

장수빈

+82 10-8620-6124 · chickby@gmail.com

핵심 요약 (Professional Summary)

예약·결제·POS·KIOSK 등 상태·캐시·동기화 복잡도가 높은 도메인을 5년간 다뤄온 프론트엔드 개발자로, React·TypeScript·React Query 기반의 상태/캐시 설계와 다중 윈도우 동기화·SSR/CSR 하이브리드·옵션 그룹 상품 모델링 같은 난도 높은 문제를 직접 풀어왔습니다.

전국 500개 단지·MAU 10만 규모 서비스의 장기 운영과, 신설 조직에서 프론트엔드 표준·CI/CD·디자인 시스템 4종을 0→1로 정착시킨 경험을 함께 보유하여, 안정적 운영과 초기 구축 양쪽이 필요한 프론트엔드 직무에 즉시 기여할 수 있습니다.

기획·디자인과 한 테이블에서 모델을 다듬는 협업을 즐기며, 인프라부터 백엔드까지 도메인 전반에 관심을 두고 일합니다.

핵심 기술 스택

- Frontend : React, TypeScript, JavaScript, HTML, CSS
- State/Data : React Query, Redux, Recoil, Zod
- UI/Design : Tailwind CSS, Ant Design, MUI, Bootstrap, Storybook
- Test : Jest, MSW
- Infra/DevOps : AWS(S3, Amplify, CodeArtifact), Jenkins, GitHub Actions, Docker, Ansible
- Backend : Node.js, NestJS, PostgreSQL, MySQL
- AI Workflow : Claude Code, Cursor — 설계·리팩토링·리뷰 보조로 활용, 결과는 코드 리뷰 관점에서 검증·선별 적용

경력 (5년 1개월)

주식회사 알스솔루션

2025.03 - 재직중 (1년 8개월) | 정규직 | 개발 | 선임

신설 기업 합류 후 입사 3개월 만에 프론트엔드 표준을 0→1로 정착시켰으며, 현재 사내 5개 프로젝트의 개발 기반을 구축·운영 중.

프론트엔드 표준화 및 CI/CD 정착 (2025.03 - 재직중)

개요

신설 조직의 멀티 프로젝트 환경에서 개발 컨벤션·빌드·배포 흐름을 단일 표준으로 정착시킨 기반 작업.

주요 역할

- 공통 API·ESLint 설정을 사내 표준 npm 패키지로 분리하여 개발 컨벤션과 빌드 환경을 일원화
- 프로젝트 특성에 맞춰 디자인 시스템 4종을 별도 설계·운영 (범용 Monorepo → 백오피스 / Antd 기반 어드민 라인 → BI·사내 백오피스 리팩토링 / 팜브릿지 전용 SSR + React Island / Emotion 기반 → BI 데이터 전처리 백오피스)
- Jenkins 기반 CI/CD에 AWS CodeArtifact 인증 자동화를 연동하여 멀티 프로젝트 빌드·배포를 단일 파이프라인으로 표준화
- Jenkins 스크립트를 Git 저장소로 이관하고 환경 설정을 문서화

성과

- 입사 3개월 만에 프론트엔드 표준을 0→1로 정착
- 사내 전 프론트엔드 프로젝트의 개발 컨벤션·빌드 환경 일원화
- CI/CD 자동화 도입으로 신규 프로젝트 빌드·배포 소요 시간 약 60% 단축
- 신규 합류자 개발 환경 셋업 시간 약 1일 → 1시간 이내로 단축

기술 스택 Monorepo, Jenkins, AWS CodeArtifact

디자인 시스템 구축 (2025.03 - 2025.04)

개요

사내 멀티 프로젝트에서 공통으로 사용할 범용 디자인 시스템을 컴포넌트 단위 npm 패키지로 신규 구축.

주요 역할

- 백오피스 도메인 비중이 높은 사내 환경에 맞춰 컴포넌트 풍부도·커스터마이징 용이성·반응형 그리드 성숙도를 종합 검토해 Bootstrap 채택 의사결정
- 사내 디자인 토큰을 CSS Variables로 추상화하여 Bootstrap 기본 스타일 위에 사내 정체성 레이어를 일관 적용
- 색상·타이포·간격·레이아웃을 단일 토큰 시스템으로 정의
- 라이트/다크 테마를 토큰 레이어에서 분기하는 아키텍처 설계 (컴포넌트 코드는 테마 비인자로 분리)

성과

- 토큰 1회 변경으로 디자인 시스템 전반 갱신 가능한 구조 확보
- 컴포넌트를 Monorepo 단위 패키지로 분리하여 프로젝트 간 독립 배포·버전 관리 가능
- 공통 컴포넌트 재사용으로 신규 화면 UI 개발 시간 약 30% 단축
- 디자인 토큰 단일화로 화면 간 중복 스타일 코드 약 50% 감소

기술 스택 Bootstrap, CSS Variables, Monorepo

사내 내부 백오피스 신규 구축 (2025.05 - 2025.06)

개요

공지사항·약관·FAQ·회원 관리 등 사내 운영 도메인 전반을 단독으로 초기 설계부터 구현까지 담당한 운영 관리팀.

주요 역할

- 별도 운영팀 없이 개발자·기획자가 운영을 겸하는 신설 조직 초기 환경에 맞춰 단독 설계·구현
- 자체 Monorepo 디자인 시스템을 첫 적용하여 토큰·컴포넌트 구조를 운영 환경에서 검증
- 기능 단위 컴포넌트 분리로 메뉴 추가 시 기존 화면 영향 없이 확장 가능한 구조 설계
- AWS S3 파일 업로드 모듈 구현 — 게시글 작성 흐름과 첨부 파일 라이프사이클을 연동해 게시글 단위 일관 관리

성과

- 단독 개발로 약 6주 만에 운영 도메인 5종(공지·약관·FAQ·회원 등) 초기 버전 출시
- 자체 Monorepo 디자인 시스템을 최초 운영 검증, 사내 표준 라인업 정착의 기반 마련
- 컴포넌트 분리 구조로 이후 신규 메뉴 추가 시 평균 작업 시간 약 30% 단축
- 이후 2026.02-03 BI 사전 검증 목적으로 Antd 디자인 시스템 마이그레이션 진행 (하단 '사내 백오피스 리팩토링' 참조)

기술 스택 React, TypeScript, Monorepo Design System, AWS S3

팜브릿지 서비스 재구축 (2025.07 - 2025.12)

개요

약국 대상 약품 입고 관리 프로그램 '팜브릿지'의 백엔드 JSP→Spring 전환에 맞춘 프론트엔드 전면 재구성.

주요 역할

- 백엔드 JSP→Spring 전환 일정에 동기화해 정해진 기간 내 안정화해야 하는 제약이 가장 컸고, 여기에 SSR 라우팅 제약·운영 인력 장기 유지보수 가능 기술 폭을 함께 고려해 Vanilla JS 선택 의사결정 (React 도입 시 구조 설계·학습 비용으로 전환 일정 내 안정화가 어렵다고 판단)
- HTML/CSS/JavaScript만으로 라우팅 진입점·상태 컨테이너·이벤트 위임 모듈을 자체 구현하여 프레임워크 의존도 없는 유지보수 구조 설계
- SSR 환경에 최적화된 자체 디자인 시스템을 별도 설계 — 사내 Monorepo·Antd 라인과 토큰·컴포넌트 분리 운영

성과

- 프레임워크 의존성 0으로 운영 인력 장기 유지보수 가능한 구조 확보
- 자체 라우팅·상태 모듈화로 화면 추가 시 신규 코드량 약 20% 감소
- 백엔드 Spring 전환 일정에 동기화하여 안정화 완료
- 이후 2026.02-03 React Island 패턴으로 부분 전환 (하단 '팜브릿지 서비스 리팩토링' 참조)

기술 스택 HTML, CSS, JavaScript (Vanilla), 자체 디자인 시스템

WBS 시스템 구축 (2026.01 - 2026.02)

개요

Jira REST API 기반 데이터로 일정 관리 가시성을 높인 간트 차트·캘린더 뷰 시스템.

주요 역할

- Jira REST API 데이터로 간트 차트·캘린더 뷰 설계·구현
- 드래그앤드롭 일정 편집에 낙관적 업데이트 적용
- Changelog 분석으로 과거 시점 조회·고스트 바 비교 기능 구현 (회고/이력 분석 지원)
- Atlassian OAuth 연동 및 URL 상태 관리

성과

- 낙관적 업데이트 적용으로 일정 편집 체감 반응 시간 약 70% 단축
- 팀/상태 기반 색상 시각화·다중 필터링으로 일정 탐색 소요 시간 약 50% 단축
- URL 상태 관리로 별도 저장 없이 공유 가능한 협업 환경 구축, 일정 공유 커뮤니케이션 비용 절감

기술 스택 React, TypeScript, Atlassian OAuth, Jira REST API

팜브릿지 서비스 리팩토링 (2026.02 - 2026.03)

개요

Vanilla JS 화면의 유지보수 비용 누적 문제를, SSR 라우팅 제약을 지키면서 부분 React 도입으로 해소.

주요 역할

- 상태·이벤트 결합으로 유지보수 비용이 누적되고 SSR 라우팅은 SEO·서버 의존성으로 변경 불가한 제약 하에 React Island 패턴 채택 의사결정 (SSR이 페이지 쉘, 동적 영역만 React 마운트하는 SSR/CSR 하이브리드)
- React Query 전면 도입으로 API 호출·캐싱·에러 처리 일원화
- 화면 단위 로직을 도메인별 혹은 유틸로 분리하고 Context 도입으로 prop drilling 제거
- 테스트 인프라(Jest+MSW) 구축

성과

- API 호출·캐싱·에러 처리를 단일 레이어로 일원화
- 테스트 인프라(Jest+MSW) 구축으로 React 영역 테스트 커버리지 0% → 약 20% 확보
- 도메인 모듈 분리·React Query 도입으로 관련 화면 버그 리포트 약 50% 감소
- prop drilling 제거 및 도메인별 모듈 분리로 신규 기능 평균 작업 시간 약 50% 단축

기술 스택 React, TypeScript, React Query, React Island Pattern (SSR/CSR Hybrid)

사내 백오피스 리팩토링 (2026.02 - 2026.03)

개요

차기 BI 서비스의 Antd 디자인 시스템 본격 적용 전, 운영 중인 백오피스에서 마이그레이션을 선행해 리스크를 사전 검증.

주요 역할

- 신규 프로젝트 본격 적용 전 운영 백오피스에서 Antd 워크플로우·사내 정체성 디자인 적용 가능성·운영 리스크를 사전 검증
- 자체 Monorepo 디자인 시스템에서 Antd 기반으로 컴포넌트 단위 점진적 교체
- 기존 디자인 토큰을 Antd props·테마로 매핑하는 변환 가이드 정의

성과

- 약 4주간 컴포넌트 약 40종을 Antd 기반으로 무중단 점진 전환, 화면별 회귀 결함 0건
- 변환 가이드 정의로 화면별 영향 범위 최소화 — 화면당 평균 수정 범위 약 50% 축소
- BI 신규 프로젝트 적용 전 운영 리스크 사전 검증 완료, 후속 프로젝트 초기 개발 속도 확보 기반 마련

기술 스택 React, TypeScript, Ant Design

BI 서비스 신규 구축 (2026.04 - 2026.08)

개요

약품 판매 실적 데이터를 제약사·사내 운영자에게 시각화·분석 형태로 제공하는 사내 BI 서비스. 제약사용 서비스 앱과 사내 어드민 콘솔 두 프로덕트를 pnpm 모노레포로 구축.

주요 역할

- 두 프로덕트 모노레포 아키텍처 설계 — 인증·차트·도메인 레이어는 공유 패키지로, 앱 셀(라우터·가드)은 각 앱이 독립 보유해 교차 의존 제거
- 서비스 앱(대시보드 / 통합분석 5축 : 출고·채널·지역·시장·제품 / 데이터 활용 / 계정관리 / 고객센터), 어드민 앱(계약사·계정 관리 / 운영 콘텐츠 / 데이터 전처리 / 보안·접근 로그) 구현
- 직전 백오피스 리팩토링에서 사전 검증한 Antd 디자인 시스템 본격 적용. 차트도 라이브러리 직접 사용을 막고 공유 래퍼 6종만 공개해 디자인 시스템에 편입
- 인증 계약 설계 — 인증 원본은 서버 HttpOnly 쿠키, 클라이언트는 표시용 캐시로 한정. 401/403만 로그아웃하고 네트워크 오류·5xx는 재시도 상태 유지
- 역할 4종(알스마스터·알스어드민·계약사마스터·계약사일반) RBAC 삼중 방어 — 버튼·메뉴 미렌더 + 라우트 가드 + 페이지 가드의 조건을 일치시켜 "보이는데 누르면 튕기는" 불일치 제거. eslint 경계 규칙으로 단방향 import·feature 간 참조·SDK 직접 호출 금지 강제
- CS 고객 화면 보기 설계 — 어드민이 계약사 화면을 그대로 보는 세션을 창 간 postMessage로 인계. 양쪽 origin 명시(* 금지)·수신 측 origin/source 검증, 토큰은 메모리 보관, CS 탭은 분석 스크립트 미로드·쿠키 세션 복원 생략·읽기 전용으로 격리. 세션 종료를 일반 로그아웃과 분리해 같은 브라우저의 본인 세션 보존
- 분석 도구(GA4·Microsoft Clarity) 도입 시 계약사 데이터 유출 경계 설계 — 공통 클라이언트에서 PII 파라미터·URL 제거, 신원 설정/해제를 인증 스토어 구독 한 곳으로 통일, Clarity 마스킹은 Antd portal 때문에 대시보드 설정이 아닌 <body> 속성으로 코드에서 강제
- SSE 실시간 알림센터 — 보정 반영 알림을 스트림으로 수신해 토스트·데스크톱 알림·인박스를 공유 패키지 모듈로 통합

성과

- 혼합 차트에서 G2 DualAxes가 자식별 y 스케일을 독립 강제해 라인이 소실되는 문제를, 두 마크가 스케일을 공유하도록 차트 뷰를 직접 조립한 컴포넌트로 해결 — 라이브러리 직접 사용을 막아둔 덕분에 대응이 공유 패키지 안에서 끝남
- React Query AbortSignal을 조회 함수 65개 전 구간에 관통 전달해 화면 이탈·필터 재조회 시 실제 HTTP 요청 취소, 누락 검사 테스트로 회귀 차단
- 라우트 코드 스플리팅에서 React.lazy가 실패를 확정 저장해 복구되지 않는 문제를 세대 기반 재시도 래퍼로 해소
- 테스트 233개 파일·1,389개 케이스(Vitest) + 역할별 인증 E2E 31개(Playwright, 8 spec). 리렌더 횟수가 계약인 화면은 렌더 카운팅으로 memo 회귀 검증
- 정적 산출물에 API 호스트를 굿지 않아 dev·prod 동일 이미지 공유, 운영 PodSecurity 정책에 맞춰 비루트 nginx 이미지로 전환

기술 스택 React 19, TypeScript, Vite, Ant Design, AntV G2, React Query, Zustand, pnpm Monorepo, Vitest, Playwright

BI 데이터 전처리 백오피스 구축 (2026.09 - 2026.10)

개요

BI 서비스에 공개되는 약품 판매 데이터의 전처리를 담당하는 사내 백오피스. 데이터비즈팀이 일일 Excel 리포트(15시트)로 반려 데이터를 검토해 회신하고 담당자가 DB에 수기 입력하던 과정을 웹 검수·보정 접수로 대체.

주요 역할

- 전처리 업무를 단계로 분리 — 반려 검수 → 초안 저장 → 서버 미리보기 → 접수 → 보정 배치 → 고객 공개. "저장됐다 / 실행됐다 / 공개됐다"를 화면에서 구분해 되돌릴 수 없는 전이를 명시
- 접수 중복 방지 — 미리보기 ID + payload 해시 + 범위별 멍든기. "결과를 모르는 실패"(5xx·무응답)는 거절 실패와 분리해 재시도 대신 이력 확인으로 유도
- 빈 값과 0 구분 — 확인 불가·준비 안 됨·기록 없음·미수신을 각각 다른 표현으로 고정, 금액은 서버의 10진 문자열을 숫자 변환 없이 표시
- 자체 디자인 시스템 패키지(컴포넌트 36종) 구축 — Emotion 기반, 테마 4종과 강조색 명암비 4.5:1 자동 보정
- 범용 셀 구조 — 제품을 설정 객체로 등록하면 메뉴·기본 화면·브레드크럼이 파생. 레지스트리 누락은 컴파일 오류로 차단

성과

- 디자인 토큰·접근성·문구 등 규약 17종을 실패하는 시험으로 전환. 문구 시험은 코드 라벨 표와 용어집을 대조해 화면·문서 불일치를 차단

- 테스트 252개 파일·약 1,766개 케이스(node:test) + e2e 6 spec(Playwright). 포맷·빌드·lint·타입·테스트·번들 검사를 한 명령으로 묶음
- 미검증 범위 명시 — 실제 직원 SSO 세션은 개발 origin 콜백 미허용으로 확인 못 했고, 서버 권한·감사·역등성과 배치 실행은 실환경 미검증

기술 스택 React 19, TypeScript, Vite, Emotion, React Query, 자체 디자인 시스템, pnpm Monorepo, node:test, Playwright

기타 프로젝트

- PWA/Electron 기반 알림 환경 구축 : FCM Push 서버 구축, 환경별 권한 모델 차이를 단일 인터페이스로 추상화
- AI 도구 활용 워크플로우 정립 : Claude Code/Cursor를 설계·리팩토링·리뷰 보조로 도입, 산출물은 코드 리뷰 관점에서 검증·선별 적용 — 반복 작업 평균 처리 시간 약 30% 단축

주식회사 에이치티비온드

2021.08 - 2024.12 (3년 5개월) | 정규직 | 개발

전국 500개 단지·MAU 10만 규모의 아파트 커뮤니티 플랫폼 프론트엔드 개발 및 장기 운영. 예약·결제·이용권·POS·KIOSK 등 도메인 복잡도가 높은 서비스를 다년간 운영하며 기능 개발과 아키텍처 개선·기술 부채 해소를 병행 수행.

커뮤니티 서비스 운영 / 파트너스 플랫폼 (2021.08 - 2024.12)

개요

5종 예약 도메인과 PG·관리비 결제, 이용권 관리가 커뮤니티·파트너스 두 프로젝트에 걸쳐 운영되는 환경에서 기능 개발과 아키텍처 개선을 병행 수행.

주요 역할

- 파트너스 핵심 메뉴 4개 영역에서 JS→TS 마이그레이션 단계적 진행 (도메인 결합이 강한 부분은 인터페이스 우선 정의 후 구현체 이관)
- 커뮤니티 서버 상태를 Redux→React Query로 분리 (캐시 정책과 클라이언트 상태가 한 store에 섞여 stale 데이터 노출 이슈 누적이 배경). 도메인별 queryKey 컨벤션·invalidation 규칙 문서화 후 팀 전파
- Axios interceptor 공통 에러 처리로 외부 연동(PG/관리비) 실패 메시지를 도메인 단위로 일관화
- 팀 내 첫 프론트엔드 테스트 체계(Jest+MSW) 도입 주도
- 기획·디자인 협업 : 예약 창구 충돌 검증 시 충돌 자원 목록을 함께 노출하는 경고 모달을 역제안·반영 / 예약 시간 설정 화면을 정보 그룹 박스·배치 재구성으로 시각적 구분 강화하는 디자인 역제안·반영

성과

- 운영 중단 없이 핵심 4개 영역 JS→TS 마이그레이션 완료
- 테스트 인프라(Jest+MSW) 구축 후 핵심 결제·예약 플로우 적용으로 배포 후 핫픽스 월 평균 5건 → 1건으로 약 80% 감소
- JS→TS 전환으로 타입 관련 런타임 에러 약 70% 감소
- Redux→React Query 분리로 stale 데이터 노출 이슈 약 90% 감소
- 충돌 자원 노출 모달 도입으로 운영팀이 충돌 자원을 즉시 식별·재배정 가능, 창구 생성 포기 사례 약 90% 감소
- 예약 시간 설정 화면 재구성으로 고객사 설정 작업 진입 장벽 완화

기술 스택 React, TypeScript, React Query, Redux, Recoil, Axios, Jest, MSW, Ant Design, MUI

POS / 주문 키오스크 시스템 (2023.03 - 2023.04)

개요

아파트 단지 내 입주민 대상 카페·매점에서 사용하는 클라우드 기반 POS·주문 키오스크 시스템 신규 개발.

주요 역할

- React Query·Recoil로 서버 상태와 UI 상태를 분리하여 컴포넌트 간 의존성 감소
- AbortController로 결제 API를 요청 단위 취소 가능하게 구성 (결제 도중 화면 이탈 시 잔여 요청이 후속 화면에 영향 주지 않도록 처리)
- POS(가로 1024×768)·KIOSK(세로 1024×1920) 두 화면 방향·비율을 Tailwind 토큰으로 추상화하여 동일 컴포넌트로 양 환경 대응
- 트러블슈팅 : 한 매장에 메인(직원)+서브(고객) 두 대의 독립 크롬 윈도우가 결제 상태·주문 내역·세션을 실시간 공유해야 하는 요구를, WebSocket 서버 신설 없이 Broadcast Channel API를 1차 채널 + localStorage event를 백업 채널로 두는 이중 구조로 설계. 수신 측이

React Query invalidate를 호출해 두 화면 동기화

성과

- WebSocket 서버 신설 없이 다중 윈도우 실시간 동기화 구현 (팀 내 윈도우 간 통신 선례 없는 상황)
- 이중 채널 동기화로 두 윈도우 간 상태 실시간 반영으로 안정성 확보
- 동일 컴포넌트로 POS·KIOSK 2개 환경 동시 대응, 화면 대응 코드 중복 약 50% 절감

기술 스택 React, TypeScript, React Query, Recoil, Tailwind CSS, Broadcast Channel API, localStorage

바이비 2.0 서비스 (2023.02 - 2023.10)

개요

예약 서비스 UI/UX 개선 및 월 반복 강좌 기능 구현.

주요 역할

- 예약 서비스 UI/UX 개선 및 월 반복 강좌 기능 구현
- 옵션 그룹 기반 상품 구조 설계
- Lexical Editor 커스터마이징으로 게시글 편집 기능 고도화
- 전역 Modal/Bottom Sheet 구현으로 모바일 네비게이션·백버튼 대응
- Storybook 기반 디자인 시스템 구축 및 공통 컴포넌트 개발

성과

- 옵션 그룹 기반 상품 구조로 다양한 상품 조합 지원
- Storybook 디자인 시스템 구축으로 공통 컴포넌트 재사용 기반 마련
- 전역 Modal/Bottom Sheet 도입으로 모바일 네비게이션 관련 버그 약 60% 감소

기술 스택 React, TypeScript, React Query, Recoil, Tailwind CSS, Storybook, Lexical

프론트엔드 인프라 및 배포 환경 개선 (2024.02 - 2024.04)

개요

3개 프론트 프로젝트의 WebServer 이관 및 단일 배포 파이프라인 구축으로 배포 흐름 표준화.

주요 역할

- blue-byb-pos-blue-byb-kiosk-blue-web-wallpad 3개 프로젝트의 WebServer를 pm2 serve → nginx 정적 콘텐츠 서비스로 이관 (build 파일 교체만으로 배포 완결되도록 구조 단순화)
- Ansible Semaphore + Jenkins + AWS S3를 잇는 단일 배포 파이프라인 구축 (GITHUB_BUNDLE_CHECK → JENKINS_BUILD → DEPLOY_TO_EDGE 3단계 yml)
- POS·KIOSK 빌드 통합 : REACT_APP_PRODUCT 환경변수 sed 치환으로 한 번의 build step에서 두 프로젝트 산출물 동시 생성
- AWS S3 빌드 산출물을 Git Tag version 단위로 관리 (build_id 기반 → version_tag 기반)
- 트러블슈팅 : 슬래시 포함 tag와 s3 폴더명 미스매치 이슈를 build parameter에 version_tag·s3_bucket_name 분리 도입으로 해결
- nginx 정적 콘텐츠 실행을 위한 build 폴더 권한 자동화 (chmod 755 + /home/htbeyond 711 최소 권한)

성과

- POS·KIOSK 빌드 통합으로 2회 빌드 → 1회로 단축
- nginx 전환·파이프라인 단일화로 배포 1회 소요 시간 약 90% 단축
- Git Tag 기반 버전 관리로 단지별 배포 버전 명시적 추적 가능, 배포 추적·롤백 대응 시간 약 60% 단축
- 환경별(dev/alpha/prod) 배포 흐름 일관화

기술 스택 Jenkins, Ansible Semaphore, AWS S3, nginx, Git Tag, YAML, Linux

인하대학교

2016.02 - 2020.02 | 졸업 | 정보통신공학 학사

알고리즘, 데이터베이스, 운영체제, 시스템프로그래밍

수상/자격증/기타

학생회 관리 홈페이지 (2018.01) | 기타

개요 학생회 운영을 위한 내부 관리 웹 애플리케이션 개발.

주요 역할 대여 사업 물품 관리 기능 구현 / 학과 정보 및 공지 사항 관리 화면 개발

성과 학생회 운영 도메인 전반을 단일 웹 애플리케이션으로 통합, 물품·공지 관리 업무 처리 시간 약 80% 단축

기술 스택 Node.js, Vue.js, MySQL

프로그래밍 경진대회 (2018.09) | 수상

개요 학생 자치기구 선거를 위한 웹 기반 전자 투표 시스템 개발.

주요 역할 전자 투표 기능 및 투표 이력 조회 기능 구현 / 데이터 무결성을 고려한 투표 기록 관리

성과 데이터 무결성 기반 투표 시스템으로 교내 경진대회 2위 수상

기술 스택 Node.js, Vue.js, MySQL

놀러와마이홈 게임 계산기 홈페이지 (2020.08) | 기타

개요 게임 「놀러와마이홈」의 재화·재료 계산을 돕는 웹 서비스 개발.

주요 역할 프론트엔드·백엔드 전반 단독 설계·구현 / 사용자 입력 기반 계산 로직 및 결과 시각화 / 배포 및 프로세스 관리

성과 실제 게임 유저 대상 서비스 운영, 누적 사용자 약 3.3천 명·재방문율 약 30% 확보

기술 스택 React, Node.js, PostgreSQL, Docker, PM2, Google Analytics

바이비 Quiz (2022.12) | 기타

개요 NestJS·PostgreSQL 기반 실시간 퀴즈 서비스 백엔드 개발.

주요 역할 실시간 퀴즈 백엔드 개발 / 문제 진행 상태 및 점수 집계 로직 구현

성과 실시간 진행 상태·점수 집계 로직 구현 완료, 동시 참여 약 50명 안정 처리

기술 스택 NestJS, PostgreSQL

링크

- GitHub : <https://github.com/shuobyan>
- 포트폴리오 : <https://portfolio-rho-eight-97.vercel.app/>