

## Education

### 삼성 청년 SW·AI 아카데미 13기 (SSAFY)

- 2025.01 - 2025.12
- 1학기 성적우수상 수상
- 1학기 프로젝트 최우수상
- 2학기 프로젝트 우수상 3회

### 전남대학교

- 2016.03 - 2024.02
- 전기공학과 학사

## Awards

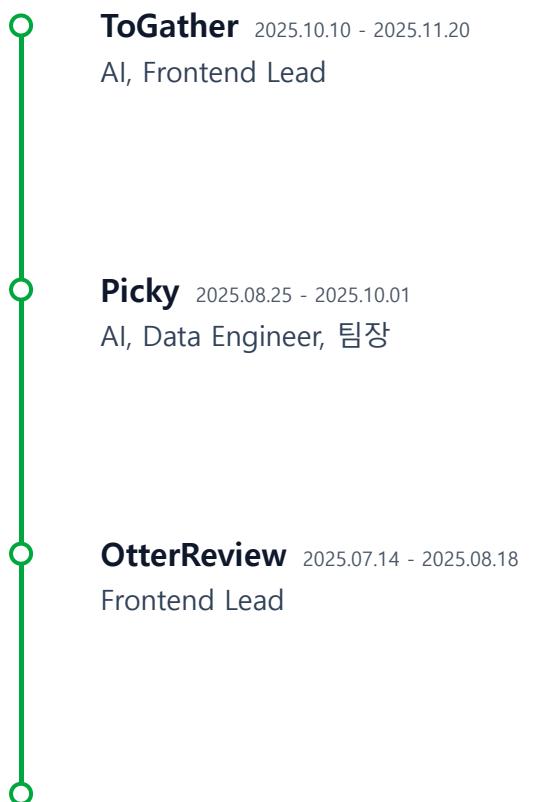
- 2025.06 삼성 청년 SW·AI 아카데미 성적우수상(3등/23명)  
2025.06 삼성 청년 SW·AI 아카데미 프로젝트 최우수상(1등/11팀)  
2025.08 삼성 청년 SW·AI 아카데미 프로젝트 우수상(2등/9팀)  
2025.10 삼성 청년 SW·AI 아카데미 프로젝트 우수상(2등/8팀)  
2025.11 삼성 청년 SW·AI 아카데미 프로젝트 우수상(2등/6팀)

## Certificates

## Experience

2021.09 - 2023.09 카페 바리스타 (Part-time)

## Project Summary



### ToGather 2025.10.10 - 2025.11.20

AI, Frontend Lead

### 커플이 일상을 기록하고 공유하는 모바일 라이프로그 서비스

- 개인화된 AI 챗봇
- 자연어 날짜 정규화 기반 Advanced RAG(Time-aware)
- Qdrant 단일 컬렉션 + 메타데이터 필터 구조
- 온보딩 상태 기반 전역 상태 설계 및 앱 재진입 시 단계 복원 로직 구현

### Picky 2025.08.25 - 2025.10.01

AI, Data Engineer, 팀장

### 브라우징 로그 기반 개인화 컨텐츠 추천 마이크로 러닝 서비스

- 크롬 확장프로그램 기반 사용자 로그 수집
- O(1) 벡터 업데이트 기반 개인화 프로필 관리
- 크롤링-KoBART 뉴스 요약 파이프라인
- MongoDB 색인 기반 대용량 데이터 설계

### OtterReview 2025.07.14 - 2025.08.18

Frontend Lead

### 주니어 개발자의 PR·리뷰 진입 장벽을 낮춘 실시간 협업 서비스

- GitHub 기반 PR 생성·리뷰 플로우 UI 설계
- Zustand 기반 복합 리뷰 상태 관리
- 코드 Diff 멀티라인 선택·스레드형 댓글 구조
- SSE 기반 push 이벤트 알림 처리

커플이 일상을 기록하고 공유하며, 두 사람의 기록을 기반으로 개인화된 경험을 제공하는 모바일 라이프로그 서비스

## 프로젝트 목표

커플의 기록을 단순 저장이 아니라 다시 꺼내볼 수 있는 맞춤형 추억 회상 경험으로 만드는 것을 목표로 했습니다.

## 핵심 기여

- 커플 기록 기반 개인화 AI 챗봇:  
자연어 날짜 파싱 → 메타데이터 필터 우선 적용 RAG 파이프라인 설계·구현으로 날짜 쿼리 정확도 개선
- Qdrant 단일 컬렉션 + 메타데이터 필터 구조로 couple\_id별 벡터 격리 및 type(profile/diary/plan) 분리 저장 → 검색 성능 최적화
- 앱 재진입 시 상태 기반 분기 로직을 설계하여 커플 상태·프로필·초대 코드에 따라 온보딩을 복원
- SSE 실시간 커플 연동: 피초대자 코드 입력 시 초대자 자동 화면 이동, 실시간 동기화로 초기 이탈률 감소

## 사용 기술

FastAPI

Qdrant

Text Embedding

RAG

React Native (Expo)

TypeScript

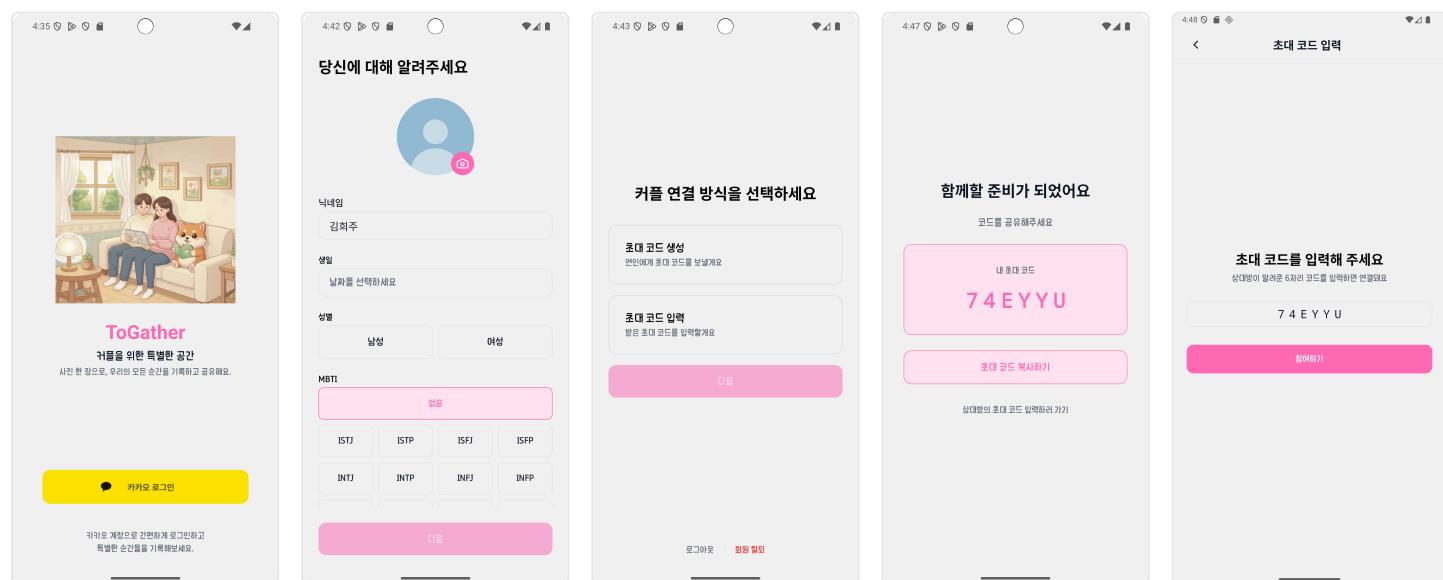
Zustand

TanStack Query

## 성과

- 날짜 쿼리 정확도 대폭 향상: 자연어 날짜 파싱 + 메타데이터 필터 우선 적용으로 날짜 관련 질문 정확도 99% 달성
- 검색 성능 최적화: 단일 컬렉션 구조와 Range 필터 적용으로 Qdrant 검색 성능 최적화
- 온보딩 이탈률 감소: SSE 동기화로 커플 연동 과정 실시간 피드백 제공

## 온보딩 플로우

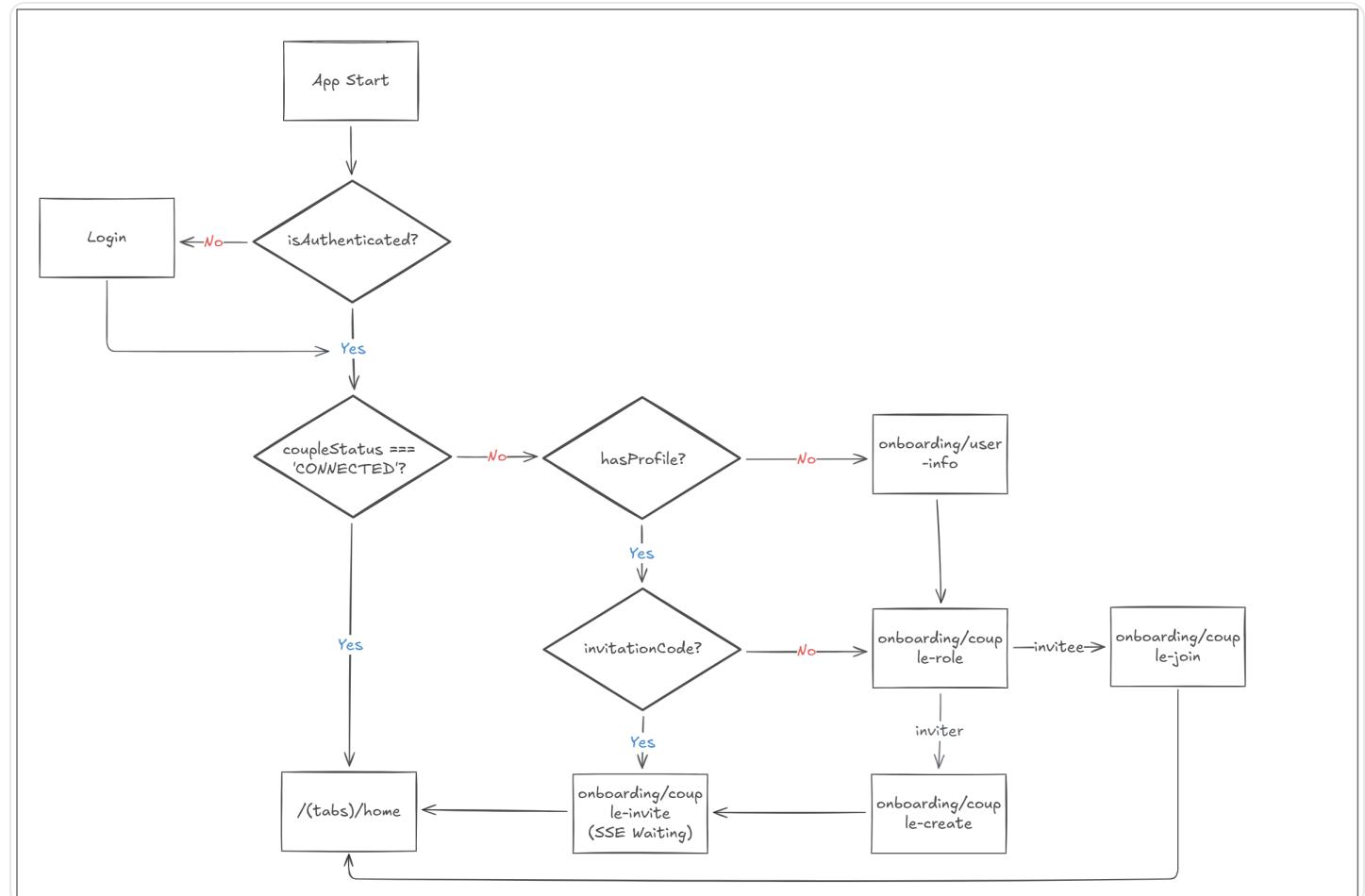


# 온보딩 상태 복원 로직

## 도입 배경

커플 상태, 프로필 정보 입력 여부, 초대 코드 존재 여부에 따라

사용자가 앱을 재진입했을 때 올바른 단계로 복원해주는 온보딩 분기 설계가 필요했습니다.



## 주요 구현

- 온보딩 Context 기반 전역 상태 관리 및 앱 재진입 시 올바른 단계 복원 로직 구현
- SSE 기반 커플 연동 실시간 동기화: 피초대자 코드 입력 → 초대자 자동 홈 이동

## 기술적 포인트

AsyncStorage 기반 상태 복원 및 SSE 실시간 동기화

## 상태 기반 분기

- coupleStatus === 'CONNECTED' → 홈 화면
- hasProfile (nickname, birthday, mbti) → 프로필 완성 여부
- invitationCode 존재 → 초대 대기 화면

# 개인화 AI 챗봇 RAG 파이프라인(Advanced RAG)

## 도입 배경

챗봇 초기에는 날짜 중심 질문에서 정확도가 낮았습니다.

텍스트 기반 질문은 잘 매칭되었지만, 날짜 기반 질문에서는 날짜를 의미로 인식하지 못해

단순 문자열 유사도로 다른 날짜의 기록을 반환하거나, '저번 주' 같은 상대 날짜 표현을 처리하지 못했습니다.

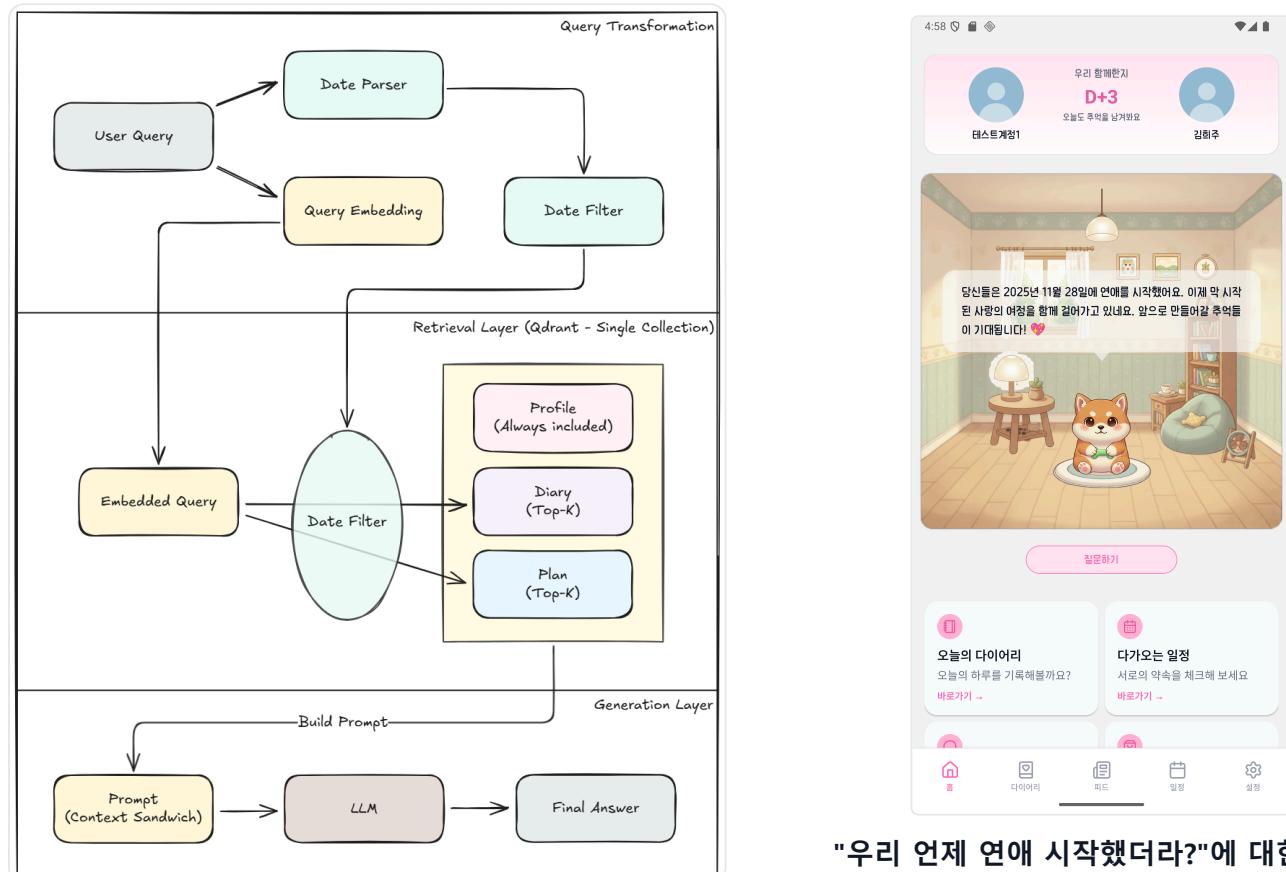
해당 문제의 원인은 벡터 검색이 날짜 개념을 이해하지 못하고,

상대 날짜 표현은 정규 날짜로 변환되지 않아 검색 자체가 불가능하기 때문입니다.

이를 해결하기 위해 자연어 날짜를 정규화하는 Query Transformation(Date Parser)을 도입해

모든 날짜 표현을 Unix timestamp 범위로 변환하고 메타데이터 필터를 우선 적용하는 구조로 개선했습니다.

아래 RAG 파이프라인은 이러한 문제를 해결하기 위해 설계한 구조입니다.



"우리 언제 연애 시작했더라?"에 대한 답변

## 주요 구현

- Qdrant 단일 컬렉션 + type(profile/diary/plan) 분리 저장: 프로필 검색 없이 직접 조회, diary/plan 각 5개씩 벡터 검색
- 날짜 인식 문제 해결: 자연어 날짜 파싱 → Query Transformation → 메타데이터 필터 우선 적용 RAG 파이프라인 구축

## 기술적 포인트

- 3단계 분리 검색 파이프라인:
  - profile(couple\_id+type 필터 직접 조회) → ② diary 벡터 검색(limit=5) → ③ plan 벡터 검색(limit=5)
- Date Parser 정규식 + 규칙 기반 12가지 패턴 파싱 후 Unix timestamp 범위로 변환 → Qdrant Range 필터 적용
- Context Sandwich: 프로필(최상단 고정) → [diary] + [plan] 검색 결과 태그 구분 → 프롬프트로 LLM 전달
- point\_id 기반 자동 업서트: couple\_{couple\_id}\_{type}\_{id} 형식으로 수정 시 벡터 자동 갱신

브라우징 로그를 기반으로 사용자 관심사를 분석하고 개인화된 뉴스·퀴즈를 추천하는 Chrome Extension 기반 마이크로 러닝 서비스

## 프로젝트 목표

웹 사용자의 브라우징 행동 데이터를 바탕으로 관심사를 추론하고, 그에 맞춰 개인화된 뉴스 및 퀴즈 형태의 마이크로 러닝 콘텐츠를 추천하는 것을 목표로 했습니다.

## 핵심 기여

- Chrome Extension에서 브라우징 로그를 수집해 서버로 전달하는 전체 파이프라인 설계·구현
- 수집한 로그를 기반으로 사용자 관심사 프로필 벡터 생성 로직 구현
- OffscreenDocument API와 Readability로 보안 제한 사이트에서도 콘텐츠 수집 기능 구현
- MongoDB 해시 색인으로 대용량 사용자 로그 분산 저장 및 관심사 프로필 벡터의 O(1) 증분 업데이트 구조 구현

## 사용 기술

Chrome Extension (MV3)

JavaScript

FastAPI

KoBART

Sentence Embeddings

Qdrant

MongoDB

## 주요 구현

- 크롬 확장 프로그램 로그 수집: Manifest V3 기반 확장을 통해 사용자의 브라우징 로그(체류 시간, 스크롤 깊이, 입력 비율, 접속 시간대 등)를 수집하여 서버로 보내는 파이프라인 설계·구현
- 관심사 프로필 벡터 생성: 수집된 로그를 백엔드에서 저장·전처리하고 문장 임베딩으로 사용자 프로필 벡터를 생성하는 로직 구현
- KoBART로 방문 뉴스 콘텐츠를 요약하는 배치 파이프라인 구축
- Offscreen Document 활용: Offscreen Document API와 Readability 라이브러리를 이용해 SPA 및 엄격한 CORS/CSP 사이트에서도 백그라운드에서 페이지 본문을 추출하여 로그 수집 제약 해소 (보안 정책으로 인한 콘텐츠 수집 불능 문제 해결)

## 기술적 포인트

- MongoDB 해시 색인 아키텍처: 사용자 ID 기반 해시로 MongoDB 컬렉션 색인을 구현하여 브라우징/히스토리 로그를 분산 저장하고, 사용자 수 증가 시에도 조회 부하를 최소화하는 확장성 확보
- 벡터 프로필 메타데이터 관리: 초기 히스토리 로그 전체를 임베딩해 사용자 관심사 초기 벡터를 생성하고, 체류 시간·스크롤 깊이에 비례한 가중치 메타데이터를 함께 저장
- O(1) 관심 벡터 증분 업데이트: 새로운 방문 로그 발생 시 기존 벡터와 가중치 메타데이터를 활용해 전체 히스토리를 재계산하지 않고도 상수 시간(O(1)) 내 프로필 벡터를 갱신하도록 설계 (실시간 프로필 업데이트)

## 성과

- O(1) 업데이트로 실시간 맞춤형 추천 제공
- 데이터 커버리지 확대: Offscreen Document 기반 수집으로 싱글 페이지 앱이나 보안 제한 사이트에서도 텍스트 로그 확보가 가능해져 추천 모델의 데이터 커버리지 증가 및 다양한 도메인의 콘텐츠에 대한 개인화 추천 구현
- 삼성 청년 SW·AI 아카데미 프로젝트 경연 우수상(2등) 수상

주니어 개발자의 PR·리뷰 진입 장벽을 낮추기 위해 GitHub 기반 코드 리뷰 과정을 재구성한 실시간 협업 웹 서비스

## 프로젝트 목표

주니어 개발자가 쉽게 코드 리뷰에 참여할 수 있도록 직관적인 UI/UX를 제공하는 것을 목표로 했습니다.

## 핵심 기여

- 실시간 협업 기능(WebRTC·tldraw·STOMP)을 제외한 프론트엔드 전반 담당
- PR 생성/리뷰 플로우 UI, Zustand 전역 상태관리, 공통 컴포넌트 구축
- 라우팅 및 SSE 알림 구조 설계

## 사용 기술

React Vite Zustand React Router SSE Tailwind CSS

## 주요 구현

- 브랜치·기준 PR 검증 → 컨벤션·제목 피드백 → 리뷰어 선택 → 코드/음성 코멘트 관리로 이어지는 5단계 PR 생성 플로우
- Repository 목록/상세 화면 (브랜치·상태 필터링, 잘못된 repold 접근 시 404 처리)
- PR 리뷰·머지·닫기·재오픈 화면 및 공통 UI 컴포넌트 (헤더·버튼·모달·토스트·스텝 인디케이터)

## 기술적 포인트

- Zustand로 사용자·알림·저장소 상태를 전역 관리
- SSE + useSSE 흑으로 GitHub 푸시 이벤트를 실시간 토스트/알림센터에 반영
- beforeunload/popstate 이벤트 + useCommentManager 흑으로 리뷰 작성 중 이탈 방지

## 성과

- 브랜치·상태 검증, 로딩·에러·빈 상태 처리로 GitHub 대비 직관적인 PR 생성·리뷰 경험 제공
- 팀 테스트 결과, 초보자 기준 PR 작성·리뷰 수행 시간 체감적으로 단축