

Jang MoonSu / DEVELOPER PORTFOLIO

Java 백엔드에서 IoT·하드웨어까지

백엔드 서버 개발 경험을 바탕으로
IoT·하드웨어 영역을 넓혀가는 개발자, 장문수

프로젝트 보기 ↓

GitHub ↗

Java Backend · IoT Integration · Learning Tools

SELECTED PROJECTS / 2024 — 2026

소개



장문수
Jang MoonSu

Java Backend Developer

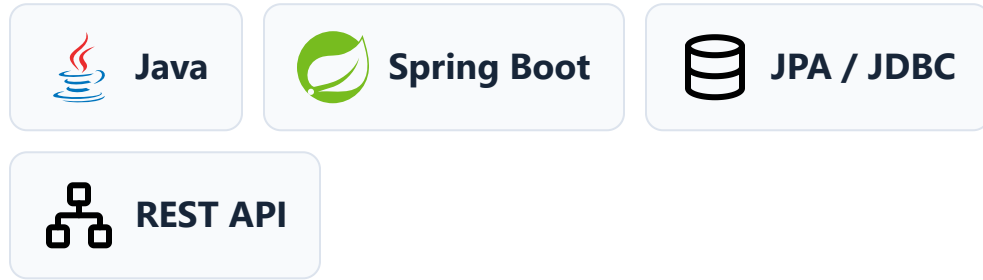
- Java 기반 백엔드 서버 개발
- Android·서버·센서 데이터 연동
- IoT·하드웨어 영역으로 학습과 구현 확장

경력 및 교육

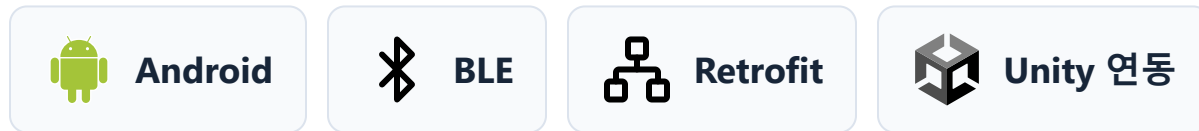
2026.07 – 현재	SSAFY 16기 · 임베디드 트랙 삼성청년SW·AI아카데미
2025.06 – 2026.03	주식회사 유더블유에스 Java 백엔드 · Quantum Computing TFT
2024.09 – 2025.03	SK Shieldus Rookies 모의해킹 · 클라우드 보안 과정 수료
2019.03 – 2025.02	한신대학교 컴퓨터공학부

보유 기술 스택

백엔드



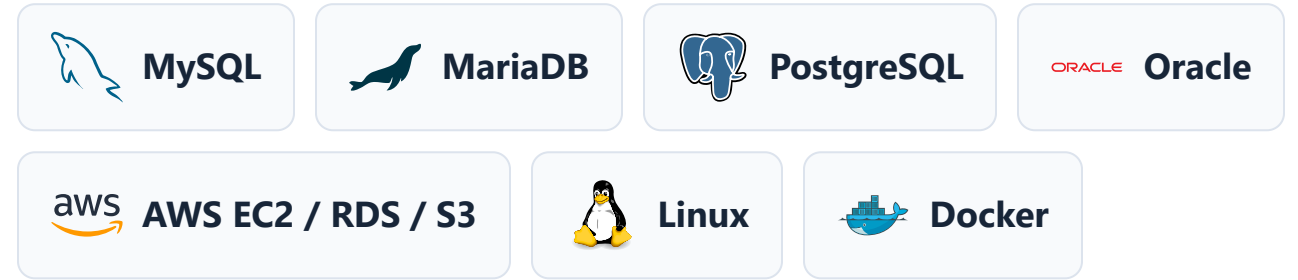
모바일 · 연동



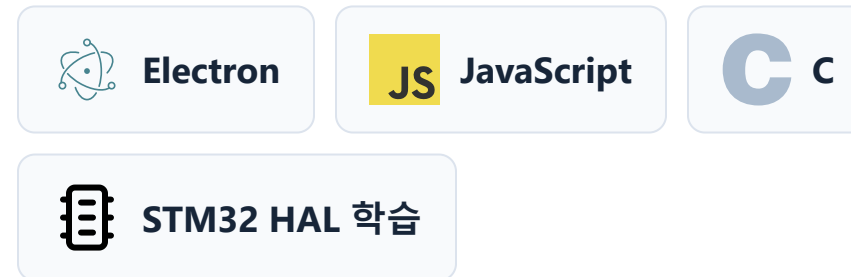
AI



데이터 · 배포



학습 도구 · 임베디드 학습



수상

SK Shieldus Rookies

슈퍼루키상 금상

2025.03.13

한신대학교 SW교육센터

2024 혁신 AI·SW페스티벌 캡스톤디자인 동상

2024.11.19

한국디지털콘텐츠학회

2024 추계종합학술대회 대학생 논문경진대회 은상

2024.11.08

한국디지털콘텐츠학회

2024 하계종합학술대회 대학생 논문경진대회 은상

2024.06.29

한신대학교 컴퓨터공학부

제29회 소프트웨어공모전 은상

2023.11.14

한신대학교 컴퓨터공학부

제28회 소프트웨어공모전 동상

2022.11.15

자격 · 어학

정보처리기사

발급기관 한국산업인력공단
취득일 2024.10.02

OPIc

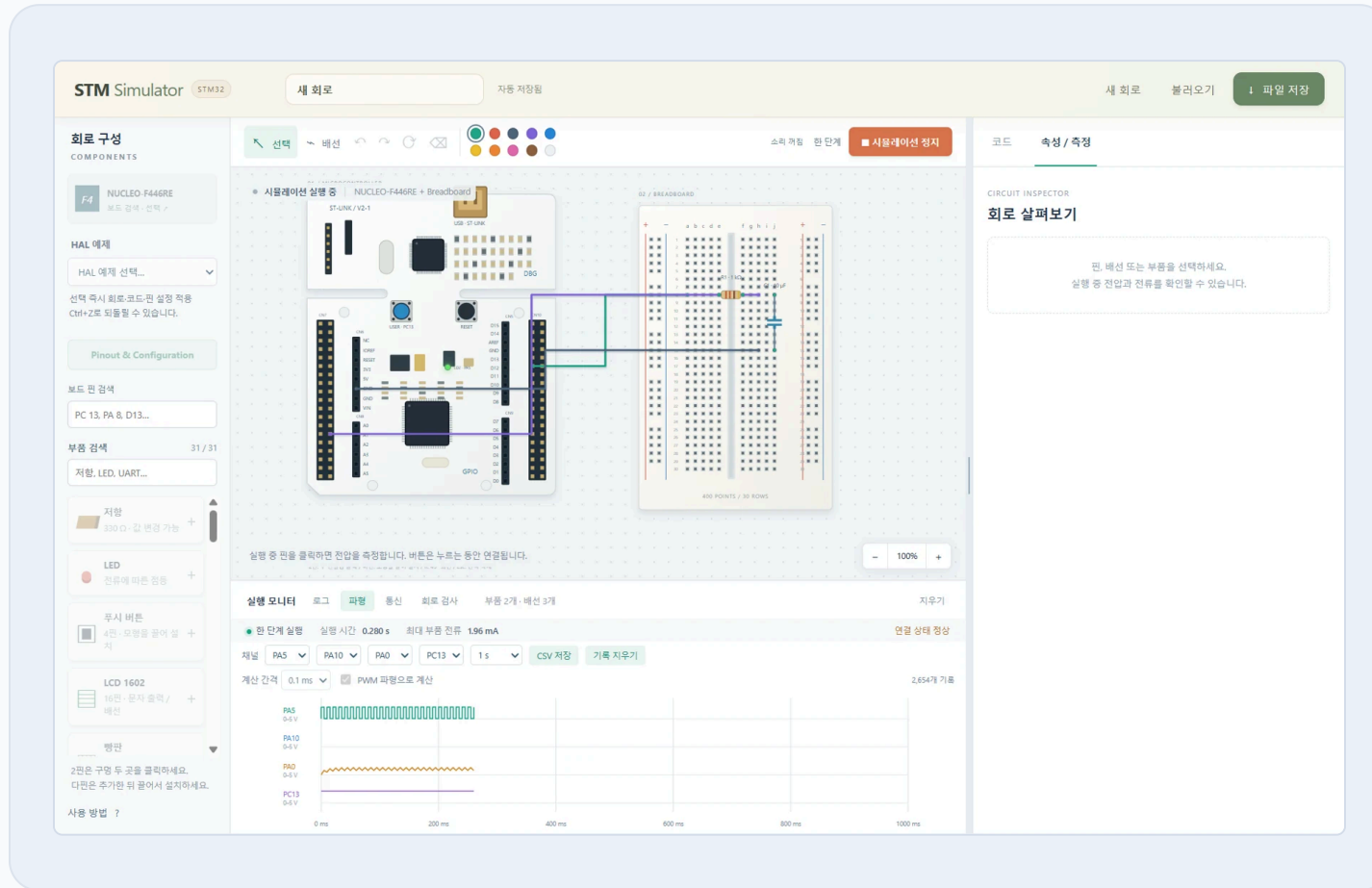
IL
Intermediate Low

취득일 2026.09.14

프로젝트

01 STM-Simulator STM32 학습용 데스크톱 도구	02 CONS BLE 실내 측위·AR 길안내	03 BOOKIES 전자책 Web·App 서비스
04 Arabica DB 백업 자동화 시스템	05 CMP 외부 API 연동·성능 시험	06 Wallet Dcommerce 백엔드 API
07 Store 커머스 DB·Wallet 인증 연동		

01 STM-Simulator



Electron 기반

회로 배선·주변장치 설정 시뮬레이터

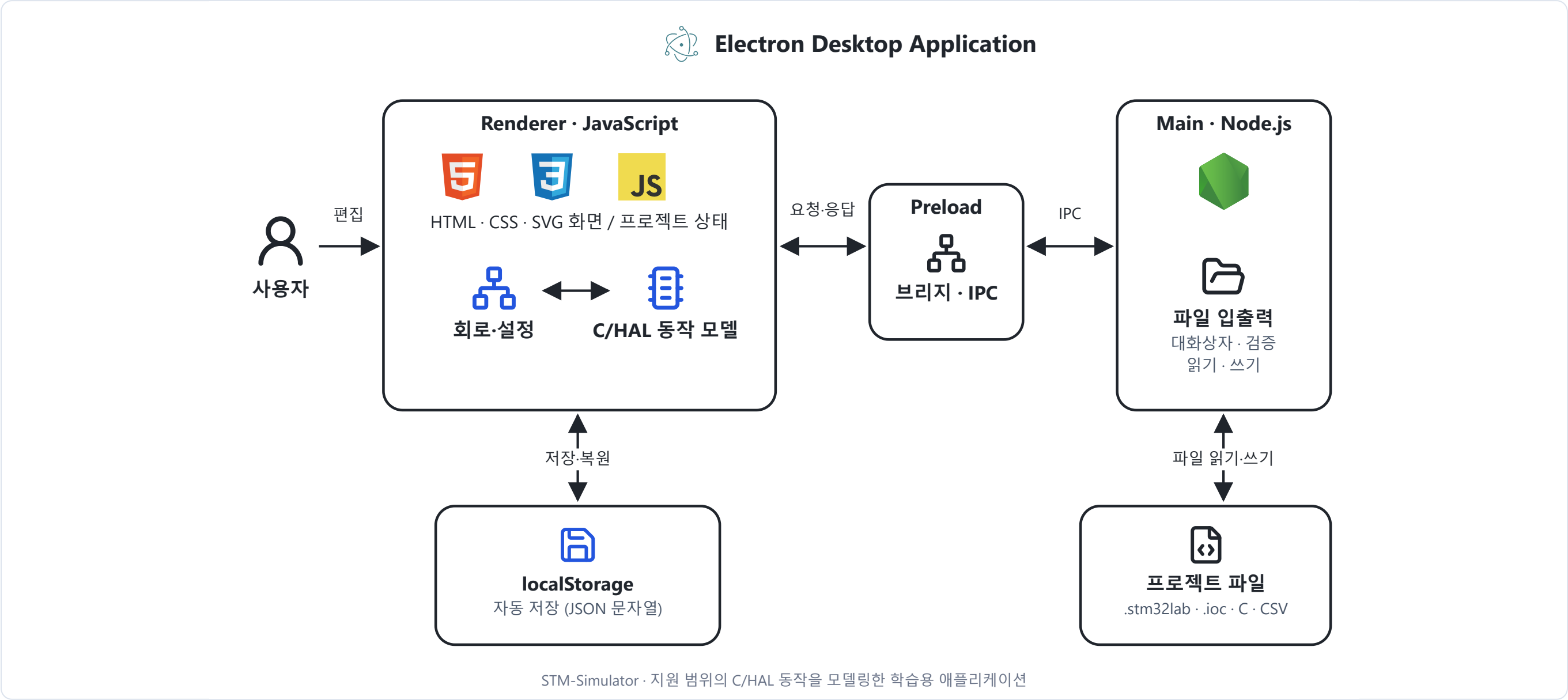
2026.09 — 현재

1인 프로젝트

- 회로 편집·주변장치 설정 기능 구현
- 지원 범위의 C/HAL 동작 모델과 장치 상태 연동
- HAL 예제의 설정·회로·실행 결과를 한 앱에서 확인



애플리케이션 아키텍처



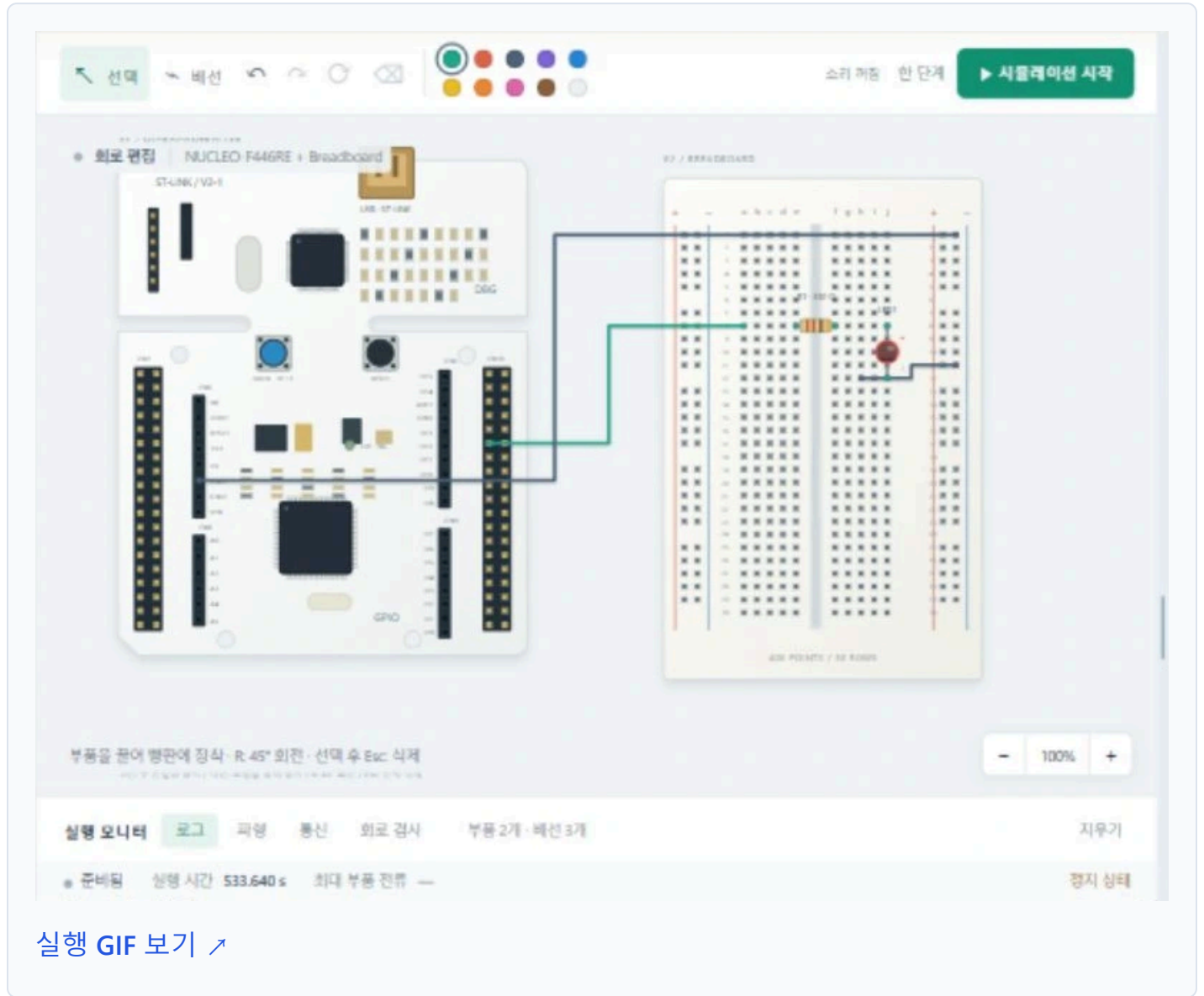
설정·배선·로직 실행 결과를 앱에서 확인

기존 실습 준비

- 핀· 주변장치 설정과 코드 작성 도구를 각각 확인
- 핀맵과 회로 배선 조건을 대조하며 실행 준비

개선 사항

- 설정과 배선을 오가던 확인 흐름을 하나의 작업 화면으로 통합
- 지원 C/HAL 동작 모델과 회로 상태를 연결해 프로그램의 실행 결과 확인



회로·설정 자동 저장과 프로젝트 파일 입출력

01

자동 저장

- 편집 상태를 JSON 문자열로 직렬화
- localStorage에 저장하고 앱 시작 시 복원

02

프로젝트 파일

- 회로·설정·소스를 .stm32lab 파일로 저장·불러오기
- 파일 대화상자를 통한 프로젝트 입출력

03

소스 연동

- .ioc·C/H 파일 가져오기
- 지원 범위의 HAL main.c 생성·내보내기

UART 시뮬레이션의 시간 경계 오류

문제

- 특정 입력 시점에서 UART 에코의 일부 문자 누락

원인

- $\mu s \rightarrow ms \rightarrow \mu s$ 변환 오차로 수신 시점 오판

해결

- 기존 송신 기준의 허용 오차를 수신 판정에도 적용
- 입력 시점·baud·tick을 교차한 24개 조건으로 회귀 검증

실행 모니터

로그

파형

통신

회로 검사

부품 1개 · 배선 4개

한 단계 실행

실행 시간 0.280 s

최대 부품 전류 0.33 mA

0.000 s [USART2 TX] HAL UART ready

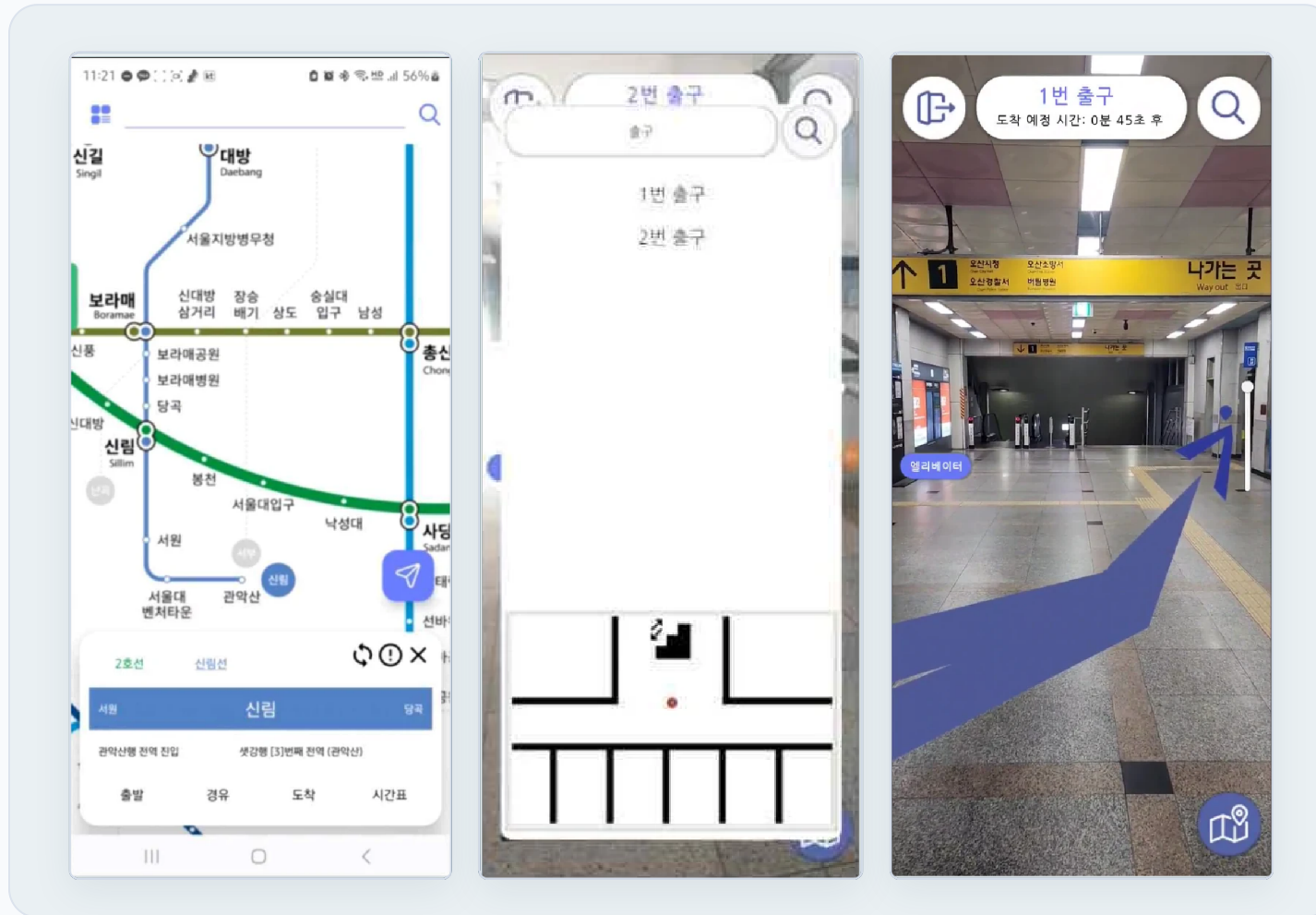
0.000 s HAL 프로그램을 시작했습니다.

0.261 s [USART2 RX] HAL log

0.261 s [USART2 TX] HAL log

RX·TX 모두 'HAL log' 일치

9600 baud · 260ms에 입력 · 20ms 한 단계 실행



BLE 비콘 기반 실내 측위·AR 길안내 시스템

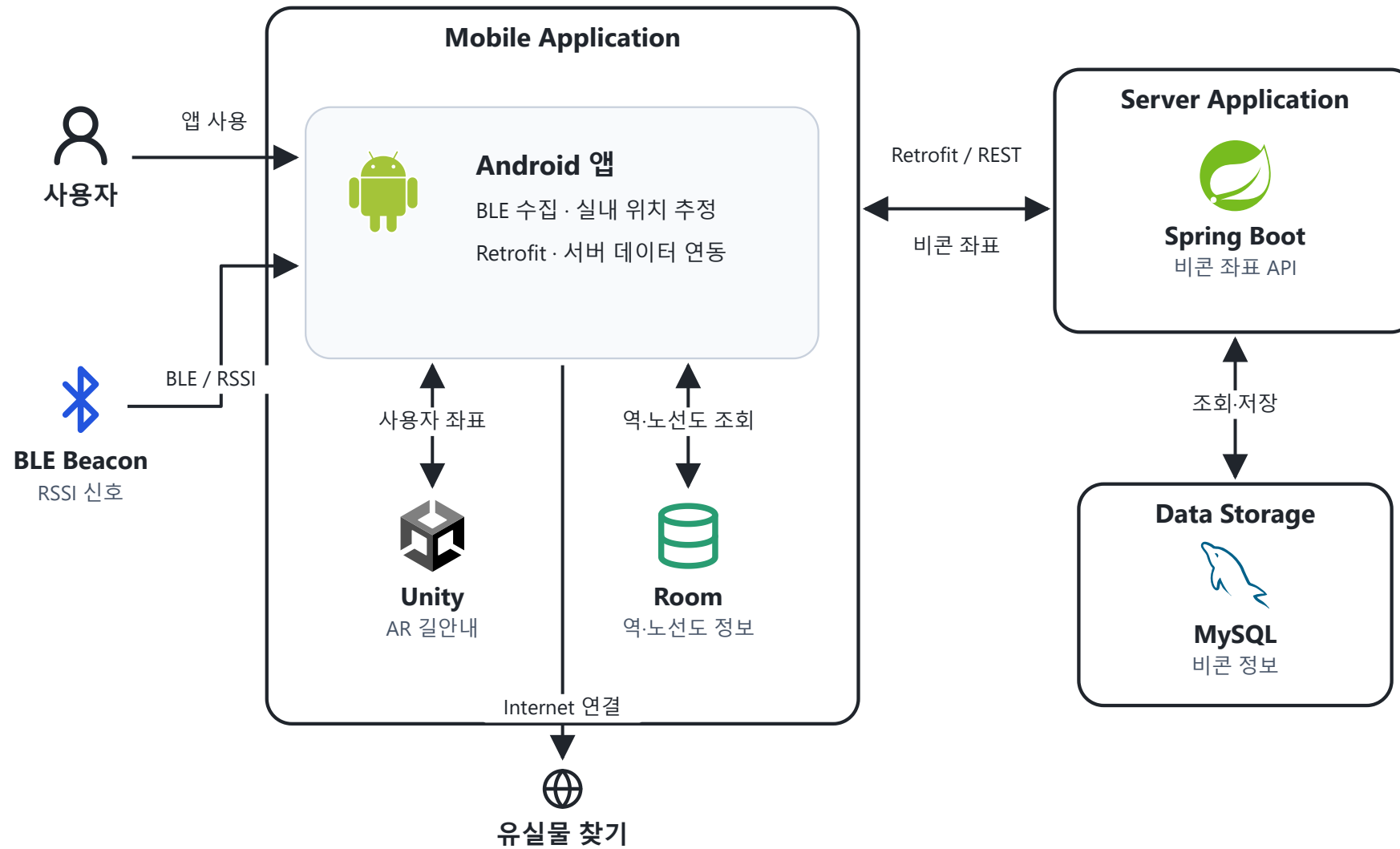
2024.03 — 2024.12

4인 프로젝트 · Android 개발

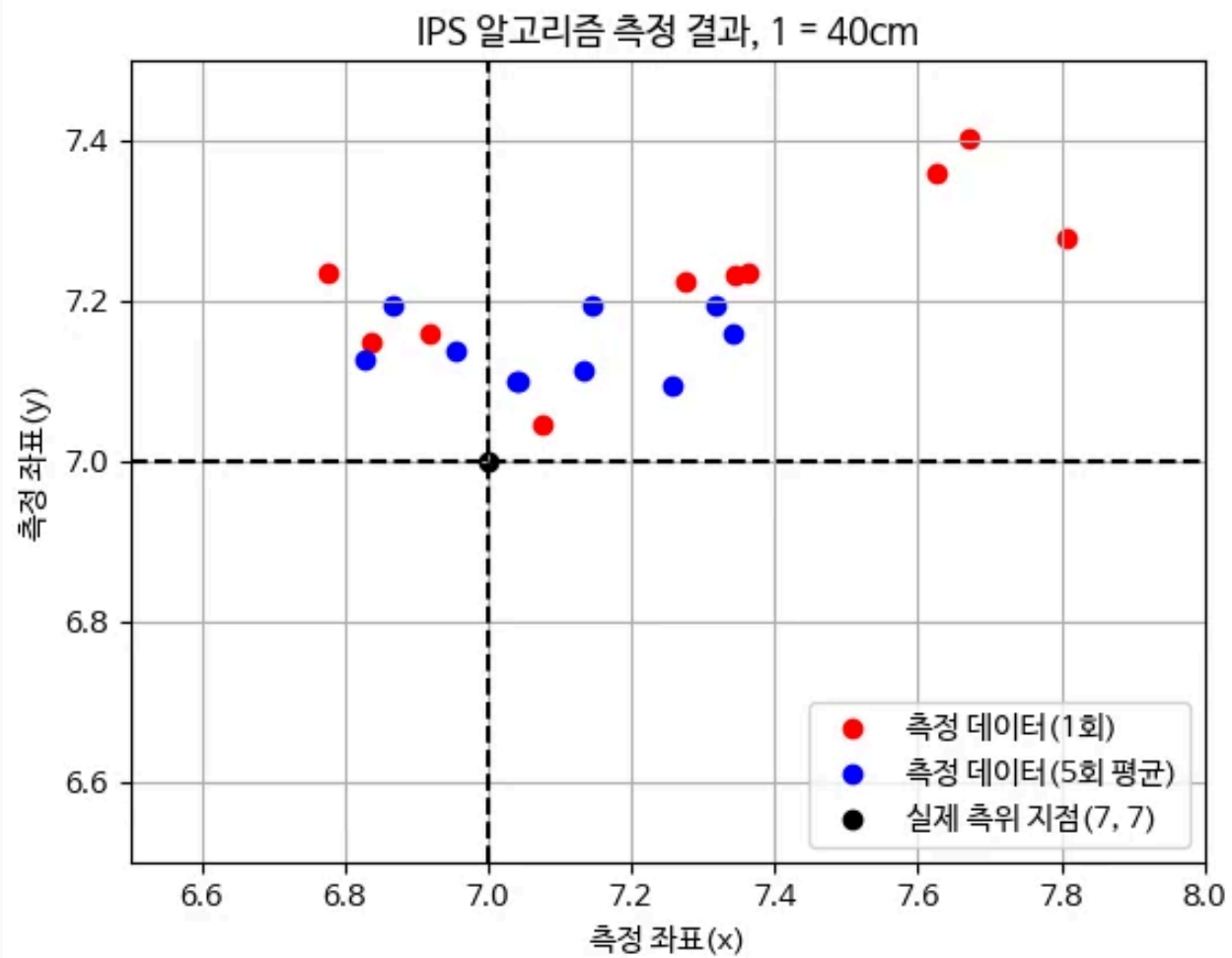
- 비콘 수집·실내 위치 추정·AR 좌표 전달 구현
- BLE 신호·서버 좌표·Unity를 연결해 현재 위치 표시
- 실내에서 현재 위치와 AR 경로를 확인하며 목적지 탐색



애플리케이션 아키텍처



측위 데이터 분포와 5회 좌표 평균



프로젝트 측위 실험의 1회·5회 평균 좌표 비교

비교 조건

- 오산역 · 비콘 3개 · Galaxy S22+
- 단일 측위 10회와 5회 평균 측위 10회 비교

기준

- 1회 측정 좌표 사용
- 측정 시점에 따라 위치 편차 발생

개선 사항

- 5회 측정 좌표의 평균값 사용
- 해당 실험의 1회·5회 평균 좌표 비교에서 편차 완화 경향 확인

Beacon, Unity 연동 및 실내 측위 구현

01

Beacon 연동

- Android BLE 스캔과 **UUID** 필터링으로 대상 비콘 수집
- **Minor** 기반 층 판별·RSSI 수신 처리

02

실내 측위

- **Retrofit**으로 서버의 비콘 좌표 조회·수신 목록 매칭
- RSSI 거리 추정·가까운 비콘 선별과 좌표 계산 공동 구현

03

Unity 연동

- 계산한 실내 좌표의 **Android-Unity** 전달 구현
- Android 앱 안에 **Unity AR** 실행 화면 통합

지원 중단된 API에 의한 Android ↔ Beacon 인식 문제

문제

- 기존 예제의 제공 API 사용 시 App Build 문제 발생

원인

- 제공된 예제 API의 지원 중단

해결

- Beacon API Reference에서 함수의 동작·인자를 비교해 바인딩·해제·탐색 API 교체
- **bind** → **bindInternal**
unbind → **unbindInternal**
startRangingBeaconsInRegion → **startRangingBeacons**



Android ↔ Unity 모듈 연동 문제

문제

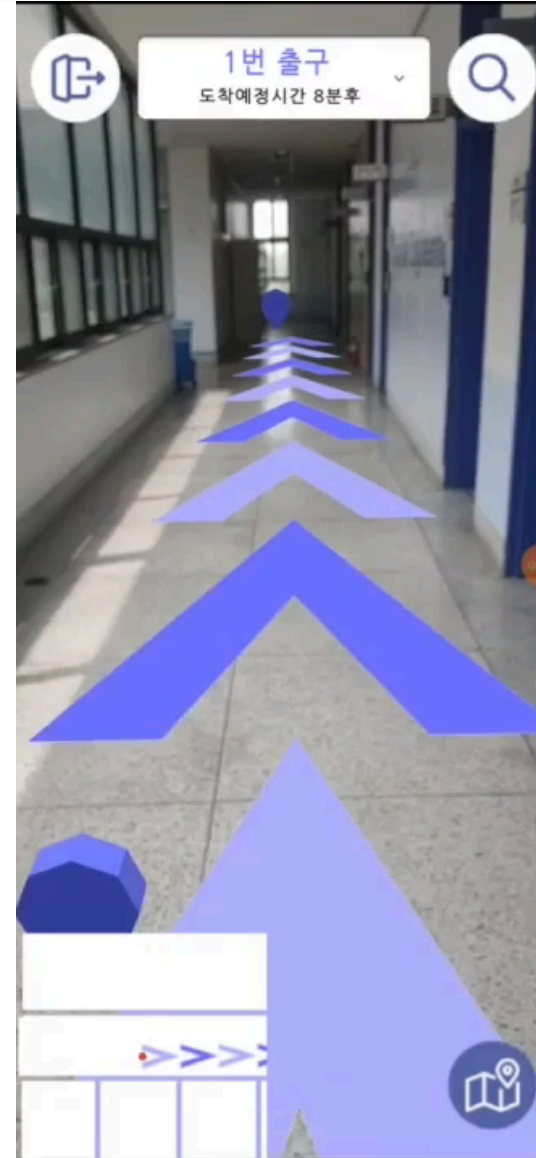
- 구버전 연동 예제 적용 후 Unity 모듈 빌드·실행 오류 발생

원인

- 예제와 현재 환경의 모듈·Java·SDK 설정 차이
- 당시 활성화한 Oculus 옵션 관련 오류 확인

해결

- Unity 모듈·의존성 연결과 빌드 설정 조정
- Java·SDK·namespace·NDK 경로 점검
- 해당 프로젝트의 Oculus 옵션 해제



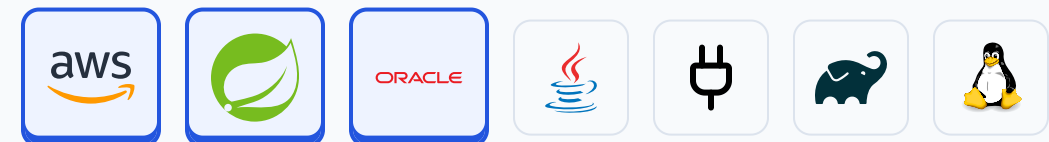


Web-Android에서 전자책을 조회·열람하는 서비스

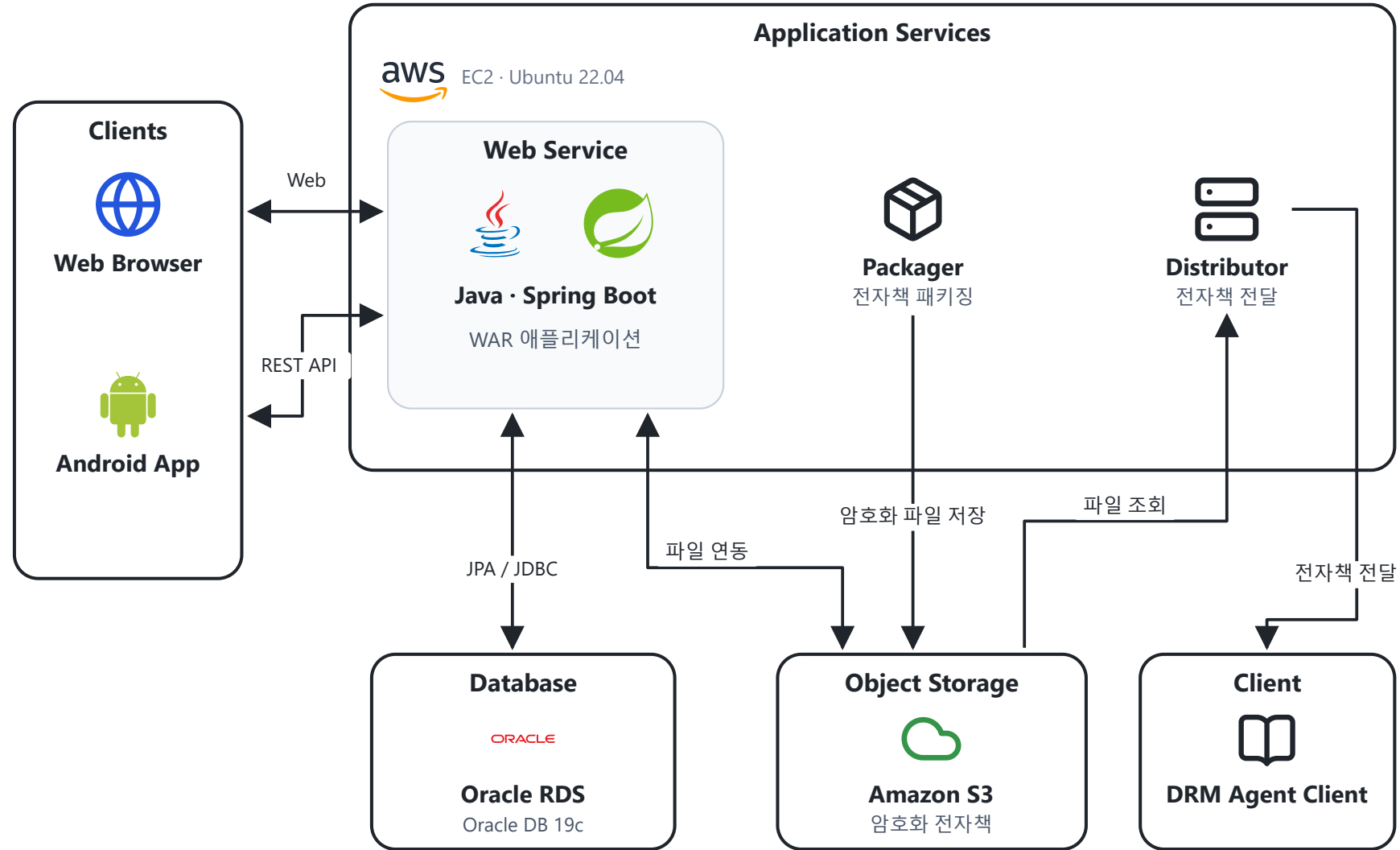
2025.02 — 2025.03

전체 10명 · Web 4명 · Web 팀장

- Oracle DB 관리·Web 배포·접근 권한 설정
- Spring Boot Web 서버와 RDS·S3를 연동해 데이터·파일 처리
- 사용자가 Web-Android에서 전자책을 조회·열람하는 서비스 지원



애플리케이션 아키텍처



Oracle DB 이전 및 Web 서비스 배포

01

DB 운영 환경 이전

- Oracle 19c → RDS 이전
- Web 서버의 JDBC 연결 구성

02

Web 서비스 배포

- Gradle로 WAR 빌드
- EC2 배포·실행 및 Web 접속 확인

03

접근 범위 설정

- DB 관리자·Web EC2 기준 RDS 접근 제어
- 관리자 SSH와 Web 서비스 포트 구분

Spring Boot 버전 변경에 따른 Jakarta·javax 호환성 문제

문제

- Spring Boot 3.0.4 → 2.7.8 전환 과정에서 기존 Servlet·JPA 코드와 의존성의 호환성 문제 발생

원인

- Boot 3 계열의 jakarta.* 코드와 Boot 2 계열의 javax.* API 환경 불일치

해결

- Servlet·JPA의 클래스 이름이 같아도 패키지명이 다르면 별개의 타입임을 확인
- Jakarta → javax import와 Servlet·JSTL 의존성 조정
- Servlet 5.0.0 → 4.0.4, JSTL 2.0.0 → 1.2로 정합화

변경 전 · Spring Boot 3.0.4

build.gradle

```
5 id 'org.springframework.boot' version '3.0.4'
```

build.gradle

```
53 implementation 'jakarta.servlet:jakarta.servlet-api:5.0.0'
54 implementation 'jakarta.servlet.jsp.jstl:jakarta.servlet.jsp.jstl-api:2.0.0'
55 implementation 'org.glassfish.web:jakarta.servlet.jsp.jstl:2.0.0'
```

QnaController.java

```
9 import jakarta.servlet.http.HttpServletResponse;
10 import jakarta.servlet.http.HttpSession;
```

Qna.java

```
4 import jakarta.persistence.*;
```

변경 후 · Spring Boot 2.7.8

build.gradle

```
5 id 'org.springframework.boot' version '2.7.8'
```

build.gradle

```
59 implementation 'jakarta.servlet:jakarta.servlet-api:4.0.4'
60 implementation 'javax.servlet.jsp.jstl:jstl-api:1.2'
61 implementation 'javax.servlet:jstl:1.2'
```

QnaController.java

```
9 import javax.servlet.http.HttpServletResponse;
10 import javax.servlet.http.HttpSession;
```

Qna.java

```
4 import javax.persistence.*;
```

외부 Tomcat WAR 배포 환경의 클라우드 접속 오류

문제

- 실습용 WAR 배포 환경에서 클라우드 접속 오류

원인

- WAR 실행 환경에 맞는 EL API·구현체 의존성 구성 미비

해결

- EL API·Tomcat EL 구현 의존성 보완
- javax.el 3.0.0과 tomcat-embed-el 9.0.58 추가

변경 전 · WAR 실행 의존성

build.gradle

```
76
77 // 내장 톰캣 미사용
78 providedRuntime 'org.springframework.boot:spring-boot-starter-tomcat'
```

변경 후 · EL API·구현 의존성 추가

build.gradle

```
76
77 // 내장 톰캣 미사용
78 providedRuntime 'org.springframework.boot:spring-boot-starter-tomcat'
79
80 // EL API (Expression Language) 추가
81 implementation 'org.glassfish:javax.el:3.0.0'
82
83 // Tomcat의 EL 구현체 추가
84 implementation 'org.apache.tomcat.embed:tomcat-embed-el:9.0.58'
```



원격 DB의 백업·복원을 관리하는 웹 기반 백업 관리 시스템

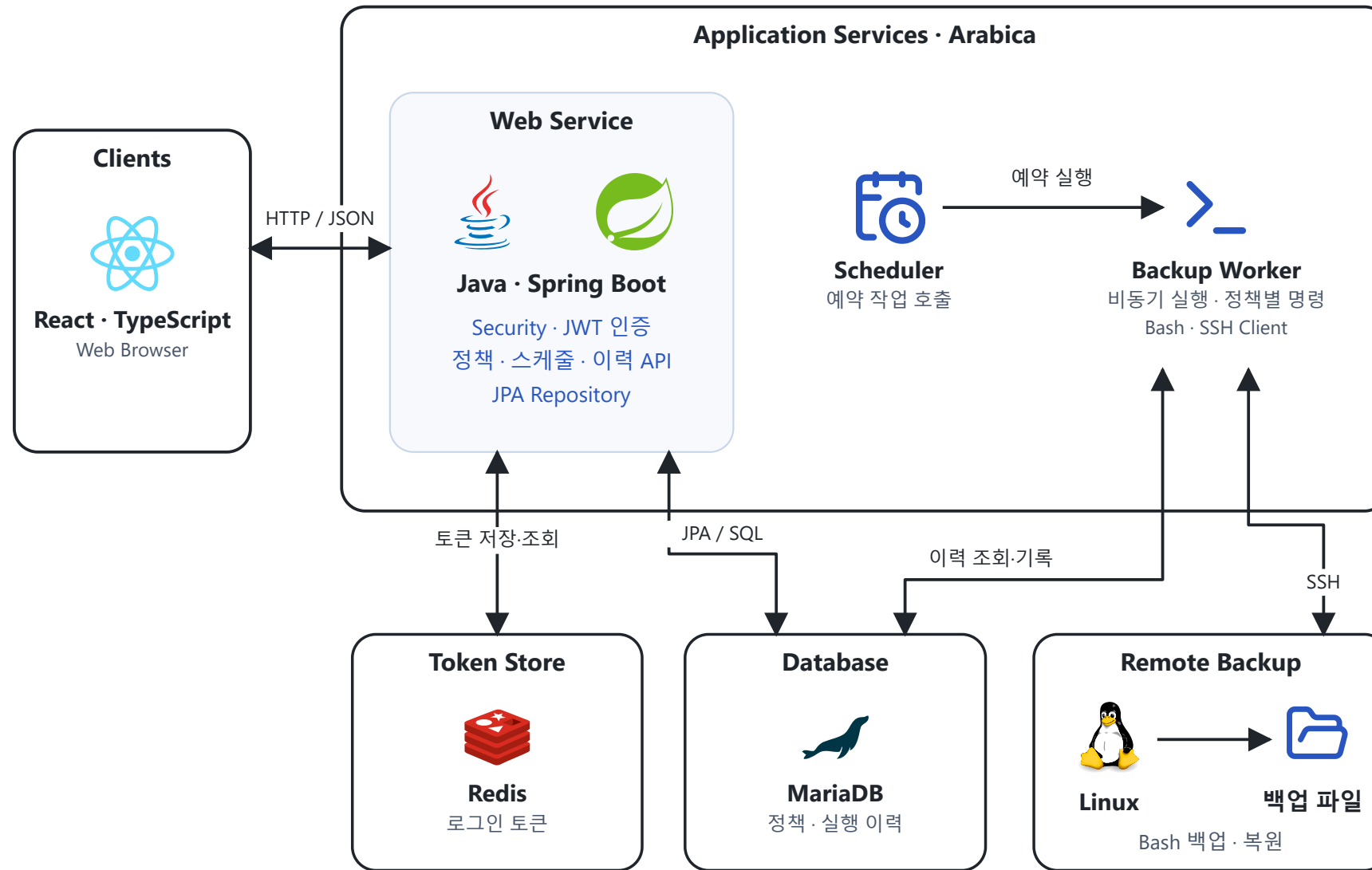
2025.06 — 2025.08

2인 프로젝트 · 백엔드 개발

- 전체·증분·차등 백업과 예약 실행·복원 API 구현
- SSH·Bash 원격 실행과 결과 파일 판독·검증 연동
- 반복 백업을 예약하고 완료 결과·복원 이력을 웹에서 확인



정책·예약 실행·이력을 연결하는 백엔드



DB 백업 스케줄 관리와 외부 서버 백업·복원

01

정책·예약 실행

- 전체·증분·차등 방식별 실행 명령 구성
- 사용자 지정 일정과 정책을 연결한 예약 실행

02

결과 판독·검증

- 결과 파일 판독 후 저장 서버에서 검증 수행
- MySQL 테이블·행 수 비교와 불일치 정보 기록

03

이력·복원

- 백업 시작 시각·성공 여부·오류 정보 저장
- 선택한 백업 파일의 복원 실행과 복원 이력 기록

DB 백업 시 장시간 Thread 점유 문제

문제

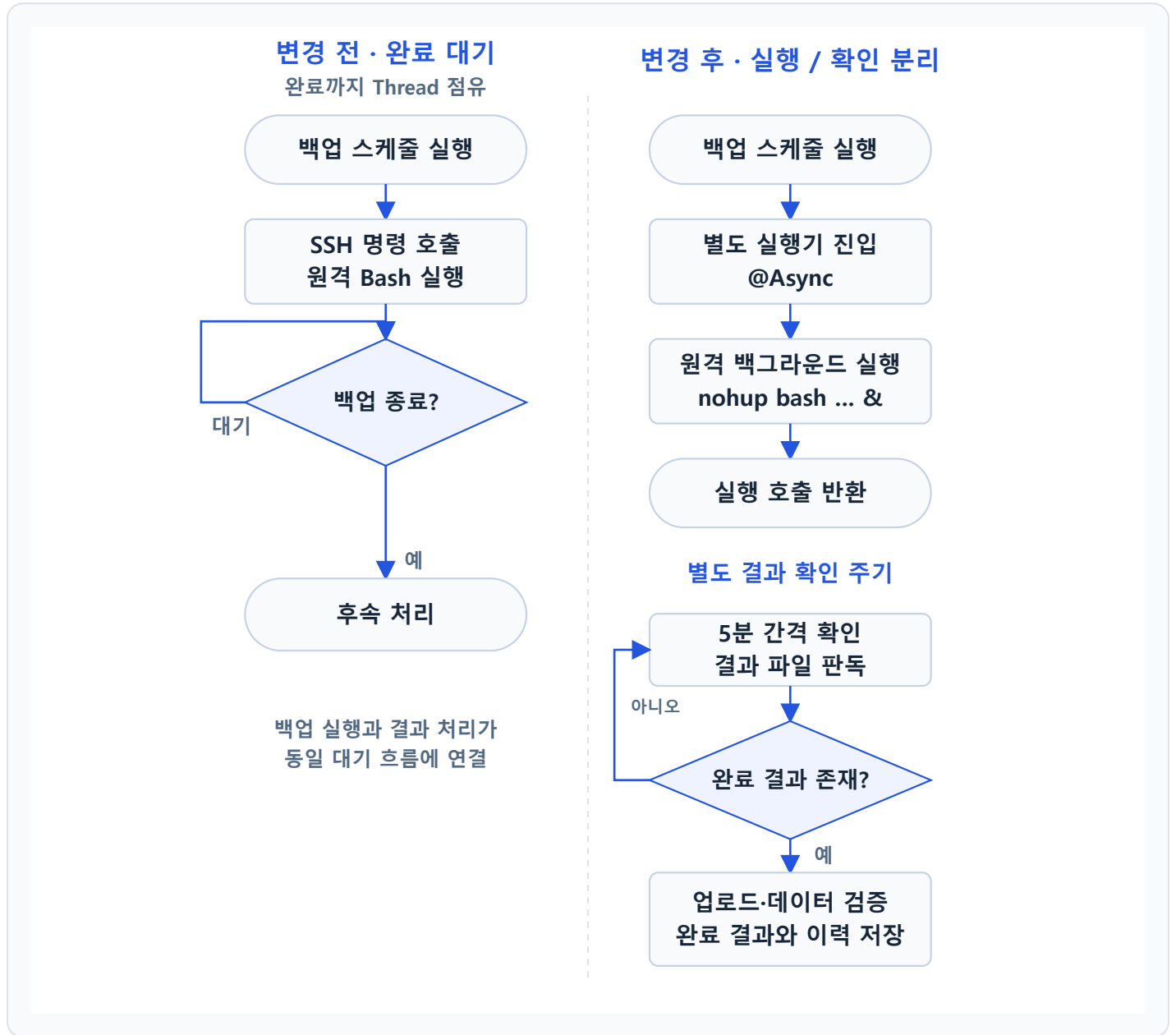
- 10개 이상의 DB 백업 시 오류 발생

원인

- DB 백업 완료를 기다리는 동안 Thread 점유

해결

- 스케줄 백업 진입점을 별도 실행기로 비동기 처리
- 원격 명령을 백그라운드 실행으로 변경
- 실행 직후 성공 기록을 제거하고 결과 판독·검증 후 이력 저장



비동기 실행 다음으로, 완료 확인과 실행 권한 보완

추가 보완안 · 미구현

- 구현한 백그라운드 실행·결과 판독·검증을 바탕으로, 작업 누락·중복·과부하와 원격 실행 권한까지 관리하는 방향 검토



완료 확인의 지연·누락

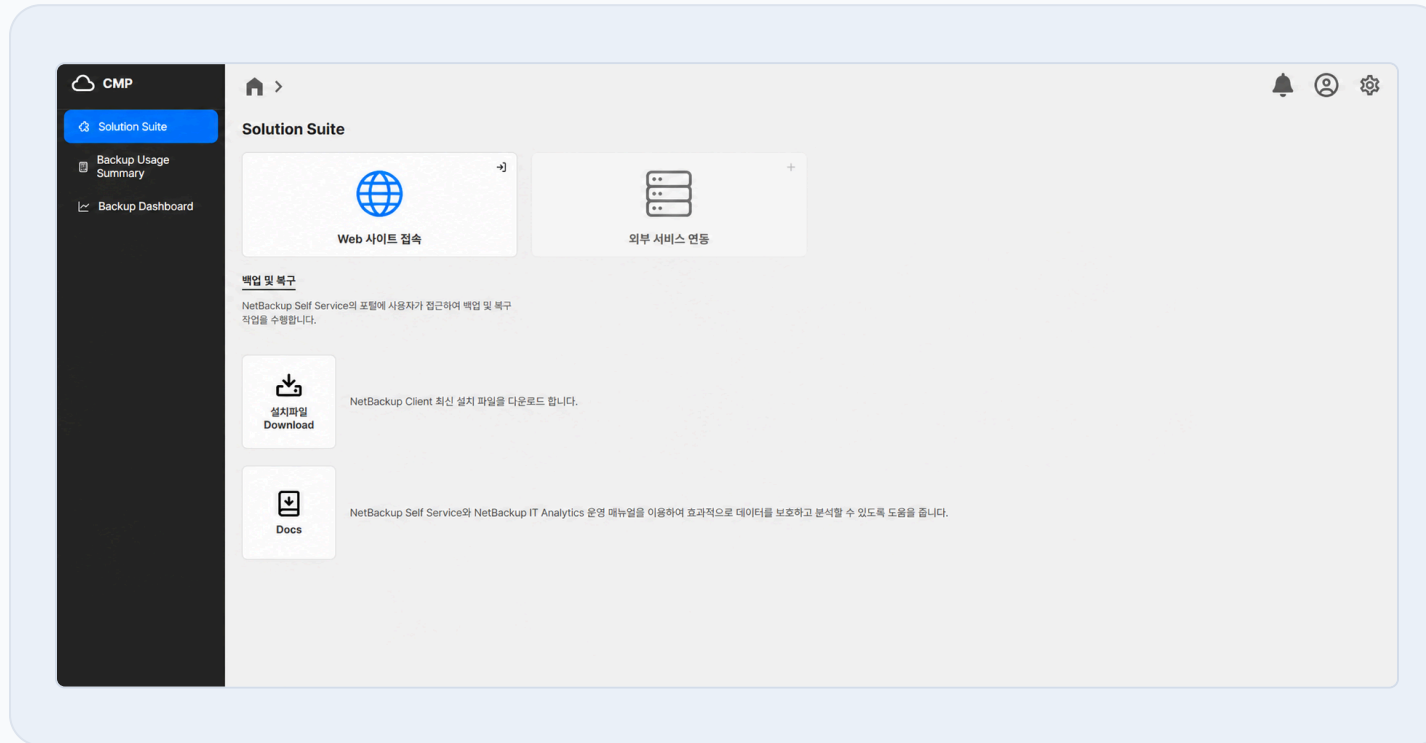
- 현재 결과 파일을 주기적으로 확인하는 방식의 대기 구간
- 인증된 완료 알림 + 누락 시 재확인으로 보완
- 작업 ID·단계별 상태·타임아웃으로 미완료 작업 추적

과부하·중복 결과 대응

- 실행기 포화 시 호출 스택에서 작업이 실행될 가능성
- 동시 실행·대기량 제한과 거절·재시도 정책 명확화
- 동일 작업 ID의 중복 결과를 한 번만 반영

원격 실행 권한 제한

- SSH 명령 실행 계정과 허용 범위를 더 엄격히 제한하는 방향 검토
- 전용 계정·키 인증·호스트 키 검증 적용
- 허용한 스크립트·인자만 실행하도록 제한

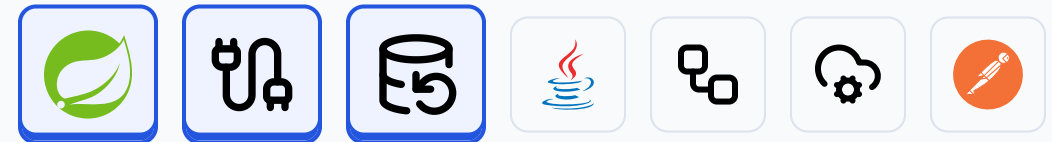


외부 백업 서비스의 데이터·정책을 조회하는 API 연동 포털

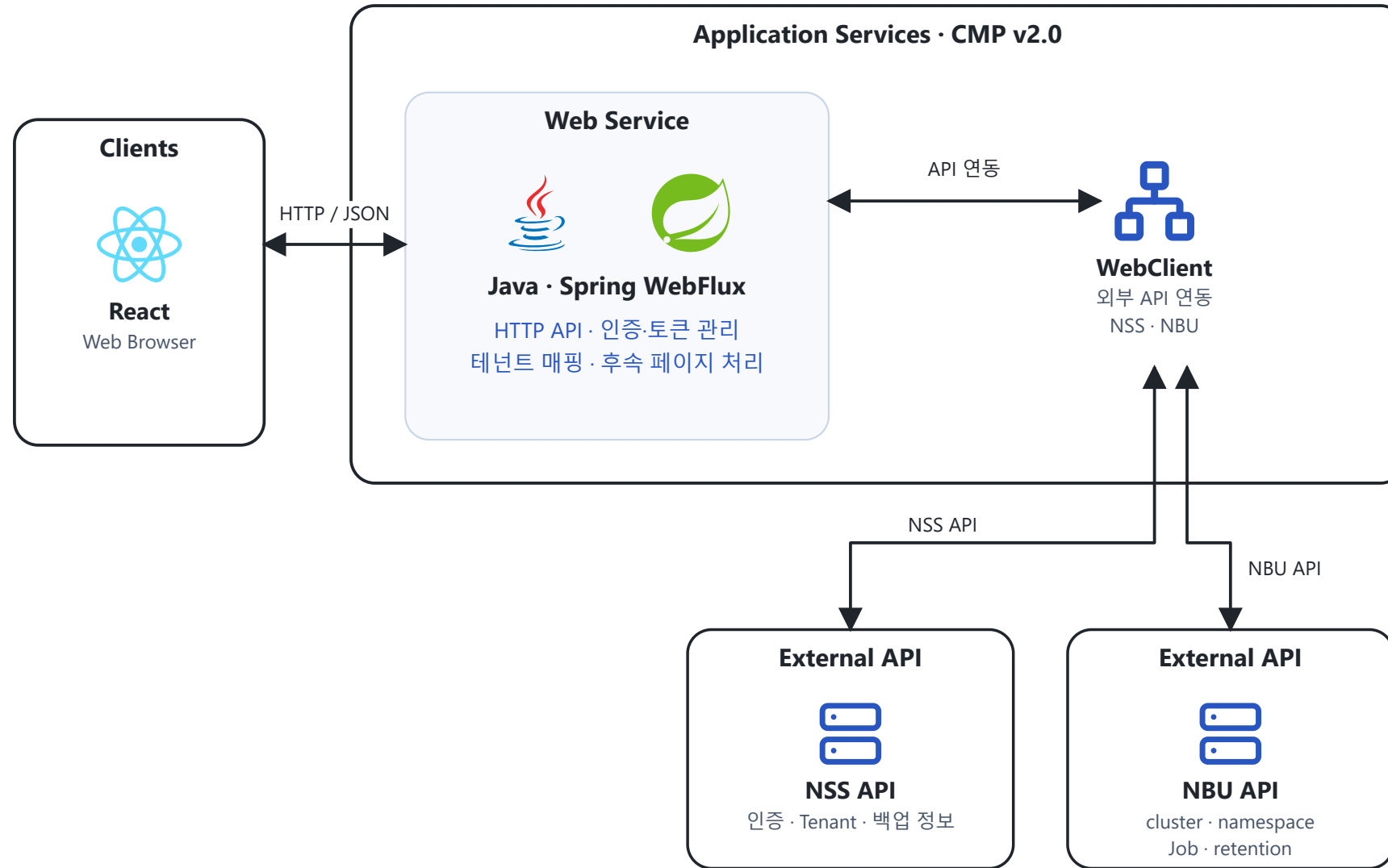
2025.06 — 07 / 2026.01 — 02

3인 프로젝트 · API 연동

- 백업 데이터·테넌트별 Kubernetes 백업 정책 조회 API 구현
- Spring WebFlux·WebClient로 외부 NSS·NetBackup API 연동
- 외부 백업 서비스의 현황과 정책을 포털에서 확인



NSS·NBU 외부 API와 테넌트별 조회 연동



대량 데이터 조회 시 외부 API 응답 대기로 인한 처리 지연

문제

- 데이터 증가에 따라 기존 MVC 처리 구조에서 장애 발생

원인

- 동기 외부 API 호출에서 응답을 기다리는 동안 요청 스레드 점유

해결

- MVC → Spring WebFlux로 프레임워크 환경 교체
- WebClient·Mono 기반으로 외부 API 응답 처리



Kubernetes 백업 연동 확장에 따른 테넌트별 정책 조회 문제

문제

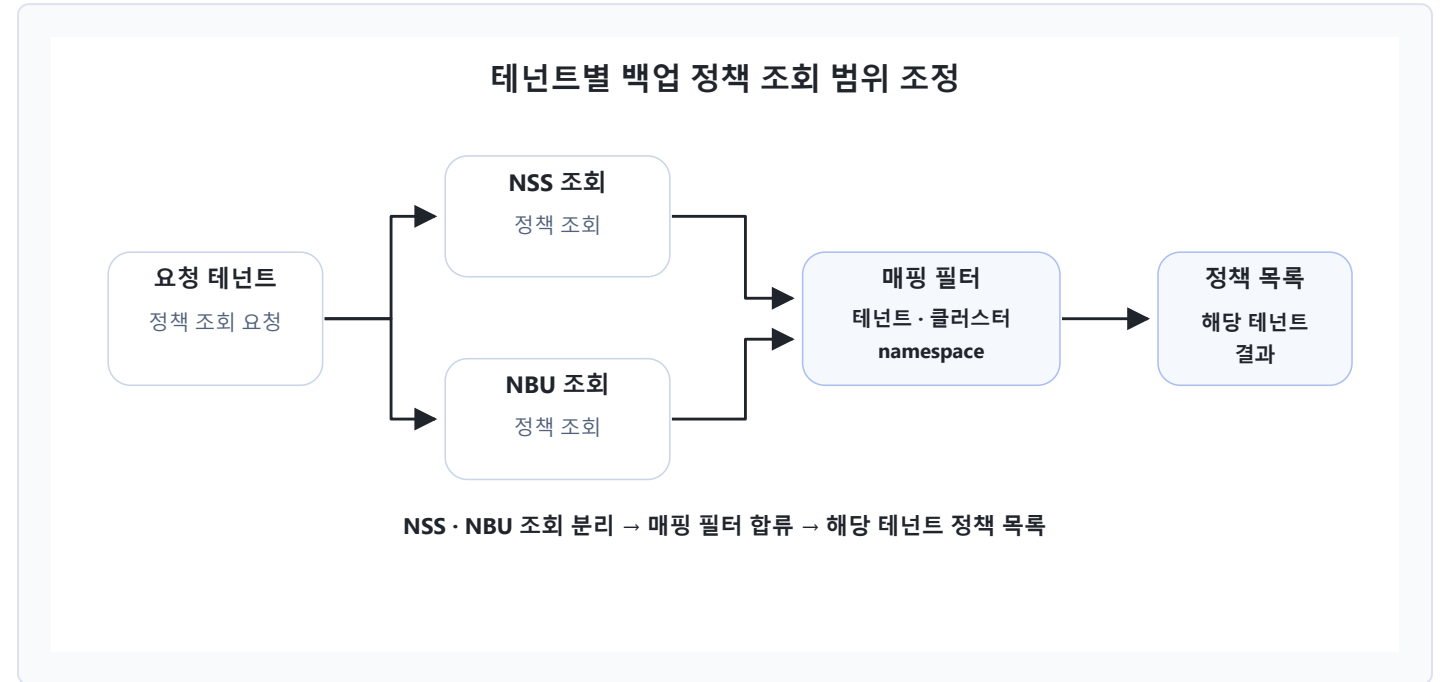
- Kubernetes 백업 정책이 모든 테넌트에 표시

원인

- 조회 결과를 요청 테넌트의 클러스터·namespace에 맞게 구분할 필요

해결

- NSS·NBU 조회 경로 분리와 페이지·API 로그 점검
- 테넌트·클러스터·namespace 매핑 필터 적용



평균 응답시간·P95·오류율을 함께 비교

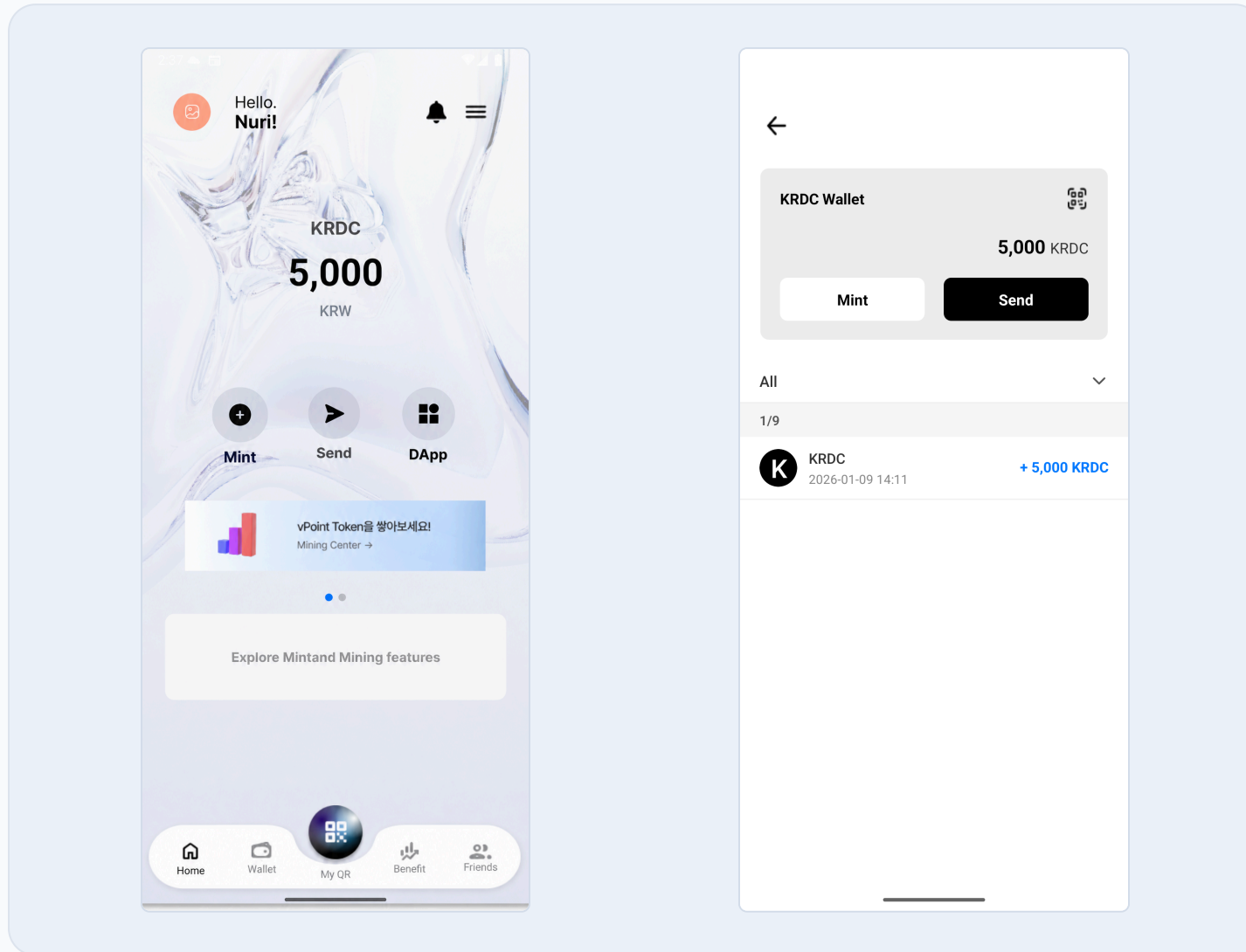
시험 조건

- 동일 호스트 사양에서 Ubuntu·RHEL 환경 비교
- Postman Runner · 3만·20만 건 · 동시 사용자 1·5·10명 · 각 1분

판단

- 20만 건·5명에서 Ubuntu의 평균 응답은 더 짧지만 오류율 10% 기록
- 20만 건·10명에서 두 환경 모두 오류 발생
- 평균 응답만으로 판단하지 않고 P95·오류율로 고부하 한계 확인



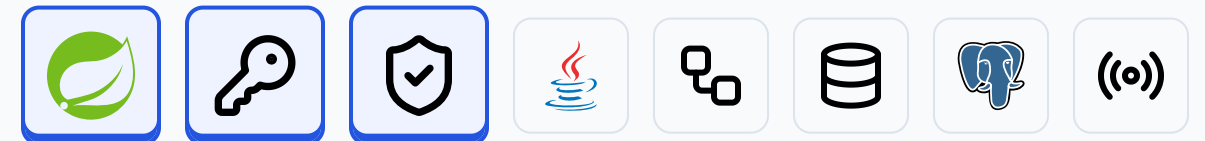


회원·지갑·인증·알림을 제공하는 모바일 디지털 지갑 서비스

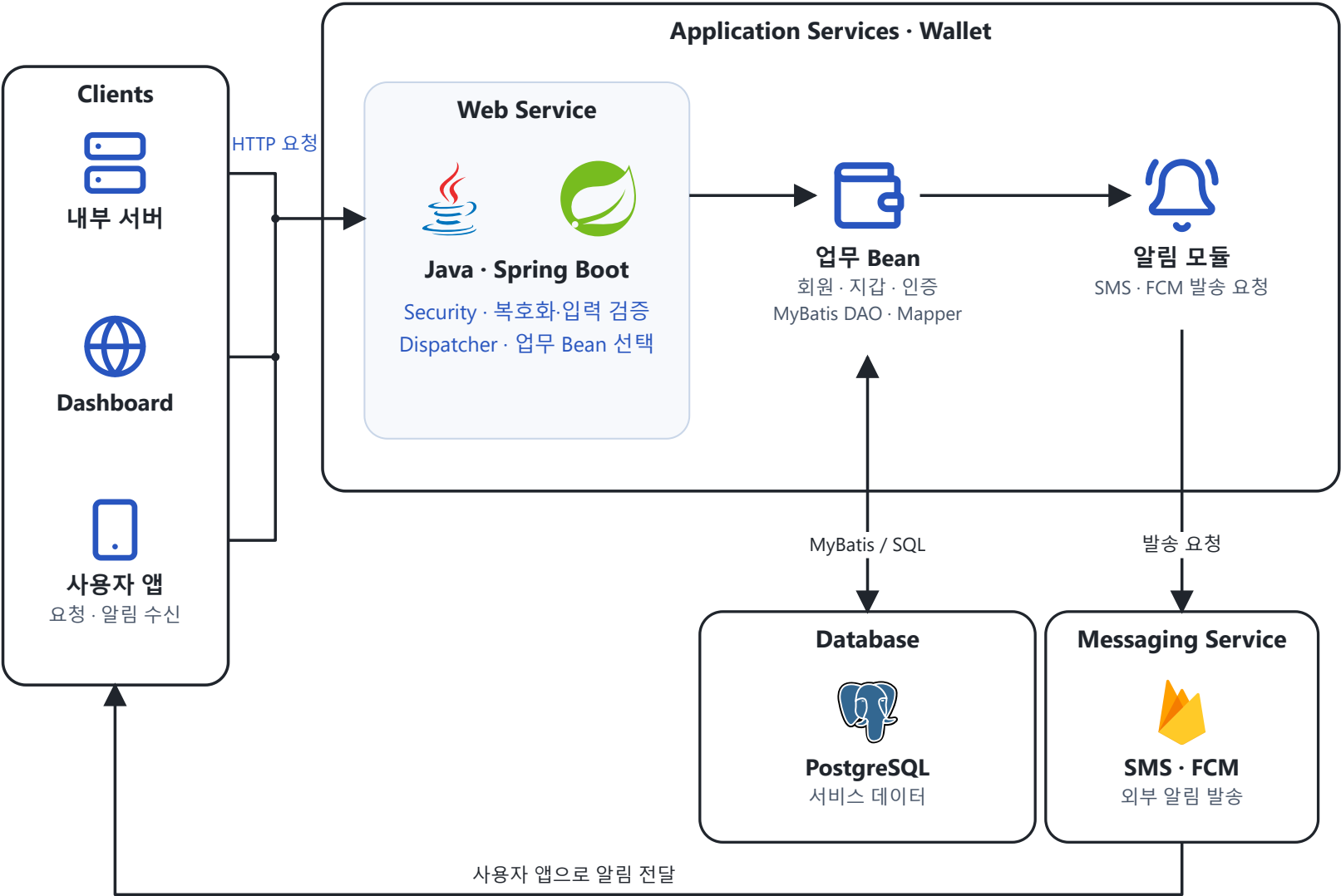
2025.09 — 2026.01

4인 프로젝트 · 백엔드 개발

- 회원·지갑 API와 QR 로그인·SMS·FCM 알림 구현
- 앱 요청을 백엔드·PostgreSQL·외부 알림 서비스와 연동
- 앱에서 지갑 관리·Store 로그인 승인·알림 확인



보안 필터·Dispatcher·업무 API 요청 처리



보안 처리·Web-App 로그인·알림 연동

01

보안 기능 구현

- RSA 통신 암호화와 키 발급·공개키 만료 검증
- AntiSamy 입력 검사와 정제값의 실제 요청 본문 반영

02

SSE·QR 로그인 연동

- Web·App 간 QR 인증 결과 전달
- 인증 상태 확인 후 SSE 로그인 완료 알림

03

SMS·FCM 알림

- 인증 문자·사용자 안내 SMS 전송과 결과 기록
- FCM 수신 토큰 등록과 푸시 알림 전송

XSS 정제 결과를 실제 요청 본문에 반영

문제

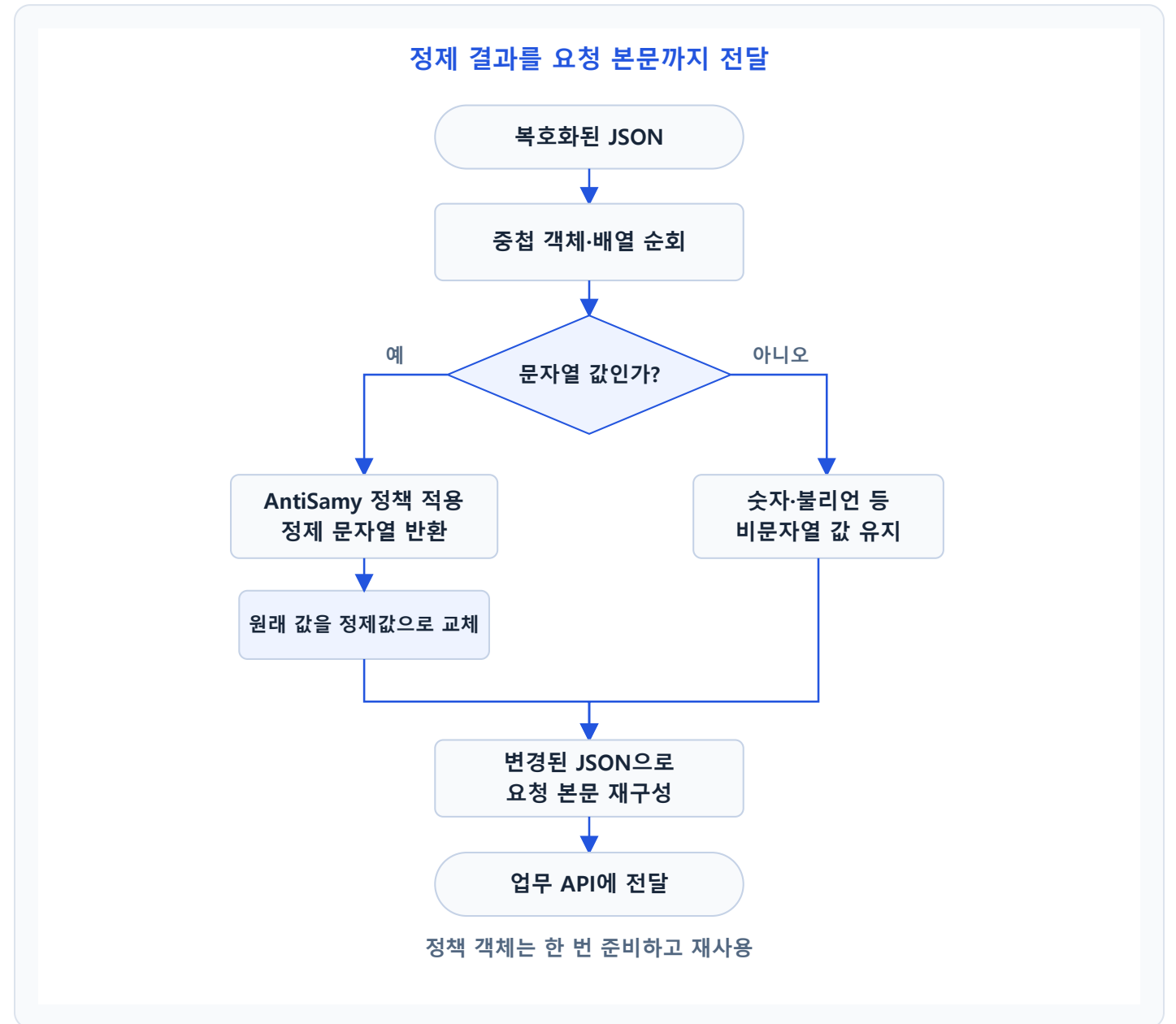
- 필터 내부에서 정제한 문자열이 실제 요청 데이터에 반영되지 않는 구조

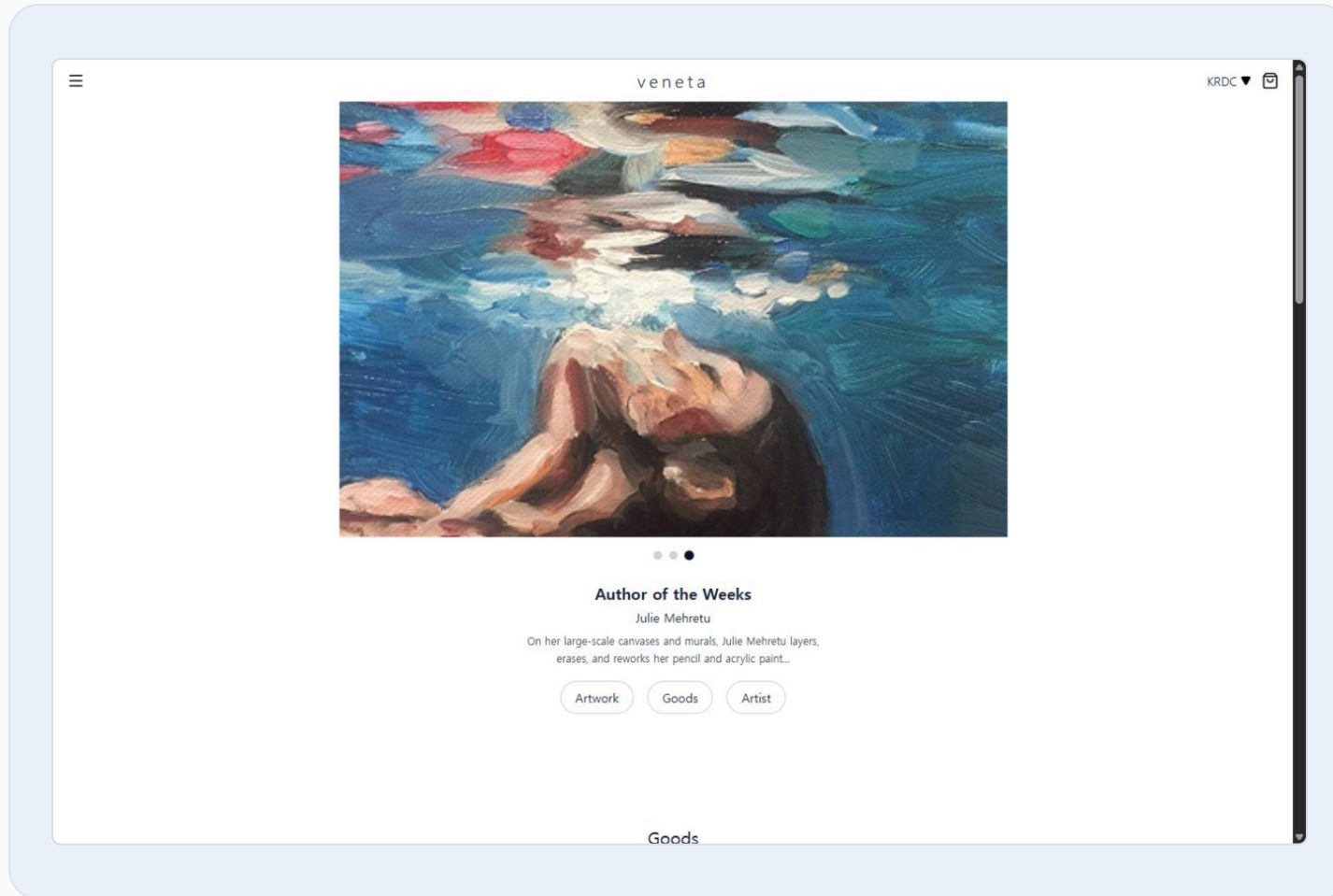
원인

- 문자열을 변경해도 통과 여부만 반환하고 원래 값을 교체하지 않음

해결

- 중첩 JSON 객체·배열을 재귀 순회해 정제한 값으로 교체
- 교체한 값으로 요청 본문을 재구성해 업무 처리에 전달
- 정책을 클래스패스에서 읽고 정책 객체를 1회 준비해 재사용





상품 탐색·장바구니·결제를 지원하는 Wallet 연동 커머스 서비스

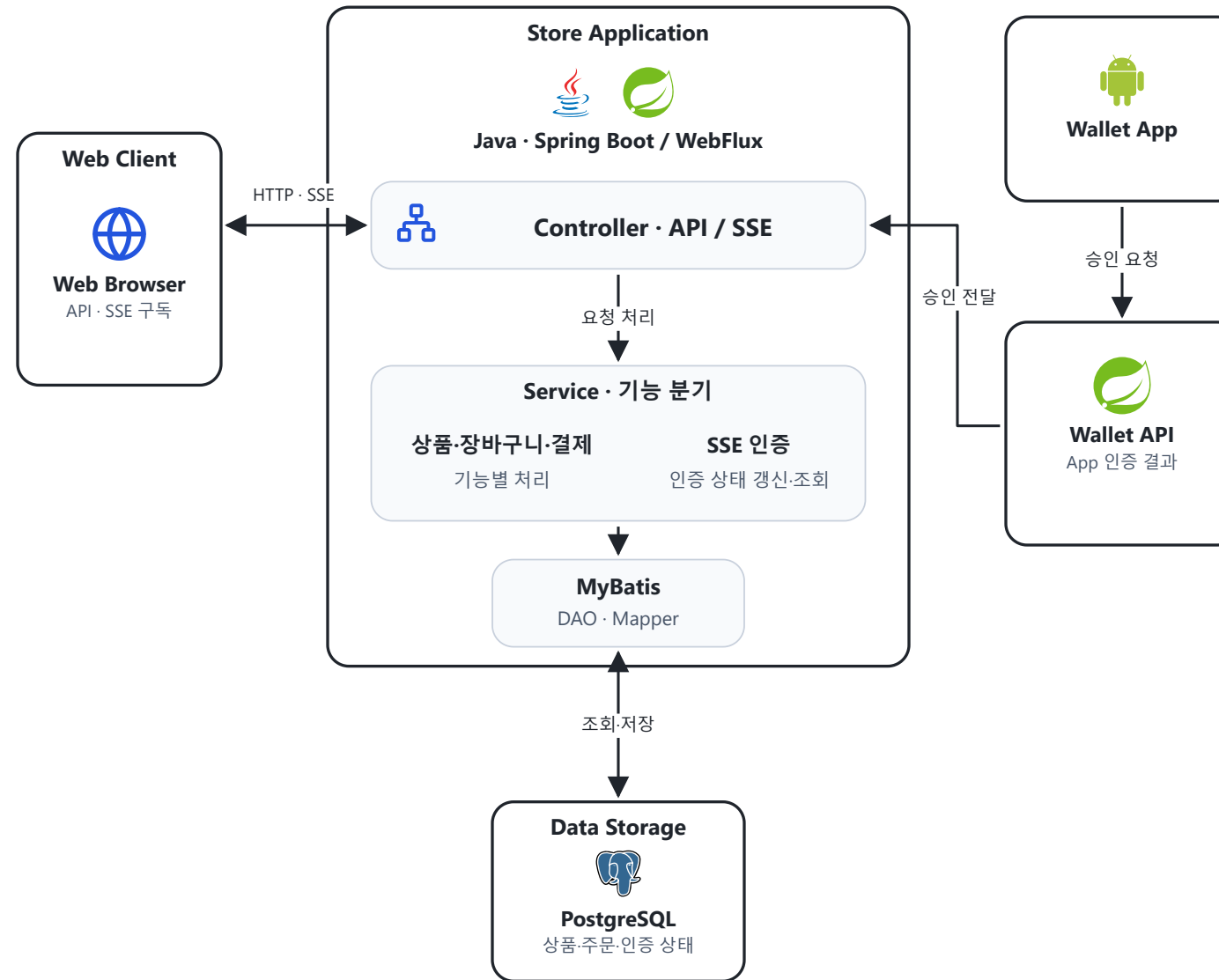
2025.09 — 2026.01

4인 프로젝트 · 백엔드 개발

- 상품 조회·Wallet 인증 결과 전달 API 구현
- MyBatis·PostgreSQL로 데이터 처리, SSE로 로그인 결과 전달
- Wallet 앱 인증으로 웹 쇼핑몰에 로그인하고 상품 구매



커머스 API·Wallet 인증·데이터 저장 연동



DB 전환·관계 구조 정리·SSE 인증 연동

01

DB Migration

- MySQL → PostgreSQL 전환
- 대상 DB에 맞춰 DDL·SQL과 드라이버 변경

02

스키마 관계 정리

- 장바구니·결제 옵션의 FK 참조 대상 교정
- 중간 매핑 구조 정리와 장바구니 복합키 반영

03

SSE 인증 연동

- 웹의 SSE 구독과 Wallet 인증 승인 연결
- DB에 인증 상태를 저장·조회해 로그인 결과 전달

Store DB 마이그레이션의 외래키 참조 오류

문제

- Store의 MySQL → PostgreSQL 마이그레이션 실패

원인

- 장바구니.결제 옵션의 외래키가 유일하지 않은 상품.옵션 컬럼을 참조
- PostgreSQL의 참조 대상 유일성 조건과 불일치

해결

- 각 행을 식별하는 기존 부모 PK를 참조하도록 외래키 컬럼 수정
- 장바구니.결제의 옵션 연결 관계 4개 교정



장문수

개발 기록과 연락처

Email msanstn@gmail.com ↗

GitHub github.com/moonsyu ↗

Velog velog.io/@moonsyu ↗
