Frontend Developer Portfolio

최준혁

최준혁

raonrabbit@gmail.com

010-4000-4887

Github https://github.com/raonrabbit

“Next.js · TypeScript 기반으로 운영 환경의 UI 구조 설계와 성능 개선을 경험한 프론트엔드 개발자 최준혁입니다.”

학력

2018-03 ~ 2024-02 안양대학교 컴퓨터공학 전공 3.89 / 4.5 학점

활동 및 교육

2025-11 ~ 2025-12 대한수중·핀수영협회 외주

2024-07 ~ 2025-06 삼성 청년 SW · AI 아카데미 수료

어학

2025-09-05 OPIC IM1

자격증

2023-11-15 정보처리기사

수상

2025-05 SSAFY 2학기 프로젝트 우수상(2등)

2025-04 SSAFY 2학기 프로젝트 우수상(1등)

2024-11 SSAFY 1학기 프로젝트 최우수상(1등)

기술 스택

JavaScript

- **ES6+ 문법** 에 대한 이해와 활용 가능
- async/await, Promise 등 비동기 통신 및 데이터 처리 이해

TypeScript

- 타입, 인터페이스, 제네릭 등에 대한 이해
- 런타임 에러를 줄이고 코드 안정성을 높이기 위해 프로젝트에 적극적으로 활용

React

- 컴포넌트 설계·재사용 및 상태 관리 가능
- **커스텀 훅 설계**로 로직 분리·재사용 경험
- Tanstack Query 등 다양한 라이브러리 활용 경험

Next.js

- Next.js 16(App Router) 기반의 상용 서비스 아키텍처 설계 및 배포 경험
- Sever/Client Component 에 대한 이해 및 설계 경험

AI Agent

- 프로젝트 구조, 컨벤션을 AI에 주입해 일관된 코드 생성 환경 구축 경험
- AI 를 설계 도구로 활용한 4주 내 실 서비스 배포 경험

Git & JIRA

- **Git Flow 전략**을 활용한 협업 경험
- JIRA 를 활용한 이슈 및 스프린트 관리 경험

Figma

- Auto Layout 으로 반응형 레이아웃 설계 가능
- 와이어프레임, 목업 제작 경험

PJT 1. 대한수중·핀수영협회

대한수중·핀수영협회 외주 개발

참고자료 • http://kua.or.kr/

> 기존 홈페이지의 노후화된 UI/UX와 비효율적인 관리 시스템을 개선하기 위해 Next.js 기반으로 전면 재개발했습니다. 공식 협회 사이트로 현재 운영 중입니다.

성과

- 실사용자를 대상으로 하는 웹페이지 개발 및 운영

개발 기간

- 2025.11 ~ 2025.12 (5주, 이후 유지보수)

사용 기술

Next.js · TypeScript · Tailwind CSS · Tanstack Query

개발 인원

Web Designer	Backend	Frontend
		* 담당 파트

주요 기능

- 협회 소개 · 종목 소개 · 대회정보 페이지
- 게시판 시스템 (공지사항, 갤러리, 대회일정 등 22개)
- 관리자 대시보드 (팝업 · 회원 · 소개 페이지 콘텐츠 관리)

담당 역할

- 공통 컴포넌트 제작 및 구조 설계
- 메인 페이지, 대회정보 페이지, 커뮤니티 페이지, 종목 소개 페이지 등 구현
- CI/CD 파이프라인 구축 (Github Actions + PM2)

운영형 게시판 공통화 설계로 22개 게시판 **일관된 구조 유지** 및 **신규 추가 비용 최소화**

Why?

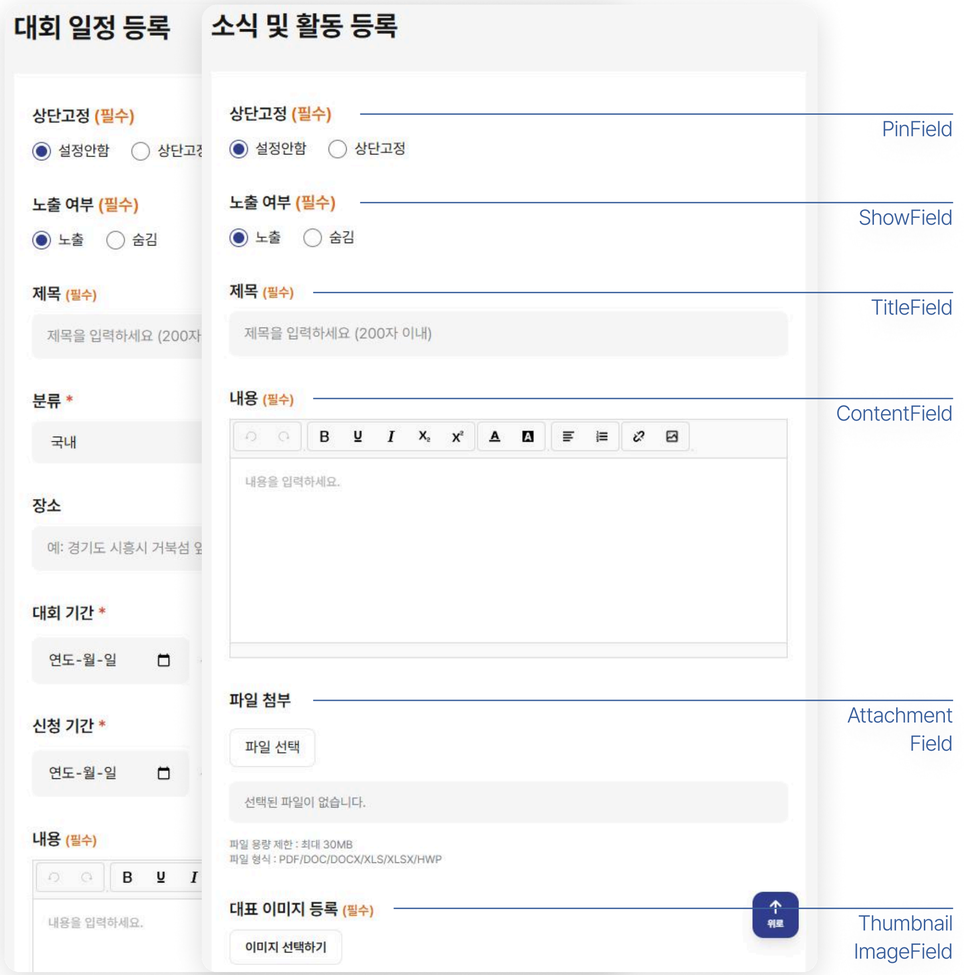
- 디자인 미확정 · 요구사항 변경이 잦은 환경에서 마감 기한은 타이트하여
  변경에 강한 구조 설계가 필수적인 상황

How

- **Compound Component 패턴**으로 PostForm 설계 - 10개 서브 컴포넌트를 조합해 게시판마다 필요한 필드만 구성
- **Generic 기반 DataTable 설계** - DataTable<T> 형태의 제네릭 타입으로 다양한 도메인 데이터를 단일 컴포넌트로 렌더링, 도메인마다 컬럼 정의만 주입하는 구조로 분리
- **JSDoc 기반 문서화** - JSDoc 기반 인터페이스 문서화로 서브 컴포넌트 조합 방식과 props 계약을 명시, 팀 개발 시 불필요한 구현 문의 최소화

Result

- 신규 게시판 추가 시 서브 컴포넌트 조합과 컬럼 정의만으로 구현 가능
- 22개 게시판 전반에 일관된 UX 유지, 운영 요구사항 변경에 **빠르게 대응 가능한 구조** 확보



운영 서비스 **보안 취약점**(XSS · SSRF · CSRF) 자체 탐지 및 대응

Why?

- dangerouslySetInnerHTML 렌더링 구조에서 XSS,
  URL 검증 없이 fetch 하는 파일 프록시 구조에서 SSRF,
  httpOnly 쿠키 기반 JWT 구조에서 CSRF 등 **보안 취약점 확인**

How

- **XSS** : **dompurify** 도입 - 렌더링 전 sanitizeHtml() 로 악성 스크립트 제거
- **SSRF** : **비 HTTP(S) 프로토콜 차단 + 허용 호스트 화이트리스트**로 서버 fetch 제한
- **CSRF** : 쿠키 직통 인증 대신 Next.js Route Handler를 API 프록시로 활용해 JWT를 Authorization 헤더로 변환하도록 인증 흐름을 재설계
  응답 쿠키에 SameSite=Lax 강제 적용을 추가해 브라우저 자동 첨부 기반 공격면을 함께 축소

Result

- XSS · SSRF · CSRF 각 보안 이슈의 발생 원리와 방어 패턴을 이해

LCP 개선으로 페이지 체감 로딩 속도 **80% 단축** (3.0s → 0.6s)

Why?

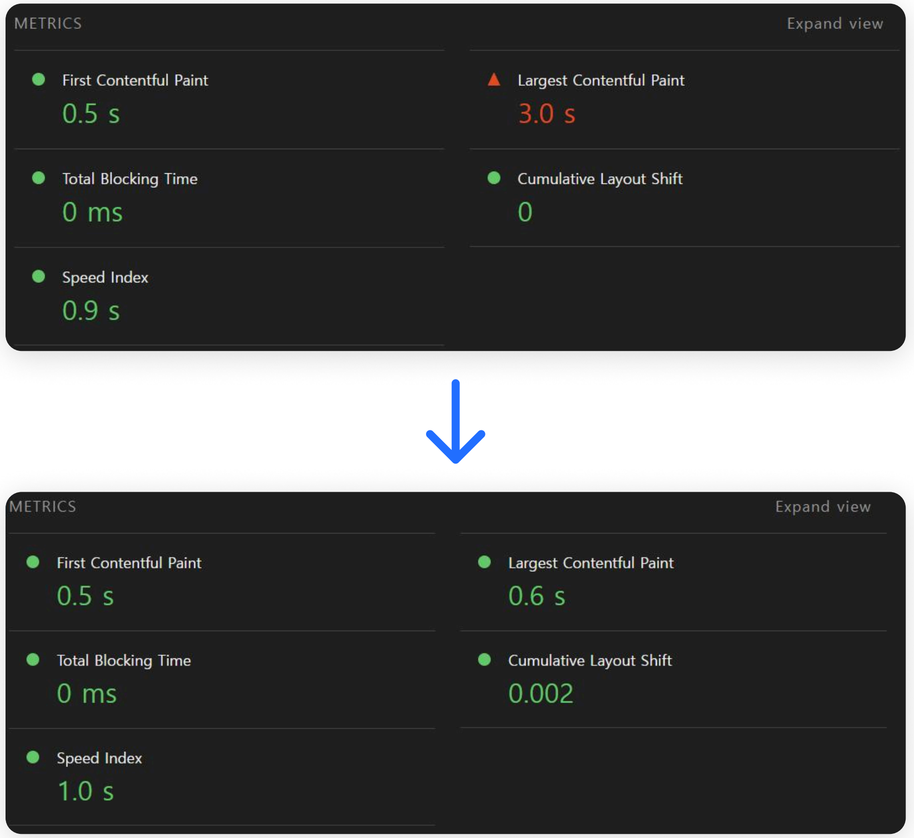
- 운영 배포 후 Lighthouse 측정 결과 Performance 83점, **LCP 가 3초를 초과하는 이슈** 확인

How

- **폰트 서브셋 분할** : pretendard-subset 으로 서브셋 woff2로 분할,
  CSS unicode-range 로 렌더링에 필요한 청크만 브라우저가 선택 요청하도록 변경
- **이미지 최적화** : <Image> sizes 속성 정비로 뷰포트 크기에 맞는 이미지만 요청하도록 제한

Result

- LCP 점수 **3.0초 → 0.6초** (약 80% 단축)
- 폰트 · 데이터 패칭 구조 개선을 통한 LCP 최적화 전략 이해



사용자 접근 가능 전 페이지 SEO 적용

Why?

- 공통 Metadata, OG 만 적용된 상태에서 링크 공유 시 모든 페이지가
  동일한 이미지, 제목으로 노출하여 **어떤 콘텐츠인지 구분 불가능**

How

- **`robots.ts` · `sitemap.ts`** (43개 정적 라우트) 구성으로 SEO 기반 인프라 구축
- 전체 정적 페이지에 **title · description · OG** 이미지 적용
- 동적 라우트(게시글 상세 등) 는 **API 데이터 기반 페이지별 메타데이터** 생성

Result

- 관리자 · 인증 · API 라우트를 제외한 사용자 접근 가능 **모든 페이지에 SEO 적용** 완료.
  페이지 공유 시 각 페이지의 개별 제목과 썸네일 노출

단일 서버 중단 최소화 배포 구현 (GitHub Actions + PM2 cluster)

Why?

- MVP 이후 지속적인 추가 요구사항이 들어오면서 **잦은 배포 필요**
- 기존 배포 방식은 배포 시마다 프로세스 종료 후 재시작으로 인해 **사용자 요청이 끊기는 문제 발생**

How

- 단일 서버 환경이기 때문에 **PM2 클러스터 모드**로 인스턴스 2개를 띄워두고,
  배포 시 pm2 reload 로 인스턴스를 하나씩 재시작하여
  나머지 인스턴스가 요청을 계속 처리하도록 구현

Result

- 추가 인프라 비용 없이 단일 서버 제약 내에서 사용자 체감 다운타임을 최소화하는 배포 환경 구축

PJT 2. 동행

시니어를 위한 쉽고 편한 AI 기반 뱅킹 키오스크

Github · donghang

사라져 가는 은행 점포의 대안을 위해 3D AI 은행원 과 음성을 통해
은행 업무를 볼 수 있는 뱅킹 키오스크 를 개발하는 프로젝트를 기획했습니다.

성과

- 삼성 청년 소프트웨어 AI 아카데미 우수상(1등)

개발 기간

- 2025.03 ~ 2025.04

사용 기술

Electron · TypeScript · Tailwind CSS · Three.js · Blender

개발 인원

	Backend	Frontend	Infra	AI
* 담당 파트				

담당 역할

- 시니어 사용자 화면 설계 및 구현
- VAD 및 3D 은행원 행동 구현
- 보이스피싱 경고 화면 구현
- 모델링 및 애니메이션 제작

주요 기능

- 3D AI 은행원과 음성 기반 대화
- 입금, 이체, 적금 등 다양한 뱅킹 기능 제공
- 음성으로 보이스 피싱 유추

접근성을 고려한 시니어 사용자 화면 UI 설계

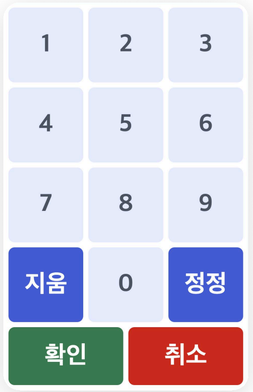
Why?

- 시니어를 위한 플랫폼인 만큼, 시니어 사용자의 접근성 중요

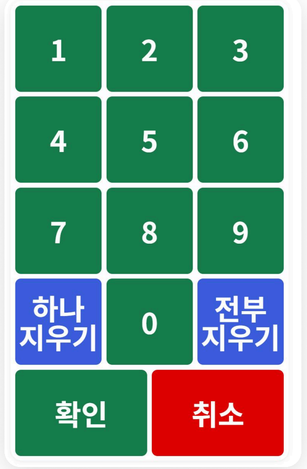
How

- 시니어 사용자의 접근성을 위해 **WCAG 2.1 AA** 레벨을 목표로,
  모든 UI 요소의 명암비를 **4.5 : 1** 이상으로 유지
- 또한, 일반 모드 대비 폰트 크기를 **25%** 확대 하고,
  모든 버튼과 상호 작용 요소의 터치 영역을 최소 **44×44px** 이상으로 확보
- 고령자를 위한 금융권 가이드라인 문서를 참고하여 이체 → 송금하기 등 이해가 쉬운 용어 사용

일반 사용자



시니어 사용자



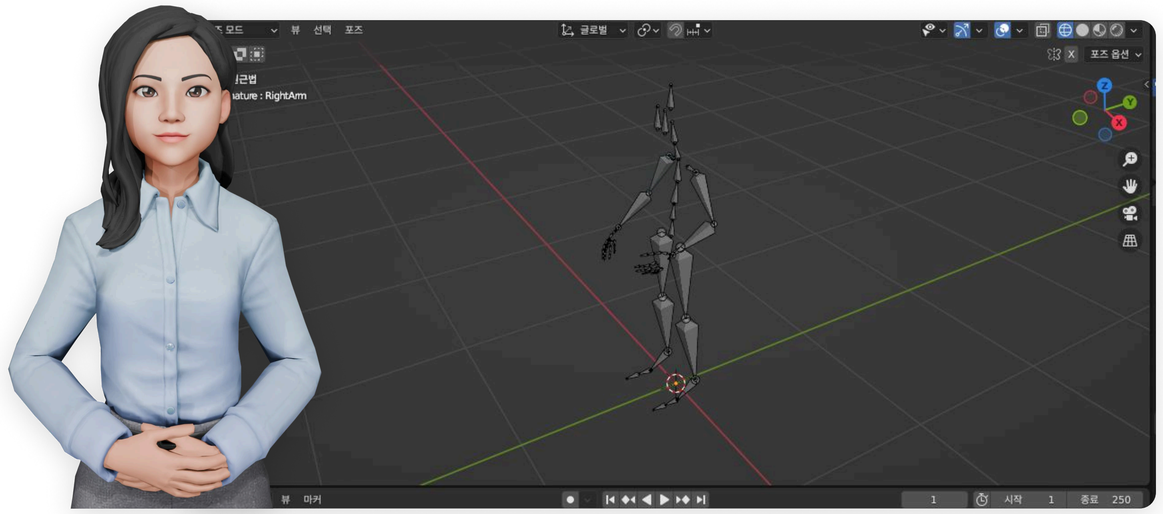
시니어 사용자를 위한 은행원 3D 모델링 및 애니메이션 제작

Why?

- 시니어 사용자에게 실제 은행원과 대화하는 듯한 친밀감 있는 경험 제공
- 애니메이션 부재로 인한 몰입감 저하

How

- Blender 을 활용하여 3D 은행원 모델링을 수정하고, **키프레임 애니메이션** 을 제작
- **Three.js** 를 통해 제작한 3D 모델과 애니메이션을 프로젝트에 적용



하나의 응용프로그램에 두개의 Window 가 있는 멀티 윈도우 키오스크 구현

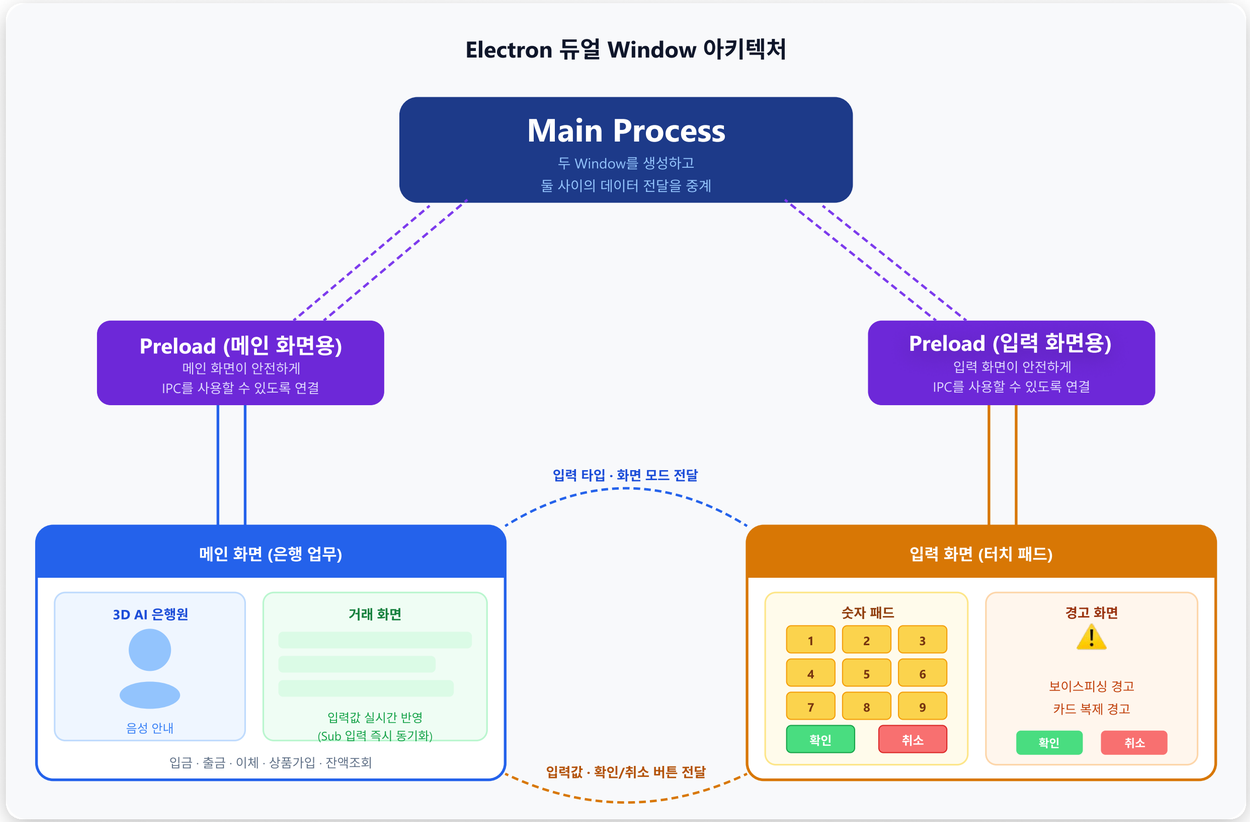
Why?

- 실제 ATM은 메인 디스플레이와 숫자 패드가 물리적으로 분리된 구조
- 이를 소프트웨어로 재현해 실제 ATM에 가까운 사용자 경험을 만들자는 방향

Result

- 메인 화면과 입력 화면이 실시간 동기화되는 듀얼 윈도우 키오스크 구현
- Electron 의 Main / Preload / Renderer 구조에 대한 이해

How



useActionPlay **커스텀 훅**으로 음성 · 자막 · 애니메이션 **3개 시스템 통합 제어**

Why?

- 27개 화면 · 60여 개 상황마다 AI 은행원의 **음성 · 자막 · Three.js 애니메이션을 동시에 관리** 해야 하는 이슈

How

- shouldActivate 하나로 3개 시스템을 동시 구동하는 선언형 인터페이스
  → 컴포넌트는 "언제" 실행할지만 전달, 비동기 타이밍 제어는 훅 내부로 캡슐화
- ref로 중복 실행 방지 → 리렌더링 없이 멱등성 보장
- `onComplete` 콜백으로 음성 종료 시점에 다음 동작을 체이닝
  → 별도 상태 관리 없이 순차 시퀀스 구성
- 훅 + Context 조합으로 관심사 분리. 프로젝트 규모상 별도 상태 머신 도입보다 구현·수정 비용이 낮다고 판단

Result

- 27개 화면 · 60여 개 상황에서 동일 인터페이스로 아바타 행동 제어.
  비동기 충돌 · 메모리 누수 방지 로직을 훅 1곳에서 일괄 관리

PJT 3. 행복하개

유기견 보호소를 위한 보호소 관리 플랫폼

Github · hangbokdog

> 소규모 사설 유기견 보호소에서 네이버 카페로 수작업 관리하던
> 후원 · 봉사 · 유기견 정보를 통합 관리할 수 있는 플랫폼을 개발했습니다.

성과

- 삼성 청년 소프트웨어 AI 아카데미 우수상(2등)

개발 기간

- 2025.04 ~ 2025.05

사용 기술

React · TypeScript · Tailwind CSS · Tanstack Query · Zustand · EasyOCR

개발 인원

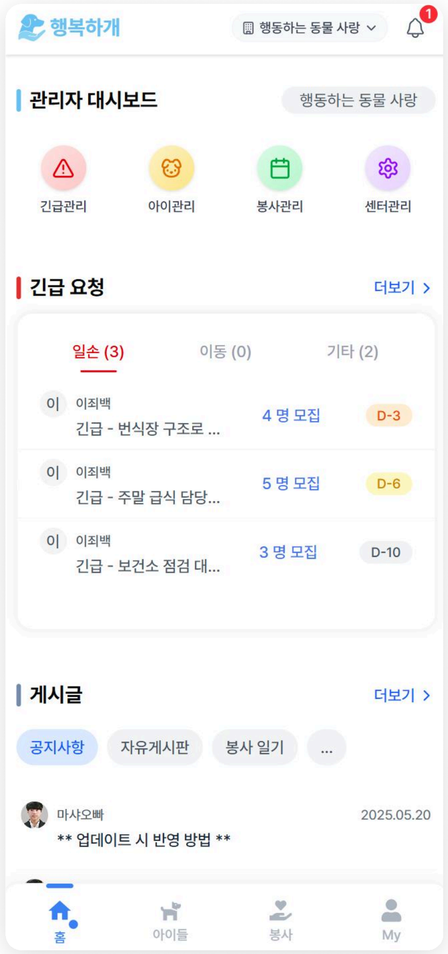
	Backend	Frontend
* 담당 파트	3	2

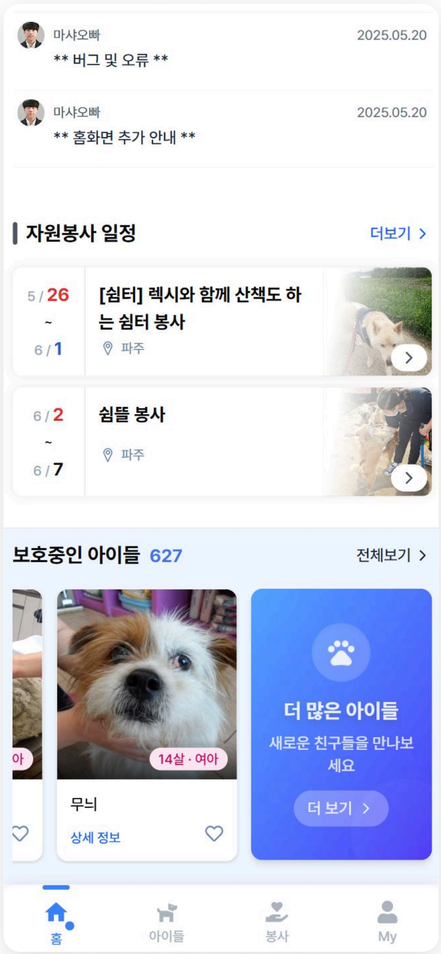
주요 기능

- 보호소 등록 및 관리
- 봉사 및 입양 / 임시보호 신청 등 일반 사용자용 기능
- 관리자 전용 통합 대시보드

담당 역할

- 공통 컴포넌트 설계 및 구현
- 보호소 관리, 관리자 대시보드, 게시글 기능 구현 및 API 연동
- 유기견 정보 이미지 OCR 로 속성 자동 입력 구현





접종 관리 검색을 클라이언트 필터링으로 전환해 반복 검색 시 API 요청 제거

Why?

- 접종 관리 화면은 강아지 이름을 하나씩 검색하며 체크하는 방식으로 운영. 검색 빈도가 높아 매 입력마다 서버 요청이 적절한지 의문 발생
- **검색 빈도가 높아** 자연스럽게 매번 서버에 요청해야 하는지 의문 발생

How

- 보호소당 최대 약 500마리 규모임을 확인 후 트레이드오프를 수치로 검토
- 전체 500마리 페이로드는 약 72KB(비압축) - 초기 로드 비용이 증가하지만 gzip 전송 시 약 15~20KB 수준으로 실사용 영향은 제한적
- 클라이언트 필터링 비용은 performance.now() 기준 CPU 6x 쓰로틀 환경에서 중앙값 0.1ms 로 무시 가능한 수준임을 실측으로 확인
- 위 근거를 바탕으로 전체 데이터를 한 번에 수신하여 **클라이언트에서 직접 필터링**하는 방식 채택
  → API 호출 1번으로 여러번의 검색처리 가능

Result

- **반복 검색 시 API 요청 완전 제거**
- 데이터 규모와 요청 패턴을 실측 수치로 검토한 뒤 아키텍처 결정하는 경험

EasyOCR 기반 유기견 공고 자동 등록 - 이미지 → 폼 자동 입력 구현

Why?

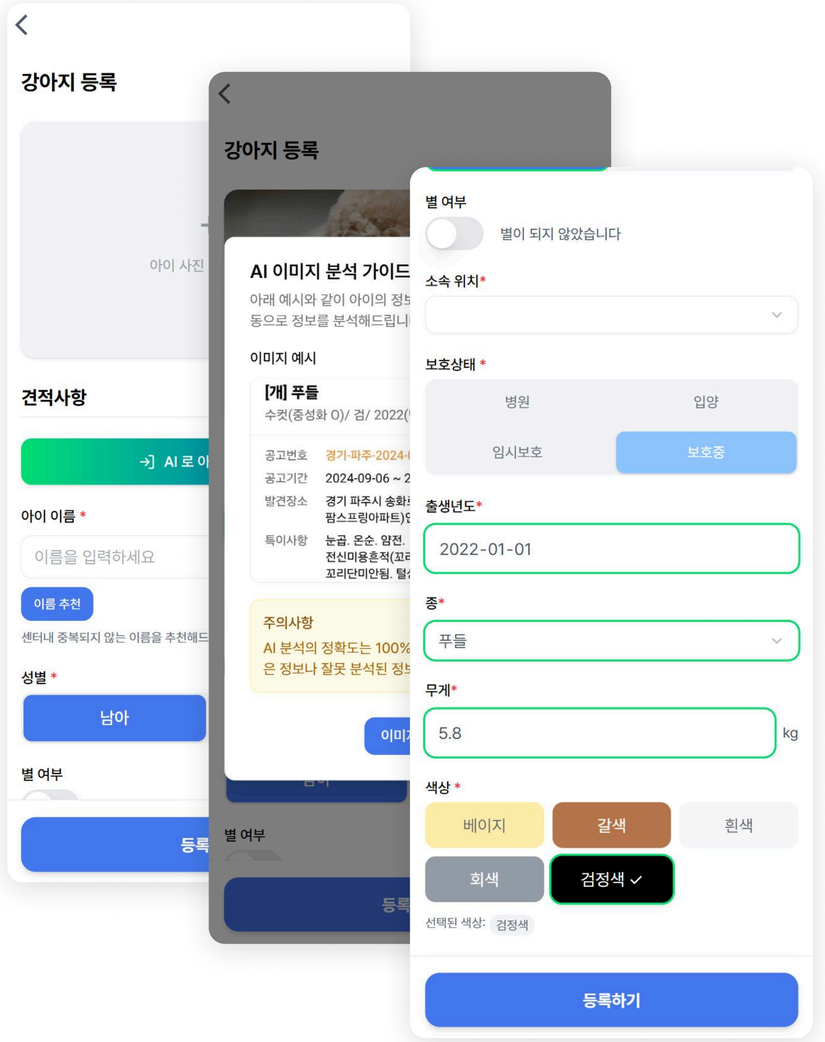
- 보호소 관리자가 유기견 공고 **이미지를 보며 유기견 정보를 직접 입력하는 반복 작업 발생**
- 공고 이미지마다 항목 위치가 달라 정규식 단순 파싱으로는 필드 추출 정확도에 한계 존재

How

- Python **EasyOCR** 라이브러리로 공고 이미지를 텍스트로 변환 후
  성별, 품종 등 키워드를 파싱하여 구조화된 객체로 반환
- 프론트엔드에서 공고 이미지 업로드 시 파싱된 필드 값을 수신하여 해당 폼 필드에 자동 입력
- OCR로 자동 입력된 필드는 하이라이팅하여 **자동 입력 여부를 시각적으로 구분**,
  사용자가 오입력 여부를 즉시 확인하고 수정 가능하도록 안내

Result

- 공고 이미지 업로드만으로 주요 필드 자동 입력 → **반복 수기 입력 작업 대폭 감소**
- EasyOCR 기반 이미지 텍스트 추출 파이프라인 구현 및 Python 서버 AWS 배포 경험 확보



PJT 4. 이집어때

AI 를 활용한 부동산 추천 플랫폼

Github Ezip

> 아파트 매물 정보 및 시세 추이 를 AI 를 활용해 쉽게 볼 수 있게 하고, Naver 뉴스, 핫한 매물 시스템 등 부동산 거래에 도움이 되는 정보를 시각화 하여 간편하게 제공하고자 기획하였습니다.

성과

- 삼성 청년 소프트웨어 AI 아카데미 최우수상(1등)

개발 기간

- 2024.10 ~ 2024.11

사용 기술

React · TypeScript · Chakra UI · Tanstack Query · Spring Boot

개발 인원

Backend	Frontend	Infra
70%	30%	

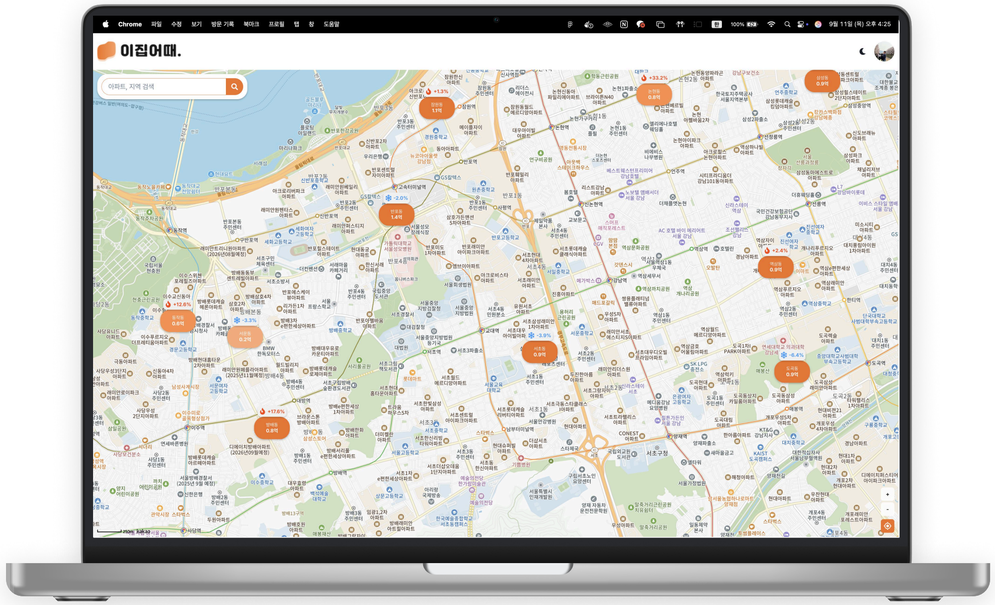
* 담당 파트

주요 기능

- 지도, 차트로 쉽게 볼 수 있는 부동산 정보
- OpenAI API 를 활용한 AI 기반 부동산 시세 예측 및 챗봇 상담
- 부동산 뉴스, 핫한 매물 시스템 등 다양한 부동산 정보 제공

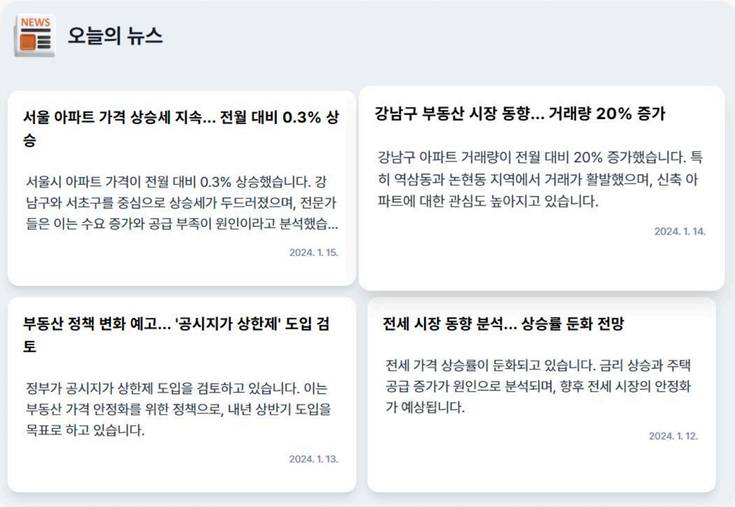
담당 역할

- 네이버 뉴스 Open API 연동 및 화면 개발, OpenAI API 연동 및 챗봇 개발
- 다크모드 구현
- Spring Boot MVC 구현, DB 설계



Naver 뉴스 Open API 를 활용하여 부동산 관련 네이버 뉴스 제공

- Naver 뉴스 API 를 통해 부동산 관련 뉴스를 수집
- Chakra UI 의 Grid 시스템 을 사용해서 반응형 카드 레이아웃과 호버 효과를 가진 뉴스 카드 UI를 구현

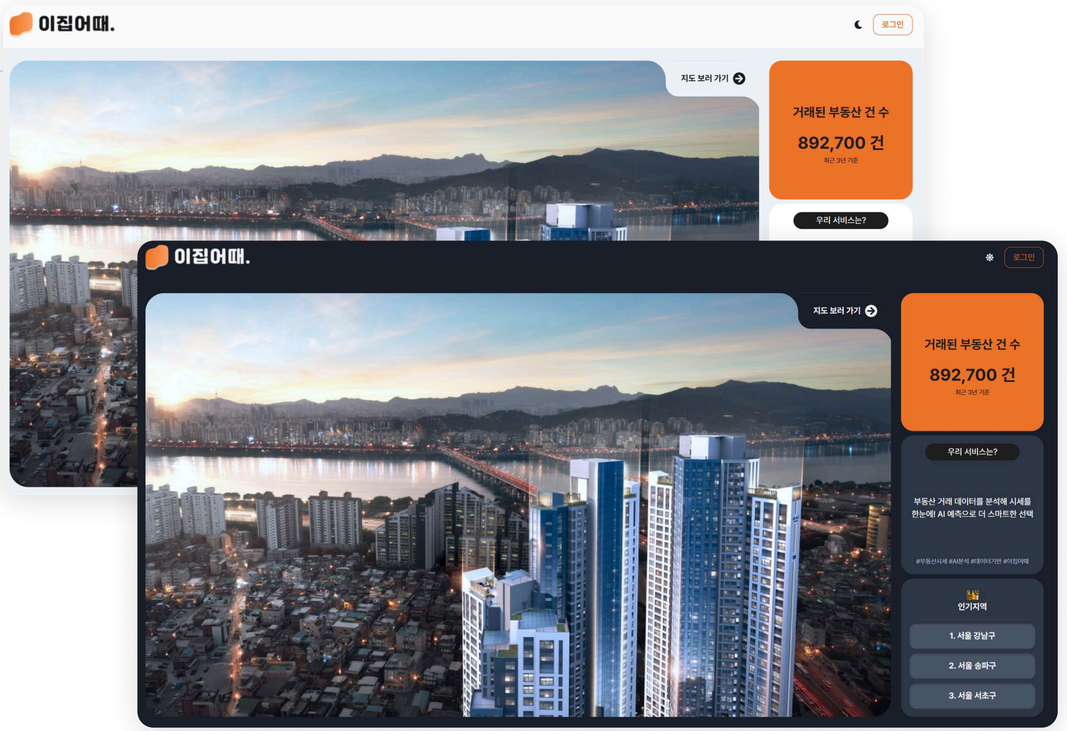


OpenAI API 를 활용한 챗봇 구현

- 사용자에게 어려운 부동산 용어를 쉽게 전달하기 위해 OpenAI API 과 사전 프롬프팅을 통해 챗봇 시스템을 개발
- AI 와 사용자의 대화를 유지시키기 위해 SessionStorage 를 사용하여 사용자의 대화 상태를 저장하고 관리

다크모드 & 반응형 레이아웃

- Chakra UI의 useColorMode와 useColorModeValue 를 활용해
  버튼, 로고, 배경 색상까지 일관된 다크모드를 구현
- extendTheme로 커스텀 색상을 정의하고, 라이트/다크 모드별 버튼 변형을 추가하여
  브랜드 일관성을 유지
- useBreakpointValue를 활용해 화면 크기에 따라 로고, 텍스트 크기, 레이아웃을
  동적으로 전환하는 반응형 UI를 구현



긴 문서 봐주셔서 감사합니다