

고등학생인 내가 MLOps라니 무리무리!!
(*무리가 아니었다?!)

yjs12180825@gmail.com

발표자 소개

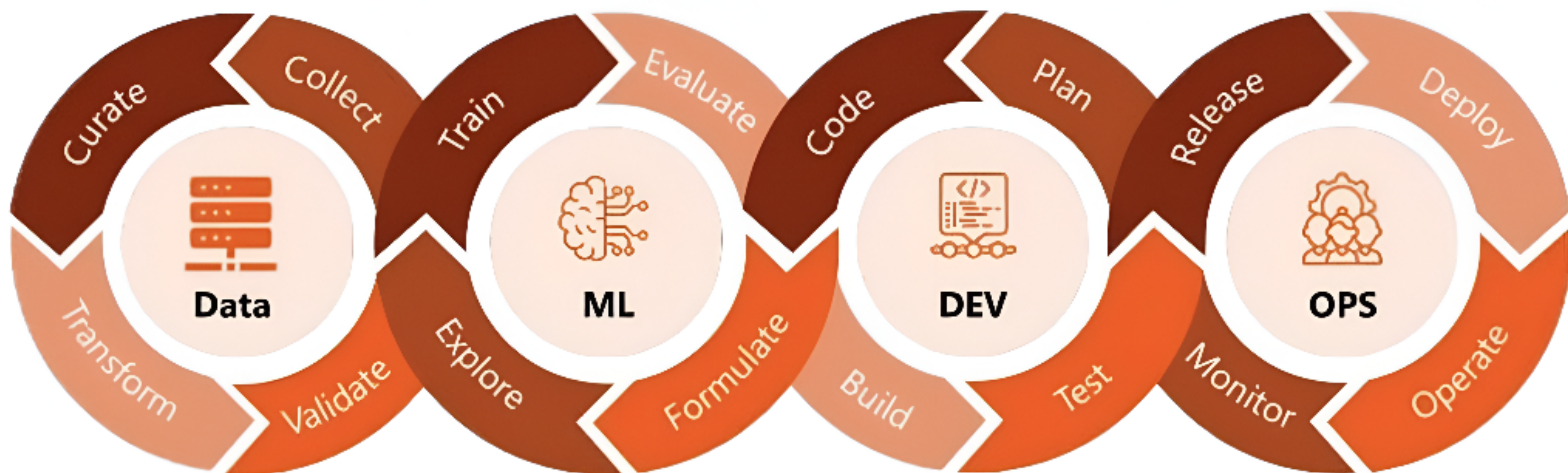
윤지상

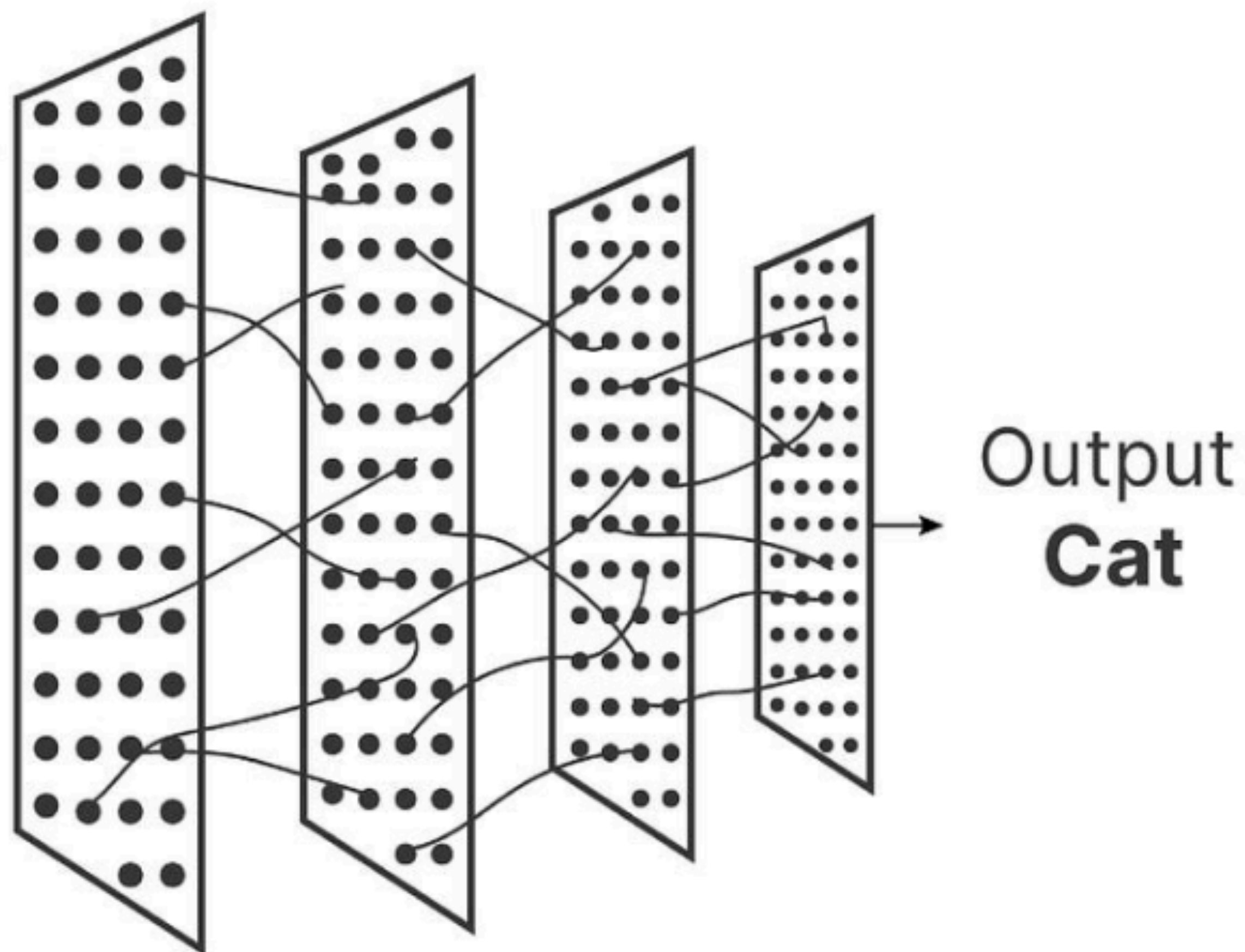
Developer

서울디지털대학교 (2023.03 ~ 2026.02)

(주)데이터플로 (2024.09 ~ 2025.11)







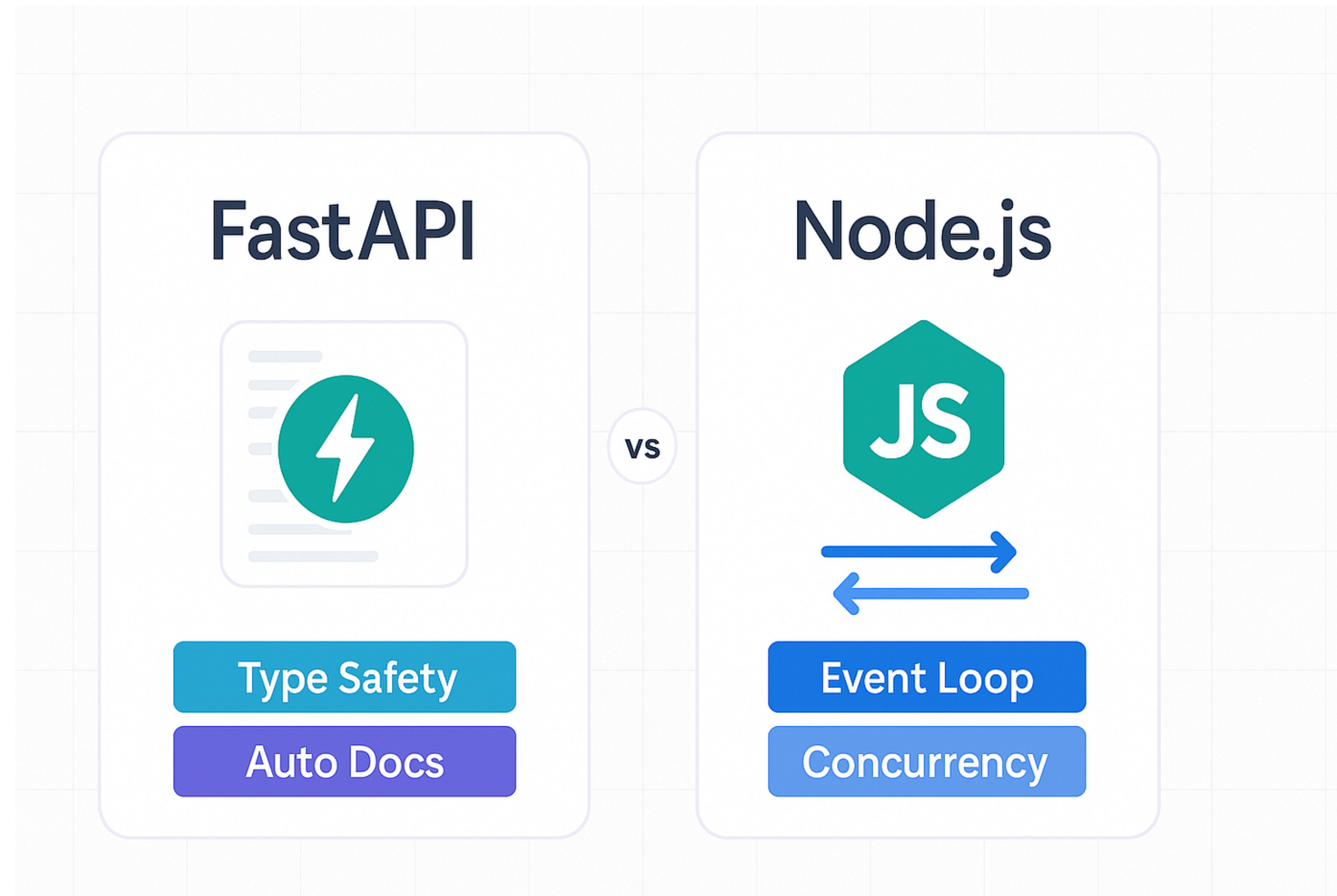
MLOps

머신러닝 모델의 개발부터 배포, 모니터링, 재학습까지 전 과정을 자동화하고 통합하는 운영 체계

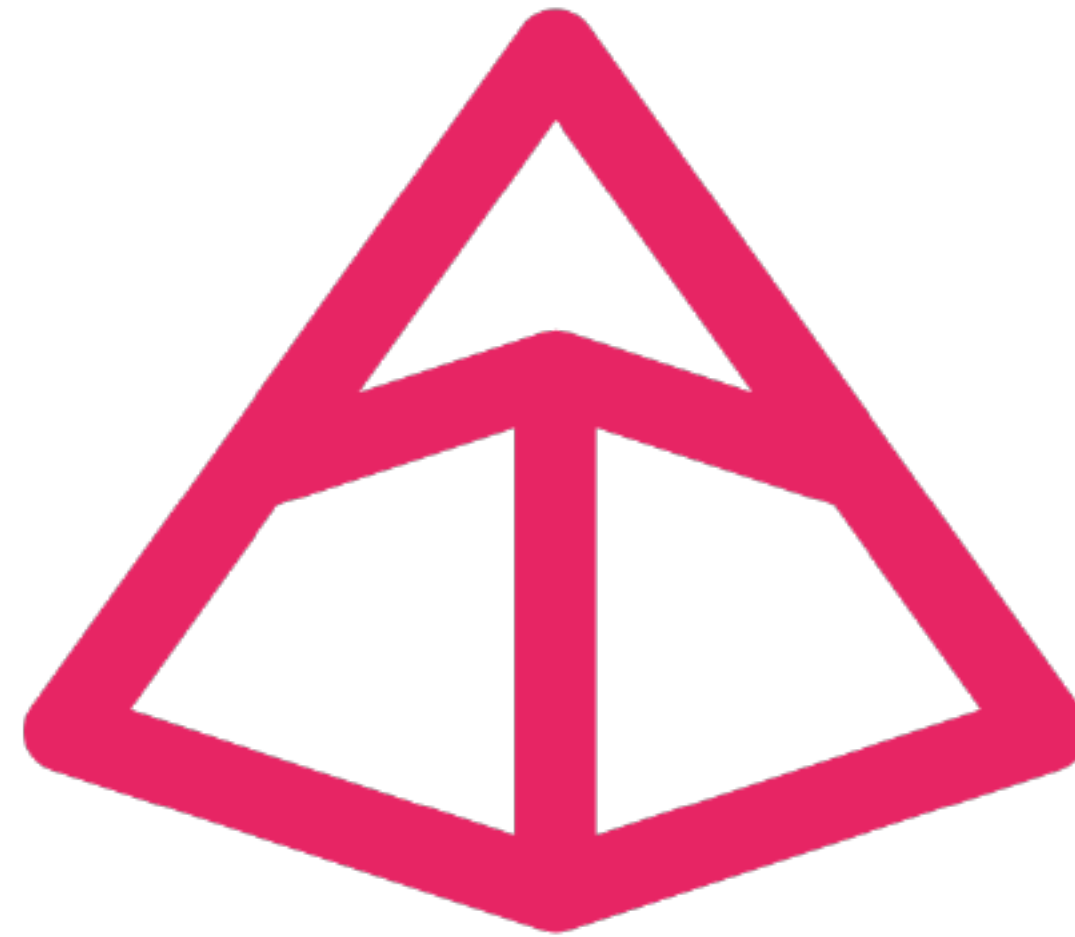
왜 FastAPI를 선택했는가?

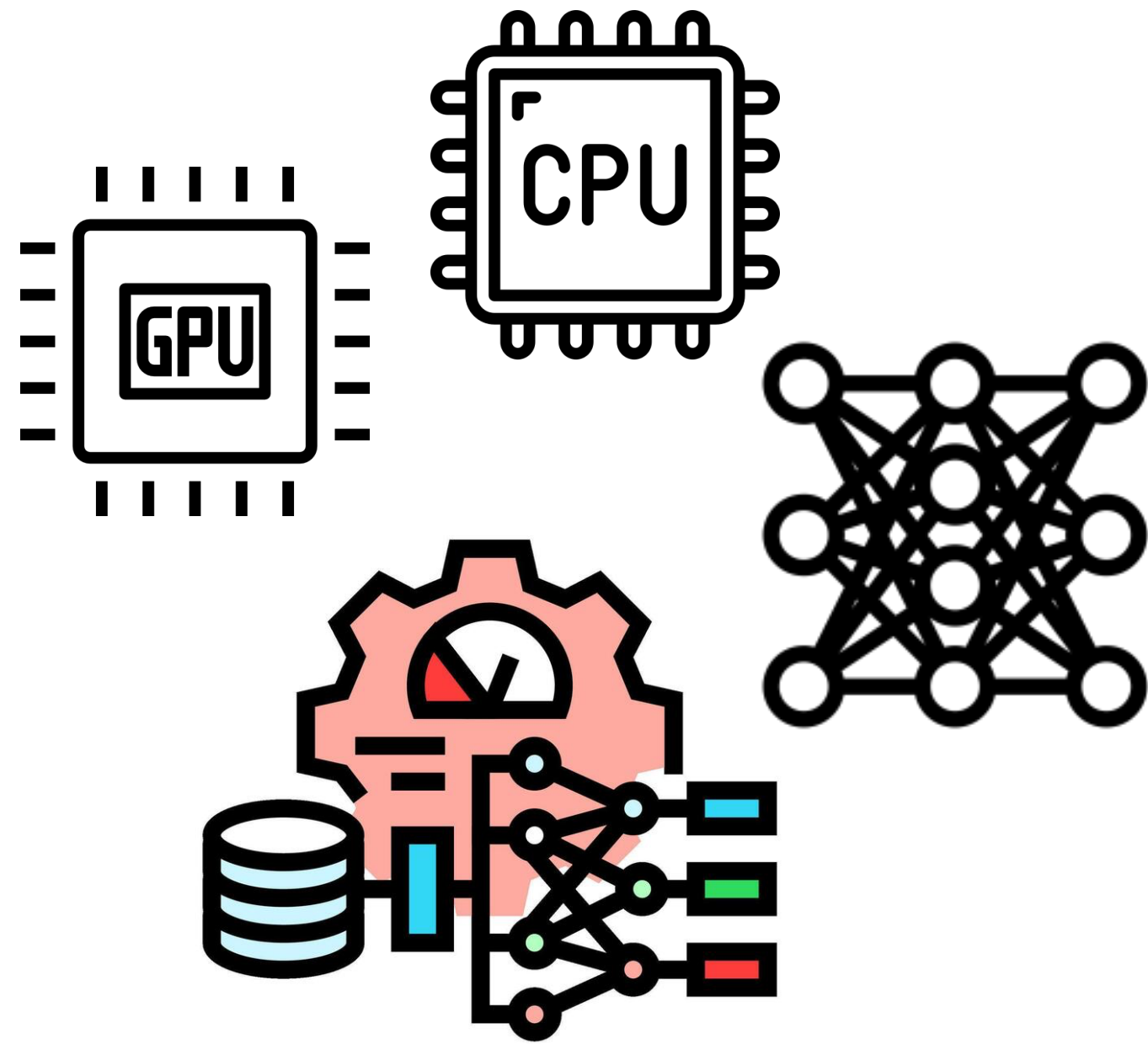


Heavy한 ML Job을 돌리면서도 API 응답성을 잃지 않는 플랫폼



Pydantic 기반의 엄격한 Validation과 타입 힌트를 통한 모델 스펙 문서화





Type Safety

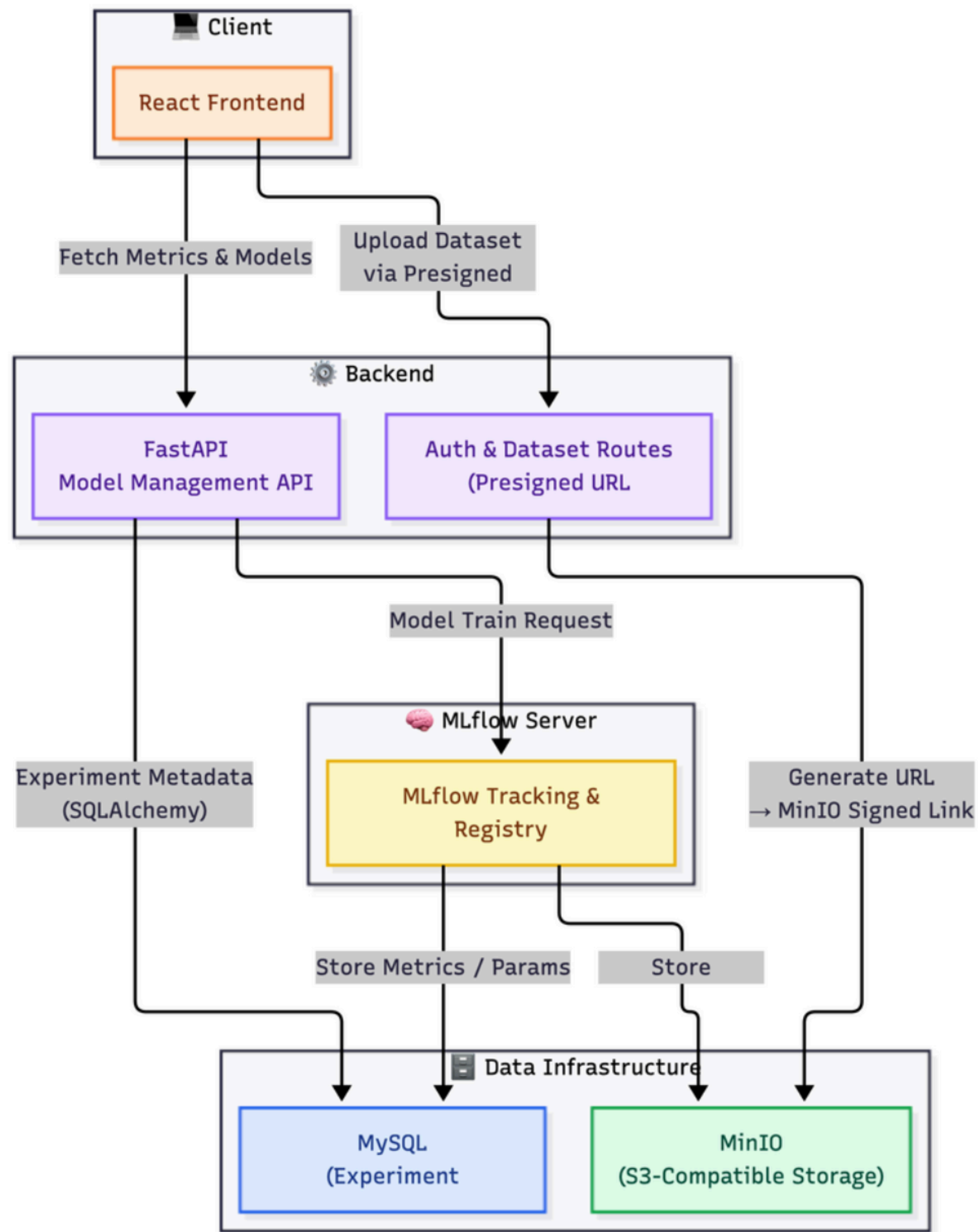
```
class WorkspaceCreate(BaseModel):  
    name: str  
    description: Optional[str] = None  
    python_version: str  
    use_gpu: bool = False  
    max_used_gpu: Optional[int] = None  
    cpu_limit: Optional[float] = None  
    memory_limit: Optional[int] = None  
    spec_digest: Optional[str] = None  
    timeout_seconds: int = 300  
    auto_resume: bool = False  
    retry_count_limit: int = 0  
    is_default: bool = False
```

왜 Type Safety 가 중요한가?

1. 파라미터 누락, 타입 오류로 학습 실패 → GPU 등 자원 낭비
2. 자동 재실행이 설정되어 있다면? → 자원 낭비가 더욱 커짐

파이썬 기반이라 ML 코드와 직접적 통합이 쉬움





FastAPI : Model Train Job, Dataset Handler

MinIO : Model Artifacts, Model output file ...

MySQL : Model Metadata

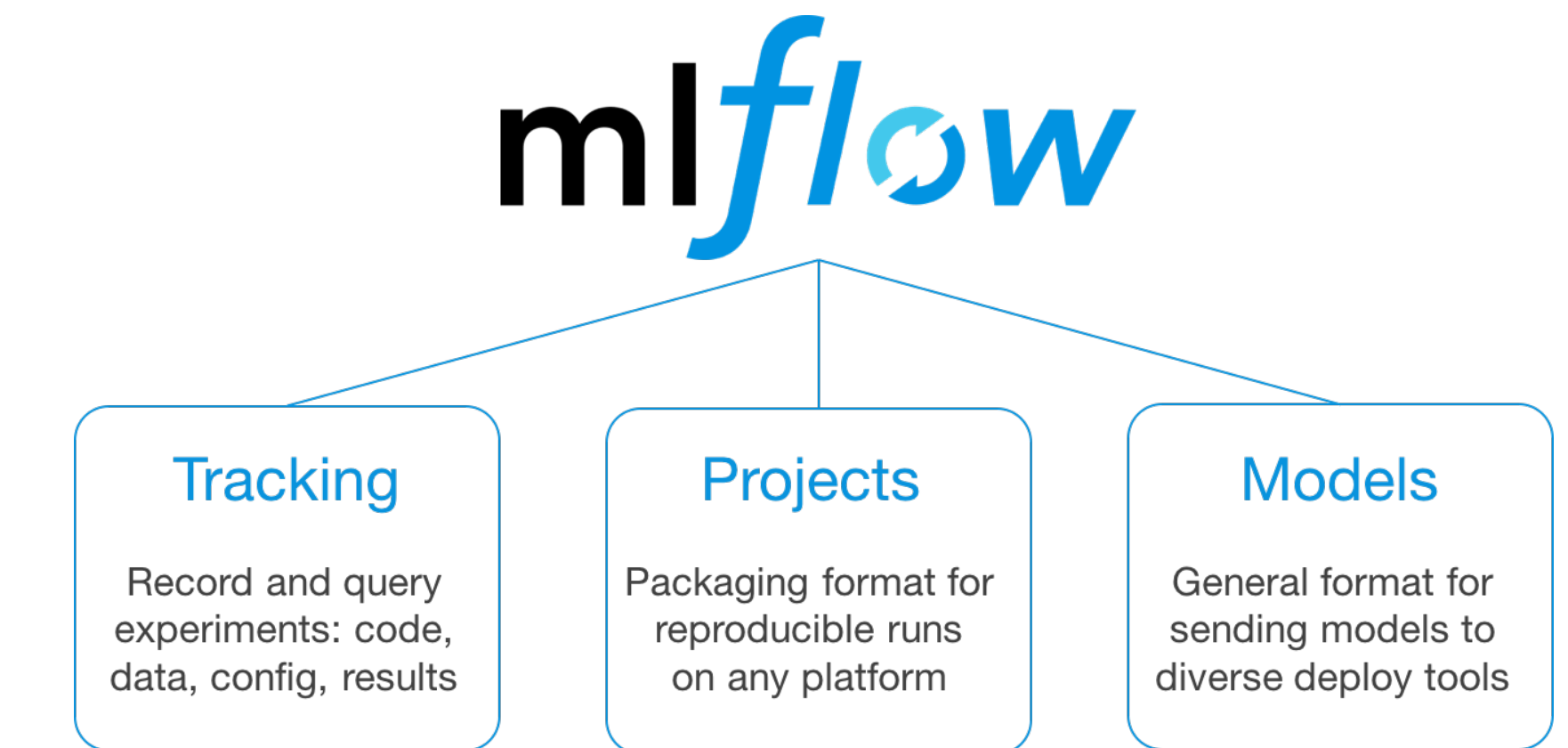
MLFlow : Model Tracking

MLFlow와 k8s를 도입하게 된 이유

재현성이 매우매우 중요

⇒ 로그와 파라미터, 아티팩트를 전부 저장

실험 기록이 자동으로 관리, 환경 차이 없음





딥러닝 모델의 추상화?

⇒ 공통 인터페이스가 필요



train()

- 학습 루프, 옵티마이저 업데이트, 체크포인트 저장 등의 규칙을 동일하게 맞춤

eval()

- 평가 지표 계산 및 검증 데이터 처리 방식을 표준화

save_model()

- 모델 저장 경로, 파일 구조, 메타데이터를 통일해 재현성 확보

로그 출력 형식 통일

- Loss, Accuracy, Step, Epoch 등을 MLFlow나 로깅 시스템에 일관적으로 기록

파라미터 구조 통일

- learning_rate, batch_size, epochs 등 모델별로 달랐던 Key를 단일 스키마로 교정

Init Container

Download Source code from MinIO

Pull Docker Image from Local Registry

Session Initialized

Train Container

Train Models

Saved Model

Save Artifacts and Model outputs

Save Log

왜 Container를 분리하였는가?

모델 이미지를 경량화

모델 재현성을 향상시키고 병렬 모델 학습 Job으로 확장하기 위해

컨테이너 이미지 빌드 최적화

⇒ Environment, Project 라는 논리적 단위를 나눔

1. Environment 생성 시 Docker Image 빌드
2. Project 실행 시 Model Train

Environment

- Python Version
- Based Image
- Dependency Layer
- Resources Limit

Project

- Source Code
- Dataset Path
- Training Script

실제 사례

Image 빌드 + 학습 동시 처리

Logistic Regression 모델 학습 시
초기 Build 1m 56s 을 기다려야 함

논리적 분리 이후

사용자가 Docker Image
빌드 기다림 없이 학습 가능

권한 관리를 어떻게 하는게 좋을까?

1. Permission 정의

2. Role 정의

3. User 와 Role 매핑

4. Resource-Scoped Policy 구현

감사합니다