# 김대희

## 사용자가 느끼지 못하는 불편까지 고민하는 iOS 엔지니어입니다.

스티브 잡스는 **「디자인은 단지 보이는 것만이 아니라 어떻게 작동하느냐이다」**라고 말했습니다. 저는 이 말에 깊이 공감하며, 보이지 않는 디테일까지 완벽하게 작동하는 앱을 만드는 데 집중합니다. UI의 미세한 오차, 애니메이션의 작은 끊김조차 용납하지 않으며, 사용자가 미처 인지하지 못한 불편까지도 제가 먼저 발견하고 해결하려고 노력합니다.

또한 코드의 명확성과 가독성을 최우선으로 생각하며, 협업하는 동료와 미래의 저를 위해 언제나 이해하기 쉽고 효율적인 코드를 작성합니다. 앞으로도 사용자 중심의 가치를 철저히 고민하며 오랫동안 사랑받는 서비스를 만들겠습니다.

### Work Experience

**넛지헬스케어 (캐시워크)** / iOS Developer
2023년 10월 23일 ~ 재직 중

**패스트포워드 (도미노)** / 계약직 (겸직), iOS Developer
넛지헬스케어 자회사 | 2025년 4월 1일 ~ 재직 중

**더줌코리아** / 산학실습, iOS Developer
넛지헬스케어 자회사 | 2023년 07월 25일 ~ 2024년 1월 31일

### Awards

**UNITHONE 9TH** 대상
2022년 09월 04일

제 7회 **전국 고등학교 동아리 소프트웨어 경진대회** 장려상
2022년 09월 04일

### Activity

**교내 iOS 스터디 운영**
2022년 7월 21일 ~ 2023년 1월 1일

제 8회 **대한민국 SW융합 해커톤** 본선 진출
2021년 10월 29일 ~ 2021년 10월 31일

**HIGHTHON 8TH** 해커톤 참여
2023년 1월 14일 ~ 2023년 1월 155일

### Certification

**정보처리기능사** / 한국산업인력공단
2023년 4월 26일

### Contact Information

career@daehee.me

kimdaehee0824

### Skills

Swift · SwiftUI · UIKit · RxSwift · Combine · Moya · SnapKit · MVVM · TCA · ReactorKit · Tuist · TMA · Needle · Swift Testing · fastlane · Xcode Cloud · Github · Figma

### Side Projects

**Ballhub** / 아트엔피지컬
2024년 5월 ~ 2025년 5월

**WAKTAVERSE MUSIC** / PARABLE Ent.
2022년 10월 ~ 2024년 3월

**기숙사 관리 시스템 DMS** / 교내 전공 동아리 DMS
2021년 5월 ~ 2024년 1월

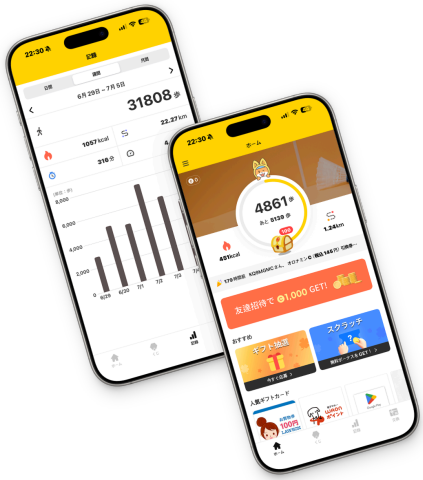
### Educations

**대덕소프트웨어마이스터고등학교**
2021년 03월 입학, 2024년 01월 졸업

# 일본 캐시워크

**걷기만 해도 포인트를 적립하는 건강 리워드 앱**

App Store | Website



## Project info

일본 전용으로 출시된 **캐시워크의 현지화 앱**으로, LINE·Google 계정 연동과 캐릭터 기반 UI 등 일본 사용자에 맞춘 전략을 반영했습니다. 100걸음마다 생성되는 박스를 터치해 포인트를 적립할 수 있으며, 적립한 포인트는 아마존, 스타벅스, 편의점 기프트카드 등으로 교환할 수 있습니다. 또한, 운동 기록을 이미지로 저장·공유할 수 있는 기능도 제공합니다.

## Key Contributions

### TMA 기반 아키텍처 설계 및 모듈 구조 구축

Swift 6.0 및 최소버전 iOS 17 환경을 고려하여, TMA(The Modular Architecture) 패턴 기반의 아키텍처를 설계하고 피처 단위로 독립적인 개발·검증이 가능한 구조를 구축했습니다. 각 피처는 Interface, Sources, Testing, Example, Tests 타겟으로 구성되어 책임이 분리되며, 공통 기능은 별도 모듈로 분리해 유지보수성과 재사용성을 극대화했습니다.

- **Repository 패턴 기반**으로 Interface에 프로토콜, Sources에 구현체를 정의하고, 비즈니스 로직과 데이터 접근 간 결합도를 낮췄습니다.
- Mock 구현체는 Testing 타겟에 포함하고, Example 타겟에서 주입해 서버 없이도 각 **피처의 동작을 독립적으로 실행·검증할 수 있도록 구성**했습니다.
- 네트워크, UI, Firebase, WebKit 등 공통 기능을 **목적별로 모듈화**하여 중복 구현을 줄이고 책임을 분리했습니다
- Swift 6.0에 맞춰 actor, @Sendable, async/await을 적용하고, Repository 전반을 actor 기반으로 구성해 **데이터 레이스를 방지**했습니다.

### AI를 이용한 API 자동화 및 문서화

AI 기반 생산성 향상을 위해 Claude Code를 프로젝트에 도입하고, 개발 전후 다양한 영역에 에이전틱 프로그래밍을 적용했습니다. 수동 반복 작업을 최소화하면서, 안정성과 확장성을 동시에 확보할 수 있는 개발 환경을 구축했습니다.

- 뷰모델 테스트 커버리지 100% 달성을 목표로 **체계적인 테스트 환경을 구성**하고, 테스트 코드는 사람이 직접 작성하며 뷰모델 로직은 Claude를 통해 자동 생성하는 TDD 기반 워크플로우를 설계했습니다.
- 기능 단위 API 개발 시 OpenAPI JSON을 기반으로 설계부터 구현, 테스트까지의 전 과정을 **문서화된 규칙에 따라 자동화**하였고, 네이밍 및 모델 구조 컨벤션을 통일하여 일관된 품질을 유지했습니다.
- 기능 구현이 완료된 후 Claude를 활용해 폴더 구조를 분석하고, 각 피처 단위의 문서를 자동 생성 및 업데이트하여 **최신 상태의 내부 문서를 유지**했습니다.
- 모든 문서는 Notion 등 외부 도구가 아닌 레포지토리 내부에 포함시켜, 사람과 AI 모두가 동일한 소스 경로에서 문서를 활용할 수 있도록 구성했습니다. 문서화 수준은 **인수인계 없이도 바로 투입 가능한 수준을 기준으로 작성**했습니다.

### 디버깅 환경 고도화 및 ViewModifier 기반 진단 도구 구축

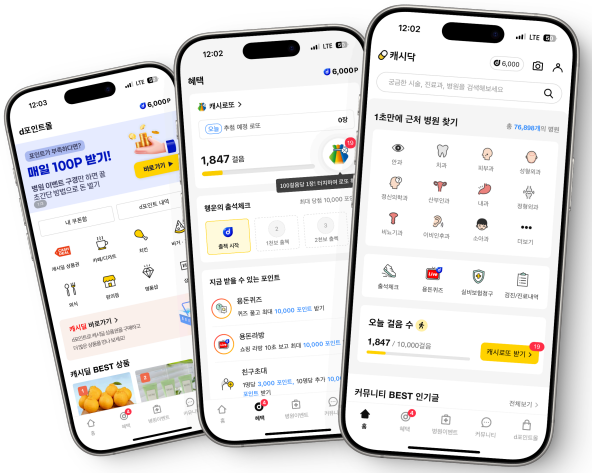
개발 중 디버깅 생산성을 극대화하기 위해 디버깅 기능을 공통화하고, 모든 진단 도구를 ViewModifier 형태로 구성해 Example 앱에서도 코드 단 한 줄로 적용 가능하도록 설계했습니다. 시뮬레이터, 실제 기기 등의 환경 조건을 고려하여 다양한 개발 시나리오에서 안정적인 디버깅을 지원합니다.

- iOS 18 이상에서 텍스트를 길게 눌러 일본어를 한국어로 번역하는 기능을 도입해, 현지 UI 디버깅과 QA 효율을 개선했습니다.
- CADisplayLink 기반 프레임레이트 모니터를 구현해 FPS와 프레임 드롭을 실시간 시각화하고, 렌더링 병목 원인을 식별할 수 있도록 지원했습니다.
- 기기 흔들기 제스처를 이용해 Pulse 기반 디버깅 콘솔을 띄울 수 있도록 구현하였으며, 네트워크 로그, Firebase 로그, 광고 타입 등 다양한 실시간 정보를 직접 확인할 수 있는 환경을 구성했습니다.
- 모든 기능을 ViewModifier로 분리해, 실제 앱과 ExampleApp에서 한 줄로 쉽게 적용할 수 있도록 구성했습니다.

# 캐시닥

**병원 정보와 포인트를 동시에 버는 병원찾기 앱**

App Store | Website



## Project info

**매월 200만 명 이상이 이용**하는 의료 리워드 앱입니다. 병원 접수·예약, 진료 조회, 실비 보험 청구 등 **의료 편의 서비스를 제공**하며, 성형·미용 이벤트도 확인할 수 있습니다. 걸음 수를 기반으로 버튼을 눌러 광고를 시청하거나 퀴즈를 풀어 리워드를 적립하고, 자사 몰에서 물건을 구매하거나, 쿠폰으로 교환할 수 있습니다.

## Key Contributions

### xcodeproj 에서 Tuist로 마이그레이션 & Tuist 4.x 대응

기존 Xcode 프로젝트 기반의 캐시닥 프로젝트를 **Tuist 기반으로 전환**하여 프로젝트 구조를 개선하고, 빌드 성능을 최적화했습니다. 마이그레이션을 통해 모듈화와 외부 라이브러리 관리 방식을 재정비해 빌드 속도를 크게 향상시켰습니다. 이를 통해 유지보수성을 높이고, 빌드 및 프로젝트 생성 **시간을 단축**하여 개발 생산성을 높이는 데 집중했습니다.

- 기존 단일 프로젝트 타겟에서 모듈 분리를 고려하여 **Shared 모듈을 신설**하고, 공통 함수 및 클래스를 분리하였습니다
- 외부 라이브러리 관리 방식도 최적화했습니다. CocoaPods을 사용하던 라이브러리 중 광고 SDK(AppLovin, AdFit, AdMob)를 제외한 95%를 Tuist External Dependencies로 전환하여 기존 **빌드 시간을 50~70% 단축**했습니다. Then, RxKeyboard, RxGesture, RxViewController와 같이 파일 수가 적은 SDK는 프로젝트 내부에 추가하여 **자체적으로 관리**함으로써, Tuist generate 시간을 더욱 단축했습니다.
- 이전에는 빈번하게 발생하던 프로젝트 파일의 **Code Conflict가 단 한 건도 발생하지 않았습니다.**

### 2.0.0 app 개편

2020년부터 유지된 홈 화면을 **전면 개편**하고, 관련 코드를 모두 ReactorKit 기반으로 변경해 **유지보수성을 개선**했습니다.

- 탭바 컨트롤러는 Custom View에서 TabbarController 상속 클래스로 변경했습니다. 이 과정에서 View life cycle이 기대하는 시점에 작동하지 않아, 방문하지 않은 화면의 광고와 웹뷰가 로드되는 문제를 해결했습니다.
- 혜택 탭을 신설하면서, 기존 홈 화면에 있던 중요한 광고 로직을 유지하면서도 ReactorKit의 라이프사이클을 따르는 구조로 코드를 작성했습니다.
- 홈 화면은 기존 네이티브 UI에서 웹뷰 기반으로 변경되었으나, 평균 5초의 긴 로딩 문제가 있었습니다. 이를 개선하기 위해 앱 시작 시 WKWebView를 생성·선로드하고, HomePreLoader에 저장한 후 홈 화면 로드 시 해당 웹뷰를 즉시 가져오도록 수정해 로딩 시간을 80% 단축했습니다. 이러한 개선을 통해 사용성이 크게 향상되었습니다.

### UI & UX 개선

프로젝트의 신규 개발과 기능 개선을 진행하면서. 기존부터 존재하던 **UI,UX 관련 문제들을 수정**했습니다.

- 기존 앱은 탭바나 상단 터치 시 스크롤이 최상단으로 이동하지 않아 사용성이 떨어지는 문제가 있었습니다. 이것을 해결하기 위해 탭바의 shouldSelect 함수에서 contentOffset을 직접 조정해 빠르게 상단으로 이동할 수 있도록 개선했습니다.
- 2.0 이전 버전에서 노란색 테마의 네비게이션 바에 애니메이션을 적용할 때, setNavigationBarHidden(_:animated:) 호출로 인해 잔상이 남는 문제가 발생했습니다. 또한 웹뷰의 hidden 상태와 Status Bar 색상을 함께 제어해야 하다 보니 코드가 복잡해졌습니다. 이를 해결하기 위해 통합 웹뷰 클래스의 라이프 사이클에서 Appearance 속성을 변경하도록 수정하고, 동시에 애플 네비게이션 바와 동일한 디자인과 기능을 갖춘 커스텀 뷰를 제작하여 애니메이션 잔상을 완전히 제거했습니다.
- 기존에는 커스텀 알럿을 띄우거나 전체 화면을 덮는 로딩 뷰를 표시할 때, 애니메이션 없이 뷰가 갑자기 나타났다 사라지는 문제가 있었습니다. 이를 개선하기 위해, 페이드 애니메이션을 추가하여 보다 부드러운 화면 전환이 이루어지도록 수정했습니다.

## DMS 교내 전공 동아리 DMS

Github App Store

DMS는 모교 기숙사의 원활한 운영을 돕기 위해 개발된 관리 시스템으로, 학생과 교직원 모두의 편의성을 높이는 서비스입니다. 학생들은 급식 확인, 외출 및 자습실 신청, 공지사항 확인 등의 기능을 통해 기숙사 생활에 필요한 정보를 빠르게 확인할 수 있습니다.

iOS 팀의 일원으로 합류하여 전국의 학교가 사용할 수 있는 DMS를 새로 만들고, 기존 DMS를 유지보수했습니다. 기존의 복잡한 UI/UX 문제를 개선하고 사용성을 높이기 위해 리빌딩과 새로운 기능 추가, 기능 디자인을 병행했습니다

- 기존 DMS 프로젝트는 Storyboard 기반으로 화면과 구조가 복잡해 유지보수가 어려웠습니다. 이를 해결하기 위해 SwiftUI와 Combine을 도입해 직관적이고 명확한 구조로 재구성했고, Needle로 의존성 주입, Tuist로 모듈화를 진행해 코드 안정성과 개발 생산성을 높였습니다.
- 초기 기획 단계에서 기숙사 선생님들과 직접 인터뷰를 진행하며 요구사항을 명확히 분석했습니다. 이 내용을 시스템 구성도, 플로우 차트, 화면 설계서로 구체화하여 프로젝트의 방향성을 명확히 했습니다.
- 특히 자습실 신청 기능은 사용자가 직관적이고 편리하게 사용할 수 있도록 직접 디자인을 기획하고 개발했습니다. 작은 화면 전환이나 버튼 애니메이션까지 세심히 고민하며, 사용자가 느끼지 못하는 불편까지도 놓치지 않고 개선했습니다.
- Xcode Cloud CI/CD 환경 구축 시, Tuist 설치 누락으로 빌드가 실패하는 문제가 있었습니다. 공식 문서로 해결 방법을 찾기 어려워 WWDC 세션을 직접 참고해 ci_post_clone.sh를 활용한 자동화 스크립트를 작성해 문제를 해결했습니다. 이를 통해 배포 환경의 안정성과 효율을 높였습니다.

## Ballhub 주) 아트엔피지컬

App Store

일일 방문자 6,000명 이상이 이용하는 해외 축구 커뮤니티 서비스입니다. 사용자는 응원하는 팀을 선택해 맞춤형 게시판을 이용할 수 있으며, 경기 일정 및 상세 정보를 한눈에 확인할 수 있습니다. 자유 게시판과 뉴스 게시판을 통해 해외 축구 팬들과 활발하게 소통할 수 있으며, 커뮤니티 중심의 참여형 서비스로 지속적으로 기능을 확장해 나가고 있습니다.

프로젝트 초기부터 앱 출시 이후까지 iOS 팀 리더로서 서비스의 성장과 운영을 주도했습니다. 신기술 도입, 아키텍처 설계, 개발자 리소스 관리, 코드 품질 및 PR 리뷰를 담당하며 안정적인 서비스 운영과 지속적인 기능 확장을 이끌었습니다.

- SwiftUI 및 TCA를 도입하고 MFA 아키텍처 기반 초기 환경을 구성하여 안정적인 프로젝트 구조를 정립했습니다. 기존 MFA와 Needle을 함께 사용했을 때 발생한 무한 초기화 문제를 TCA Scope를 활용해 해결하며 SwiftUI 환경에서 최적의 UI 전환 성능을 확보했습니다.
- Pulse SDK를 커스텀하여 로깅 시스템과 네트워크 디버깅 도구를 개발해 iOS팀의 디버깅 효율을 크게 높였습니다.
- 팀원들과 긴밀한 소통을 통해 업무 우선순위를 명확히 설정하고 PR 리뷰 프로세스를 개선했습니다. 이로 인해 코드 리뷰 시간이 단축되고 개발 리소스를 효율적으로 배분하여 프로젝트 일정을 안정적으로 관리했습니다.

## WAKTAVERSE MUSIC PARABLE Ent.

Github App Store

**하루 약 2,000**명 이상이 사용하는 음악 서비스입니다.  크리에이터 (우왁굳, 이세계 아이돌, 왁타버스)가 업로드한 음악의 시간대별 순위를 확인하거나 검색할 수 있으며, 사용자가 직접 플레이리스트를 제작할 수 있는 기능도 제공합니다

iOS 팀의 일원으로 합류하여 기존 1.0 서비스를 유지보수 하고. UIKit 기반의 2.0 개발을 진행했습니다.

- 기존 1.0 버전은 아키텍처 설계가 미흡하고 코드의 가독성이 떨어져 유지보수에 어려움이 있었습니다. 이에 SwiftUI 기반 코드를 과감히 제거하고, MVVM과 Clean Architecture를 기반으로 프로젝트를 재설계했습니다.
- Tuist를 활용한 Modular Architecture 설계와 Network 모듈 구축을 통해 프로젝트 구조를 정립하고, 유지보수와 확장성을 높였습니다
- Google Sign-In을 라이브러리 없이 직접 구현하여 의존성을 줄이고 인증 프로세스를 간소화했습니다.