소프트웨어공학 Teamproject

설계 문서 (Design Document) 2팀 이예린, 김재이, 김아리

1. 시스템 아키텍처 (System Architecture)
본 애플리케이션은 클라이언트 중심(Client-Centric Architecture) 구조를 기반으로 하며, Firebase는 인증(Authentication), 데이터 저장(Storage), 데이터베이스 관리(Database)를 담당한다. 본 문서는 「한 눈에 보는 일기장」의 시스템 아키텍처, 데이터 구조, UI 구성 및 핵심 모듈 설계를 기술한다.

전체 구조는 다음과 같은 단방향 흐름을 가진다.
[Flutter Client]
↓
[Firebase Authentication]
↓
[Firebase Realtime Database]
↘
[Firebase Storage]

1.1 아키텍처 특징
서버리스(Serverless) 구조: 별도의 백엔드 서버 없이 Firebase 기능만으로 앱이 운영된다.
클라이언트 로직 강화: UI 렌더링, 데이터 조작, 유효성 검증 등의 대부분의 로직은 Flutter에서 처리된다.
실시간 동기화: Realtime Database를 활용하여 즉각적인 데이터 반영을 지원한다.
저비용 구조: Firebase의 무료 플랜으로도 충분히 동작 가능하도록 설계됨.

2. 시스템 아키텍처
2.1 기술 스택
Frontend: Android Studio 기반 (Kotlin/Java 또는 Flutter)
Backend: Firebase Realtime Database, Firebase Storage, Firebase Authentication
Deployment: APK 및 WebApp(테스트용 QR 링크)
Design: Figma

2.2 전체 아키텍처 개요
클라이언트 중심 구조 (Client-Centric, Serverless)
Firebase가 인증, 데이터 저장, 파일 저장 역할 수행
데이터 흐름
사용자 익명 로그인
→ UID 기반 User 노드 생성
→ 달력에서 날짜 선택 후 일기 작성
→ Diary 데이터는 users/{userId}/diaries/{date}에 저장

→ 이미지 업로드 시 Firebase Storage에 저장 후 URL을 DB에 반영
→ 커플 연결 시 coupleCodes에서 초대코드 검증
→ 검증 후 couples/{coupleId} 노드 생성 및 각 사용자에 coupleId 저장
→ 커플 모드 일기는 couples/{coupleId}/diaries/{date}에 저장
→ 각 사용자 권한: 자기 일기 수정/삭제, 상대 일기는 읽기 전용

3. 데이터베이스 설계

3.1 개념적 모델 (Conceptual Model)
User
Diary
Couple
CoupleDiary
CoupleCode

3.2 논리적 모델 (Logical ERD – 텍스트)
User (1) – (N) Diary
User (1) – (0..1) Couple 관계 (coupleId)
Couple (1) – (N) CoupleDiary
CoupleCode (1) – (1) User (초대 코드 생성자)
User
userId (PK)
nickname
profileImage
coupleId (nullable)
Diary
diaryId
userId (FK: User)
date (YYYY-MM-DD)
emotion, emotionColor
text, imageUrl, timestamp
Couple
coupleId (PK)
user1, user2
status, createdAt
CoupleDiary
coupleId (FK)
date
sharedImage
user1(text, emotion)
user2(text, emotion)

CoupleCode
inviteCode (PK)
userId, createdAt, expiresAt

3.3 Firebase Realtime Database 구조 (실제 물리 모델)

```
/users/{userId}/
  profile/
    nickname
    profileImage
  diaries/{date}/
    diaryId
    emotion
    emotionColor
    imageUrl
    text
    timestamp
  coupleId
/couples/{coupleId}/
  user1
  user2
  status
  createdAt
  diaries/{date}/
    sharedImage
    user1/
      text
      emotion
    user2/
      text
      emotion
/coupleCodes/{inviteCode}/
  userId
  createdAt
  expiresAt
```

3.4 Firebase Storage 구조

```
storage/
└── diaries/
    └── {userId}/
        ├── {diaryId}.jpg
        ├── {diaryId}.png
```

└── ...

4. 모듈 설계 (Module Design)

4.1 Authentication Module (AuthService)
익명 로그인(Anonymous Auth) 처리
UID 발급 및 관리
커플 연결 상태(coupleId) 조회/갱신

4.2 Diary Module (DiaryService)
createDiary(userId, date, text, emotion, image)
updateDiary(userId, date, newData)
deleteDiary(userId, date)
getDiary(userId, date)
달력 화면용 데이터(썸네일/감정) 요약 조회

4.3 Couple Module (CoupleService)
createInviteCode(userId)
connectCouple(inviteCode, partnerUserId)
disconnectCouple(coupleId)
getCoupleDiary(coupleId, date)
setCoupleDiary(coupleId, date, userRole, data)

4.4 Storage Module (StorageService)
uploadDiaryImage(userId, diaryId, imageFile)
deleteDiaryImage(userId, diaryId)
Storage 오류 핸들링 및 재시도 로직

4.5 Analytics/Logging Module (추후 확장)
감정 통계 기반 집계
Crashlytics/Analytics 이벤트 로깅

5. 데이터 설계 원칙 및 이유
5.1 비정규화(De-normalization) 기반 설계
Firebase Realtime Database는 관계형 DB가 아닌 NoSQL 구조이므로,
불필요한 깊은 참조를 피하기 위해 flat 구조 기반으로 설계하였다.

5.2 날짜 기반 접근 최적화
일기장 앱 특성상 사용자의 조회 패턴이 날짜 기반이므로
diaries/날짜 구조로 빠르게 접근할 수 있도록 구성했다.

5.3 감정 통계 저장 전략
감정별 횟수는 내부 캐시 or 별도 통계 노드에 저장 가능
또는 매월 통계를 재계산하여 최신 값만 유지 가능
→ 둘 중 어떤 전략을 사용해도 DB 부하가 낮음

5.4 데이터 무결성 보장
저장 시점에 timestamp 저장
Storage 이미지 삭제 시 DB 이미지 경로 동기 삭제
UID 기반 관리로 사용자별 데이터가 분리되어 있음

6. UI 설계 (UI Structure)
UI는 직관적 구조 + 초보 사용자 경험 최적화를 목표로 설계되었다.

6.1 홈(Home) 화면
구성 요소
감정, 기록 수, 커플 기록 수 등,
최근 업로드한 사진 케러셀

설계 의도
사용자가 자신의 감정 패턴을 쉽게 이해하도록 시각화
PageView 애니메이션으로 인터랙션 완성도 증가
"이 달의 기록활동"을 보여줘 동기부여 효과 제공

6.2 달력(Calendar) 화면
구성 요소
table_calendar 패키지 기반 월별 렌더링
날짜별 썸네일 표시
기록 여부 dot 표시
설계 의도
앱 이름("한 눈에 보는 일기장")과 직접적으로 연결되는 핵심 UI
날짜별 감정·이미지를 바로 확인 가능
사용자가 기억을 회상하기 쉽도록 시각적 힌트 제공

6.3 작성(Create) 화면
기능 구성
텍스트 입력
감정 선택 UI
이미지 첨부(image_picker)
저장 버튼
설계 목적
기록 과정을 단순화하여 20초 내 작성 완료 가능

감정 선택 UI를 명확하게 분리하여 사용자 혼동 방지
저장 중 업로드 표시 제공(로딩 UX 포함)

7. 흐름도 (Flow Description – Text Version)
7.1 일기 작성 프로세스 흐름

```
[사용자]
    ↓
날짜 선택 (달력 화면)
    ↓
"기록하기" 버튼 클릭
    ↓
[작성 화면 진입]
    ↓
텍스트 입력
    ↓
감정 선택
    ↓
이미지 첨부 (Optional)
    ↓
저장 버튼 클릭
    ↓
[유효성 검사 수행]
  - 텍스트 비어있는지 확인
  - 감정 선택 여부 확인
  - 이미지 용량 검사
    ↓
Storage 업로드 요청
    ↓
업로드 성공?
  ├── YES → imageUrl 생성 → DB 저장
  └── NO  → 오류 알림 → 재시도
    ↓
DB 저장 완료
    ↓
감정 카운트 증가(updateEmotionCount)
    ↓
홈 화면 통계 자동 반영
    ↓
[작성 완료]
```