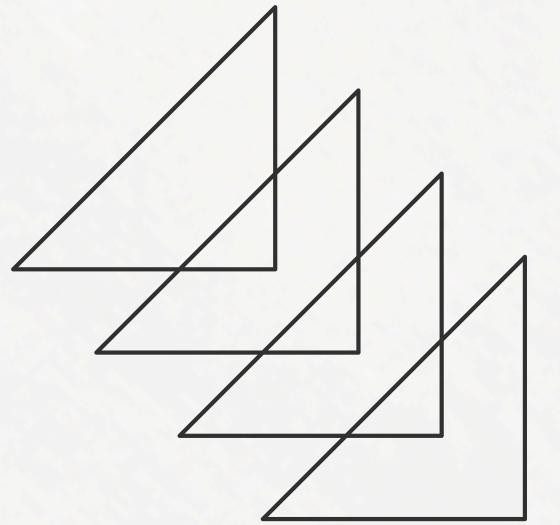


딥러닝 기반 실시간 수위 예측 및 데이터 시각화 웹서비스 구축

물알림단: 배기영, 이윤숙, 황준오

MULALIM DAN

2025-11-10



CONTENTS

SECTION 01

프로젝트 개요

SECTION 02

문제인식 & 목표

SECTION 03

시스템 개요

SECTION 04

역할별 구현

SECTION 05

기술적 문제 해결

SECTION 06

통합 결과 & 시연

SECTION 07

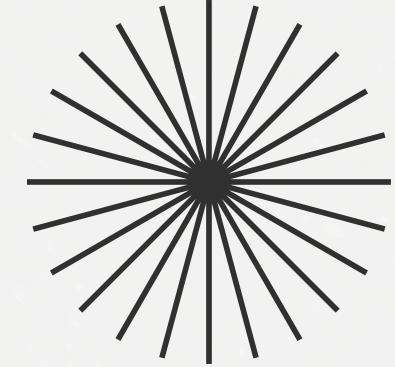
비즈니스 기대 효과

SECTION 08

프로젝트 고도화 및 향후 계획

프로젝트 개요

AI 기반 실시간 수위 예측 및 데이터 시각화 웹서비스 구축



프로젝트 개요

『2025 DATA·AI 분석 경진대회』
'지속가능한 수자원을 위한 지하수 함양
특성 분석 및 AI기반 지하수위 변동
예측' 과제

지하수위 변동을 예측하는
딥러닝 기반 모델을 개발

웹 대시보드로 시각화하여
실시간 의사결정 지원

성과 및 특징

NSE 0.97,
KGE 0.96
수준의 안정적 예측 성능

기상데이터를 활용해
전국 12개 관측소 수위 예측 가능

AI-WEB
END-TO-END 통합 구현
(데이터 수집→예측→시각화)

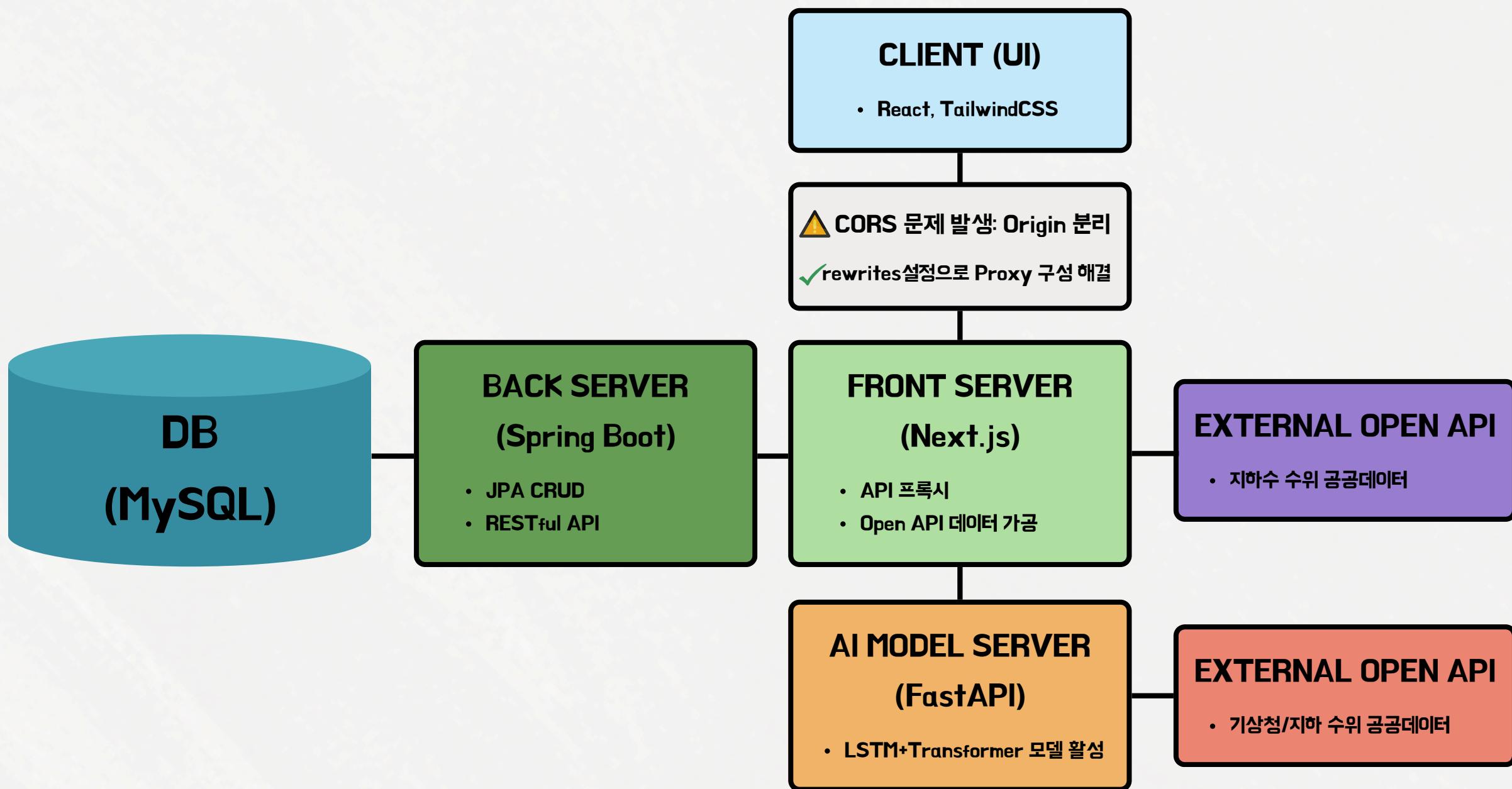
문제 인식 & 목표

기후변화로 인한 수자원 불확실성에 대응하다



- 이상기후(가뭄·집중호우)로 지하수 수위의 변동성이 커짐
- 우리나라 담수의 95% 이상이 지표수에 의존 → 가뭄에 취약
- 물리 기반 수문모델은 복잡한 요인을 완전히 반영하기 어려움
- 시계열 예측 정확도가 낮고, 실시간 정책 활용이 어려움
- 지역별 데이터 품질 불균형 및 관측소별 반응 차이 존재
- 기상데이터를 기반으로 지하수 수위를 예측 하는 AI 모델 구축
- LSTM + Transformer를 결합하여 단기·장기 패턴 동시 학습
- 전국 12개 관측소 대상 예측 → 웹 대시보드로 시각화
- 가뭄/홍수 대응, 수자원 관리 의사결정 지원

시스템 개요



구분	기술스택	주요 기능
프론트 엔드	Next.js(App Router), React, TypeScript, TailwindCSS, Highcharts	SPA 웹서비스/상태 관리(Jotai)/데이터 시각화/서버 라우트 기반 API 중계
백엔드	Spring Boot, Spring Security, MySQL, JPA	백엔드 RESTful API 기반 인증/데이터 관리
AI/데이터	FastAPI, Pandas, PyTorch	LSTM-Transformer 예측 모델 API화

역할별 구현 > Data Engineering

EDA

- 결측치 및 이상치 처리
- 정규화

Deep Learning

- 모델 학습

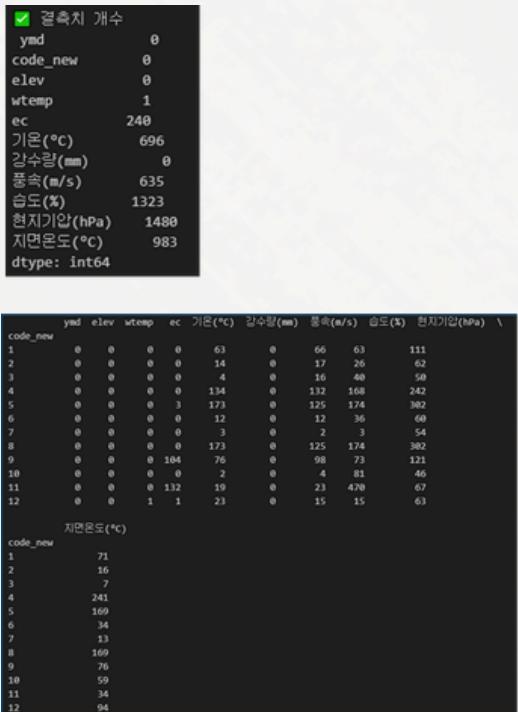
Feature Engineering

- 특성공학

성능평가

역할별 구현 > Data Engineering > EDA

결측치 파악 및 상관관계 분석



데이터 통합 및 결합

통합 데이터셋 구축

- 수위 테스트 데이터 + 국가지하수정보센터 OPENAPI 데이터 공간적 특성 반영
- 관측소 지점 정보 (수리전도도, 표고 등) 병합

시계열 범위

- 2014.01.01 ~ 2023.12.31

	ymd	code_new	elev	wtemp	ec	기온(°C)	강수량(mm)	풍속(m/s)	습도(%)	현지기압(hPa)	지면온도(°C)	K	EL_m
0	2014-01-01 00:00:00	1	105.47	16.3	592.0	2.2	0.0	0.4	76.0	998.4	-0.1	0.0132	108.29
1	2014-01-01 01:00:00	1	105.47	16.3	592.0	0.7	0.0	1.0	81.0	998.6	-0.1	0.0132	108.29
2	2014-01-01 02:00:00	1	105.47	16.3	592.0	2.6	0.0	0.6	77.0	998.8	-0.1	0.0132	108.29
3	2014-01-01 03:00:00	1	105.47	16.3	592.0	3.8	0.0	1.4	65.0	999.1	-0.1	0.0132	108.29
4	2014-01-01 04:00:00	1	105.47	16.3	592.0	3.5	0.0	0.9	65.0	999.1	-0.3	0.0132	108.29

결측치 처리

- 시간적 결측 보완 : 시계열 특성 고려, 시간적 보간 적용
- 파생 변수 결측: 0으로 처리
- 기타 결측 : 선형 보간법(INTERPOLATE) 적용

데이터 정규화

- 수위 데이터 : MINMAXSCALER -> 예측 값 복원 용이, 학습 안정성 확보
- 기상 변수 : STANDARDSCALER -> 단위 차이 제거, 정규분포 가정 만족

역할별 구현 > Data Engineering > 특성공학

시계열적 특성

- 주기성 반영 : 날짜 -> 월(MONTH), 일(DAY), 시간(HOUR) 분리
- 월,시간 변수 : SIN/COS 변환 -> 계절/시간 주기 학습
- 지연(LAG) 변수 : 강수량, 전기전도도 -> 1, 3, 7, 14, 30일
- 누적 / 이동평균 : 강수량 -> 3일/7일 누적합(ROLLING SUM) 기온, 지면온도 -> 3일 이동평균(ROLLING MEAN)

수문학적 특성

- 잠재증발산량 계산: PENMAN 공식 활용 -> 강수량과 결합 -> 순강수량 생성
- 변화량(DIFF) 반영 : 현지기압, 전기전도도 -> 시간 변화 동적 반영
- 온도 상호작용 : 지면온도- 수온 -> 열적 영향 반영

공간적 특성

- 지점 특성 활용 : 수리전도도 , 표고
- 상호작용 변수 : 수리전도도 * 3일 누적 강수량 -> 지하수 함양 반영, 강수량 * 표고 -> 지형 영향 반영

역할별 구현 > Data Engineering > 모델 개발

LSTM

- 시계열 내 장기 의존성을 학습하기 위한 RNN 구조
- 입력 게이트, 망각 게이트, 출력 게이트를 통해 과거 정보 선택적 기억

구조 요약

- 입력 : (시퀀스 길이 X 특성 개수)
- 1 LSTM (64, RETURN_SEQ = TRUE)
- 2 LSTM (64, RETURN_SEQ = FALSE)
- 각 층 DROPOUT = 0.2 적용
- DENSE(1) -> 다음 시점 수위 예측

하이퍼 파라미터

- 시퀀스 길이 : 30 (강수 후 지하수 반응 고려)
- 은닉 노드 : 64 (복잡도, 성능 균형)
- 배치 크기 : 32
- 학습률 : 0.001 -> REDUCELRONPLATEAU로 조정
- 조기 종료 : PATIENCE = 10

LSTM + TRANSFORMER 결합 모델

- LSTM의 순차적 학습 + TRANSFORMER의 전역 패턴 인식 결합

구조 요약

- LSTM 인코더(1층, RETURN_SEQ =TRUE)
- TRANSFORMER 인코더(2층, 4-HEAD ATTENTION,
- DROPOUT = 0.1)
- DENSE(1) -> 수위 회귀 예측

설계 근거

- 단기 반응(LSTM) + 누적/계절성(Transformer) 동시 반영

학습 설정

- 손실 함수: MSE
- 최적화 함수: ADAM (LR=0.001)
- 최대 EPOCH: 100
- LSTM 평균 45 EPOCH
- LSTM-TRANSFORMER 평균 52 EPOCH

역할별 구현 > Data Engineering > 성능 평가

지표	LSTM 평균 성능	LSTM-Transformer 평균 성능	변화(향상 정도)	해석
R ² (결정계수)	0.9539	0.9685	0.0146	모델의 설명력 향상 — 실제 수위 변동을 더 정확히 재현
RMSE (평균제곱근오차)	0.0466	0.0377	-0.0089	예측 오차 감소 — 수치적 정확도 개선
NSE (나쉬-서틀리프 효율계수)	0.9539	0.9685	0.0146	시계열 재현 능력 향상 — 모델 안정성 강화
KGE (클링-구프 효율계수)	0.9542	0.9616	0.0074	예측 일관성 개선 — 평균·분산 균형 유지

- 전 관측소 평균 성능 개선
- 특히 장기 패턴구간에서 향상 두드러짐
- TRANSFORMER의 전역의존성(SELF-ATTENTION)이 LSTM의 단기 기억 한계를 효과적으로 보완

역할별 구현 > FastAPI

FAST API 서버 개요

- PYTHON 기반 고성능 웹 프레임워크 FASTAPI 사용
- REST API 구조로 예측 결과를 JSON 형태로 반환
- 비동기 처리 및 간단한 라우팅으로 데이터 처리 흐름 관리

데이터 처리 흐름

- Open-Meteo → 기상 데이터 수집
- GIMS API → 지하수 수위 및 lag 데이터 수집
- 수집된 데이터 전처리 → 모델 입력 형태로 변환
- 관측소별 학습된 모델 불러와 예측 수행

주요 기능 및 엔드포인트

- /api/v1/model/forecast?station=[관측소코드] : 특정 관측소에 대한 7일간의 시간별 지하수 수위 예측 결과를 제공합니다.
- /api/v1/model/forecast/all : 등록된 모든 관측소에 대한 7일간의 시간별 수위 예측 결과를 일괄적으로 제공합니다.
- /api/v1/model/analysis/summary : 모든 관측소의 예측 결과를 분석하여, 강수시 가장 큰 수위 상승이 예상되는 상위 5개 관측소, 가뭄 시 가장 큰 수위 하강이 예상 되는 관측소, 그리고 수위 변동폭이 가장 큰 상위 5개 관측소 정보를 요약하여 제공합니다.

핵심 역할

역할별 구현 > Backend

Spring Boot Framework

 **MVC:** Controller → Service → Repository → Entity
→ Spring Boot – Next.js 간 분리형 MVC 구조
→ 프론트엔드(Next.js)가 View 역할을 수행,
Spring Boot는 데이터(Model + Controller) 제공

 **데이터 패턴:** Spring Data JPA CRUD 중심
→ 예) Predicted3yRepository, :
관측 데이터·예측 데이터를 Entity ↔ DTO ↔ JSON 형태로 송수신
→ DB 테이블: users, predicted_3y 등

 **공통 처리: SecurityConfig**
세션 기반 인증(Spring Security) +
CorsConfigurationSource 로
Next.js 도메인(<http://10.125.121.222:3000>) 접근 허용
→ JSESSIONID 쿠키 기반 세션 유지
→ 인증/인가 절차 자동화
(UsernamePasswordAuthenticationFilter 기반)

① RESTful API 구현

- Controller 계층에서
@GetMapping,
@PostMapping 중심
REST API 구성
- /api/v1/me/results/**
등 → Next.js의 fetch()
요청과 연결되어 JSON
응답 반환

② JPA 중심 DB

CRUD

- Spring Data JPA
- findAll(), save(),
delete() 등 CRUD 자동화
- 일부 데이터는 네이티브
쿼리로 통계 계산 → 비즈
니스 로직 최소화 + 데이
터 접근 표준화

③ 인증

(Spring Security)

- SecurityConfig로 세션
기반 로그인
- 로그인 성공 시
JSESSIONID 쿠키 발급
- /login, /logout,
/auth/me 엔드포인트로
인증 상태 관리

역할별 구현 > Backend DB 설계



users			
	회원		
users	회원(영문)	타입	제약조건/설명
user_id	고유ID		PK, AUTO_INCREMENT
username	사용자명	VARCHAR(100)	
email	이메일	VARCHAR(200)	
password_hash	비밀번호 해시	VARCHAR(200)	
role	역할	VARCHAR(50)	user.admin
is_active	활성 상태	TINYINT(1)	1:활성, 0:비활성
created_dt	생성일시	DATETIME	
updated_dt	수정일시	DATETIME	



observed_data			
	원천 데이터	타입	제약조건/설명
obs_id	컬럼(영문)		
obs_id	실측 ID	BIGINT UNSIGNED	PK, AUTO_INCREMENT
modelrun_id	실행 ID	BIGINT UNSIGNED	NULL, (선택) 업로드/배치 원본 주적용, FK→filelog_modelrun(modelrun_id)
station	관측소코드	TINYINT UNSIGNED	NOT NULL, CSV: code_new (예: 1~12)
ts	예측 시각	DATETIME	NOT NULL, CSV: ymd, 형식 YYYY-MM-DD HH:00 (KST)
elev	수위	DECIMAL(10,5)	NULL 허용, CSV: elev (train에는 존재 / test_inputs에는 없음)
wtemp	수온	DECIMAL(10,5)	NULL, CSV: wtemp
ec	전기 친도도	DECIMAL(10,5)	NULL, CSV: ec
temp_c	기온(°C)	DECIMAL(10,5)	NULL, CSV: 기온(°C)
rain_mm	강수량(mm)	DECIMAL(10,5)	NULL, CSV: 강수량(mm)
wind_speed_ms	풍속(m/s)	DECIMAL(10,5)	NULL, CSV: wind_speed_ms
humidity_pct	습도(%)	DECIMAL(10,5)	NULL, CSV: 습도(%)
pressure_hpa	현지기압(hPa)	DECIMAL(10,5)	NULL, CSV: 현지기압(hPa)
ground_temp_c	지면온도(°C)	DECIMAL(10,5)	NULL, CSV: 지면온도(°C)
created_dt	생성 일시	DATETIME(6)	DEFAULT CURRENT_TIMESTAMP(6)



predicted_3y			
	3년 장기 예측		
pred3y_id	예측 ID	BIGINT UNSIGNED	PK, AUTO_INCREMENT
modelrun_id	실행 ID	BIGINT UNSIGNED	NOT NULL, FK → filelog_modelrun(modelrun_id)
station	관측소코드	TINYINT UNSIGNED	NOT NULL, CHECK(1~12)
ts	예측 시각	DATETIME	NOT NULL (미래 시각, KST 기준)
y_predicted	예측값	DECIMAL(10,5)	NOT NULL
yhat_lower	예측 하한	DECIMAL(10,5)	NULL
yhat_upper	예측 상한	DECIMAL(10,5)	NULL
created_dt	생성 시각	DATETIME(6)	NOT NULL, DEFAULT CURRENT_TIMESTAMP(6)



model_metrics			
	모델 성능지표		
metric_id	컬럼(영문)		
metric_id	고유ID	BIGINT UNSIGNED	PK, AUTO_INCREMENT
modelrun_id	모델 실행 고유ID	BIGINT UNSIGNED	NOT NULL, FK → filelog_modelrun(modelrun_id)
station	지점코드	TINYINT UNSIGNED	NOT NULL, CHECK (station BETWEEN 1 AND 12)
ts	예측 시각	DATETIME	NOT NULL 예측 실측이 적용되는 시작: 202509100000 202509100100
nse	NSE(나և-서플리프 훌륭계수)	DOUBLE	NULL 허용
kge	KGE(블링-글라 훌륭계수)	DOUBLE	NULL 허용
rmse	RMSE(평균제곱근오차)	DOUBLE	NULL 허용
r2	R ² (결정계수)	DOUBLE	NULL 허용
metrics_json	지표 속성 JSON	JSON	JSON NULL 지표 전체를 JSON으로 보관
created_dt	생성 일자	DATETIME	NOT NULL, DEFAULT CURRENT_TIMESTAMP(6)



predicted_data			
	예측된 데이터		
predicted_id	컬럼(영문)		
predicted_id	고유ID	BIGINT UNSIGNED	PK, AUTO_INCREMENT
model_runid	모델 실행 고유ID	BIGINT UNSIGNED	NOT NULL, FK → filelog_modelrun(modelrun_id)
station	지점코드	TINYINT	NOT NULL, CHECK (station BETWEEN 1 AND 12)
valid_time	유효 시각	DATETIME	NOT NULL 예측 실측이 적용되는 시작: 202509100000 202509100100
actual	실제 수위	DECIMAL(10,5)	NOT NULL
predicted	예측 수위	DECIMAL(10,5)	NOT NULL
residual	잔차	DECIMAL(10,5)	NOT NULL (= actual - predicted) 저장 가능)
created_dt	생성 일자	DATETIME	NOT NULL, DEFAULT CURRENT_TIMESTAMP(6)



filelog_modelrun			
	파일로그 모델 실행		
filelog_id	컬럼(영문)		
user_id	사용자 고유ID	BIGINT UNSIGNED	FK → users(user_id)
original_filename	원본 파일명	VARCHAR(255)	NOT NULL (업로드 시 파일명)
save_filename	저장 파일명	VARCHAR(512)	NOT NULL (서버 경로)
file_size_bytes	파일 크기	BIGINT UNSIGNED	NULL (파일 크기 byte 단위)
model_name	모델명	VARCHAR(100)	NULL null 허용 (ex:LSTM)
model_version	모델 버전	VARCHAR(50)	NULL (ex v1.0)
hyperparams_json	하이퍼파라미터 JSON	JSON	NULL (하이퍼파라미터 스냅샷)
status	실행 상태	VARCHAR(16)	NOT NULL DEFAULT 'PENDING' ENUM('PENDING', 'RUNNING', 'DONE', 'FAILED')
fail_reason	실패 사유	VARCHAR(500)	NULL
result_path	결과 CSV 경로	VARCHAR(512)	NULL
model_filename	모델 결과 파일명	VARCHAR(512)	NULL
started_dt	시작 시각	DATETIME	NULL
finished_dt	종료 시각	DATETIME	NULL
created_dt	생성 일자	DATETIME	NOT NULL, DEFAULT CURRENT_TIMESTAMP(6)

역할별 구현

> Frontend 설계

컴포넌트 기반 아키텍처 (Component-Based)

- ▣ **캡슐화:** 컴포넌트는 자신의 상태·로직·스타일을 내부에 가짐
→ 예) GeoMap은 렌더/마커 로직 내부에, props로 데이터만 입력
- ▣ **재사용성:** 동일 UI를 다양한 맥락에서 props만 바꿔 재사용
→ 예) CustomButton(bType, bStyle, handler)
- ▣ **합성:** 작은 블록을 조합해 페이지 구성
→ 예) DashBoardContents = GeoMap + Line/Table 등

디자인 패턴 (Atomic Design 매핑)

- ▣ **Atoms:** 버튼/아이콘/입력, 그리고 Jotai 상태 원자(atoms.tsx)
- ▣ **Molecules:** 차트 조각(BarChart, LineChartShadeZoom)
- ▣ **Organisms:** GeoMap, 성능패널, Header/Footer/SubNav
- ▣ **Templates:** app/layout.tsx, 공통 template.tsx
- ▣ **Pages:** app/(페이지)/page.tsx (App Router = 폴더=라우트)

```
src/
  └── atoms/
    └── atoms.tsx      # State Atoms (상태 원자) - UI가 아닌 Jotai를 이용한 전역 상태의 최소 단위

  └── components/
    └── atoms/          # Molecules / Atoms (분자/원자 단위 컴포넌트)
      ├── CustomButton.tsx # 재사용 가능한 버튼
      ├── CustomModal.tsx # 모달 컴포넌트
      └── ...etc          # 기타 기본 UI 빌딩 블록

    └── ui/             # Organisms (유기체) - 페이지의 주요 골격을 형성
      ├── Header.tsx     # 로고 + Nav + UserButton
      ├── Footer.tsx     # 푸터 정보
      └── SubNav.tsx     # 서브 네비게이션 메뉴

    └── dashboard/       # Organisms (유기체) - 핵심 기능의 조합
      ├── GeoMap.tsx      # 지도 + 마커 + 정보창
      ├── BarChart.tsx     # 차트 분자(Molecule)
      ├── LineChartShadeZoom.tsx # 차트 분자(Molecule)
      └── DashBoardContents.tsx # 대시보드 내 위젯들의 조합 (Template 역할도 수행)

    └── userpage/        # Organisms (유기체) - 사용자 페이지 기능 조합
      ├── UserInfoBox.tsx
      └── UserEditBox.tsx

  └── app/
    ├── layout.tsx      # Templates (템플릿) - 최상위 페이지 레이아웃
    └── template.tsx     # Templates (템플릿) - 페이지 공통 래퍼

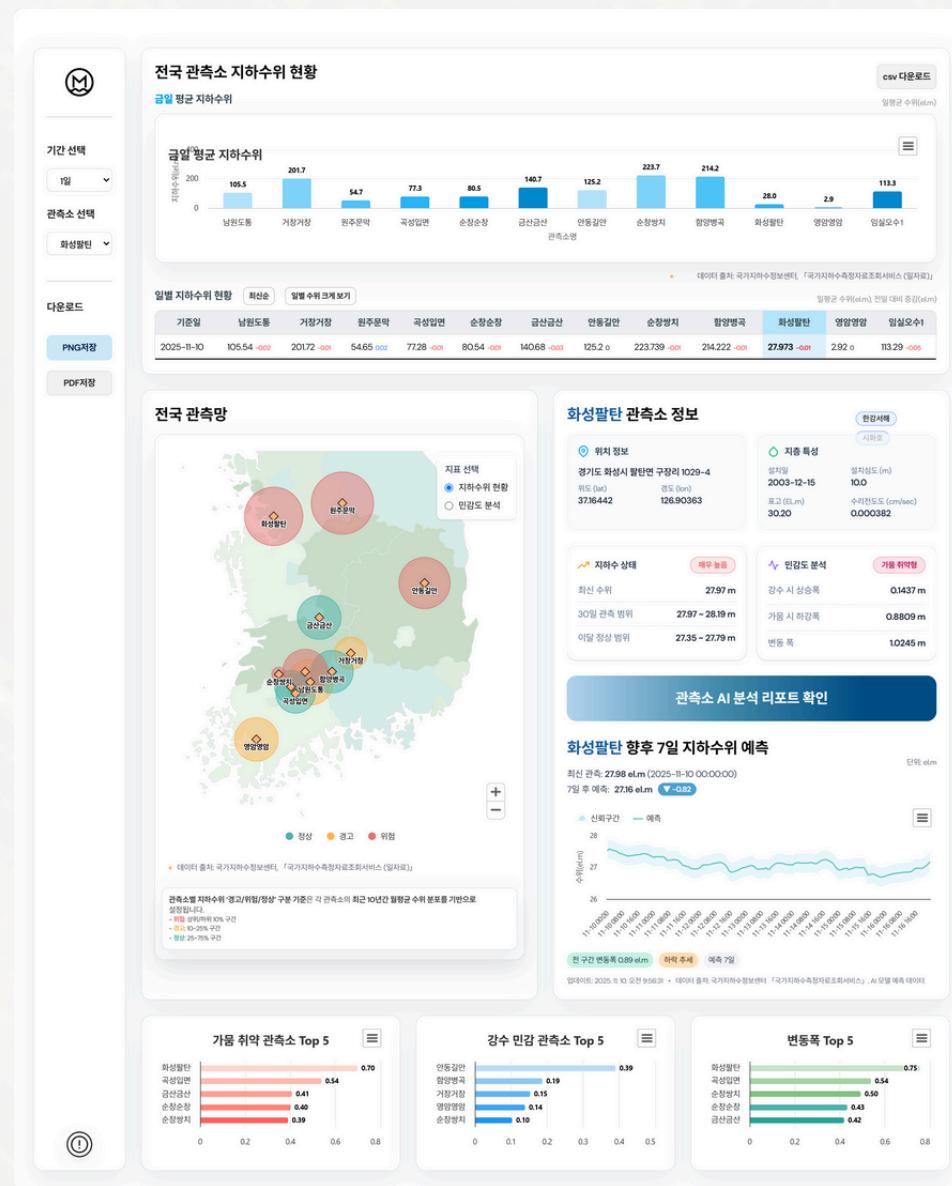
    └── explain/
      └── page.tsx       # Pages (페이지) - 대시보드 페이지

    └── login/
      └── page.tsx       # Pages (페이지) - 로그인 페이지

    └── userpage/
      └── page.tsx       # Pages (페이지) - 사용자 정보 페이지
```

역할별 구현

> Frontend



핵심 역할

① 컴포넌트 설계

- 대시보드 설계 및 컴포넌트화 (GeoMap/LineChart/Zoom/성능패널/테이블 등)

② 데이터 연동 및 가공

- Open API/백엔드/AI(FASTAPI) 데이터 연동, 로딩·에러 안정화
- Next App Router에서 데이터 가공

③ 상태 관리 및 프록시

- 상태 관리 최적화(Jotai) 및 프록시(rewrites)로 CORS 해결

대시보드 화면 구성

① 지하수위 현황

- 전국 12개 관측소의 지하수위 현황과 기간별 지하수위 변동 및 전일대비 증감량 표시 테이블

② 지도

- 전국 12개 관측소 위치 + 버블 크기/색상 분류 (기간 평균/상태 등급)
- 범례/툴팁/클릭 연동 선택 관측소 정보 출력

③ 관측소 정보

- 관측소 물리특성
- 민감도 통계

④ 예측 시각화

- 7일 예측 라인 + 신뢰구간

⑤ 강수민감/가뭄취약 통계

- 가뭄/홍수 대응 수자원 관리 우선 지역 Top5

⑥ 관측소별 AI 분석 리포트

- 지하수위 장기 추세 및 예측 수위
- 기상요인-지하수위 예측 상관 시각화
- 관측소에 대한 AI 예측 성능 요약
- 예측 영향 요인 분석

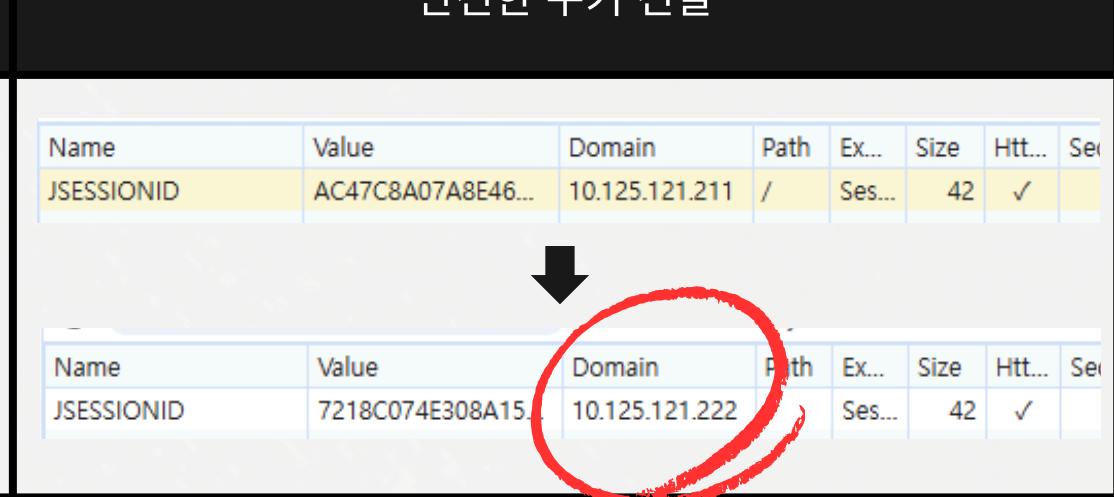
기술적 문제 해결

문제 인식	문제 사항	해결 방법	결과
다중서버 CORS (백엔드)	프론트(Next.js)와 백엔드(Spring Boot) Origin 불일치	CorsconfigurationSource에서 프론트 Origin 명시	데이터 요청 정상화
다중서버 CORS (프론트엔드)	프론트(Next.js)와 백엔드(Spring Boot) Origin 불일치	Next.js rewrites 프록시 구성	세션 유지를 위한 안전한 쿠키 전달
OPEN API 중단 오류	일부 관측소 502 실패	Promise.allSettled() 적용	부분 성공 로직으로 안정성 개선
FastAPI 지하수위 예측의 응답 시간 지연	FastAPI에서 수행하는 데이터수집-전처리(특성공학)-추론 과정의 소요시간이 큼	OPEN API 데이터 수집을 병렬로 처리	<ul style="list-style-type: none">개선: 16초 → 8초로 단축한계와 보완 필요: 데이터 수집을 Spring Boot에서 웹 크론으로 DB에 저장하는 방향의 확장이 필요

기술적 문제 해결 > 다중 서버 CORS(백엔드 서버)

문제 인식	문제 해결 상세	결과
<p>프론트(Next.js) 에서 원천 데이터 API 요청</p> <p>↓</p> <p>브라우저 CORS (Cross Site Resource Sharing) 정책에 의해 막힘</p>	<ul style="list-style-type: none">CorsConfigurationSource 적용<ul style="list-style-type: none">▶ [해결] RESTful API를 통한 데이터 요청 정상화 / 세션 쿠키 포함 응답 가능▶ [한계] 다만, JSESSIONID는 백엔드 Origin 기준으로 발급되므로 다중 서버 구조에서는 프록시 또는 HTTPS 설정이 필요▶ [보완] 프론트엔드 서버에서 프록시 설정 추가 <pre>@Bean public SecurityFilterChain filterChain(HttpSecurity http) throws Exception { // 1) CORS: Next.js 개발 서버(예: http://localhost:3000) 허용 http.cors(cors -> cors.configurationSource(req -> { CorsConfiguration cfg = new CorsConfiguration(); cfg.setAllowedOrigins(List.of("http://10.125.121.222:3000", // ● 실제 프론트 주소 "http://localhost:3000" // (필요시 개발 로컬도 허용)))); }); }</pre>	<p>REST API 데이터 요청 정상화</p>

기술적 문제 해결 > 다중 서버 CORS(프론트엔드 서버)

문제 인식	문제 해결 상세	결과																								
<p>프론트(Next.js) 백엔드(Spring Boot) AI 모델(FastAPI) 구조의 서로 다른 Origin ↓ 브라우저 CORS (Cross Site Resource Sharing) 정책에 의해 막힘</p>	<ul style="list-style-type: none">해결 방법1: HTTPS + SameSite=None; Secure; → Chrome 3rd-party 쿠키 축소(불완전)해결 방법2: 배포 단계에서 Nginx 리버스 프록시 구성 → 동일 Origin으로 인식되도록 설정해결 방법3: Next.js rewrites 설정 (개발 단계용 Proxy) <p>→ [채택] 개발 단계의 해결책으로 Next.js의 rewrites 설정으로 Next 서버를 중계 서버로 사용</p> <div style="display: flex; justify-content: space-around;"><div style="text-align: center;"><p>Next.js의 rewrites 설정</p><pre>const nextConfig: NextConfig = { async rewrites() { return [{ source: '/java/:path*', destination: 'http://10.125.121.222:8080/:path*' }, { source: '/ml/:path*', destination: 'http://10.125.121.216:8000/:path*' },] } };</pre></div><div style="text-align: center;"><p>안전한 쿠키 전달</p><table border="1"><thead><tr><th>Name</th><th>Value</th><th>Domain</th><th>Path</th><th>Ex...</th><th>Size</th><th>Htt...</th><th>Set...</th></tr></thead><tbody><tr><td>JSESSIONID</td><td>AC47C8A07A8E46...</td><td>10.125.121.211</td><td>/</td><td>Ses...</td><td>42</td><td>✓</td><td></td></tr><tr><td>JSESSIONID</td><td>7218C074E308A15...</td><td>10.125.121.222</td><td>/</td><td>Ses...</td><td>42</td><td>✓</td><td></td></tr></tbody></table></div></div>	Name	Value	Domain	Path	Ex...	Size	Htt...	Set...	JSESSIONID	AC47C8A07A8E46...	10.125.121.211	/	Ses...	42	✓		JSESSIONID	7218C074E308A15...	10.125.121.222	/	Ses...	42	✓		세션 유지를 위한 안전한 쿠키 전달 및 서버 간 통신 구조 확보
Name	Value	Domain	Path	Ex...	Size	Htt...	Set...																			
JSESSIONID	AC47C8A07A8E46...	10.125.121.211	/	Ses...	42	✓																				
JSESSIONID	7218C074E308A15...	10.125.121.222	/	Ses...	42	✓																				

기술적 문제 해결 > OPEN API 중단 오류

문제 인식	문제 해결 상세	결과
<p>Next 서버에서 Open API를 통해 데이터 수집</p> <p>↓</p> <p>일부 관측소 (Open API)에서 간헐적 502 Bad Gateway 오류 발생</p> <p>↓</p> <p>전체 데이터 요청이 중단됨</p>	<ul style="list-style-type: none">Promise.allSettled()를 적용하여 일부 요청 실패 시에도 다른 관측소 데이터는 정상 수집성공 실패 결과를 분리하여 부분 성공 처리 구조 구현 <pre>try { // settledResults => [{ status: 'fulfilled', value: ['5724', [Array]] }, ...] const settledResults = await Promise.allSettled(GENNUMS.map((gen: string) => fetchFromEachStation(gen, begindate, enddate) .then((units: UnitFromOpenApiT[]): [string, UnitFromOpenApiT[]] => [gen, units]))); // 성공한 것만 추려서 객체로 변환 const entriesWithFallbacks = settledResults.map((result, idx) => result.status === "fulfilled" ? result.value : [GENNUMS[idx], []]); You, 14 hours ago • 대시보드 수정 ... // 관측소별 데이터로 변환 const dataByStation: Record<string, UnitFromOpenApiT[]> = Object.fromEntries(entriesWithFallbacks); // 실패한 항목 로깅하기</pre>	<p>API 요청 안정성 개선/ 실패 항목만 예외 처리</p>

기술적 문제 해결 > FastAPI 응답 지연

문제 인식	문제 해결 상세	결과
FastAPI 지하수위 예측의 응답 시간 지연 ↓ FastAPI에서 수행하는 데이터수집-전처리(특성공학)-추론 과정의 소요시간이 큼	<ul style="list-style-type: none">OPEN API 데이터 수집을 병렬로 처리병렬화 핵심: ThreadPoolExecutor(...), executor.submit(...), as_completed(...) 조합 <pre># 최대 5개 스레드로 동시에 요청 dfs = [] with ThreadPoolExecutor(max_workers=5) as executor: futures = [executor.submit(fetch_station_data, st) for st in stations] for future in tqdm(as_completed(futures), total=len(futures)): dfs.append(future.result())</pre>	개선: 16초 → 8초로 단축 한계와 보완: 데이터 수집을 Spring Boot에서 웹 크론으로 DB에 저장하는 방향의 확장이 필요

통합 결과 발표 및 시연

풀 동영상: <https://www.awesomescreenshot.com/video/46379582?key=841a26872d250d5c3c5fcddca08a67d5>

신국 시야구위 연봉과 AI 기반 시야 구위 예측 보Hell 군의 글과 시각화

이 대시보드는 2014~2023년 가상지표를 기반으로 AI 모델이 예측한 지하수위 변동을 시각화합니다. 관측소별 단위 분석을 통해 주요 기상 요인과 수위 변화를 함께 탐색할 수 있습니다.

전국 관측소 지하수위 현황

금일 평균 지하수위

기간 선택: 1일

관측소 선택: 안동길안

다운로드: PNG저장, PDF저장

데이터 출처: 국가지하수정보센터, 「국가지하수측정자료조사회서비스(일자료)」

일별 지하수위 현황 (최신순)

기준일	남원도통	거창거창	원주문역	곡성입면	순창순창	금산금산	안동길안	순창방치	함양병곡	화성팔탄	영암영암	임실오수1
2025-11-10	105.54 -0.02	201.72 -0.01	54.65 0.02	77.28 -0.01	80.54 -0.01	140.68 -0.03	125.2 0	223.739 -0.01	214.222 -0.01	27.973 -0.01	2.92 0	113.29 -0.05

전국 관측망

지표 선택: 지하수위 현황

0:02 < >

안동길안 관측소 정보

위치 정보: 경상북도 안동시 길안면 천지리 731-1
위도 (lat): 36.45867
경도 (lon): 128.89419

지층 특성: 설치일: 2001-12-20
설치심도 (m): 10.0
표고 (EL.m): 128.91
수리전도도 (cm/sec): 0.0001332

비즈니스 기대 효과



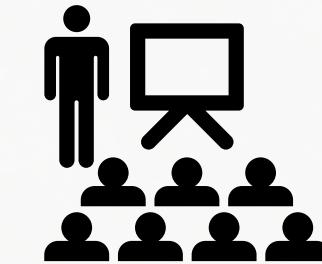
모니터링

실시간 예측 기반 이상징후 조기 감지



의사결정 지원 및 정책 활용

가뭄·홍수 시 취수·방류 계획 등 행정 의사
결정 근거 제공



서비스·도메인 확장

지하수 → 하천·댐·농업용수 등
타 수문 분야 확대

AI 예측 + 공공데이터 융합 기반의 지속 가능한 수자원 관리 플랫폼으로 확장

프로젝트 고도화 및 향후 계획

현재



데이터 허브 Next.js

- 로컬 3서버 통합 완료
(Spring Boot - Next.js - FastAPI)
- 개발용 프록시 기반, 기능 중심 구조

운영 단계 전환 방향

Spring Boot 중심 아키텍쳐

- Spring Boot → API Gateway
 - 모든 외부/내부 요청: Nginx → Spring Boot → FastAPI/DB/Next.js
 - 인증·인가, CORS, 로깅·레이트리밋 중앙 집중 관리
- Next.js → 프레젠테이션/UI 전담
- FastAPI → 모델 예측 전담

End-to-End 데이터 파이프라인 확장

Open API - Spring Boot - MySQL
Spring Boot - FastAPI - MySQL

- Spring Boot는 실시간 데이터 수집
웹 크론으로 DB에 저장
- FastAPI는 추론 후 DB에 저장
- Next.js는 저장된 데이터를 응답받음

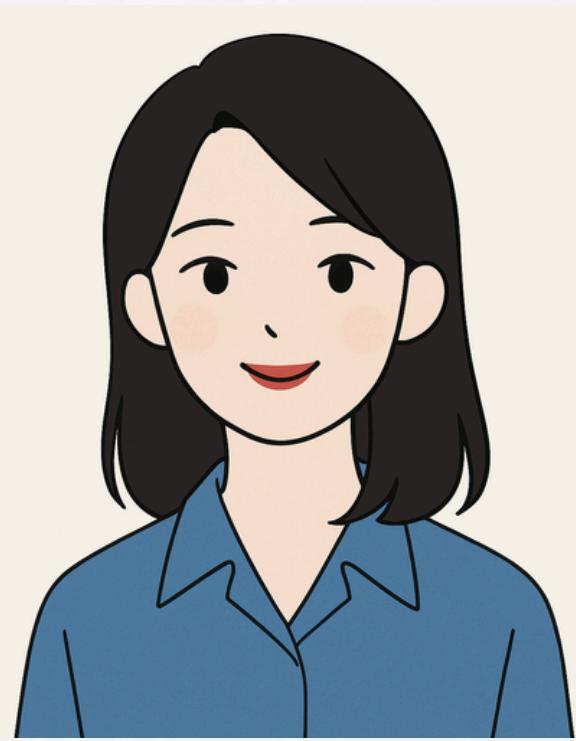
개선 효과



보안·성능·확장성 강화

- Nginx + HTTPS(HTTP/2) : 종단간 암호화 및 라우팅 표준화
- SameSite=None; Secure 쿠키 기반 세션 유지
- CORS 단일화: Gateway에서만 허용, 프론트는 동일 Origin처럼 사용
- 감사 로그·메트릭 수집: 요청 지연·오류율 실시간 모니터링

Team



배기영

Frontend



이윤숙

Backend



황준오

Data Engineering
/ Fast API

The End

THANK YOU FOR LISTENING

MULALIM DAN

2025-11-10