

이지호

Full Stack Developer

Computer Science와 Algorithm을 개발에 접목시키는 풀스택 개발자입니다.
시간 복잡도와 빌드 최적화에 집착하지만, 의미있는 현실 세계의 문제를 풀려고 노력합니다.
현재 테스트 자동화를 비롯한 Software Verification에 관심이 많습니다.

보유 기술

Python, Django, DRF, Django ORM, Celery, Docker, Kubernetes, TypeScript, React, SQL, Property Based Testing(hypothesis, fast-check)

Front-end Engineering

- Typescript를 주 언어로 사용
- React, Jotai, Material UI등 리액트 기반 프레임워크에 능숙
- React-Konva 프레임워크를 통해 캔버스 기반 인터랙티브 툴 개발 경험

Back-end Engineering

- Django, DRF 기반 백엔드 애플리케이션 작성 경험 다수
- drf-yasg를 사용한 swagger 명세 관리
- Uvicorn과 ASGI를 사용한 multiple worker 서버 구축
- Celery와 Channel 이용 리얼타임 WebSocket 서버 / 워커 / 클라이언트 작성
- Celery 환경에서 Pytorch 작업 내장 multiprocessing으로 병렬화 [#관련 이슈](#)

Infrastructure as Code

- 프론트엔드 빌드 툴체인에 대한 이해와 최적화에 능숙
- create-react-app에 의존하지 않고 더 최적화된 리액트 웹앱 구축 가능
- Github Actions 기반 Docker, Kubernetes 빌드 및 배포 자동화 구축 경험 다수

교육사항

한국산업기술대학교 / 컴퓨터공학과 학사과정 (2021년 휴학)

2019.03 ~

Aurola / Freelancer Task (2021.08 ~)

머신러닝 기반 데이터 라벨링 플랫폼

프로젝트 상세

- 오토라벨링 기능이 있는 이미지 라벨링, 검증, 모니터링 플랫폼
- FE: Typescript, React, Jotai, MUI, react-router, React-konva
- BE: Python, Django, DRF, drf-yasg(Swagger), Celery, Channel(WebSocket Server)
- CI/CD: Github Actions, Docker(+ Private Registry), Kubernetes
- Github Flow로 주별 스프린트 관리

역할 및 성과

- 팀 내 프론트엔드, 백엔드 메인 개발자 역할
- webpack 빌드 최적화로 46초 -> 2초 이내 빌드 환경 구성 (23배 향상, M1 Pro 기준)
- 빌드를 최적화한 이유: 단순히 시간 문제가 아닌 프론트엔드 개발 사이클 속도에 정비례하기 때문
- yarn berry 사용함으로써 zero install 환경 구성
- react lazy component loading 도입, 유의미한 수준의 초기 렌더링 시간 축소 및 번들 파일 분리로 인한 빌드 성능 추가 향상 (4초 -> 2초)
- react custom hook으로 JWT two token 워크플로우 자동화
- react-konva를 이용하여 라벨링된 json을 이미지에 렌더링시켜주는 파서 작성
- yolov5, resnet50 모델 사용해서 오토 라벨링 프로토타입 구현
- 배치 이미지 라벨링 프로세스 Worker Job으로 pool 사용해서 병렬화
- Inner Join과 Prefrech_related 메소드 사용함으로써 ORM N + 1 문제 해결
- AutoCrawler 오픈소스 크롤러 Celery + Docker 환경으로 포팅, Django ORM + Boto3 사용하여 DB 업로드 방식으로 변경
- 점진적으로 백엔드 단위 테스트, PBT 도입
- Django ORM을 사용한 데이터베이스 스키마 공동 설계
- Sliding Window 알고리즘 사용한 N:M 3배수 교차 검증 분배 알고리즘 작성
- Kubernetes yaml 작성으로 전체적 Active / Active Pod 이중화로 SPOF 제거
- Github Actions 사용 로컬 docker registry 사용해서 docker image build & push, kubectl로 쿠버네티스 원격 배포까지 룰링 릴리즈로 자동화
- CD의 이전 조건으로 CI가 돌아가도록 구성, 테스트 스위트 작성으로 의미있는 CI 구성