조 현 태

+821048194519 @ whgusxo2055@gmail.com

호서대학교 컴퓨터공학부에 재학 중이며, 디지털융합연구실 학부연구생으로 백엔드·분산 시스템을 연구하는 개발자입니다.

Java/Spring 기반 서비스에서 자동 채점, 결제·인증, 실시간 알림, 메시징 기반 비동기 처리, Kubernetes 오토스케일링을 구현했습니다.

CodeQuest 프로젝트에서는 WebFlux와 JPA 조합의 기술 정합성 문제를 Spring MVC 전환으로 해결하고, N+1 쿼리 제거, Kafka 기반 채점 결과 처리, KEDA 오토스케일링을 적용했습니다. 그 결과 로그인 p50 25,242ms → 159ms, 제출 p95 2,320ms → 271ms, Java 채점 실행시간 0.844s → 0.060s로 개선했습니다.

실운영 서비스인 한국비건인증센터 인증 플랫폼에서는 PortOne 결제 검증, 사업자등록증 OCR 자동화, SSE 기반 인증 상태 알림을 구현했습니다. 성능 수치와 운영 로그를 기준으로 문제를 정의하고, 아키텍처·DB·인프라 레벨에서 개선하는 데 강점이 있습니다.

경력 신입

CodeQuest - 코딩 교육 플랫폼

2026.01 - 2026.06

- 역할: 백엔드 주도 개발
- 형태: SW중심대학사업단 산학협력 프로젝트 / 호서대학교 컴퓨터공학부 정규 수업 실사용
- 기술: Java 17, Spring Boot 3.4.3, Spring MVC, Spring Cloud Gateway, Gradle Multi Module, Kafka, KEDA, Redis, MySQL, JPA/Hibernate, Kubernetes, GitHub Actions
- 링크: https://github.com/hoseo-dclab/CodeQuest_server / https://codequest.hoseo.ac.kr

코스 → 퀘스트 → 문제 계층으로 구성된 코딩 교육·자동 채점 플랫폼의 백엔드입니다.
데모 단계 프로젝트를 인수해 백엔드 아키텍처를 재설계하고 구현·배포·성능개선까지 주도했습니다. SW중심대학사업단과의 산학협력 프로젝트로, 호서대학교 컴퓨터공학부의 자료구조·알고리즘 등 정규 수업에서 학생들이 실제로 사용 중인 운영 서비스입니다. 수업 실사용 과정에서 드러난 아키텍처·통신·실행·확장 4개 층위의 병목을, 추측이 아니라 부하테스트·운영 DB 측정에 근거해 차례로 해결한 것이 핵심 경험입니다.

주요 성과

- WebFlux + JPA 조합의 블로킹 병목을 분석하고 Spring MVC로 전환해 제출 경로 성능 개선
  - p50 886ms → 81ms
  - p95 2,320ms → 271ms
  - SLA 300ms 위반율 77.5% → 4.5%
- 단일 파드 병목을 JMeter와 kubectl top으로 확인하고 KEDA 기반 수평 확장 적용
  - 로그인 p50 25,242ms → 159ms

  - 로그인 p95 44,166ms → 256ms
  - 성적 조회 p50 15,729ms → 2,212ms
- 성적 조회 N+1 문제를 IN + JOIN FETCH 단일 쿼리로 개선
  - 요청당 최대 1,500 SELECT → 1 SELECT
  - Submit 목록 21쿼리 → 1쿼리
- Java 채점 워커의 JVM 콜드스타트 비용 제거
  - 테스트케이스별 JVM 실행 → 단일 JVM Multi-Runner 구조로 전환
  - 평균 실행시간 0.844s → 0.060s
  - 1초 이상 실행 비중 54% → 0%
- 채점 결과 회신을 HTTP POST에서 Kafka 토픽 기반 구조로 전환
  - retry/API Key/콜백 엔드포인트 제거
  - 워커와 서버 간 결합도 감소
  - 브로커 보관 기반 결과 유실 방지
- Gradle 멀티모듈과 GitHub Actions 선택 배포 구성
  - 단일 산출물 → 5개 독립 배포 단위
  - 변경 모듈만 재빌드·재배포하도록 배포 범위 축소
- Kafka Consumer Lag 기반 KEDA 오토스케일링 설계
  - 언어별 채점 워커를 consumer group/topic 단위로 분리
  - 제출 폭주 시 대기열 기반 선행 확장 구조 구성
  - 해당 설계를 논문화해 정보통신학회 2026 춘계종합학술대회 학생우수논문상 수상

한국비건인증센터 비건 인증 플랫폼

2025.06 - 2026.06

- 역할: PortOne 결제 연동, JWT 인증, 사업자등록증 OCR, 인증 도메인, SSE 구현
- 형태: 기업 연계 실운영 프로젝트
- 기술: Java 21, Spring Boot 3.5, Spring Security + JWT, PortOne, MySQL, Redis, WebFlux, PDFBox, Google's Vision AI, SSE
- 링크: https://github.com/hoseo-dclab/Vegun_Backend / https://kvegan.co.kr

기업이 비건 인증을 신청 → 서류 제출 → 심사 → 인증서 발급 받는 전 과정을 처리하는 백엔드입니다.
한국비건인증센터와 연계해 실제 운영 중인 서비스이며, 인증 신청에 필수인 유료 결제의 외부 PG(PortOne) 연동을 직접 담당하고, JWT 인증, 사업자등록증 자동 추출 OCR을 개선했습니다.

주요 성과

- PortOne 결제 사전등록 및 서버 검증 플로우 구현
  - 클라이언트 결제 금액을 신뢰하지 않고 서버에서 주문 금액 생성
  - 결제 완료 후 PortOne API 단건 조회로 실제 결제 금액과 주문 금액 대조
  - 결제 금액 위·변조 방지 구조 구성
- 사업자등록증 OCR을 Tesseract에서 Gemini Vision 기반 구조로 교체
  - PDFBox로 PDF를 이미지로 변환 후 Gemini Vision 호출
  - JSON 스키마 기반 구조화 출력으로 필드 추출 안정화
  - 후처리 정규식과 Tesseract 의존 제거
- SSE 기반 인증 상태 실시간 알림 구현
  - 인증 심사 단계 변경 시 클라이언트에 즉시 푸시
  - 폴링 대비 불필요한 요청과 상태 확인 지연 감소

Keystroke Auth — 타이핑 리듬 기반 AI 연속 인증 서버

2025.10 - 2025.10

- 역할: ML 모델, 인증 API, OTP 단독 개발
- 기술: Python 3.12, FastAPI, scikit-learn, IsolationForest, SQLAlchemy, SQLite, Pydantic, aiosmtplib
- 링크: https://github.com/whgusxo2055/keystroke_auth

사용자의 타이핑 리듬을 학습해 로그인 시 입력 패턴 이상 여부를 탐지하고, 이상으로 판단되면 OTP 추가 인증을 수행하는 연속 인증 서버입니다.
비밀번호와 키스트로크를 결합한 이중 인증 구조로, 타이핑 리듬 기반 AI 연속 인증 아이디어로 2025 캡스톤 디자인 및 AI 해커톤 경진대회 대상을 수상했습니다.

주요 구현 내용

- Dwell Time(키 누름 시간)과 Flight Time(키 간 이동 시간)에서 각각 평균·표준편차·중앙값을 추출해 6차원 고정 특징 벡터 구성, 단일/다중 행 입력을 모두 처리하도록 검증 분리
- 사용자별 IsolationForest 모델(contamination=0.05)을 학습·저장해 인증 모델 격리, 최소 10개 샘플 이상에서 학습하고 추가 로그인 데이터로 점진적 재학습(update) 지원
- decision_function, score_samples, 학습 데이터와의 최소 거리(distance)를 조합한 다중 신뢰도 판정으로 단일 모델 판정의 오탐을 완화
- 예측 결과를 grant_access(정상) / require_otp_verification(이상)으로 분기해, 이상 탐지 시 이메일 OTP 인증으로 폴백 → 모델 오탐이 곧 로그인 차단으로 이어지지 않도록 설계

모델 품질 평가 체계

- 학습 직후 모델을 5개 지표로 자동 평가: inlier_ratio(정상 판정 비율, 40%) + avg_confidence(평균 신뢰도, 40%) + consistency_score(일관성, 20%)를 가중합해 0~100점 quality_score 산출
- 70점 이상에서만 인증 모델을 활성화(is_acceptable)하고, 미달 시 추가 샘플 수집을 요청하는 운영 기준 수립
- 학습 데이터 자가 평가의 과적합 한계를 인지하고 별도 검증 데이터 사용을 권장 사항으로 문서화

Sepolia Gacha NFT — 확률형 NFT 가챠 dApp

2026.05 - 2026.06

- 역할: 스마트 컨트랙트, 배포 스크립트, 프론트엔드 단독 개발
- 기술: Solidity 0.8.20, OpenZeppelin, Chainlink VRF v2.5, Hardhat, TypeScript, TypeChain, Next.js 14, ethers.js v6, IPFS/Pinata, Ethereum Sepolia
- 링크: https://github.com/whgusxo2055/SepoliaNFT_Gacha

Chainlink VRF v2.5의 검증 가능한 온체인 난수로 NFT 등급(Normal 70% / Rare 20% / Epic 8% / Legendary 2%)을 결정하는 확률형 가챠 dApp입니다.

주요 구현 내용

- pull() → fulfillRandomWords() → claim() 3단 상태 머신(NoRequest → Requested → Fulfilled → Claimed)으로 결제·난수 도착·민팅을 분리
- VRF fulfillment 순서가 요청 순서와 다를 수 있어 requestId 기준으로 요청·결과를 매핑하고, 결제 시점의 user·seasonId를 저장

- 등급별 임계값(randomWord % 10000 → 7000/9000/9800 경계)으로 rarity 결정
- 시즌 기반 asset pool 구조로 등급별 metadata URI 관리

보안 설계

- 콜백 revert 방지: fulfillRandomWords() 콜백에서는 결과 저장·이벤트 emit만 수행하고, 가스 비용이 큰 mint는 `claim()`으로 분리
- 재진입·중복 claim 방지: claim()에 ReentrancyGuard 적용 + claimed 상태를 먼저 갱신한 뒤 mint(Checks-Effects-Interactions)
- 난수 무결성: block.timestamp / blockhash 기반 의사난수 금지, Chainlink VRF만 사용
- 권한·불변성: owner만 asset 등록·시즌 변경·pause 가능하되 확률 변경 함수 자체를 두지 않아 등급 확률을 배포 후 변경 불가하게 고정
- 등급 경계 단위 테스트 등 15개 컨트랙트 유닛 테스트로 확률 로직·revert 케이스 검증, Etherscan 소스 검증 완료

Paper Radar - Elasticsearch 논문·기술 트렌드 레이더

2025.12 - 2025.12

- 역할: 단독 개발
- 기술: Spring MVC, Thymeleaf, Elasticsearch, Docker Compose
- 링크: https://github.com/whgusxo2055/Paper_Radar

OpenAlex 등 공개 학술 메타데이터 API에서 논문 정보를 주기적으로 수집하고, Elasticsearch 검색·집계로 키워드·기관 중심의 연구·기술 트렌드를 탐색하는 레이더입니다

주요 구현 내용

- 수집 파이프라인: Spring Scheduler(@Scheduled)로 하루 1회 OpenAlex API에서 최근 3년 논문 메타데이터를 수집, 수동 실행·수집 상태/로그 확인 관리 기능 제공
- 검색·자동완성: 제목·초록·키워드·저자·기관 전문 검색에 기간/기관/키워드/저자 필터와 관련도·최신·인용수 정렬을 결합, 키워드·기관·저자 Ajax 자동완성(suggest) API 구현
- 이동평균 기반 트렌드 분석: 일 단위 집계 위에 MA7/MA30 이동평균을 산출하고 trend_score = (MA7 − MA30) / max(MA30, 1)로 키워드·기관 Top N 트렌드 도출
- 인덱스 설계: works / institutions / keyword_configs / ingest_jobs 인덱스로 검색·집계·운영 데이터를 분리, 키워드 자동 후보 제안 + 운영자 승인 흐름 구성
- 외부 원문 링크 정책: DOI → landing → PDF → OA 순 우선순위로 가능한 링크만 유형 배지와 함께 노출
- 무중단 운영: reindex + alias 스위칭으로 무중단 매핑 변경, best_link_url/type 재계산 백필을 maintenance 흐름으로 분리해 docs/ops.md에 문서화

성능 목표 : 검색 응답 p95 ≤ 1초, 트렌드 집계 p95 ≤ 2초, 단일 ES 노드 장애 허용

OTT_QUIC - C 기반 QUIC OTT 스트리밍 서버

2025.11 - 2025.12

- 역할: QUIC 엔진, 서버, DB, 웹 단독 개발
- 기술: C11, OpenSSL, QUIC/HTTP3, WebSocket, SQLite, Multi-threading, Docker, nginx, Makefile
- 링크: https://github.com/whgusxo2055/OTT_QUIC

QUIC 전송 계층을 외부 라이브러리 없이 C로 직접 구현해 네트워크 프로그래밍 핵심 구조를 학습한 OTT 스트리밍 서버입니다.
브라우저의 네이티브 QUIC을 쓰지 않고 UDP 위에 QUIC 연결·스트림·암호화를 직접 올렸습니다.

주요 구현 내용

- QUIC + TLS 1.3 엔진 직접 구현: OpenSSL 3.2+ 연동으로 TLS 핸드셰이크를 CRYPTO 프레임으로 송수신하고, key export 메커니즘으로 1-RTT 패킷 보호 키를 도출해 스트리밍 데이터 암·복호화
- 스트림/패킷 관리: connection_id 기반 연결 관리를 락으로 동시성 보호, 영상을 1MB 청크로 분할해 QUIC 전송하고 seek(시킹) 처리
- 포트 분리 보안 모델: 외부는 TCP 8443(HTTPS/WSS, nginx TLS 종료)만 공개하고, 내부 백엔드 HTTP(8080)와 QUIC(UDP 9443)는 비공개로 격리. 평문 HTTP는 차단/리다이렉트
- 인증·세션: bcrypt로 비밀번호 해시 저장(평문 저장 방지), 세션 토큰 발급 후 30분 만료, role 기반 admin 권한 분리(관리자 API는 관리자 세션 필수)
- 데이터/기능: SQLite 스키마(users / videos / watch_history / sessions) 기반 CRUD, 영상 업로드·썸네일·세그먼트 저장, 10초마다 시청 위치를 저장하는 이어보기
- Docker(Ubuntu 22.04, OpenSSL 소스 빌드) 기반 실행 환경 구성, QUIC/DB/WebSocket 단위 테스트 작성

인트인 직무체험형 인턴

2025.08 - 2025.08

Spring Boot기반 백엔드 실습을 통한 리스크, 데이터, 품질관리 역량 강화

- 현장실습 기간 동안 Spring Boot 기반 백엔드 프로젝트 회의와 개발 실습에 참여하며 요구사항 분석, 설계 방향 검토, 기술 선택 과정에서 발생할 수 있는 현장 리스크를 파악했습니다.
- 프로젝트 초기 단계에서 발생 가능한 일정 지연, 기술적 문제, 개발 범위 변경 등의 리스크를 논의하며 사전 식별과 관리의 중요성을 이해했습니다.
- 백엔드 개발 환경에서 MySQL과 MariaDB 기반 데이터베이스 연동 및 최적화 작업을 학습하며 데이터의 효율적인 저장과 조회 방식을 실습했습니다.
- 데이터 무결성, 트랜잭션 관리, 데이터베이스 연동 구조 등 데이터 관리의 핵심 요소를 이해하고 실제 개발 과정에 적용하는 경험을 쌓았습니다.
- 테스트 코드 작성, 코드 리뷰, QA 지침 교육에 참여하며 소프트웨어 품질 보증 프로세스와 코드 품질 관리 방식에 대한 실무 역량을 강화했습니다.
- 이를 통해 버그 예방, 테스트 자동화, 코드 리뷰 기반 품질 향상의 중요성을 이해하고 실제 프로젝트 흐름 속에서 개발 품질을 관리하는 방법을 익혔습니다.
- 관련기술: Spring Boot, Java, MySQL, MariaDB, Database, Transaction, QA, Test Code, Code Review, Backend Development, Risk Management

성과
- 개발 초기 단계에서 발생할 수 있는 일정 지연과 기술적 문제를 사전에 인지하고 관리하는 리스크 분석 역량을 강화했습니다.
- MySQL, MariaDB 기반 데이터베이스 연동 및 최적화 실습을 통해 데이터 저장, 조회, 무결성, 트랜잭션 관리 역량을 습득했습니다.
- 테스트 코드 작성, 코드 리뷰, QA 지침 교육을 통해 소프트웨어 품질 보증과 코드 품질 관리 역량을 강화했습니다.

인사이너리 직무체험형 인턴

2026.01 - 2026.03

ClaritySC(Clarity Supply Chain) 품질 보증(QA)

- 현장실습 기간 동안 인사이너리의 오픈소스 보안 및 라이선스 관리 솔루션인 ClaritySC(Clarity Supply Chain)의 품질 보증(QA) 업무를 주도적으로 수행했습니다.

- 실습 초기에는 제품군 및 Clarity, FossID 솔루션에 대한 교육을 이수하고 사용 가이드를 정독하며 필요한 도메인 지식을 빠르게 습득했습니다.
- 이후 할당받은 구역인 테스트 케이스를 직접 설계하고, 이를 바탕으로 시스템의 각 기능이 요구사항에 맞게 정상적으로 동작하는지 꼼꼼하게 검증했습니다.
- 정해진 시나리오에만 의존하지 않고, 실제 사용자 환경을 가정한 탐색적 테스팅(Exploratory Testing)을 병행하여 숨겨진 결함과 다국어 지원 과정에서의 영문/한글 번역 오류를 지속적으로 추적했습니다.
- 또한, 매주 부서 주간 회의에 참석하여 진행 상황과 이슈를 공유하고, 팀원들이 작성한 테스트 케이스와 상호 교차 검증(Cross-Check)을 진행하며 프로젝트의 QA 과정을 지원했습니다.
- 관련기술: ClaritySC, Clarity, FossID, QA, Exploratory Testing, Cross-Check, DevOps

성과
- 품질 보증(QA) 업무를 주도적으로 수행했습니다.
- 필요한 도메인 지식을 빠르게 습득했습니다.
- 숨겨진 결함과 다국어 지원 과정에서의 영문/한글 번역 오류를 지속적으로 추적했습니다.

학력

호서대학교
2021.03 - | 재학 중 | 컴퓨터공학부

- 컴퓨터공학부 재학 중 전공 평점 4.20/4.50, 전체 평점 4.03/4.50 달성
- 자료구조, 알고리즘, 운영체제, 데이터베이스, 네트워크 등 컴퓨터공학 핵심 전공 과목 이수
- 디지털융합연구실 학부연구생으로 백엔드 및 분산 시스템 분야 연구 수행
- SW중심대학사업단 산학협력 프로젝트로 실제 전공 수업에 활용 중인 코딩 교육 플랫폼 CodeQuest 개발

스킬

Java | Spring Boot | Redis | MySQL | Kubernetes | Restful API | Web Socket | Docker | GitHub Actions | Git | PostgreSQL | ElasticSearch | JUnit | Python | C | Solidity | AWS | Linux | TypeScript | SQLite | JPA

수상/자격증/기타

2025 캡스톤 디자인 및 AI 해커톤 경진대회 대상
2025.10 | 수상

한국컴퓨터교육학회 주관 2025 캡스톤 디자인 및 AI 해커톤 경진대회 대상 수상

2026 춘계종합학술대회 학생우수논문상

2026.05 | 수상

정보통신학회 주관 2026 춘계종합학술대회
"저지연 실시간성을 보장하는 코딩 테스트 플랫폼 설계"
제1저자로 학생 우수 논문상 수상
관련 내용: Kafka Consumer Lag 기반 KEDA 채점 워커 오토스케일링 설계

AI융합프리젠테이션 경진대회 AI공과대학장상

2025.10 | 수상

호서대 AI공과대학장상

TOPCIT(소프트웨어 역량 검정시험)

2026.05 | 기타

429점

링크

포트폴리오 사이트 링크
https://hyeontae.dev/

깃허브 프로필
https://github.com/whgusxo2055