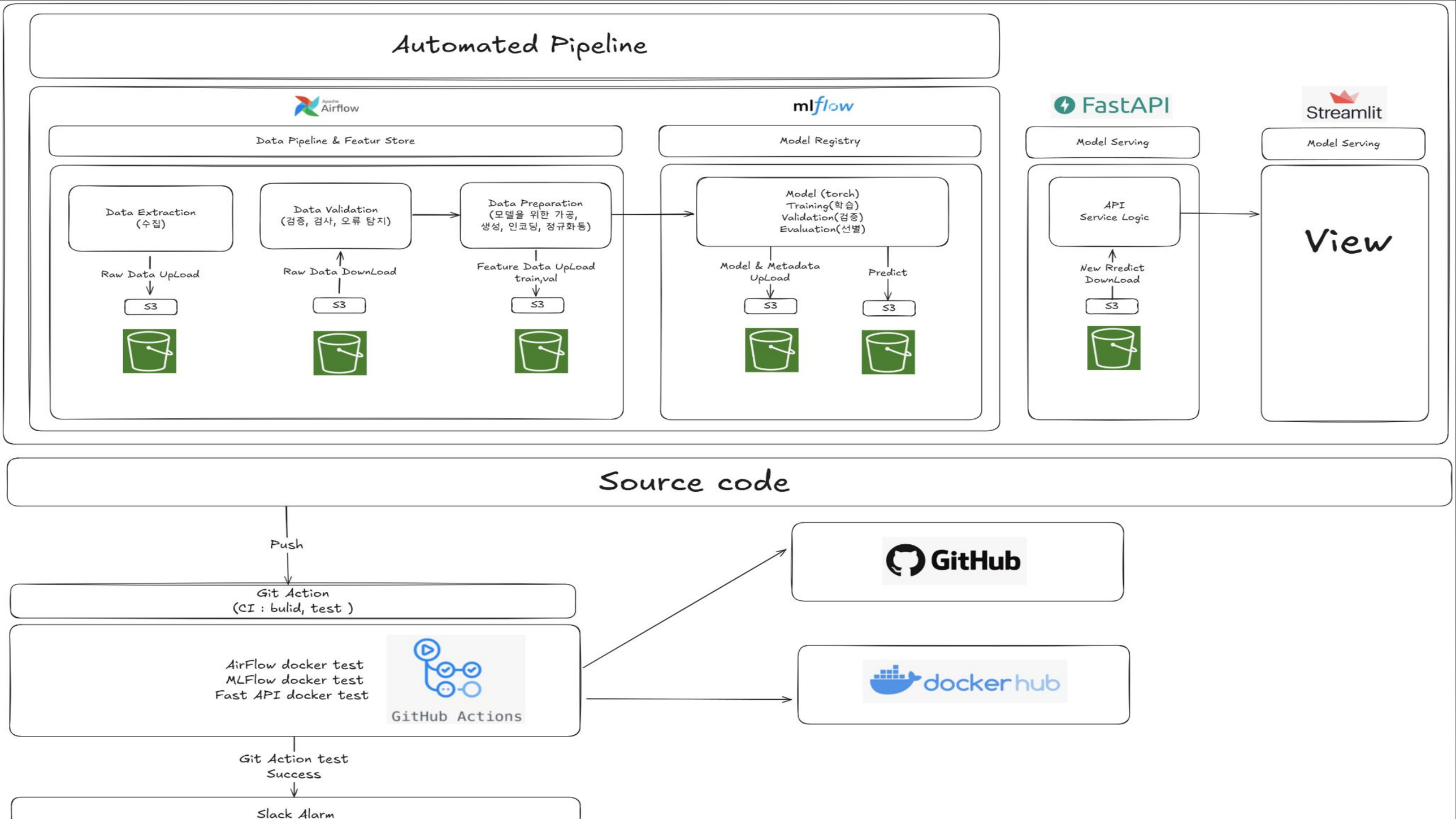
목표 수립

프로젝트 앱 데모

: MLOps Project | 시연






Automated Pipeline
Airflow
Data Pipeline & Featur Store
Data Extraction
(수집)
Raw Data UpLoad
S3
Data Validation
(검증, 검사, 오류 탐지)
Raw Data DownLoad
S3
Data Preparation
(모델을 위한 가공,
생성, 인코딩, 정규화등)
Feature Data UpLoad
train,val
S3
mlflow
Model Registry
Model (torch)
Training(학습)
Validation(검증)
Evaluation(선별)
Model & Metadata
UpLoad
S3
Predict
S3
FastAPI
Model Serving
API
Service Logic
New Rredict
DownLoad
S3
Streamlit
Model Serving
View
Source code
Push
Git Action
(CI : bulid, test)
AirFlow docker test
MLFlow docker test
Fast API docker test
GitHub Actions
GitHub
dockerhub
Git Action test
Success
Slack Alarm

Tech Task

프로젝트 개요

: MLOps Project | 기술 스택 및 아키텍처 설계

분야	사용 기술 / 도구	역할
데이터 수집	공공데이터포털 OpenAPI, Requests	지역 시간별 기온, 습도 등 날씨 데이터 수집
전처리 & 분석	Pandas, Numpy	결측치 처리, 파생변수 생성, 학습용 데이터셋 구성
모델 학습	LSTM (Pytorch)	회귀 모델을 통한 기온 예측, RMSE 평가
실험 관리	MLflow	실험 기록 및 하이퍼파라미터 /성능 추적
자동화 파이프라인	Apache Airflow	DAG 스케줄링으로 데이터 수집 및 전처리 자동화
환경	Docker	학습 파이프라인 컨테이너화, 환경 재현성 확보
스토리지	AWS S3	원시 데이터 및 모델 아티팩트 저장
API 구현	FastAPI	날씨 기반 옷차림 추천 API 구현
CI/CD 자동화	Github Actions + DockerHub + Slack(알람) EC2	코드 체크, 도커 이미지 빌드, 도커 허브 자동 푸시, 컨테이너 실행과 스모크 테스트, 슬랙 자동 알림기능

03

프로젝트 수행 절차 및 방법

데이터셋 및 데이터 처리 / 모델 개발 /
모델 배포 / MLOps 워크 플로우

프로젝트 수행 절차 및 방법

: MLOps Project | 데이터 전처리

** 데이터 수집*

```python
def fetch_weather_data(start_datetime: str, end_datetime: str) -> pd.DataFrame:…

def generate_date_ranges(start_date: datetime, end_date: datetime) -> list:…

def get_latest_weather_data() -> datetime:…

def save_to_s3(df: pd.DataFrame, year: int, month: int, day: int):…

def initialize_weather_database():…

def update_weather_database():…

def check_s3_path_exists() -> bool:…

def collect_weather_data_with_time():
    """
    S3에 weather 데이터가 없으면 초기화하고, 있으면 업데이트합니다.
    """
    if check_s3_path_exists():
        print("기존 날씨 데이터가 발견되어 업데이트를 시작합니다.")
        update_weather_database()
    else:
        print("기존 날씨 데이터가 없어 초기화를 시작합니다.")      You, last week • feat
        initialize_weather_database()

if __name__ == "__main__":
    collect_weather_data_with_time()
```

1. s3 최신 데이터 검토
2. raw 데이터 수집
3. s3 업로드

프로젝트 수행 절차 및 방법

: MLOps Project | 데이터 전처리

* 전처리

```python
class Feature_Engineering:
    def __init__(self, df=None, is_train=True):
        self.s3_bucket = os.getenv("S3_BUCKET_NAME")
        self.s3 = s3fs.S3FileSystem()


    def load_data(self, start_year=None, prefix='data/weather/raw'):···

    def missing_value(self):···


    def impossible_negative(self):···

    def spearman_test(self, target_col):···


    def kruskal_test(self, target_col, min_group_size=5):···


    def feature_selection(self, target_col):···


    def add_feature(self):···


    def split_data(self, HORIZON=168, SEQ_LEN=336):···


    def scaler(self, train_df, val_df, latest_input):···
Ctrl+L to chat, Ctrl+K to generate
    def encoding(self, train, val, latest):···


    def save_split_data(self, train, val, latest):···
```

1. s3 raw 데이터 다운
2. 결측치 & 이상치 처리
3. train,val,test 분리
4. 인코딩
5. s3 업로드

프로젝트 수행 절차 및 방법

: MLOps Project | 모델 개발

* ML으로 시작했다가 기온 예측의 정확도를 위해 **딥러닝 LSTM모델 사용 결정**

사용한 모델	모델 선택 이유
LSTM	**머신러닝에서 딥러닝으로 변경 이유** **- 머신러닝 (랜덤포레스트 , XGBoost 등)은** 미래 시점의 입력값(피처)이 필요해 하루 전체 24시간 온도를 한 번에 예측하기 어려움 **-과거 데이터만 존재하므로 ,** 미래 여러 시점을 동시에 예측하는 데 한계가 있음 **-LSTM 같은 시계열 딥러닝 모델은** 과거 여러 시간 데이터를 기반으로 시간 흐름과 패턴을 학습해 미래 24시간 온도를 한꺼번에 예측 가능 **-> 우리 목적에 맞는 '다음 날 24시간 온도 전체 예측'에는 딥러닝이 더 적합함**

프로젝트 수행 절차 및 방법

: MLflow | 모델 버전 관리

* *모델: LSTM (Pytorch)*

LSTM-Weather Provide Feedback Add Description — Share

Runs | Evaluation Experimental | Traces

metrics.rmse < 1 and params.model = "tree" | Time created | State: Active | Datasets | Sort: Best_RMSE | Columns | New run | Group by

							Metrics	Parameters			
	Run Name	Created	Duration	User	Source	Models	Best_RMSE	epochs	hidden_size	learning_rate	num_layers
	LSTM_2025-06-09_21...	14 hours ago	1.8h	mungughye...	pipeline...	LSTM-Weather v9	1.7126448...	30	128	0.0007	2
	LSTM_2025-06-09_11...	1 day ago	28.2min	mungughye...	pipeline...	LSTM-Weather v3	1.9559876...	20	128	0.001	2
	LSTM_2025-06-09_14...	21 hours ago	2.0min	tiqmf	pipeline...	LSTM-Weather v4	1.9695927...	15	128	0.001	2
	LSTM_2025-06-09_14...	21 hours ago	5.9min	tiqmf	pipeline...	LSTM-Weather v5	2.1373510...	15	128	0.001	5
	LSTM_2025-06-10_00...	11 hours ago	23.1min	mungughye...	pipeline...	LSTM-Weather v10	2.6769853...	30	128	0.0008	2
	LSTM_2025-06-09_01...	1 day ago	37.2min	mungughye...	pipeline...	LSTM-Weather v2	2.7594071...	15	192	0.0007	2

프로젝트 수행 절차 및 방법

: MLflow | 모델 버전 관리

Status	Finished
Run ID	54b040491b754a9ca4171702655cc3fe
Duration	1.8h
Datasets used	–
Tags	Training_duration: 1:49:51 Trained_at: 2025-06-09 22:57:57 training_started_at: 2025-06-09 21:08:05
Source	pipeline.py c6bbfb2
Logged models	pytorch
Registered models	LSTM-Weather v9
Registered prompts	–

Parameters (7)

Search parameters

Parameter	Value
start_year	2020
hidden_size	128
num_layers	2
dropout	0.4
learning_rate	0.0007
epochs	30
batch_size	32

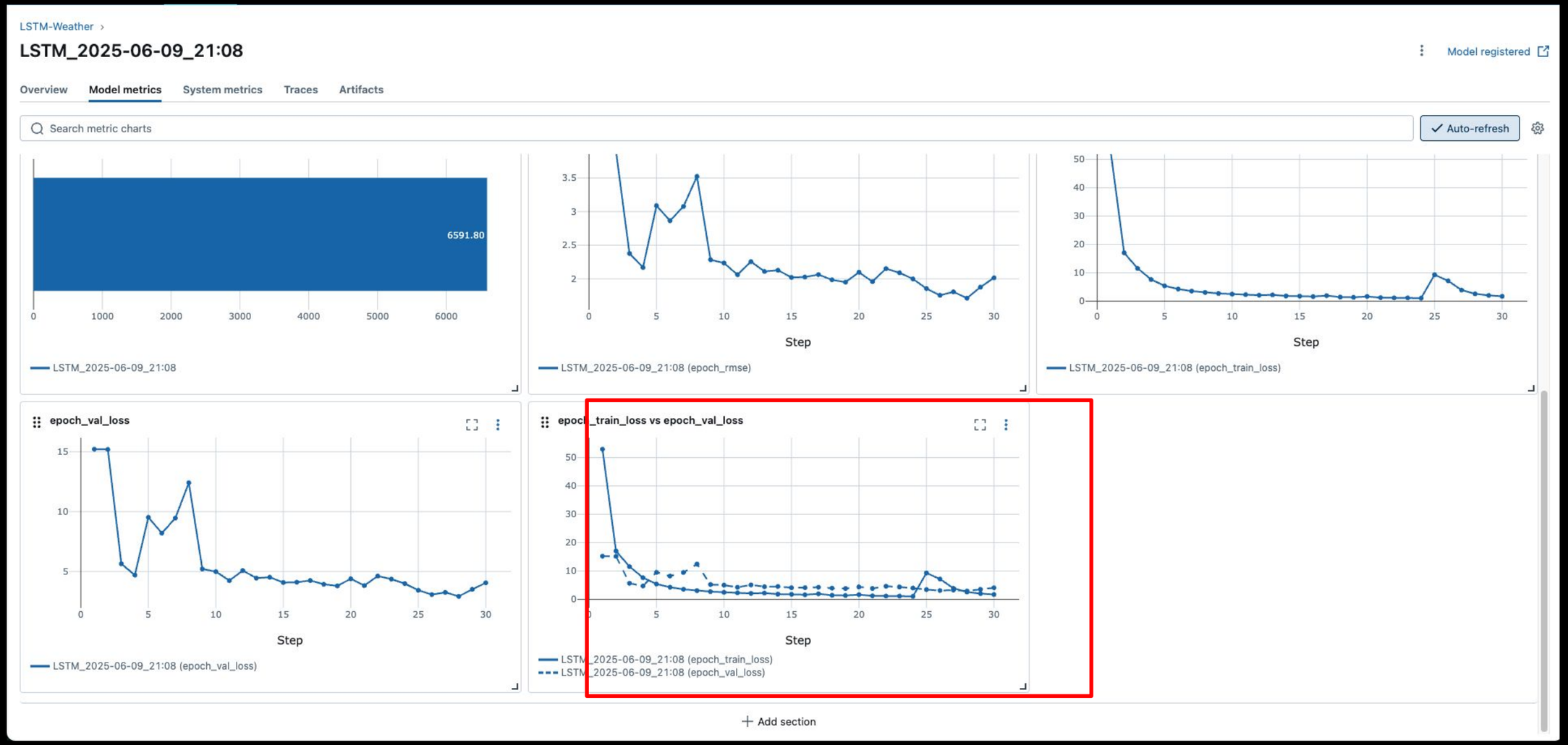
Metrics (7)

Search metrics

Metric	Value
epoch_train_loss	1.6966484159169821
epoch_val_loss	4.061673641204834
epoch_rmse	2.0153593996504533
Best_Train_loss	2.5814892731877985
Best_Val_loss	2.933152437210083
Best_RMSE	1.7126448659637277
Training_Time	6591.803446

프로젝트 수행 절차 및 방법

: MLflow | 모델 버전 관리



프로젝트 수행 절차 및 방법

: Docker | 동일한 개발 환경 구성

mlops_team > app > Dockerfile.app > ...

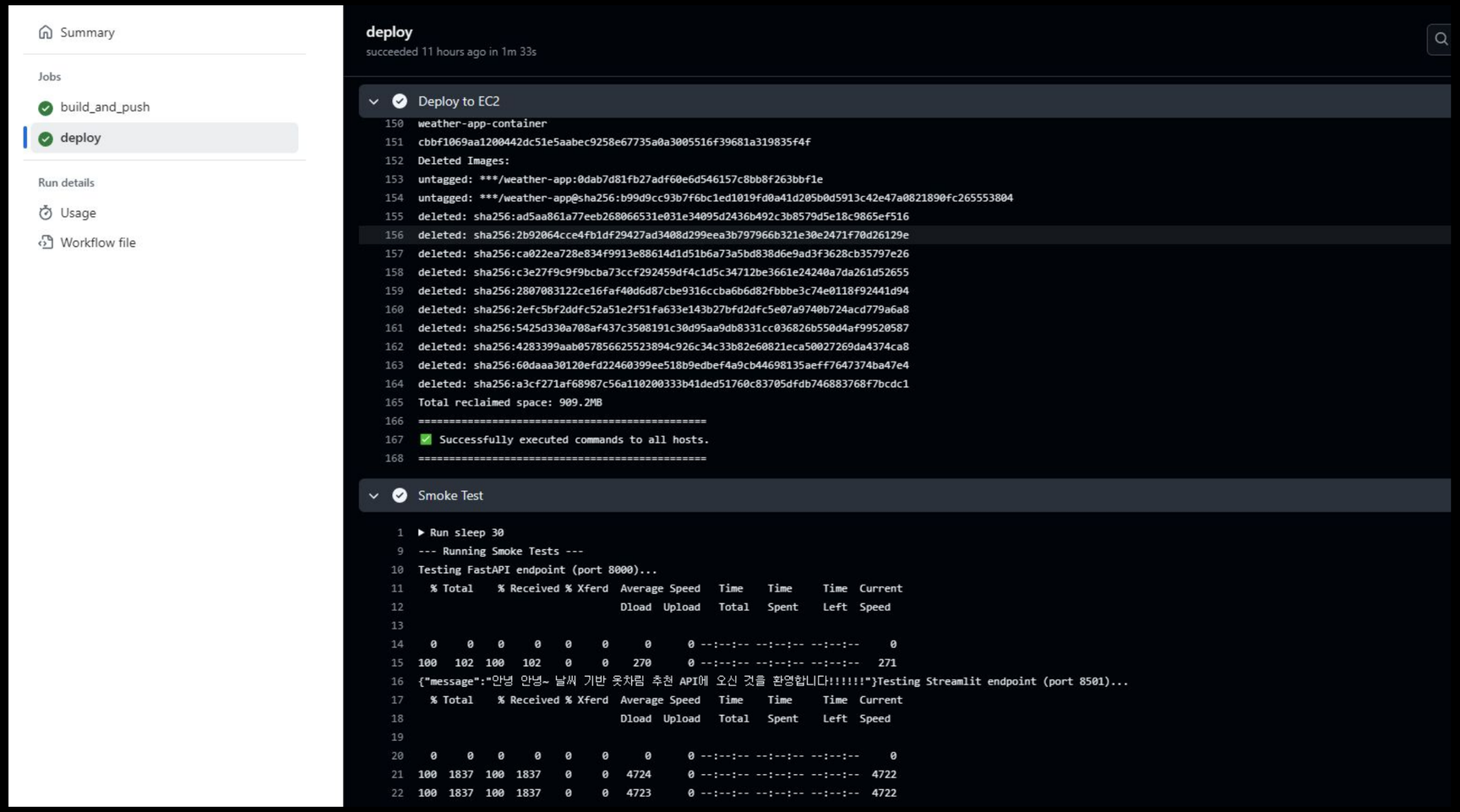
```
FROM python:3.9-slim

ENV TZ=Asia/Seoul

RUN apt-get update && \
    apt-get install -y build-essential curl git tzdata && \
    rm -rf /var/lib/apt/lists/*

WORKDIR /app

# 의존성 목록 복사
COPY mlops_team/app/requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

# 실행 스크립트 복사 및 권한 부여
COPY mlops_team/app/app.start.sh .
RUN chmod +x ./app.start.sh

# 모든 소스 코드 폴더들을 각각 복사
COPY mlops_team/src ./src
COPY mlops_team/streamlit_app ./streamlit_app
COPY mlops_team/common ./common


# 최종 실행 명령어
CMD ["bash", "app.start.sh"]
```

mlops_team > app > requirements.txt

```
# FastAPI & Uvicorn
fastapi
uvicorn[standard]

# Streamlit App
streamlit
pandas
plotly

# AWS S3 Connection
boto3
s3fs
pyarrow

# .env file
python-dotenv
```

프로젝트 수행 절차 및 방법

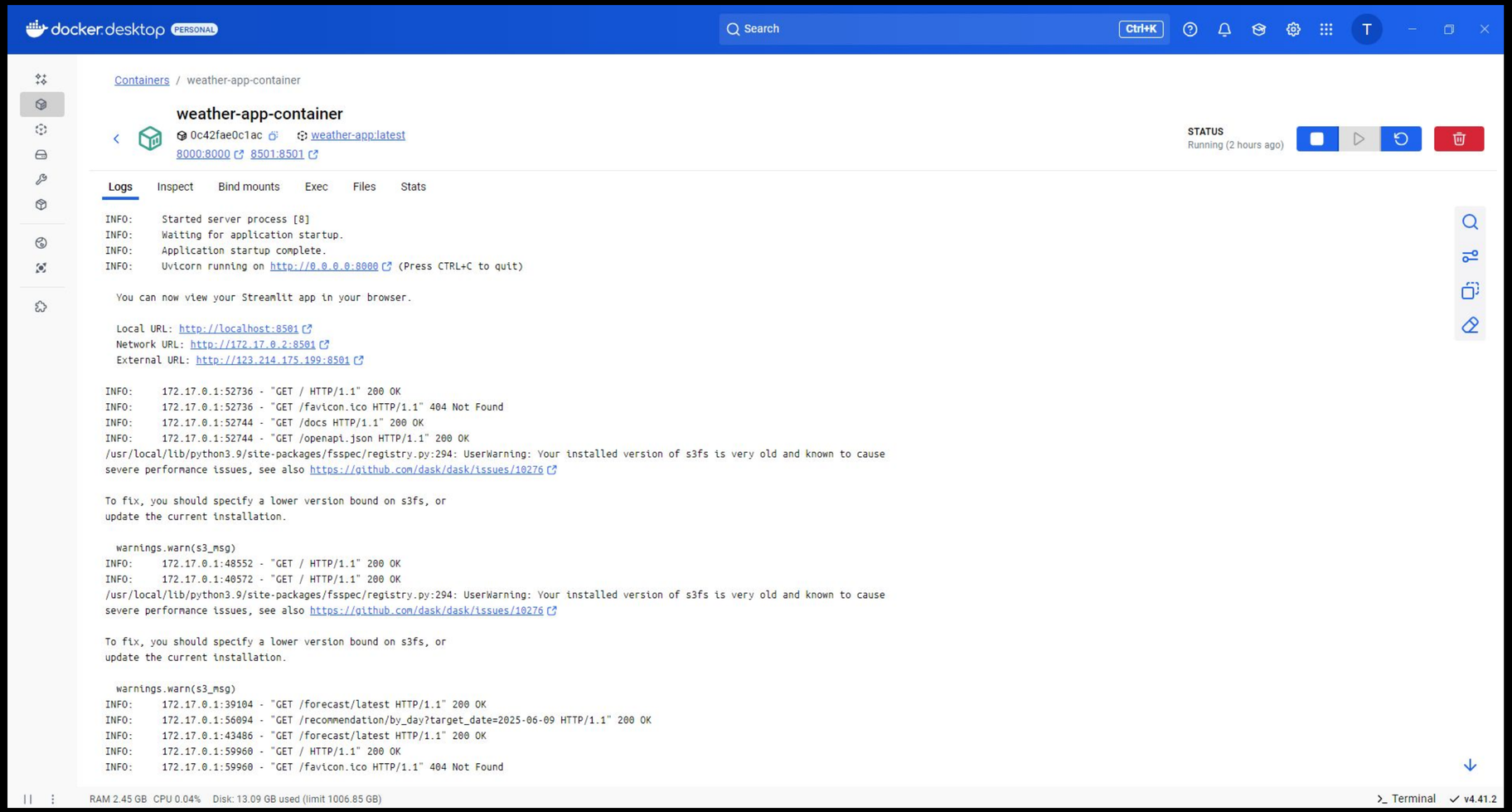
: Github Actions(CI/CD) | 개발 환경



프로젝트 수행 절차 및 방법

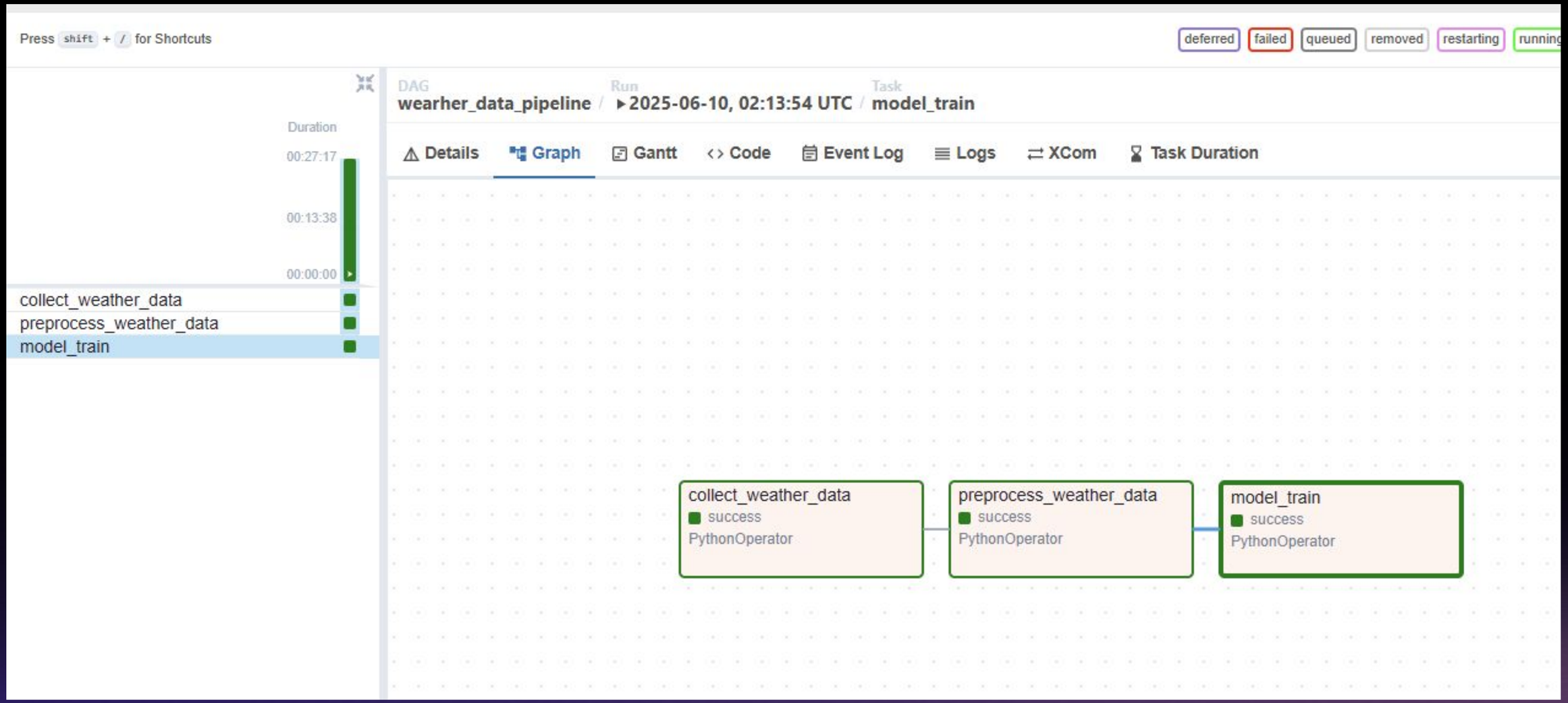
: Docker | 개발 환경

동일한 개발환경 구성



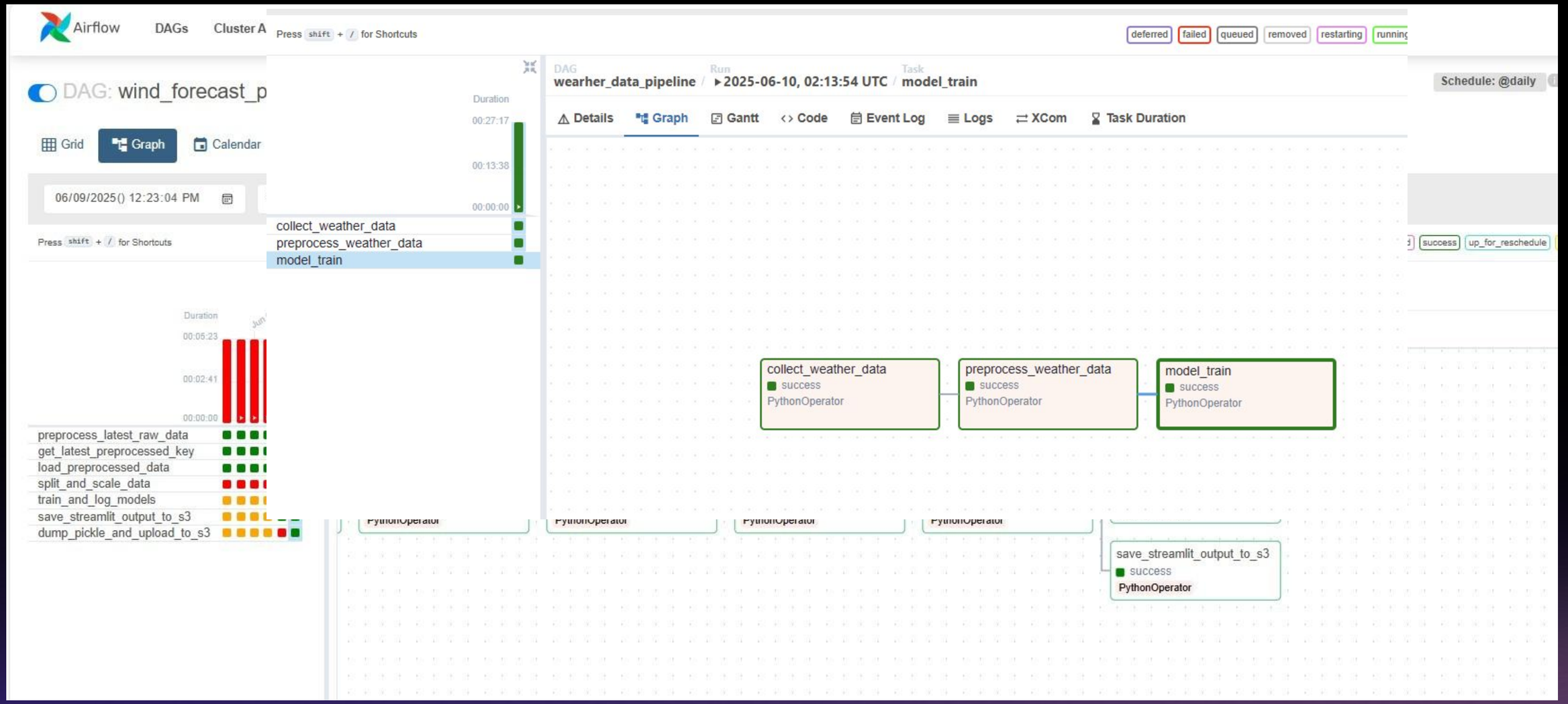
프로젝트 수행 절차 및 방법

: Airflow | 모델 자동화



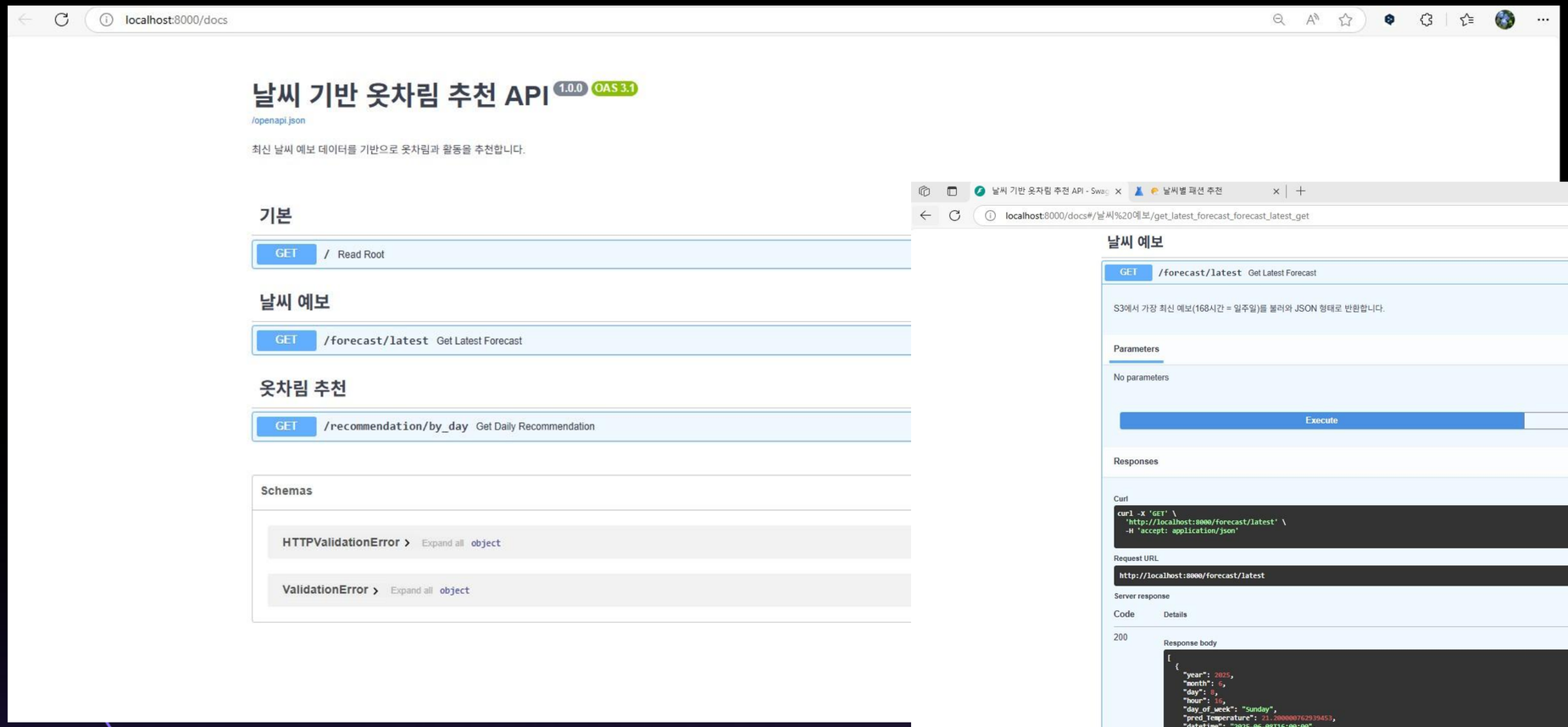
프로젝트 수행 절차 및 방법

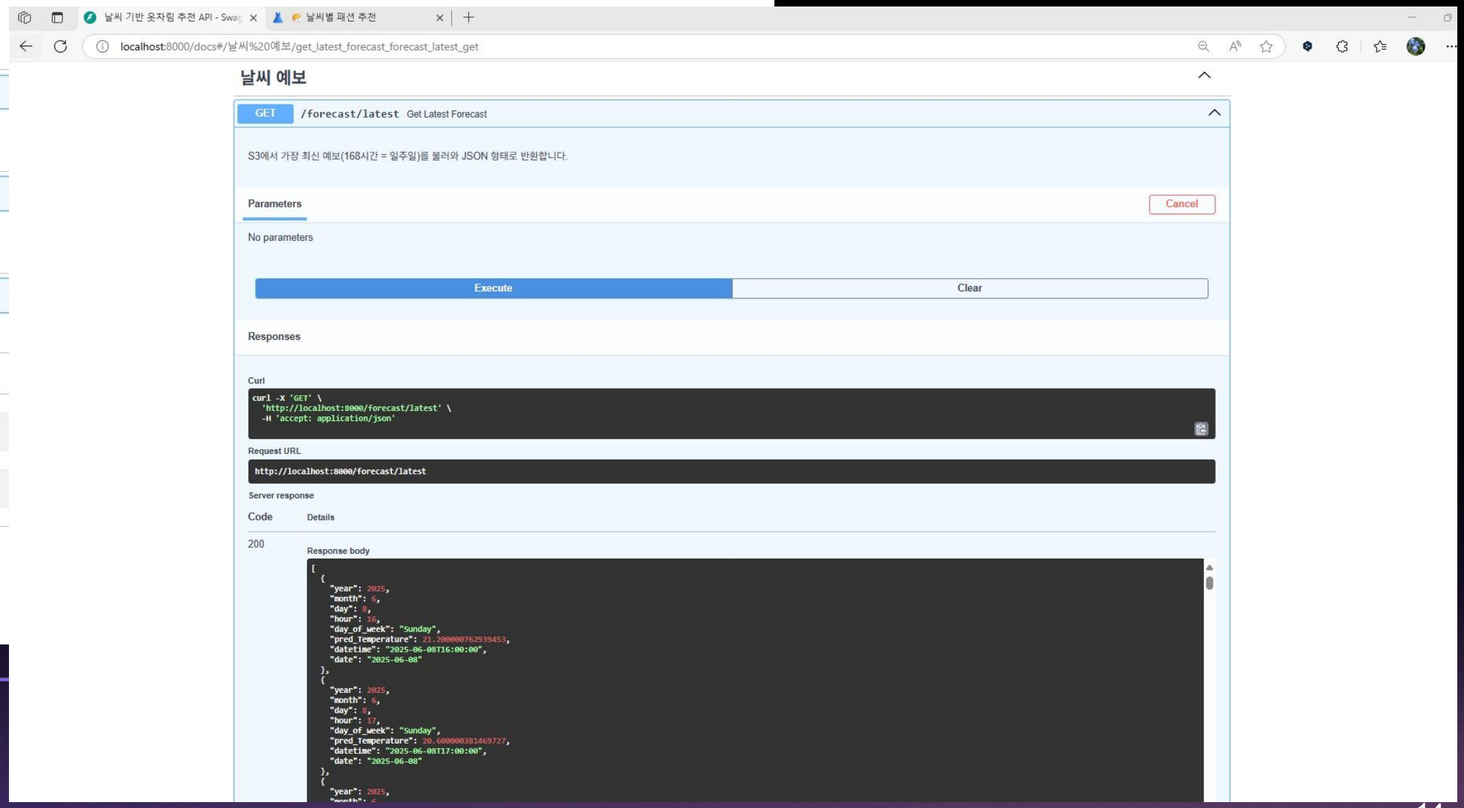
: Airflow | 모델 자동화 테스트



프로젝트 수행 절차 및 방법

: FastAPI | 모델 배포



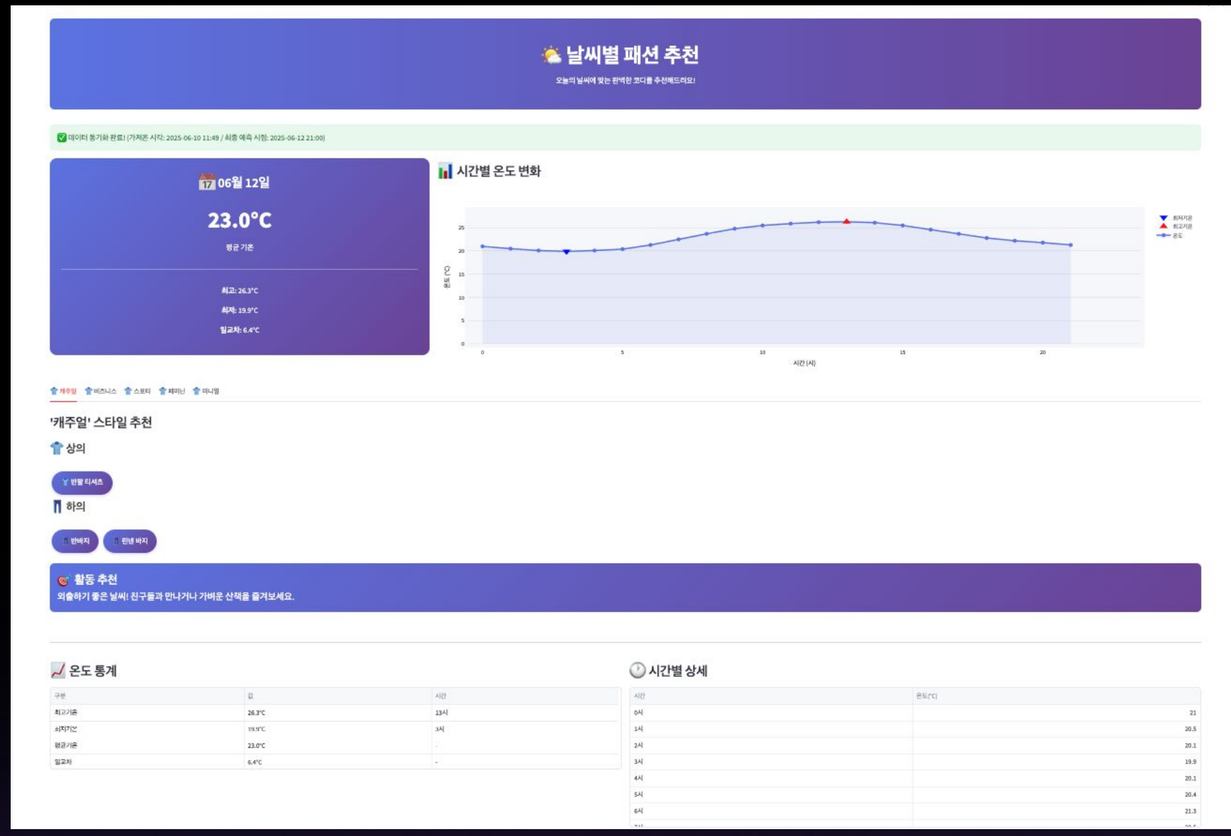


프로젝트 수행 절차 및 방법

MLOps 워크 플로우

* *날짜별 패션 추천 (코드, 페이지 구현)*

```python
# 1. 스타일별 추천 데이터를 모두 정리 (활동 추천 팁 추가)
STYLE_RECOMMENDATIONS = {
    "5도 이하": {
        "items": {
            "캐주얼": {"아우터": ["🧥 롱패딩", "🧥 숏패딩"], "상의": ["🧶 두꺼운 니트", "👕 기모 맨투맨"], "하의": ["👖 기모 바지", "👖 코듀로이 팬츠"]},
            "비즈니스": {"아우터": ["🧥 두꺼운 오버핏 코트"], "상의": ["🧶 터틀넥 니트", "👔 셔츠"], "하의": ["👖 울 슬랙스"]},
            "스포티": {"아우터": ["🧥 벤치파카", "🧥 윈드브레이커"], "상의": ["👕 기능성 긴팔", "👕 후드티"], "하의": ["👖 트레이닝 팬츠"]},
            "페미닌": {"아우터": ["🧥 알파카 코트"], "상의": ["🧶 앙고라 니트"], "하의": ["👗 롱 기장 스커트", "🎻 두꺼운 스타킹"]},
            "미니멀": {"아우터": ["🧥 블랙 패딩"], "상의": ["👕 기본 터틀넥"], "하의": ["👖 와이드 슬랙스"]},
        },
        "activity_tip": "실내 활동을 추천해요. 외출 시에는 목도리, 장갑 등 방한용품을 꼭 챙기세요!"
    },
    "10도 이하": {
        "items": {
            "캐주얼": {"아우터": ["🧥 경량패딩", "🧥 야상"], "상의": ["👕 맨투맨", "👕 후드티"], "하의": ["👖 청바지", "👖 면바지"]},
            "비즈니스": {"아우터": ["🧥 트렌치 코트", "🧥 울 자켓"], "상의": ["👔 셔츠", "🧶 얇은 니트"], "하의": ["👖 슬랙스"]},
            "스포티": {"아우터": ["🧥 플리스 자켓"], "상의": ["👕 긴팔 티셔츠"], "하의": ["👖 조거 팬츠"]},
            "페미닌": {"아우터": ["🧥 트위드 자켓"], "상의": ["👚 블라우스", "🧶 가디건"], "하의": ["👗 미디 스커트"]},
            "미니멀": {"아우터": ["🧥 바람막이"], "상의": ["👕 기본 긴팔티"], "하의": ["👖 블랙진"]},
        },
        "activity_tip": "따뜻한 차 한 잔과 함께 공원을 산책하거나, 예쁜 카페에 가기 좋은 날씨예요."
    },
    "20도 이하": {
        "items": {
            "캐주얼": {"아우터": ["🧥 가디건", "🧥 청자켓"], "상의": ["👕 셔츠", "👕 긴팔티"], "하의": ["👖 면바지", "👖 청바지"]},
            "비즈니스": {"아우터": ["🧥 얇은 자켓", "🧥 블레이저"], "상의": ["👔 셔츠"], "하의": ["👖 슬랙스", "👗 원피스"]},
            "스포티": {"아우터": ["🧥 바람막이"], "상의": ["👕 PK 티셔츠"], "하의": ["👖 반바지", "👖 레깅스"]},
            "페미닌": {"아우터": ["🧥 가디건"], "상의": ["👚 블라우스", "👗 얇은 원피스"], "하의": ["👗 롱스커트"]},
            "미니멀": {"아우터": ["🧥 얇은 바람막이"], "상의": ["👕 U넥 티셔츠"], "하의": ["👖 청바지"]},
        },
        "activity_tip": "야외 활동하기 완벽한 날씨! 자전거를 타거나 가까운 곳으로 피크닉을 떠나보세요."
    },
    "28도 이하": {
        "items": {
            "캐주얼": {"상의": ["👕 반팔 티셔츠"], "하의": ["👖 반바지", "👖 린넨 바지"]},
            "비즈니스": {"상의": ["👔 린넨 셔츠", "👕 PK 셔츠"], "하의": ["👖 린넨 슬랙스"]},
            "스포티": {"상의": ["👕 기능성 반팔"], "하의": ["🩳 스포츠 반바지"]},
            "페미닌": {"상의": ["👚 퍼프 소매 블라우스"], "하의": ["👗 롱 원피스", "👗 숏츠"]},
            "미니멀": {"상의": ["👕 기본 반팔티"], "하의": ["👖 와이드 숏츠"]},
        },
        "activity_tip": "외출하기 좋은 날씨! 친구들과 만나거나 가벼운 산책을 즐겨보세요."
    },
```



04

회고

결과 / 인사이트 도출 /

향후 계획 느낀점

프로젝트 회고

: MLOps Project | 결과 및 향후 계획

1

** 최종 결과*

- **날씨 예측 결과 기반으로 옷차림 추천 서비스 완성**

딥러닝 모델 예측부터 API서빙, UI 대시보드 제공 및 자동 배포까지의 전체 MLOps 라이프 사이클을 자동화

2

** 도전 과제 및 해결*

- 팀원끼리 개발 환경 불일치 문제

도커파일 3개 만들어서 앱과 모든 의존성을 하나의 이미지로 만들어서 동일한 환경 보장

3

** 인사이트 도출*

- 전체 워크플로우를 모듈화하여 각 단계의 독립성과 효율성을 높이는 데 집중했지만, 데이터와 모델을 과도하게 분리할 경우 수정 및 고도화 과정이 오히려 복잡해질 수 있다는 점을 경험함.

4

** 개선 방향*

- 프로젝트 주제와 규모에 맞춰 규모에 맞춰 버전 관리의 우선순위를 정하고 집중할 부분을 선택하는 것이 중요.

느낀점

프로젝트 진행 소감

: MLOps Project | 느낀점

* 이승민 : 전체 워크플로우를 단계별로 나누고, 각 Task 를 독립적인 모듈로 분리해 맡은 역할에 집중할 수 있었고 모듈화 덕분에 작업 분담이 명확해졌고, 각자 맡은 영역의 관리와 고도화가 용이해 좋았고 이러한 구조는 협업을 어떻게 하는지 경험 할 수 있어 좋았습니다.

* 문국현 : MLOps 인프라를 구현하며 실시간 데이터 누락, 시계열 정렬, 모델 예측 문제, 등 다양한 문제를 직접 해결해보며 자동화의 복잡성과 중요성을 경험할 수 있어서 좋았습니다.

* 홍정민 : MLops를 전반적으로 경험할 수 있어서 의미 있는 시간이었습니다. MLOps 너무 재미있어서 오랜만에 몰입을 경험했습니다.

* 조선미 : 혼자서는 이해하기 어려웠을 MLOps 개념(AWS S3, MLflow, Airflow, Docker)을 팀원들의 도움으로 잘 익힐 수 있었습니다. 팀 프로젝트에 함께할 수 있어 감사했습니다.

* 문진숙 : MLOps를 처음 접하다보니 개념 설정부터 구현까지 각 과정마다 무수히 많은 시행착오를 겪었는데 노션과 슬랙, 블로그에 상세한 튜토리얼을 공유해주셔서 전체적인 프로세스를 이해하는 데 매우 큰 도움을 받았습니다. 팀원분들께 정말 감사합니다.

Life-Changing Education

감사합니다.