

이력서

소개



- 이름 : 이재현
- 출생 : 1996.12.25
- Email : swaa23@gmail.com
- Github : <https://github.com/JaehyunL>
- blog : <https://edudeveloper.tistory.com/>
- 코멘트: 데이터 흐름 설계부터 AI 모델 서빙까지, 현장 중심의 실무 프로젝트를 수행해온 백엔드/AI 개발자입니다.

✉ 경력

- 로딕스 2020.11 ~ 재직중
 - GIS 기반 회사로 R&D 프로젝트 및 SI 프로젝트 위주로 업무 경험을 쌓았습니다.

🎓 학력

- 중부대학교(정보보호학과) 2015-03 ~ 2021.02

🛠 기술 스택

Front Skills

- JavaScript, TypeScript(ES6) - React, NextJS
- Android Studio (JAVA)
- JAVA - JavaFX, JavaSwing
- Python - PyQt5

Backend Skills

- 🐍 Python - Flask, FastAPI, Django
- ORM, SQLAlchemy, Postgresql, MySQL
- spatial Query, GeoServer, GDAL
- RDB
- Celery, rabbitmq, Redis
- selenium, scheduler
- 🎨 Torch, huggingFace
- Swagger / ReDoc
- Postman, Pytest
- Java - Spring

Dev Ops

- 🚤 Docker, DockeSwarm, K8s, GCP, AWS
- 사내 Gitlab운용 경험, NAS운용 경험(Synology)
- CI: Gitlab 파이프라인 구현

AI/ML

- CycleGAN, StyleGAN
- Stable Diffusion, Gen-2
- BERT, GPT3.5, GPT4
- YOLO, LightGlue, SuperGlue

프로젝트

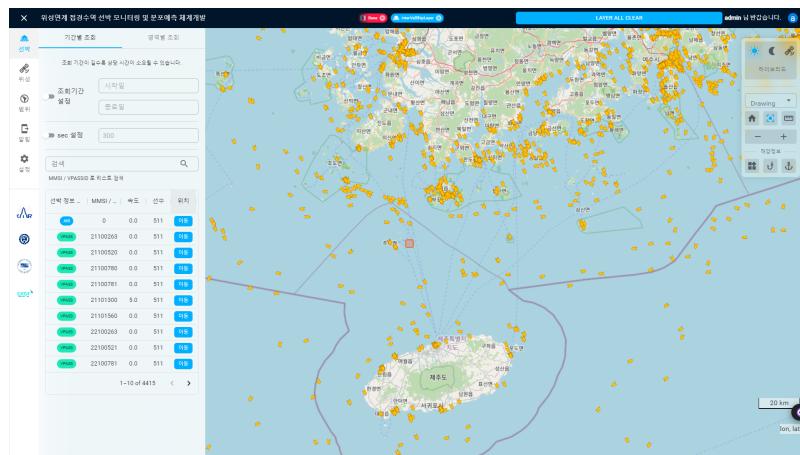
인공지능 기반 라이다 및 열영상 융합을 통한 야간 상황인지 기술

- 2024.03 ~ 2025.02
- 담당업무: 카메라 to 데이터 모델 파이프라인 처리, GAN 모델 Train 및 Fine Tuning
- 사용기술: OpenCV, Pytorch, ML(CycleGAN, UNSB, Lightglue)
- 주요성과: 다중밴드를 통한 ML모델 학습, 8비트 모델 16비트 변환, 센서간 켈리브레이션 수행, 모델 서빙
- [관련논문](#)



선박 모니터링 시스템

- 2023.02 ~ 2024.02
- 담당업무: API작업, 데이터 파이프라인 구축, 스케줄링, 불법선박 탐지 알고리즘 연동
- 사용기술: Flask, Celery, ORM, Docker
- 주요성과: 외부기관 알고리즘 수행 사이클 로직 구축 [데이터 수집, 전처리, 후처리], AIS 데이터 핸들링 (일 약 2천만 건), 긴급사항 알림기능 (MMS, Email, 카카오톡)



스마트현장 재난 가이던스 시스템

- 2023.02 ~ 2023.11
- 담당업무: API작업, 스케줄링, 데이터 파이프라인 구축, OSM 전세계 구축
- 사용기술: FastAPI, Docker, Celery, OSM, File-Stream
- 주요성과: Docker OSM 부하 조절(Swap Memory, 캐싱값 조절 등), ISO-3166 코드 매칭 (<https://pypi.org/project/iso3166KOR/>) 라이브러리 제작

디지털 상황판

- 2022-09 ~ 2023-01
- 담당업무: HTML 정적 내 검색 로직 구현, 기타 API 보조 작업
- 주요성과: 크롤링을 통한 HTML상 데이터 검색 로직 구현
- 사용기술: Selenium, Flask, BS4

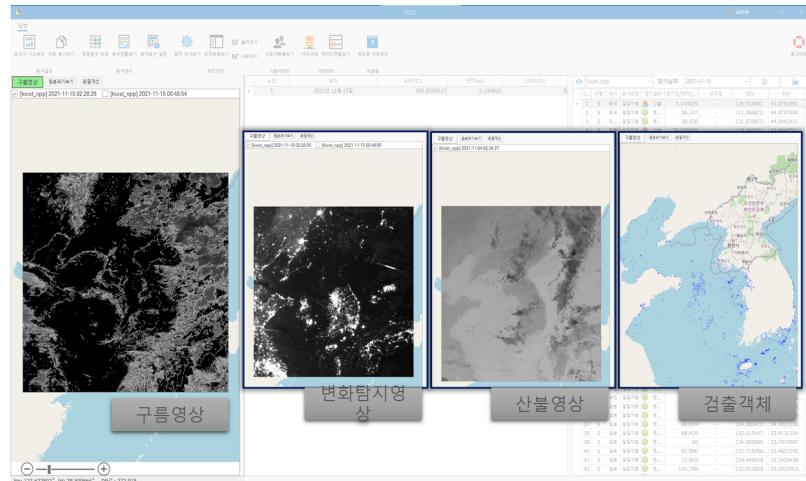
스마트팜

- 2021-11~2022-11
- 담당업무: 백엔드 API 구현, 프론트 화면 구현, 외부데이터 수집, 정사영상제작(드론이미지)
- 주요성과: 비 정제 데이터 수집 자동화
- 사용기술: Flask, React, OpenLayer, PostGresql ,OpenDronMap
- 관련논문



달빛영상 기반 변화탐지

- 2021-05 ~ 2021-11
- 담당업무: 위성 영상 수집(FTP) 자동화 프로그램 구축 및 데이터 파이프라인 구축, 외부기관 알고리즘 연동, API 구현
- 사용기술: Scheduler, FTP, geoserver, Postgrsql, Flask, GDAL
- 주요성과: 실시간 Schedule을 통한 데이터(위성영) 전처리 과정 자동화



국가해양영토 광역 감시망 구축

- 2020-11 ~ 2021-04
- 담당업무: 데이터 파이프라인 구축, 외부기관 알고리즘 연동, API 구현
- 사용기술: Celery, RabbitMQ, Flask, Postgresql, geoserver, SFTP, Docker, Docker-Swarm
- 주요 성과: 쿼리 조회시 데이터 크기로(조회시 약 3GB) 인한 메모리 이슈 현상 개선(페이지 처리), 데이터 수집 전처리 파이프라인 코드
- 국토 광역감시망 소개 영상

The screenshot shows two side-by-side windows. The left window is a 'WPS Admin' interface with a tree view of '국내 관리환경' (Domestic Management Environment) and a '선택 정보' (Selected Information) panel. The right window is a code editor with Python code related to task scheduling:

```

# celery app task bind=True, name='wiss.schedule.work_exec'
def work_exec(celid):
    # ...
    # celery에서 수행되는 Task
    # 스케줄 정보를 불러와 현재시간이 설정되어있는 시간을 대 (cron 포맷 가능)
    해당_Schedule_id가 현재를 work_execute 실행

    # Schedule DB 조회 -> TASK 등록, 시간 범위가 예상되는 것만 (start_dt < 현재 시간 < end_dt)
    cur_time = datetime.datetime.now()
    cur_dt = cur_time.strftime("%Y-%m-%d %H:%M:%S")
    schedules = session.query(ScheduleInfo).filter(
        and_(cur_time >= ScheduleInfo.start_dt, cur_time < ScheduleInfo.end_dt))
    # ...
    finish_schedules = session.query(ScheduleInfo).filter(
        ScheduleInfo.end_dt <= cur_time)
    # ...
    finish_in_finish_schedules = session.query(ScheduleInfo).filter(
        ScheduleInfo.end_dt <= cur_time)

    # TODO: 원하는 순서대로 처리하기 위해 각각의 스케줄에 따른 작업 번호 정하기
    for schedule in schedules:
        if schedule.type == 'Cron':
            cron = CronExpression(schedule.rc_cycle)
            if cron.is_valid():
                time = cron.get_next(datetime.datetime.now())
                log.info("running task (%s(%s))" % (schedule.schedule_nm))
                if schedule.type == 0:
                    sp_link.execute.apply_async(args=(schedule.id,))
                elif schedule.type == 1:
                    work_exec_apply_async(args=(schedule.id,))
            elif schedule.type == 2:
                log.info("running task (%s(%s))" % (schedule.schedule_nm))
                work_exec_apply_async(args=(schedule.id,))
            else:
                raise WISEException("schedule type invalid!")
        create_rd_info(schedule.dts_colct_id, "SUCCESS", schedule.type)
    
```

긱스킬UP!

항목

내용

개발자리 edudeveloper.tistory.com

쓰기 블로그관리 흠

콘텐츠 글 관리 페이지 관리 카페고리 관리 공지 관리 서식 관리 설정

댓글·방명록 댓글 관리 방명록 관리 설정

통계 방문 통계 유입 경로

수익 애드센스 관리

오늘 방문수 29
어제 방문수 36
누적 방문수 82,510

방문통계

최근 7일 통계

인기글

- 1 Python Postgresql 연동하기
- 2 Python Flask Example 플러스크 예제 !! 정복하기(1)
- 3 Docker 예모리 관련 정리
- 4 <class 'openpyxl.styles.named_styles._NamedCellStyle'>.name sh...
- 5 문제 해결 공유 Flask Werkzeug (ImportError: cannot import name '...')
- 6 1) 우분투 nginx 설치 및 설정
- 7 MYSQL Join(inner, outer, left , right) 기능 설명
- 8 Python Pandas(2. 누락된 데이터 처리하기.)
- 9 Android studio 프로그먼트에서 토스트메시지 띄우기

유입 채널

검색 92.2%
SNS 0.0%
기타 7.8%

유입 키워드

유입된 키워드가 없습니다.

블로그

JaeHyunL / README.md

🔥 Skills

- Main
- Python python
- Flask Flask
- Postgres Postgres
- fastapi fastapi
- docker docker

- Sub
- java java
- javascript javascript
- typescript typescript
- openstreetmap openstreetmap
- geoserver geoserver
- Qgis Qgis
- React React
- Next Next
- Openlayers Openlayers
- Android Android

Popular repositories

- Python-Alg
- FinalCapston
- algorithm_study
- JaeHyunL.github.io
- LeeoaeHyun
- Capstone

Achievements

Customize your pins

318 contributions in the last year

Learn how we count contributions

Activity overview

Contribution settings

2023 2022 2021 2020 2019

Github

0 selected

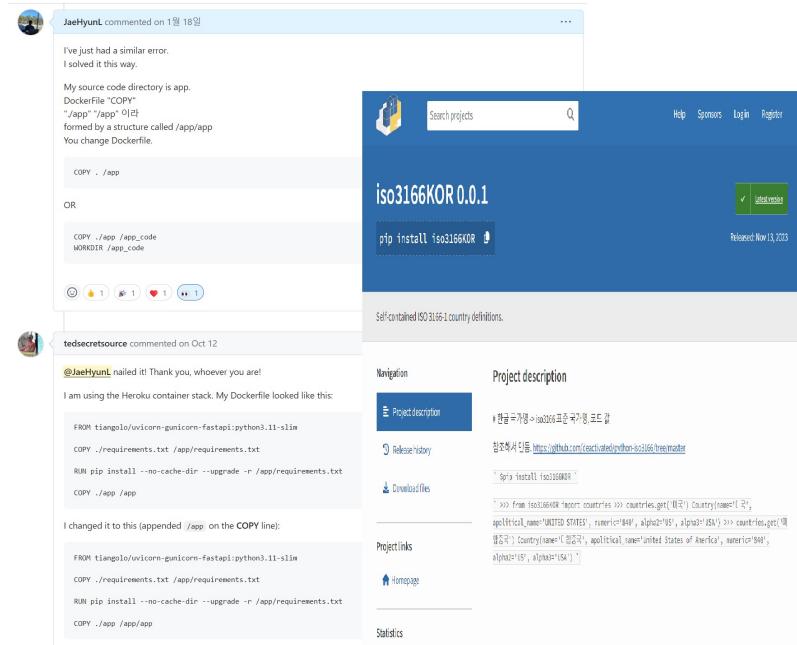
- Binary Prediction with a Rainfall Dataset
- Spaceship Titanic
- I'm Something of a Painter Myself
- Natural Language Processing with Disaster Tweets
- Housing Prices Competition for Kaggle Learn Users
- Titanic - Machine Learning from Disaster

1985/4381
238/2041*
27/103*
92/847*
532/7013*
2843/16125*

Kaggle

항목

내용



▶ Before (학생)

얼굴 인식 출석 자동화 프로그램

사용기술 :

<http://isweb.joongbu.ac.kr/~jbuis/2020/ppt-2020-4.pdf>

- Python / Flask
- Angular
- MYSQL
- Docker
- GCP

제작기간 :

- 2020-08~2020-10

스미싱 탐지 프로그램

사용기술 :

- Python / Flask
- Anroid studio
- MYSQL
- AWS
- Google SafeBrowsing

제작기간:

- 2020-04~2020-06

자바 GUI 기반 암호화 프로그램

사용기술 :

-JAVA /**** GUI -JAVAFX

제작기간:

- 2019-06~2019-06