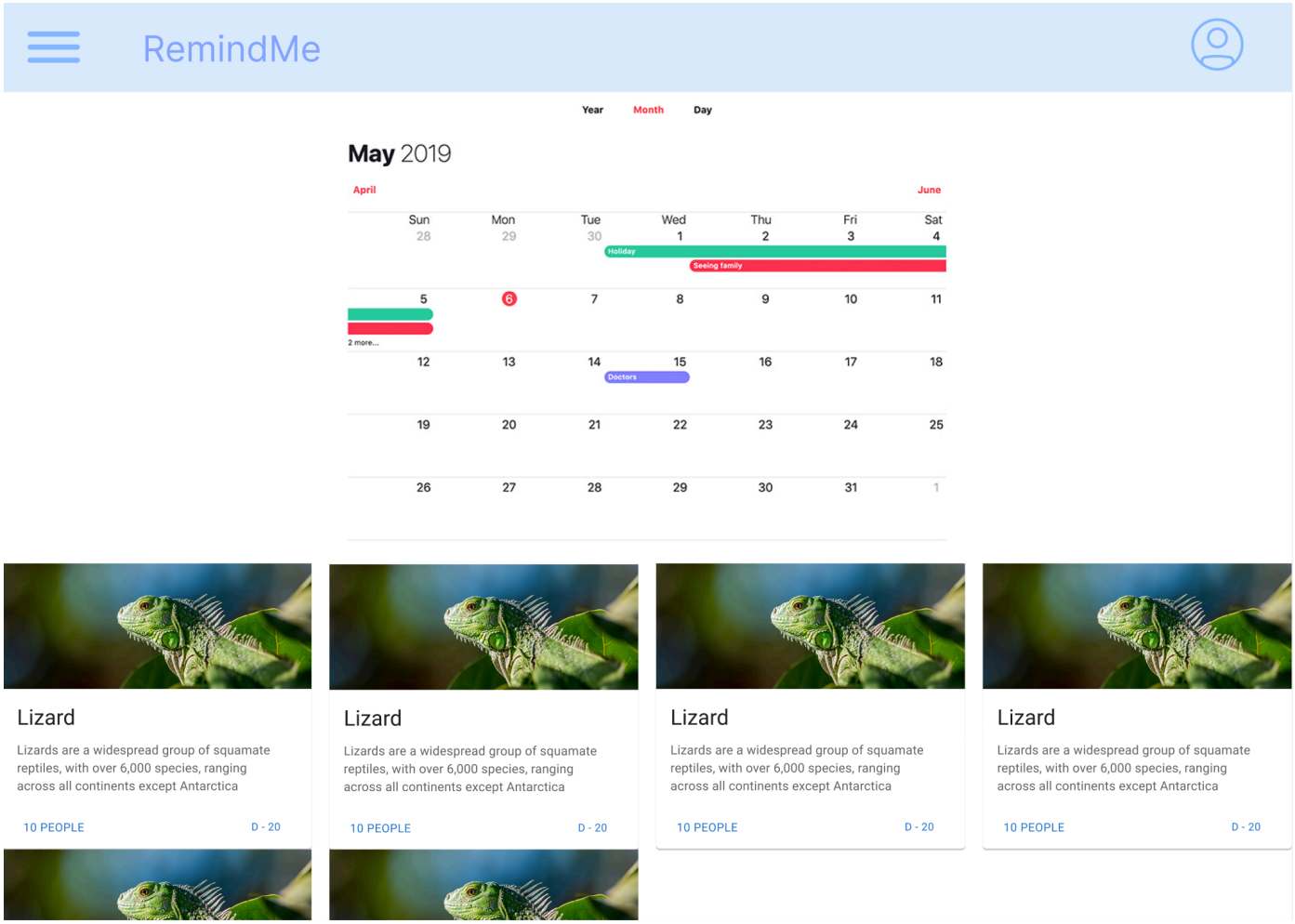
RemindMe

개요

주기적인 일정이 아니라 산발적인 일정들을 편하게 관리하기 위하여 여러 사람이 참가하는 각각의 일정을 카드 형식으로 D-day와 일정, 설명등을 확인할 수 있는 일정 관리 어플리케이션.

카드 형식으로 각각의 일정을 볼 수 있어 다가오는 일정들을 한눈에 볼 수 있고 달력으로 전체적인 일정도 같이 관리할 수 있도록 제작된 어플리케이션입니다.



어플리케이션 기능

1. 개인 스케줄 관리를 위한 구글 로그인.
2. 스케줄 생성 및 멤버 추가.
3. 편리한 스케줄 멤버 추가를 위한 친구 추가 기능.
4. 달력을 통한 전체 스케줄 관리.
5. 스케줄 카드를 통한 스케줄 확인 및 관리.

개발 환경

1. Spring 개발 환경

- springboot
- junit
- jpa
- mysql-connector-java
- logback-classic
- slf4j
- spring-boot-starter-security
- spring-boot-starter-oauth2-client
- spring-security-oauth2-autoconfigure
- spring-security-test
- spring-boot-test-autoconfigure
- spring-boot-starter-data-redis
- spring-session-data-redis

2. Database

mysql 8.0.29

어플리케이션 특징

1. JPA 사용.

- JPA를 사용하여 특정 데이터베이스에 종속되지 않도록 하였습니다.
- JPA를 사용하여 Entity위주의 객체지향적 프로그래밍을 하였습니다.
- JPA의 구현체는 Hibernate사용하였습니다.
- Entity의 생성시간과 수정시간을 관리하기 위하여 JPA Auditing기능을 사용하였습니다.

2. SLF4J을 통한 Logging, 구현체로 Logback-classic 사용.

- SLF4J Logging Framework를 사용하여 Logging 구현체에 관계 없이 일관된 Logging을 구연하였습니다.
- SLF4J의 구현체로는 Logback-classic을 사용하였습니다.

3. Git을 통한 Version Control.

- Git을 사용하여 Version Control을 하였습니다.
- Git Repository로는 Github를 사용하였습니다.

4. Spring Secutiry를 이용한 Authorization 설정.

- Spring Security를 사용하여 User의 Role에 따라 다르게 요청을 처리.
- Spring Security를 사용하여 User의 Authorization을 기억.

5. OAuth 2.0을 사용한 Google 로그인 구현.

- Spring Security의 oauth2Login기능으로 Google Oauth 2.0 Login을 구현하였습니다.
- UserInfoEndpoint의 OAuth2UserService를 직접 구현하여 Google에서 제공하는 User의 Attritubes를 Session에 저장할 수 있게 하였습니다.

6. Session저장소로 Redis를 사용.

- MSA를 사용하게 될 시 발생할 수 있는 Login Session의 문제를 해결하기 위해 HttpSession의 저장소를 In-Memory에서 Redis로 교체하여 해결하였습니다.
- 이외의 HttpSession의 정보를 여러 서버에서 공유하기 위하여 Redis 서버를 따로 두었습니다.
- Session에 저장할 DTO등의 Class등은 Serializable를 implements하여 외부로 저장할 수 있게 하였습니다.

7. Junit5를 이용한 단위 테스트.

- 단위 테스트를 통하여 코드를 테스트하고 검토하였습니다.

개발 예정 사항

CI/CD 구현

- AWS에 EC2 인스턴스를 생성하여 배포 서버로 사용.
- Jenkins를 사용하여 Github Repository를 polling해 백엔드 서버에 배포.

Google Calendar 연동