**High-level Architecture**

**UML Diagrams Document**

Project Name: 시대생 모여라

Team Name: 경김권김송최

Team Member:

2020920001 경민수

2020520008 김세연

2021920006 권민재

2021920020 김혜주

2022920038 송채희

2022920060 최재원

Date: 2024. 11.16

Version info: Version 4

[Table Of Contents ]

**1.** [High level architecture](#일)

2. [Class Diagrams for static view](#이)

3. [Sequence/Activity/State etc for dynamic view](#삼)

4. [Revision History , Table Of Contents](#사)

1. [High level architecture](#일)
   1. Cache 도입 및 push model/kafka

텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

활동 기록이 포스트 될 때마다 피드에 업데이트, 피드 조회는 캐시로부터

* feed cache: 사용자별 피드에 들어가 활동 기록 id 리스트를 저장한 캐시가 존재한다. DB에서 직접 조회를 하지 않는 만큼 읽기 속도가 많이 향상된다.
* push model: 포스트될 떄마다 해당 모임 구성원들의 피드 캐시에 해당 포스트를 업데이트한다. 이 경우 모임의 사용자가 많은 경우 피드 갱신에 많은 시간이 소요될 수 있으며, 서비스를 자주 사용하지 않는 이의 피드까지 갱신하므로 불필요한 컴퓨 팅 자언이 소모된다.
* 이 경우 post할 때마다 messaging queue service인 kafka를 이용하여 비동기 처리한다. 이로 인해 서버의 처리 속도를 향상시킬 수 있다.
* pull model: 피드를 읽는 시점에 데이터를 조회하여 갱신한다. 이 경우 active user에 대해서만 컴퓨팅 자원이 소모된다. 하지만 들어있는 모임이 많을수록 읽는 데 시간이 많이 소요될 수 있다.
  1. 전체 아키텍처: 읽기 성능을 위한 cache server, 전체 서비스를 제공하는 WAS, file 저장용 S3 server, PostgreSQL로 구성된다.

텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

* 1. ERD diagram

스크린샷, 텍스트, 도표, 디자인이(가) 표시된 사진

자동 생성된 설명

1. [Class Diagram for static view](#이)

텍스트, 도표, 평면도, 라인이(가) 표시된 사진

자동 생성된 설명

클래스 다이어그램은 시스템의 정적 구조를 나타내는 다이어그램으로, 클래스 간의 관계 및 각 클래스의 속성과 메서드를 나타낸다. 주어진 요구사항을 바탕으로 시스템의 주요 클래스, 그들 간의 관계, 그리고 속성과 메서드를 정의할 수 있다.

**클래스 다이어그램 구성 요소**

* **사용자 (User)**: 시스템에 로그인하고, 회원가입 및 탈퇴를 수행하는 역할을 하는 클래스이다.
* **모임장 (MeetingLeader)**: 모임을 개설하고, 구성원 관리 및 활동 내역을 기록하는 역할을 맡은 클래스이다.
* **부모임장 (MeetingSubLeader)**: 모임장과 비슷하지만, 개설된 모임에 대한 정보와 구성원 관리 및 활동 기록을 관리하는 역할을 하는 클래스이다.
* **모임원 (MeetingMember)**: 참여한 모임의 기록을 열람하고, 평가하는 역할을 담당하는 클래스이다.
* **모임 신청자 (MeetingApplicant)**: 모임에 참가하기 위해 모임을 조회하고 신청할 수 있는 역할을 하는 클래스이다.
* **모임 (Meeting)**: 모임에 대한 정보, 구성원 및 활동 기록을 관리하는 클래스이다.

**클래스 간 관계**

* **상속 (Inheritance)**: Leader와 SubLeader 클래스는 모두 User 클래스를 상속받는다.
* **연관 관계 (Association)**:
  + User 클래스는 Group 클래스와 연관되어 있다. 사용자는 하나 이상의 모임에 참여하거나 개설할 수 있다.
  + Group 클래스는 여러 Activity 클래스를 포함할 수 있다. 모임 활동 기록이 작성되고 수정될 수 있다.
  + Leader와 SubLeader는 Group과 연관되며, 그들은 각각 모임을 관리하거나 활동 기록을 수정하는 역할을 한다.

1. [Sequence Diagram for dynamic view](#삼)

텍스트, 번호, 영수증이(가) 표시된 사진

자동 생성된 설명

시퀀스 다이어그램은 시스템 내에서 객체들 간의 동적 상호작용을 나타내는 다이어그램으로, 각 객체의 생명주기와 메시지 흐름을 나타낸다. 이 다이어그램을 통해 사용자의 특정 행동이나 시스템의 기능이 어떻게 실행되는지에 대한 흐름을 시각적으로 표현할 수 있다.

**시퀀스 다이어그램 구성 요소**

* **사용자 (User)**: 시스템을 사용하는 주체로, 다양한 작업을 수행한다.
* **시스템 객체 (System)**: 사용자 요청을 처리하고 결과를 반환하는 객체들로, 예를 들어 Meeting, MeetingLeader, MeetingMemeber등이 있다.
* **메시지 흐름 (Message Flow)**: 사용자가 특정 작업을 요청할 때 객체 간의 메시지 흐름을 보여준다.

**시퀀스 다이어그램 예시: '모임 개설'**

1. 사용자 (User)가 **시스템**에 로그인한다.
   * 메시지: login(email, password)
   * 시스템은 사용자 정보를 확인하고 로그인 결과를 반환한다.
2. 사용자 (User)가 **모임장 (Leader)** 역할을 선택하고 **모임 개설**을 요청한다.
   * 시스템은 모임 정보를 받아 새 모임을 생성하고, 성공적으로 모임이 개설되었음을 반환한다.
3. **시스템**은 **Leader**에게 생성된 모임에 대한 정보를 반환하고, **Leader**는 모임을 관리하기 위한 메서드들을 사용한다.
   * 시스템은 모임 상세 정보를 표시한다.

**시퀀스 다이어그램 예시: '모임 신청'**

1. 모임 신청자 (Participant)가 모임 목록을 조회한다.
   * 시스템은 필터된 모임 목록을 반환한다.
2. 모임 신청자 (Participant)가 원하는 모임을 선택하고 신청한다.
   * 시스템은 해당 모임에 참여 신청을 처리하고, 신청이 완료되었음을 반환한다.
3. **시스템**은 신청자의 상태를 갱신하고, 신청한 모임에 참여자로 추가된 것을 확인한다.

**시퀀스 다이어그램 예시: '모임 활동 기록 작성'**

1. **모임장 (Leader)** 또는 부모임장 (SubLeader)가 활동을 기록하려고 요청한다.
   * 시스템은 해당 활동을 기록하고 활동 ID를 반환한다.
2. **시스템**은 모임의 활동 기록을 저장하고, 해당 기록을 조회할 수 있도록 한다.
3. **Leader** 또는 **SubLeader**는 기록된 활동을 조회하고, 수정하거나 삭제할 수 있다.
   * 시스템은 활동 기록을 수정하거나 삭제하고 결과를 반환한다.
4. [Revision History](#사)

|  |  |  |
| --- | --- | --- |
| 순번 | 내용 | 비고 |
| 1 | ERD diagram 작성 | 2024.10.01 |
| 2 | 아키텍처 후보 아키텍처 선정 (분산서버, 분산DB, DB다중화, Cache 도입 및 push model/kafka) | 2024.10.25 |
| 3 | 최종 아키텍처 선정 | 2024.10.27 |
| 4 | documentation check list에 맞추어 문서 수정 | 2024.1210 |