PORTFOLIO

Frontend Software Engineer **유 상호**

Phone

010-3346-7508

Email

qierapu1im@gmail.com

Resume

Go to link →

INTRODUCTION

안녕하세요! 유상호입니다.

React Native와 Expo를 중심으로 모바일 앱 개발에 몰두하고 있는 2년 차 개발자 유상호입니다.

저는 사용자 경험을 향상시키기 위해 끊임없이 고민하고 탐구하는 과정을 즐기며,
현재에 안주하지 않고 더 나은 방법을 찾기 위해 꾸준히 성장하기위해 노력하고 있습니다.

업무 외에도 개인적으로 관심 있는 아이디어를 직접 앱으로 구현하며 React Native의
애니메이션과 Skia를 활용한 인터랙티브 UI, 창의적인 그래픽 요소를 만드는 것을 좋아합니다.

단순히 코드만 작성하는 것보다 팀이 원하는 방향을 파악하고 함께 고민하여 최고의 결과물을 도출했을 때 느끼는 보람을 몹시 좋아합니다. 피드백은 빠르게 수용하고 적용하는 것을 중요하게 여기며, 독립적인 판단보다 팀원들과의 소통과 협력을 항상 최우선으로 생각합니다.

앞으로도 꾸준한 성장과 협력을 통해 팀의 소중한 일원으로서 함께 발전해 나가고 싶습니다.

WORK EXPERIENCE

경력 사항

(주) 시릴리

사원, 2023.04 – 2025.01

시리얼박스

React Native

MZ세대를 위한 무보정 일상 공유 SNS 앱 “시리얼박스”의 초기부터 합류해 개발과 유지보수를 전반적으로 담당하고,
핵심 기능 구현부터 UI/UX 개선, 성능 최적화, 코드 리팩토링 등 앱의 완성도와 유지 보수성을 높였습니다.

시리얼박스 관리자

NextJS

앱 분석과 콘텐츠를 효율적으로 관리하고, 데이터 시각화를 위한 관리자 페이지를 단독으로 개발했습니다.
다양한 데이터 시각화를 구현하고, 반응형 UI와 관리자 전용 기능으로 앱을 분석하고 관리자 편의성을 높였습니다.

Quality Assurance

서비스 품질 향상을 위해 QA 프로세스를 구축하고 팀 운영을 주도했습니다.
테스트 케이스를 체계화하고 업무를 분담해서 개발 효율성을 늘리고 앱 버그 발생률을 줄였습니다.

PERSONAL EXPERIENCE

개인 프로젝트

RIDERS CLUB

2025.05 – 2025.06

Expo

라이더를 위한 투어 코스 추천 및 스탬프 기능이 포함된 앱으로 기획부터 디자인까지 주도하며 프론트엔드 단독 개발 중입니다.
Expo 기반으로 개발 안정성을 확보하고, 위치 기반 추천, 지도 연동 등 주요 기능을 구현하고 있습니다.

NAMU

2025.04 – 2025.05

React Native

하루를 돌아보며 기록하는 다이어리 앱으로 프론트엔드를 단독으로 설계 및 개발했습니다.
React Native CLI 기반으로 Play Store 비공개 테스트 진행 중이며 소셜 로그인, 푸시 알림, CRUD 등
핵심 기능들을 직접 구현했습니다.

WORK EXPERIENCE PROJECTS

1. 시리얼박스

리얼한 일상을 담은 SNS 앱

1. 개요
2. 이미지 낙서 기능
3. Vision Camera 기반 리얼 카메라

WORK EXPERIENCE PROJECTS

1. 시리얼박스: 개요

SUMMARY

- 무보정 카메라로 리얼한 일상을 공유하는 SNS 앱
- React Native 기반, 약 45,000명의 사용자 확보
- 개발팀 3명 구성, Frontend 개발 및 유지보수 담당

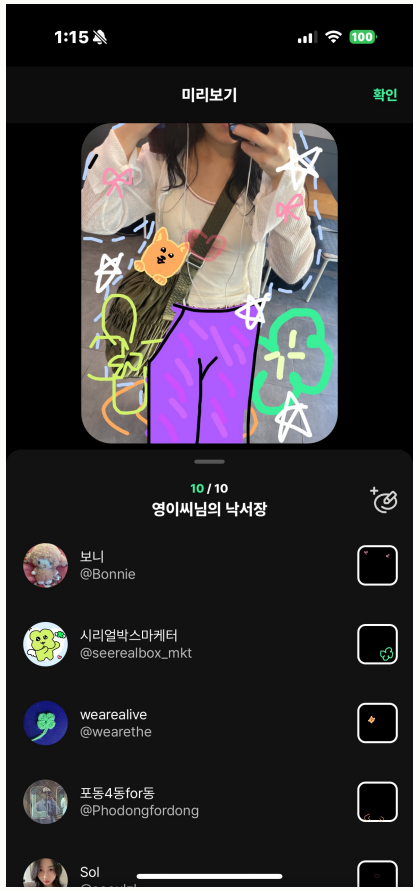
주요 기능

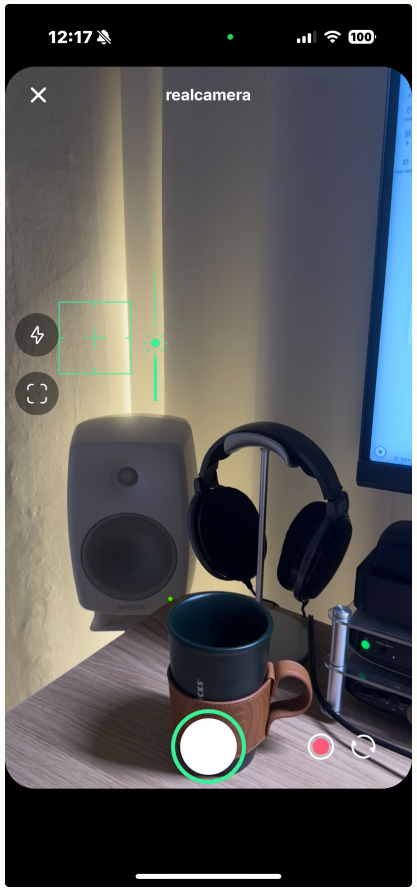
- 이미지 낙서 기능
- Vision Camera 기반 리얼 카메라
- Reanimated를 사용한 Splash, Sign In 화면
- 전역 상태관리 마이그레이션 및 TanStack Query 도입

SKILLS

- React Native
- React Native Skia
- React Native Gesture Handler
- React Native Vision Camera
- TanStack Query
- React Native Reanimated
- React Native View Shot
- React Native Compressor
- Zustand
- Etc

PREVIEW





WORK EXPERIENCE PROJECTS

1. 시리얼박스: 이미지 낙서 기능

어떻게 구현했나요?

1. 브러시 선택 (펜, 형광펜, 지우개)

- 사용자가 원하는 브러시를 선택합니다. 브러시는 각기 다른 효과가 있습니다.
  예를 들어 형광펜은 반투명한 색으로 처리되고, 지우개는 'clear' 모드로 지우개 역할을 하는 방식으로 구현했습니다.
- 다양한 색상을 선택할 수 있도록 컬러 팔레트를 제공합니다.
  브러시 굵기 조절은 iOS에서는 슬라이더, Android에서는 증감 버튼 방식으로 구현해 플랫폼별 UI 적합성도 고려했습니다.
  브러시 굵기는 실시간으로 캔버스 미리보기 영역에 반영됩니다.

2. 낙서 그리기 (사용자 입력 처리 + 실시간 렌더링)

- Gesture Handler의 Pan Gesture 기반으로 사용자의 터치 제스처를 실시간으로 추적했습니다.
  사용자가 화면을 터치하면 그 좌표를 시작점으로 새로운 선을 생성하고, Skia의 Canvas에 실시간으로 렌더링 됩니다.
  드로잉이 완료된 후에 각 path는 배열로 저장해 관리되고 Undo 기능을 통해 언제든지 되돌릴 수 있도록 구현했습니다.

3. 낙서 이미지 저장 및 업로드

- 그림 그리기가 완료되면, 그려진 낙서 Canvas를 이미지로 저장할 수 있도록 구현했습니다.
- View Shot을 사용해 그림 그린 Canvas 영역을 그대로 캡처하고,
  Image Resizer로 사진 품질을 압축(75%)해서 고화질을 유지한 상태로 업로드 속도 및 트래픽 최적화를 적용했습니다.
- 이미지 업로드가 완료되면 캡처한 Canvas 이미지를 낙관적 업데이트를 통해 해당 콘텐츠 화면에 즉시 반영되도록 처리했습니다.

WORK EXPERIENCE PROJECTS

1. 시리얼박스: 이미지 낙서 기능

문제를 어떻게 해결했나요?

1. 드로잉 중 초당 프레임(FPS)이 20~30대까지 떨어지는 현상 발생

문제 설명: 초기에는 드로잉 Canvas, 브러시 선택 UI, 컬러 팔레트, 버튼 등 모든 컴포넌트가 하나의 화면 트리에 포함되어 렌더링.
이러한 문제 때문에 사용자가 그림을 그릴 때마다 모든 UI 요소가 함께 리 렌더링 되는 구조였고,
드로잉이 빠르게 이어질 경우 프레임 드랍 및 터치 지연 현상 발생.

개선 방법: 1. UI 컴포넌트와 드로잉 Canvas 영역, 브러시 선택 UI, 컬러 팔레트를 별도 컨테이너에서 렌더링 되도록 분리해서
그림을 그릴 때는 오직 Canvas만 리 렌더링 되도록 리팩토링.

2. PanGesture 객체를 useMemo()를 사용해 의존성이 바뀌지 않으면 새로 생성되지 않게 최적화.

3. DrawContent 영역을 React.memo()로 감싸서 브러시 UI, 색상 변경 등의 state가 바뀔 때도 드로잉 Canvas 자체는 리 렌더링 되지 않도록 처리.
Gesture가 아닌 UI 상태 변경으로 인한 불필요한 렌더링을 근본적으로 차단.

개선 결과: 1. 드로잉 중에도 초당 프레임(FPS) 52~60 이상 유지.

2. 손가락을 빠르게 움직이며 그릴 때 렌더링이 딜레이되는 현상 없이 부드러운 드로잉 경험 제공.

2. 낙서를 그리는 동안 불필요한 UI 요소가 시야를 방해해서 UX 저하

개선 방법: 1. 제스처가 시작되면 불필요한 UI 요소를 자동으로 투명화해서 숨기는 방식으로 해결해 시야를 방해하지 않도록 설정했고,
제스처 종료 시 애니메이션을 통해 부드럽게 나타나도록 처리.

3. 업로드 후 낙서가 바로 업데이트되지 않고 서버에 요청해서 업데이트

개선 방법: 1. 낙서 업로드 성공 시, 해당 낙서 이미지를 기존 콘텐츠에 낙관적 업데이트를 적용해서 실시간으로 업데이트되게끔 변경.

WORK EXPERIENCE PROJECTS

1. 시리얼박스: Vision Camera 기반 리얼 카메라

어떻게 구현했나요?

1. 카메라 프리뷰 및 UI 최소화, 격자

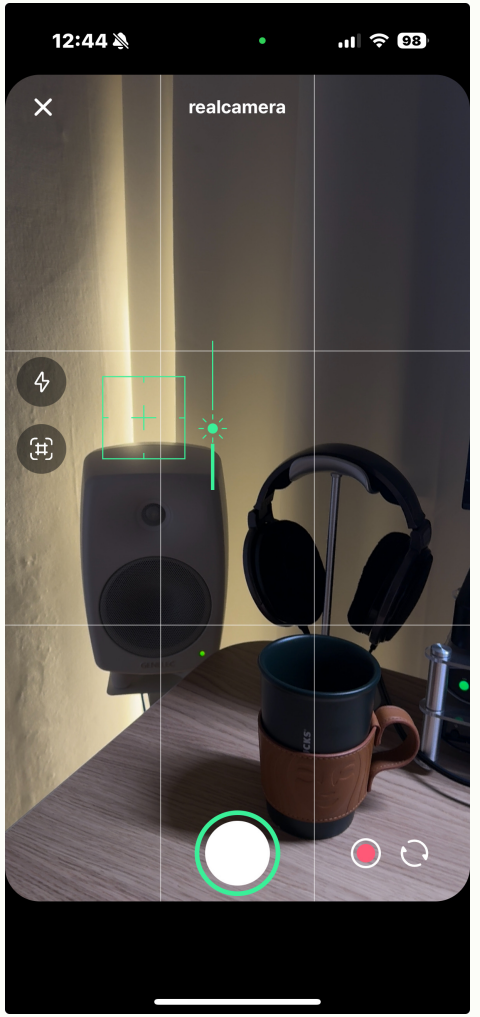
- Camera 컴포넌트를 사용해 16:9 화면으로 프리뷰를 구현하고, 어떤 기기에서든 동일한 비율로 화면이 나오게 구현.
- 스크린 길이를 계산해서 수평을 맞출 수 있는 격자 기능 제공.

2. 카메라 줌인/아웃, 포커스(Focus), 노출/밝기 조절

- Gesture Handler의 Pinch 제스처를 감지해서 사용자의 손가락 움직임에 따라 zoom.value를 계산해서 줌 인/아웃 구현.
- 화면 터치 시 Tap Gesture로 좌표를 감지하고, 해당 좌표를 Camera ref.focus(x, y)에 전달해 초점 이동.
- 노출 조절은 Pan Gesture로 progress bar의 상하 이동량을 감지해서 exposure.value를 실시간으로 반영해 노출값 조절 구현.

3. 사진 촬영 및 저장

- takePhoto()를 사용해 현재 프레임을 JPEG 이미지로 캡처.
- 캡처한 사진을 Blob 형태로 변경 후 서버 전송.



WORK EXPERIENCE PROJECTS

1. 시리얼박스: Vision Camera 기반 리얼 카메라

문제를 어떻게 해결했나요?

1. 기존 카메라 라이브러리의 유지보수 중단 및 최적화 문제

문제 설명: 앱 초기부터 React Native Camera를 사용하고 있었지만, 해당 라이브러리가 공식적으로 지원 중단되고,
일부 보급형~중급기 Android 기기에서 프리뷰 프레임 딜레이 발생.

개선 방법: 1. 지속적인 업데이트와 퍼포먼스를 고려해 React Native Vision Camera로 카메라 시스템 전체를 마이그레이션.

2. useCameraDevice, useCameraPermission 등 hook 기반으로 기기별 카메라 상태를 제어할 수 있도록 개선.

개선 결과: 1. 카메라 프리뷰 프레임 딜레이 개선.

2. 촬영 반응속도와 다양한 카메라 기능 도입으로 유저 불만 리뷰 감소.

2. 유저들의 다양한 카메라 기능 도입 요청

문제 설명: 초점 맞추기, 줌 인/아웃, 노출 조절 등 다양한 고급 카메라 기능이 필요하다는 지속적인 유저 요청이 들어왔음.

개선 방법: 1. Gesture Handler, Reanimated를 기반으로 커스텀 제스처를 구현.
Pinch, Pan, Tab 제스처로 카메라 초점, 줌 인/아웃, 노출 조절 기능을 구현.

개선 결과: 1. UX 상 일반적인 카메라 앱처럼 직관적인 조작이 가능하다는 긍정적인 피드백 확보.

2. 어두운 환경에서 초점 조절 및 노출 조절 기능을 통해 야간 촬영 품질 향상.

WORK EXPERIENCE PROJECTS

2. 시리얼박스 관리자

데이터 시각화와 콘텐츠 관리

1. 개요
2. Dashboard (데이터 시각화)
3. Admin (콘텐츠 관리)

WORK EXPERIENCE PROJECTS

2. 시리얼박스 관리자: 개요

SUMMARY

- 데이터 시각화와 콘텐츠 운영을 위한 통합 관리자 시스템
- 공식 계정 운영, 콘텐츠 큐레이션, 권한 제어까지 포함한 전체 운영관리
- 1인 개발 및 유지보수

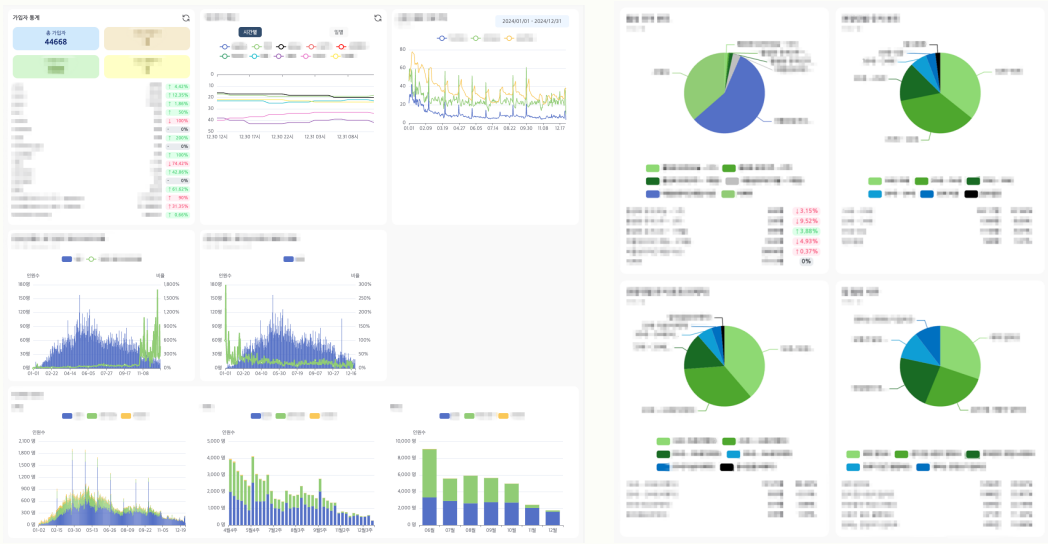
주요 기능

- 접속자 수/리텐션/트래픽 등 유저 지표 시각화 대시보드
- 콘텐츠 추천/비추천 및 태그 관리
- 공식계정 메시지 발송 및 앱 연동 메시징 기능
- 슈퍼관리자 권한 설정 및 관리자 접근 권한 관리
- FSD Architecture 도입
- 태블릿, 모바일 대응 반응형 웹 구현

SKILLS

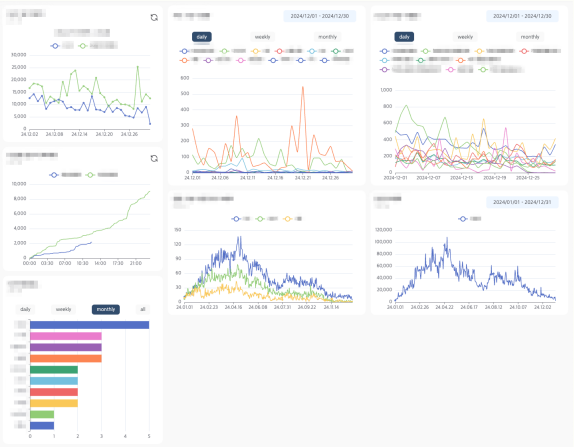
- NextJS app router
- Redux-Toolkit
- socket.io
- ECharts
- TanStack Query
- AWS EC2

PREVIEW



2. 시리얼박스 관리자: Dashboard (데이터 시각화)

어떻게 구현했나요?

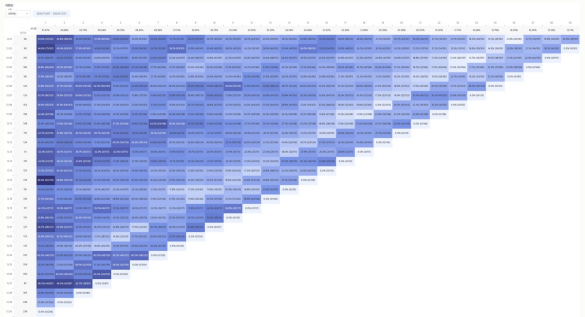


1. ECharts를 기반으로 다양한 통계 지표를 시각화

- 사용자 분포, 가입자/탈퇴자 변화, 기능 사용량, 리텐션, 트래픽, 유저 활동 추이 등 주요 운영 데이터를 Line, Bar, Hitmap, Nightingale 등으로 구성해 한눈에 파악할 수 있도록 시각화했습니다.

2. 차트 커스터마이징

- ECharts의 유연한 커스터마이징 기능을 활용해 색상, 라벨, 강조 효과, 터치 효과, 애니메이션 등을 직접 설정하고, 운영 목적에 맞는 레이아웃을 구현했습니다.
- 서버에서 받아오는 데이터 형식을 차트 구조에 맞게 변환하고, 조건에 따라 색상 테마를 다르게 적용할 수 있도록 적용.



3. 성능 최적화

- lazyUpdate를 사용해 데이터가 바뀔 때만 업데이트가 되게 설정해서 특히 데이터가 많거나 멀티 차트일 경우에 렌더링 비용을 효과적으로 줄였습니다.
- 사용자가 전체 데이터를 한 번에 보지 않고 스크롤(Zoom In, Out) 하며 구간별로 볼 수 있게 구성해서 렌더링 부하를 감소.
- 변화량이 많거나 수천 개 이상의 데이터가 존재할 경우 애니메이션을 false 설정해 시각적 딜레이 없이 즉시 반응하는 UI 구현.

2. 시리얼박스 관리자: Admin (콘텐츠 관리)

어떻게 구현했나요?

1. 콘텐츠 추천 관리, 업로드, 콘텐츠 상세 페이지

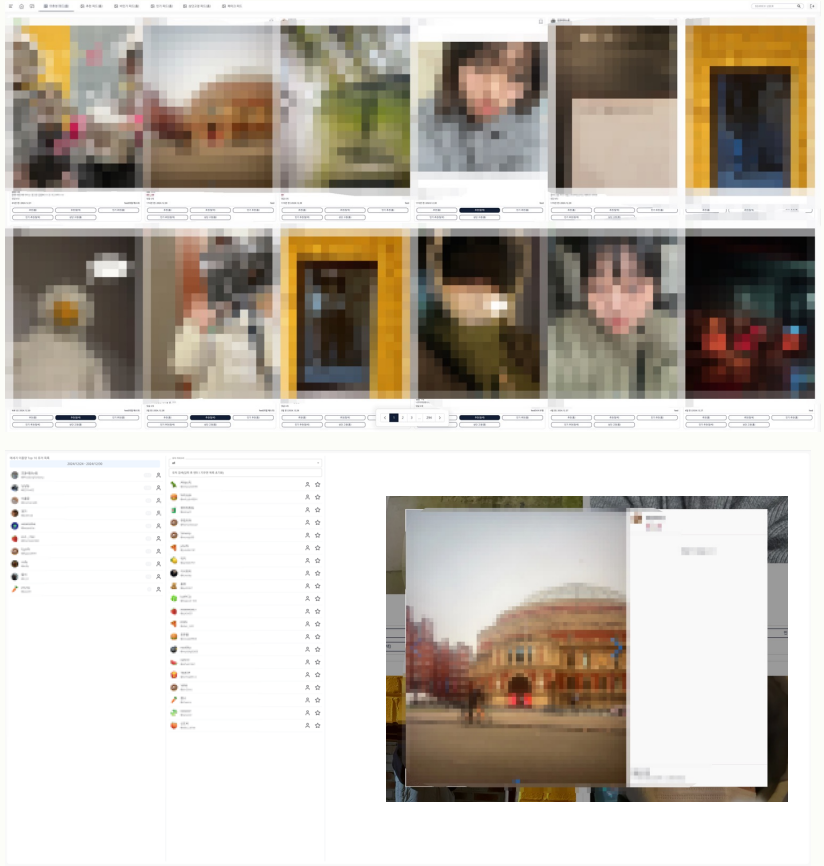
- 콘텐츠를 추천해서 피드 알고리즘과 별개로 상단에 추천하는 콘텐츠를 설정할 수 있게 구현했습니다.
- 공식 계정 및 봇 계정에 콘텐츠 업로드 기능 구현.
- NextJS Image를 사용해 성능 최적화(lazy loading)

2. 공식 계정 전체 메시지 및 1:1 대화 기능

- socket.io 기반 실시간 메시지 시스템을 구현해 전체 공지와 1:1 대화가 모두 가능하게 구현했습니다.

3. 성능 최적화

- 한번 받아오면 자주 변경되는 데이터가 아닌 부분들을 TanStack Query로 캐싱과 데이터 페칭 최적화.
- Observer, memo, lazy loading 등을 통해 렌더링 최소화 및 반응속도 개선



WORK EXPERIENCE PROJECTS

3. Quality Assurance

완벽한 앱이 되기 위한 필수 코스

WORK EXPERIENCE PROJECTS

3. Quality Assurance

SUMMARY

- 앱 테스트 및 품질 향상을 위한 프로세스 정립
- QA Lead, Notion 기반 협업 시스템 도입

주요 기능

- 기능 중심 테스트 전략 수립
- Notion 기반 에러 리포트
- QA-개발자 간 실시간 협업 구조 확립

SKILLS

- Microsoft Excel
- Notion
- Slack

QA 팀에서 어떤 일을 했나요?

- 시리얼박스 앱의 품질을 안정적으로 유지하고, 기능 배포 전후의 에러를 체계적으로 관리하기 위해 테스트 프로세스를 구축하고 팀을 운영했습니다.
- 팀 구성은 총 3명이었으며, 테스트 계획 수립, 테스트 케이스 작성, 에러 리포팅, 개발자 협업 등 QA 전 과정을 리딩했습니다.

3. Quality Assurance

QA 팀에서 문제를 어떻게 해결했나요?

1. QA 테스트 프로세스 개선

문제 설명: 초기 엑셀 기반 TC(테스트케이스)를 활용해 모든 기능을 문서로 정리하고 전수 테스트하는 탑다운 방식으로 QA를 진행.
하지만 시리얼박스 앱은 빠르게 기능이 추가되고 수정되는 구조였기 때문에 기능이 바뀔 때마다 TC를 새로 작성하거나 수정해야 했음.
테스트보다 문서 작업에 시간이 더 많이 소요되면서 테스트 본연의 목적이 약화함.
버그를 빠르게 발견하고 수정해야 하는 상황과 맞지 않게 비효율적인 구조였습니다.

개선 방법: 1. 10년 차 QA 전문가(중견 기업 QA 팀장)를 초청해 팀 내 실무 중심 교육을 주도.

2. 탑다운 전수 테스트를 핵심 기능 중심의 동적 QA 전략으로 전환.

3. TC는 최소한의 틀만 작성하고, 실제 테스트 중에 발생한 에러를 중심으로 에러 리포팅.

4. 엑셀에서 Notion 기반 에러 리포팅 시스템으로 전환하고 상세 분류 및 공통 템플릿을 만들어 문서 일관성을 확보.

개선 결과: 1. 200건 이상의 에러를 기록 및 분류.

2. 개발자는 QA 부담 없이 개발에만 집중할 수 있게 돼서 개발 속도 및 에러 처리 속도가 크게 향상됨.

3. QA-개발자 간 소통 비용이 90% 이상 감소.

PERSONAL EXPERIENCE PROJECTS

1. RIDERS CLUB

라이더를 위한 필수 투어 앱

PERSONAL EXPERIENCE PROJECTS

1. RIDERS CLUB

SUMMARY

- 오토바이 라이더를 위해 코스 추천 및 투어 스탬프 기능 앱
- 검증된 SDK 사용으로 안정성을 높이고 새로운 기술 습득을 위해 Expo 도입
- 기획부터 디자인, 개발까지 전 과정을 직접 리드하며 MVP 개발 중
- 프론트엔드 1명, 백엔드 1명, 디자이너 1명으로 구성

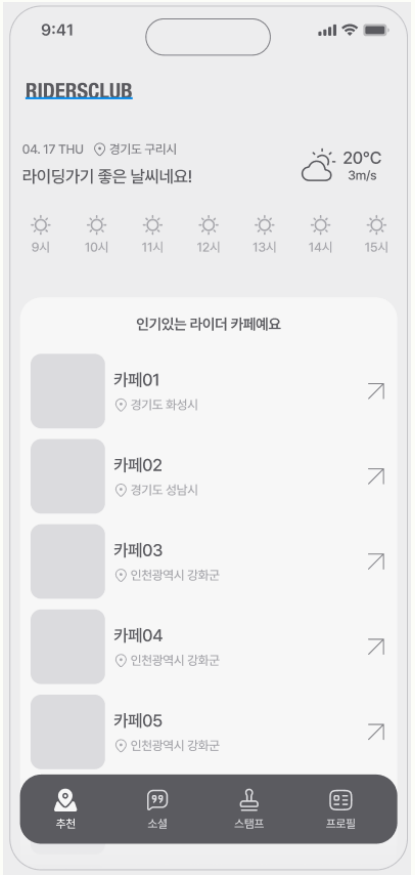
주요 기능

- 현재 위치 기반 날씨, 투어 코스 및 장소 추천
- 네이버 지도 연동으로 코스 위치 확인
- 선택한 코스를 네이버 지도 및 티맵으로 딥링크 적용
- Splash, Sign In 화면 등 Reanimated를 사용한 UI 구성
- Firebase Analytics, Message 도입

SKILLS

- Expo
- React Native
- React Native Skia
- React Native Reanimated
- React Native Gesture Handler
- Naver Map, T MAP
- TanStack Query
- Zustand
- Etc

PREVIEW



PERSONAL EXPERIENCE PROJECTS

2. NAMU

매일의 일상 기록을 담는 앱

PERSONAL EXPERIENCE PROJECTS

2. NAMU

SUMMARY

- 매일 일상을 기록하며 하루를 되돌아보고 한 달 기록을 남기는 다이어리 앱
- 5월 중순 Google Play Store 출시 예정
- 프론트엔드 1명, 백엔드 1명, 디자이너 1명, 기획자 1명으로 구성

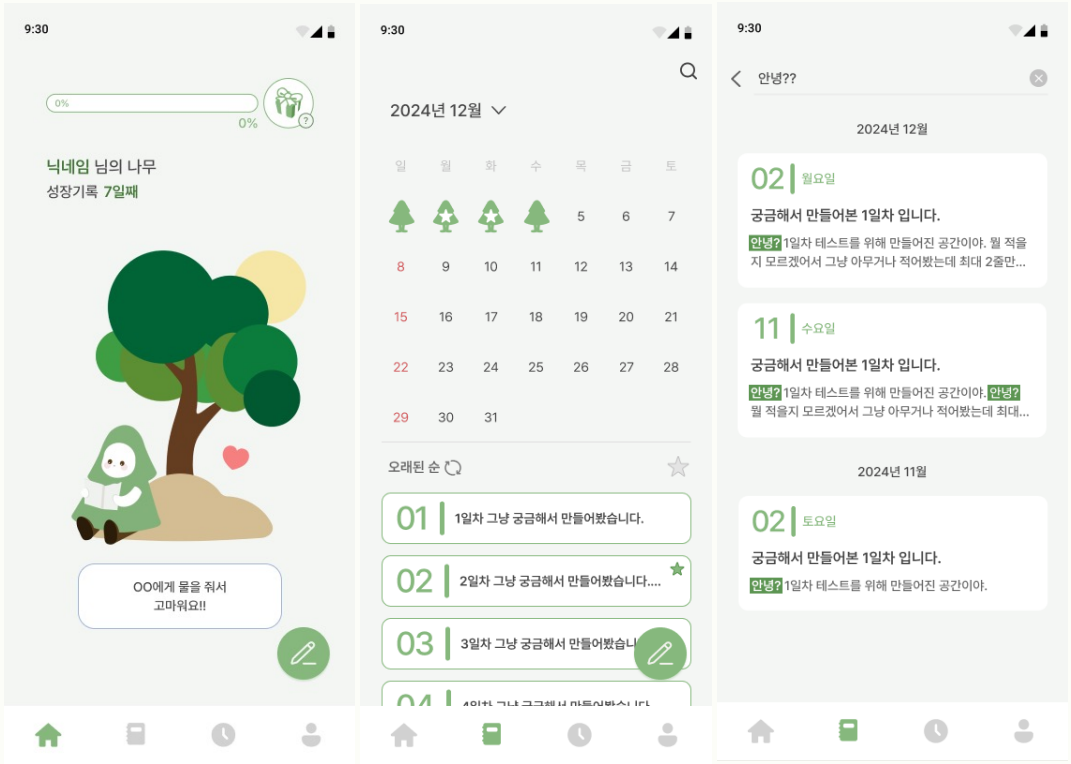
주요 기능

- 하루 기록, 한 달 기록 남기기
- 작성한 일기 검색
- 캘린더 커스터마이징
- Android, iOS에서 동일한 UI의 커스텀 Date Picker
- FCM, Google Login

SKILLS

- React Native
- React Native Reanimated
- React Native Gesture Handler
- React Native Calendars
- Firebase
- Etc

PREVIEW



유 상호

Phone

010-3346-7508

Email

qierapu1im@gmail.com

Resume

Go to link →