

PORTFOLIO

이원희 포트폴리오

포트폴리오 사이트:	whd793.github.io
이력서 pdf:	디자인 동아리 회장 1년
깃허브:	github.com/whd793
기술 블로그:	whd793.blog
Linkedin:	linkedin.com/whd793
영문 이력서:	linkedin.com/whd793
이메일:	whd793@gmail.com
전화번호:	010-2551-5567

안녕하세요, 이원희입니다

학력 사항

2019.12 Purdue University (미국 퍼듀 대학교)
웹 프로그래밍 & 디자인과 학사 졸업

경력

2024.07 ~ 2024.12 Twiqli - 스타트업 (풀스택
개발자)

2023.08 ~ 2024.03 Pockapp Games -
스타트업 (게임 개발자)

2022.07 ~ 2023.07 Voodoo.io - 게임 개발자

2020.01 ~ 2020.03 프리랜서 - 곰샤브샤브
(프론트엔드 웹 개발자)

2019.06 ~ 2019.10 프리랜서 - PRPC
(프론트엔드 웹 개발자)

자격증

2023.01 JavaScript (Intermediate) Certificate /
HackerRank

2023.01 React Certificate / HackerRank

2023.01 React Hooks / Linkedin

2021.08 CompTIA Network+ (N10-007) Cert Prep /
Linkedin

2021.07 MySQL Advanced Topics / Linkedin

2021.07 NoSQL Essential Training / Linkedin

2021.06 Learning SQL Programming / Linkedin

2021.05 Advanced Node.js: Scaling Applications /
Linkedin

2021.01 React.js Essential Training / Linkedin

링크

포트폴리오 사이트: whd793.github.io

이력서: whd793.github.io/resumekor

깃허브: github.com/whd793

블로그: whd793.blog

LinkedIn: linkedin.com/whd793

영문 이력서: whd793.github.io/resume

이메일: whd793@gmail.com

전화번호: 010-2551-5567

해외경험

미국: 05.2009 - 05.2020

SKILLS

Frontend

- HTML, CSS3, Javascript, React.js, Redux, Typescript, Next.js, Tailwind CSS, Shadcn, Zustand

Backend

- Node.js, Express.js

DB

- MongoDB, AWS, PostgreSQL, MySQL, Firebase, Redis, Prisma, DrizzleORM, GraphQL

기타

- OpenAI API, Python, C#, Unity, Java, Git, Github, Docker

PROJECTS

- 01 AI Interview - 면접 연습 & 피드백 도우미** 2024.04 - 2024.06 | 프론트엔드 & 백엔드
- 02 PodStream - 팟캐스트 라디오 플랫폼** 2024.01 - 2024.03 | 프론트엔드 & 백엔드
- 03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스** 2024.07 - 2024.12 | 프론트엔드 & 백엔드
- 04 이커머스 사이트** 2023.08 - 2023.10 | 프론트엔드 & 백엔드
- 05 Speedy Hero - 47,000명 다운로드!** 2017.01 - 2017.05 | 게임 개발
- 06 'Hyper Casual Game' 프로젝트** 2023.08 ~ 2024.03 | 게임 개발

01 AI 면접 & 피드백 도우미 소개

Problem

취업 준비 과정에서 면접은 가장 중요한 관문임에도 불구하고, 효과적인 연습 방법이 부족하다는 점을 직접 경험했습니다. 특히 혼자서는 객관적인 피드백을 받기 어렵고, 직무에 특화된 질문을 준비하는 데 한계가 있다는 점이 가장 큰 문제였습니다.

Solution

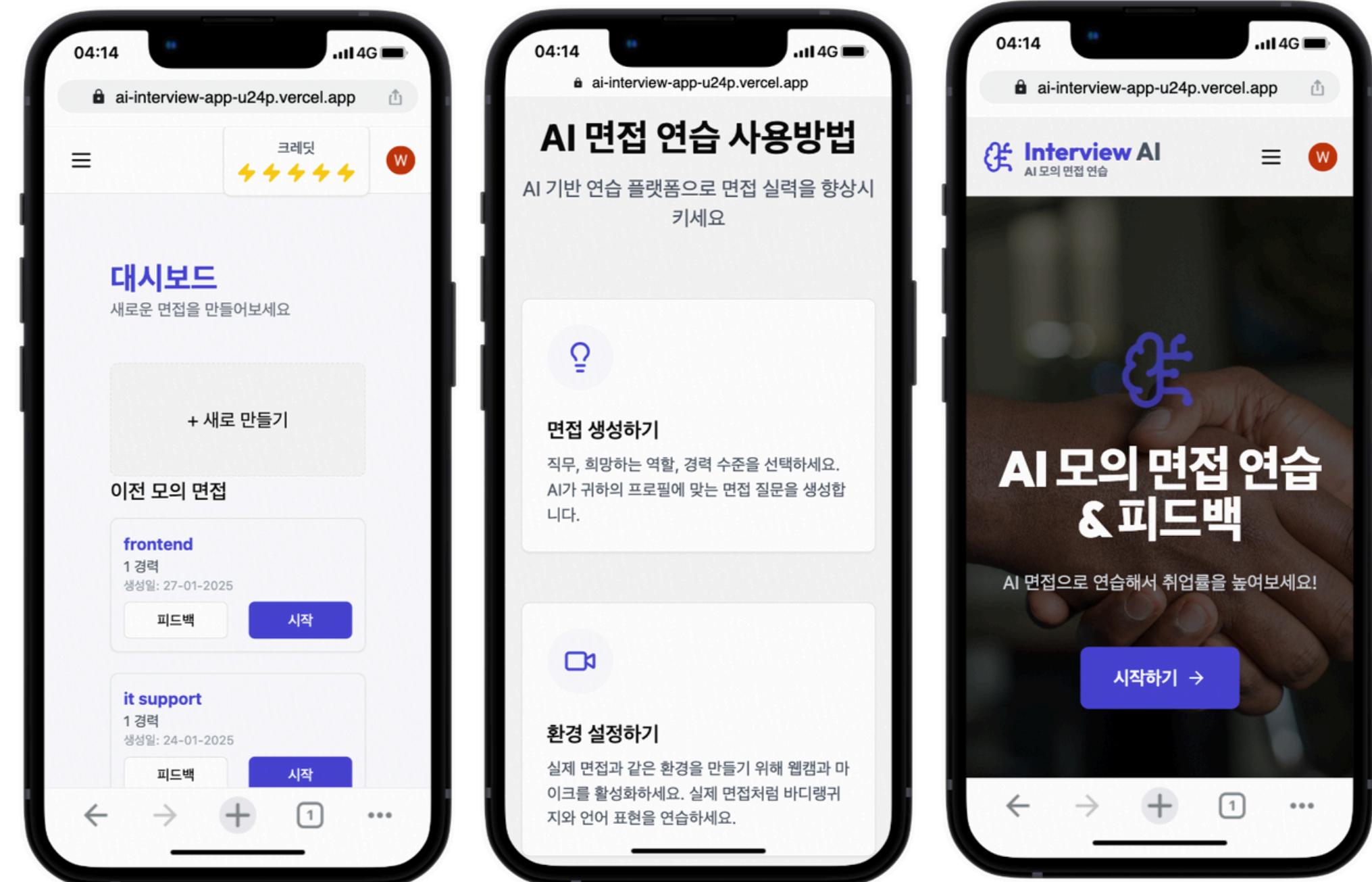
이러한 문제를 해결하기 위해 AI 기술을 활용한 모의면접 플랫폼을 개발하게 되었습니다. 단순히 정해진 질문을 반복하는 것 이 아닌, 사용자의 직무와 경력 수준에 맞춘 맞춤형 질문을 제공하고, 실시간으로 피드백을 제공하는 것을 목표로 했습니다.

개발 기간 기획 및 설계, 디자인: 2024.04.01 ~ 2024.04.15
 개발: 2024.04.16 ~ 2024.05.28
 테스트 및 배포: 2024.05.29 ~ 2024.06.05

개발인원 2명 (PM 및 디자인, 개발)

담당역할 PM, 프론트엔드, 백엔드
 • 서비스 기획 및 방향성 설정
 • 메인 로직 구현 (프론트엔드, 백엔드), API design, DB 설계

깃허브 <https://github.com/whd793/AI-Interview-App>
Demo <https://ai-interview-app-u24p.vercel.app/>



Frontend

- Next.js 13: App Router를 활용한 서버 사이드 렌더링 및 라우팅 구현
- React: 컴포넌트 기반 UI 개발 및 Hooks를 활용한 상태 관리
- Tailwind CSS: 반응형 디자인 및 커스텀 UI 구현
- Lucide React: 아이콘 시스템
- Clerk: 사용자 인증 시스템
- Shadcn UI: 재사용 가능한 컴포넌트 라이브러리 활용
- 모바일 지원 반응형 디자인

Backend

- PostgreSQL: 관계형 데이터베이스 설계 및 구현
- Drizzle ORM: 타입 안전성이 보장된 데이터베이스 쿼리 작성

State Management

- React Context를 활용한 언어 설정 관리
- useState를 통한 로컬 상태 관리
- useEffect를 활용한 실시간 업데이트

DevOps & Tools

- Git & GitHub: 버전 관리 및 협업
- Vercel: 자동화된 배포 및 호스팅
- ESLint & Prettier: 코드 품질 관리

APIs and Integration

- Google Gemini AI API: 맞춤형 면접 질문 생성 및 답변 평가
- 음성 입력을 위한 Speech Recognition API
- WebRTC: 실시간 음성 인식 및 처리

AI 면접 시스템

- Google Gemini AI를 활용한 직무별 맞춤형 면접 질문 생성
- 음성 인식 API를 통한 실시간 답변 수집 및 텍스트 변환
- 다국어 지원 시스템 설계 및 구현 (한국어, 영어, 일본어, 중국어)
- 답변에 대한 AI 기반 실시간 평가 및 피드백 제공

크레딧 시스템

- 사용자별 크레딧 관리 시스템 구현
- 실시간 크레딧 자동 충전 기능 (1분당 1크레딧)
- React Hooks와 Moment.js를 활용한 실시간 타이머 구현
- 크레딧 사용 및 충전 히스토리 관리

면접 관리 시스템

- 과거 면접 기록 저장 및 검토 기능
- 종합적인 피드백 이력 관리
- 상세한 성과 지표 제공
- 개선 영역 식별 및 학습 방향 제시

종합 분석 대시보드

- Recharts 라이브러리를 활용한 면접 데이터 시각화
- 개인별 면접 성과 추적 및 상세 분석
- 답변 품질 평가 및 개선점 도출
- 직무별 성과 분석 및 맞춤형 학습 방향 제시
- 시간에 따른 성장 과정 모니터링

보안 및 사용자 관리

- Clerk을 활용한 안전한 사용자 인증 시스템
- 철저한 개인정보 보호 및 데이터 암호화
- API 경로 보호 및 접근 제한
- 입력값 검증 및 보안 데이터베이스 쿼리 적용
- 사용자 활동 기록 및 모니터링

실시간 처리 최적화

- 음성 인식 및 AI 응답 처리 지연 문제 해결
- WebSocket을 활용한 실시간 데이터 동기화
- 브라우저 호환성 이슈 해결

프론트엔드 최적화

- Code Splitting을 통한 초기 로딩 시간 단축
- 이미지 최적화 및 레이지 로딩 구현
- 컴포넌트 메모이제이션을 통한 리렌더링 최소화

상태 관리 최적화

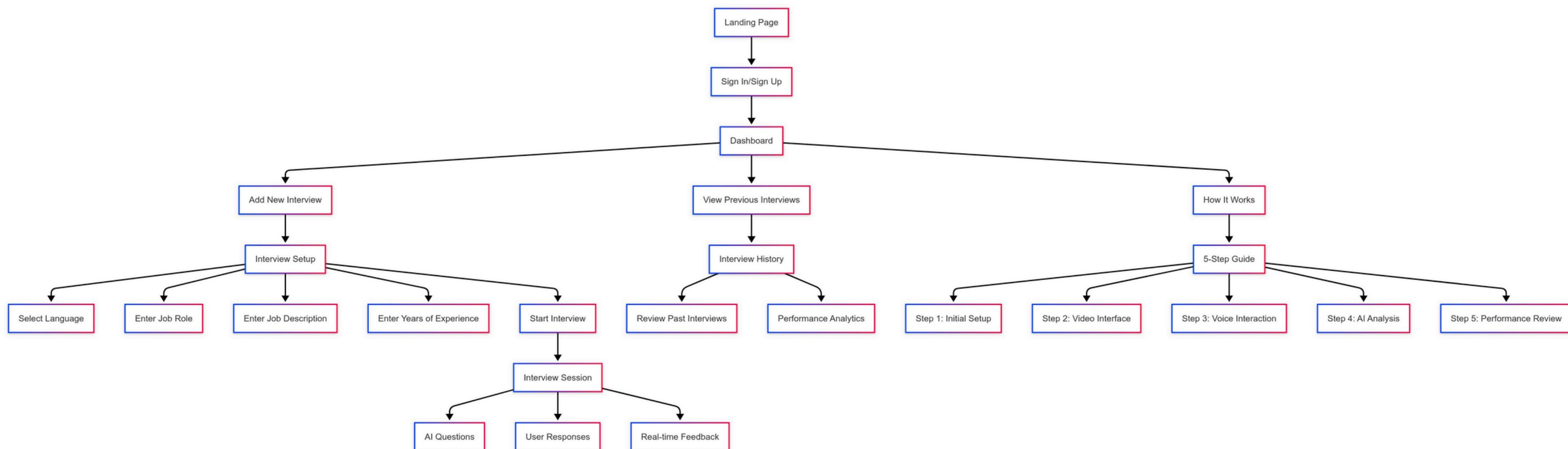
- React Context와 Custom Hooks를 활용한 전역 상태 관리
- 크레딧 시스템의 동시성 이슈 해결
- 캐싱 전략 구현으로 성능 개선

다국어 처리

- 동적 언어 전환 시스템 구현
- AI 모델의 다국어 처리 최적화
- 언어별 폰트 및 레이아웃 대응

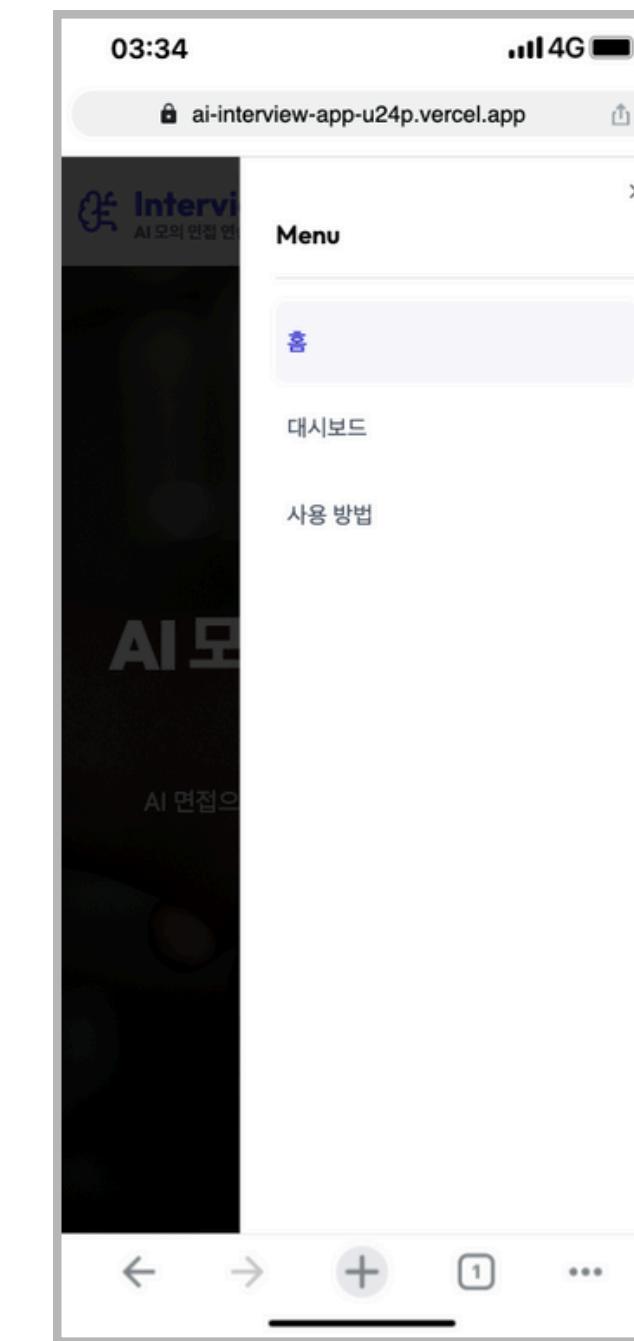
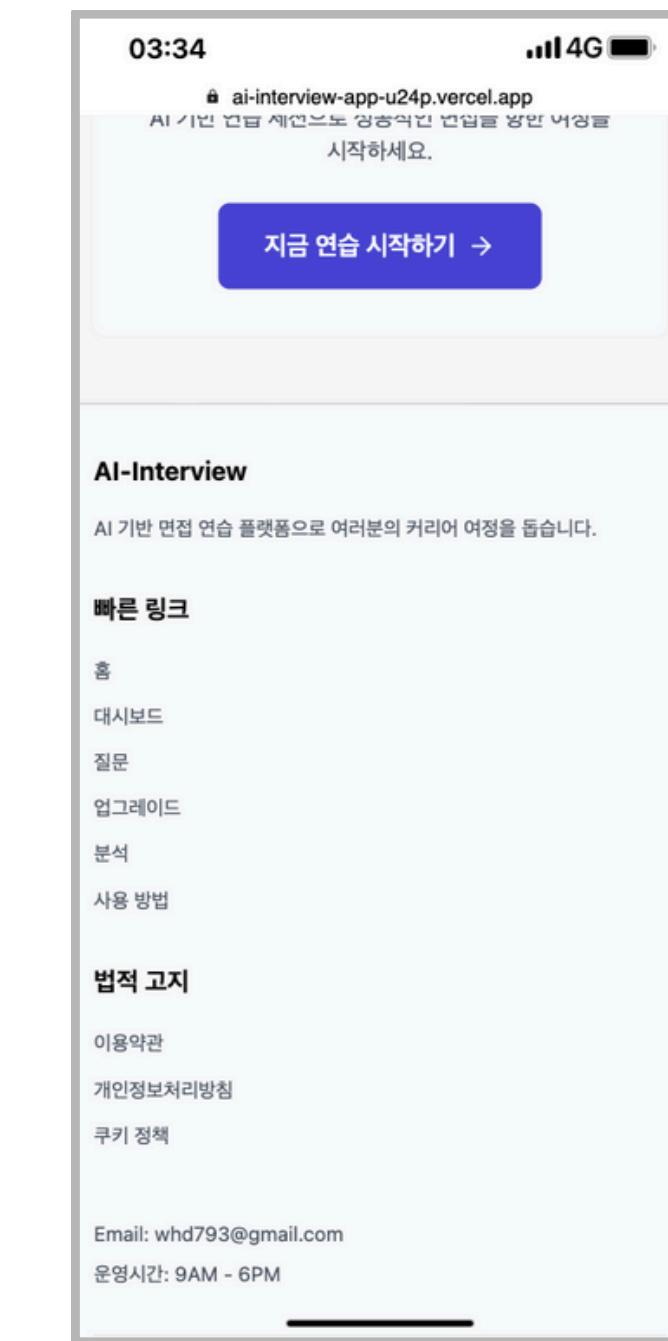
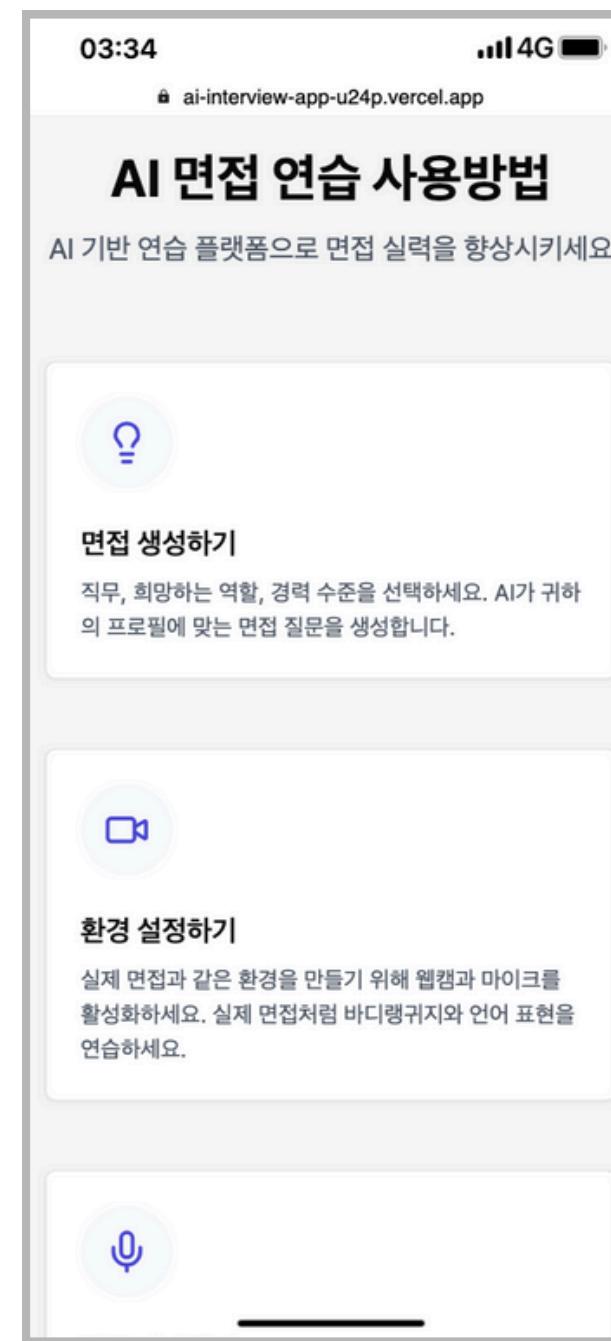
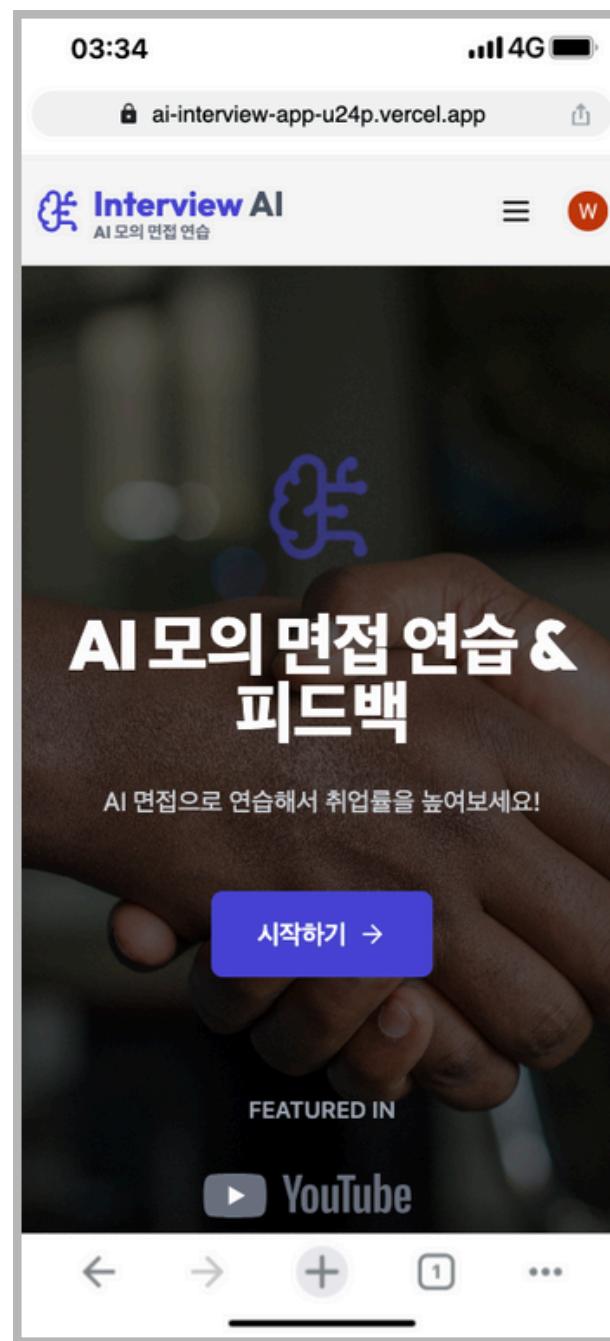
01 AI 면접 & 피드백 도우미

User Flow



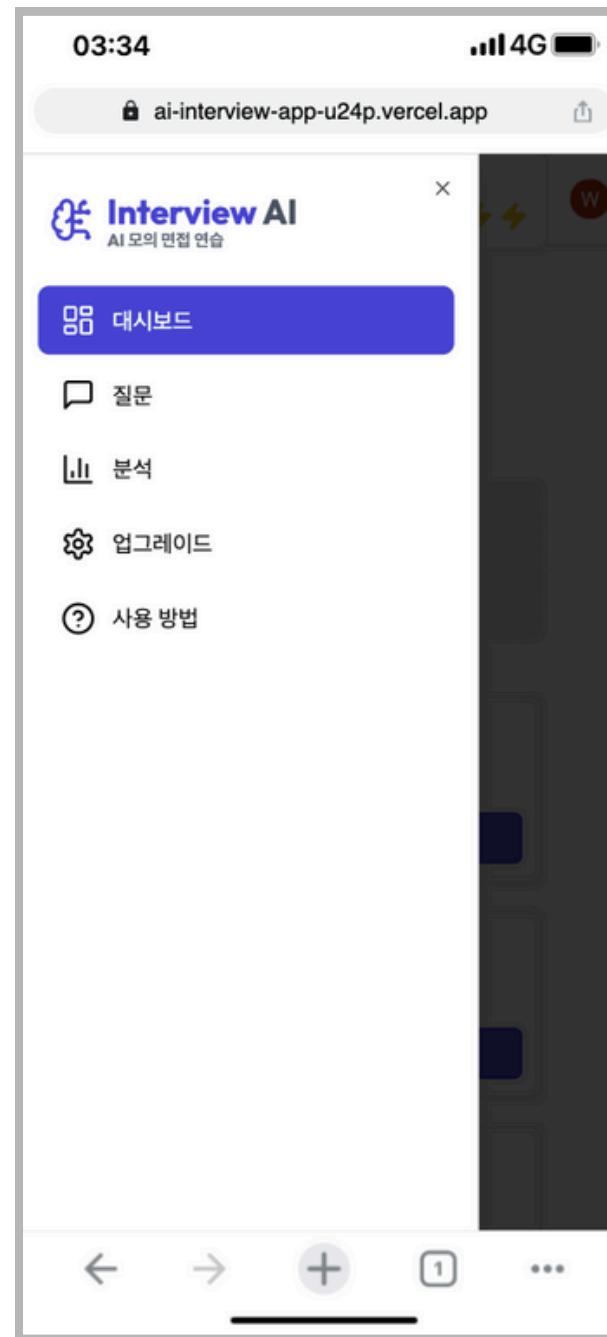
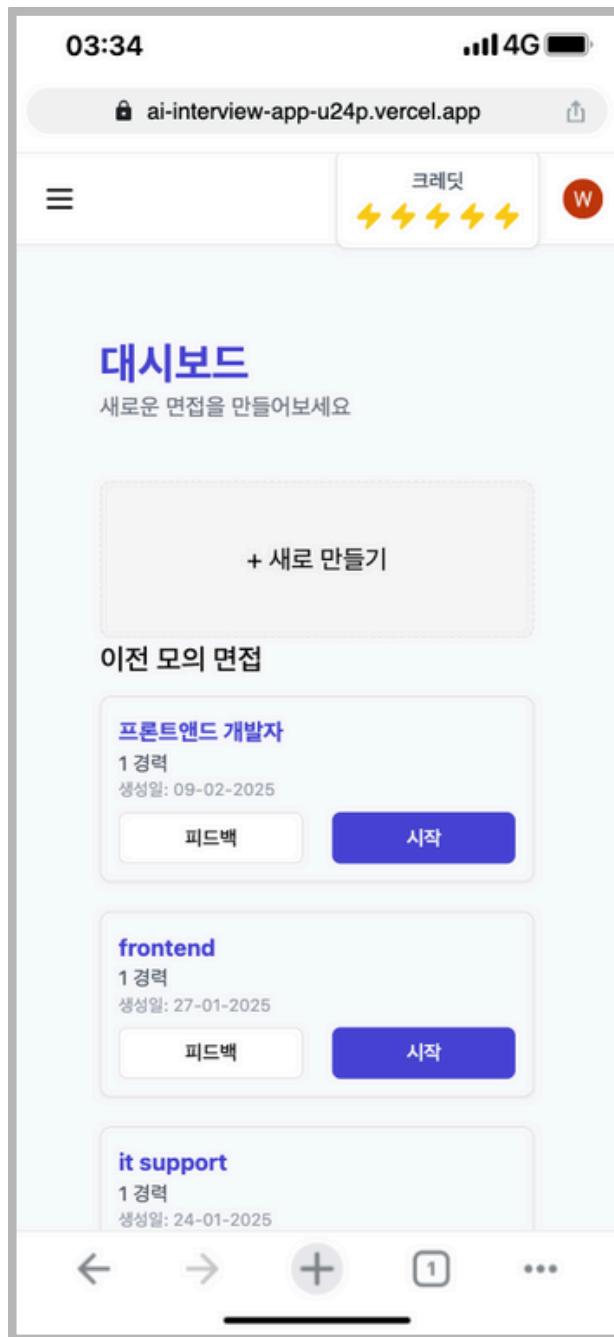
01 AI 면접 & 피드백 도우미

홈페이지



01 AI 면접 & 피드백 도우미

대시보드, 다국어 지원, 크레딧 관리 및 자동 충전



Interview AI

AI Mock Interview Practice

Credits 5

English

Dashboard

Questions

Analytics

Upgrade

How it Works?

Dashboard

Create and Start your AI Mockup Interview

+ Add New

Previous Mock Interview

frontend
1 Years of Experience
Created At: 20-02-2025
Feedback Start

프론트엔드 개발자
1 Years of Experience
Created At: 09-02-2025
Feedback Start

frontend
1 Years of Experience
Created At: 27-01-2025
Feedback Start

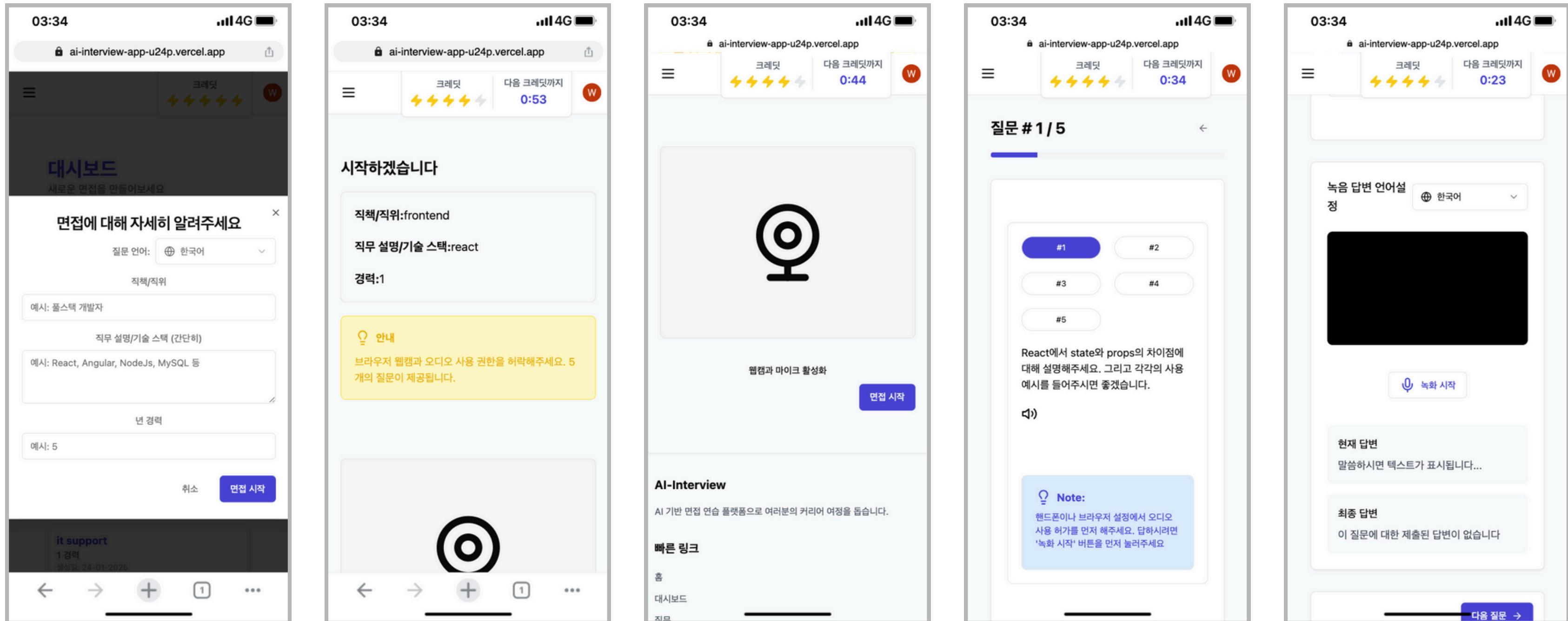
it support
1 Years of Experience
Created At: 24-01-2025
Feedback Start

react
2 Years of Experience
Created At: 16-12-2024
Feedback Start

test
2 Years of Experience
Created At: 16-12-2024
Feedback Start

01 AI 면접 & 피드백 도우미

AI를 활용한 직무별 맞춤형 면접 질문 생성, 음성 인식을 통한 실시간 답변 수집 및 텍스트 변환



01 AI 면접 & 피드백 도우미

답변에 대한 AI 기반 실시간 평가 및 피드백 제공

The image displays three screenshots of the Interview AI platform, illustrating the AI-based interview feedback process.

Screenshot 1: Feedback Details

Congratulations! Here is your interview feedback. Your overall interview rating: 7/10. Find below interview question with correct answer, Your answer and feedback for improvement.

리액트에서 상태 관리를 어떻게 하고 있습니까? Redux, Context API, Zustand 등 다양한 방법이 있는데, 어떤 것을 선호하고 그 이유는 무엇입니까?

Rating: 2
Your Answer: 상태 관리는 어떻게 하고 있습니까
Correct Answer: 프로젝트의 규모와 복잡도에 따라 다르게 접근합니다. 작은 프로젝트에서는 Context API를 사용하여 간편하게 상태를 관리하고, 복잡하고 대규모 프로젝트에서는 Redux를 사용하여 상태 관리를 명확하게 분리하고 예측 가능하게 만들습니다. Redux를 사용할 때는 Redux Toolkit을 활용하여 보일러플레이트 코드를 줄이고 개발 효율을 높입니다. Zustand도 간단한 상태 관리에 효과적이라고 알고 있으며, 경량성이 필요한 프로젝트에서 유용하게 활용할 수 있다고 생각합니다. 각각의 장단점을 잘 이해하고 프로젝트 특성에 맞춰 최적의 방법을 선택하는 것이 중요하다고 생각합니다.
Feedback: 답변이 질문에 제대로 답하지 못했습니다.

Screenshot 2: Platform Navigation

Interview AI
AI Mock Interview Practice

- Dashboard
- Questions
- Analytics
- Upgrade
- How it Works?

Screenshot 3: Final Feedback Summary

Congratulations! Here is your interview feedback. Your overall interview rating: 7/10. Find below interview question with correct answer, Your answer and feedback for improvement.

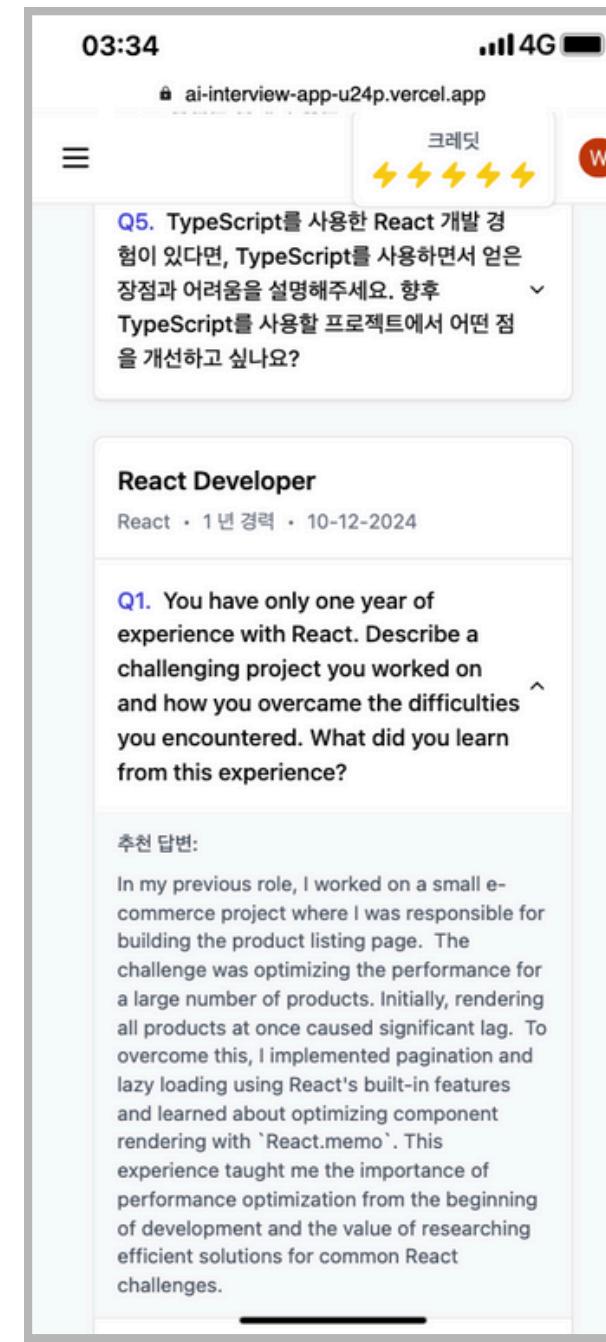
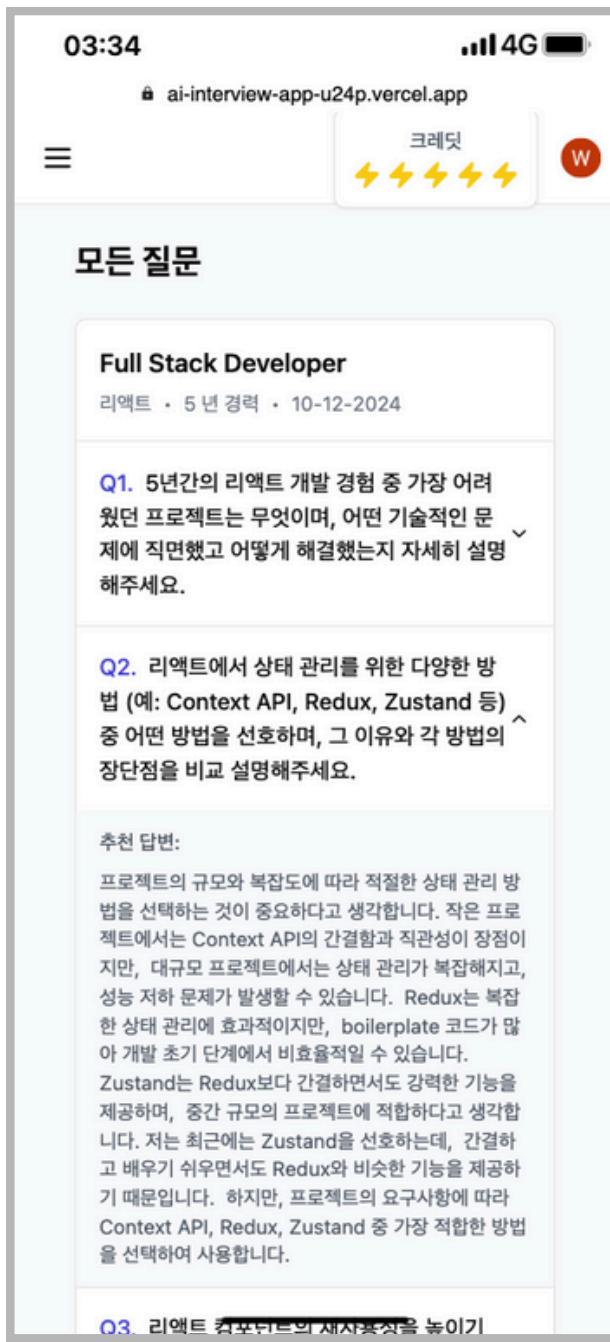
리액트에서 상태 관리를 어떻게 하고 있습니까? Redux, Context API, Zustand 등 다양한 방법이 있는데, 어떤 것을 선호하고 그 이유는 무엇입니까?

Rating: 2
Your Answer: 상태 관리는 어떻게 하고 있습니까
Correct Answer: 프로젝트의 규모와 복잡도에 따라 다르게 접근합니다. 작은 프로젝트에서는 Context API를 사용하여 간편하게 상태를 관리하고, 복잡하고 대규모 프로젝트에서는 Redux를 사용하여 상태 관리를 명확하게 분리하고 예측 가능하게 만듭니다. Redux를 사용할 때는 Redux Toolkit을 활용하여 보일러플레이트 코드를 줄이고 개발 효율을 높입니다. Zustand도 간단한 상태 관리에 효과적이라고 알고 있으며, 경량성이 필요한 프로젝트에서 유용하게 활용할 수 있다고 생각합니다. 각각의 장단점을 잘 이해하고 프로젝트 특성에 맞춰 최적의 방법을 선택하는 것이 중요하다고 생각합니다.
Feedback: 답변이 질문에 제대로 답하지 못했습니다. '상태 관리는 어떻게 하고 있습니까?'라는 질문은 어떤 라이브러리(Redux, Context API, Zustand 등)를 사용하는지, 그리고 그 이유는 무엇인지 구체적으로 설명하라는 질문입니다. 사용하는 방법과 이유를 자세히 설명해야 합니다. 예를 들어, "저는 Zustand를 사용하고 있는데, 간결하고 배우기 쉽기 때문입니다. 특히 (구체적인 상황이나 예시)"와 같이 답변해야 합니다.

Go Home

01 AI 면접 & 피드백 도우미

과거 면접 모든 질문 저장 및 검토 기능, 피드백 및 추천 답변 이력 관리



This screenshot shows a desktop browser window for the 'Interview AI' platform. The header features the logo 'Interview AI' and the subtitle 'AI Mock Interview Practice'. On the right side of the header are buttons for 'English' (with a dropdown arrow), 'Credits' (with five yellow lightning bolt icons), and a red circular 'W' button. The main content area is titled 'All Questions' and shows a list of interview questions for a 'Full Stack Developer' role. The first question is identical to the one shown in the smartphone screenshots. Subsequent questions are in English:

Q1. 5년간의 리액트 개발 경험 중 가장 어려웠던 프로젝트는 무엇이며, 어떤 기술적인 문제에 직면했고 어떻게 해결했는지 자세히 설명해주세요.

Q2. 리액트에서 상태 관리를 위한 다양한 방법 (예: Context API, Redux, Zustand 등) 중 어떤 방법을 선호하며, 그 이유와 각 방법의 장단점을 비교 설명해주세요.

Q3. 리액트 컴포넌트의 재사용성을 높이기 위한 당신만의 전략이나 기법은 무엇입니까? 구체적인 예시를 들어 설명해주세요.

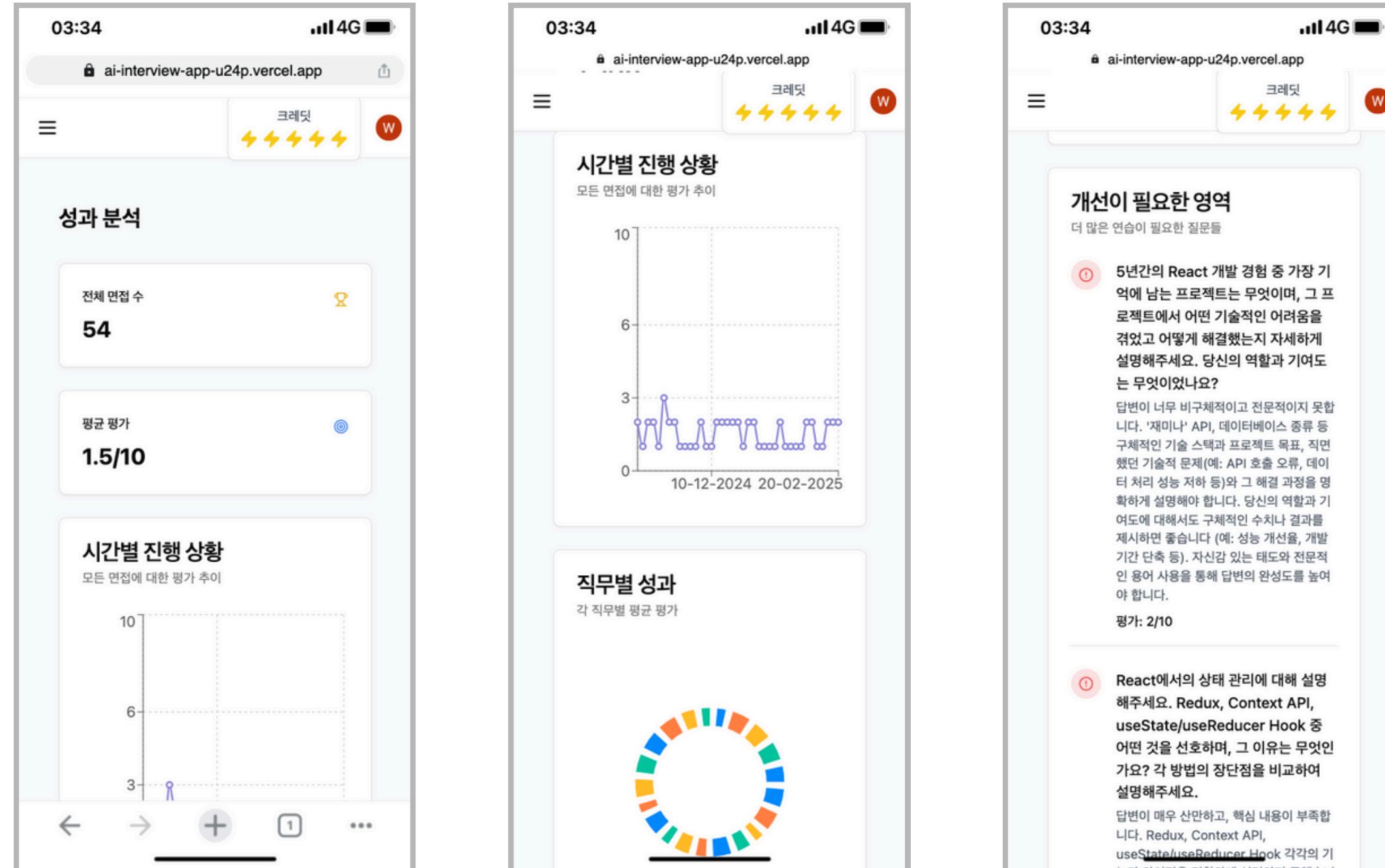
Q4. TypeScript를 사용한 리액트 개발 경험이 있다면, TypeScript를 사용하면서 얻은 장점과 어려움을 설명해주세요.

Q5. 리액트 개발에서 테스트(unit test, integration test 등)에 대한 당신의 접근 방식은 무엇이며, 어떤 테스트 프레임워크를 사용해 왔는지 설명해주세요.

At the bottom of the page, another 'Full Stack Developer' section is visible, showing a different profile and a brief description.

01 AI 면접 & 피드백 도우미

종합 분석 대시보드, 면접 데이터 시각화, 개인별 면접 성과 추적 및 상세 분석, 답변 품질 평가 및 개선점 도출



02 PodStream - 팟캐스트 라디오 플랫폼 소개

PodStream은 사용자들이 오디오와 비디오 팟캐스트를 쉽게 공유하고 시청/청취할 수 있는 스트리밍 플랫폼입니다. React와 Node.js를 기반으로 개발되었으며, 직관적인 UI/UX와 안정적인 스트리밍 서비스를 제공합니다.

이 프로젝트를 통해 현대적인 웹 애플리케이션 개발에 필요한 다양한 기술과 방법론을 실제로 적용하고 배울 수 있었습니다. 특히 사용자 경험과 성능 최적화에 중점을 두어 실제 서비스 수준의 애플리케이션을 구현하였습니다.

개발 기간 기획 및 설계, 디자인: 2024.01.15 ~ 2024.01.29

 개발: 2024.01.30 ~ 2024.03.05

 테스트 및 배포: 2024.03.06 ~ 2024.03.13

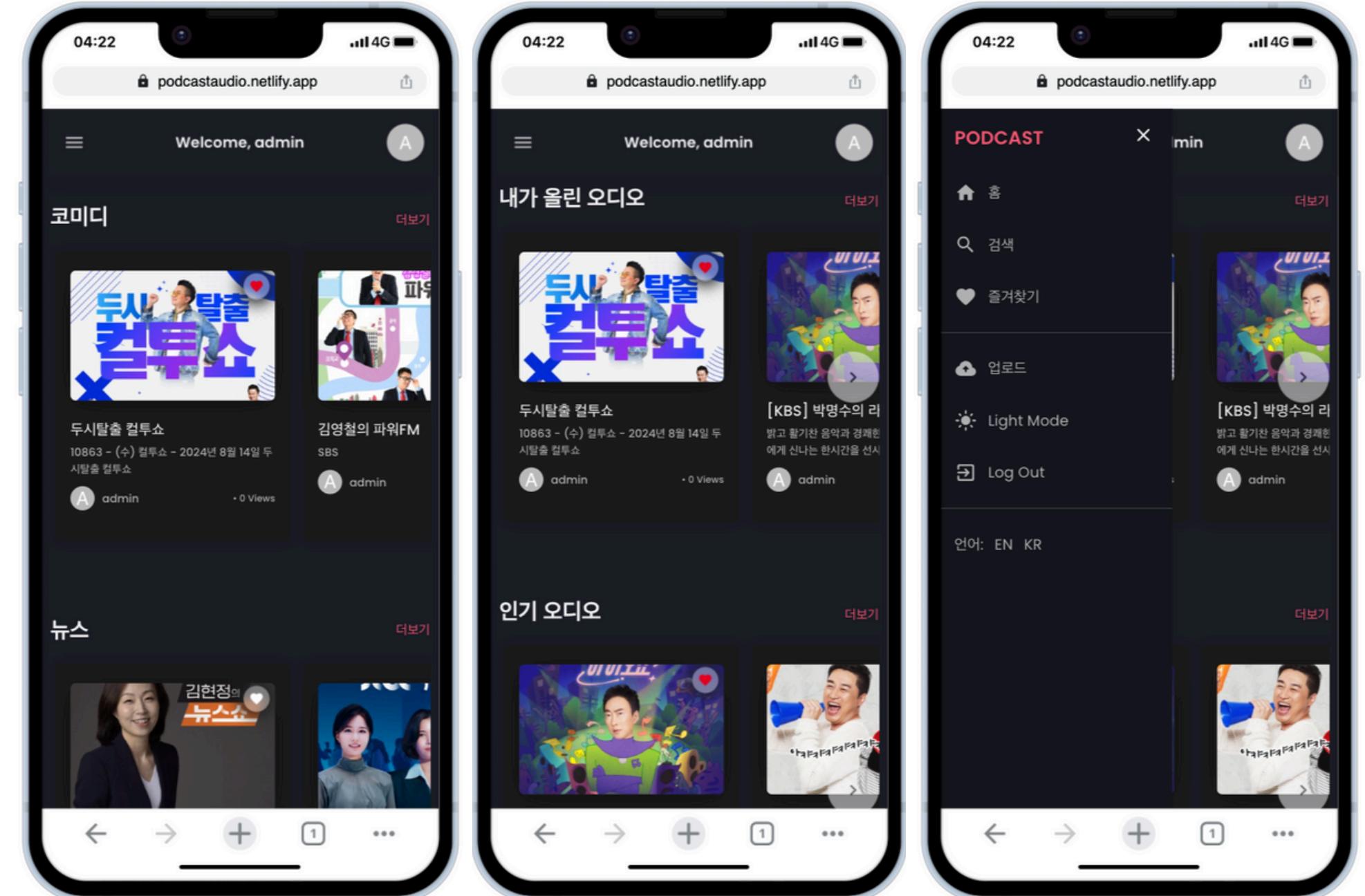
개발인원 2명 (PM 및 디자인, 개발)

담당역할 PM, 프론트엔드, 백엔드

- 서비스 기획 및 방향성 설정
- 메인 로직 구현 (프론트엔드, 백엔드), API design, DB 설계

깃허브 github.com/whd793/podcast

Demo <https://podcastaudio.netlify.app/>



02 PodStream - 팟캐스트 라디오 플랫폼

개발 환경

Frontend:

- React - 컴포넌트 기반 아키텍처를 통한 효율적인 UI 구현
- Redux Toolkit (상태 관리) - 중앙 집중식 상태 관리로 데이터 흐름 최적화
- Styled-Components (스타일링) - 동적이고 재사용 가능한 스타일링 구현
- Material-UI (UI 컴포넌트) - 반응형 디자인과 일관된 사용자 경험 제공
- i18next를 활용한 한영 다국어 지원 기능 구현

Backend:

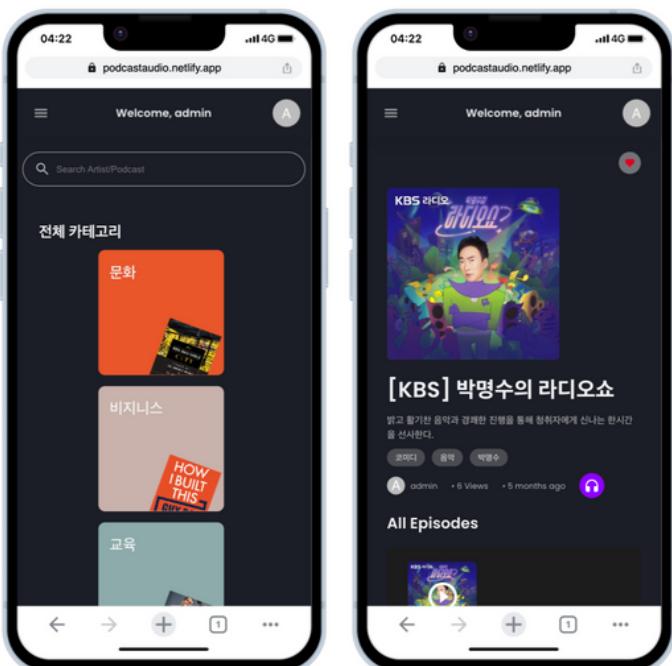
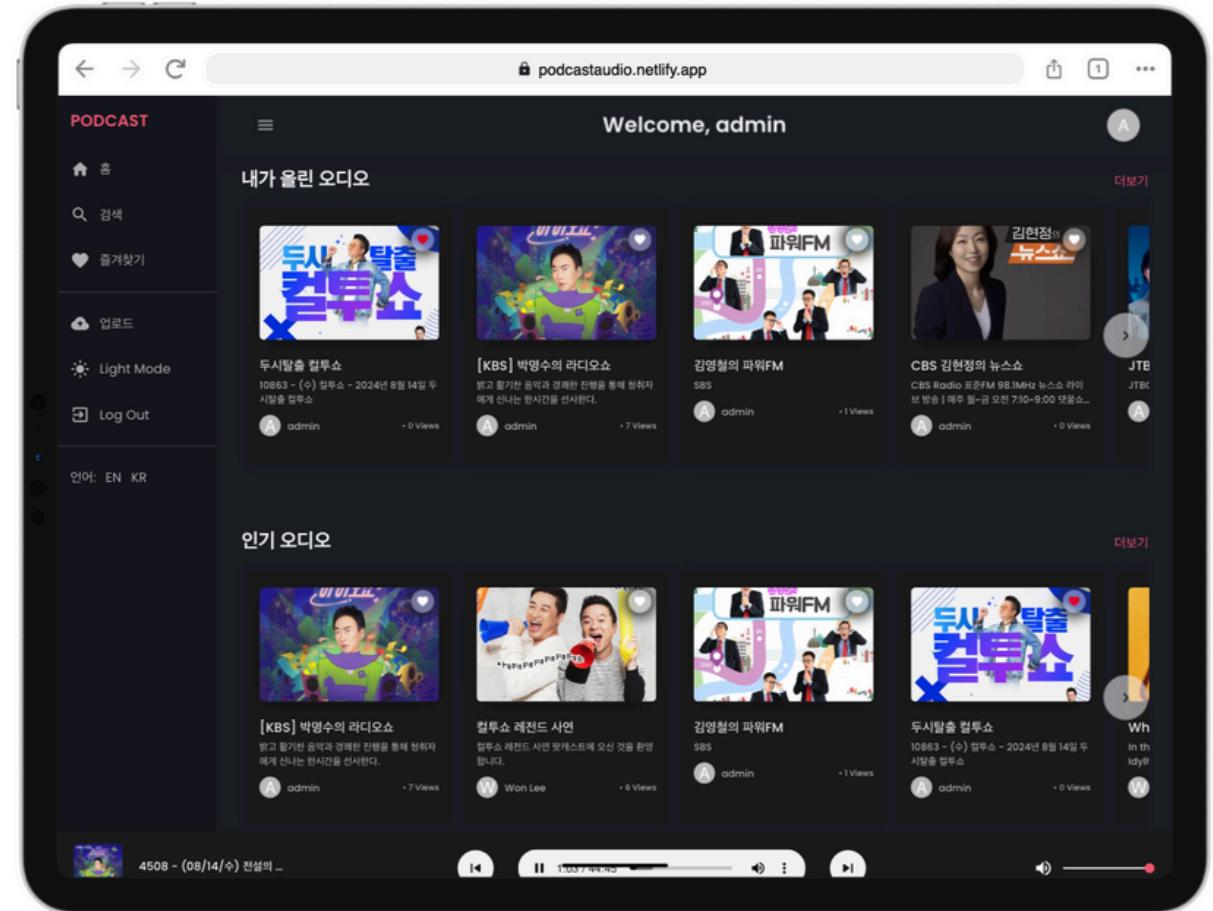
- Node.js - RESTful API, 메인 로직 구축
- Express.js
- JWT와 카카오 OAuth를 활용한 사용자 인증 시스템 구현

DB:

- MongoDB (데이터베이스) - NoSQL 데이터베이스를 활용한 유연한 데이터 구조
- AWS S3 (파일 스토리지) - 미디어 파일 저장 및 스트리밍 서비스

배포(Deployment):

- 프론트엔드 - Vercel
- 백엔드 - Render



02 PodStream - 팟캐스트 라디오 플랫폼

핵심 기능 구현

- 사용자 인증 및 보안 시스템 구현

- JWT 토큰 기반의 보안 인증 시스템 설계 및 구현
- 카카오 OAuth를 활용한 소셜 로그인 통합
- 이메일 인증 및 비밀번호 재설정 기능 개발
- 사용자 세션 관리 및 권한 기반 접근 제어 구현

- 멀티미디어 스트리밍 시스템 개발

- 실시간 재생 진행률 추적 및 동기화 기능이 포함된 커스텀 오디오 플레이어 구현
- 재생 속도 조절, 볼륨 제어 등 고급 플레이어 기능 개발
- 사용자별 재생 기록 저장 및 이어듣기 기능 구현
- 모바일 환경에서의 백그라운드 재생 지원
- 이전/다음 에피소드 전환

- 클라우드 기반 미디어 관리 시스템

- AWS S3를 활용한 대용량 미디어 파일 업로드/다운로드 시스템 구축
- 미디어 파일 메타데이터 관리 및 최적화
- 안정적인 스트리밍을 위한 파일 포맷 처리 및 저장소 관리
- 클라우드 스토리지 비용 최적화 구현

- 콘텐츠 관리

- 팟캐스트 업로드 및 관리
- 에피소드 추가 및 수정
- 카테고리별 콘텐츠 분류
- 즐겨찾기 기능

- UI/UX 및 다국어 지원

- Redux Toolkit을 활용한 효율적인 상태 관리 구현
- i18next를 활용한 한국어/영어 다국어 처리 시스템 개발
- 사용자 위치 기반 자동 언어 전환 기능 구현
- 반응형 디자인으로 모바일/데스크톱 환경 모두 지원

- 추가 주요 기능

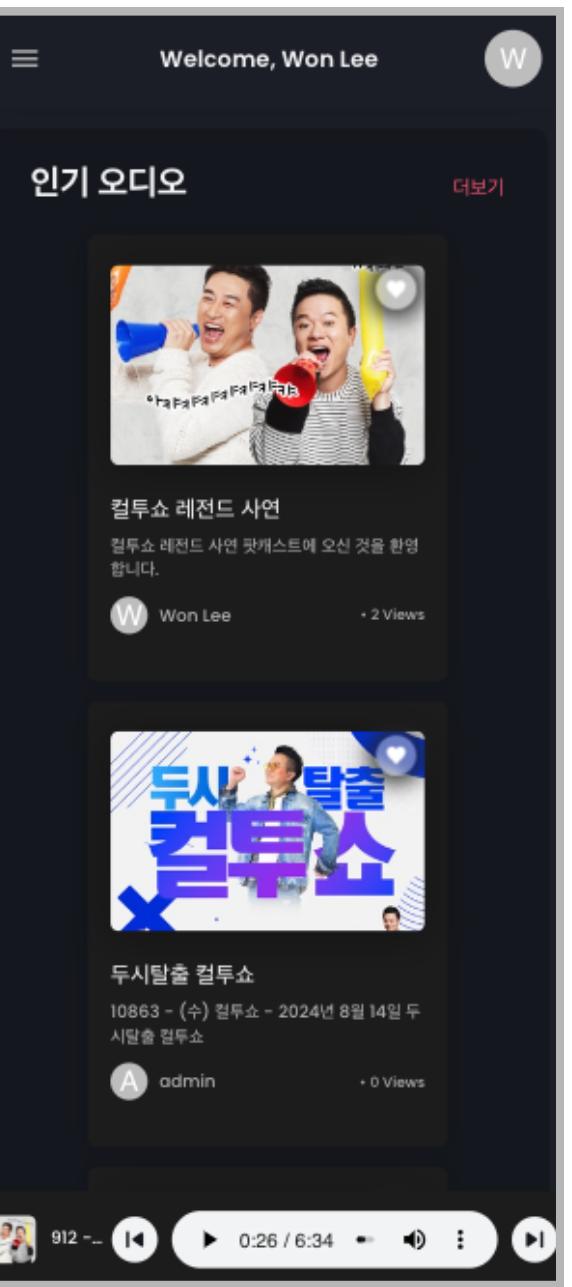
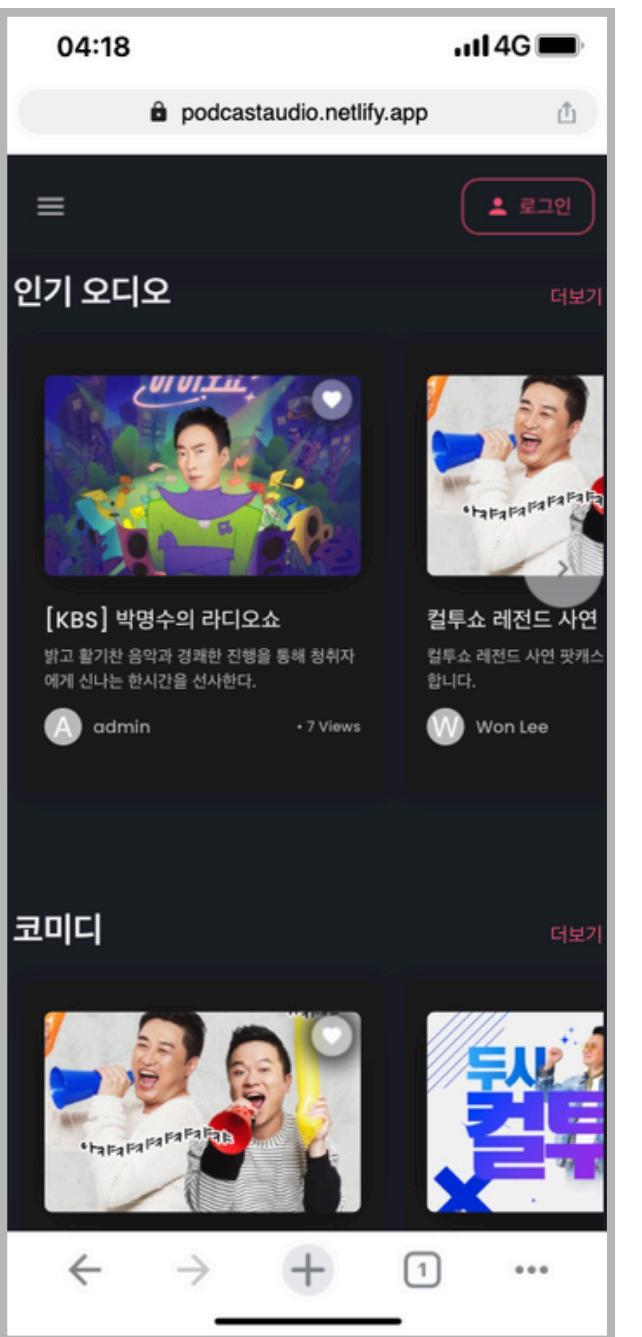
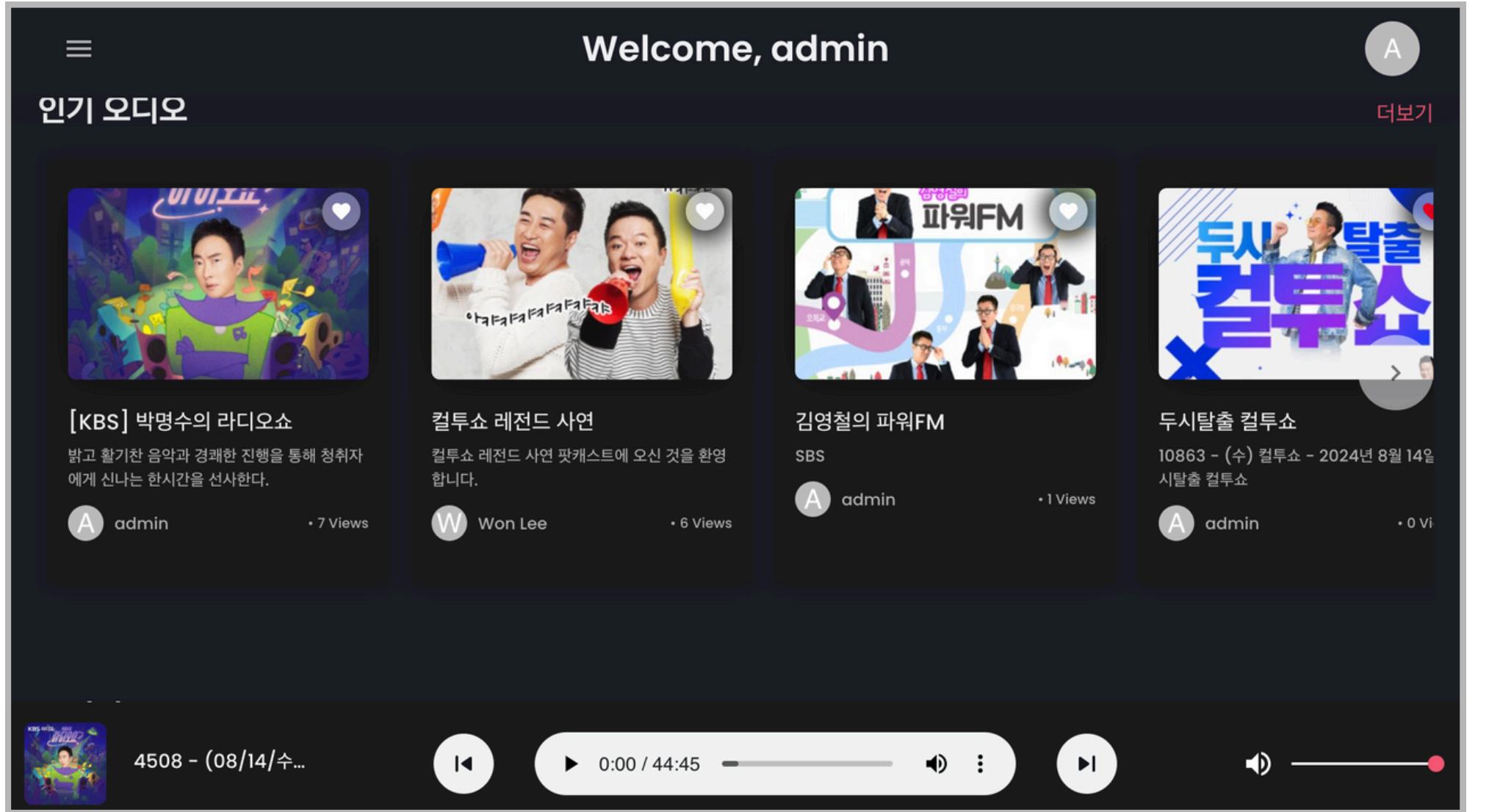
- 팟캐스트 즐겨찾기 및 재생목록 관리 기능
- 시청/청취 기록 기반 개인화된 콘텐츠 추천
- 실시간 조회수 집계 및 통계 시스템
- 카테고리/태그 기반 검색 및 필터링 기능

- 성능 최적화

- 지역 로딩을 통한 초기 로딩 시간 최적화
- React.memo를 활용한 불필요한 리렌더링 방지
- 컴포넌트 메모이제이션을 통한 렌더링 성능 개선
- REST API 최적화로 서버 응답 시간 단축
- 모바일 환경에서의 성능 최적화
- Redux Toolkit을 활용한 효율적인 상태 관리
- 여러 사용자 간의 실시간 데이터 동기화

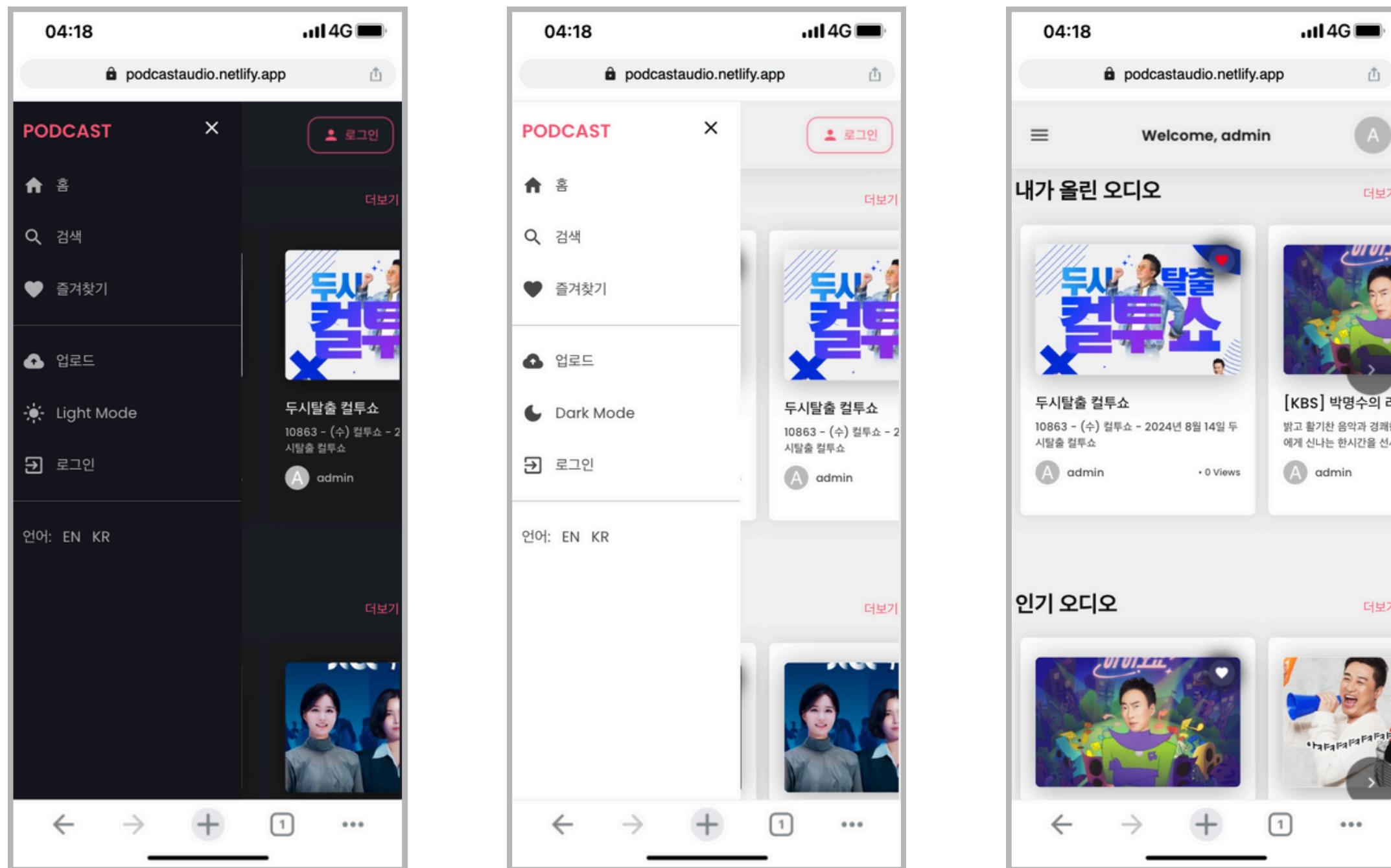
02 PodStream - 팟캐스트 라디오 플랫폼

홈페이지 및 인기 오디오, 카테고리별 콘텐츠 분류, 멀티미디어 스트리밍 시스템



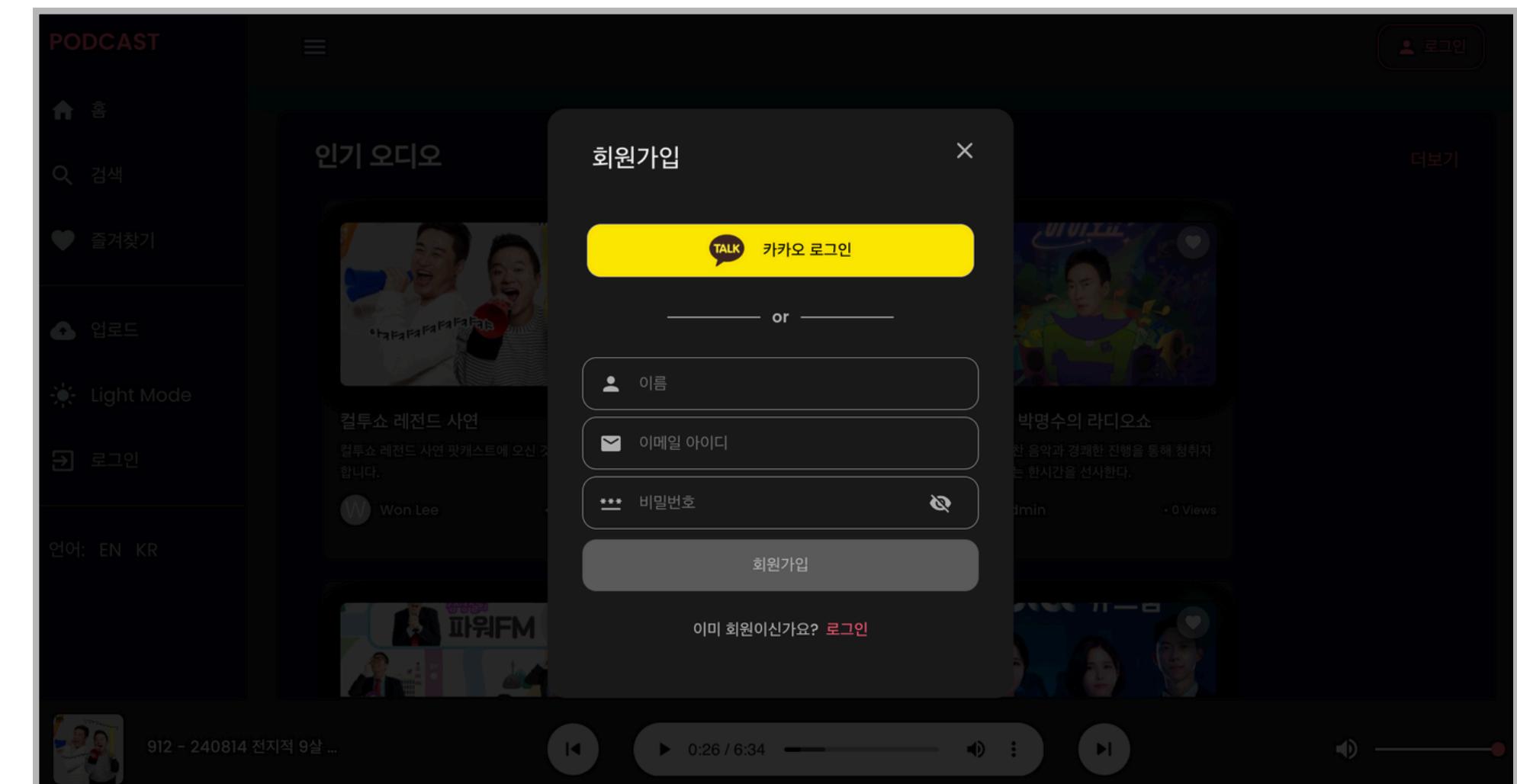
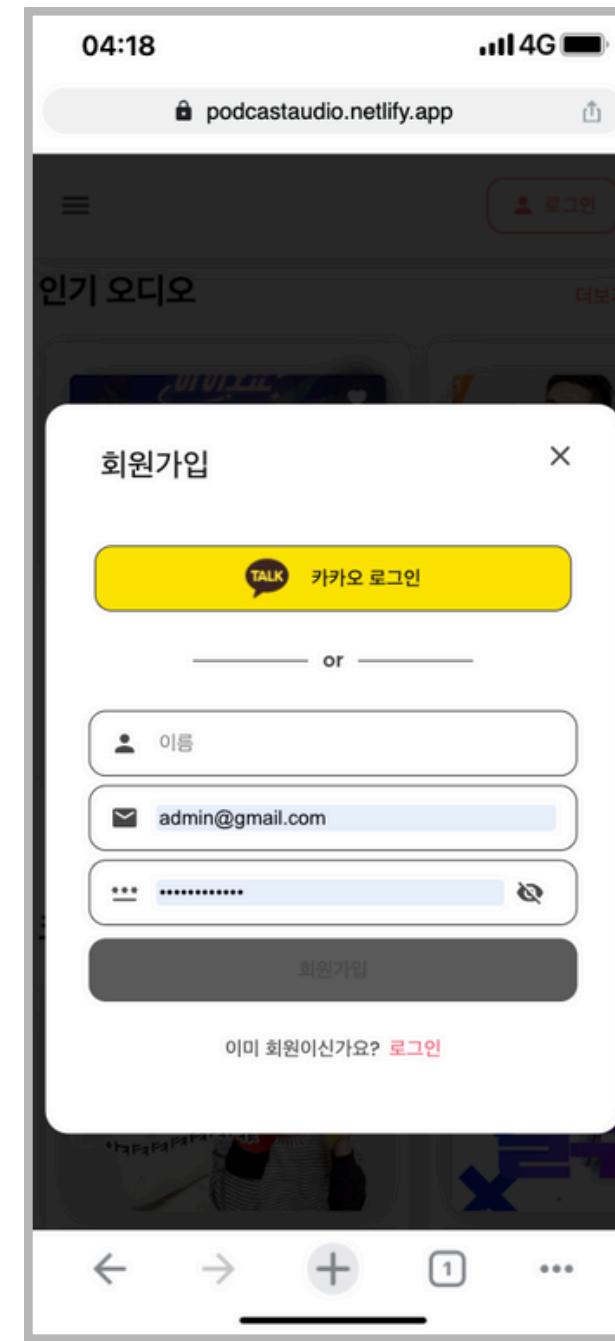
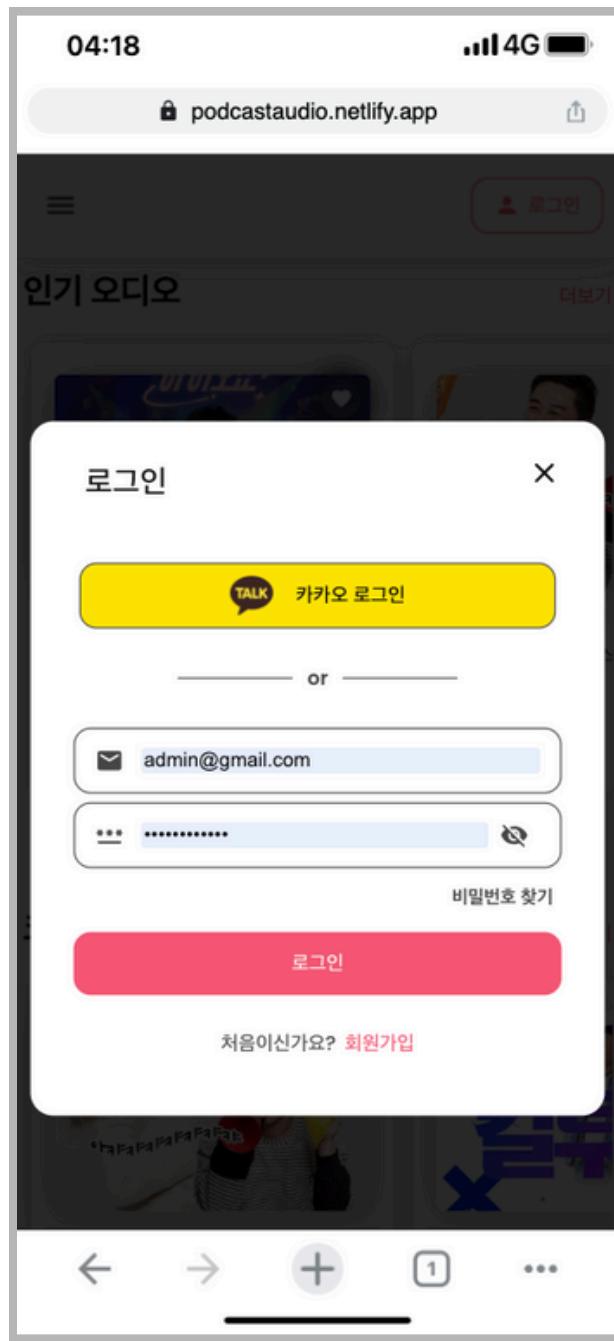
02 PodStream - 팟캐스트 라디오 플랫폼

라이트 & 다크 모드, 한국어/영어 다국어



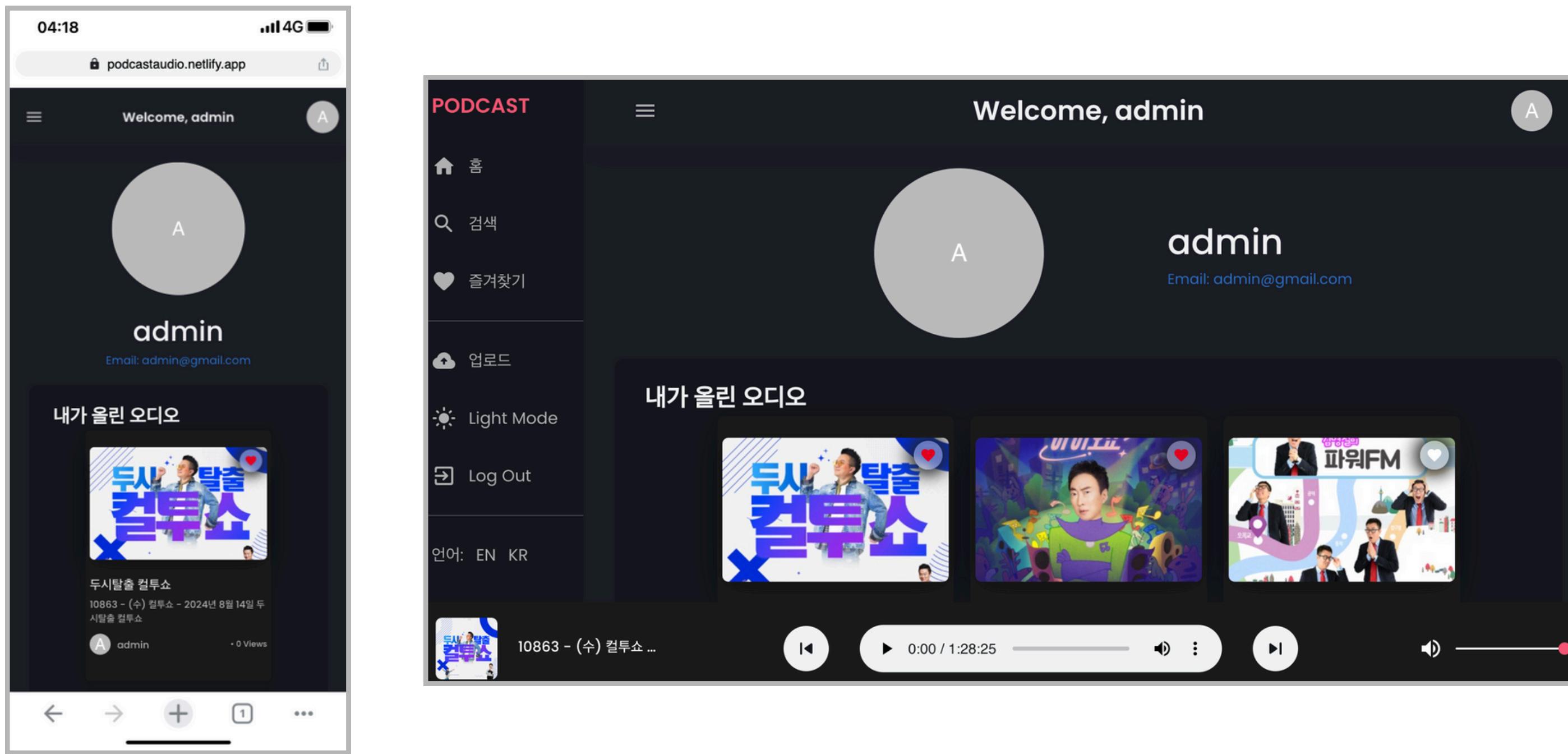
02 PodStream - 팟캐스트 라디오 플랫폼

회원가입 & 로그인



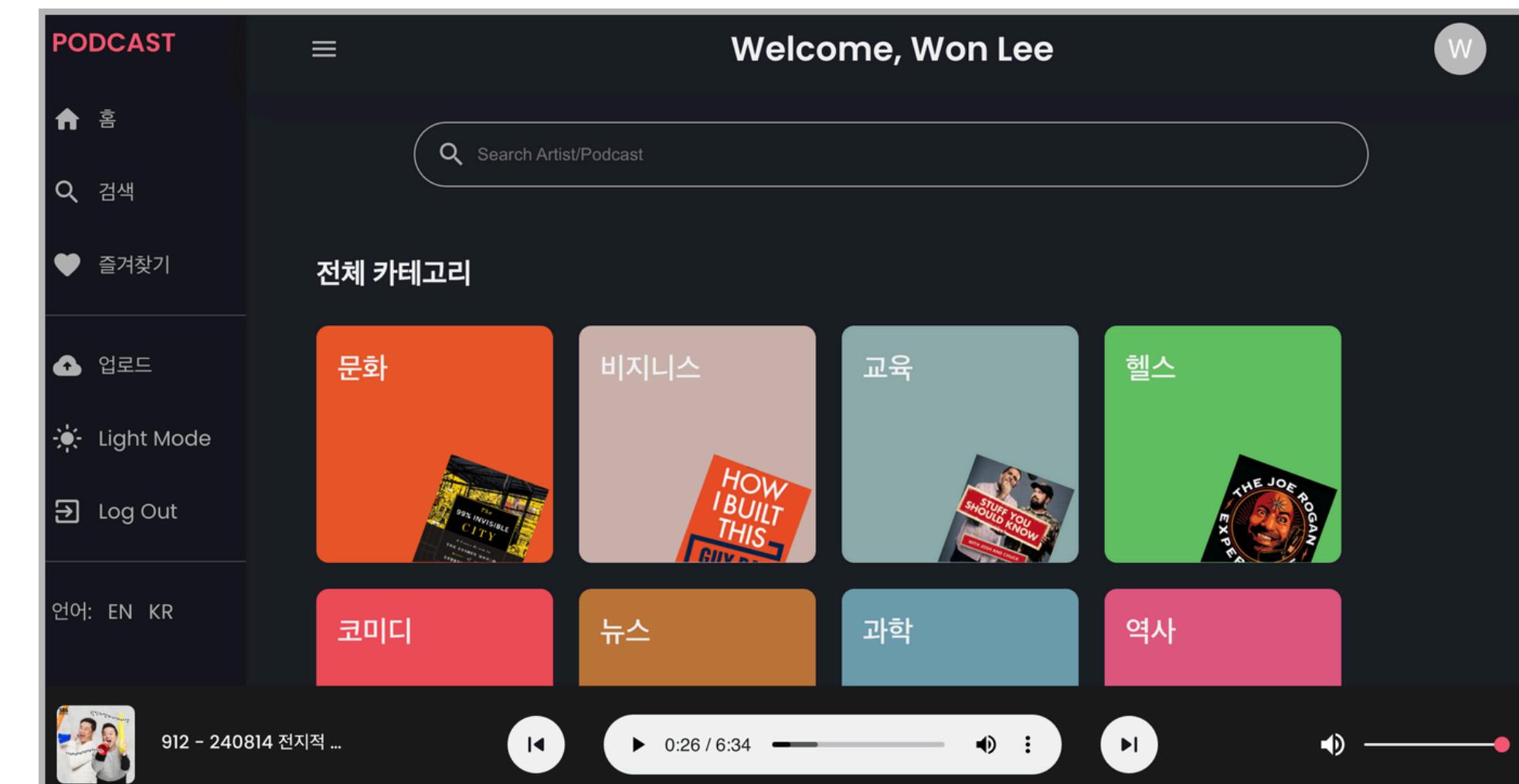
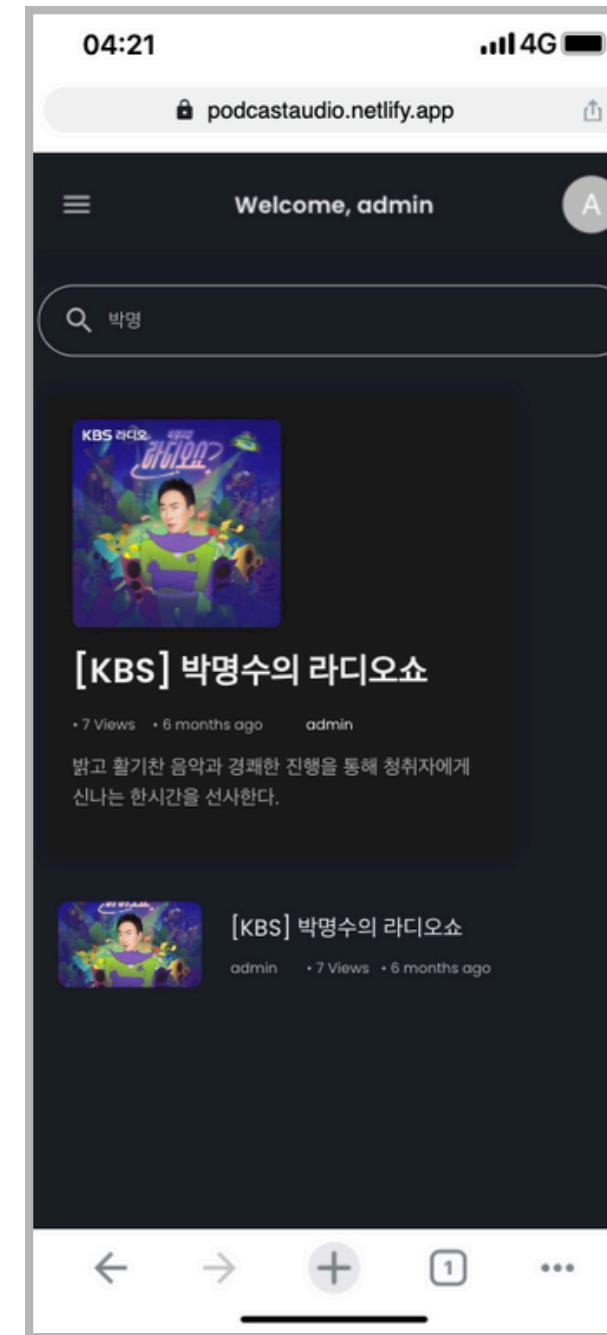
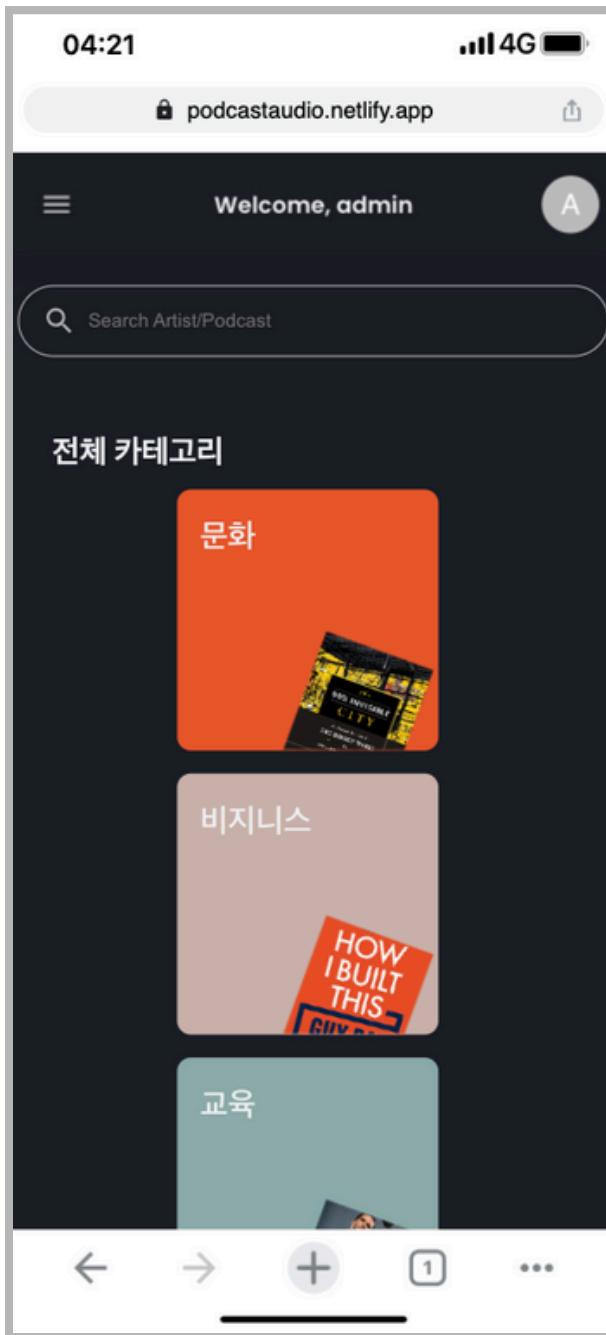
02 PodStream - 팟캐스트 라디오 플랫폼

프로필 페이지, 팟캐스트 업로드 및 관리



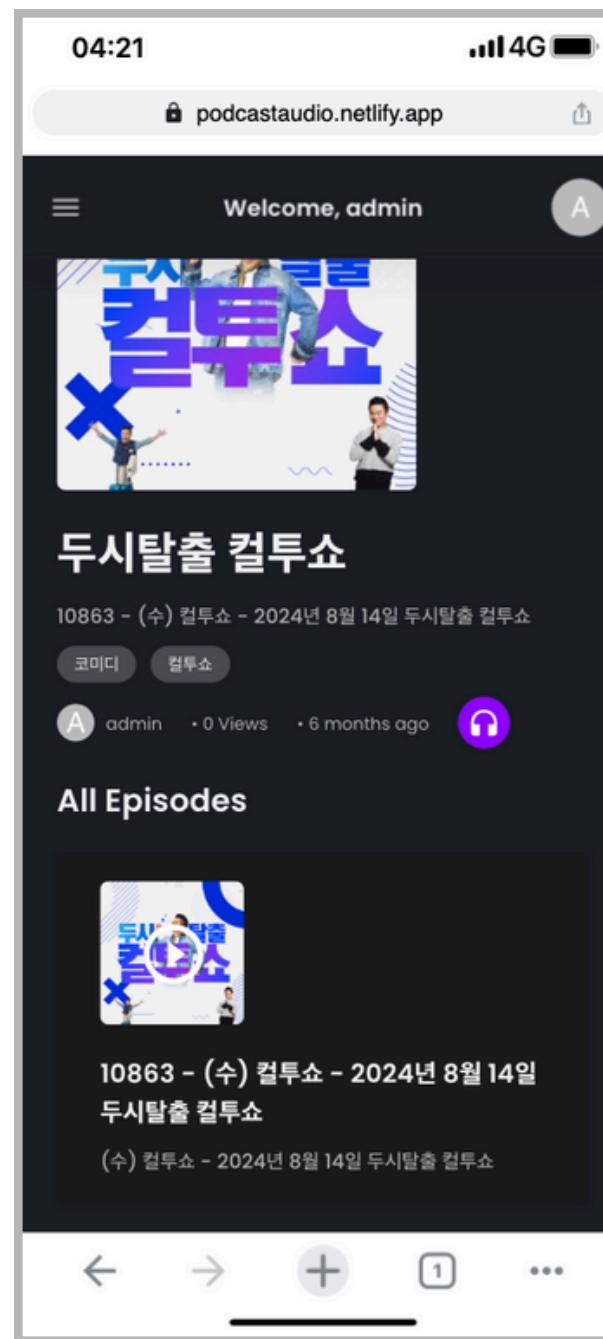
02 PodStream - 팟캐스트 라디오 플랫폼

검색 & 카테고리 페이지



02 PodStream - 팟캐스트 라디오 플랫폼

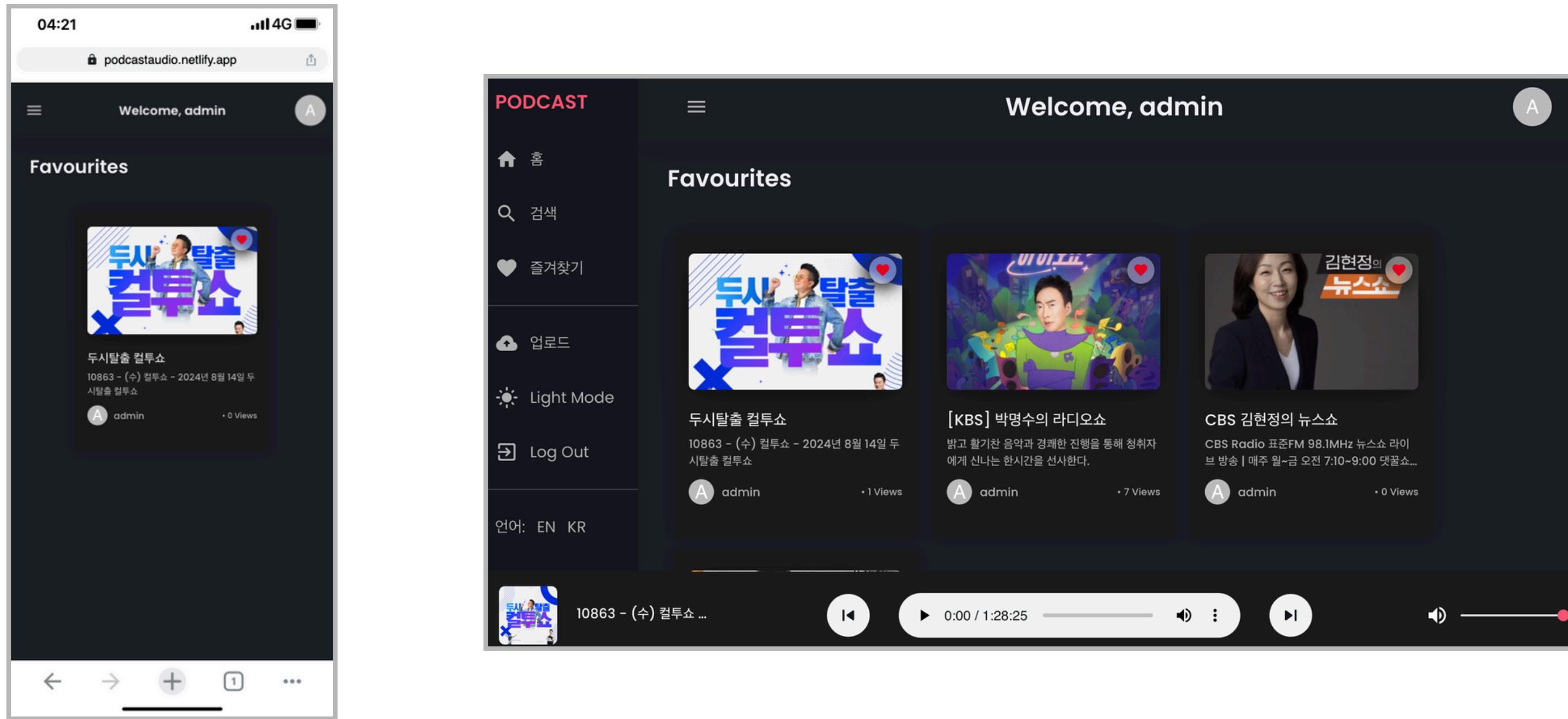
라디오 상세 및 에피소드 페이지



A web browser screenshot of the PodStream platform. The left sidebar has a 'PODCAST' header and links for 'Home', 'Search', 'Favorites', 'Upload', 'Dark Mode', and 'Login'. It also shows language settings '언어: EN KR'. The main content area displays the '두시탈출 컬투쇼' episode with the same details as the mobile app. Below it, a 'All Episodes' section shows the same episode card. At the bottom, there is a media player interface with a play button, a progress bar showing 0:26 / 6:34, and other control icons.

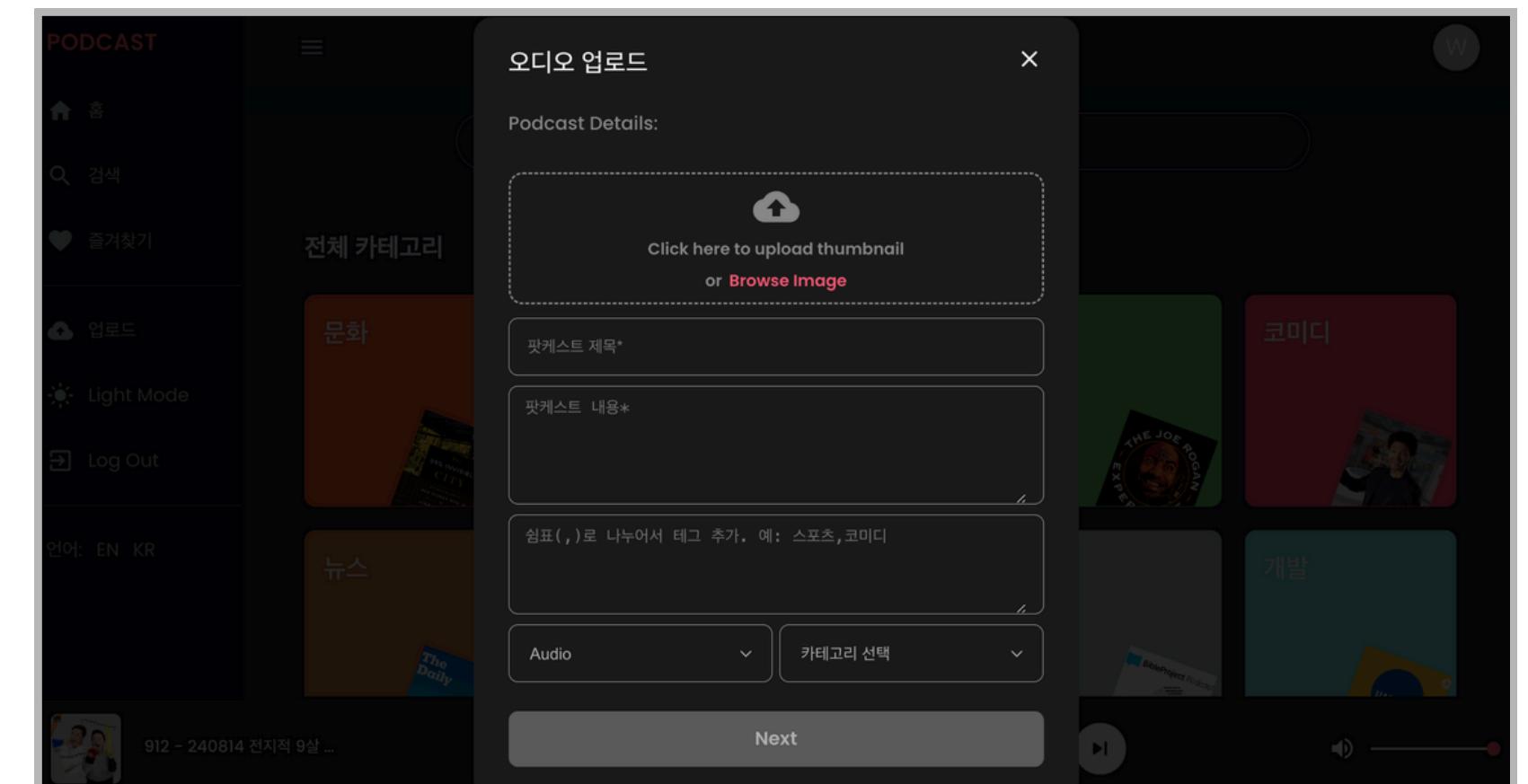
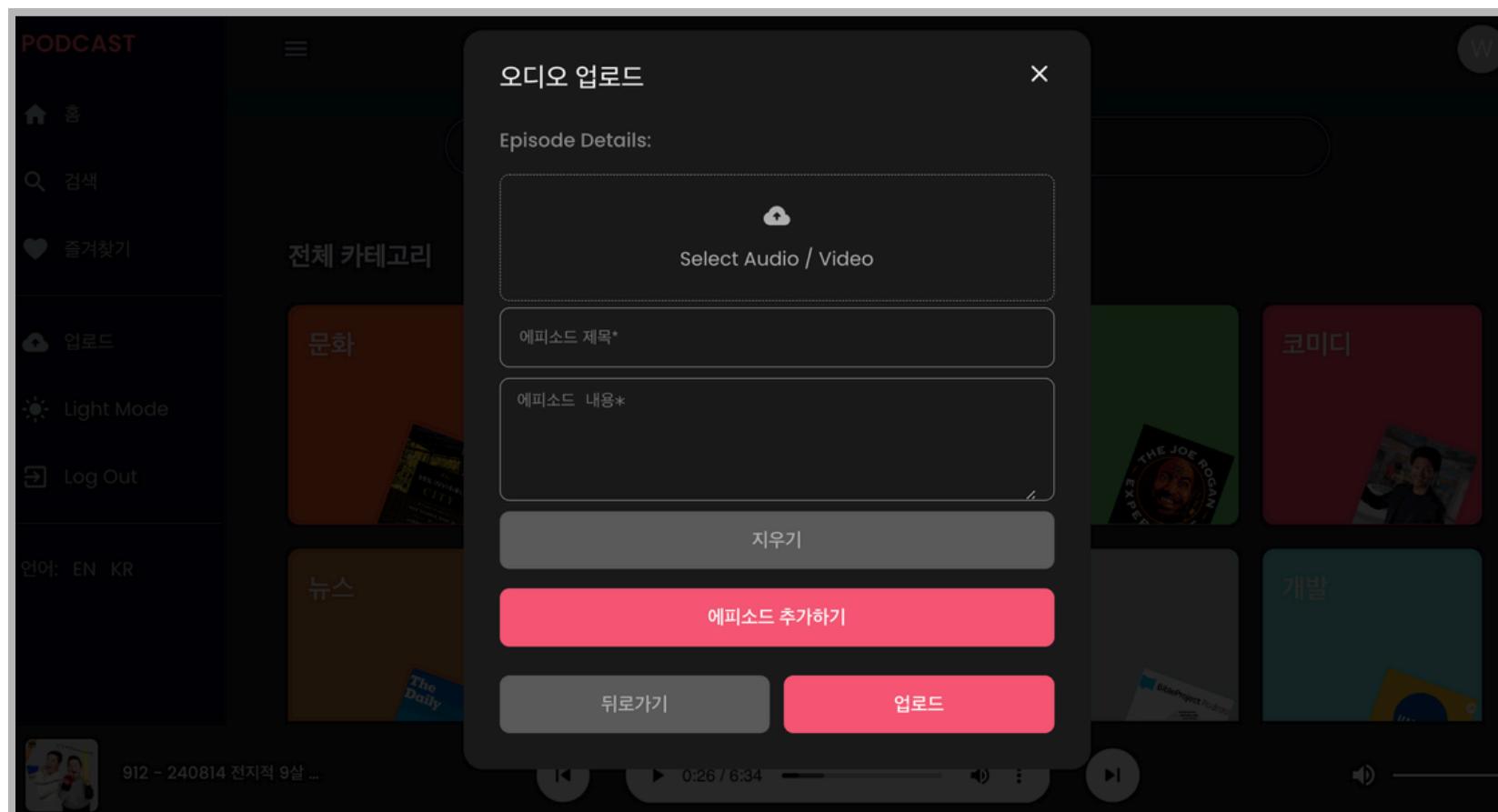
02 PodStream - 팟캐스트 라디오 플랫폼

즐겨찾기



02 PodStream - 팟캐스트 라디오 플랫폼

오디오 업로드



03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스 소개

레스토랑 업계가 직면한 디지털 전환의 어려움을 해결하고자 통합 마케팅 플랫폼을 개발하였습니다. 현장에서 식당 운영자들과 대화를 나누며, 고객과의 소통, 마케팅, 로열티 프로그램 운영 등에서 겪는 어려움을 직접 확인할 수 있었습니다. 특히 디지털 도구의 부재로 인한 비효율적인 운영과 고객 관리의 한계를 극복하고자, 식당 운영자와 고객 모두에게 도움이 되는 솔루션을 만들어보고자 했습니다.

개발 기간 기획 및 설계, 디자인: 2024.07.23 ~ 2024.08.13

 개발: 2024.08.14 ~ 2024.12.11

 테스트 및 배포: 2024.12.12 ~ 2024.12.26

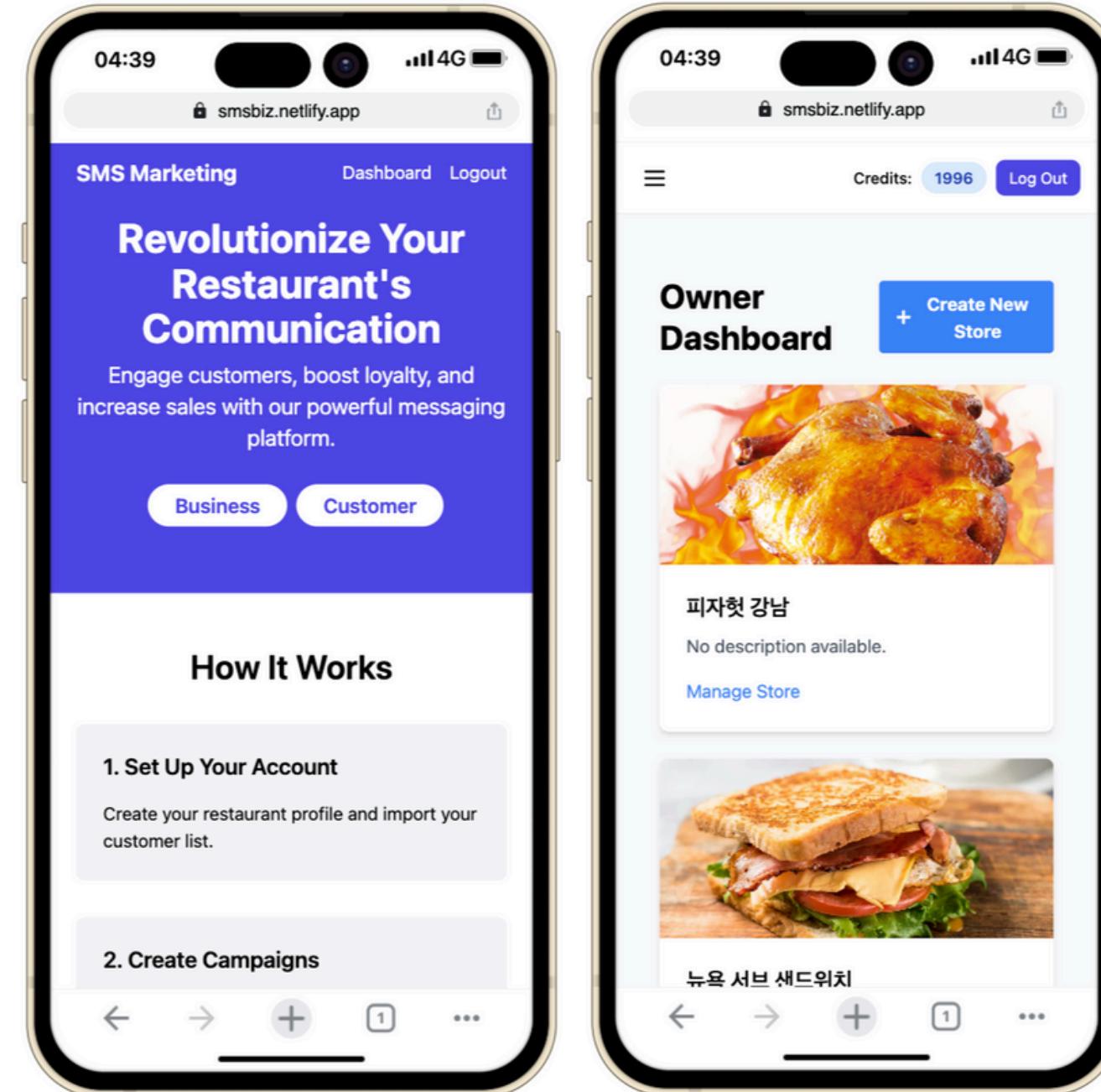
개발인원 2명 (PM 및 디자인, 개발)

담당역할 PM, 프론트엔드, 백엔드

- 서비스 기획 및 방향성 설정
- 메인 로직 구현 (프론트엔드, 백엔드), API design, DB 설계

깃허브 <https://github.com/whd793/sms-startup>

Demo <https://smsbiz.netlify.app/>



03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

개발 환경

Frontend:

- 핵심 프레임워크: React 18
- 상태 관리: React Query, Context API
- 라우팅 관리: React Router
- API 통신: Axios
- 스타일링: Tailwind CSS, Framer Motion
- 폼 관리: React Hook Form, Yup
- HTTP 클라이언트: Axios
- 결제 모듈: Stripe Elements

Backend:

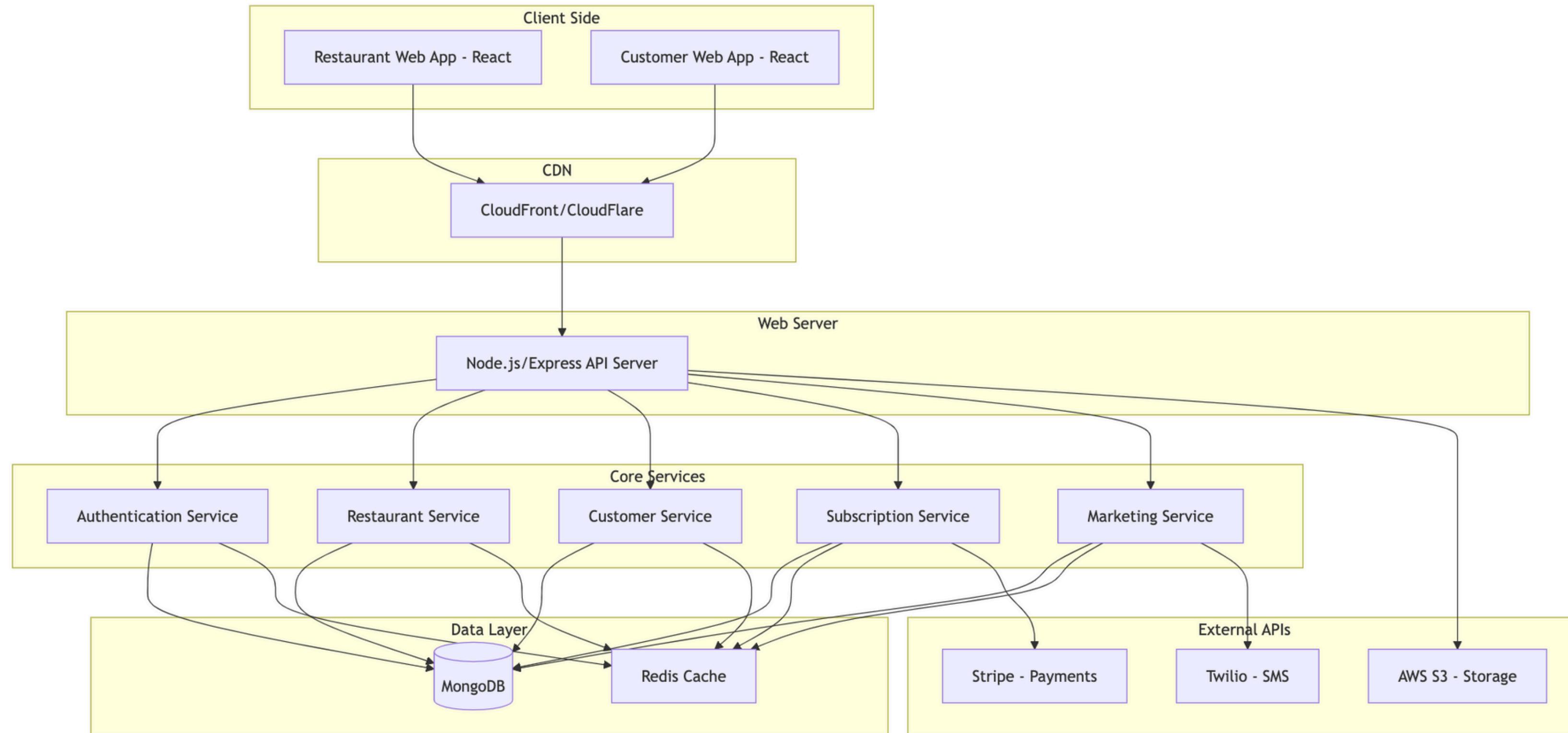
- 런타임: Node.js 18
- 프레임워크: Express.js
- 데이터베이스: MongoDB/Mongoose
- 캐싱: Redis
- 인증: JWT Authentication, bcrypt
- 파일 저장소: AWS S3
- SMS 서비스: Twilio API
- REST API: RESTful 아키텍처
- Stripe API: 결제 시스템

Development Tools:

- 버전 관리: Git, GitHub
- 코드 품질 관리: ESLint, Prettier
- 테스트: Jest, React Testing Library
- API 테스트: Postman
- 컨테이너화: Docker

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

System Architecture



03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

핵심 기능 (1/4)

점주용 기능 (1/2)

계정 관리

- 회원가입 및 로그인
- 프로필 관리
- 비밀번호 변경
- 계정 삭제 기능

텍스트 메시지 관리

- 마케팅 메시지 작성 및 예약 발송
- 메시지 템플릿 저장
- 이미지 첨부 기능
- 반복 발송 설정 (일간/주간/월간)
- Twilio API 연동
- 예약 발송 시스템
- 대량 메시지 발송/처리
- 발송 상태 추적

스탬프 시스템

- 스탬프 적립 규칙 설정
- 일일 제한 및 쿠다운 시스템
- 리워드 교환 시스템
- 고객별 스팸프 현황 확인
- NFC 태그 기반 스팸프 적립
- 만료일 관리

사용자 인증 시스템

- JWT 기반 멀티 유저 인증 시스템 설계 및 구현
- OTP 기반 휴대폰 인증 시스템 개발
- 권한 기반 접근 제어 시스템 구축
- 세션 관리 및 토큰 검증 로직 구현

구독자 관리

- 구독자 목록 확인
- 구독자 통계
- 구독 해지 관리

식당 관리

- 식당 정보 등록/수정/삭제
- 식당 프로필 이미지 업로드
- QR 코드 자동 생성
- 영업 시간 및 위치 정보 관리

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

핵심 기능 (2/4)

점주용 기능 (2/2)

쿠폰 시스템

- 쿠폰 생성 및 관리
- QR 코드 기반 쿠폰 인증
- 쿠폰 만료일 관리 로직 구현
- 사용 현황 추적
- 자동 만료 설정
- 사용 제한 설정
- 할인율/금액 설정
- 타겟팅 옵션

검색 및 필터링

- 고급 검색 기능
- 다중 필터링 옵션
- 정렬 기능
- 실시간 검색 결과

이미지 처리 시스템

- AWS S3 기반 이미지 업로드
- 이미지 보안 검증
- 파일 관리 시스템
- 동적 이미지 URL 처리

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

핵심 기능 (3/4)

고객용 기능

계정 관리

- 휴대폰 번호 기반 회원가입
- OTP 인증
- 프로필 설정
- 계정 설정 관리

식당 구독

- QR 코드 스캔으로 구독
- 구독 중인 식당 목록 확인
- 구독 취소 기능

플레이리스트 기능

- 즐겨찾기
- 좋아하는 식당 저장
- 커스텀 플레이리스트 생성
- 플레이리스트 관리

쿠폰 관리

- 발급받은 쿠폰 목록 확인
- 쿠폰 사용 및 관리
- 만료 예정 쿠폰 확인

스탬프 적립

- NFC 태그/QR 코드로 스팩프 적립
- 적립 현황 확인
- 리워드 교환

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

핵심 기능 (4/4)

Stripe API를 활용한 구독 시스템을 개발했습니다. 사용자가 다양한 구독 플랜을 선택하고 관리할 수 있으며, 신용카드 결제, 플랜 업그레이드/다운그레이드, 크레딧 구매 등의 기능을 제공합니다.

구독 시스템 아키텍처 설계 및 구현

- 무료/유료 구독 플랜 관리 시스템 구축
- 플랜 변경 시 자동 비례 배분(proration) 시스템 개발
- 구독 갱신, 업그레이드, 다운그레이드 로직 구현
- 무료 플랜 및 3가지 유료 구독 플랜 (Basic, Pro, Premium) 구현
- 플랜 업그레이드시 즉시 반영 및 차액 정산
- 다운그레이드시 현재 구독 기간 종료 후 자동 변경
- 구독 취소 및 재활성화 기능

크레딧 시스템 개발

- 크레딧 시스템(구독/구매) 설계 및 구현
- 자동 크레딧 갱신 및 이월 시스템 개발 (Webhook)
- 크레딧 사용 우선순위 로직 구현
- 구독 플랜별 기본 크레딧 제공
- 구독 크레딧과 구매 크레딧의 분리 관리

결제 시스템 통합

- Stripe Elements를 활용한 PCI 준수 결제 프로세스 구현 및 안전한 카드 결제 시스템
- 다중 결제 수단 관리 기능 개발
- 자동 결제 및 실패 복구 시스템 구축
- 기본 결제 수단 설정 기능
- 결제 내역 조회 및 영수증/인보이스 다운로드
- 결제 실패 자동 복구 시스템
- 환불 처리 자동화
- 결제 데이터 동기화 시스템
- 다중 통화 지원 시스템

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

기술적 문제 해결

성능 최적화 및 보안 강화 (1/4)

- React 컴포넌트 메모이제이션을 통한 렌더링 최적화
- 사용자 인터페이스의 성능 최적화를 위해 React.memo, useCallback, useMemo 등의 메모이제이션 기법을 적극 활용
- 커스텀 훅을 개발하여 비즈니스 로직의 재사용성을 높였습니다. 예를 들어, useCustomerAuthCheck라는 커스텀 훅을 개발하여 사용자 인증 상태 관리와 토큰 검증을 효율적으로 처리
- 상태 관리 측면에서는 Context API를 활용하여 전역 상태를 관리했으며, 특히 인증 상태와 같은 중요 데이터의 일관성을 유지하기 위해 노력했습니다.
- 로딩 상태, 에러 처리, 성공/실패 메시지 등을 toast 알림으로 구현하여 사용자에게 즉각적인 피드백을 제공

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

기술적 문제 해결

성능 최적화 및 보안 강화 (2/4)

- MongoDB 트랜잭션 처리로 데이터 일관성 보장
- 결제 실패 시 자동 복구 메커니즘 구현
- 데이터베이스 설계에서는 MongoDB를 사용하여 유연하고 확장 가능한 스키마를 구현했습니다. 특히 매장, 고객, 쿠폰, 스탬프 등 복잡한 관계를 가진 데이터 모델을 효율적으로 설계했으며, 인덱싱을 통해 쿼리 성능을 최적화했습니다. 또한 트랜잭션을 활용하여 데이터 일관성을 보장하고, 동시성 문제를 해결했습니다.
- 보안 측면에서는 JWT를 활용한 토큰 기반 인증 시스템을 구현했으며, bcrypt를 사용하여 비밀번호를 안전하게 해시화했습니다. 특히 사용자 인증 과정에서 OTP 시스템을 구현하여 이중 인증을 제공했으며, rate limiting을 적용하여 무차별 공격을 방지했습니다.
- 문자 메시지 발송 시스템은 Twilio API를 활용하여 구현했으며, 특히 대량 발송 시 발생할 수 있는 성능 문제를 해결하기 위해 큐 시스템을 도입했습니다. 또한 예약 발송 기능을 구현하여 매장 오너가 효율적으로 마케팅을 진행할 수 있도록 했습니다.
- 파일 업로드 시스템에서는 multer를 사용하여 이미지 업로드를 구현했으며, sharp 라이브러리를 활용하여 이미지 최적화를 진행했습니다. 특히 프로필 이미지나 쿠폰 이미지와 같은 미디어 파일의 효율적인 저장과 관리를 위해 AWS S3를 활용했습니다.
- 성능 최적화 측면에서는 데이터베이스 쿼리 최적화, 캐싱 전략 수립, 페이지네이션 구현 등을 통해 대규모 데이터 처리 시에도 빠른 응답 속도를 보장했습니다. 특히 Redis를 활용하여 자주 접근하는 데이터를 캐싱하고, 세션 관리를 효율적으로 수행했습니다.

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

기술적 문제 해결

성능 최적화 및 보안 강화 (3/4)

동시성 제어 문제 해결

결제 처리 과정에서 발생할 수 있는 동시성 문제를 해결하기 위해 다음과 같은 방법을 적용

- 락 메커니즘 구현으로 동시 접근 제어
 - 결제 처리 시 고유한 idempotency key 사용
 - 데이터베이스 레벨의 트랜잭션 락 적용
 - 동시 요청에 대한 대기열 처리
- 중복 결제 방지
- 분산 락 시스템 구현

데이터 정합성 보장

구독 상태 변경과 결제 처리가 동시에 일어날 때의 데이터 정합성을 보장하기 위해:

- 트랜잭션 처리

트랜잭션 정합성 문제 해결

구독 플랜 변경 시 결제와 데이터 업데이트의 정합성 문제가 발생했습니다. 이를 해결하기 위해:

- MongoDB 트랜잭션 도입
- 결제 실패 시 자동 롤백 메커니즘 구현
- 면등성을 보장하는 API 설계

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

기술적 문제 해결

성능 최적화 및 보안 강화 (4/4)

대량의 쿠폰과 메시지 데이터 처리 시 성능 저하 & 실시간 데이터 동기화로 인한 서버 부하

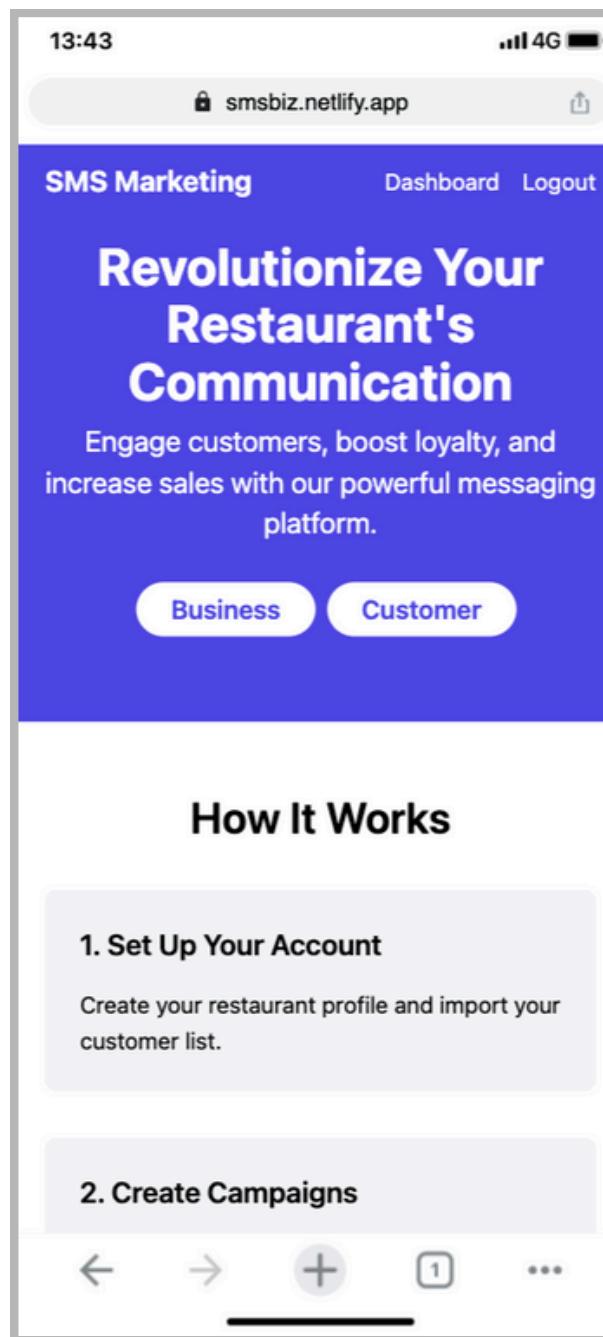
- Redis 캐싱 도입으로 빈번한 조회 데이터 최적화
- 데이터베이스 인덱싱 전략 수립
- React 컴포넌트 메모이제이션으로 불필요한 리렌더링 방지
- 페이지네이션 구현으로 대량 데이터 처리 최적화

결제 정보 보안 & PCI DSS 컴플라이언스 준수 & 결제 실패 시 복구 처리

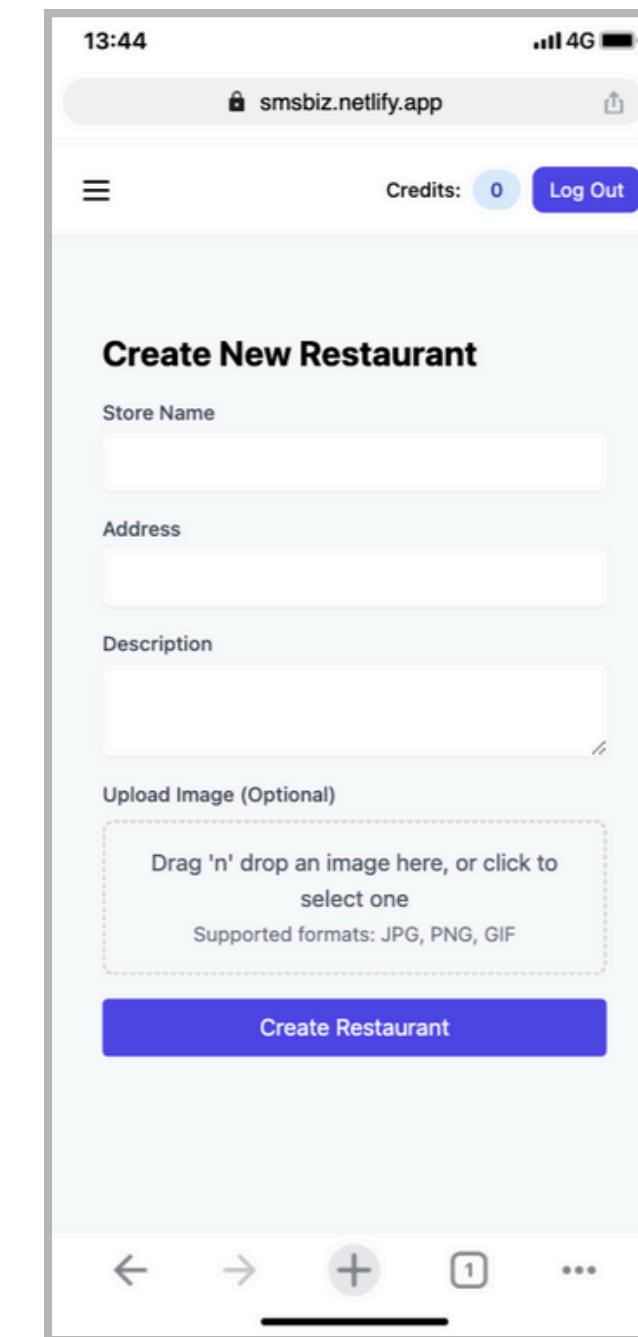
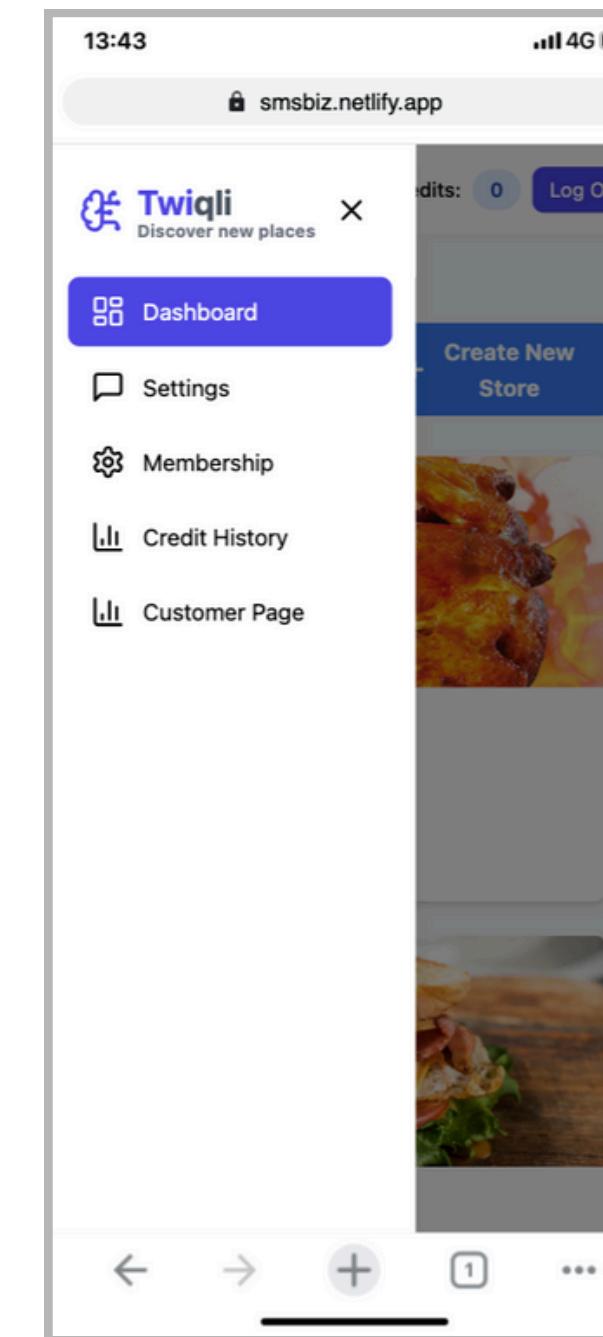
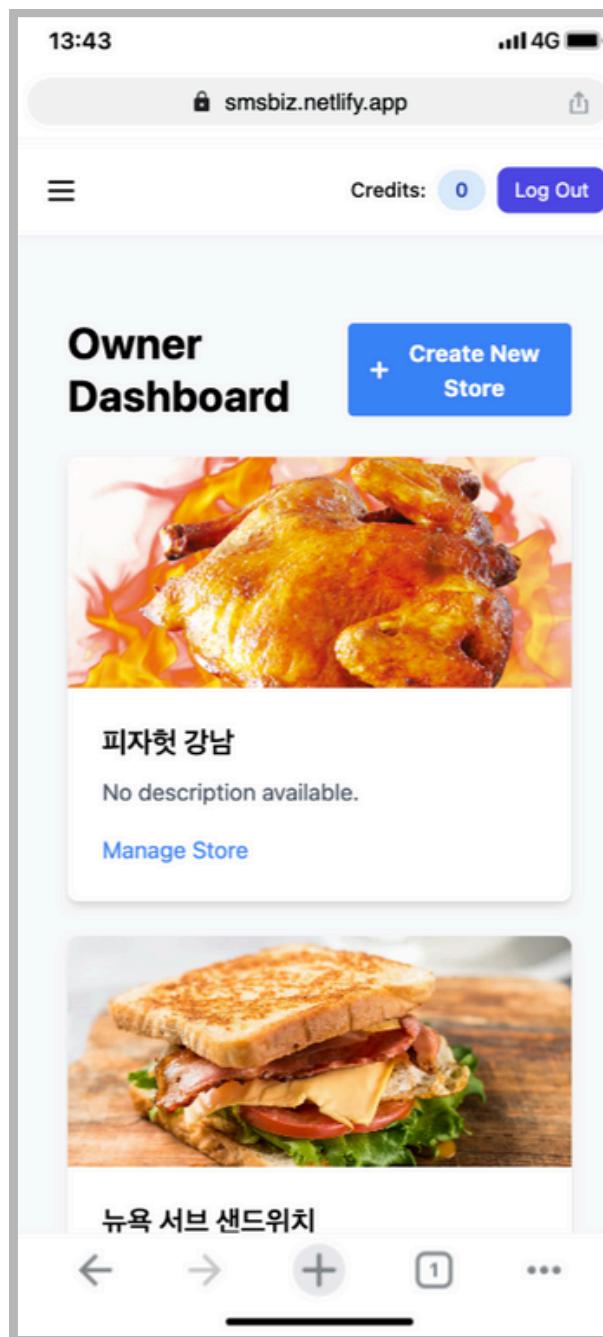
- Stripe Elements 도입으로 카드 정보 직접 처리 회피
- 토큰 기반 결제 처리 구현
- 자동 복구 메커니즘 구현으로 결제 실패 대응
- 트랜잭션 롤백 시스템 구축

03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

홈페이지

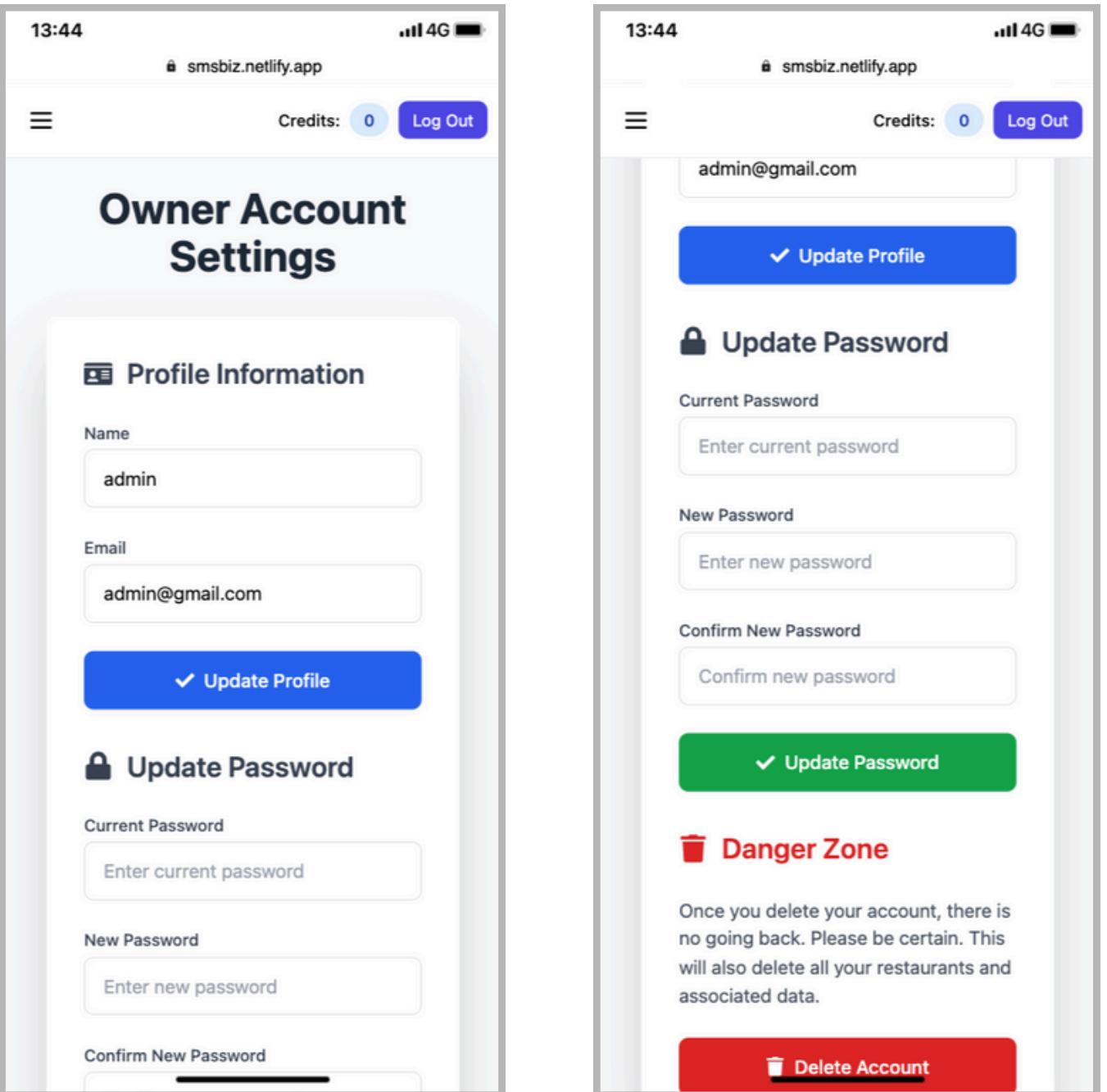


점주용 대시보드, 비즈니스 정보 등록/수정/삭제, 프로필 이미지 업로드



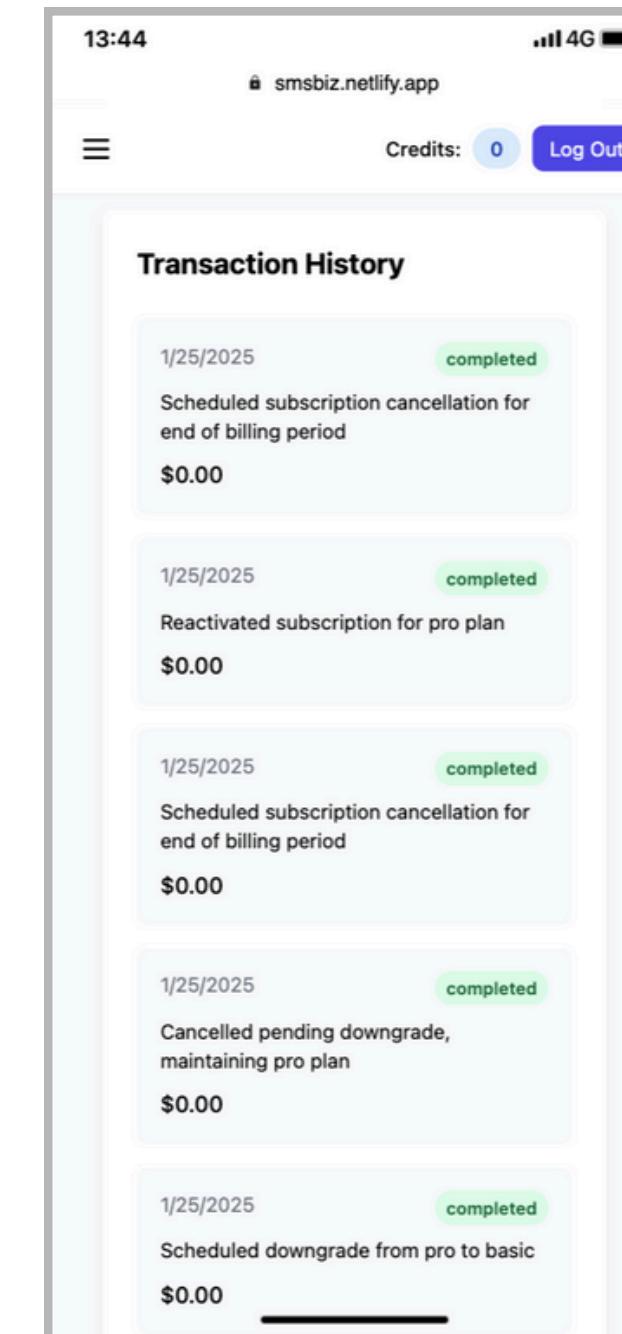
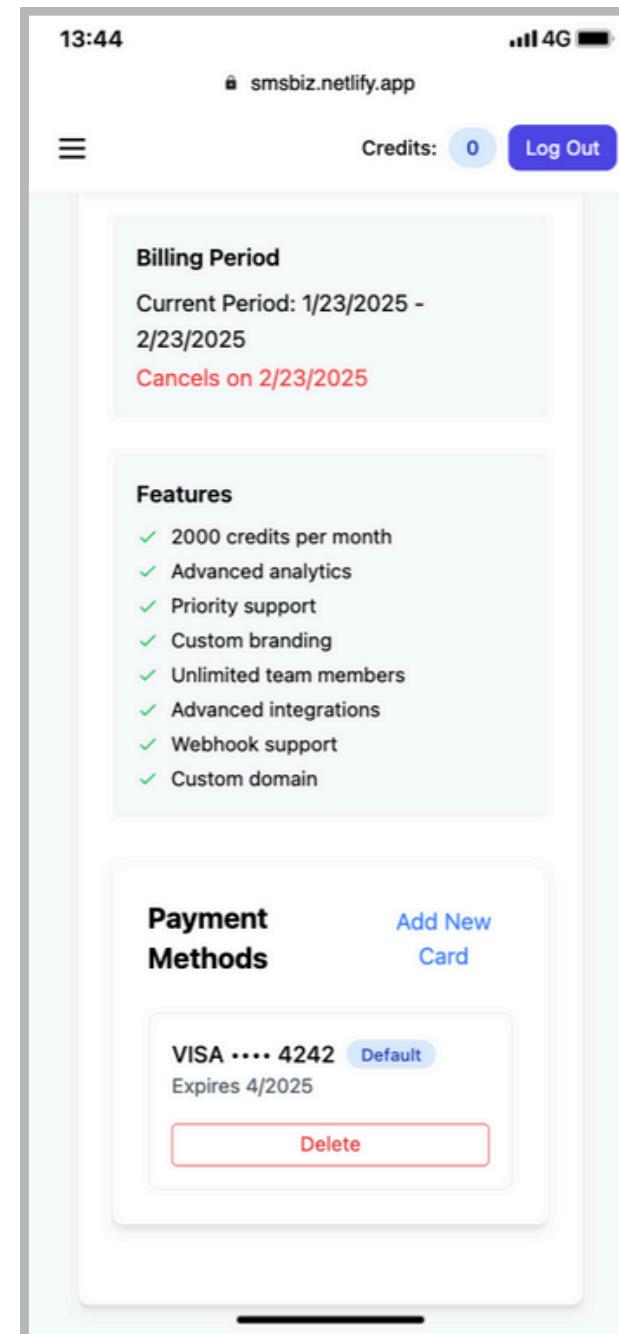
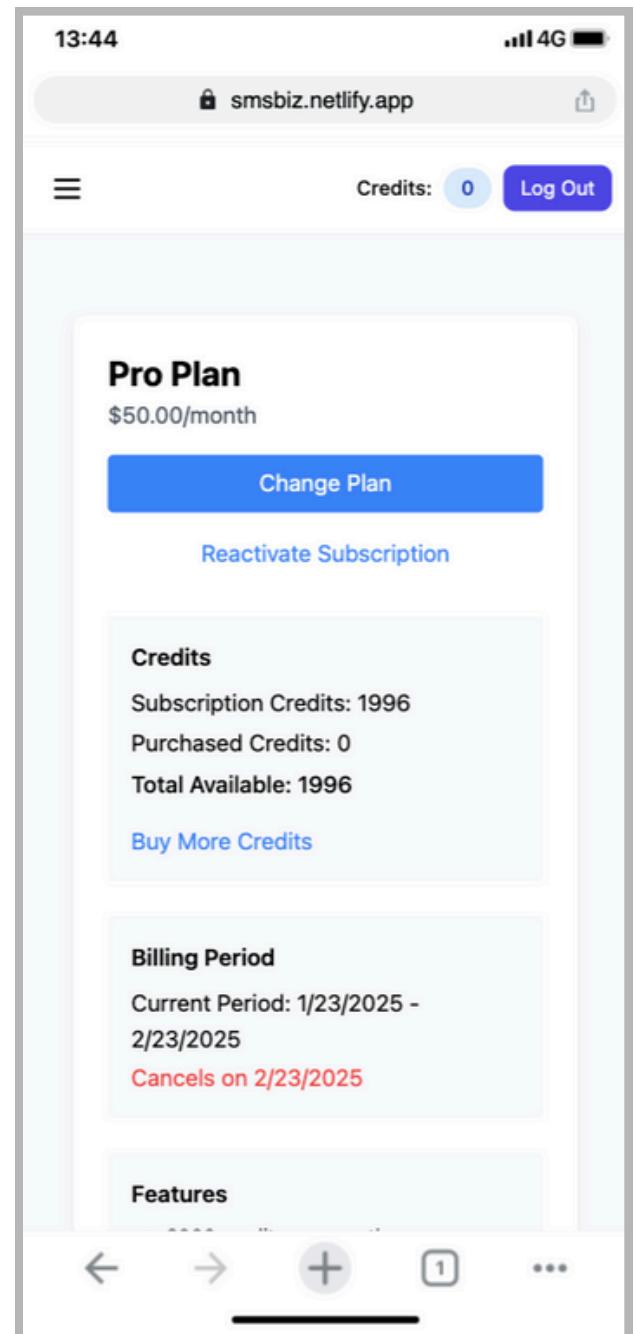
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

점주용 계정 설정 - 프로필 관리, 비밀번호 변경, 계정 삭제 가능

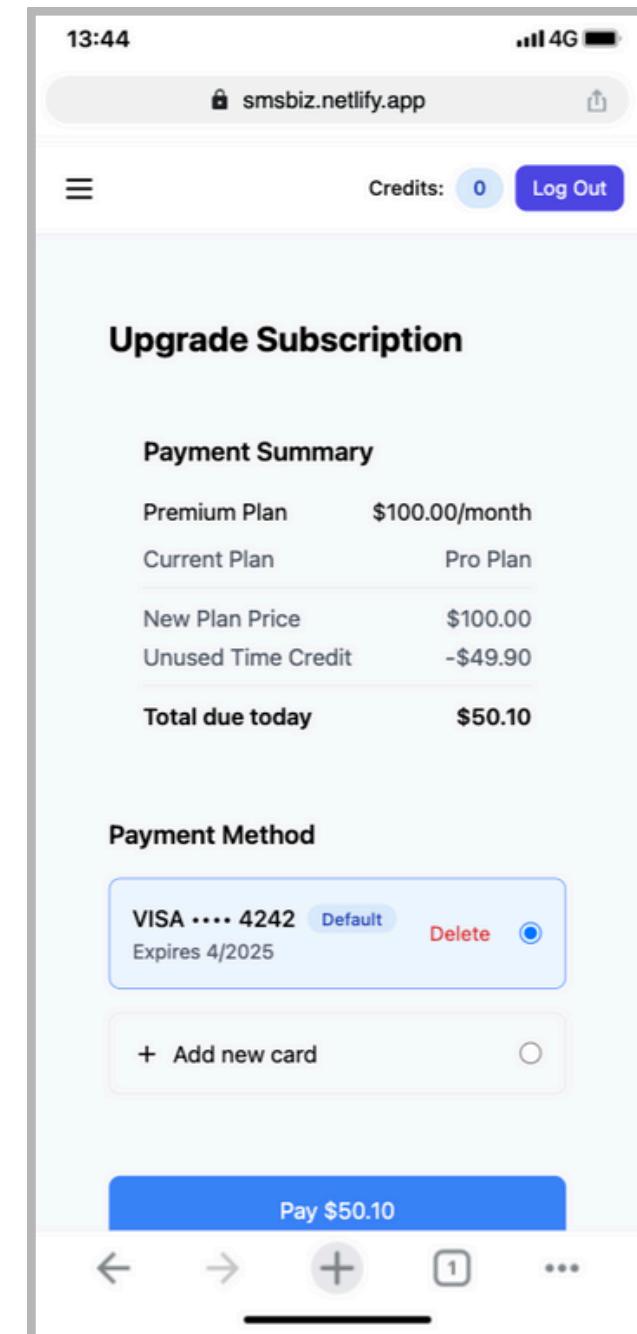
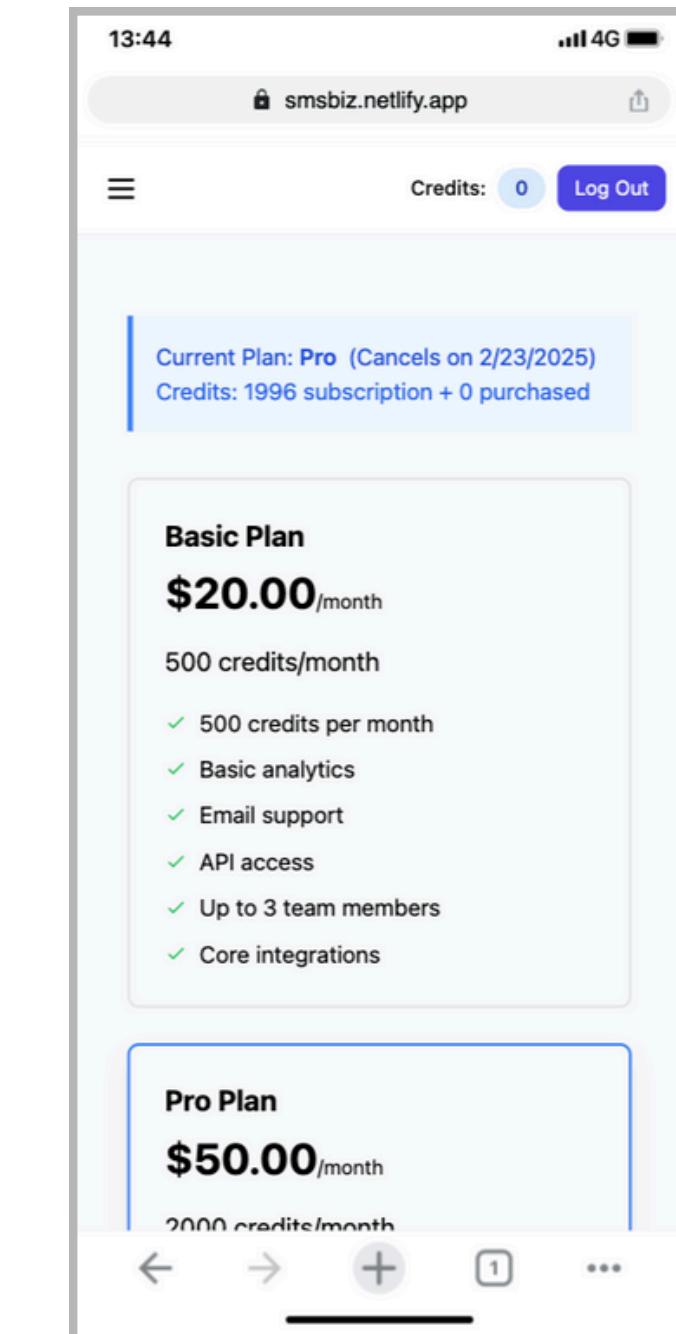


03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

구독 시스템, 무료/유료 플랜, 플랜 변경 및 자동 비례 배분(proration), 플랜 변경 내역



구독 결제 시스템, 다중 결제 수단 관리, 기본 결제 수단 설정



03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

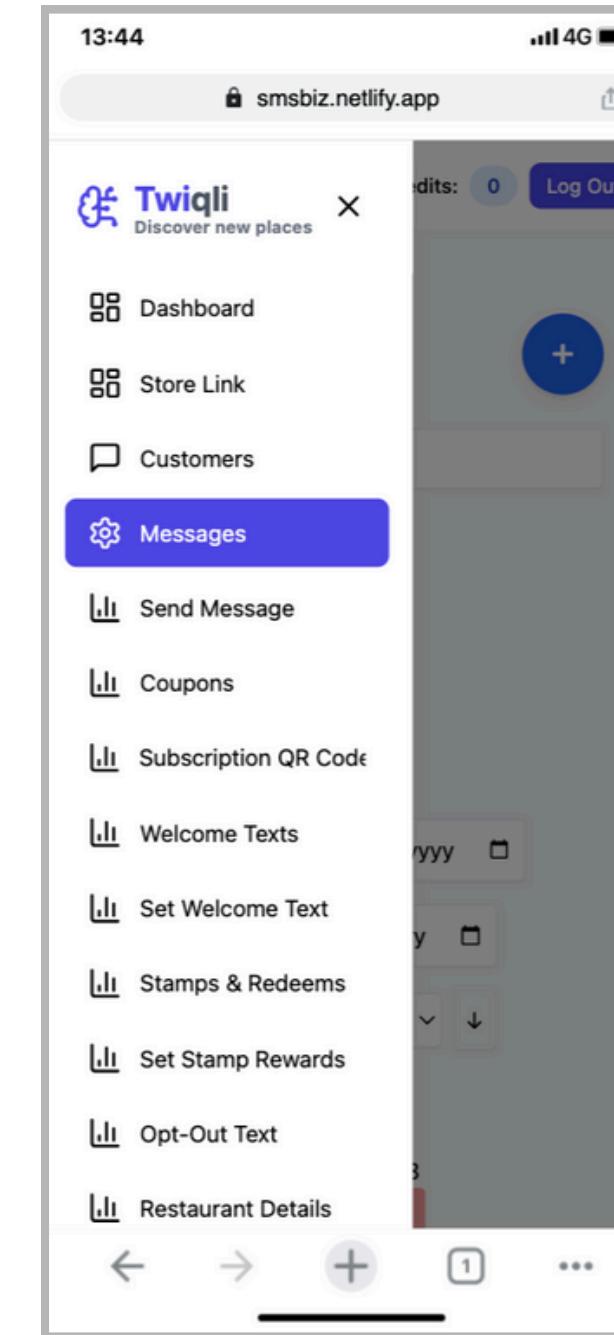
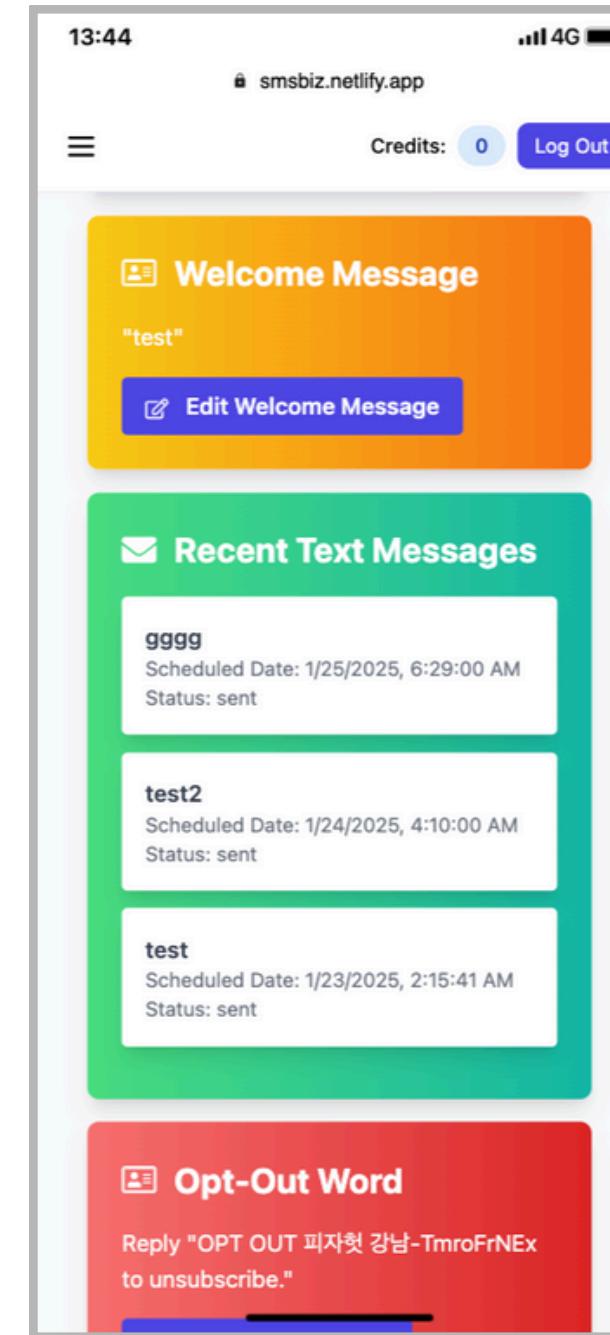
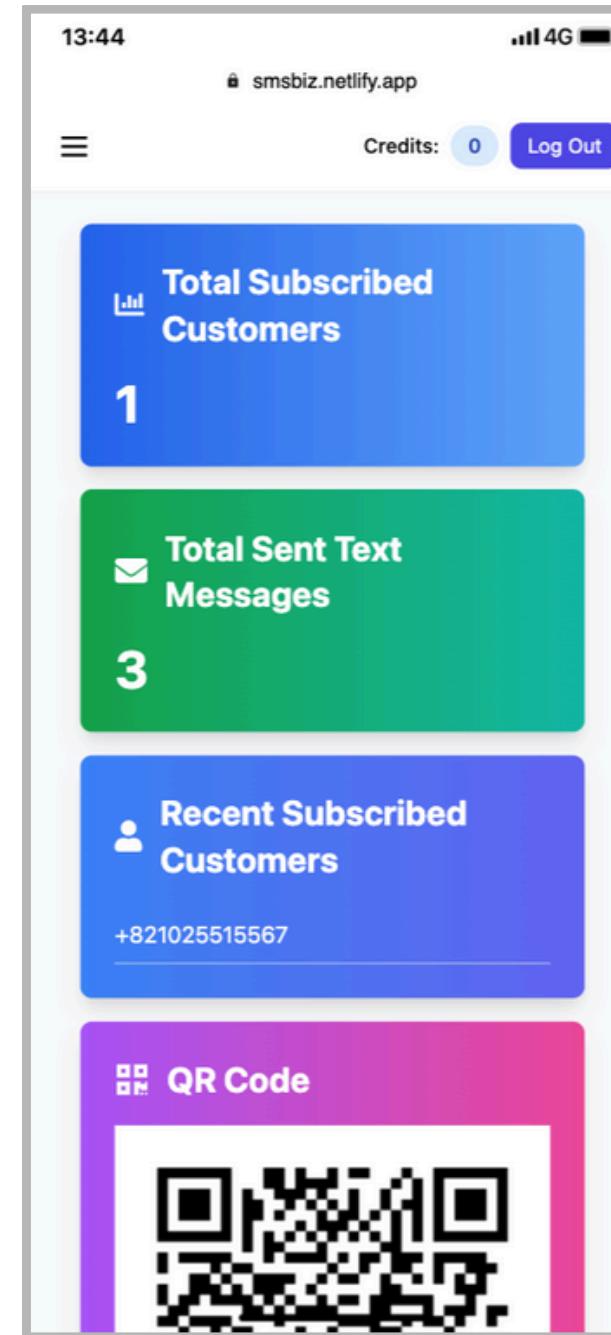
크래딧 사용 내역, 자동 크래딧 갱신

Date	Type
1/25/2025	message_sent
1/24/2025	message_sent
1/24/2025	message_sent
1/24/2025	message_sent

Description	Credits	Balance
Credits used for message: gg...	-1	1996
Credits used for message: koko...	-1	1997
Credits used for message: test2323...	-1	1998
Credits used for message: test2...	-1	1999

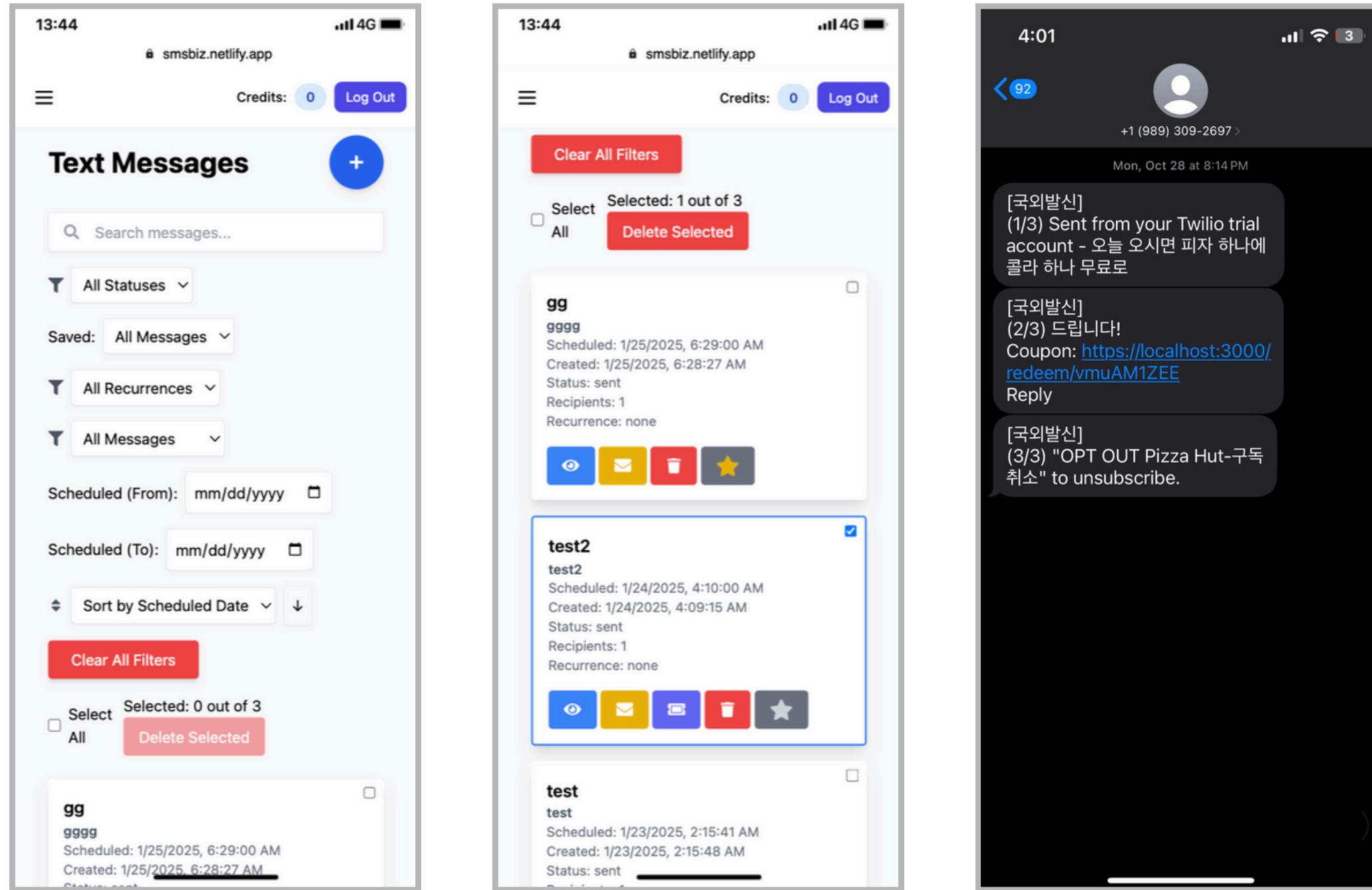
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

비즈니스 상세 대시보드



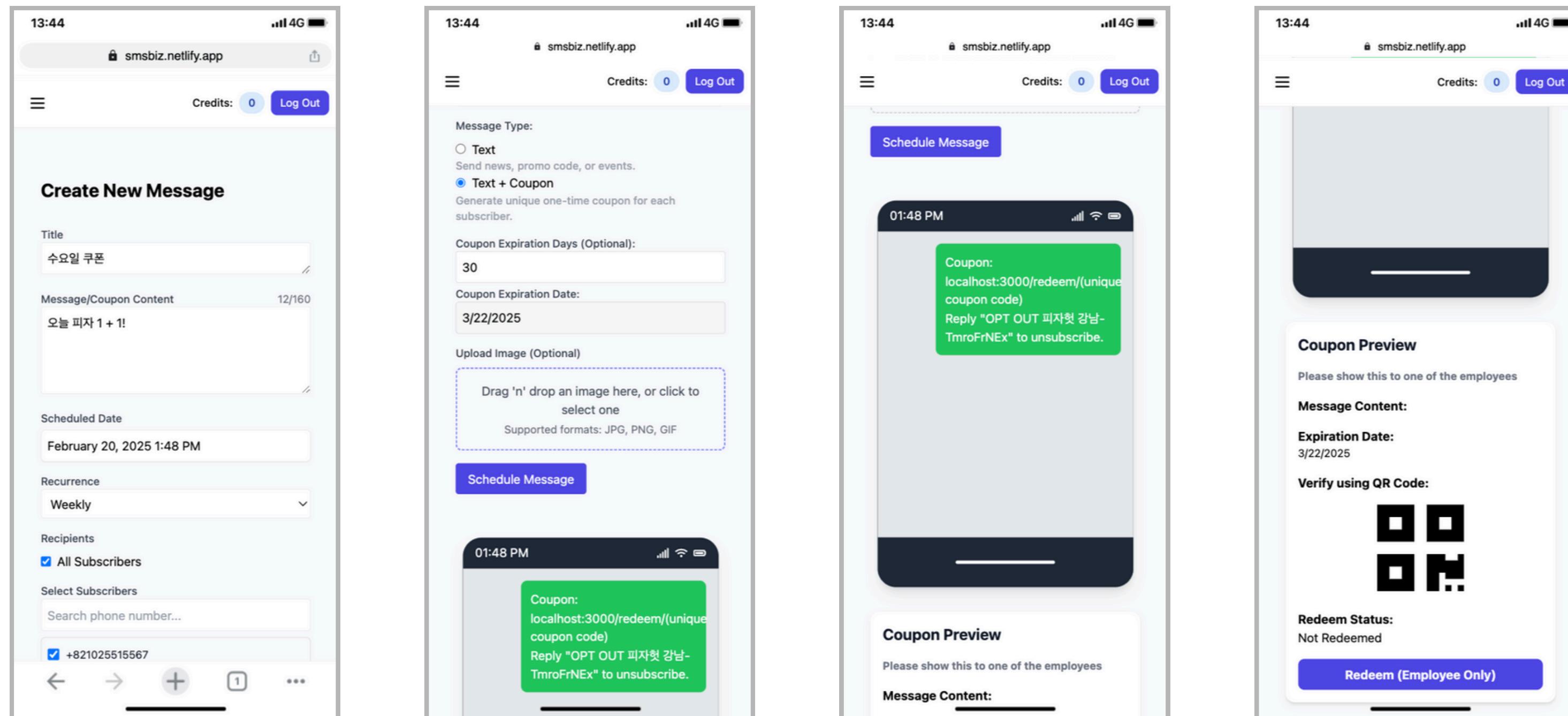
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

마케팅 메시지 관리, 예약 발송 시스템, 발송 상태 추적, 검색 기능, 다중 필터링, 정렬 기능



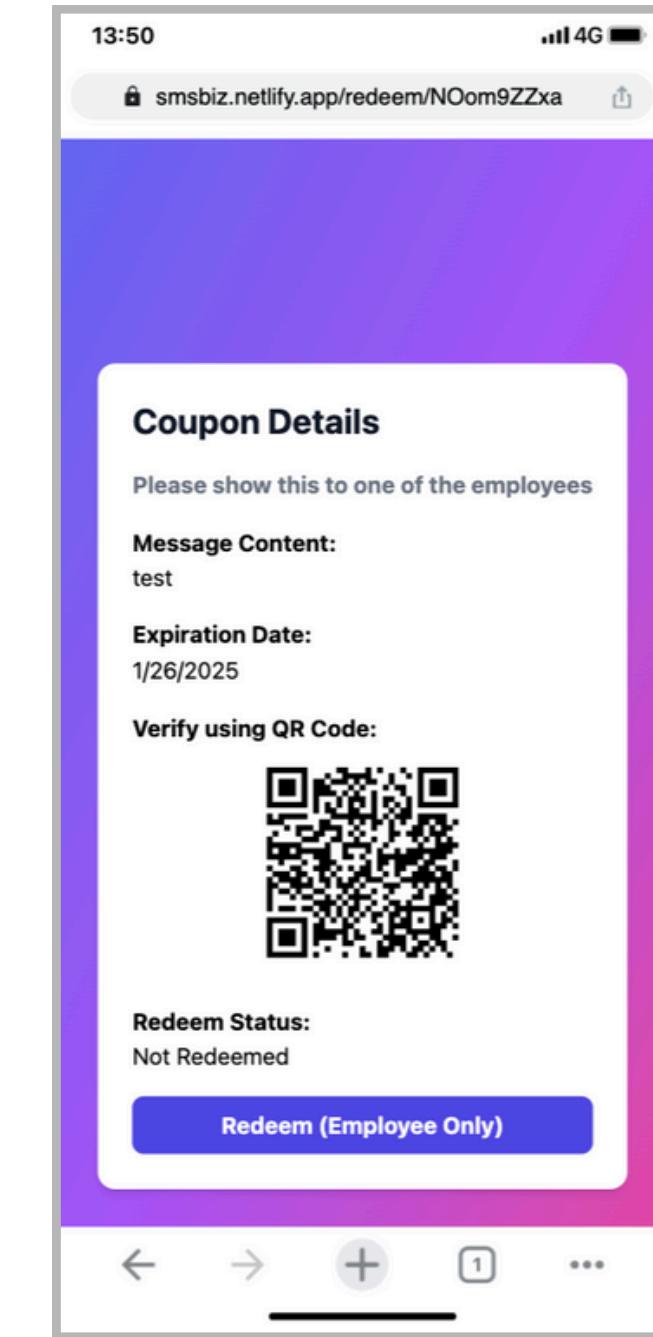
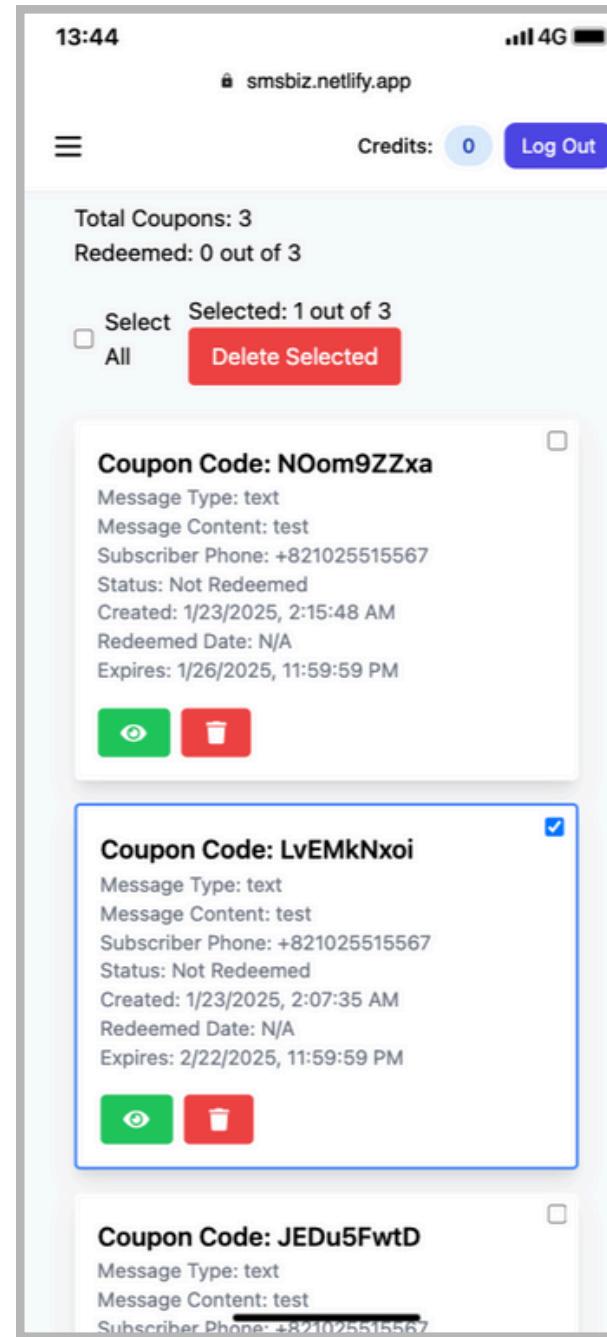
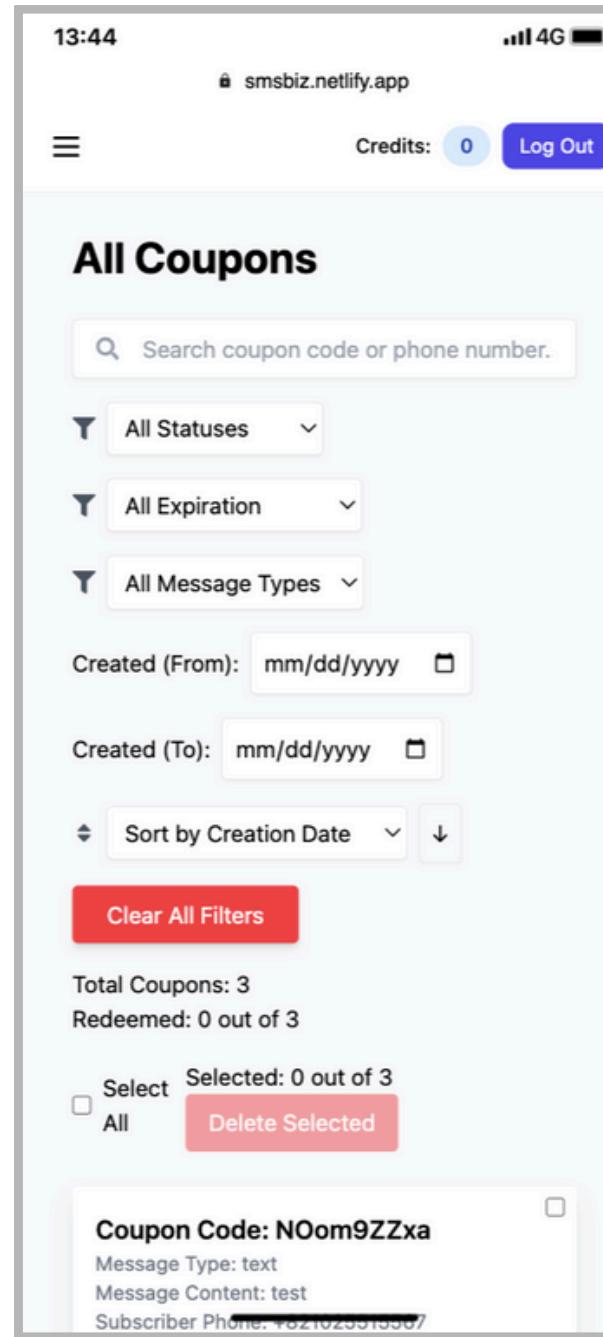
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

마케팅 메시지 작성 및 예약 발송, 메시지 템플릿 저장, 이미지 첨부, 반복 발송 설정, 예약 발송, 대량 메시지 발송/처리, 문자 & 쿠폰 미리보기



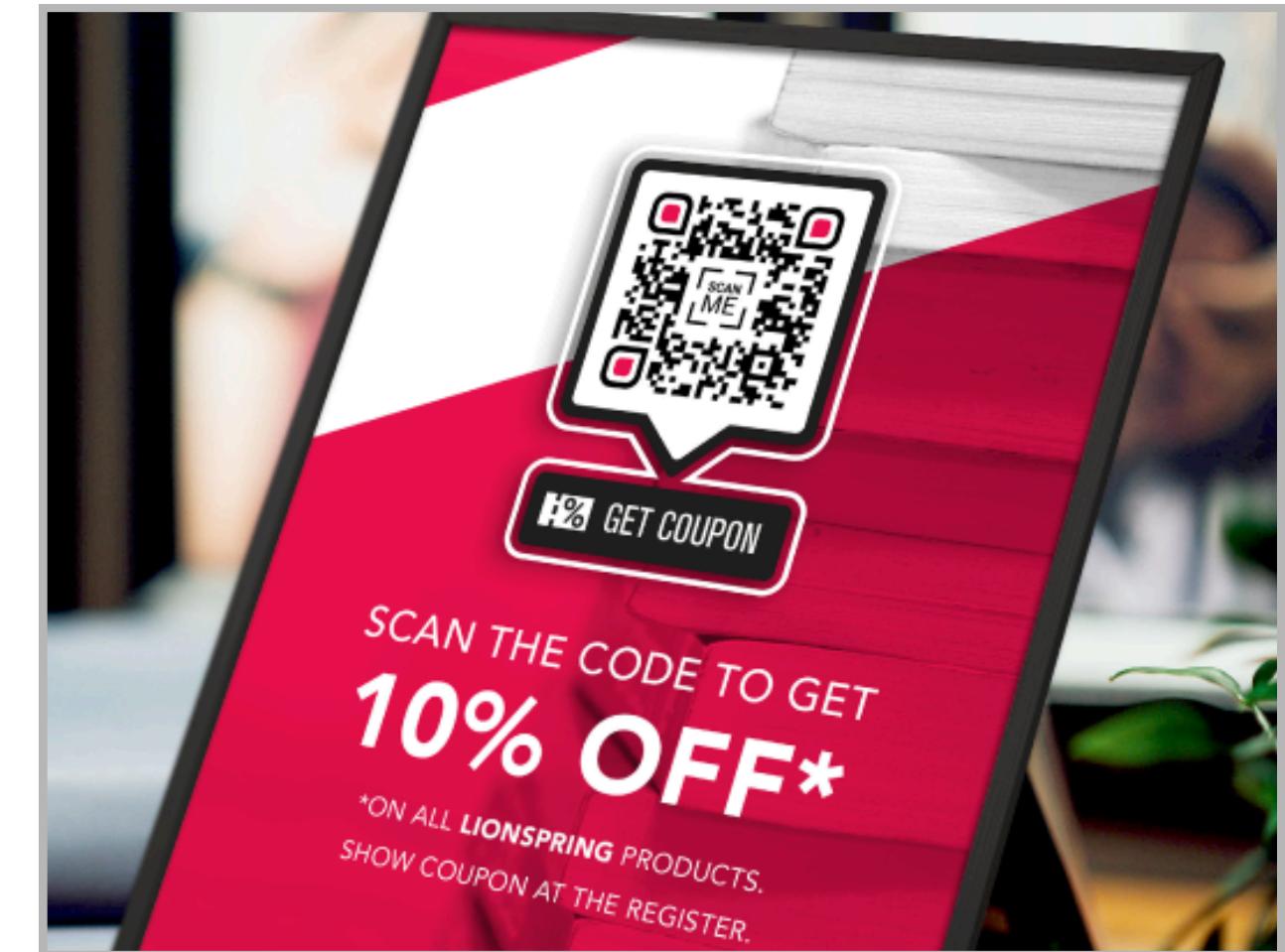
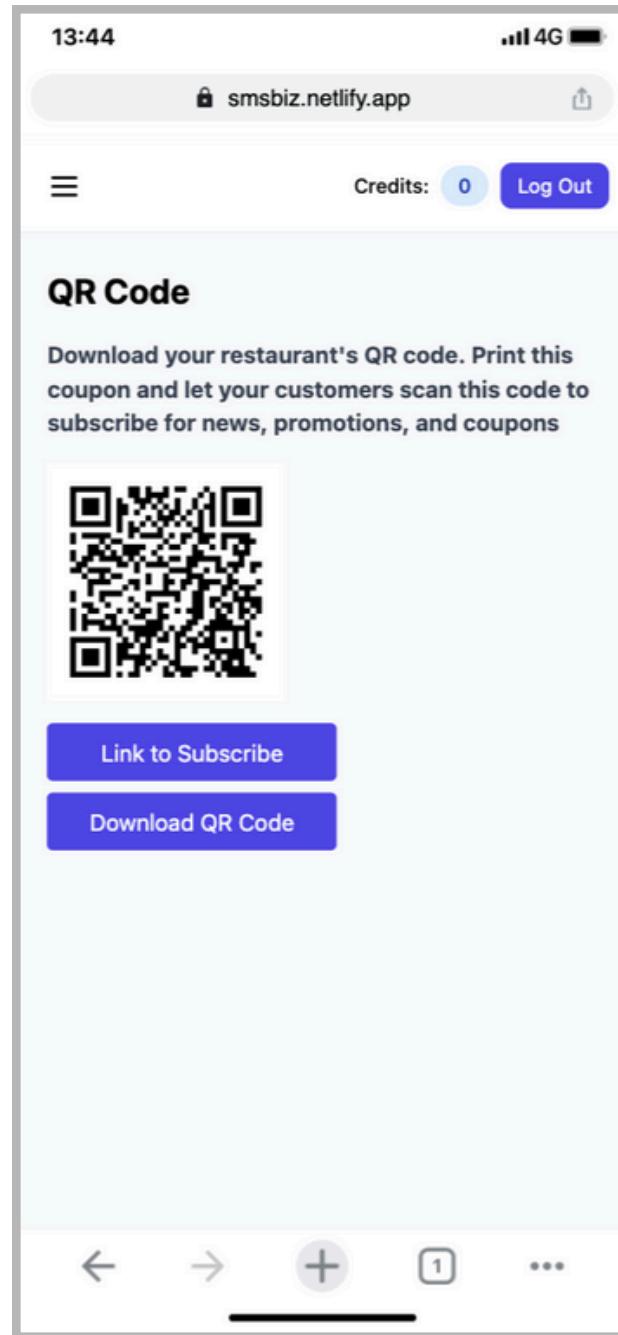
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

쿠폰 생성 및 관리, QR 코드 기반 쿠폰 인증, 쿠폰 만료일 관리, 사용 현황 추적, 자동 만료 설정



03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

식당 QR 코드 다운로드, QR 코드 스캔 비즈니스 마케팅 구독



03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

구독자 메시지/쿠폰 자동 발생, 메시지 작성, 템플릿 저장

The image consists of three vertically stacked screenshots of a mobile application for managing text messages and coupons. All three screenshots show a top header with the time (13:44), signal strength (4G), battery level, and the URL 'smsbiz.netlify.app'. A 'Log Out' button is also present in the top right.

- Screenshot 1: Sent Welcome Text Messages**

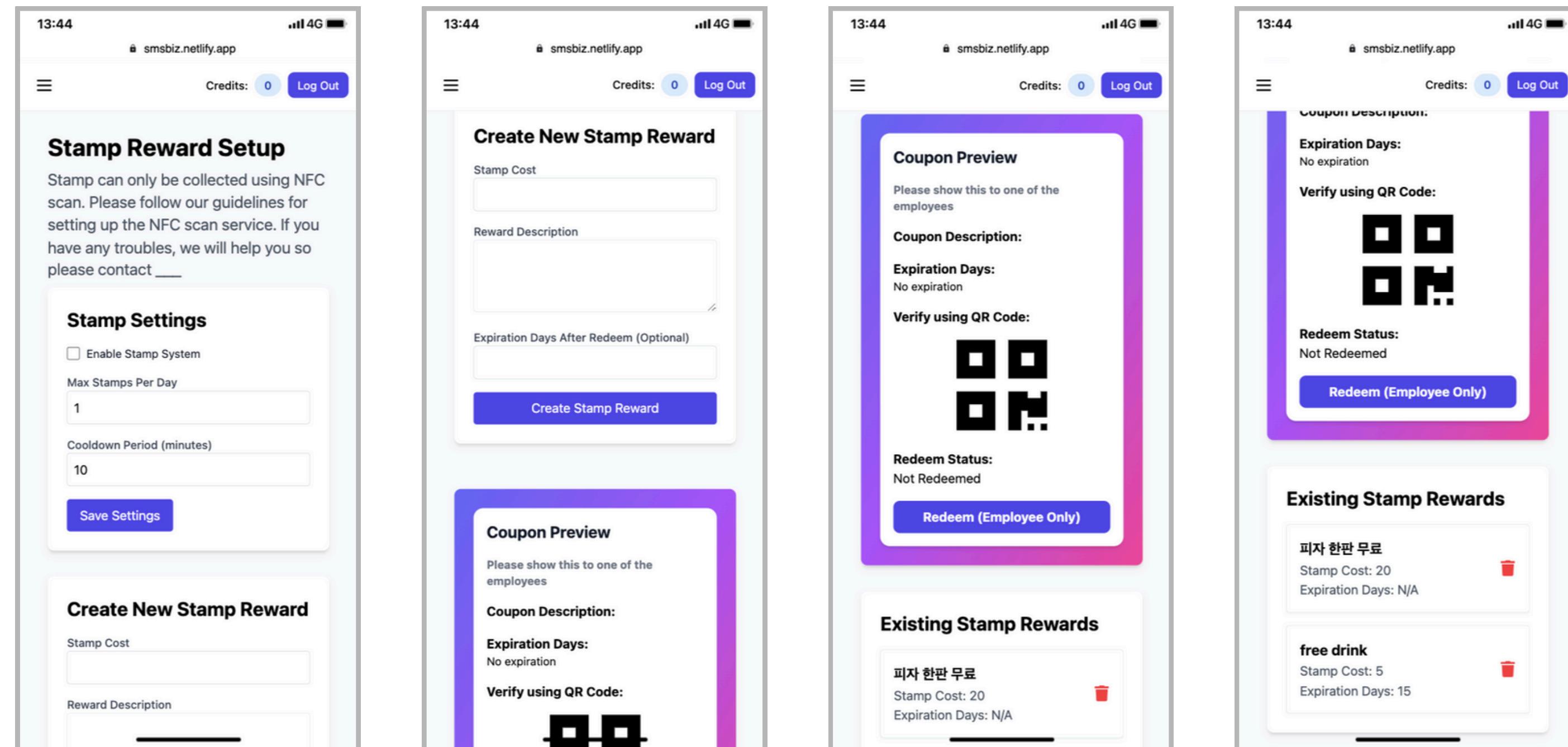
This screen shows a list of sent messages. At the top, it says 'Sent' and 'Welcome Text Messages'. Below is a search bar and filter options ('All Statuses', 'All Types'). It includes fields for 'Sent (From)' and 'Sent (To)'. A red 'Clear All Filters' button is visible. At the bottom, there's a message card for a message sent to '+821025515567' on '1/24/2025, 5:07:45 AM'.
- Screenshot 2: Welcome Message Setup**

This screen is titled 'Welcome Message' and instructs the user to 'Setup text message to send when customer first subscribes'. It features a text input field containing the message: '첫 구독 감사합니다! 무료 음료 쿠폰을 드립니다.' Below this is a 'Message/Coupon Content' section with a character limit of 28/160. It includes a 'View All' button. Further down are sections for 'Message Type' (radio buttons for 'Text' and 'Text + Coupon', with 'Text + Coupon' selected), 'Coupon Expiration Days' (set to 5), 'Coupon Expiration Date' (set to 2/25/2025), and an 'Upload Image (Optional)' section with a file input field.
- Screenshot 3: Message Preview and Save**

This screen shows a preview of the message that will be sent. The message content is identical to the one in the setup screen. It includes a 'Save Welcome Text' button. Below the preview is a representation of a smartphone displaying the message: '01:52 PM 첫 구독 감사합니다! 무료 음료 쿠폰을 드립니다.' The message is highlighted in green. At the bottom of the phone screen, there is a coupon code: 'Coupon: localhost:3000/redeem/(unique coupon code)' and instructions: 'Reply "OPT OUT 피자헛 강남-TmroFrNEx" to unsubscribe.'

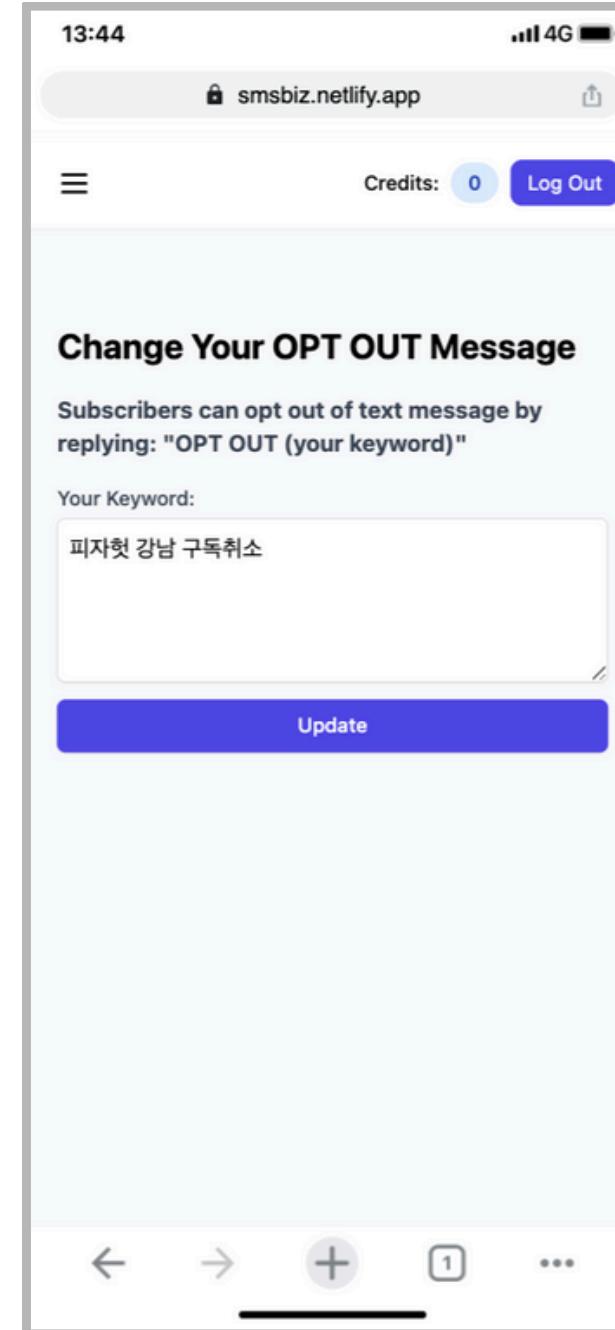
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

스탬프 적립 규칙 설정, 일일 제한 쿨다운 시스템, 리워드 교환 시스템, 만료일 관리, 메시지/쿠폰 자동 발송

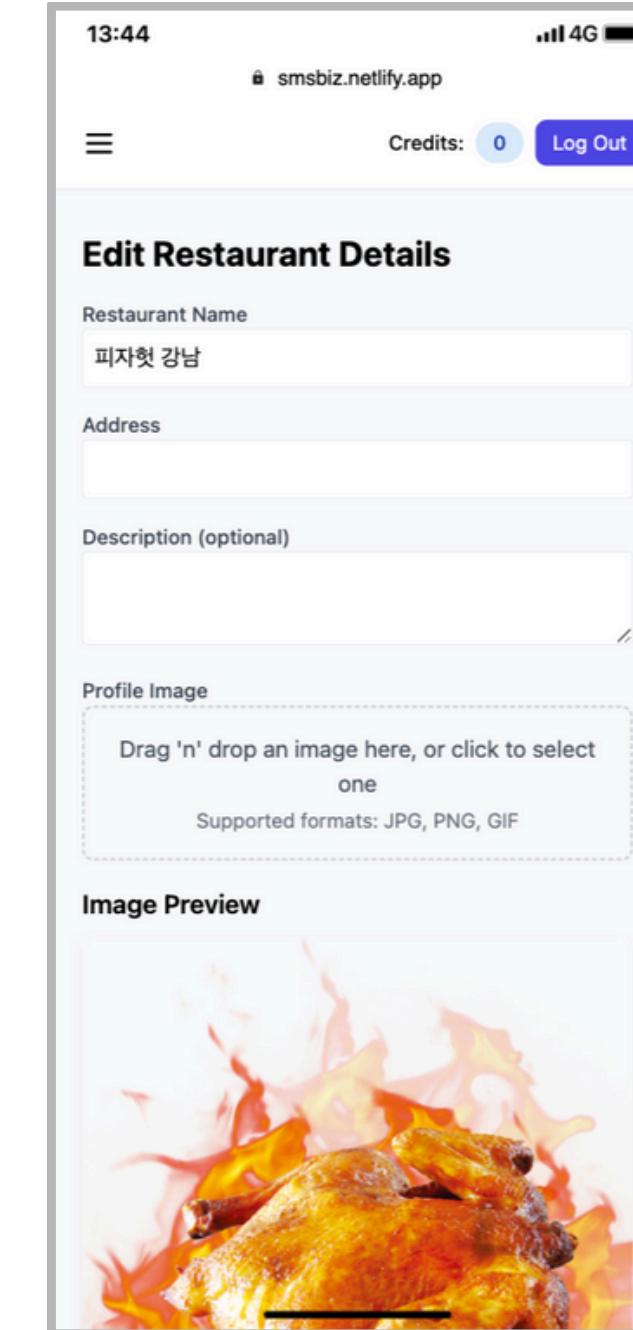


03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

자동 구독 취소 문자 답장 설정

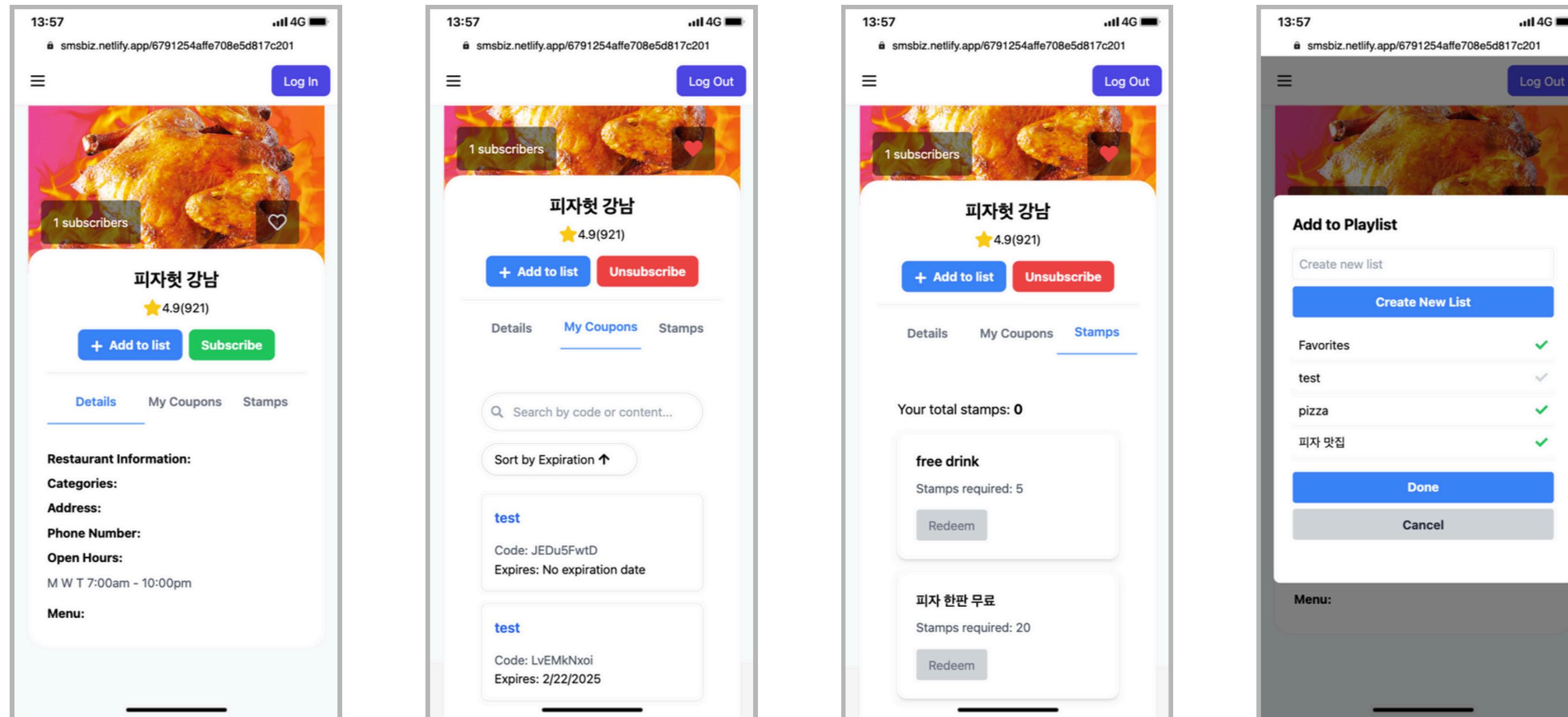


식당 정보 수정/삭제



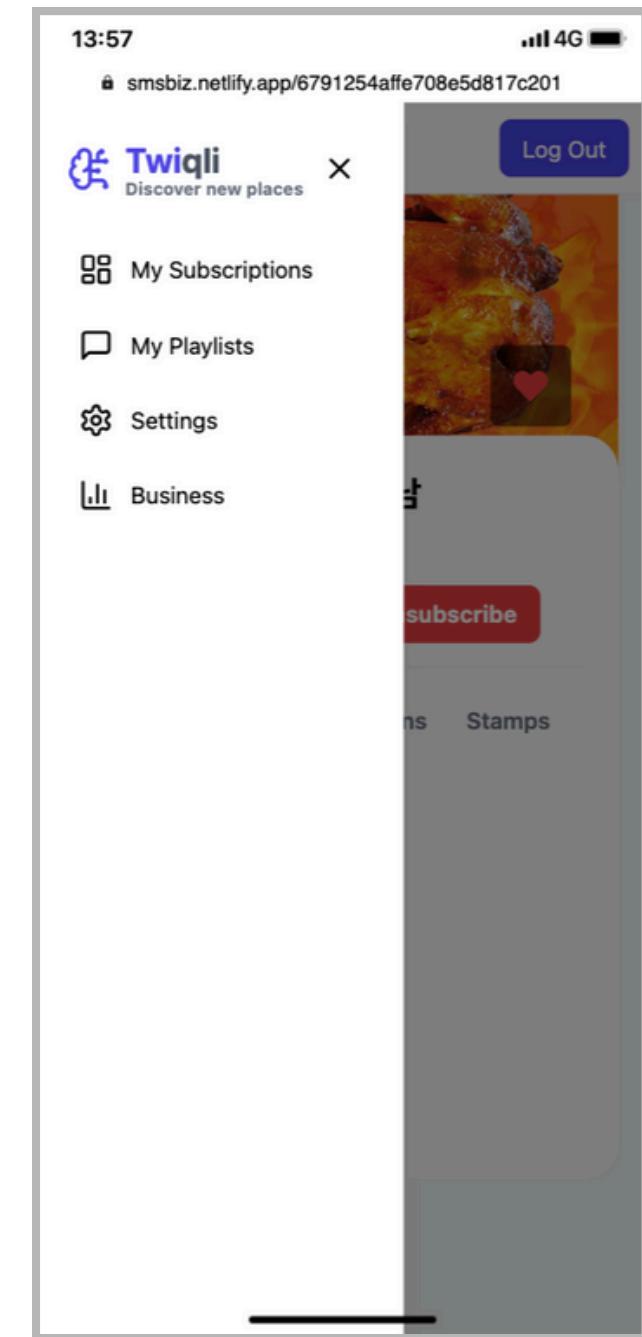
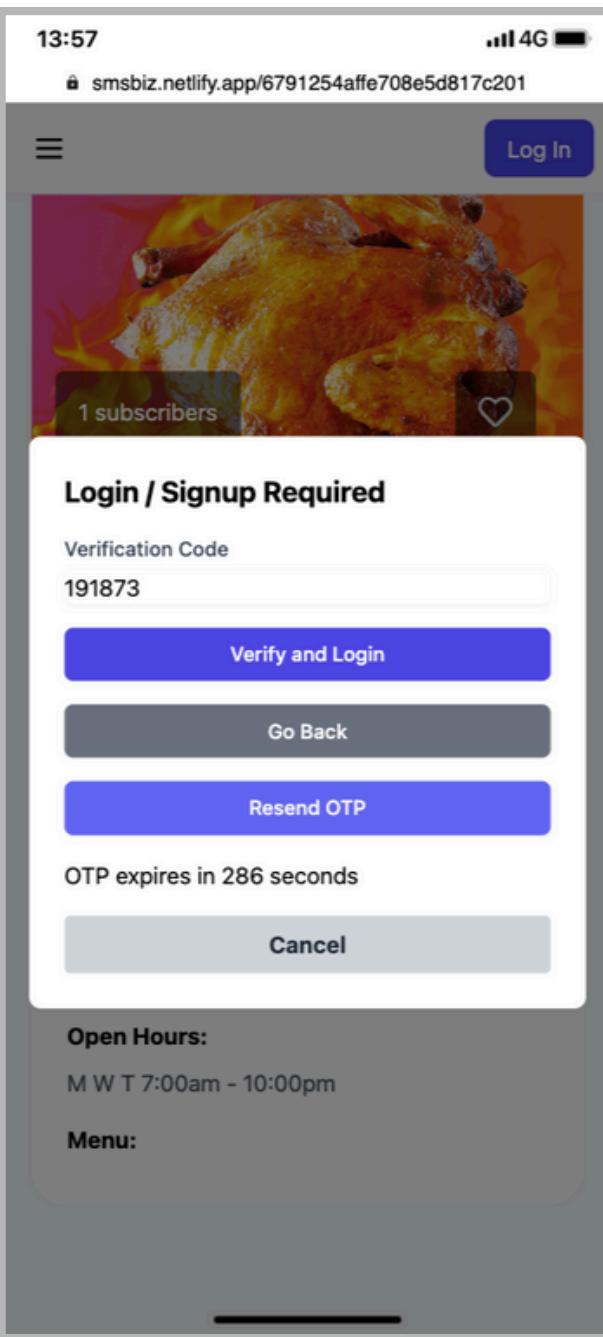
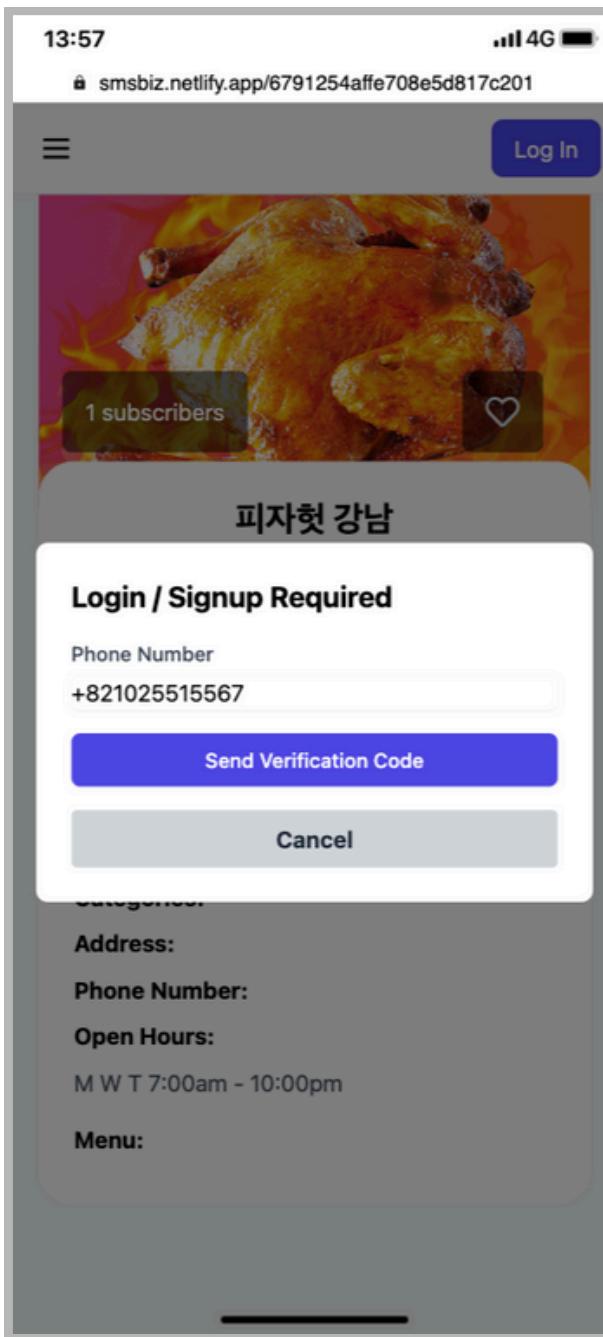
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

고객용 비즈니스 상세 페이지, 구독 및 취소, 즐겨찾기, 좋아하는 식당 저장, 커스텀 플레이리스트 생성 및 관리, 발급받은 쿠폰 목록 확인, 적립 현황, 리워드 교환



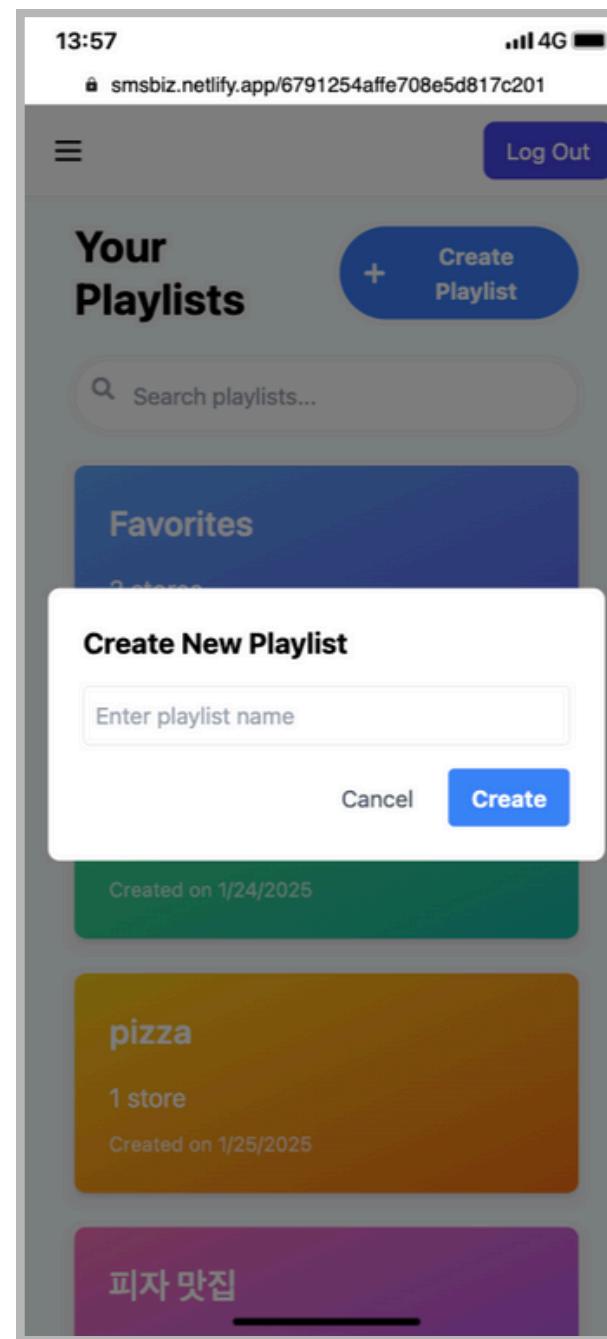
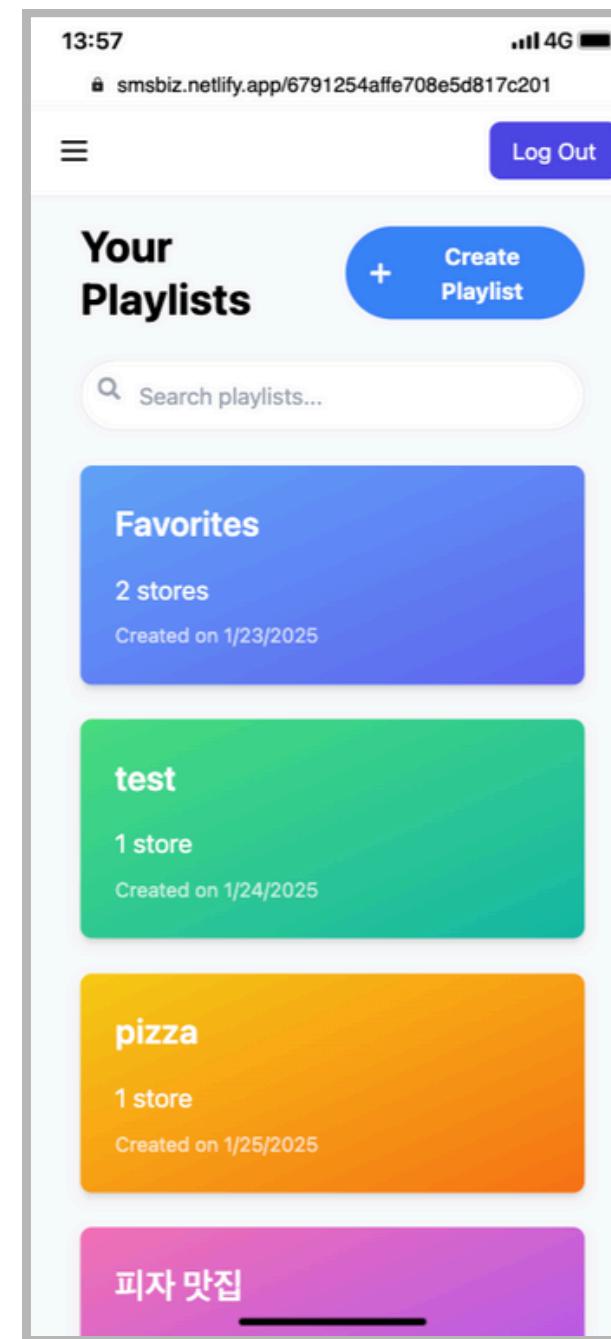
03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

식당 구독, OTP 인증

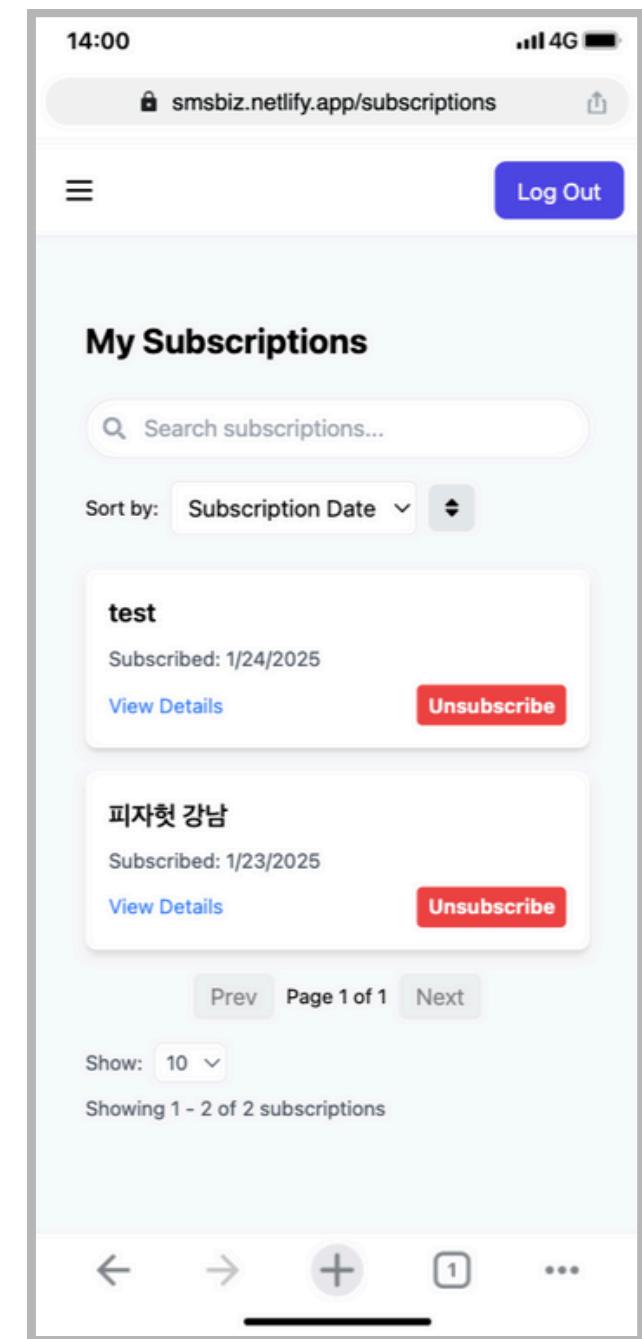


03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

플레이리스트 관리

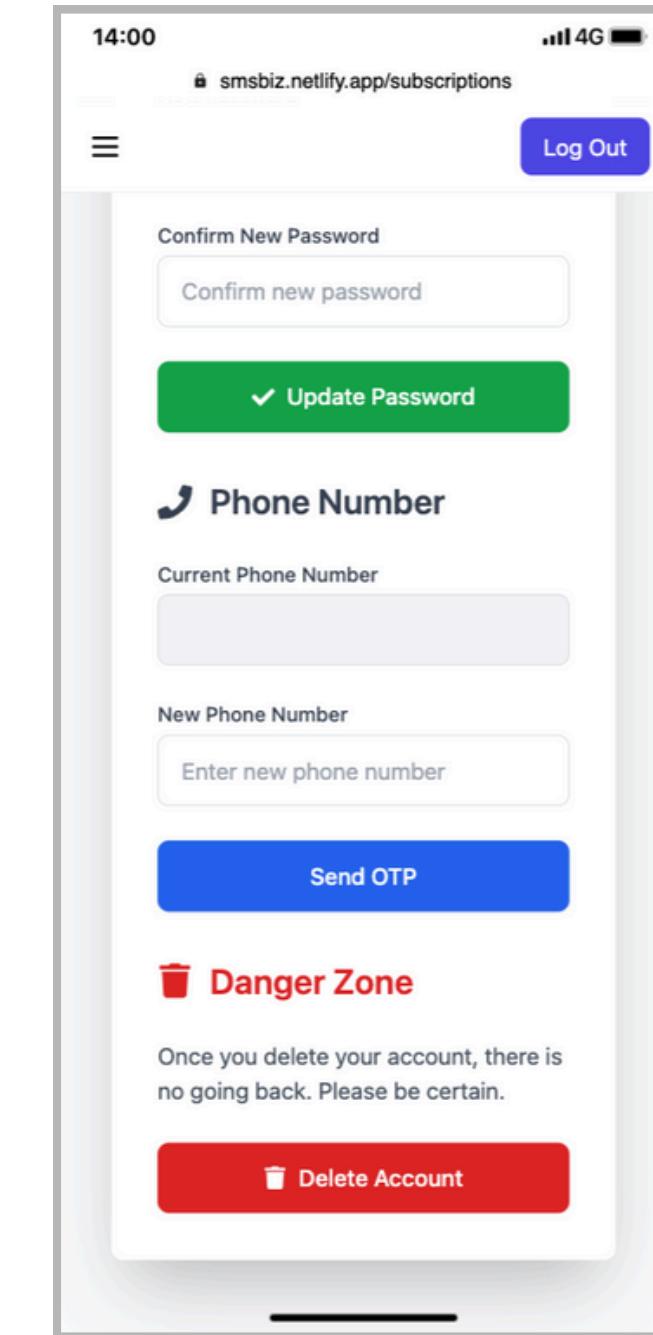
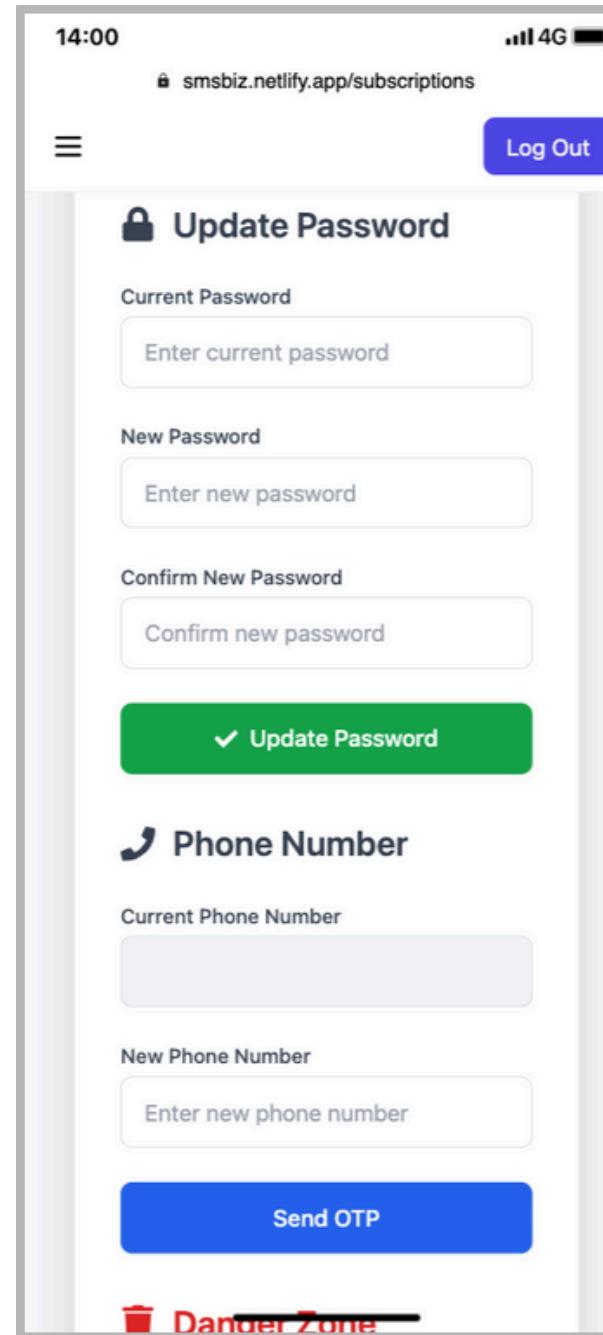
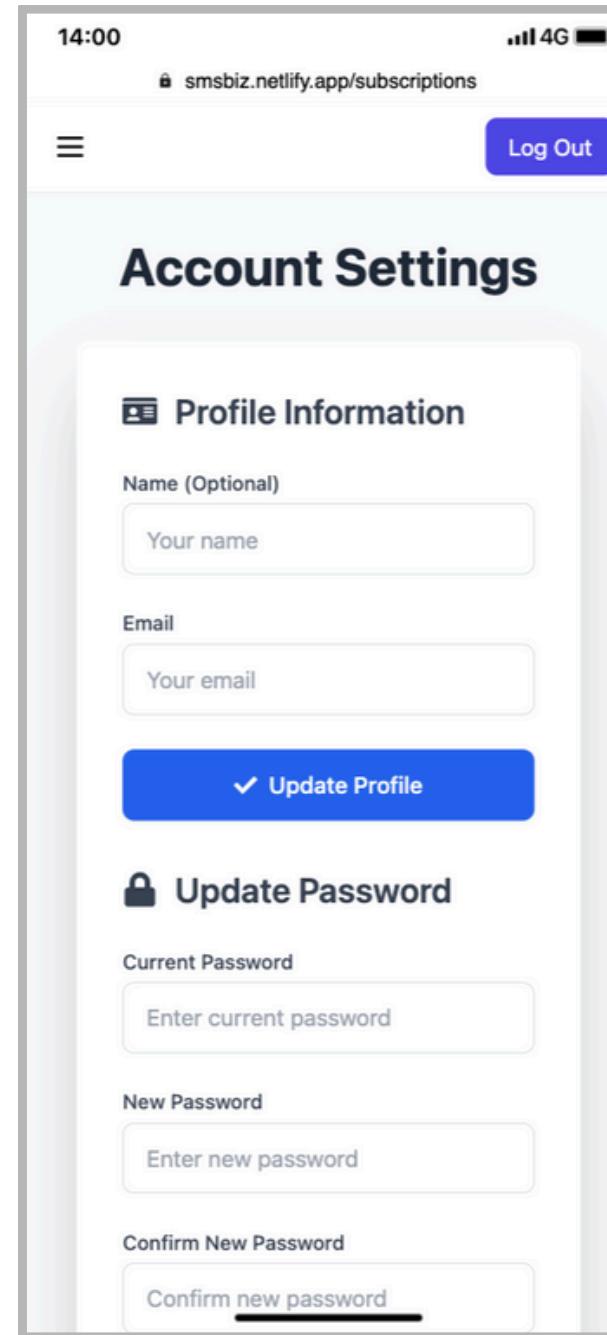


구독 중인 비즈니스 목록 확인



03 오프라인 매장을 위한 문자 구독 & 마케팅 서비스

고객 계정 설정 관리



04 이커머스 사이트 소개

이커머스 플랫폼 개발 프로젝트를 통해 제가 가진 기술력과 문제 해결 능력을 증명하고자 했습니다. React와 Firebase를 기반으로 한 이 프로젝트는 단순한 기술 구현을 넘어, 실제 사용자 경험을 고려한 서비스를 만드는 것을 목표로 했습니다. 특히 상태 관리, 성능 최적화, 반응형 디자인 과정에서 프론트엔드 개발자로서의 역량을 크게 향상시킬 수 있었습니다.

이커머스 플랫폼으로, 실시간 상품 업데이트, 사용자 인증, 장바구니 관리, 위시리스트 기능, 상품 리뷰 등 종합적인 쇼핑 경험을 제공합니다.

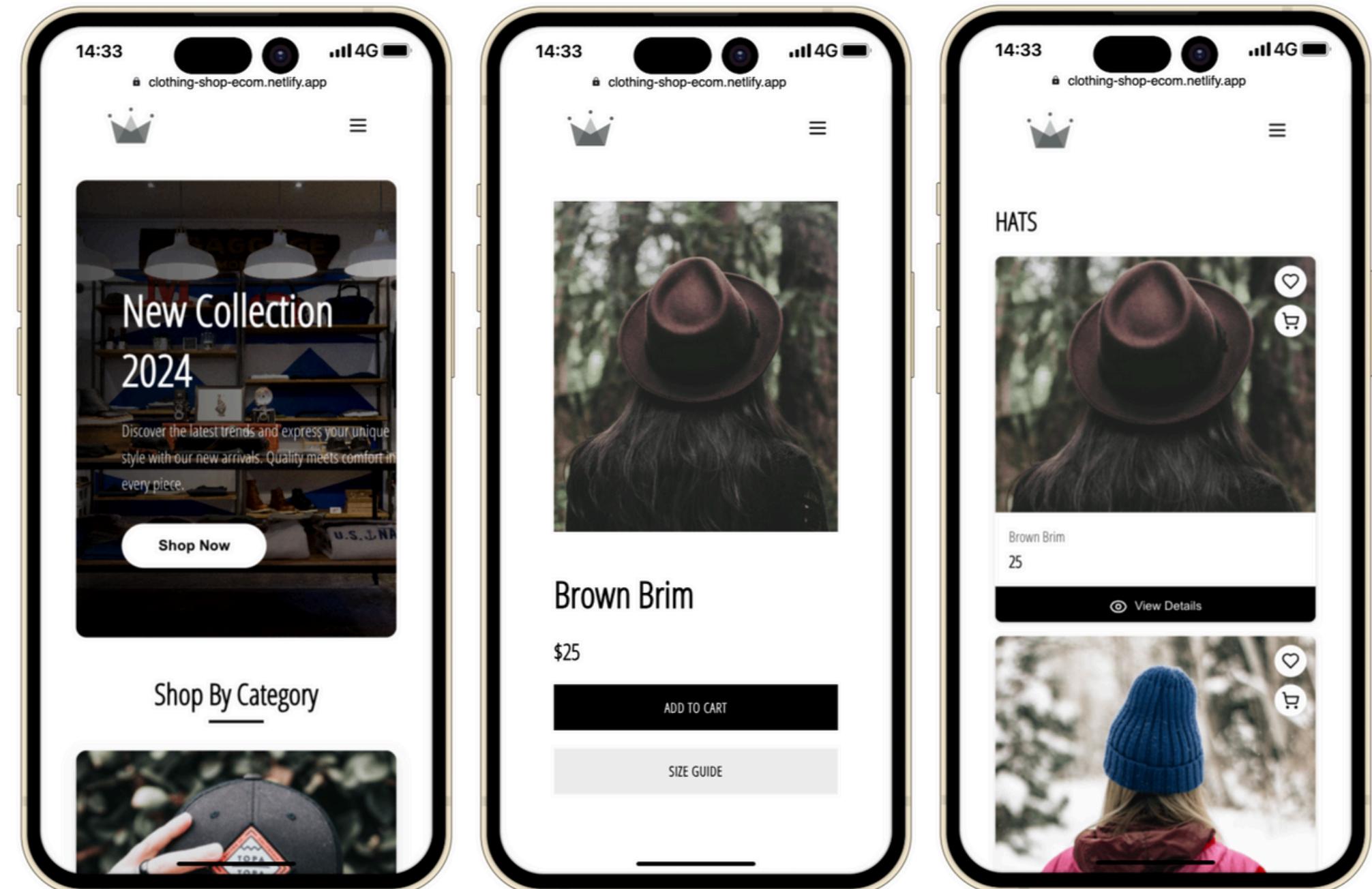
개발 기간 기획 및 설계, 디자인: 2023.08.20 ~ 2023.08.27
 개발: 2023.08.28 ~ 2023.10.09
 테스트 및 배포: 2023.10.10 ~ 2023.10.17

개발인원 1명

담당역할 프론트엔드, 백엔드
 • 서비스 기획 및 방향성 설정
 • 메인 로직 구현 (프론트엔드, 백엔드), API design, DB 설계

깃허브 <https://github.com/whd793/Shopping-eCommerce>

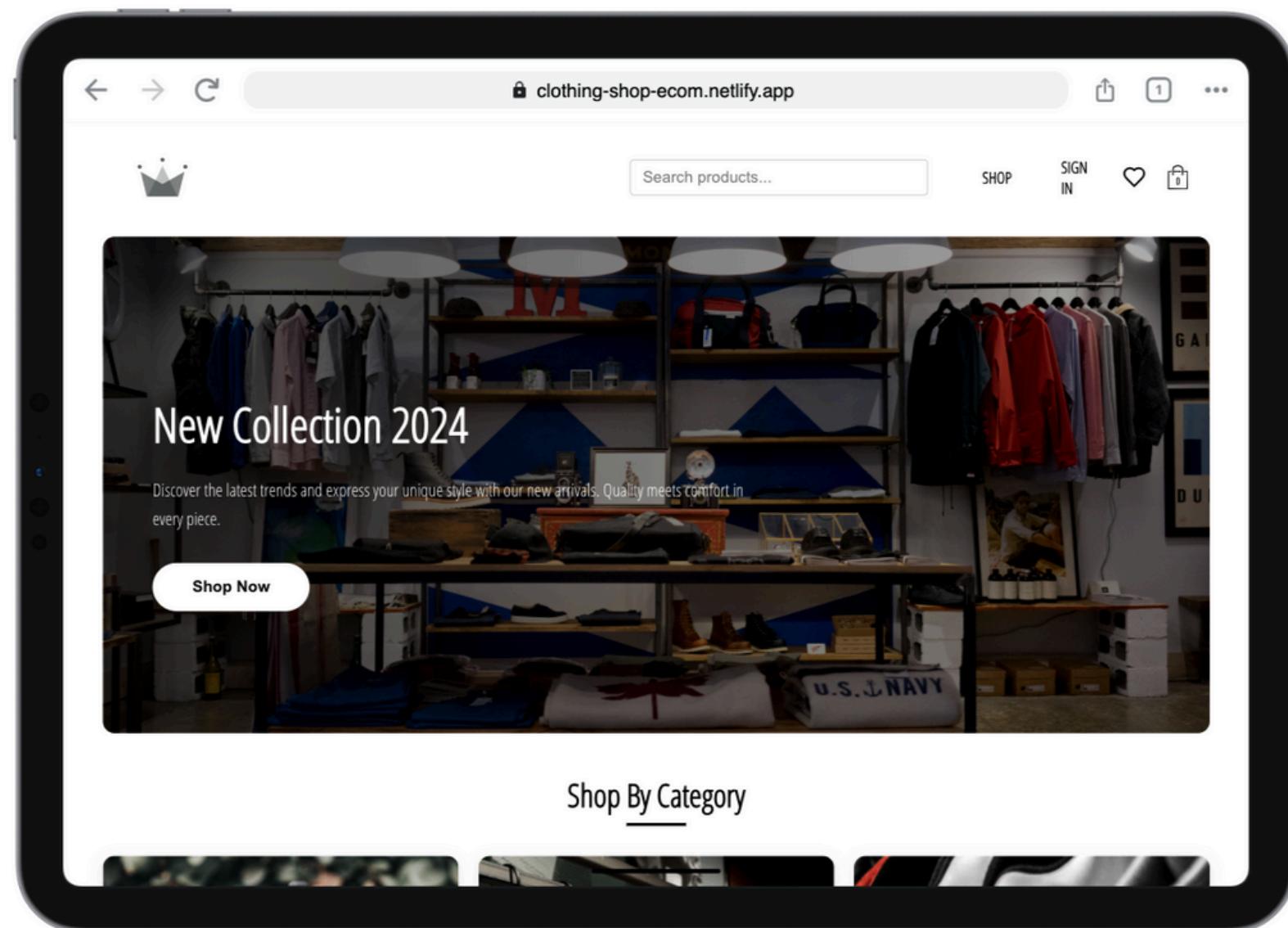
Demo <https://clothing-shop-ecom.netlify.app/>



04 이커머스 사이트

개발 환경

- 프론트엔드 코어: React 18, Redux Toolkit
- 스타일링: Styled-components
- 백엔드: Firebase (Auth, Firestore)
- 상태 관리: Redux Toolkit, Redux-persist
- 라우팅: React Router v6
- UI 컴포넌트: 커스텀 컴포넌트 라이브러리
- 버전 관리: Git
- 개발 도구: VS Code, Chrome DevTools



04 이커머스 사이트

주요 기능 (1/2)

Redux Toolkit을 활용한 전역 상태 관리 시스템 구축

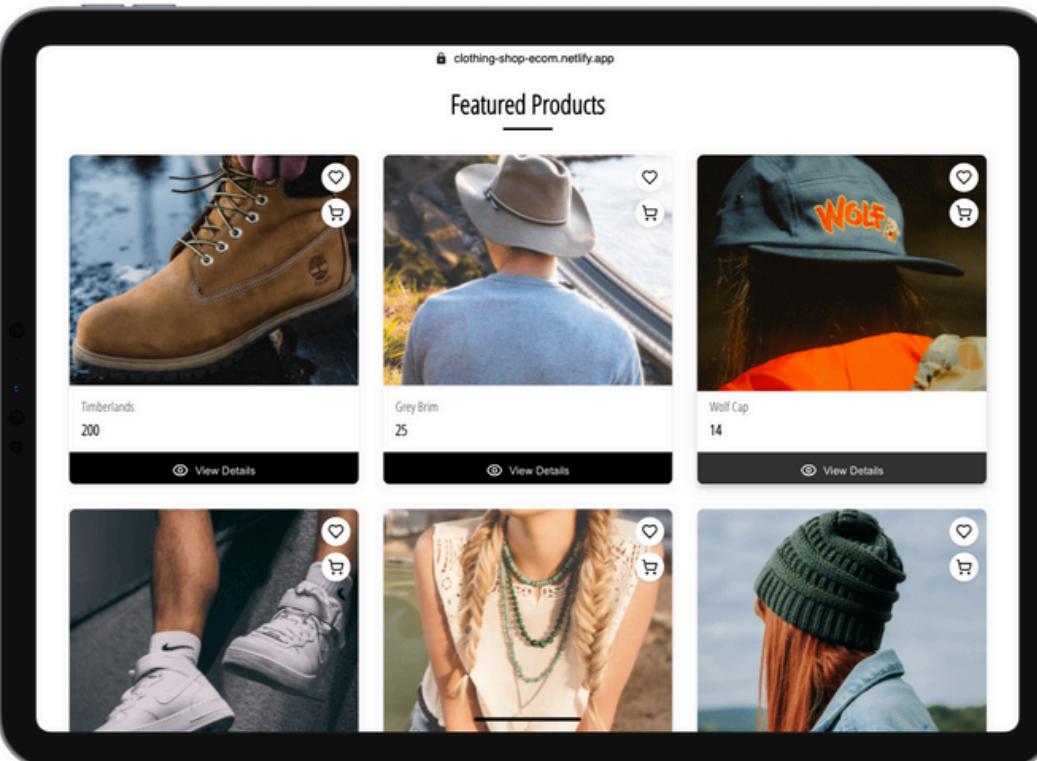
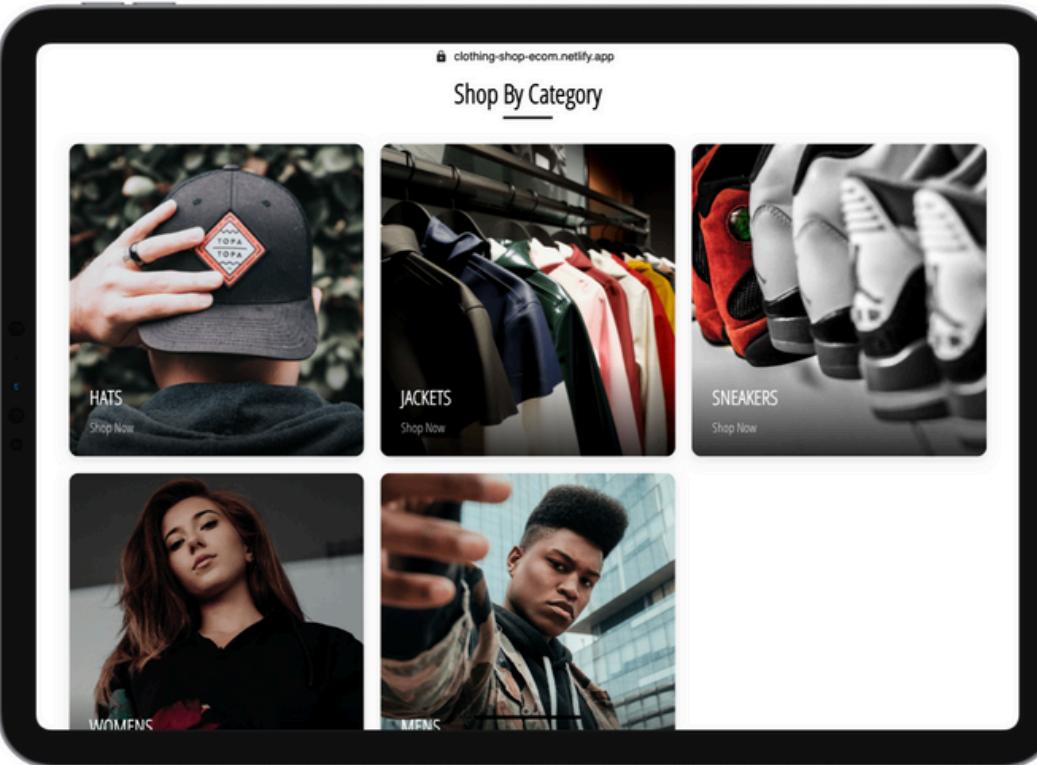
- 장바구니, 위시리스트, 사용자 인증 상태를 효율적으로 관리
- Selector 패턴 적용으로 불필요한 리렌더링 최적화
- Redux slice 패턴으로 상태 관리 로직 모듈화
- Redux 상태 정규화로 데이터 중복 제거

Firebase 기반 사용자 인증 시스템 구현

- 이메일/비밀번호 및 Google OAuth 로그인
- Firestore 데이터베이스 설계 및 쿼리 최적화
- 보안 규칙 설정으로 데이터 접근 제어
- 실시간 사용자 세션 관리 구현

Styled-components를 활용한 컴포넌트 스타일링

- 재사용 가능한 UI 컴포넌트 라이브러리 구축
- 반응형 디자인 적용으로 모바일 지원
- 테마 기반 디자인 시스템 구현
- 모던한 반응형 디자인



04 이커머스 사이트

주요 기능 (2/2)

쇼핑몰 핵심 기능 개발

상품 관리

- 카테고리별 상품 필터링
- 추천 상품 섹션
- 상세 상품 페이지
- 상품 이미지 최적화
- 최근 본 상품 추적
- 상품 리뷰 시스템 개발 및 Firestore 연동
- 검색 최적화 (debounce 기법 적용)

장바구니 및 결제 시스템

- 실시간 장바구니 업데이트
- 수량 조절 기능
- 장바구니 상태 유지
- 장바구니 요약
- 드롭다운이 있는 실시간 장바구니 관리

위시리스트 시스템

- 위시리스트 추가 기능

홈페이지

- 추천 컬렉션 히어로 배너
- 카테고리 쇼케이스
- 추천 상품 섹션

사용자 경험 최적화

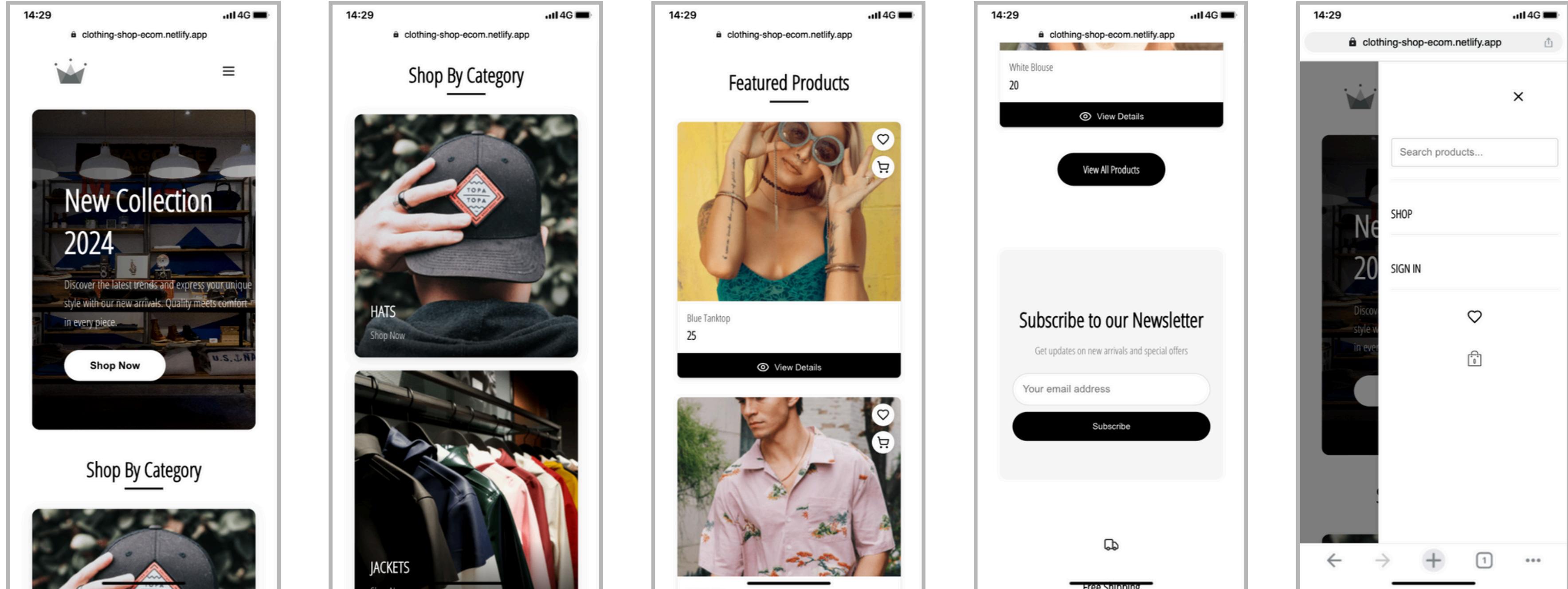
- Toast 알림 시스템 구현으로 사용자 액션 피드백 제공
- 모던한 반응형 디자인
- 인터랙티브 상품 카드
- 토스트 알림
- 로딩 상태 및 애니메이션

성능 최적화

- Code Splitting을 통한 초기 로딩 시간 최적화
- React.memo와 useMemo를 활용한 컴포넌트 최적화
- 이미지 레이저 로딩 구현
- Redux Toolkit을 활용한 상태 관리 구조화
- React 컴포넌트 최적화로 성능 향상

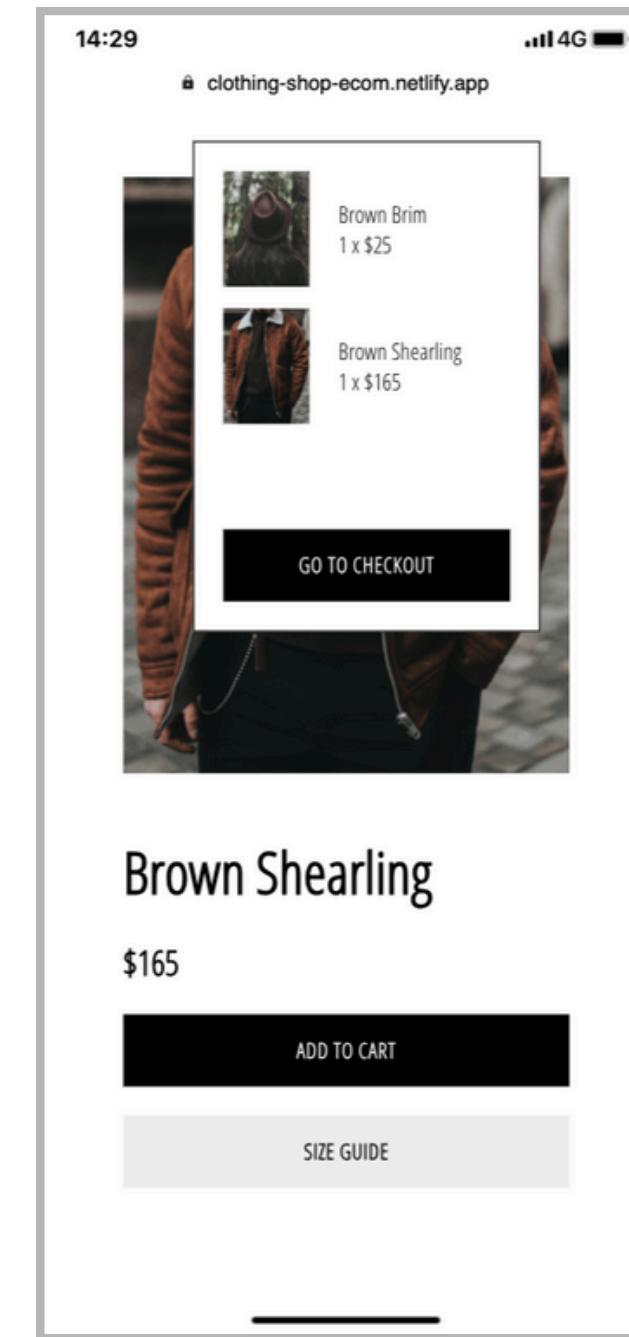
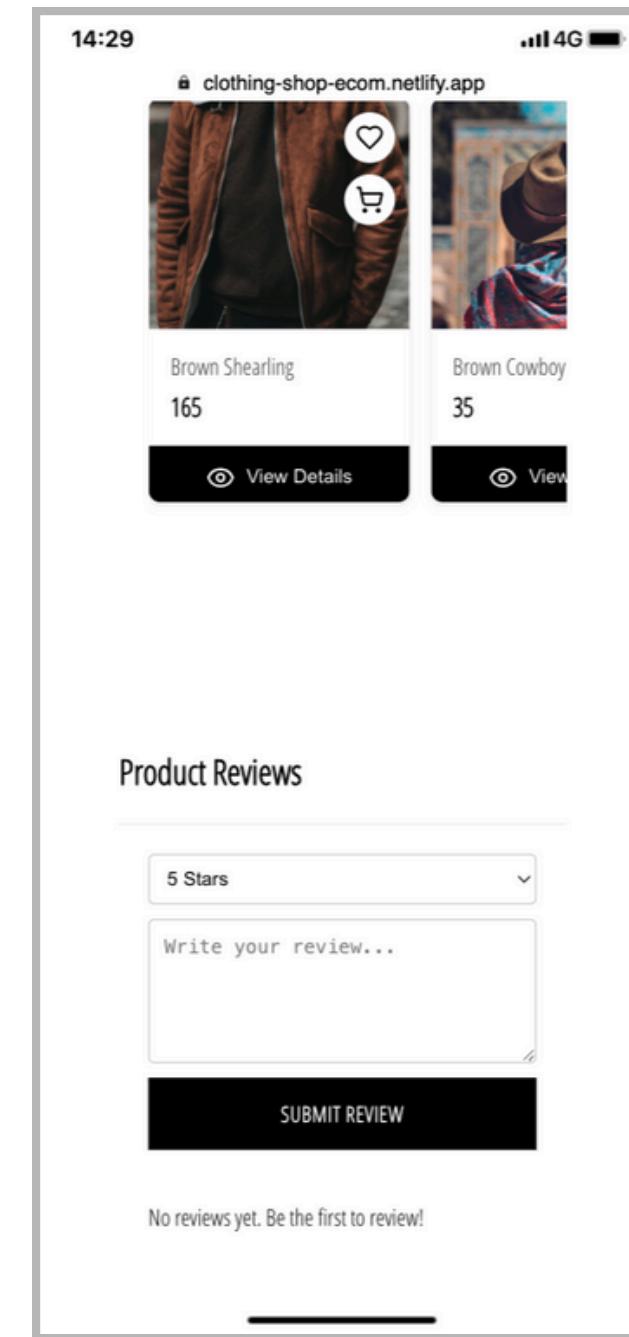
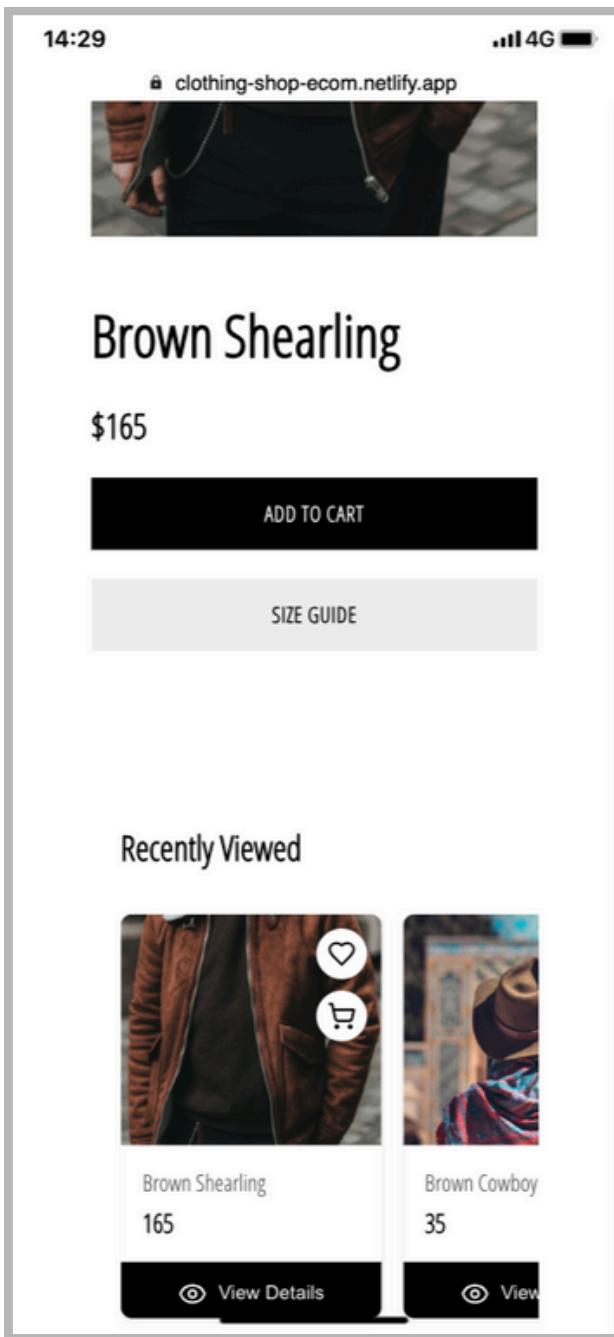
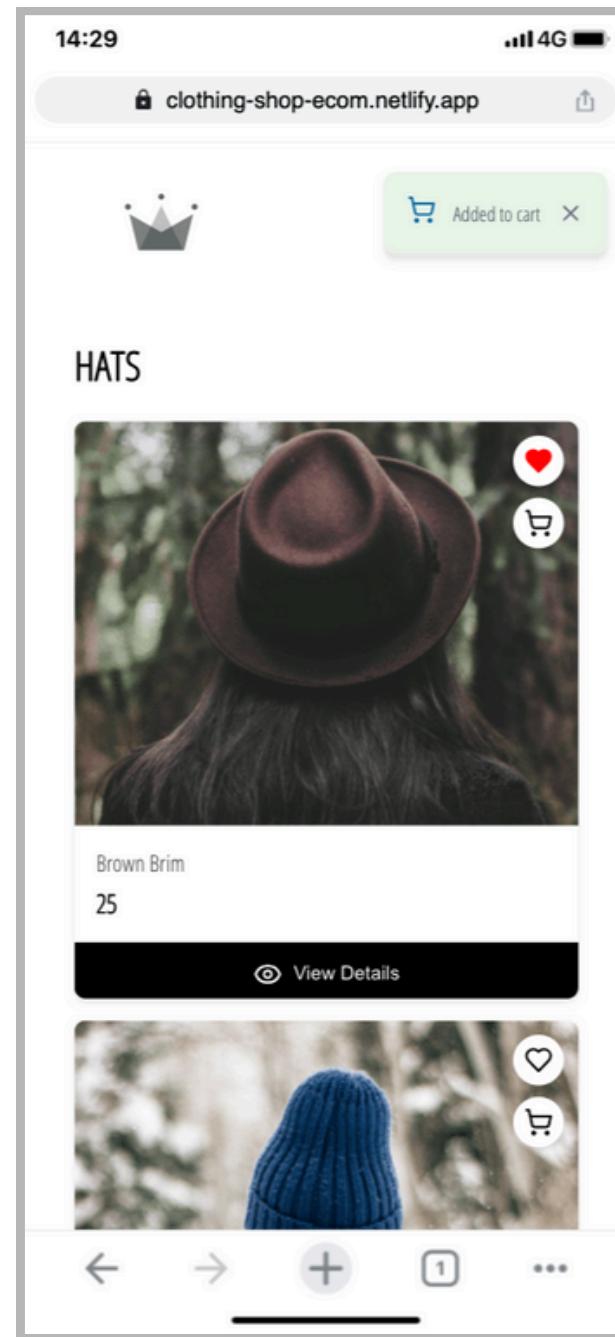
04 이커머스 사이트

홈페이지, 카테고리 쇼케이스, 추천 상품



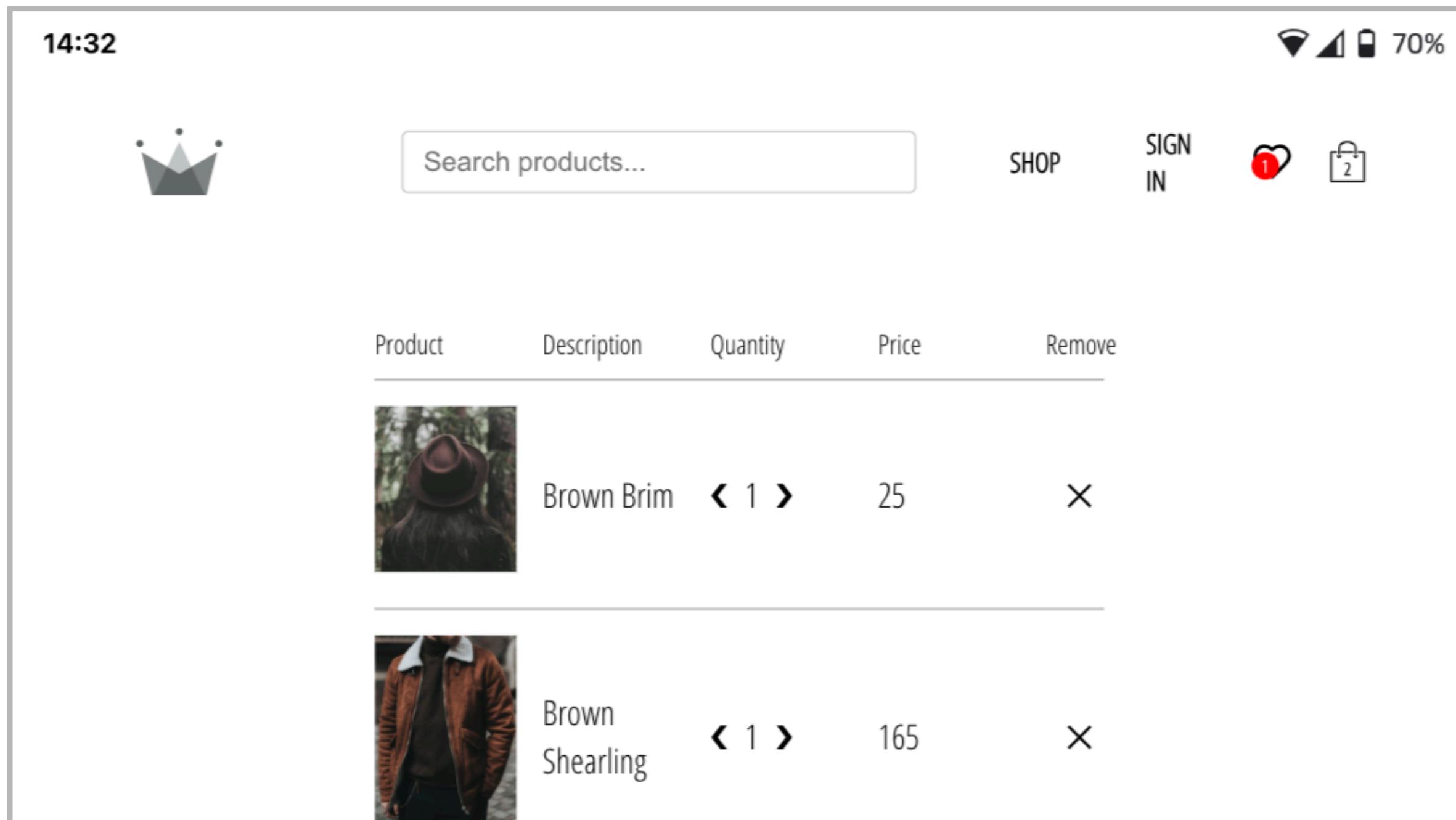
04 이커머스 사이트

상세 상품 페이지, 상품 이미지 최적화, 최근 본 상품 추적, 상품 리뷰 시스템



04 이커머스 사이트

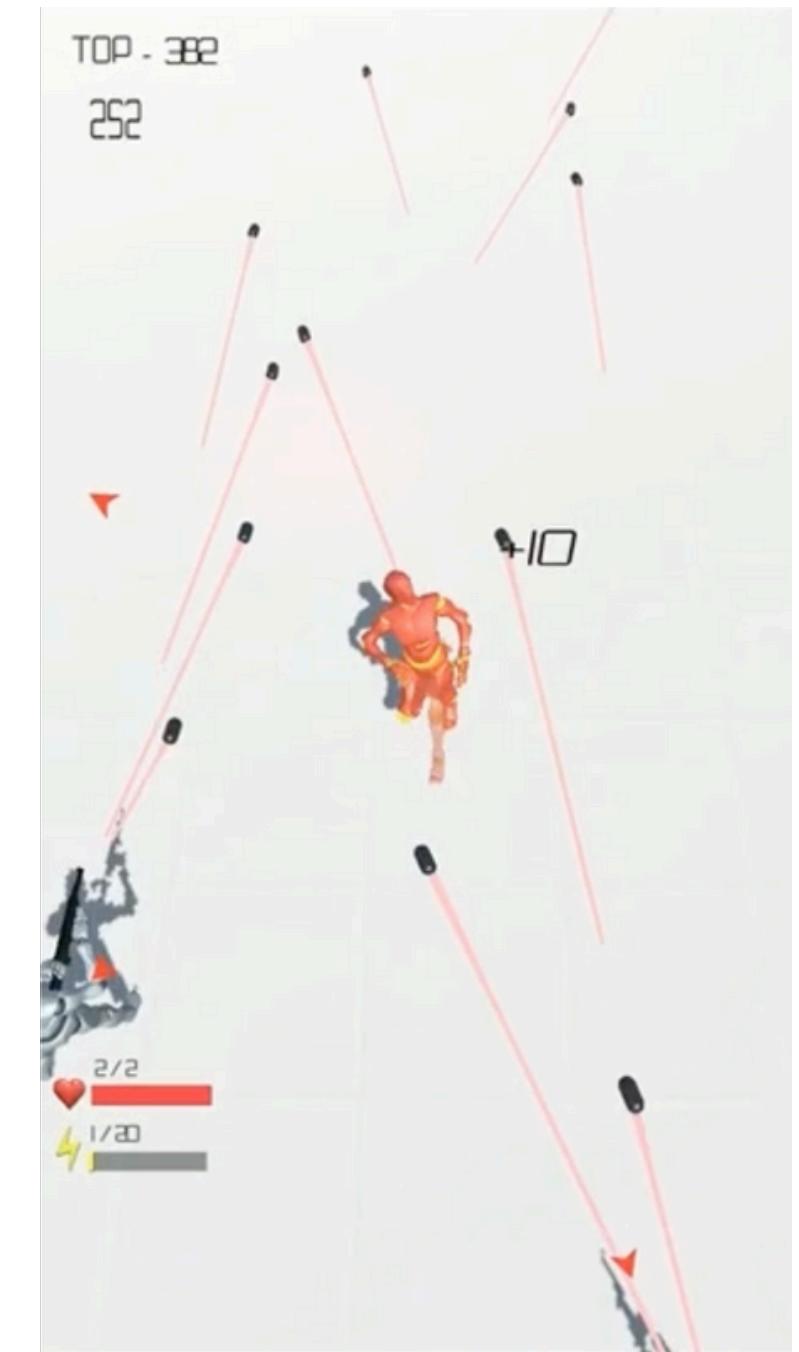
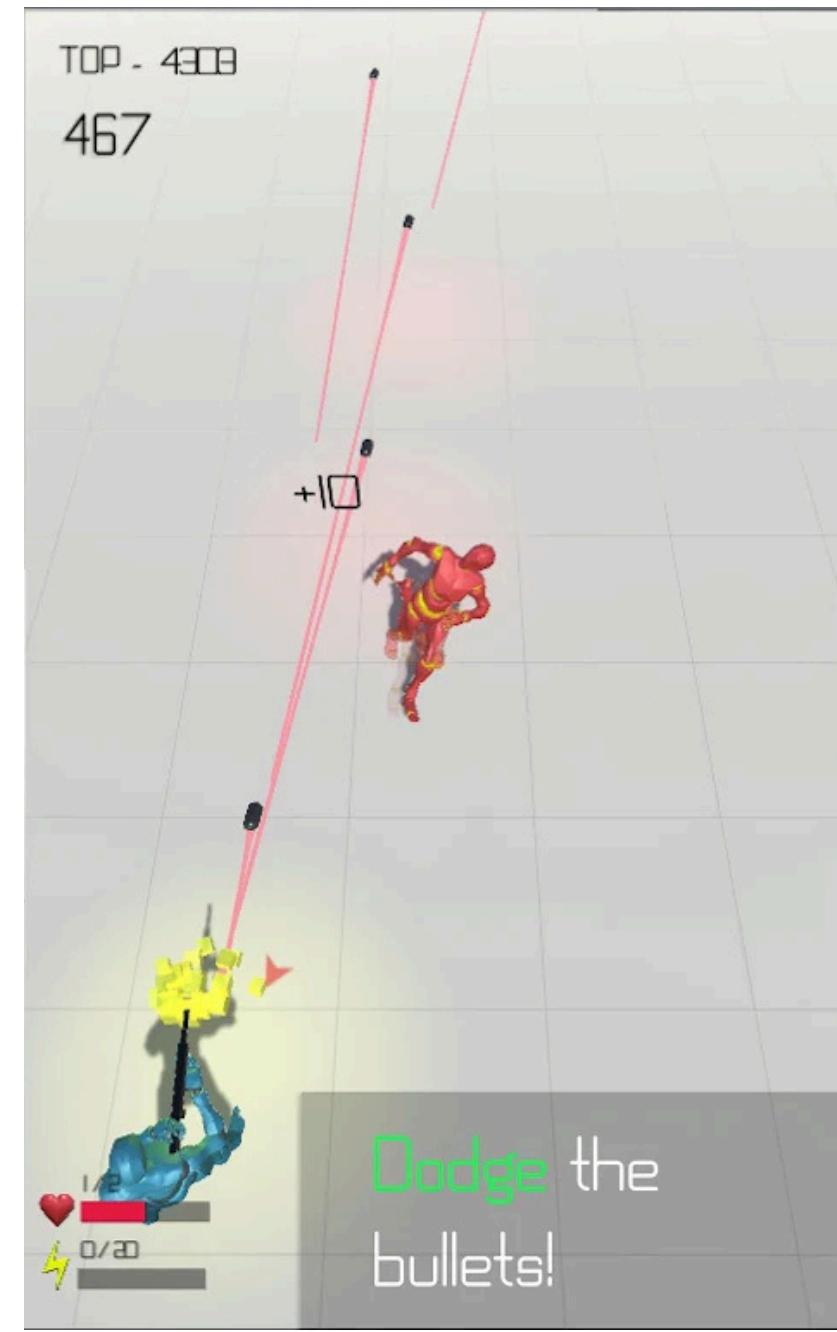
장바구니 요약, 수량 조절 기능



05 Speedy Hero - 모바일 게임 소개

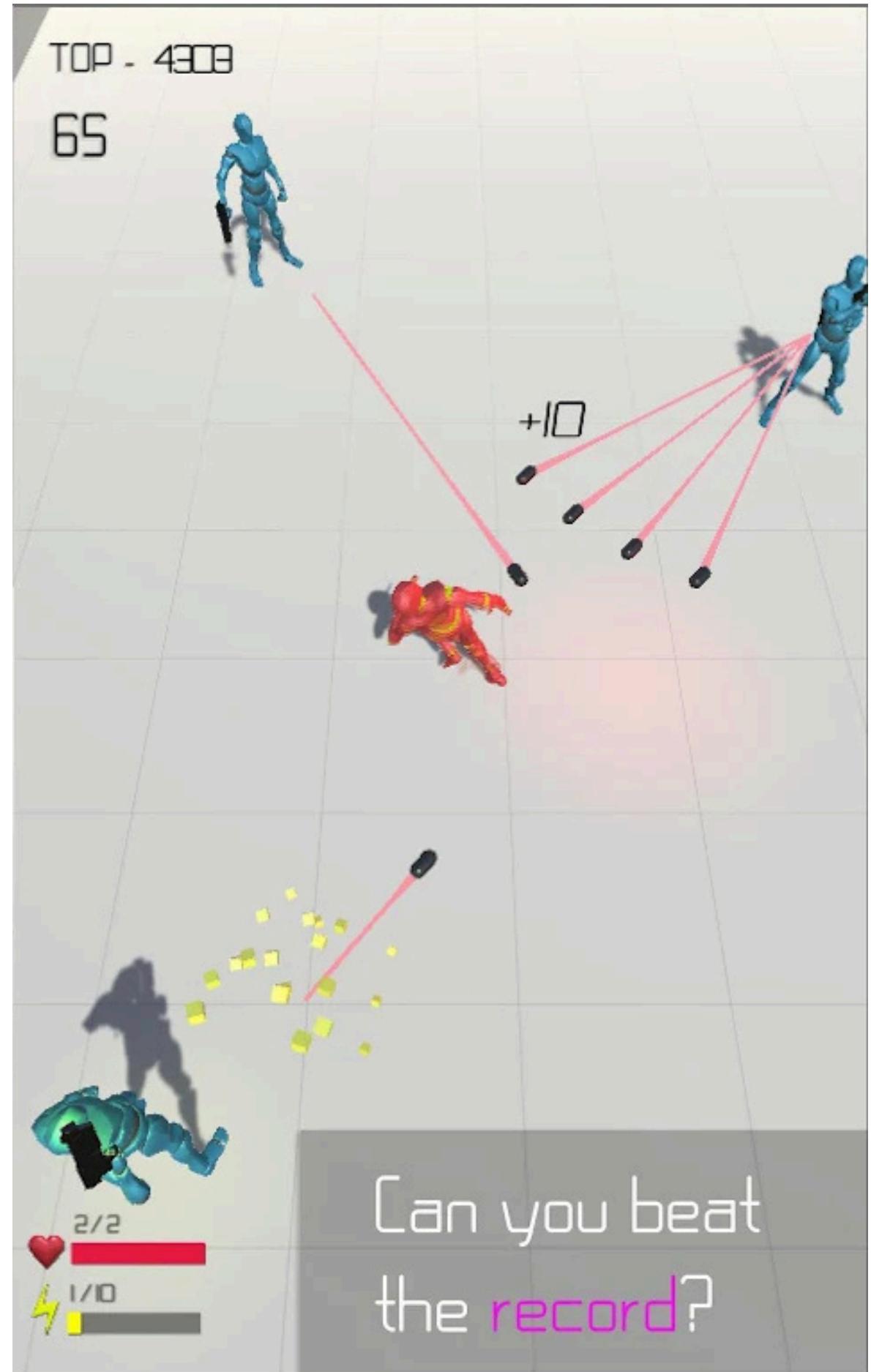
Speedy Hero는 초고속 이동이 가능한 캐릭터가 장애물을 피하며 끝없이 달리는 러너(Runner) 게임입니다. 플레이어는 속도를 조절 하며 장애물을 회피하고, AI 적들을 따돌리며 높은 점수를 기록하는 것이 목표입니다. 절제된 물리 엔진 사용, AI와의 상호작용, 그리고 최적화된 그래픽과 성능이 주요 기술적 특징입니다.

Unity 기반의 게임 개발에 필요한 물리 엔진, AI, 그래픽 최적화, 비동기 로딩, 입력 시스템, UI/UX 설계 등의 다양한 기술을 학습하고 실전에서 적용하였습니다. 이 프로젝트를 통해 기획부터 프로토타입 개발, 최적화, 배포까지 전 과정을 경험할 수 있었으며, 특히 성능 최적화와 플레이어 경험 향상을 위한 다양한 해결책을 고민하는 과정에서 개발자로서 한층 성장할 수 있었습니다.



05 Speedy Hero - 모바일 게임 소개

사용 기술	Unity (C#), NavMesh AI, Universal Render Pipeline (URP), ShaderLab, Raycasting, Object Pooling, Game Analytics, Facebook Analytics
개발 기간	기획 및 설계: 2017.01.24 ~ 2017.02.07 개발: 2017.02.08 ~ 2017.04.12 테스트 및 배포, 최적화: 2017.04.13 ~ 2017.05.04
개발인원	2명 (PM 및 디자인, 개발자)
담당역할	게임 디자인: 게임의 컨셉, 캐릭터, 레벨 디자인 등 기획 및 구현. 프로그래밍: Unity와 C#을 이용한 게임 로직 및 인터랙션 구현. 테스트 및 배포: 다양한 기기에서의 테스트를 통해 버그 수정 및 최적화 작업.
성과	다운로드 수 50,000회 돌파, 사용자 평점 4.5/5 달성. Retention: D1 45%, D7 12%
깃허브	https://github.com/whd793/speed-hero-prototype
Demo	https://www.youtube.com/watch?v=tH6Hlx6yb9I



05 Speedy Hero - 모바일 게임

핵심 기능

플레이어 이동 및 게임플레이 핵심 시스템 개발

- 고속 이동 및 충돌 처리

- 고속 캐릭터 이동: Unity의 Rigidbody 물리를 사용하여 부드럽고 현실적인 속도 기반 모션 구현
- 캐릭터 능력: Rigidbody.AddForce와 Raycasting을 사용한 대시, 회피 메카닉
- Raycast를 활용한 고속 이동 시 충돌 감지 보완 (Continuous Collision Detection 적용)
- DashController.cs를 개발하여 순간 가속(Dash) 기능 및 재사용 대기시간(Cooldown) 적용
- 동적 장애물 생성: 장애물이 절차적으로 생성되어 매 실행마다 고유한 경험 제공

AI 및 적(Enemy) NPC 시스템 개발

- AI Pathfinding 및 추격 시스템

- Unity의 NavMesh를 활용하여 AI가 동적으로 플레이어를 추격하도록 구현
- Finite State Machine (FSM) 적용 (Idle, Chase, Attack 상태 전환)
- 장애물과의 충돌을 피하도록 NavMeshObstacle 추가
- 동적 난이도 조절(DDA): 플레이어 성능에 기반한 적 스폰 비율 조절

- AI 반응 속도 및 동작 최적화

- AI가 특정 거리 이상 떨어지면 Object Pooling을 사용해 비활성화하여 성능 최적화

05 Speedy Hero - 모바일 게임

핵심 기능

그래픽 및 성능 최적화

- 최적화된 렌더링 파이프라인: 성능 향상을 위해 Unity의 URP(Universal Render Pipeline) 사용
- ShaderLab 기반의 커스텀 셰이더 개발
 - 초고속 이동 시 모션 블러(Motion Blur) 효과 적용
 - 속도에 따른 트레일 효과(Afterimage Shader) 구현
- LOD (Level of Detail) 시스템 적용
 - 멀리 있는 오브젝트의 디테일을 낮추어 성능 개선
- 비동기(Asynchronous) 씬 로딩 적용
 - 배경 및 맵을 비동기 방식으로 로딩하여 부드러운 씬 전환 구현
- Object Pooling 기법 적용
 - 빈번한 생성/소멸 대신 오브젝트 재사용을 위한 폴 매니저 활용
 - 반복적으로 생성되는 오브젝트를 폴에서 재사용하여 CPU 및 메모리 사용량 감소

05 Speedy Hero - 모바일 게임

핵심 기능

입력 시스템

- 크로스-플랫폼 지원: 키보드, 마우스, 게임패드, 터치 입력을 위한 Unity의 New Input System 사용
- 제스처 인식: 모바일 플랫폼용 스와이프 기반 컨트롤

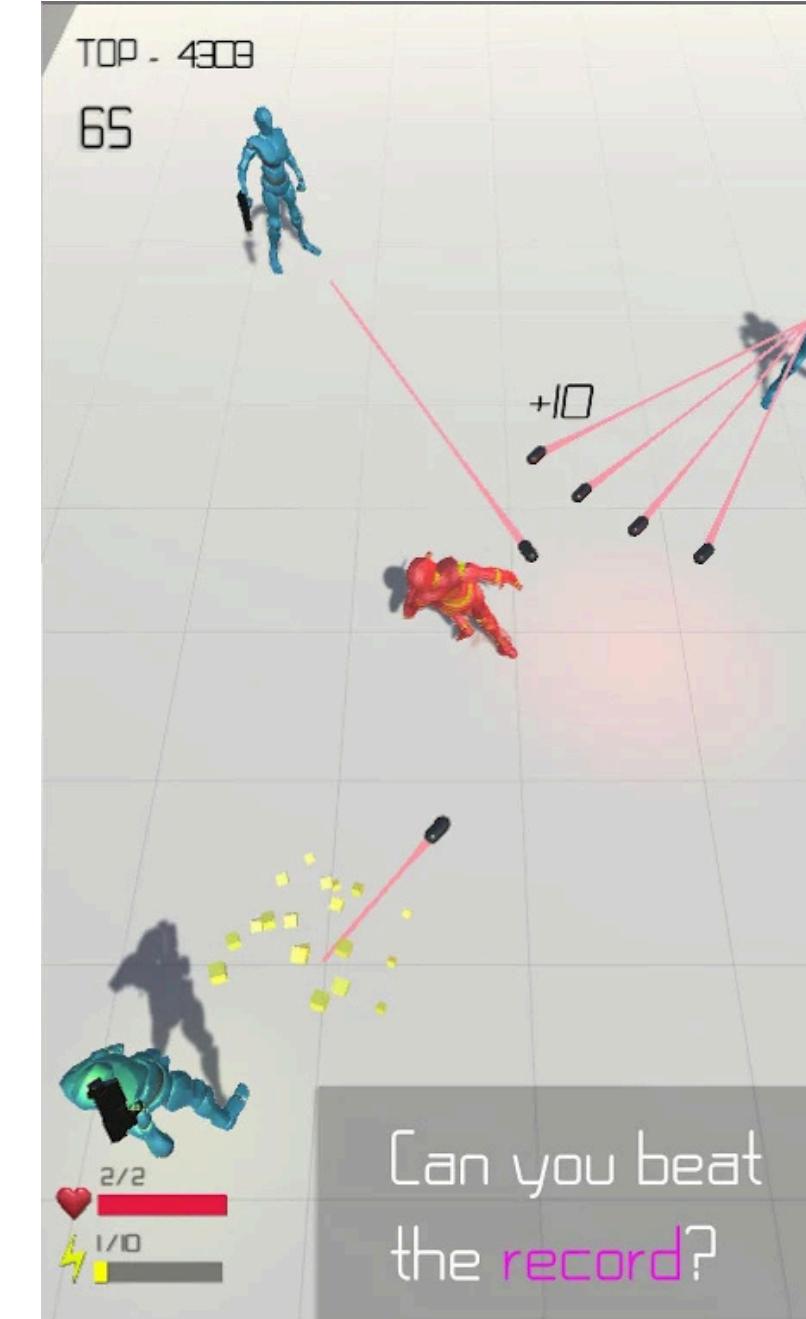
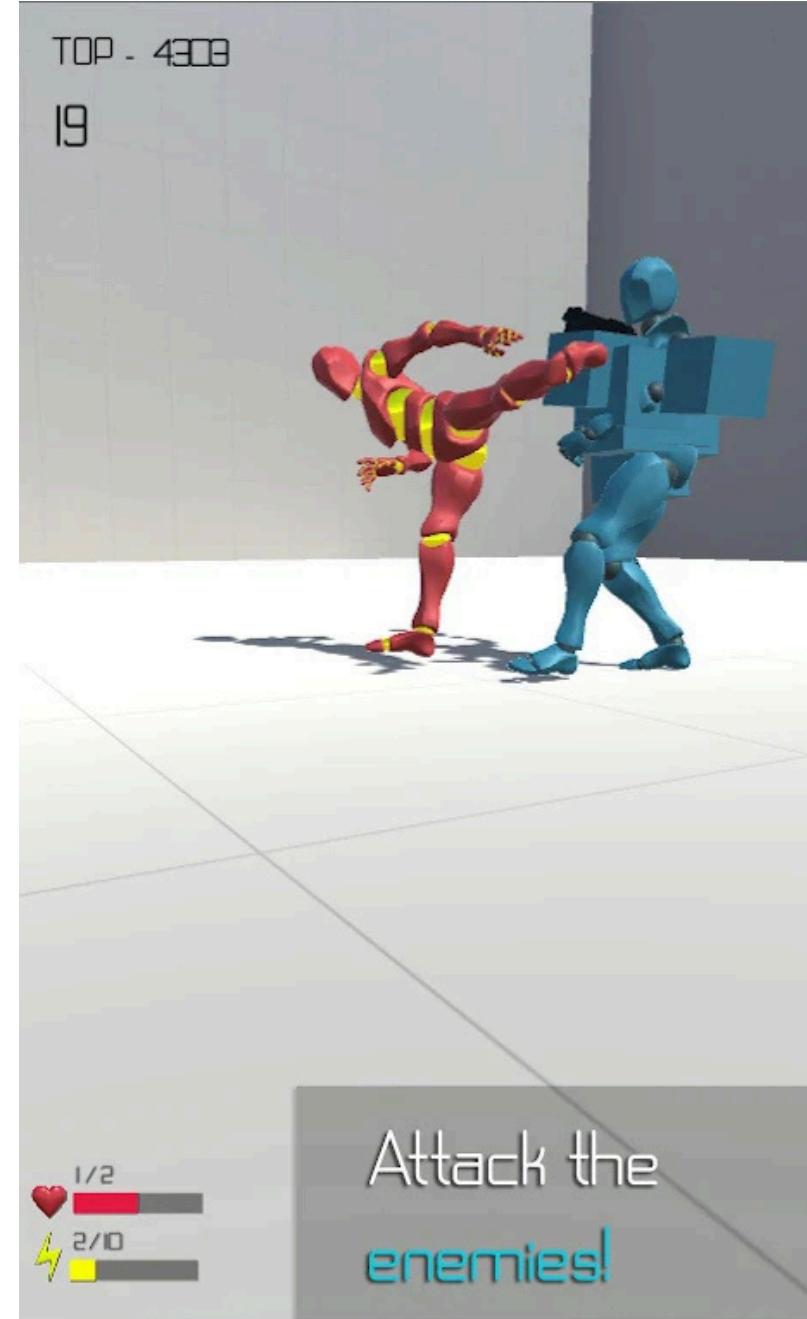
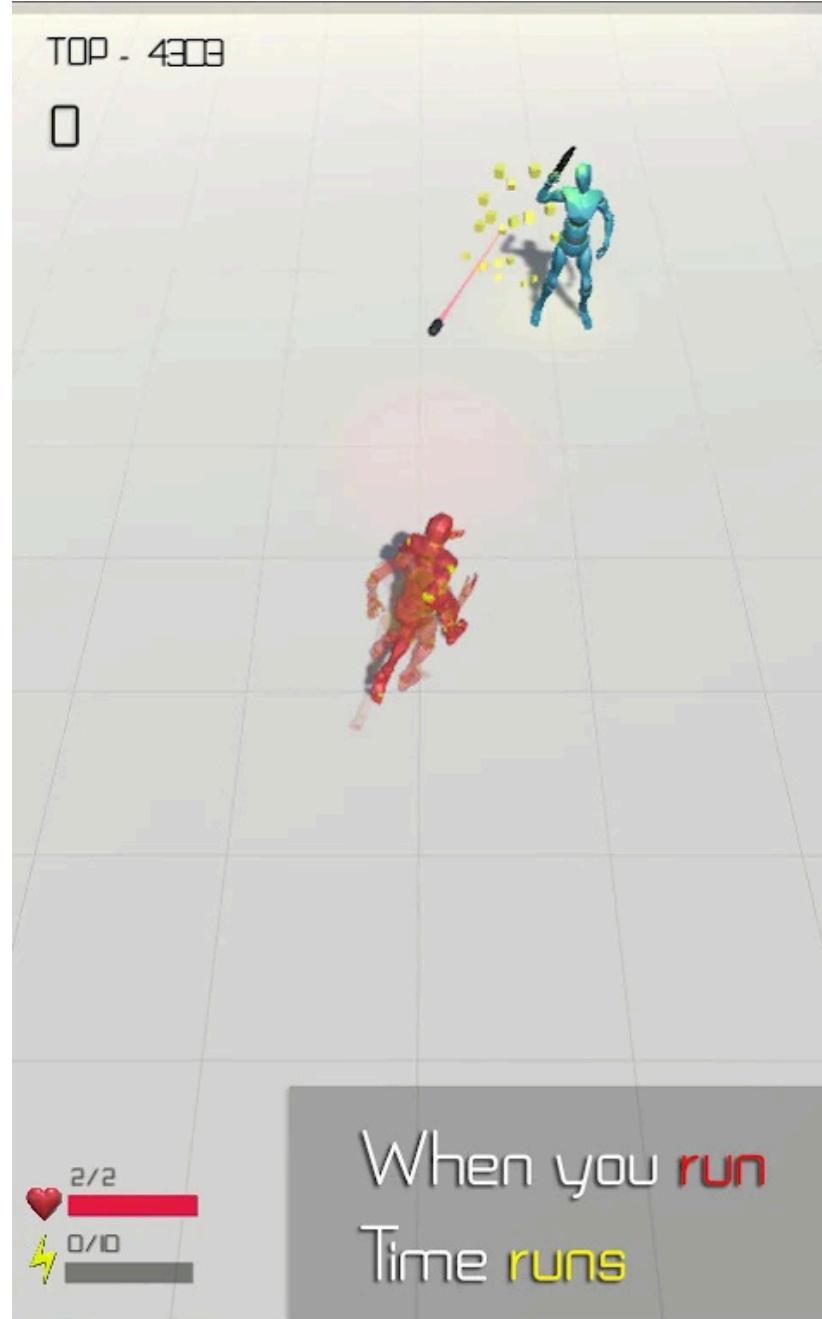
오디오 및 사운드 이펙트

- 3D 공간 오디오: 몰입형 사운드 위치 지정을 위한 AudioSource.spatialBlend 구현
- 적응형 음악 시스템: 게임 강도에 따라 배경 음악 템포 증가

UI/UX 개선 및 모바일 환경 대응

- UIManager.cs 개발
 - 점수 시스템, 속도 표시 UI, 플레이어 상태 정보 표시
 - HealthSystem.cs 개발하여 체력 및 속도 감소 시스템 구현
- 모바일 환경 최적화
 - Unity의 New Input System 활용하여 터치 및 제스처 컨트롤 구현
 - 모바일 화면 해상도 대응 및 최적화

05 Speedy Hero - 모바일 게임



06 Hyper Casual Game 프로젝트 소개

Unity 엔진을 활용하여 22개의 하이퍼 캐주얼 게임을 성공적으로 개발했습니다. 개발 과정에서는 철저한 시장 조사를 실시하고, 신속한 prototyping과 반복적인 개발 주기를 통해 사용자 피드백을 기반으로 게임플레이 메커니즘을 지속적으로 개선했습니다. C# 프로그래밍을 사용하여 정교한 시스템을 구현했으며, 모듈화되고 재사용 가능한 코드 아키텍처를 만들어 개발 프로세스를 효율화하고 높은 코드 품질 표준을 유지했습니다.

Game Analytics와 Facebook Analytics를 활용한 철저한 A/B 테스트와 데이터 분석을 통해 사용자 행동 패턴과 수익화 전략에 대한 깊은 통찰력을 얻을 수 있었습니다. 이러한 데이터 기반 접근 방식을 통해 주요 성과 지표를 최적화할 수 있었고, 그 결과 인상적인 리텐션율, user session과 사용자 참여도를 달성했습니다.

이 프로젝트는 게임 개발에서의 제 기술적 역량뿐만 아니라 시장 수요를 이해하고 대응하는 능력을 배웠습니다. 이러한 경험을 통해 데이터 분석, 사용자 경험 디자인, 비즈니스 전략 수립 능력이 한층 강화되었습니다.



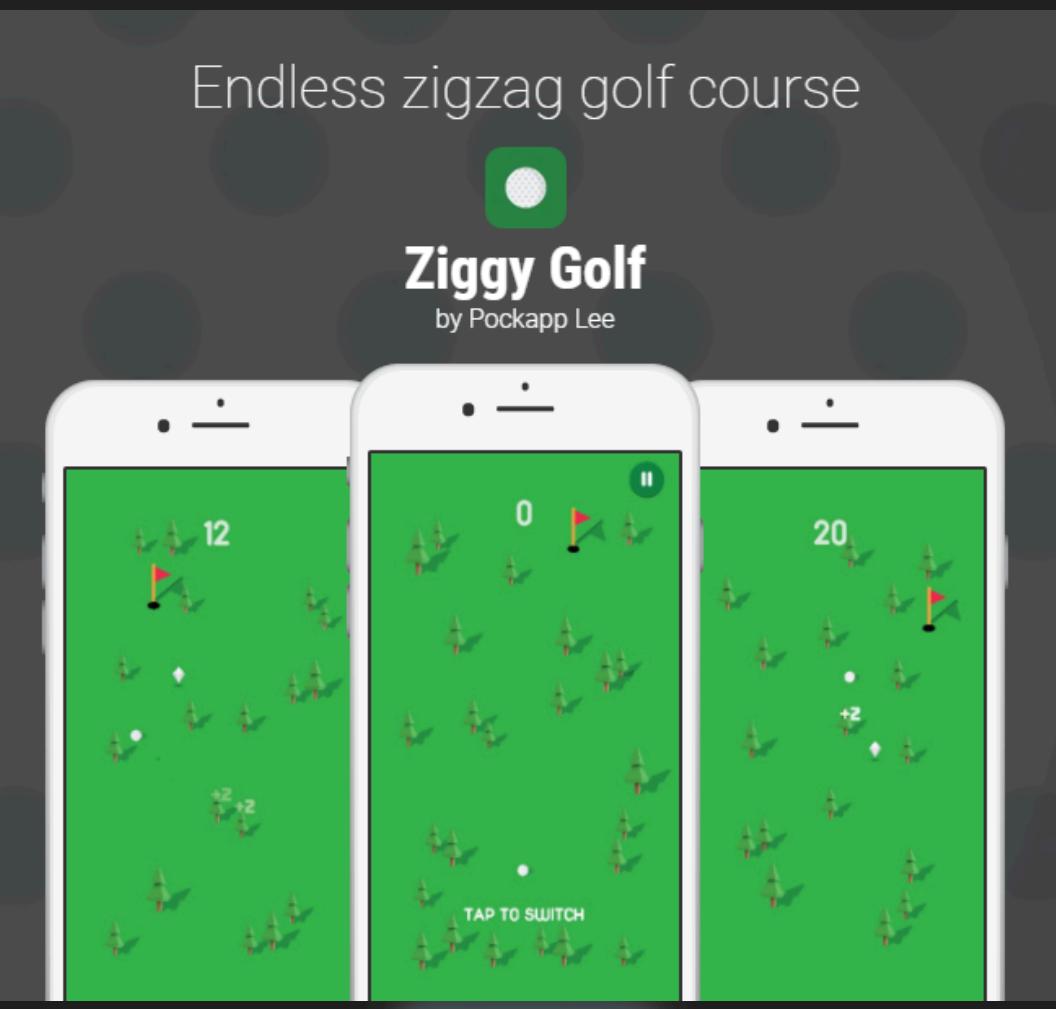
06 Hyper Casual Game 프로젝트

발환경

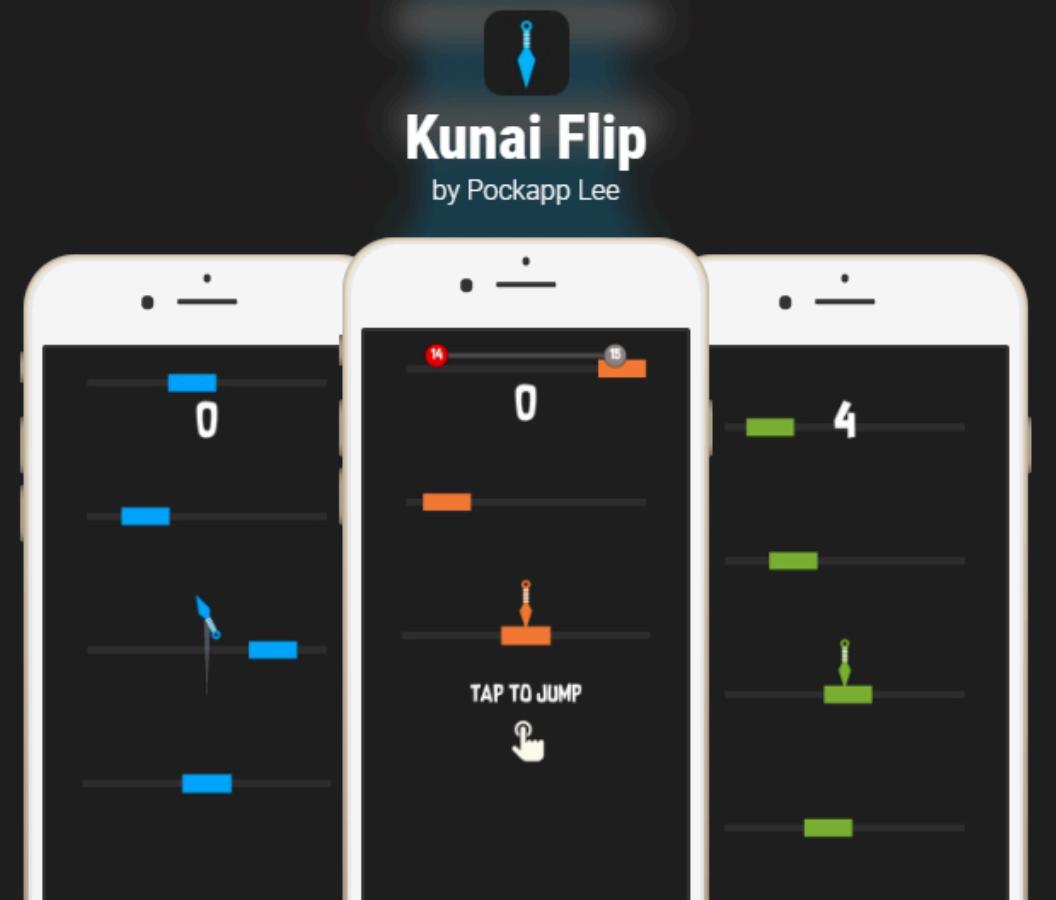
사용 기술 Unity (C#), Game Analytics, Facebook Analytics

개발 기간 2023.08 ~ 2024.03

개발인원 2명 (PM 및 디자인, 개발자)



How high can you flip

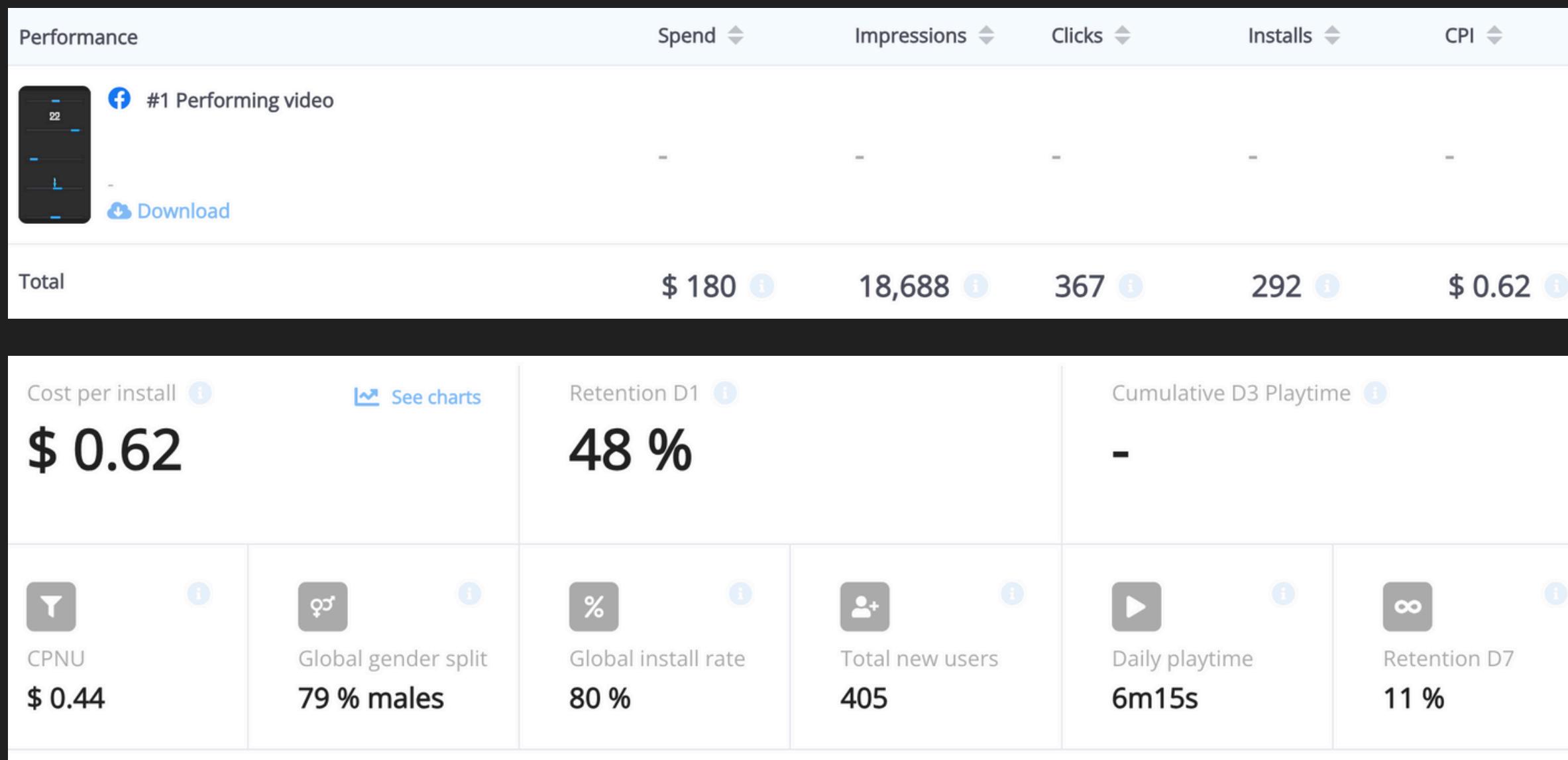


06 Hyper Casual Game 프로젝트

Case Study

데이터 기반의 의사결정을 위해 핵심 성과 지표(KPI)를 설정하고 추적하는 시스템을 구축했습니다. Game Analytics와 Facebook Analytics를 활용하여 사용자 행동을 분석했고, A/B 테스트를 통해 게임의 주요 요소들을 최적화했습니다. 그 결과, D1 리텐션을 48%까지 끌어올렸습니다. 또한 세션 시간은 평균 5분으로 증가시켰으며, 광고 최적화를 통해 US 시장 CPI를 \$0.62까지 낮추는데 성공했습니다.

Kunai Flip Analytics

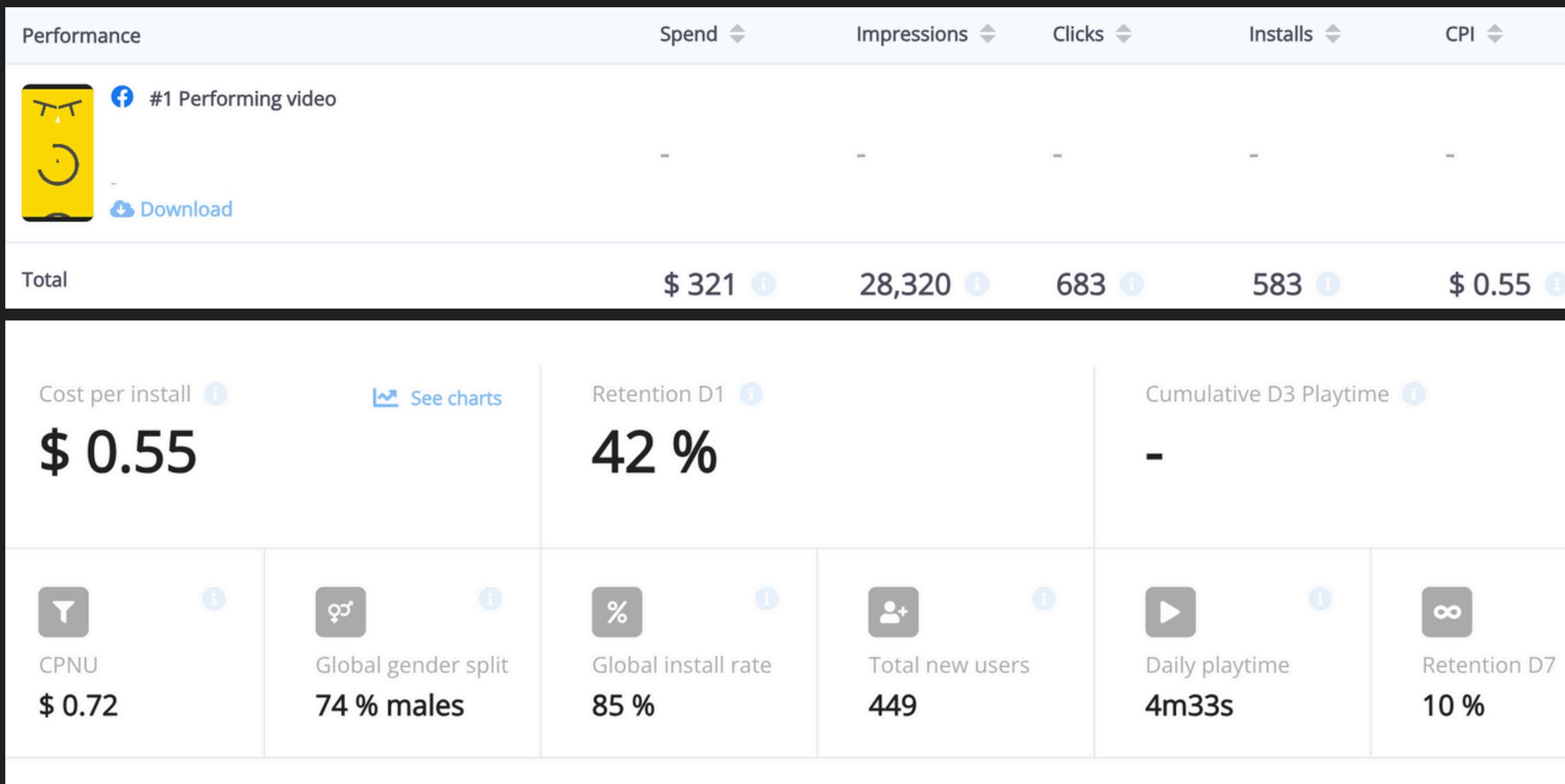


06 Hyper Casual Game 프로젝트

Case Study

데이터 기반의 의사결정을 위해 핵심 성과 지표(KPI)를 설정하고 추적하는 시스템을 구축했습니다. Game Analytics와 Facebook Analytics를 활용하여 사용자 행동을 분석했고, A/B 테스트를 통해 게임의 주요 요소들을 최적화했습니다. 그 결과, D1 리텐션을 42%까지 끌어올렸습니다. 또한 세션 시간은 평균 5분으로 증가시켰으며, 광고 최적화를 통해 US 시장 CPI를 \$0.55까지 낮추는데 성공했습니다.

Dashy Arrow Analytics

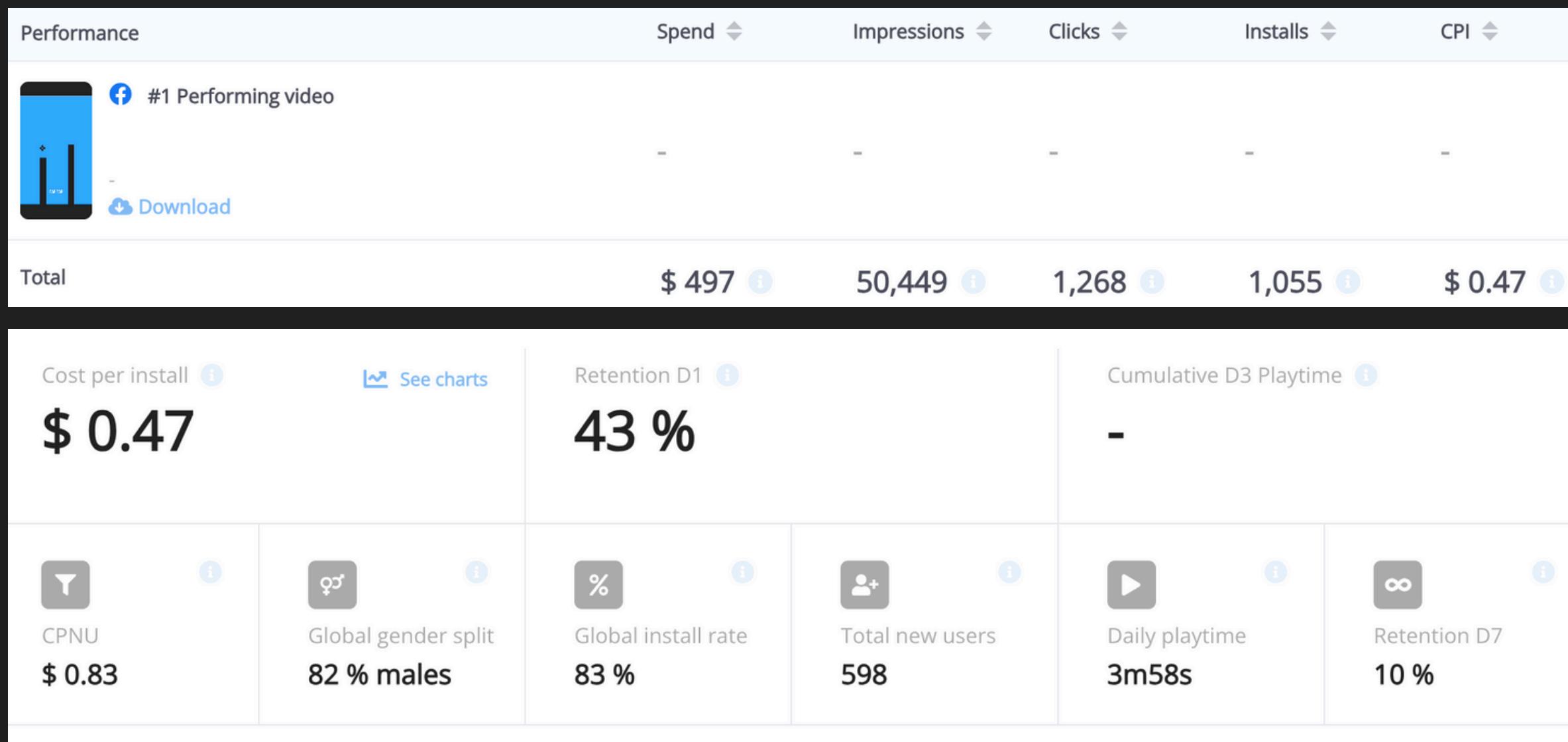


06 Hyper Casual Game 프로젝트

Case Study

데이터 기반의 의사결정을 위해 핵심 성과 지표(KPI)를 설정하고 추적하는 시스템을 구축했습니다. Game Analytics와 Facebook Analytics를 활용하여 사용자 행동을 분석했고, A/B 테스트를 통해 게임의 주요 요소들을 최적화했습니다. 그 결과, D1 리텐션을 43%까지 끌어올렸습니다. 또한 세션 시간은 평균 5분으로 증가시켰으며, 광고 최적화를 통해 US 시장 CPI를 \$0.47까지 낮추는데 성공했습니다.

Kunai Flip Analytics



06 Hyper Casual Game 프로젝트

게임 최적화 및 기능적 문제 해결

게임 개발 속도에 영향을 미치고, 게임 성능(특히 확장 시 KPI)에 영향을 주는 기능들 최적화

퀄리티 설정 최적화

- 대부분 게임은 높은 설정이 필요하지 않음
- 비용이 많이 드는 프로젝트에서는 그림자를 비활성화하고 캐릭터 아래 가짜 원형 이미지로 대체

LOD(Level of Detail) 활용

- 카메라 거리에 따라 메시를 단순화하는 스크립트
- 성능 측면에서 많은 리소스 절약 가능

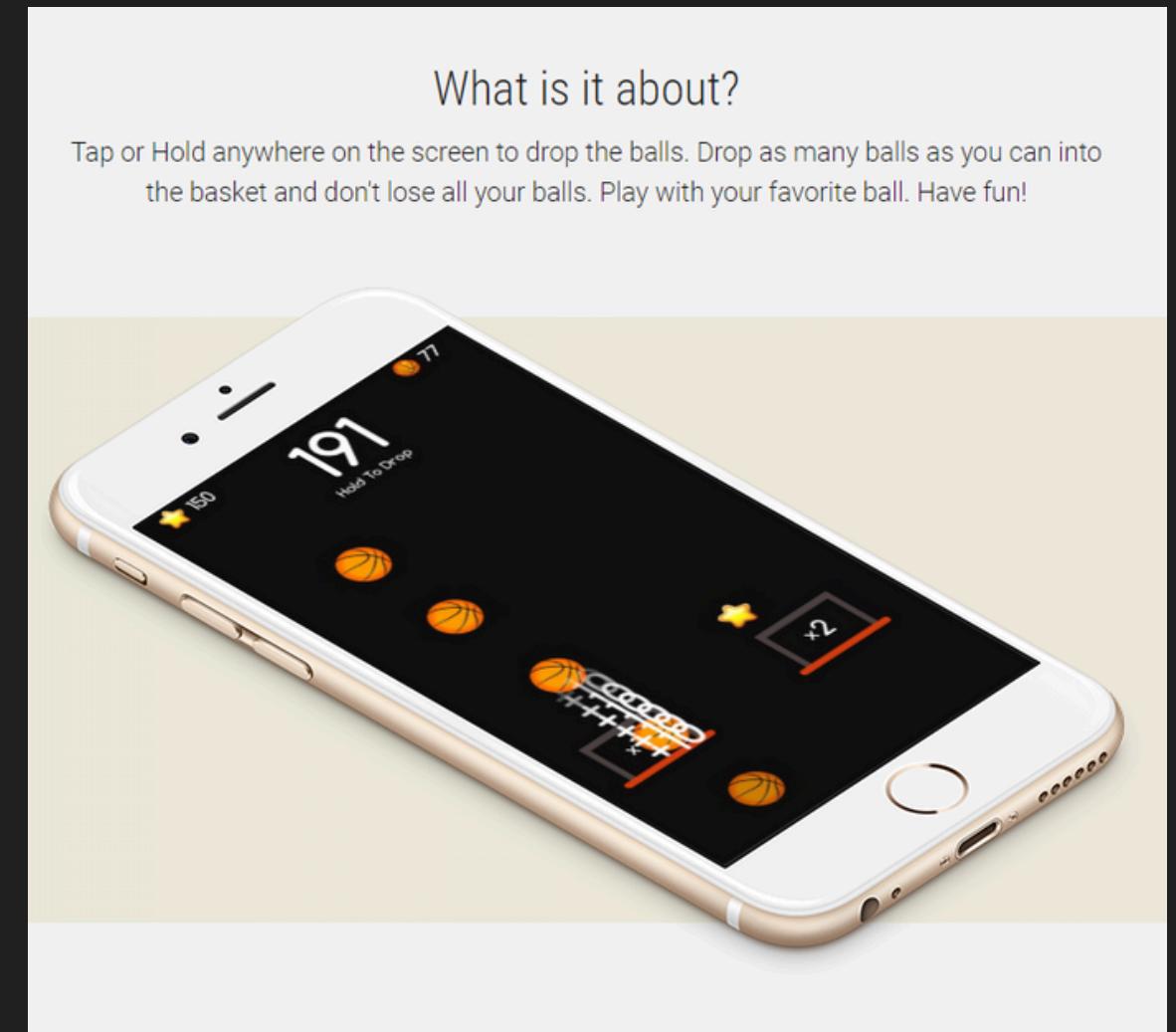
캐싱 사용

- Unity 컴포넌트 getter 함수는 참조를 가져오는 데 한 번만 사용

필요할 때만 코드 실행

2의 제곱 텍스처 사용

- UI 요소도 포함하여 항상 2의 제곱 텍스처 사용 (그렇지 않으면 압축 비활성화)
- 예시: 작은 이미지 - 128x128, 256x256, 중간 이미지 - 512x512, 1024x1024
- 화면에서 차지하는 영역에 비례하여 이미지 해상도 결정
- 2의 제곱이 아닌 이미지는 일반적으로 기기 메모리에 4배 더 부담



06 Hyper Casual Game 프로젝트

게임 최적화 및 기능적 문제 해결

Android 햅틱 최적화

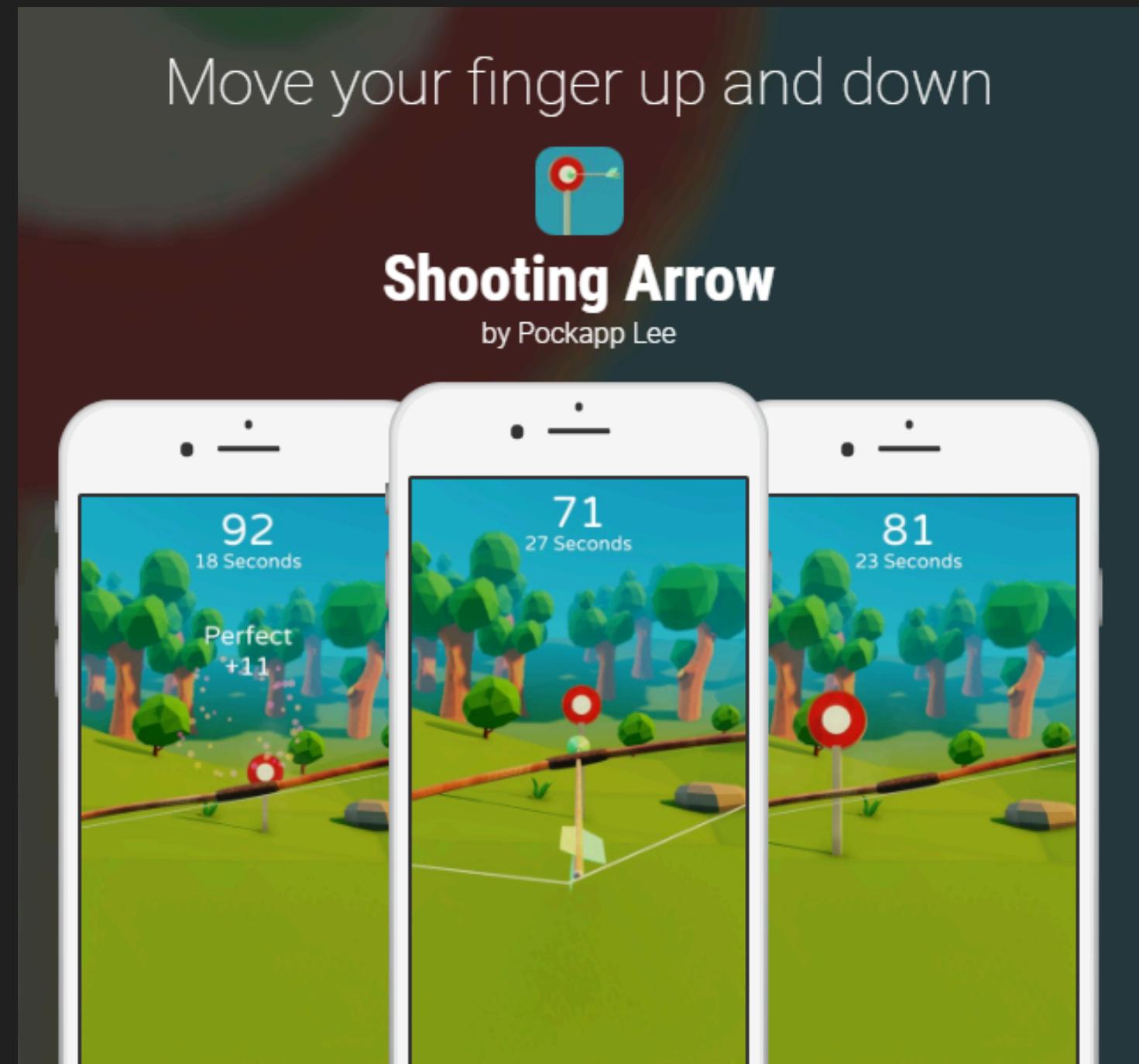
- 많은 Android 기기는 햅틱으로 과잉반응하여 발열이나 지연 발생
- Android에서는 가벼운 햅틱만 사용, 연속 햅틱 사용 금지
- 두 햅틱 호출 사이에 최소 0.5초 타이머 설정

프로파일러 사용

- Windows -> Analysis -> Profile에서 접근
- CPU 사용량으로 성능 문제 파악
- 렌더링, 물리 또는 그래픽 사용량 문제 발견
- 정밀 조사를 위한 Deep Profile 활용

압축 도구

- 많은 드로우 콜은 Android에서 CPU 및 성능 문제 발생
- 장면의 메시 병합으로 드로우 콜 수 감소
- 필요시 결합된 메시 "Release" 가능
- 카메라 뒤 객체 비활성화를 위해 하나의 큰 메시만 사용하는 것은 피할 것



06 Hyper Casual Game 프로젝트

게임 최적화 및 기능적 문제 해결

물리 엔진 최적화

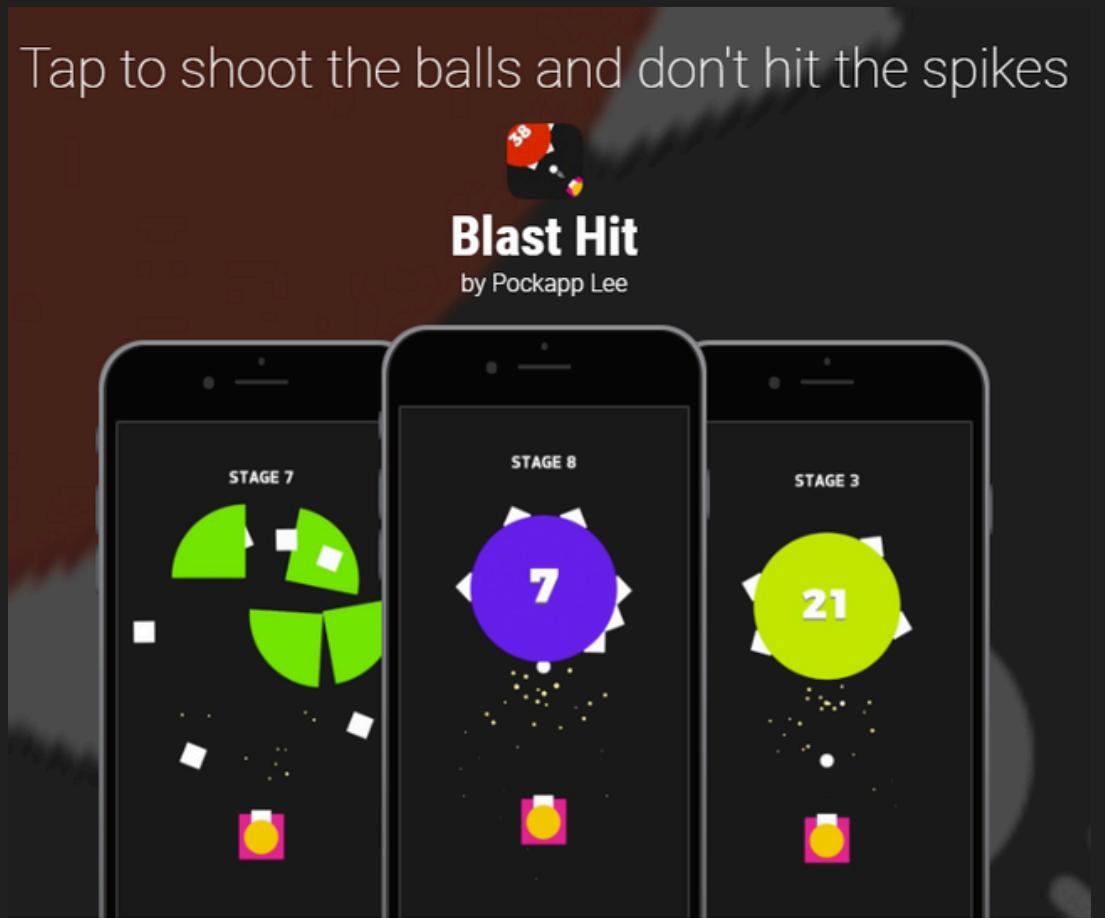
- 단순한 콜라이더로 충분할 때 메시 콜라이더 사용 피하기
- 상호작용 불가능한 장식 요소에서 콜라이더와 Rigidbody 제거
- 상호작용에 따라 레이어로 요소 그룹화
- 물리 정밀도가 중요하지 않은 경우 Fixed Timestep 및 Maximum Allowed Timestep 증가

물리 설정

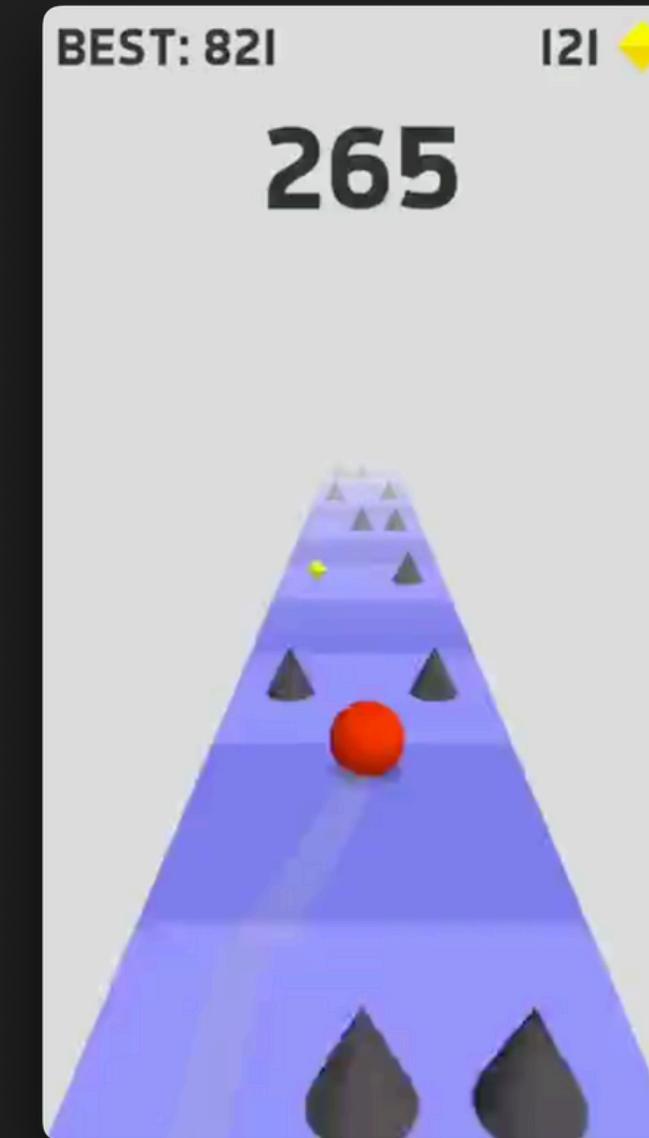
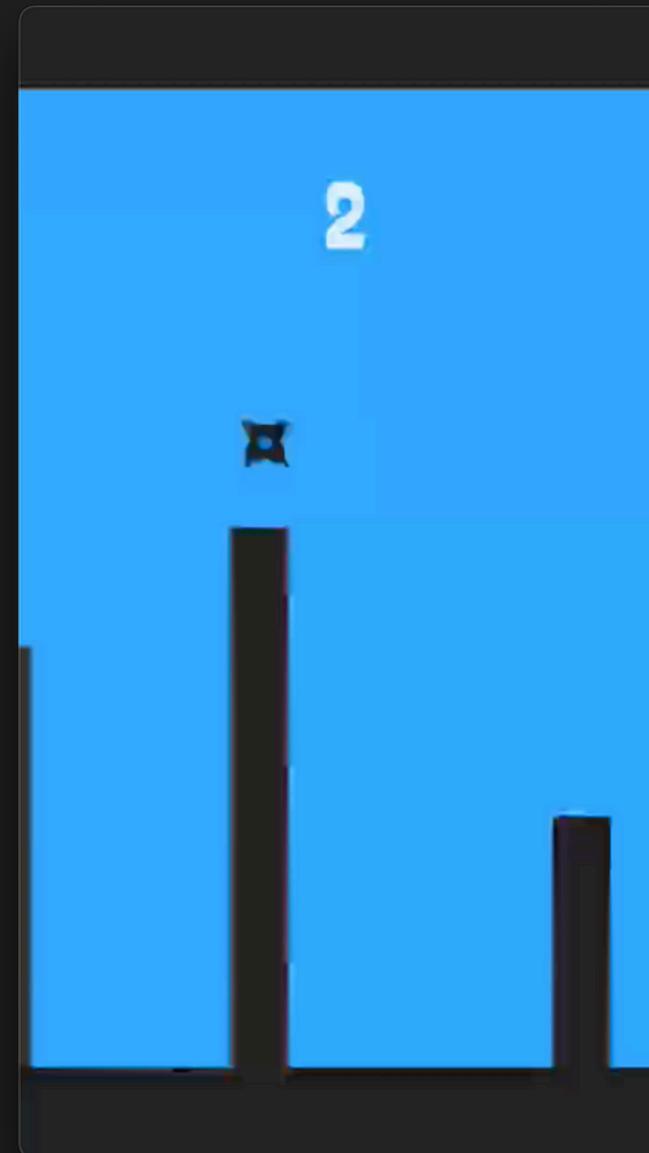
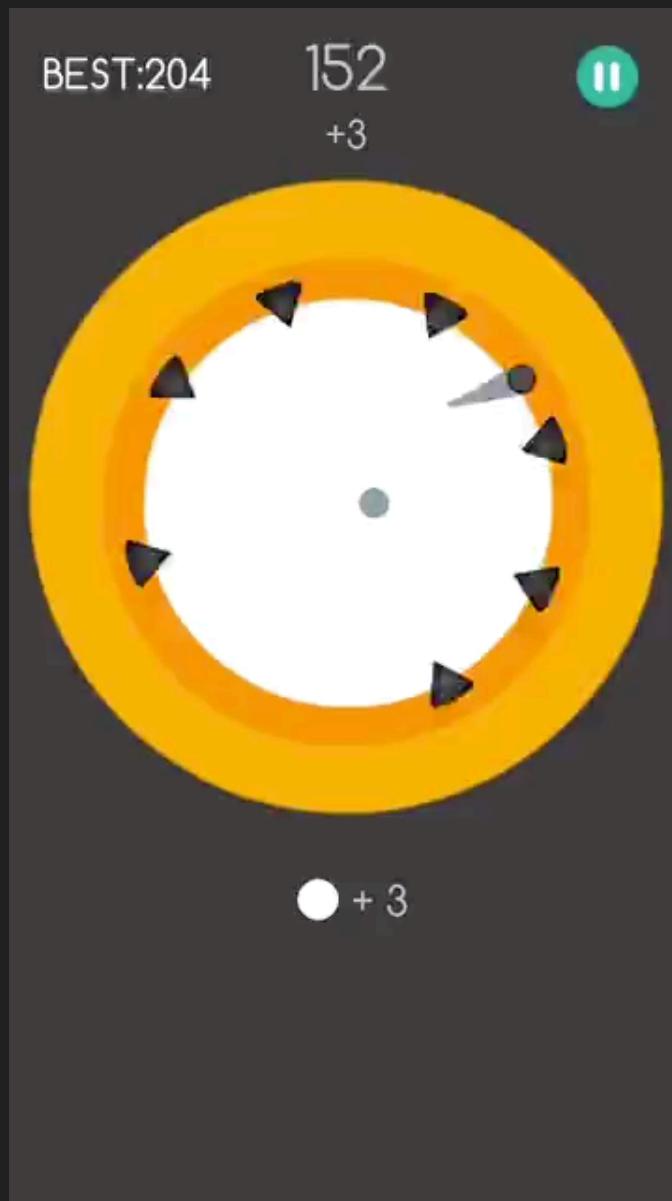
- 중력 설정 변경 최소화 (변경이 필요하면 게임 스케일 재검토)
- 성능 향상을 위해 가능한 한 1:1 스케일의 콜라이더 유지
- 상호작용하지 않는 레이어 간 불필요한 계산 피하기 위한 충돌 매트릭스 설정

모범 사례

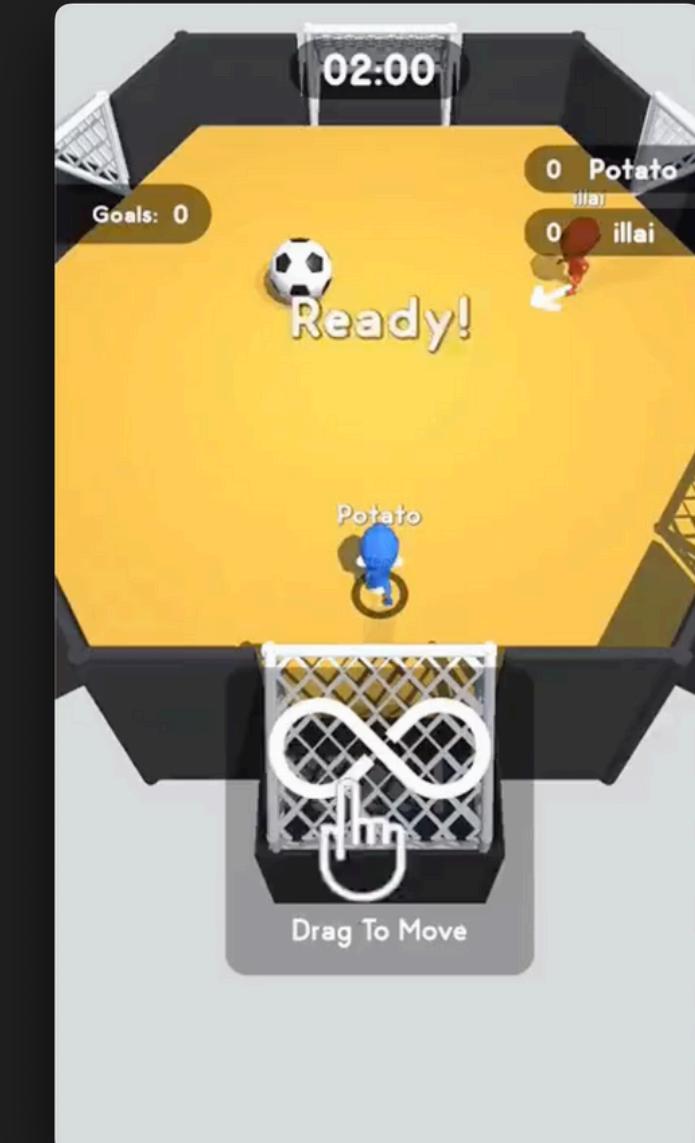
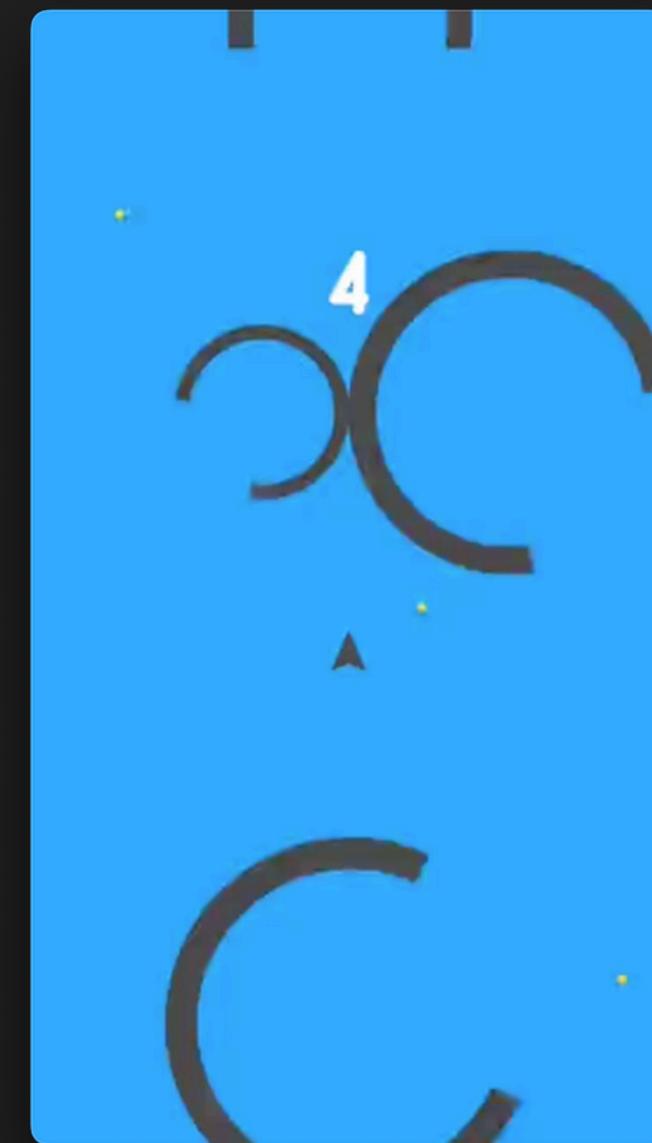
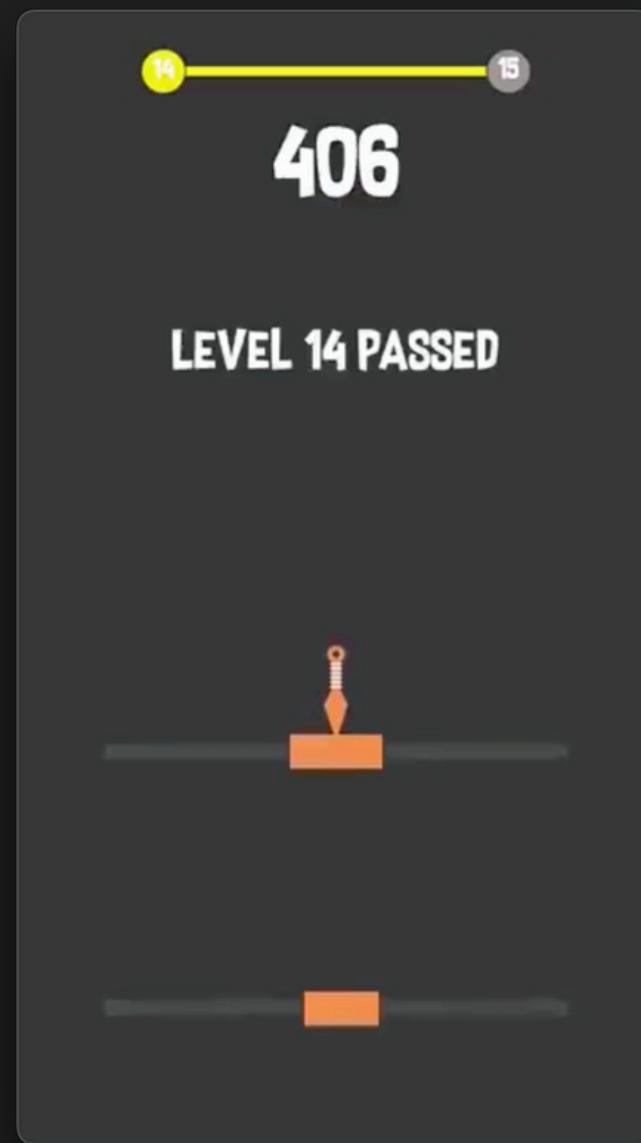
- 게임플레이 중 반복 객체는 인스턴스화/제거 대신 객체 풀링 사용
- 플레이어 설정에서 Dynamic Batching 활성화, 작은 요소에 반복 사용되는 셰이더에 GPU 인스턴싱 활성화
- 무거운 계산은 로딩 시간에 처리하여 유동적인 게임플레이 확보
- 기기별 게임 최적화 (FPS가 25 이하로 유지될 경우 그림자/무거운 파티클 비활성화, 배경 요소 숨기기 등)
- 가비지 컬렉션 유발하는 메모리 누수 추적
- 사용하지 않는 요소 제거 (미사용 코드, 오래된 에셋, 미사용 플러그인)



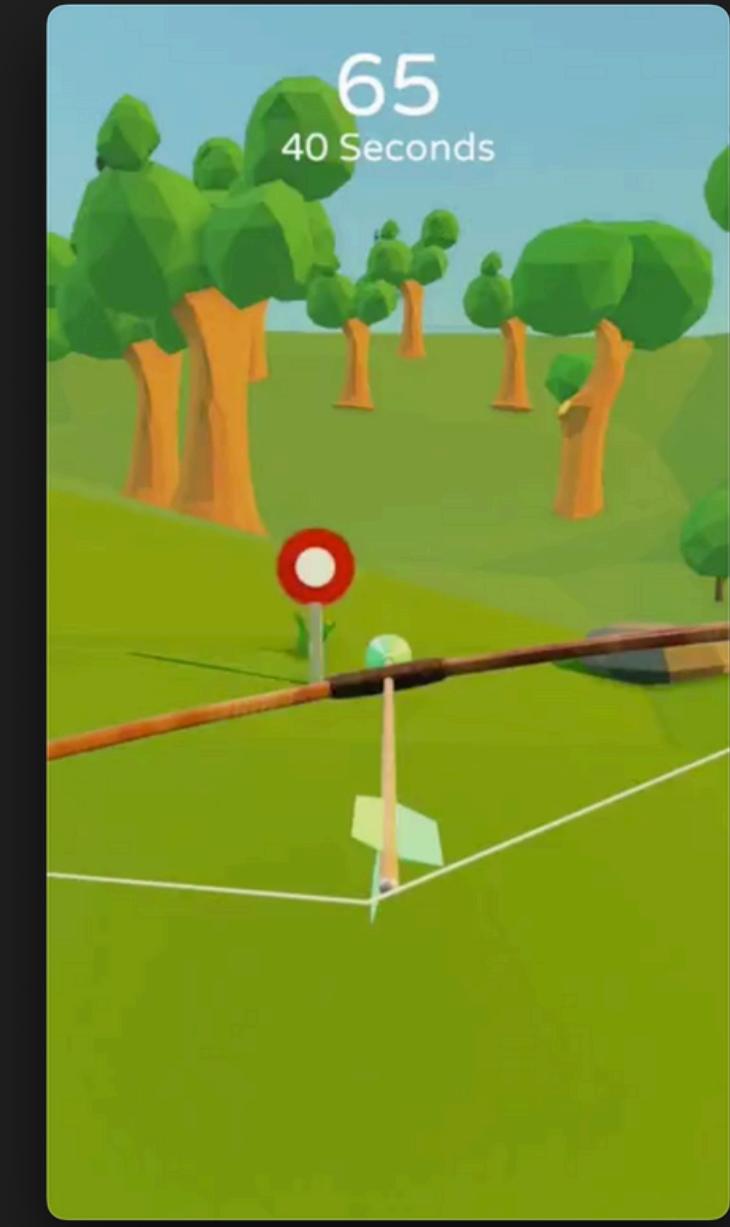
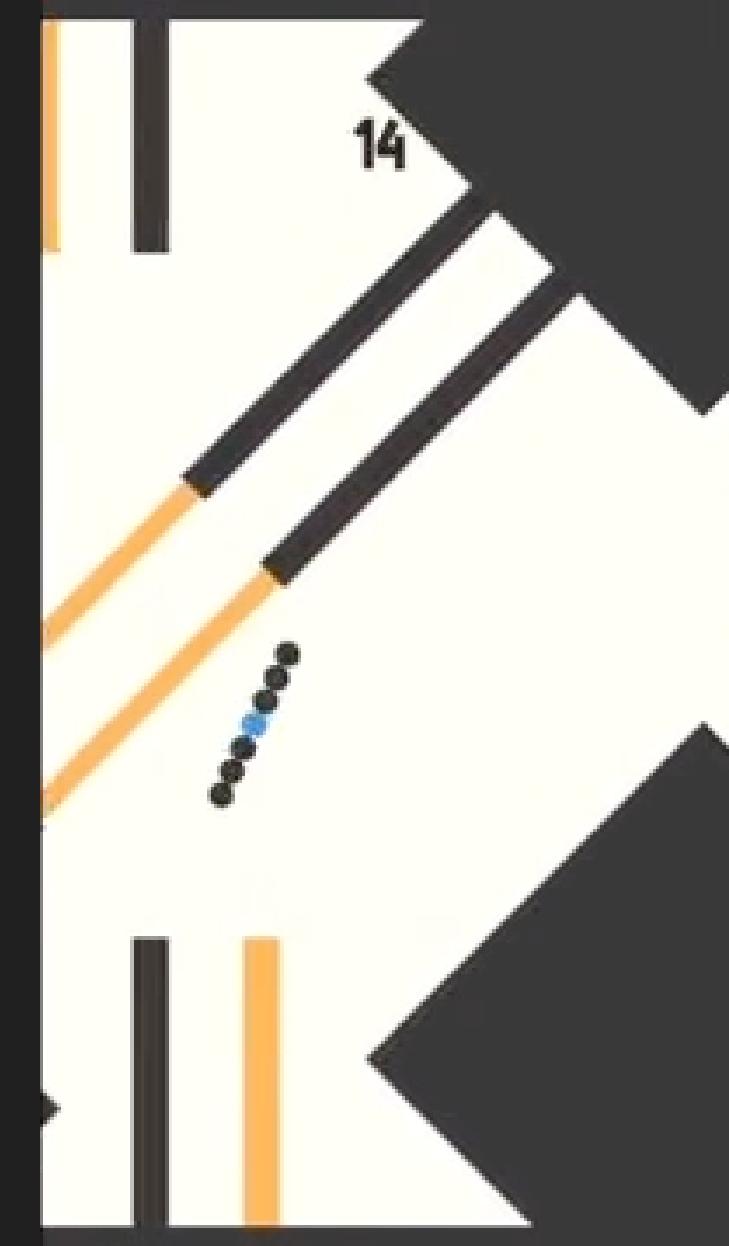
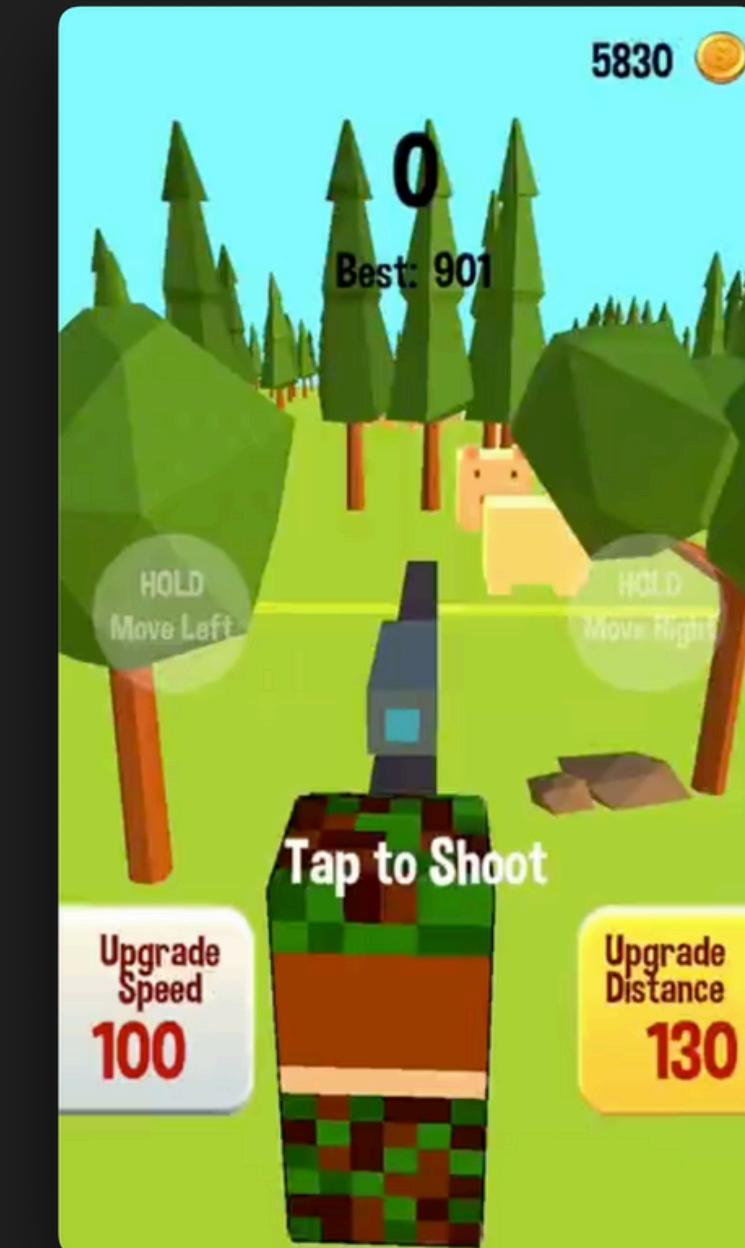
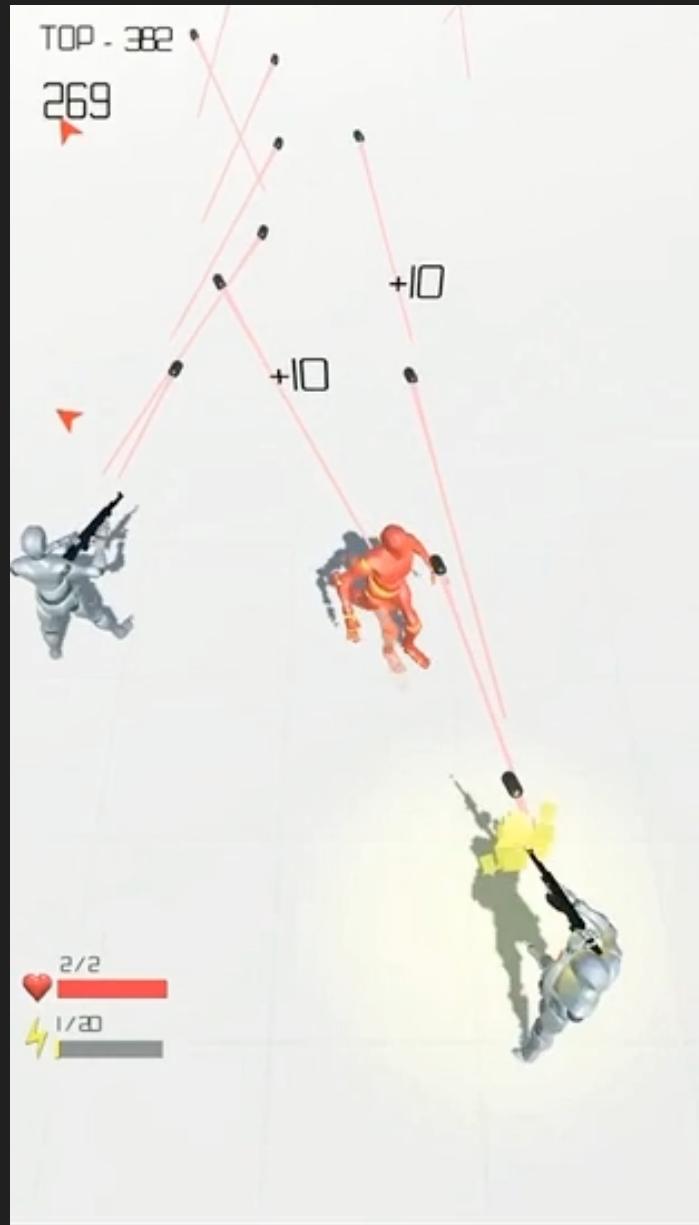
06 Hyper Casual Game 프로젝트



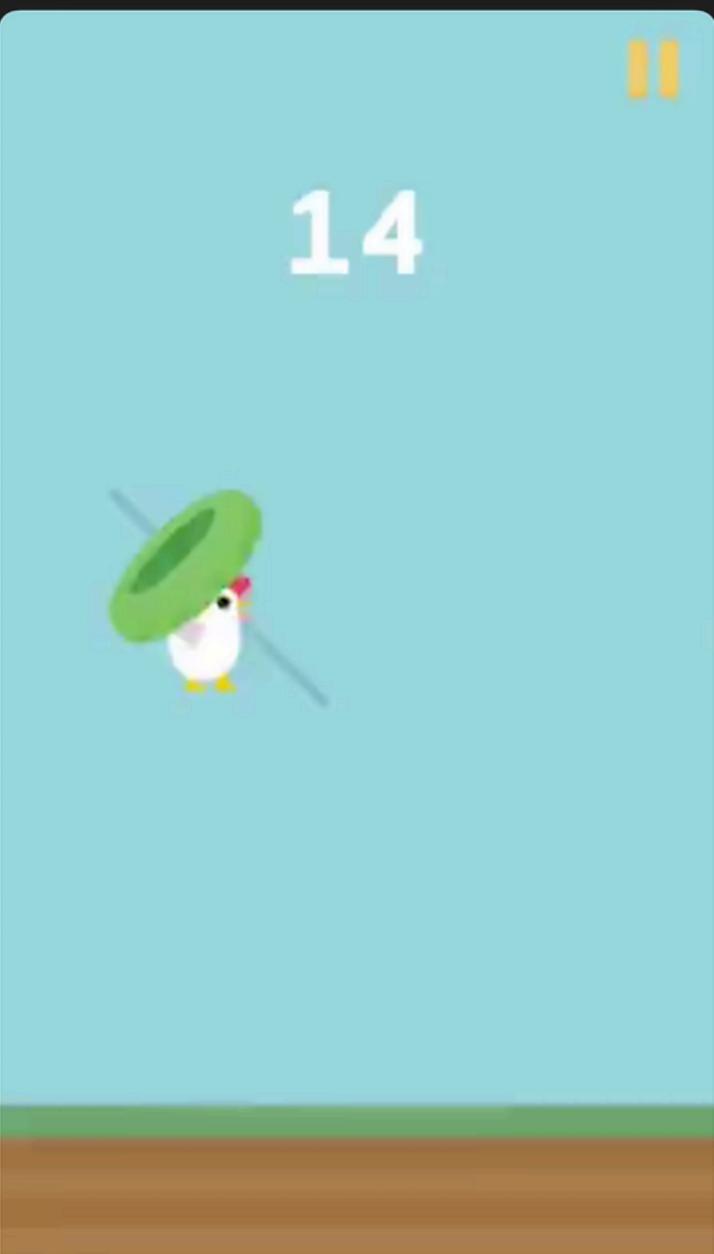
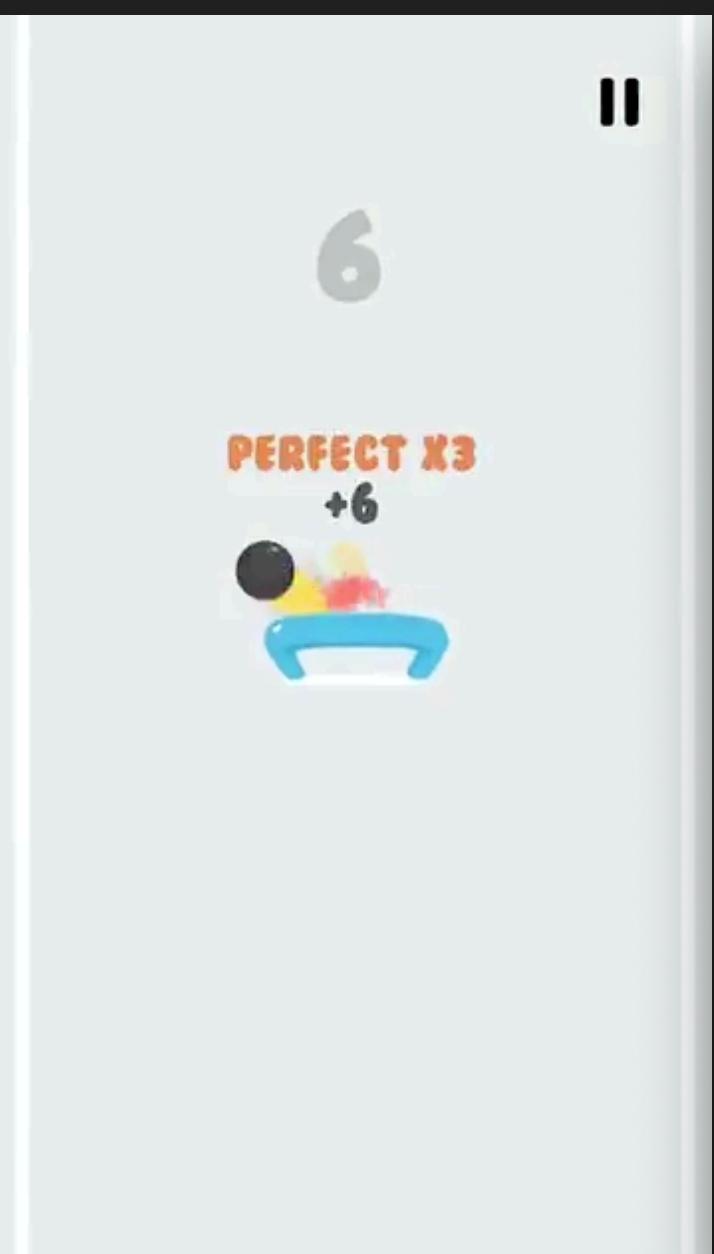
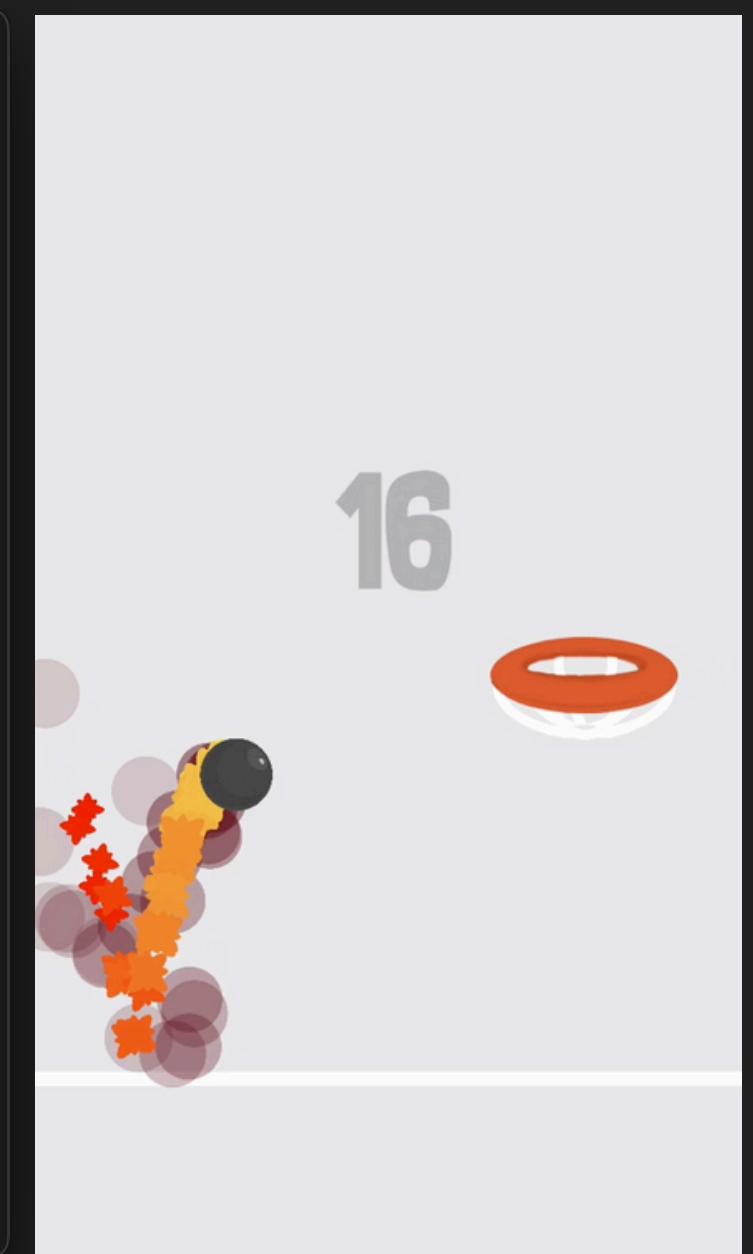
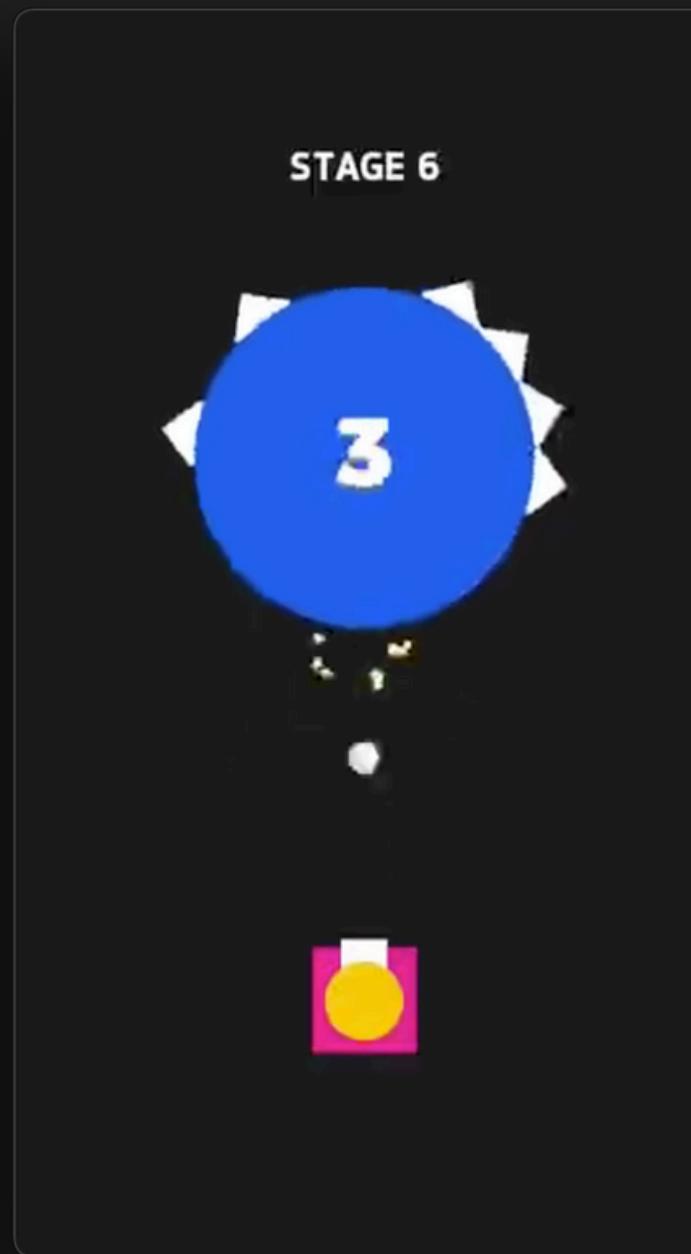
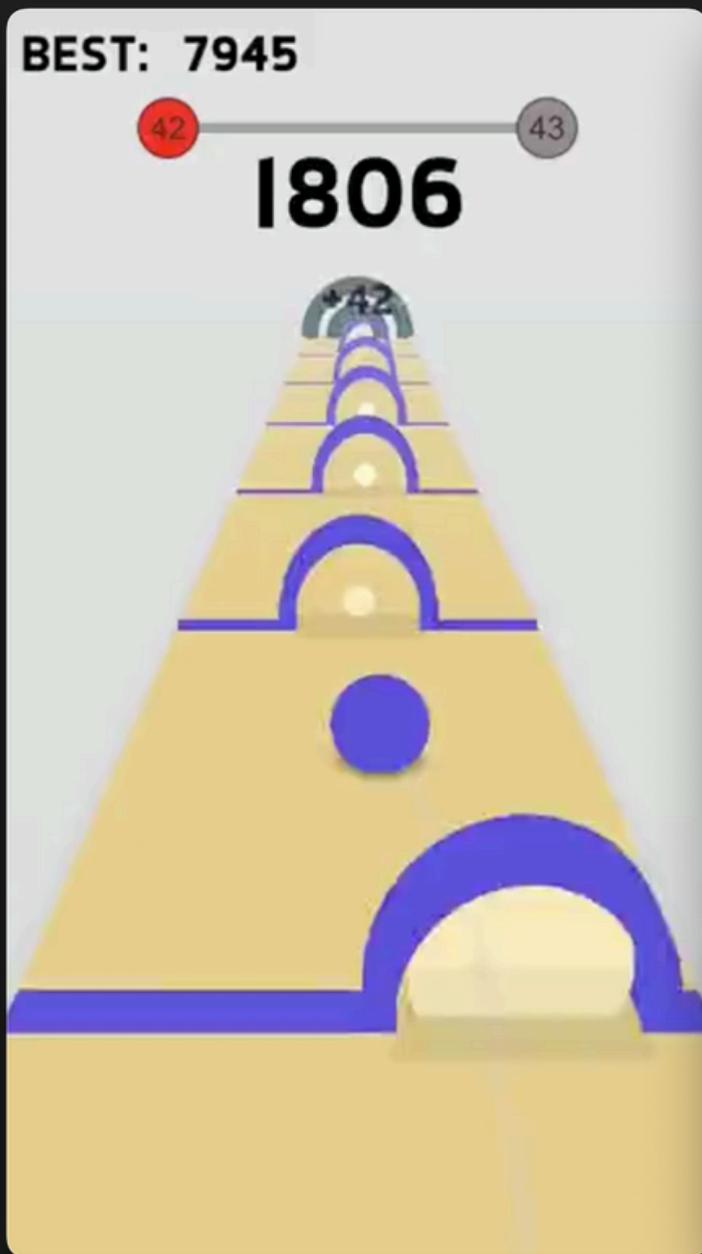
06 Hyper Casual Game 프로젝트



06 Hyper Casual Game 프로젝트



06 Hyper Casual Game 프로젝트



THANK YOU

더 많은 프로젝트를 보시려면 포트폴리오 사이트를 방문해주세요: whd793.github.io

포트폴리오 사이트: whd793.github.io

이력서 pdf: 디자인 동아리 회장 1년

깃허브: github.com/whd793

기술 블로그: whd793.blog

LinkedIn: [llinkedin.com/whd793](https://www.linkedin.com/in/whd793)

영문 이력서: [llinkedin.com/whd793](https://www.linkedin.com/in/whd793)

이메일: whd793@gmail.com

전화번호: 010-2551-5567