

웹 애플리케이션과 기본 운영 방법

2025.03.27

권 승 도



INDEX

1. 강사소개
2. 회사소개
3. 웹 애플리케이션 소개
4. 웹 애플리케이션 통신
5. 웹 애플리케이션 운영 방법
6. FastAPI를 이용한 모듈 관리
7. Postman을 이용한 API 통신
8. NginX를 이용한 로드밸런싱
9. Jmeter를 이용한 부하 테스트
10. 참고

강사 소개

❖ *aectarine@naver.com*

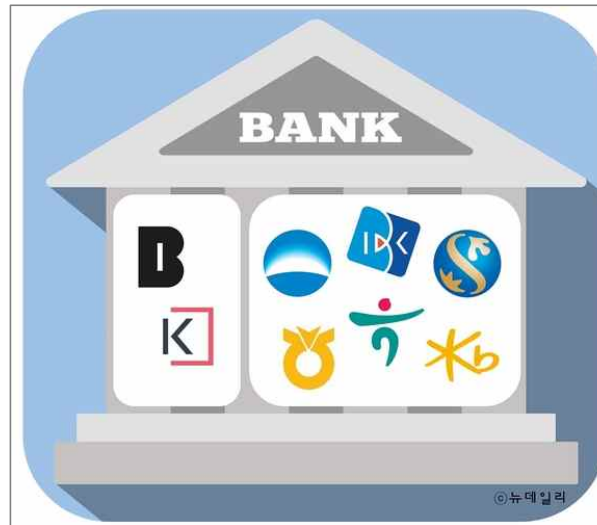


강사 소개

❖ 프로젝트 소개

• EDI 시스템 관리

- 전자문서 교환 (Electronic Data Interchange)
- 비즈니스 또는 거래 파트너 간의 문서를 전자적으로 교환
- 각 기업의 서로 다른 형식의 문서를 국제 표준 문서 구조로 변경하여 주고 받는 시스템



강사 소개

❖ 프로젝트 소개

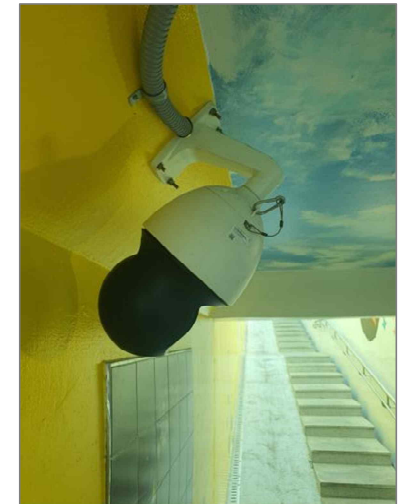
- AI를 이용한 지하공동구 작업자 동작감지 시스템
 - CCTV를 통해 작업자의 쓰러짐, 제스처를 AI로 감지 후 상황실 경고 알림 발생
 - CCTV를 통해 작업자의 동선 및 상황 파악



강사 소개

❖ 프로젝트 소개

- AI를 이용한 홍수대비 경보 알림 시스템
 - CCTV를 통해 홍수 빈번 발생지역 모니터링
 - 수위센서 및 AI 감지를 통한 상황실에 경고 알림



회사 소개

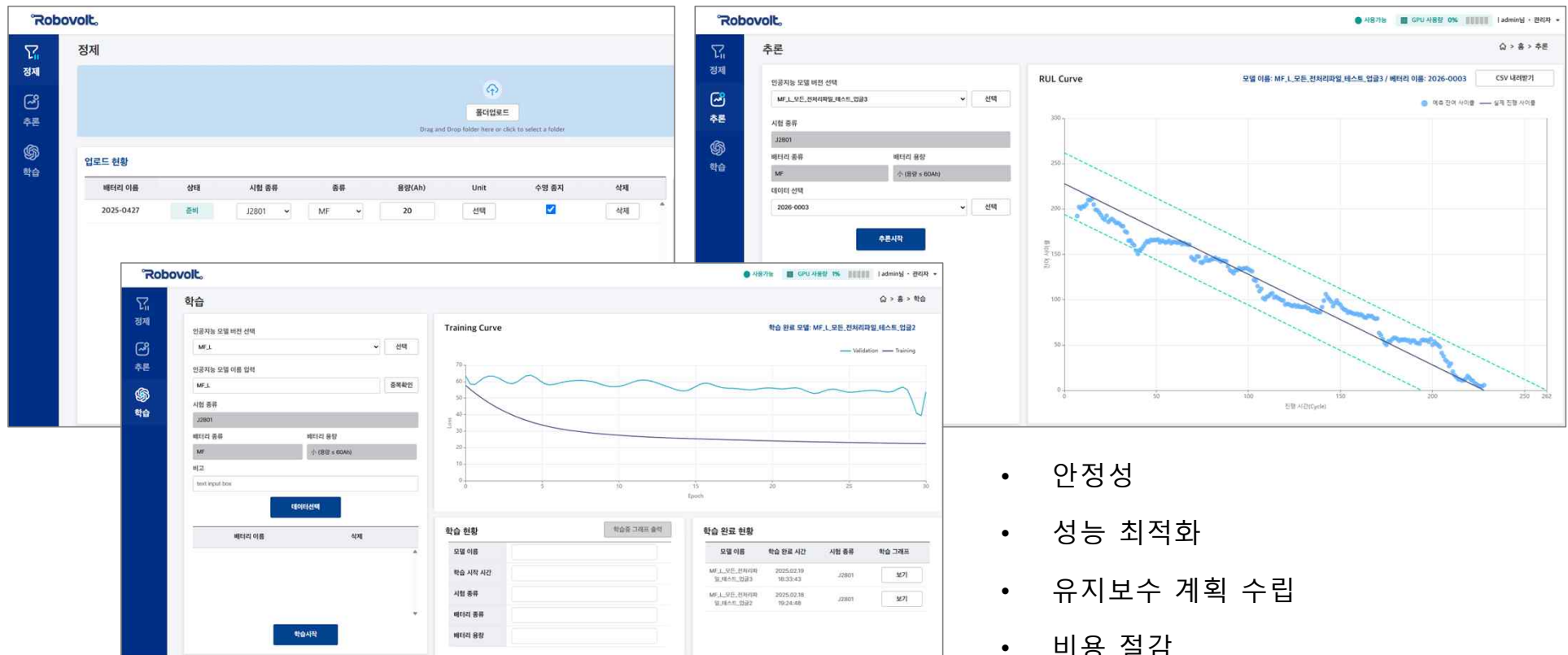
❖ (주)로보볼트

- 납축/리튬 배터리의 잔존수명 및 불량예측 AI 개발



회사 소개

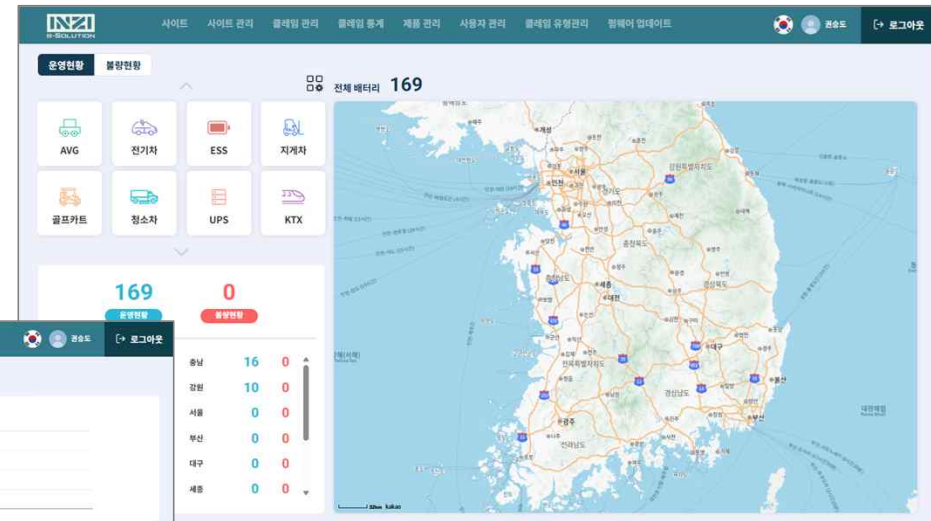
❖ 프로젝트 소개 납축 배터리 잔여수명(RUL) 예측 AI 모델 개발



- 안정성
- 성능 최적화
- 유지보수 계획 수립
- 비용 절감

회사 소개

❖ 프로젝트 소개 리튬 배터리 잔여수명(RUL) 예측 AI 모델 개발



- 안정성
- 성능 최적화
- 유지보수 계획 수립
- 비용 절감

웹 애플리케이션 소개

❖ 웹 애플리케이션 필요성

- PyQT (GUI or 원폼)를 넘어선 **디자인의 다양성**
- 백엔드와 프론트엔드 개발을 분리하여 **동시 작업** 가능
- 인공지능 개발자와 서비스 개발자의 **협업**을 위한 **이해**
- 채용시 웹 + 인공지능 동시 개발 가능 또는 이해도가 있는 **인재 선호 증가**
- IT 인으로서 **다양한 기술 소양과 경험** 보유
- AI 기술의 온라인 서비스를 통한 **기업의 이윤 창출**

웹 애플리케이션 소개

❖ Web Application 개발 프레임워크

웹 프론트엔드 프레임워크



언어 (플랫폼)



웹 백엔드 프레임워크



ORM (SQL-Mapper)



Spring Data JPA

Query DSL



JavaScript



EXPRESS



Prisma



django



FastAPI

SQLAlchemy



ASP.NET



DataBase



ORACLE
DATABASE



웹 애플리케이션 소개

❖ *Web Application* 구성

- 웹 애플리케이션

인터넷을 통해 웹 브라우저에서 이용할 수 있는 프로그램

- 웹 서버 (Web Server)

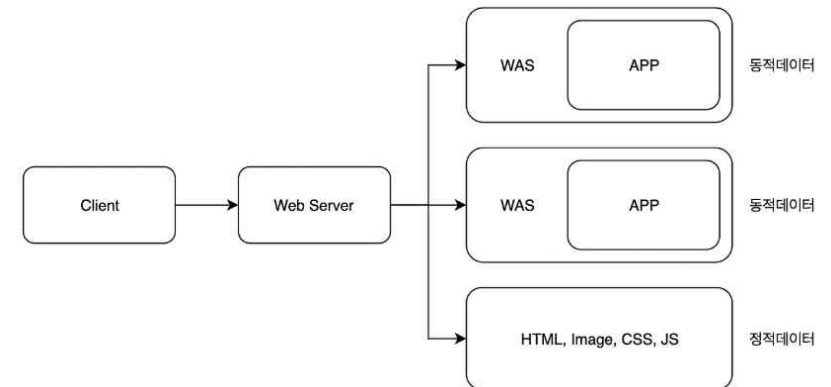
클라이언트가 페이지 요청시 **정적 페이지** (HTML, 이미지, CSS 등) 제공

- Apache HTTP Server, NginX, IIS 등

- 웹 애플리케이션 서버 (Web Application Server)

HTML 만으로 할 수 없는 DB 처리, 비즈니스 로직 처리 등 다양한 **동적 콘텐츠** 제공

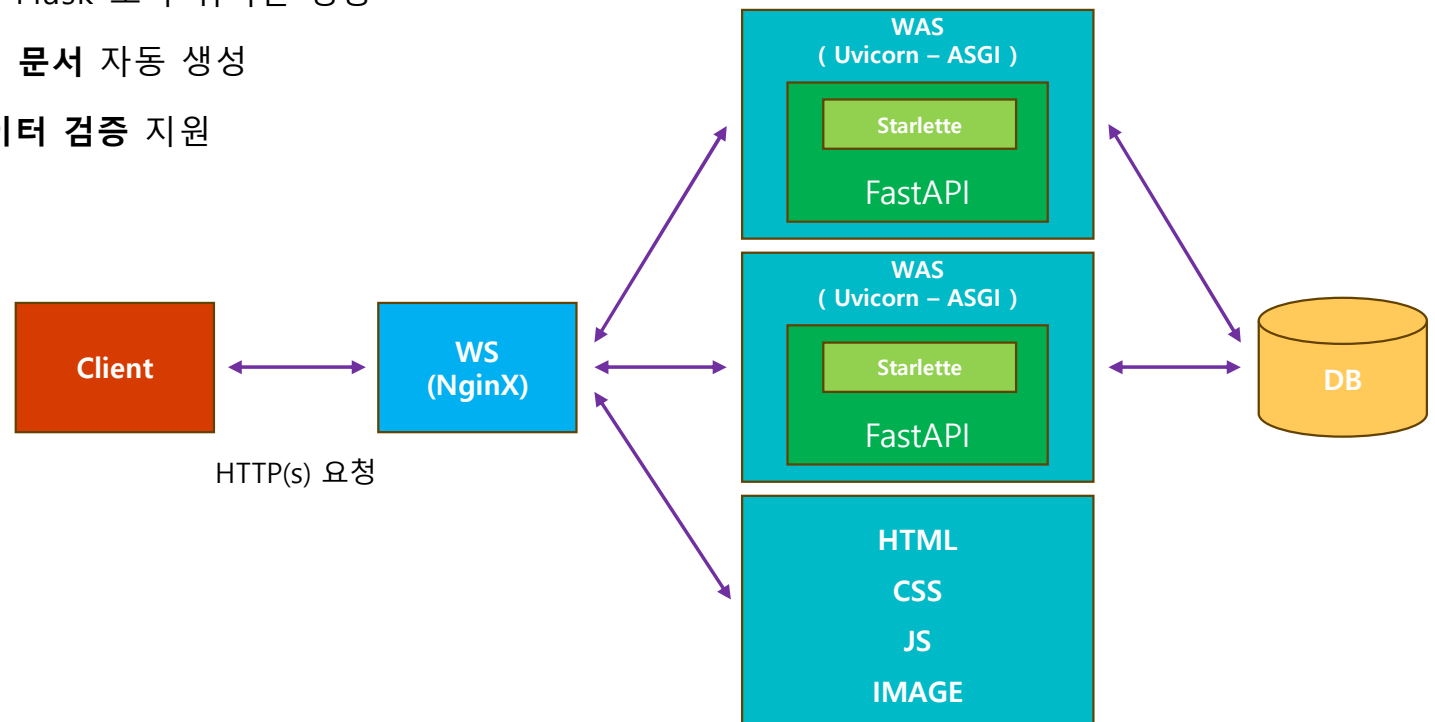
- Apache Tomcat, Jeus



웹 애플리케이션 소개

❖ *FastAPI*란?

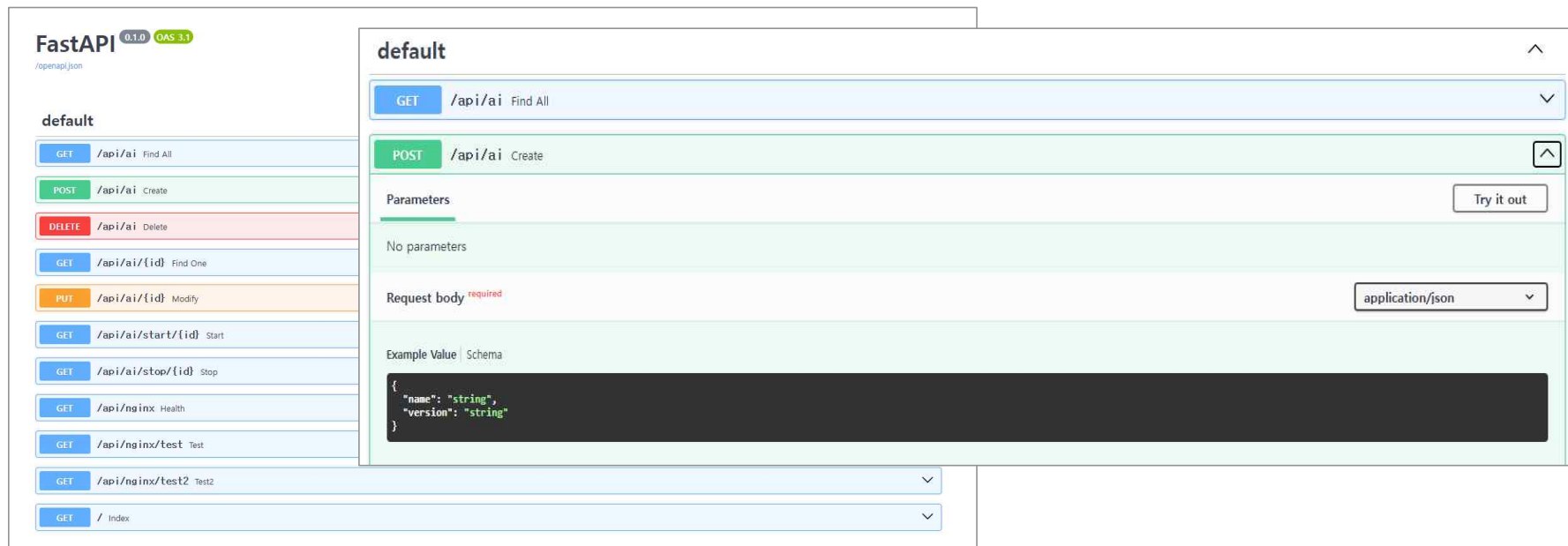
- 현대적이고 빠른 (고성능) 웹 애플리케이션 서버를 구축하기 위한 파이썬 웹 프레임워크
- Django 보다 가볍고, Flask 보다 뛰어난 성능
- Swagger를 통한 **API 문서** 자동 생성
- Pydantic을 통한 **데이터 검증** 지원
- ORM 지원
- **비동기** 처리 지원



웹 애플리케이션 소개

❖ *Swagger*

- 프로그램 라우터에 등록된 모든 API 목록 조회
- Swagger를 통한 API 문서 자동 생성
- (<http://127.0.0.1:8000/docs>)



웹 애플리케이션 소개

❖ *Pydantic*

- API 호출시 데이터가 올바른지 **유효성(Validation) 검증**
- 서버의 요청/응답시 대체 객체를 사용하여 보안 강화 (DTO 또는 VO)
- DB 조회 데이터의 자동 변환 가능

```
class AIModuleRequest(BaseModel):
    name: str = Field(...) # NN
    version: str = Field(default='2.0.0', min_length=5, max_length=20)

ksd

class AIModuleResponse(BaseModel):
    model_config = ConfigDict(from_attributes=True)
    id: int
    name: str
    version: str
    status: str
    inserted: datetime
    updated: datetime
```

```
1 {
2   "name": "테스트 AI - 1",
3   "version": "1.0.012312312312321312312321233"
4 }
```

Body Cookies Headers (4) Test Results (1)

{ } JSON Preview Visualize

```
1 {
2   "detail": [
3     {
4       "type": "string_too_long",
5       "loc": [
6         "body",
7         "version"
8       ],
9       "msg": "String should have at most 20 characters",
10      "input": "1.0.012312312312321312312321233",
11      "ctx": {
12        "max_length": 20
13      }
14    }
15  ]
16 }
```

웹 애플리케이션 소개

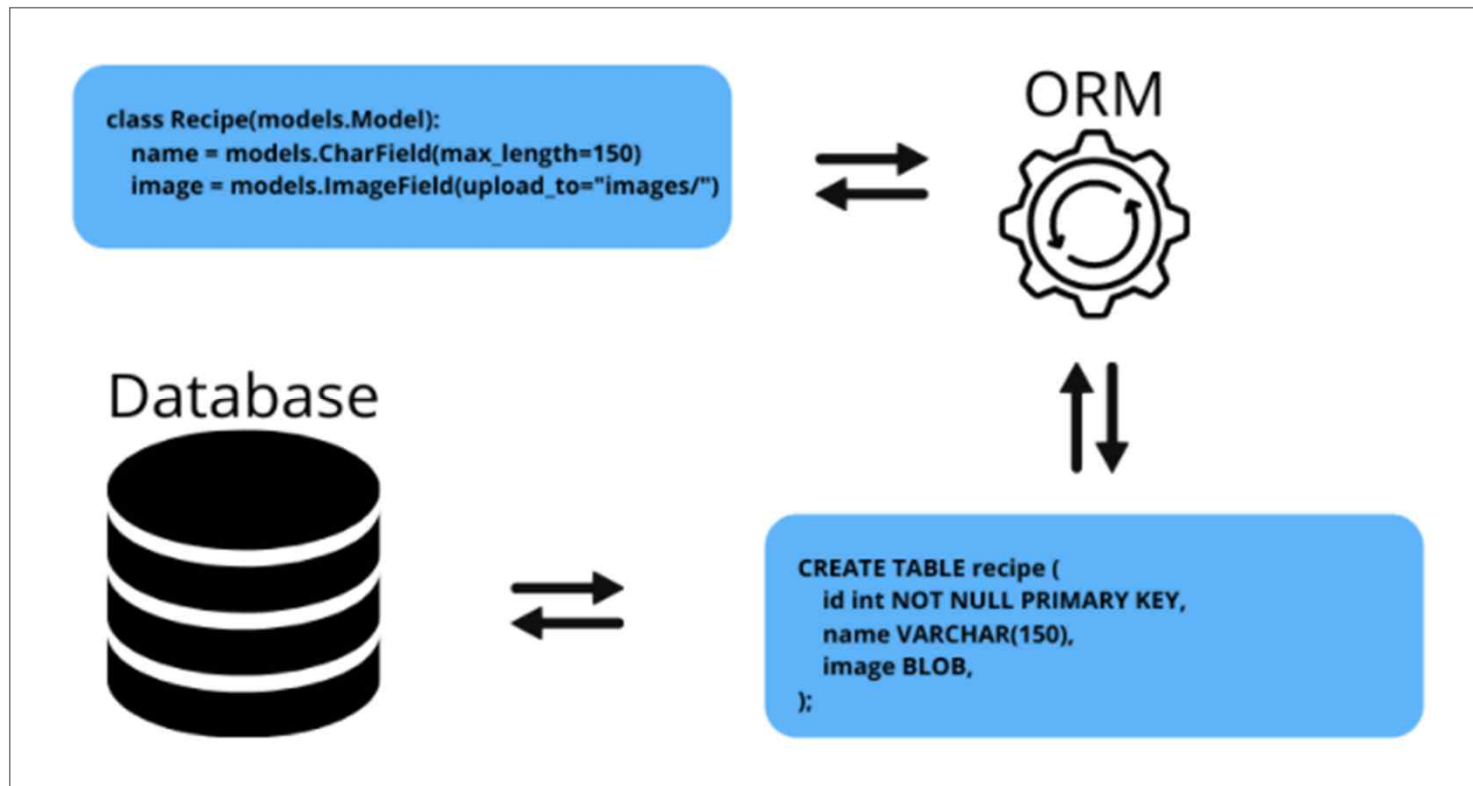
❖ *ORM*

- ORM (Object-Relational Mapping)은 객체 지향 프로그래밍(OOP)에서 사용하는 **객체**와 관계형 데이터베이스(RDB)의 **테이블**을 **매핑**(연결)하는 기술
- **SQL 쿼리 없이** 객체를 이용해서 데이터베이스 조작
- DB 종류에 관계없이 **하나의 구조로 개발** 가능



웹 애플리케이션 소개

❖ ORM 예시



웹 애플리케이션 소개

❖ ORM 예시 코드

```
ksd +1
class PreprocessInfo(Base):
    __tablename__ = 'preprocess_info'
    id = Column(BigInteger, primary_key=True, index=True)
    name = Column(String, nullable=False)
    e_type = Column(Enum(ExamType), nullable=False, default=ExamType.J2801, index=True)
    b_type = Column(Enum(BatteryType), nullable=False, default=BatteryType.MF, index=True)
    ah = Column(Integer, nullable=False, default=0)
    total_unit = Column.ARRAY(Integer, nullable=False, default=[])
    output_dir = Column(String)
    inserted = Column(DateTime, default=datetime.now)
    updated = Column(DateTime, default=func.now(), onupdate=func.now())
    # updated = Column(DateTime, default=datetime.now, onupdate=datetime.now())
    units = relationship('PreprocessUnit', back_populates='info', cascade='all, delete-orphan')
```

```
rs = await db.execute(select(AI_Module).order_by(AI_Module.id.asc()))
find_ai_modules = rs.scalars().all()
return [AI_ModuleResponse.model_validate(ai) for ai in find_ai_modules]
```

preprocess_info	
columns 9	
id	bigint = nextval(preproces...)
name	varchar
e_type	examtype
b_type	batterytype
ah	integer
total_unit	integer[]
output_dir	varchar
inserted	timestamp
updated	timestamp
> keys 1	
> indexes 4	

웹 애플리케이션 통신

❖ URI와 구성

- `http://www.naver.com:80/hanbat/index.html?key1=value1&key2=value2`
- 프로토콜: `http` 또는 `https`
- 호스트: `www.naver.com` 또는 `192.168.0.100`
- 포트: `:80` (`http` 기본 포트, `https`는 433)
- 경로: `/hanbat/index.html` 또는 `/api/hanbat/ai`
- 쿼리 파라미터(쿼리 스트링): `?key1=value1&key2=value2`
 - 검색조건: `?filter=id:1,2&sort=id:desc`
 - 페이징: `?page=1&size=10`

웹 애플리케이션 통신

❖ HTTP 프로토콜

- HTTP (Hyper Text Transfer Protocol)
- 웹 환경에서 클라이언트와 서버 사이의 요청과 응답 통신을 위해 설계된 프로토콜
- 비연결형
- 단방향 통신
- Request (요청) / Response (응답) 구조
- 요청 종류에 따라 GET (조회) / POST (등록) / PUT (수정) / DELETE (삭제) 등 메소드 사용
- 응답 코드 존재. Ex) 성공 - 200, 잘못된 요청 - 400, 페이지 찾을 수 없음 - 404, 서버 에러 - 500
- 요청 및 응답 형식은 회사 개발 규칙에 따름
- 보안 필요시 HTTPS 프로토콜과 인증서 사용

웹 애플리케이션 통신

❖ HTTP 프로토콜 - GET 메소드

➤ 요청 (Request)

Start Line

GET /api/ai?id=1 HTTP/1.1

Headers

Host: example.com

User-Agent: Mozilla/5.0

...

Body

➤ 응답 (Response)

Status Line

HTTP/1.1 200 OK

Header

Content-Type: application/json

Content-Length: 30

...

Body

```
{  
  "id": 1,  
  "data": "id 1번 값의 데이터 "  
}
```

웹 애플리케이션 통신

❖ HTTP 프로토콜 - POST 메소드

➤ 요청 (Request)

Start Line

GET /api/ai HTTP/1.1

Headers

Host: example.com

User-Agent: Mozilla/5.0

Content-Type: application/x-www-form-urlencoded

Content-Length: 27

...

Body

name=AI_Module-1&version=1.0.0

➤ 응답 (Response)

Status Line

HTTP/1.1 201 Created

Header

Content-Type: application/json

Content-Length: 30

...

Body

```
{  
  "id": 1,  
  "result": "등록 완료 "  
}
```

웹 애플리케이션 통신

❖ **HTTP 프로토콜 - PUT 메소드** (PATCH 방식도 있으나, 지원하지 않는 서버들이 있음)

➤ 요청 (Request)

Start Line

GET /api/ai?id=1 HTTP/1.1

Headers

Host: example.com

User-Agent: Mozilla/5.0

Content-Type: application/json

Content-Length: 47

...

Body

{"name": "테스트-AI-1 수정", "version": "2.0.0"}

➤ 응답 (Response)

Status Line

HTTP/1.1 200 OK

Header

Content-Type: application/json

Content-Length: 30

...

Body

```
{  
  "id": 1,  
  "result": "수정 완료 "  
}
```

웹 애플리케이션 통신

❖ HTTP 프로토콜 - DELETE 메소드

➤ 요청 (Request)

Start Line

GET /api/ai?id=1 HTTP/1.1

Headers

Host: example.com

User-Agent: Mozilla/5.0

...

Body

➤ 응답 (Response)

Status Line

HTTP/1.1 200 OK

Header

Content-Type: application/json

Content-Length: 30

...

Body

```
{  
  "id": 1,  
  "result": "삭제 완료 "  
}
```

웹 애플리케이션 통신

❖ 주로 사용되는 데이터 형식: XML과 JSON

- **XML** (eXtensible Markup Language): 확장 가능한 마크업 언어
- **JSON** (Javascript Object Notation): 자바스크립트 객체 표기법

XML

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <endereco>
3   <cep>31270901</cep>
4   <city>Belo Horizonte</city>
5   <neighborhood>Pampulha</neighborhood>
6   <service>correios</service>
7   <state>MG</state>
8   <street>Av. Presidente Antônio Carlos, 6627</street>
9 </endereco>
```

vs.

JSON

```
1 {
2   "endereco": {
3     "cep": "31270901",
4     "city": "Belo Horizonte",
5     "neighborhood": "Pampulha",
6     "service": "correios",
7     "state": "MG",
8     "street": "Av. Presidente Antônio Carlos, 6627"
9   }
10 }
```

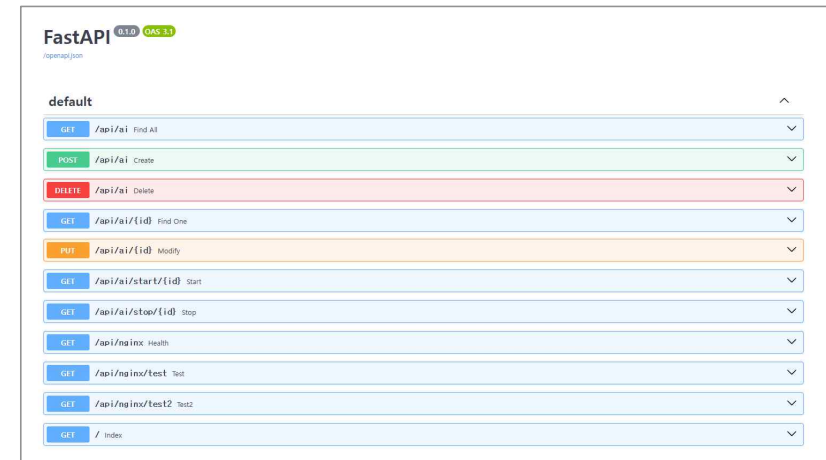
웹 애플리케이션 통신

❖ API

- Application Programing Interface
- S/W 또는 애플리케이션 간의 **통신 인터페이스 규칙**

❖ REST API

- **HTTP, URI 구성, 데이터**를 이용하여 웹에서 서버와 클라이언트 간의 데이터 **통신 인터페이스**를 구현한 것
- / (슬래쉬)는 계층관계 표현
- _ (언더바)는 사용X
- 대문자는 피하기 등
- 일반적으로 웹 개발시 **API 개발은 REST API 개발을 의미**



웹 애플리케이션 통신

❖ *WS(Web Socket) 프로토콜*

- 웹 환경에서 클라이언트와 서버 사이의 통신을 위해 설계된 프로토콜
- 일반적인 소켓 통신과 유사
- 연결형
- 양방향 통신
- 메시지 형식은 회사 개발 규칙에 따름
- 보안 필요시 **WSS** 프로토콜과 인증서 사용
- 실시간 채팅 / 영상 스트리밍 / 주기적 데이터 전송 (GPU 사용률 등)

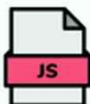
웹 애플리케이션 운영 방법

❖ 다양한 서비스 운영 방법

언어 (플랫폼)



실행 파일



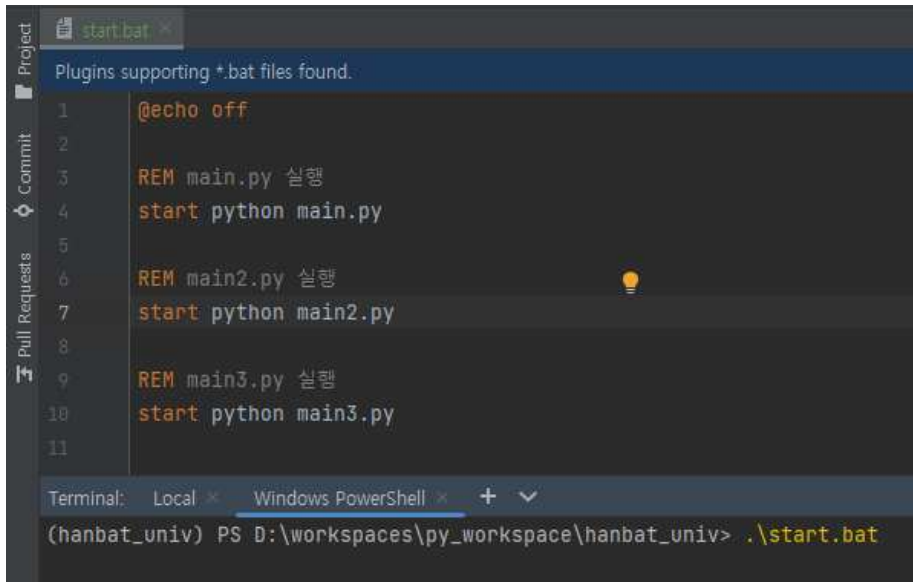
운영 방법



웹 애플리케이션 운영 방법

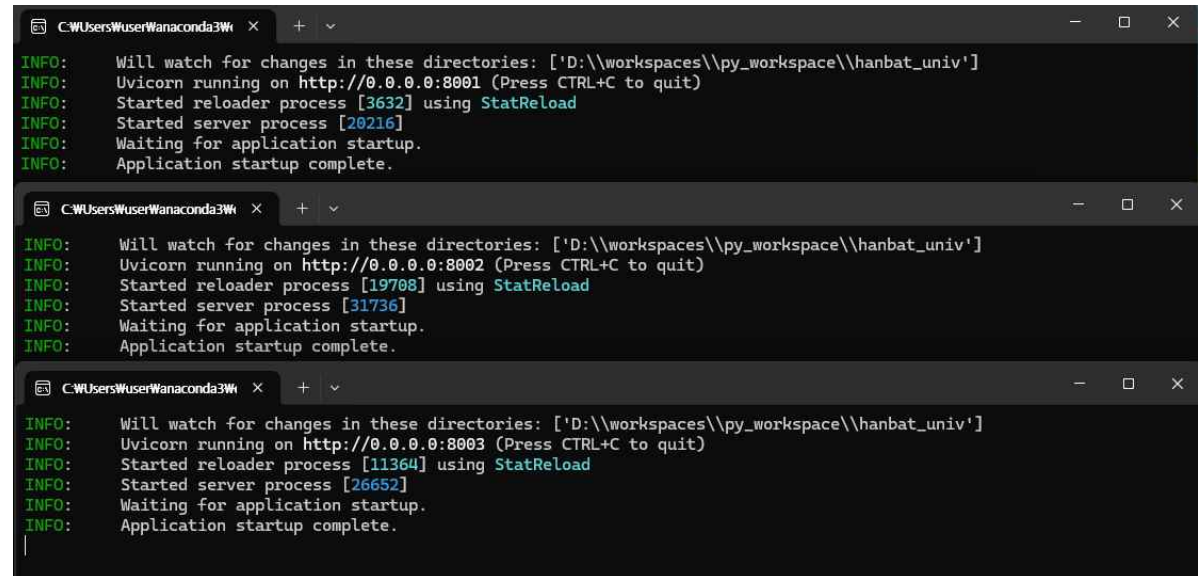
❖ 배치파일

- Windows 운영 체제에서 명령을 순차적으로 실행할 수 있는 텍스트 스크립트 파일
- [실행파일명].bat



```
start.bat
Plugins supporting *.bat files found.
1 @echo off
2
3 REM main.py 실행
4 start python main.py
5
6 REM main2.py 실행
7 start python main2.py
8
9 REM main3.py 실행
10 start python main3.py
11

Terminal: Local x Windows PowerShell x + v
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> .\start.bat
```



```
C:\Users\User\Wanaconda3\W... x + v
INFO: Will watch for changes in these directories: ['D:\\workspaces\\py_workspace\\hanbat_univ']
INFO: Uvicorn running on http://0.0.0.0:8001 (Press CTRL+C to quit)
INFO: Started reloader process [3632] using StatReload
INFO: Started server process [20216]
INFO: Waiting for application startup.
INFO: Application startup complete.

C:\Users\User\Wanaconda3\W... x + v
INFO: Will watch for changes in these directories: ['D:\\workspaces\\py_workspace\\hanbat_univ']
INFO: Uvicorn running on http://0.0.0.0:8002 (Press CTRL+C to quit)
INFO: Started reloader process [19708] using StatReload
INFO: Started server process [31736]
INFO: Waiting for application startup.
INFO: Application startup complete.

C:\Users\User\Wanaconda3\W... x + v
INFO: Will watch for changes in these directories: ['D:\\workspaces\\py_workspace\\hanbat_univ']
INFO: Uvicorn running on http://0.0.0.0:8003 (Press CTRL+C to quit)
INFO: Started reloader process [11364] using StatReload
INFO: Started server process [26652]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

웹 애플리케이션 운영 방법

❖ *FOREVER*

- Node.js 애플리케이션을 위한 운영 관리 프로그램
- npm 패키지를 통해 설치
- `forever start -c [컴파일러] | forever list | forever delete [id]`

```
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> forever start -c python main.py
warn: --minUptime not set. Defaulting to: 1000ms
warn: --spinSleepTime not set. Your script will exit if it does not stay up for at least 1000ms
info: Forever processing file: main.py
(node:24692) Warning: Accessing non-existent property 'padLevels' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
(node:24692) Warning: Accessing non-existent property 'padLevels' of module exports inside circular dependency
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> forever list
(node:4712) Warning: Accessing non-existent property 'padLevels' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
(node:4712) Warning: Accessing non-existent property 'padLevels' of module exports inside circular dependency
info: Forever processes running
data:      uid  command script                                forever pid  id logfile
data:  [0] D5UF python  D:\workspaces\py_workspace\hanbat_univ\main.py 15584  7988  C:\Users\user\.forever\D5UF.log
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ>
```

```
C:\Users\Wuser\Wanaconda3\W... X + v
INFO: Started server process [29672]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

웹 애플리케이션 운영 방법

❖ PM2

- Node.js 애플리케이션을 위한 운영 관리 프로그램 (Forever + 알파)
- npm 패키지를 통해 설치
- `pm2 start [실행파일] --interpreter [인터프리터] | pm2 list | pm2 monit [id] | pm2 logs`
- `pm2 logs [id] | pm2 delete [id]`

```
([FAILING] tailing last 15 lines for [0] process (change the value with --lines option)
C:\Users\user\.pm2\logs\main-out.log last 15 lines:

Process List | Logs
-----|-----
In memory PM2 version: 5.4.3
Local PM2 version: 4.5.6

id  name  namespace  version  mode  pid  uptime  ⚡  status  cpu  mem  user  watching
--  --  --  --  --  --  --  --  --  --  --  --  --
[PM2] Starting D:\workspaces\py_workspace\hanbat_univ\main.py in fork_mode (1 instance)
[PM2] Done.
```

id	name	namespace	version	mode	pid	uptime	⚡	status	cpu	mem	user	watching
0	main	default	N/A	fork	16632	0s	0	online	0%	35.1mb	user	disabled

```
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ>
```

```
C:\Users\user\Wanaconda3\W... x + v
INFO: Started server process [29672]
INFO: Waiting for application startup.
INFO: Application startup complete.
```

웹 애플리케이션 운영 방법

❖ 도커

- 도커 이미지를 이용하여 컨테이너 생성
- 개별 컨테이너를 단순 직접 관리
- 개별 포트번호를 직접 지정하여 동일한 여러 서버 구동

```
Dockerfile
1 FROM python:3.8-slim
2
3 WORKDIR /app
4
5 COPY requirements.txt .
6 RUN pip install --upgrade pip
7 RUN pip install -r requirements.txt
8
9 COPY . .
10
11 EXPOSE 8001
12
13 CMD ["python", "main.py"]
14
```

```
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> docker run -d main
bb28df927207c1782a834f58de8064e6e779113bdf5169f3d945b244eaba6c2d
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS          NAMES
bb28df927207   main     "python main.py"        2 seconds ago Up 1 second   8001/tcp      gracious_heyrovsky
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> docker logs -f bb28
INFO: Will watch for changes in these directories: ['/app']
INFO: Uvicorn running on http://0.0.0.0:8001 (Press CTRL+C to quit)
INFO: Started reloader process [1] using StatReload
INFO: Started server process [9]
INFO: Waiting for application startup.
INFO: Application startup complete.
```


웹 애플리케이션 운영 방법

❖ 도커 스웸

- 여러 서버에서 컨테이너 배포 및 관리
- 컨테이너 복제본(Replica) 생성을 통한 무중단 서비스
- 로드밸런싱 및 부하분산

```
(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hanbat_img     latest    c2c08cb26da8   16 minutes ago 396MB
postgres       latest    0321e2252ebf   2 weeks ago   621MB
mysql          latest    146682692a3a   6 weeks ago   1.09GB
rabbitmq       3-management 5e147fe65a7e   5 months ago  394MB

(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> docker service ls
ID                NAME                MODE           REPLICAS        IMAGE              PORTS
unn7ir4obct2      hanbat_service       replicated      2/2             hanbat_img:latest  *:8001->8001/tcp
pbe3ev7g180k      hanbat_service_2     replicated      2/2             hanbat_img:latest  *:8002->8002/tcp
tnrucn10wkds      hanbat_service_3     replicated      2/2             hanbat_img:latest  *:8003->8003/tcp

(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED        STATUS        PORTS        NAMES
1568ea384cc6   hanbat_img:latest  "python main3.py"        13 minutes ago Up 13 minutes          hanbat_service.3.2.bkb1kqqiquchoub2mreu31t8p
c0eb042fc9f9   hanbat_img:latest  "python main3.py"        13 minutes ago Up 13 minutes          hanbat_service.3.1.lclb5rhcf8j2t6jqeuv63m
dfadb9bb41f0   hanbat_img:latest  "python main2.py"        13 minutes ago Up 13 minutes          hanbat_service.2.1.r0akhwutmj9r4c7s1kqk90dc0
686b9996bdb0   hanbat_img:latest  "python main2.py"        13 minutes ago Up 13 minutes          hanbat_service.2.2.ly9yin4kh0e0vbyuqesscz5mq
f8f9a44bc9c1   hanbat_img:latest  "python main.py"         14 minutes ago Up 14 minutes          hanbat_service.1.x74gxm6w0x42z3lh3krpwemxa
25becd3e9d6b   hanbat_img:latest  "python main.py"         14 minutes ago Up 14 minutes          hanbat_service.2.nqi0m4y4py1w6l5cqifog7ru6

(hanbat_univ) PS D:\workspaces\py_workspace\hanbat_univ> docker service create --name hanbat_service --publish 8001:8001 --replicas 2 hanbat_img:latest python main.py^Q
```

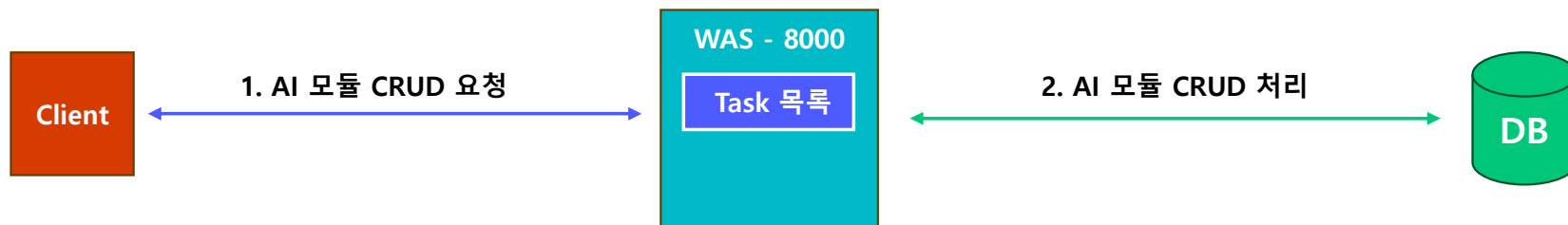
FastAPI를 이용한 모듈 관리

❖ 개요

- 내가 제작한 **AI 모듈**을 웹 사이트에서 **서비스로 관리**하기
- HTTP API 통신을 통해 모듈을 **CRUD 구현**하기
- HTTP API 통신을 통해 모듈의 **구동 상태**를 **START, STOP** 하기

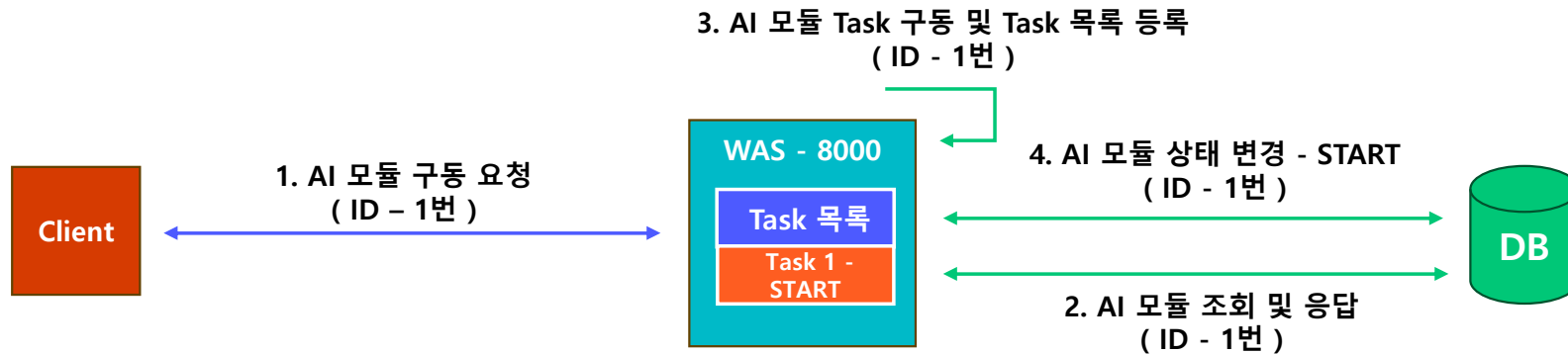
FastAPI를 이용한 모듈 관리

❖ 샘플 코드의 AI 모듈 CRUD 처리



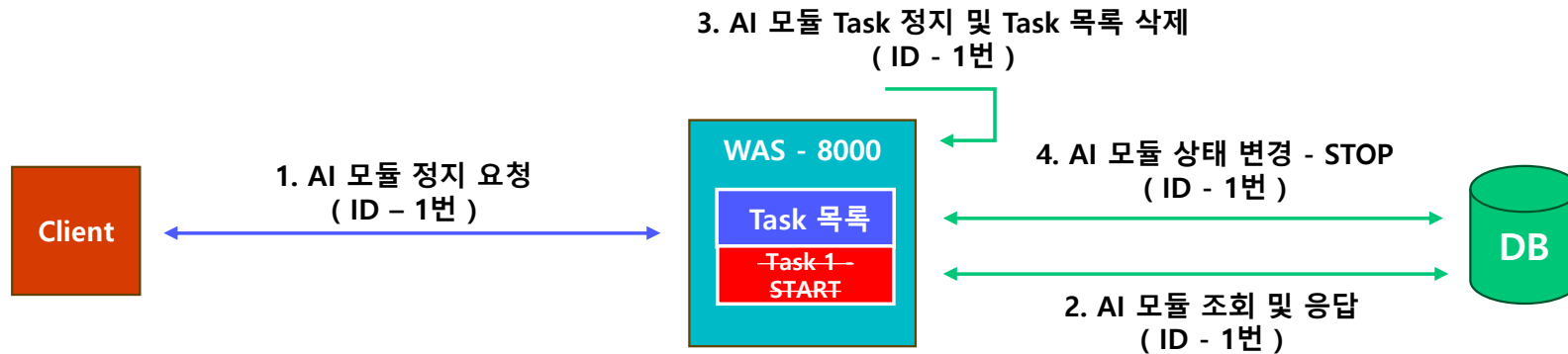
FastAPI를 이용한 모듈 관리

❖ 샘플 코드의 AI 모듈 구동 처리



FastAPI를 이용한 모듈 관리

❖ 샘플 코드의 AI 모듈 정지 처리



FastAPI를 이용한 모듈 관리

❖ 대시보드



FastAPI를 이용한 모듈 관리

❖ *FastAPI를 이용한 내 모듈관리 실습*

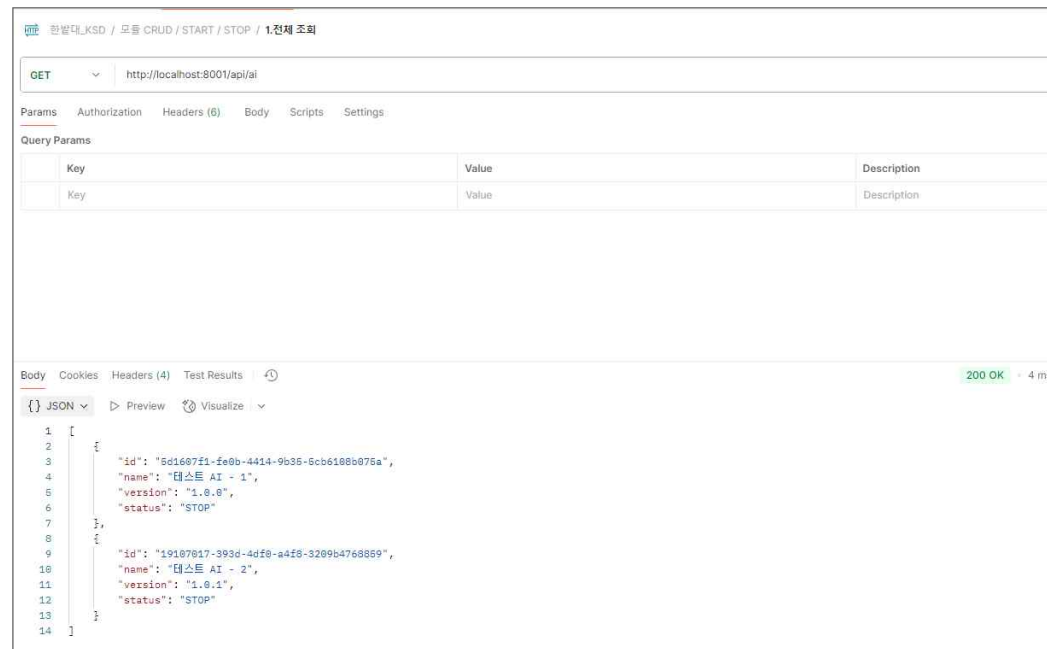
❖ 실습 참조 주소

<https://github.com/aectarine>

Postman을 이용한 API 통신

❖ *Postman*이란?

- API 개발 및 테스트를 위한 API 클라이언트 도구
- 무료 오픈소스



Postman을 이용한 API 통신

❖ *Postman을 이용한 API 통신 테스트*

❖ 참조 주소

<https://github.com/aectarine>

NginX를 이용한 로드밸런싱

❖ 로드밸런싱

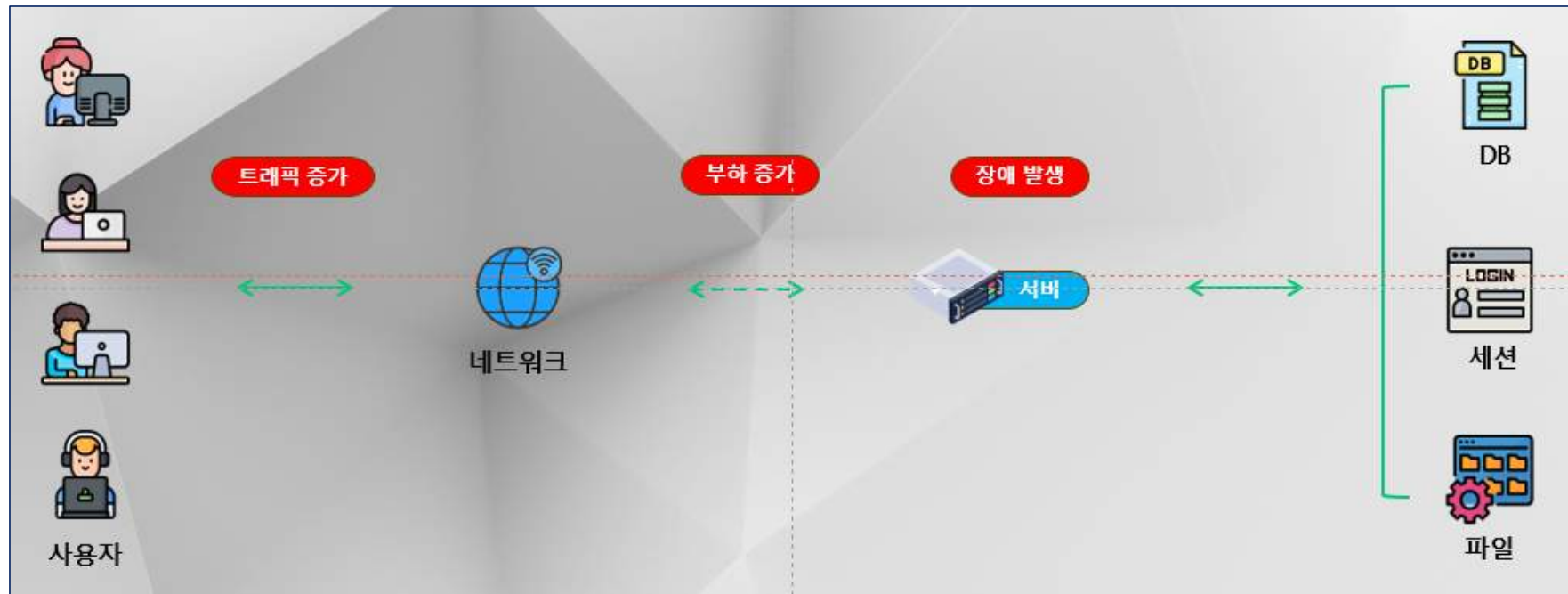
- 하나의 서버에 몰리는 트래픽을 분산시키는 기술
- 서버 과부하를 방지하고, 시스템 성능과 안정성을 향상
- 여러 서버 중 하나의 서버에 문제가 발생해도 다른 서버가 대응

❖ 로드밸런싱 알고리즘 종류

- 라운드 로빈 (Round Robin)
 - 순차적 요청 분배
- 가중 라운드 로빈 (Weighted Round Robin)
 - 서버 가중치 분배
- 최소 연결 (Least Connection)
- 최소 응답 시간 (Least Response Time)

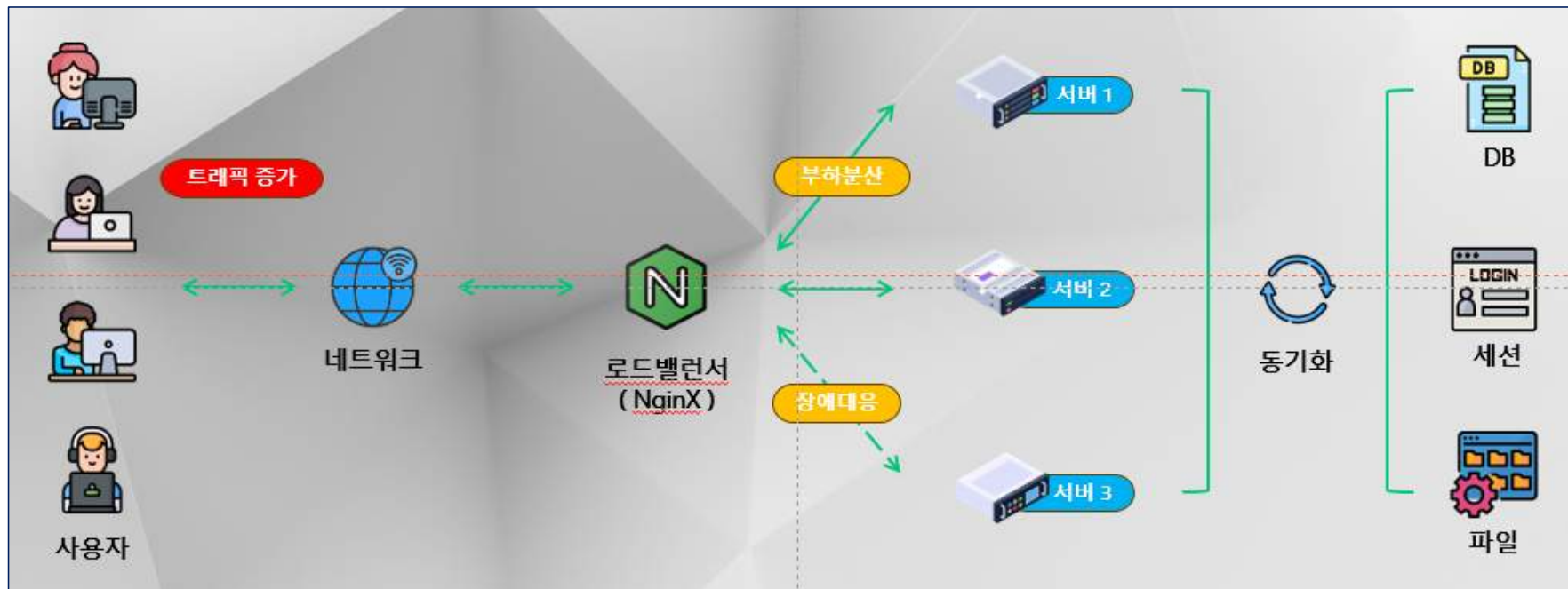
NginX를 이용한 로드밸런싱

❖ 로드밸런싱 적용되지 않은 경우



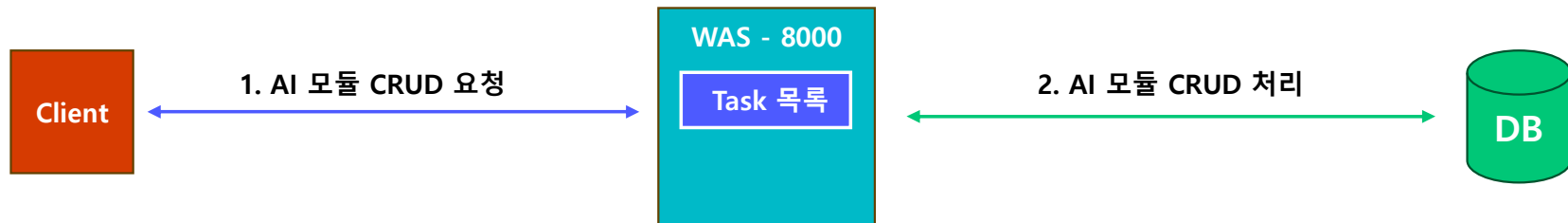
NginX를 이용한 로드밸런싱

❖ 로드밸런싱 적용된 경우



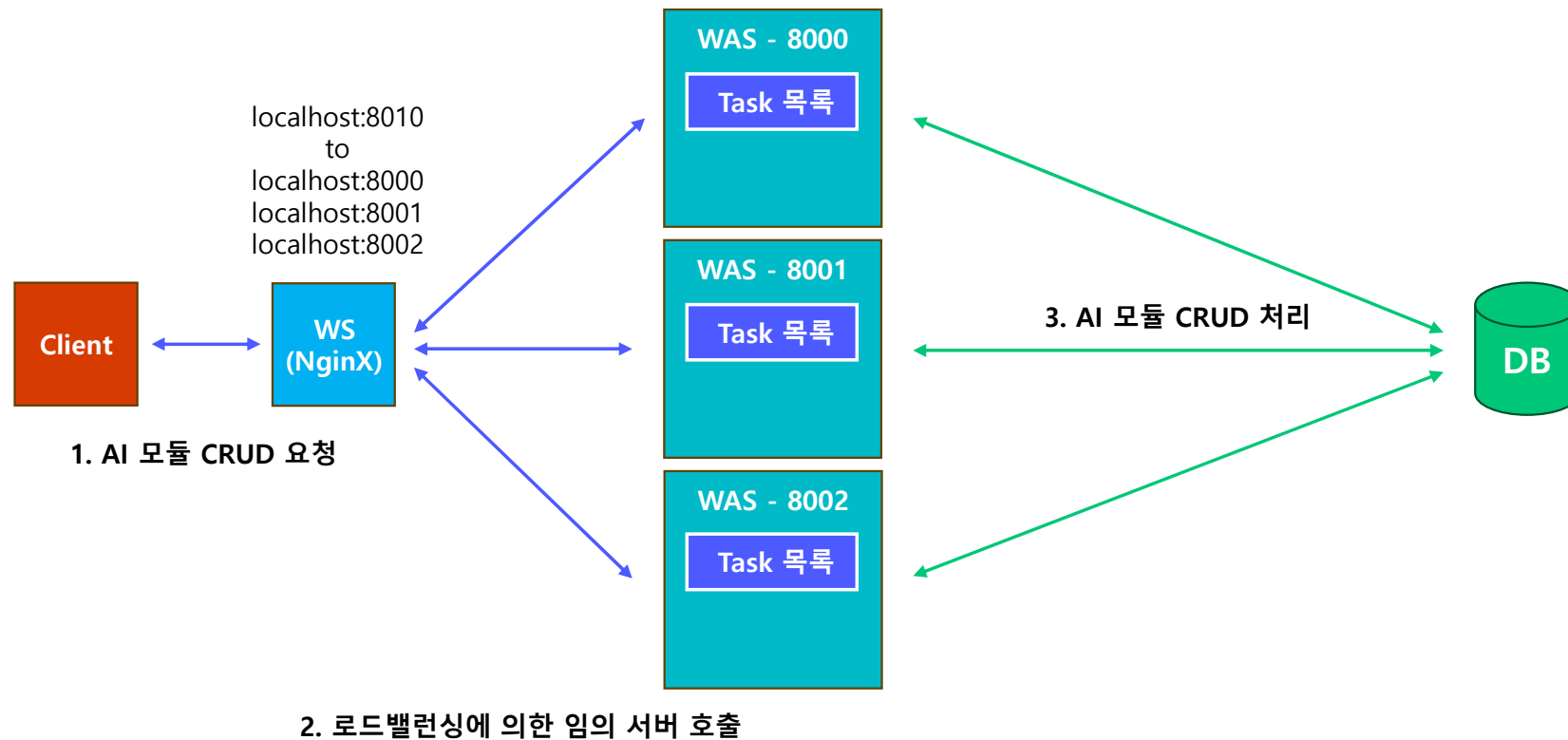
NginX를 이용한 로드밸런싱

❖ 샘플 코드의 AI 모듈 CRUD 처리



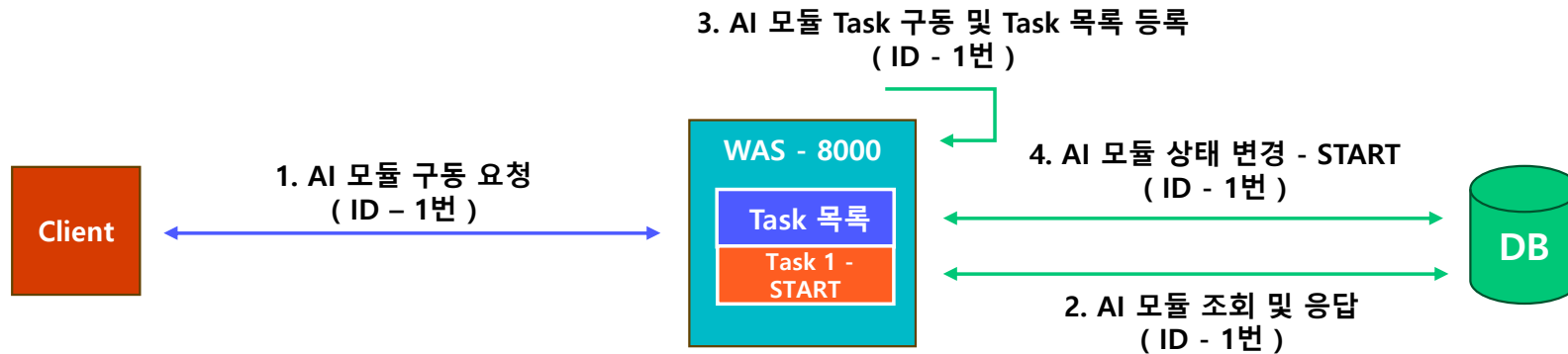
NginX를 이용한 로드밸런싱

❖ 로드밸런서를 적용한 AI 모듈 CRUD 처리



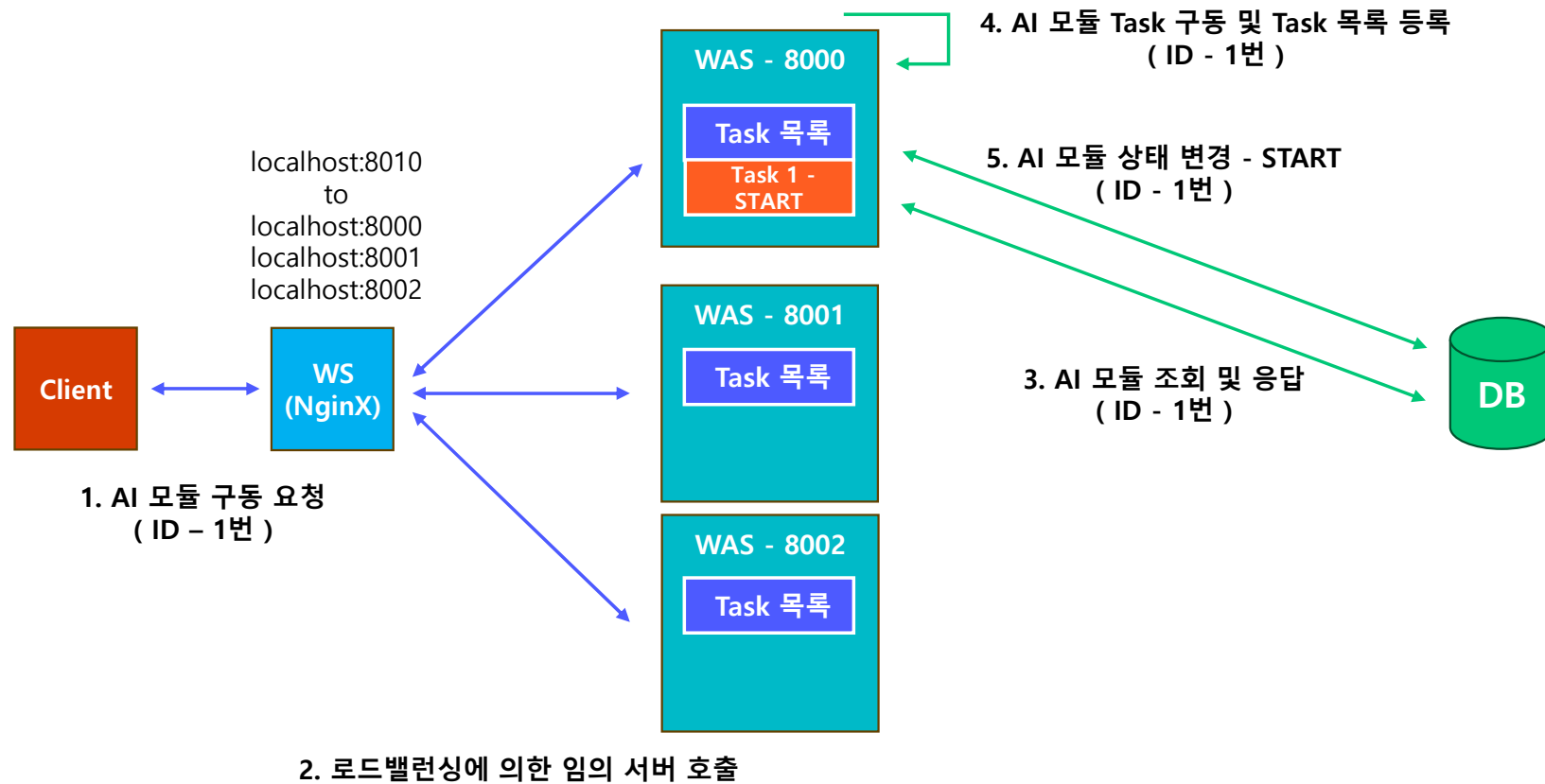
NginX를 이용한 로드밸런싱

❖ 샘플 코드의 AI 모듈 구동 처리



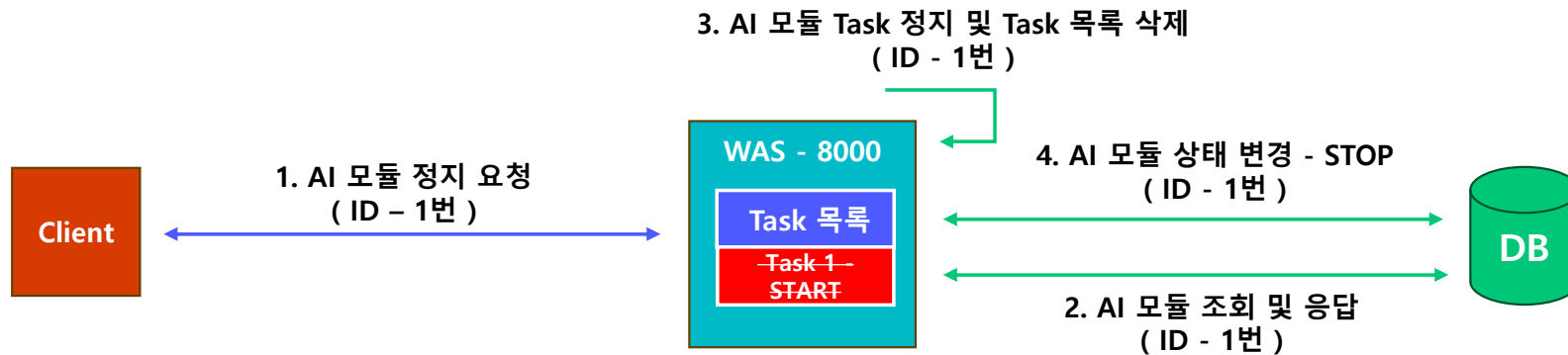
NginX를 이용한 로드밸런싱

❖ 로드밸런서를 적용한 AI 모듈 구동 처리



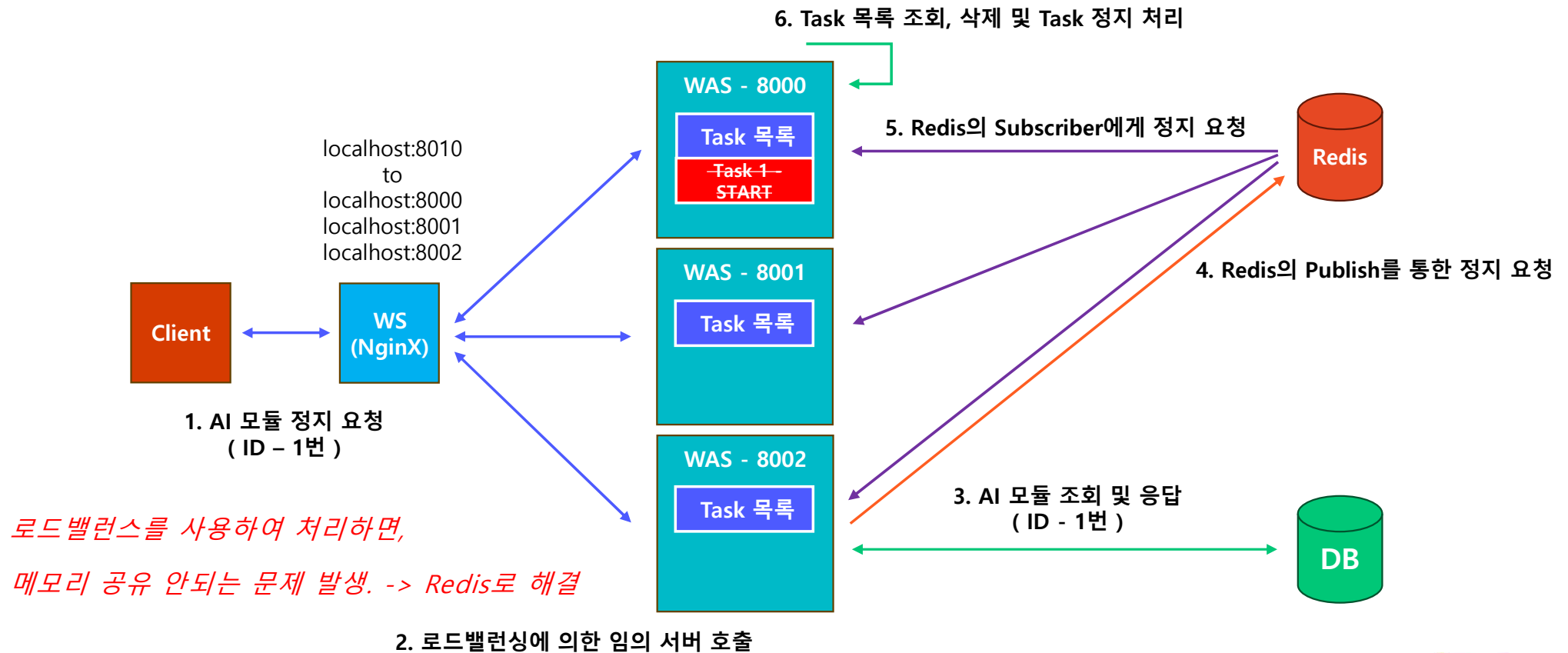
NginX를 이용한 로드밸런싱

❖ 샘플 코드의 AI 모듈 정지 처리



NginX를 이용한 로드밸런싱

❖ 로드밸런서를 적용한 AI 모듈 정지 처리



NginX를 이용한 로드밸런싱

❖ 로드밸런싱 실습

❖ 참조 주소

<https://github.com/aectarine>

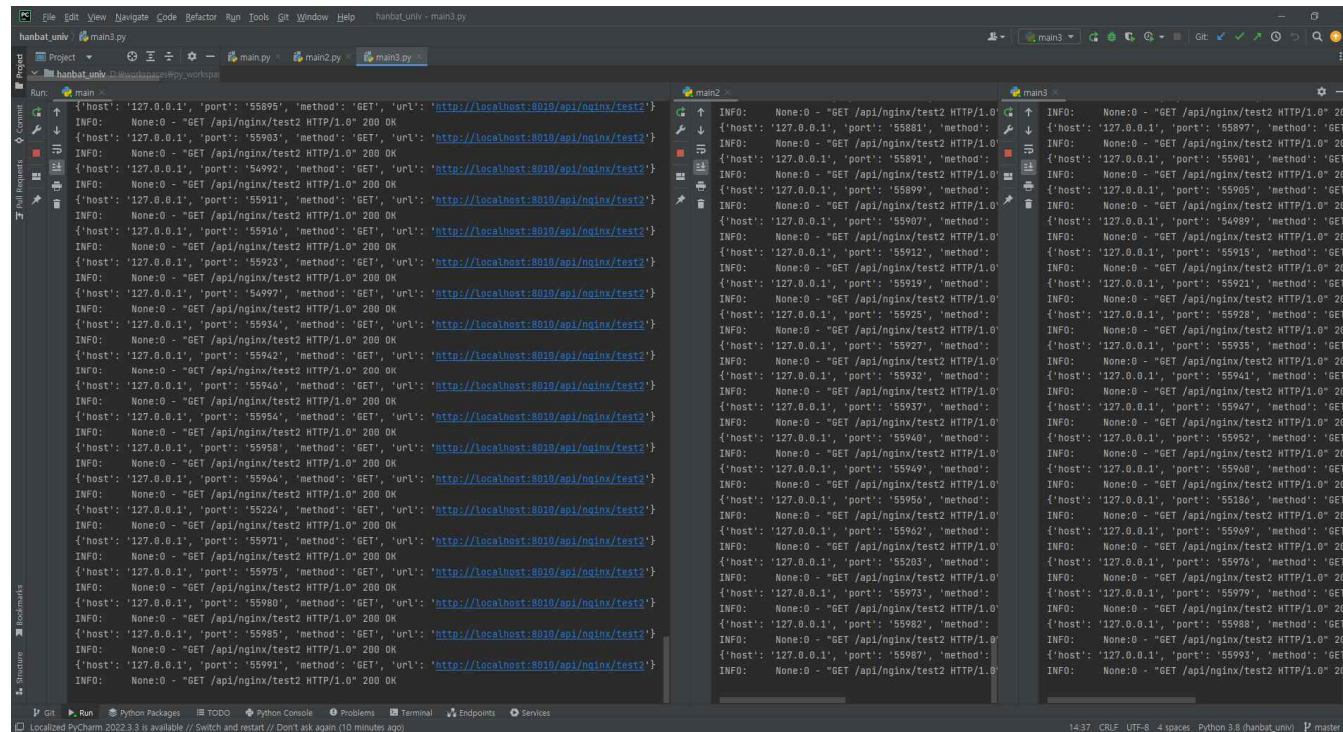
JMeter를 이용한 부하 테스트

❖ *JMeter란?*

- Apache 재단에서 개발한 성능 테스트 도구
- 프로젝트에서 고객이 원하는 웹 애플리케이션의 성능 테스트 가능
- 무료 오픈소스
- HTTP, HTTPS, WS, WSS, SOAP, MQTT 등 프로토콜 지원

JMeter를 이용한 부하 테스트

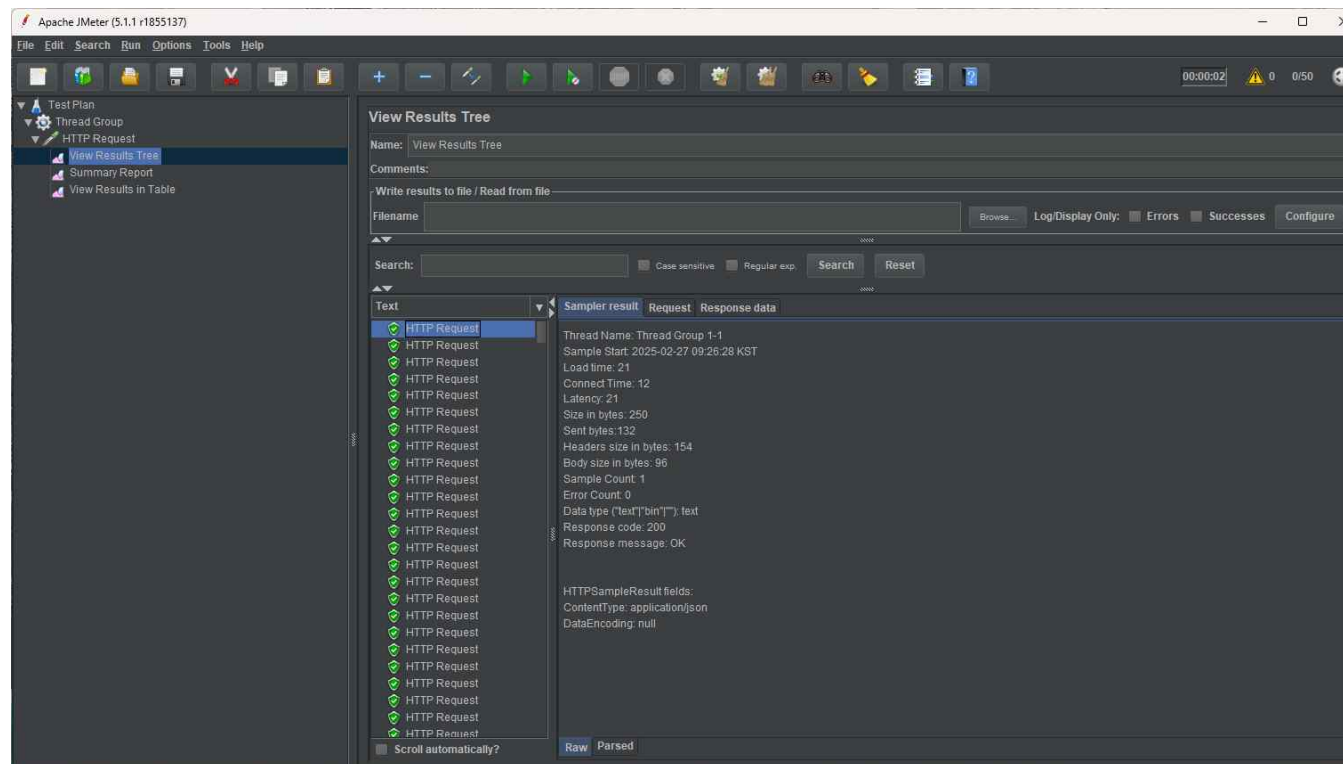
❖ JMeter 예시



```
main3.py
{
  'host': '127.0.0.1', 'port': '55895', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55903', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '54992', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55911', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55916', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55923', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '54997', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55934', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55942', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55946', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55954', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55958', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55964', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55224', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55971', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55975', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55980', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55985', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
{
  'host': '127.0.0.1', 'port': '55991', 'method': 'GET', 'url': 'http://localhost:8010/api/nginx/test2'
}
INFO: None:0 - "GET /api/nginx/test2 HTTP/1.0" 200 OK
```

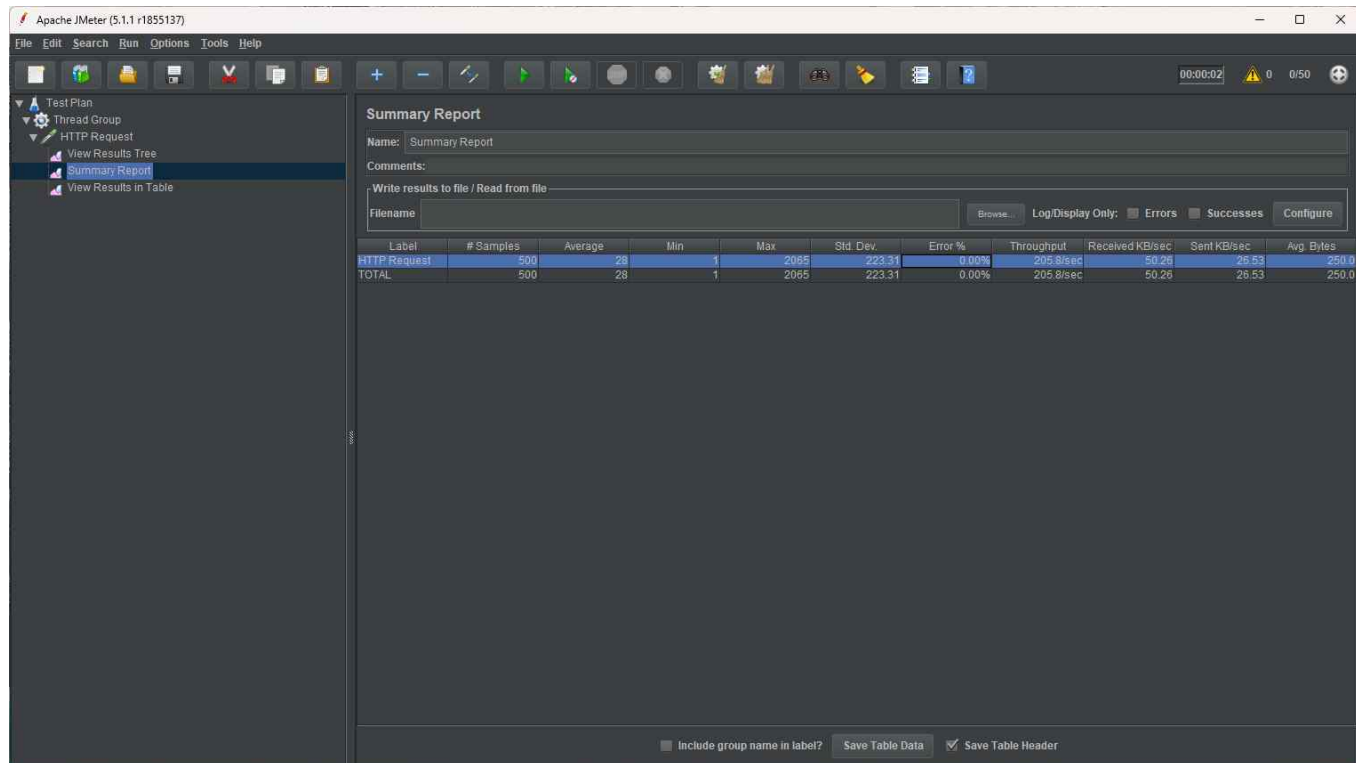
JMeter를 이용한 부하 테스트

❖ JMeter 예시



JMeter를 이용한 부하 테스트

❖ JMeter 예시



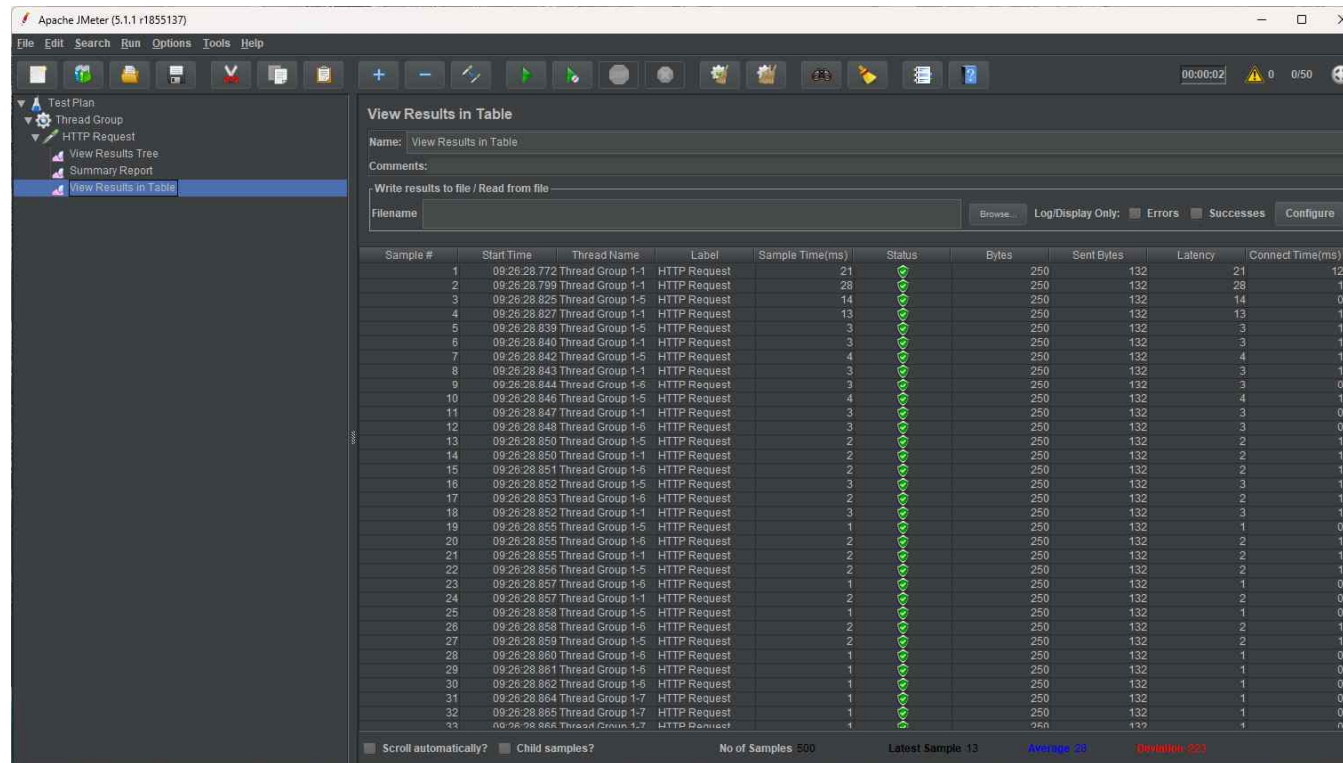
The screenshot displays the Apache JMeter 5.1.1 interface. The left sidebar shows a tree view with 'Test Plan', 'Thread Group', 'HTTP Request', 'View Results Tree', 'Summary Report', and 'View Results in Table'. The 'Summary Report' is selected and displayed in the main panel. The report includes a table with the following data:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	500	28	1	2085	223.31	0.00%	205.8/sec	50.26	26.53	250.0
TOTAL	500	28	1	2085	223.31	0.00%	205.8/sec	50.26	26.53	250.0

At the bottom of the interface, there are checkboxes for 'Include group name in label?' (unchecked), 'Save Table Data' (checked), and 'Save Table Header' (checked).

JMeter를 이용한 부하 테스트

❖ JMeter 예시



The screenshot displays the Apache JMeter 5.1.1 interface. The left sidebar shows the test plan structure: Test Plan, Thread Group, HTTP Request, View Results Tree, Summary Report, and View Results in Table (selected). The main panel, titled 'View Results in Table', shows a table of test results. The table has columns for Sample #, Start Time, Thread Name, Label, Sample Time(ms), Status, Bytes, Sent Bytes, Latency, and Connect Time(ms). The table contains 33 rows of data, all with a status of 'Success' (indicated by green checkmarks). The bottom status bar shows 'No of Samples: 500', 'Latest Sample: 13', 'Average: 28', and 'Deviation: 221'.

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	09:26:28.772	Thread Group 1-1	HTTP Request	21	Success	250	132	21	12
2	09:26:28.799	Thread Group 1-1	HTTP Request	28	Success	250	132	28	1
3	09:26:28.825	Thread Group 1-5	HTTP Request	14	Success	250	132	14	0
4	09:26:28.827	Thread Group 1-1	HTTP Request	13	Success	250	132	13	1
5	09:26:28.839	Thread Group 1-5	HTTP Request	3	Success	250	132	3	1
6	09:26:28.840	Thread Group 1-1	HTTP Request	3	Success	250	132	3	1
7	09:26:28.842	Thread Group 1-5	HTTP Request	4	Success	250	132	4	1
8	09:26:28.843	Thread Group 1-1	HTTP Request	3	Success	250	132	3	1
9	09:26:28.844	Thread Group 1-6	HTTP Request	3	Success	250	132	3	0
10	09:26:28.846	Thread Group 1-5	HTTP Request	4	Success	250	132	4	1
11	09:26:28.847	Thread Group 1-1	HTTP Request	3	Success	250	132	3	0
12	09:26:28.848	Thread Group 1-6	HTTP Request	3	Success	250	132	3	0
13	09:26:28.850	Thread Group 1-5	HTTP Request	2	Success	250	132	2	1
14	09:26:28.850	Thread Group 1-1	HTTP Request	2	Success	250	132	2	1
15	09:26:28.851	Thread Group 1-6	HTTP Request	2	Success	250	132	2	1
16	09:26:28.852	Thread Group 1-5	HTTP Request	3	Success	250	132	3	1
17	09:26:28.853	Thread Group 1-6	HTTP Request	2	Success	250	132	2	1
18	09:26:28.852	Thread Group 1-1	HTTP Request	3	Success	250	132	3	1
19	09:26:28.855	Thread Group 1-5	HTTP Request	1	Success	250	132	1	0
20	09:26:28.855	Thread Group 1-6	HTTP Request	2	Success	250	132	2	1
21	09:26:28.855	Thread Group 1-1	HTTP Request	2	Success	250	132	2	1
22	09:26:28.856	Thread Group 1-5	HTTP Request	2	Success	250	132	2	1
23	09:26:28.857	Thread Group 1-6	HTTP Request	1	Success	250	132	1	0
24	09:26:28.857	Thread Group 1-1	HTTP Request	2	Success	250	132	2	0
25	09:26:28.858	Thread Group 1-5	HTTP Request	1	Success	250	132	1	0
26	09:26:28.858	Thread Group 1-6	HTTP Request	2	Success	250	132	2	1
27	09:26:28.859	Thread Group 1-5	HTTP Request	2	Success	250	132	2	1
28	09:26:28.860	Thread Group 1-6	HTTP Request	1	Success	250	132	1	0
29	09:26:28.861	Thread Group 1-5	HTTP Request	1	Success	250	132	1	0
30	09:26:28.862	Thread Group 1-6	HTTP Request	1	Success	250	132	1	0
31	09:26:28.864	Thread Group 1-7	HTTP Request	1	Success	250	132	1	0
32	09:26:28.865	Thread Group 1-7	HTTP Request	1	Success	250	132	1	0
33	09:26:28.866	Thread Group 1-7	HTTP Request	1	Success	250	132	1	0

JMeter를 이용한 부하 테스트

❖ *JMeter* 실습

❖ 참조 주소

<https://github.com/aectarine>

참고 프로그램

- **Pigma**: 디자인 및 프로그램 포트폴리오 작성 도구
 - <https://www.figma.com/ko-kr/>
- **Mermaid**: 다이어그램 및 차트 작성 도구
 - <https://mermaid.js.org/>
- **DBDiagram**: DB ERD 작성 및 테이블 생성 도구 (무료. 커맨드를 통해 사용)
 - <https://dbdiagram.io/home>
- **eXERD**: DB ERD 작성 도구 (30일 무료. 쉬운 사용법)
 - <https://www.exerd.com/index.do>
- **PyCharm**: 파이썬 에디터 (유료. 다양한 편의 기능. 대학 이메일 계정 연동시 1년 무료. 연장 가능)
 - <https://www.jetbrains.com/ko-kr/pycharm>
- **JIRA**: 프로젝트 이슈 및 일정 관리도구
 - <https://www.atlassian.com/ko/software/jira>

참고 사이트

- **TIOBE**: 인기언어 지표 순위 사이트
 - <https://www.tiobe.com/tiobe-index/>
- **TechEmpower**: 웹 프레임워크 비교 순위 사이트
 - <https://www.techempower.com/benchmarks>
- **요즘IT**: 쉽고 재미있게 읽을 수 있는 IT 관련 매거진
 - <https://yozm.wishket.com/magazine/>
- **TheMiilk**: 실리콘밸리의 최신 AI 뉴스
 - <https://www.themiilk.com/>
- **인프런**: 무료, 유료 온라인 강의 사이트
 - <https://www.inflearn.com/>
- **렛플 or 홀라월드**: 사이드 프로젝트 관련 사이트
 - <https://letspl.me/> or <https://holaworld.io/>
- **Learn GIT Branching**
 - <https://learngitbranching.js.org/?locale=ko>



수고하셨습니다

