인턴 최종 발표 보고서

장연수

목차

1) 진행 경과

1주차(3/20~3/23) -> Google Calendar API 조사. 월별 캘린더 출력. 하루 일정 출력. Java Spring 공부

2주차(3/26~3/30) -> 프로젝트 구조 설계. 모델링. AJAX를 이용한 비동기 전송 방식 이용. 이어지는 일정 출력. 체크된 여러 캘린더들의 일정 출력. 일정의 이름을 클릭하면 일정의 정보가 화면에 나오는 것을 구현.

3주차(4/2~4/6) -> 일정이 날짜 칸을 넘어가는 경우 더보기 기능 구현. 일정 상세 정보 페이지 구현. 일정 생성, 수정, 삭제 구현. UI 개선. 목록 캘린더(한달 동안의 일정 목록) 출력.

4주차(4/9~4/13)-> 일정 알람 생성, 수정, 일정 초대 기능 구현(참석자 목록 보여주기). thread를 이용한 응답속도 개선. 페이지 이동 없앰. 일정을 취소하거나 저장하면 사용자가 전에 봤던 페이지로 이동하게 하는 기능 구현

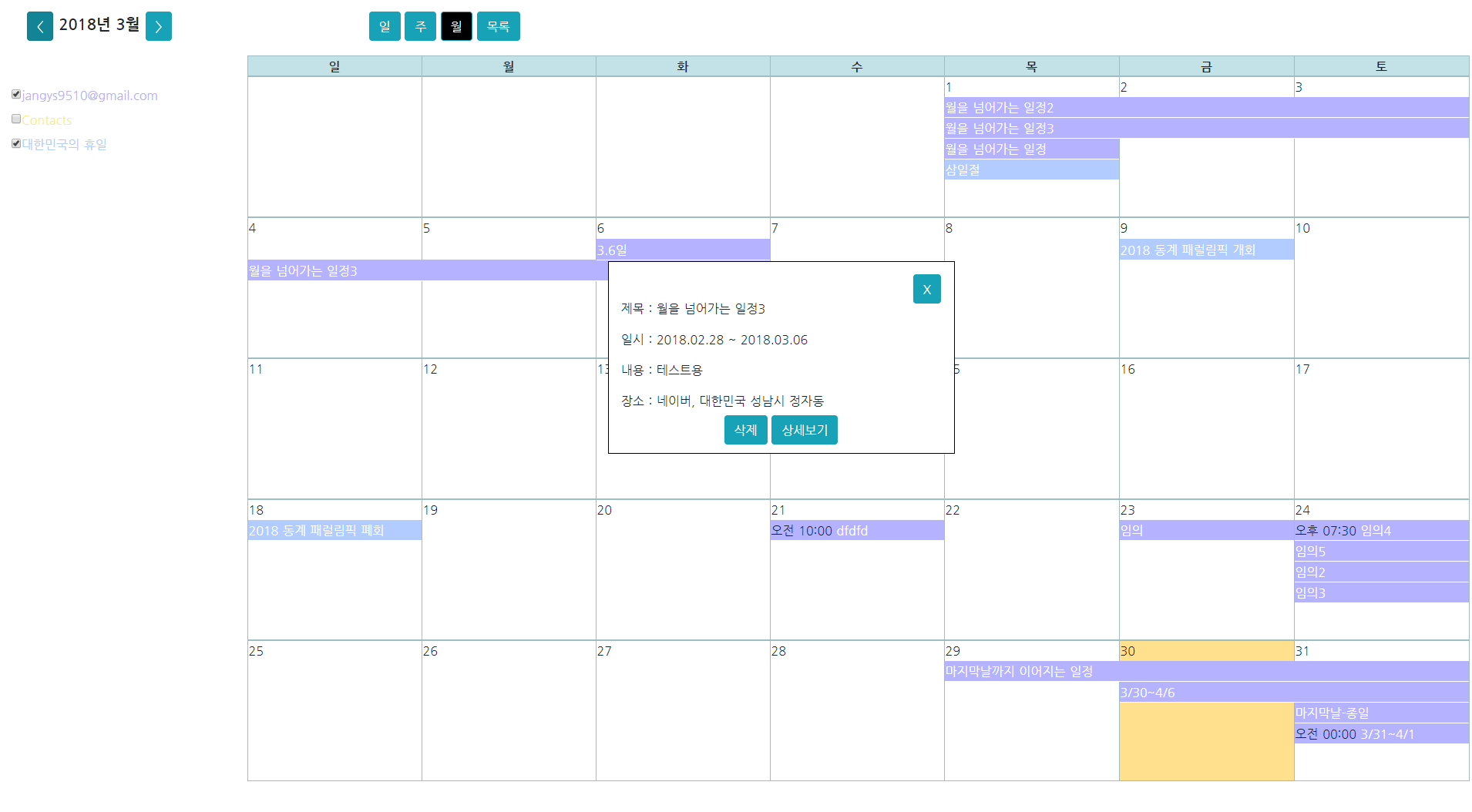
5주차(4/16~4/20)-> 참석 여부를 요약창에서 선택하는 기능 구현. 일정 요약창에 참석자 목록 볼 수 있는 기능 구현. 참석 여부 캘린더에 표시. 반복 일정 처리, 생성. 반복 일정 정보 상세 페이지에 표현. 반복 일정 맞춤 선택지 제공.

6주차(4/23~4/27)-> 반복 일정 수정 구현(3가지 중 선택). 반복 일정 삭제. side에 미니 캘린더 추가. 종일 일정 알림 표시 변경. 캘린더 생성, 수정, 삭제 구현. 캘린더 접근 권한 생성, 수정, 삭제 구현. daily/weekly 캘린더 종일 일정 부분 구현.

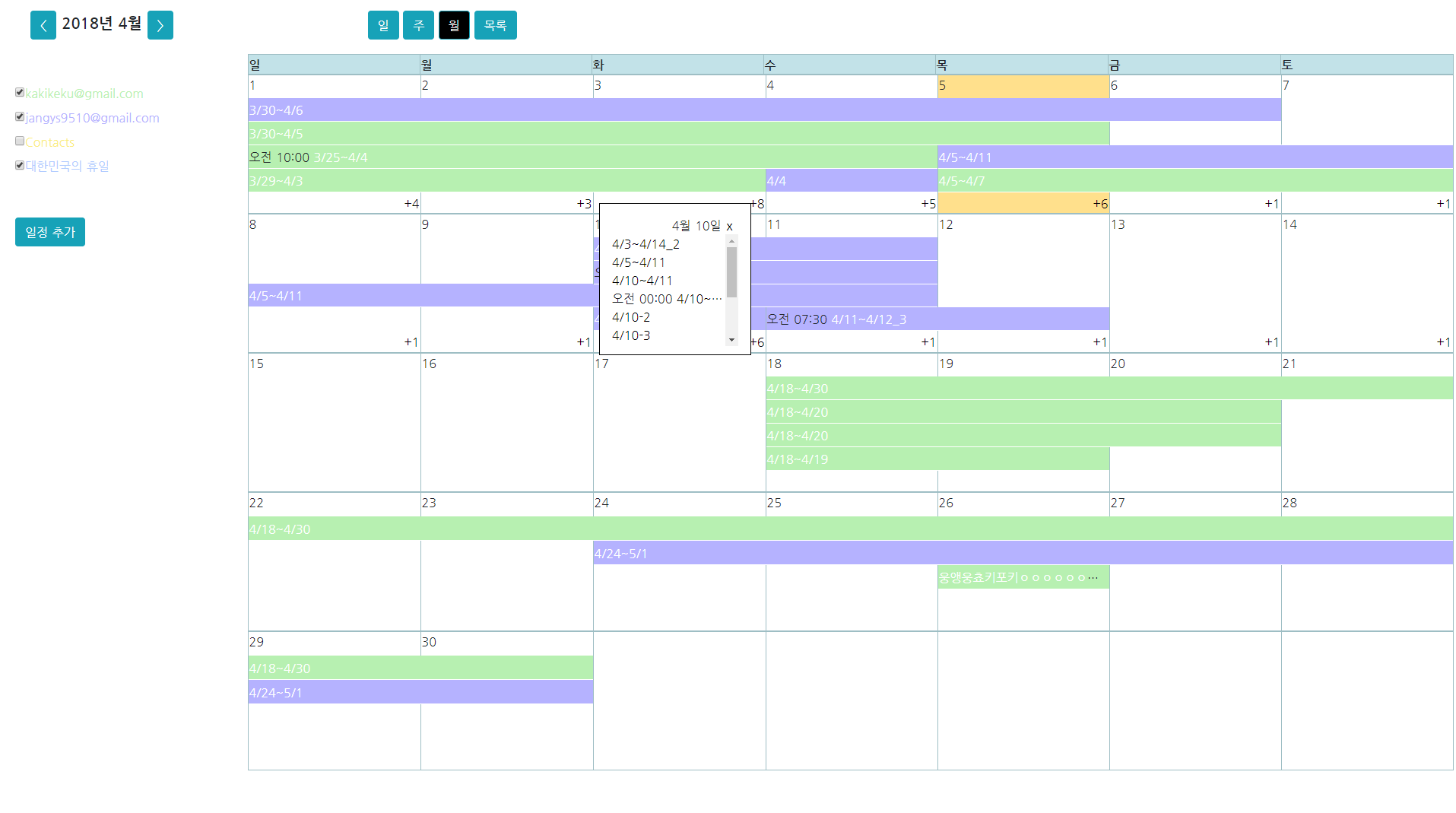
7주차(4/30~5/4)-> daily/weekly 캘린더 시간이 있는 일정 부분 구현. daily/weekly 캘린더에서 드래그하여 일정 생성하는 기능 구현. daily/weekly 캘린더 현재 시간 표시. monthly 캘린더의 일정을 원하는 날짜로 드래그하여 옮기는 기능 구현. daily/weekly 캘린더 시간 일정 드래그 하여 시간대 조정하는 기능 구현.

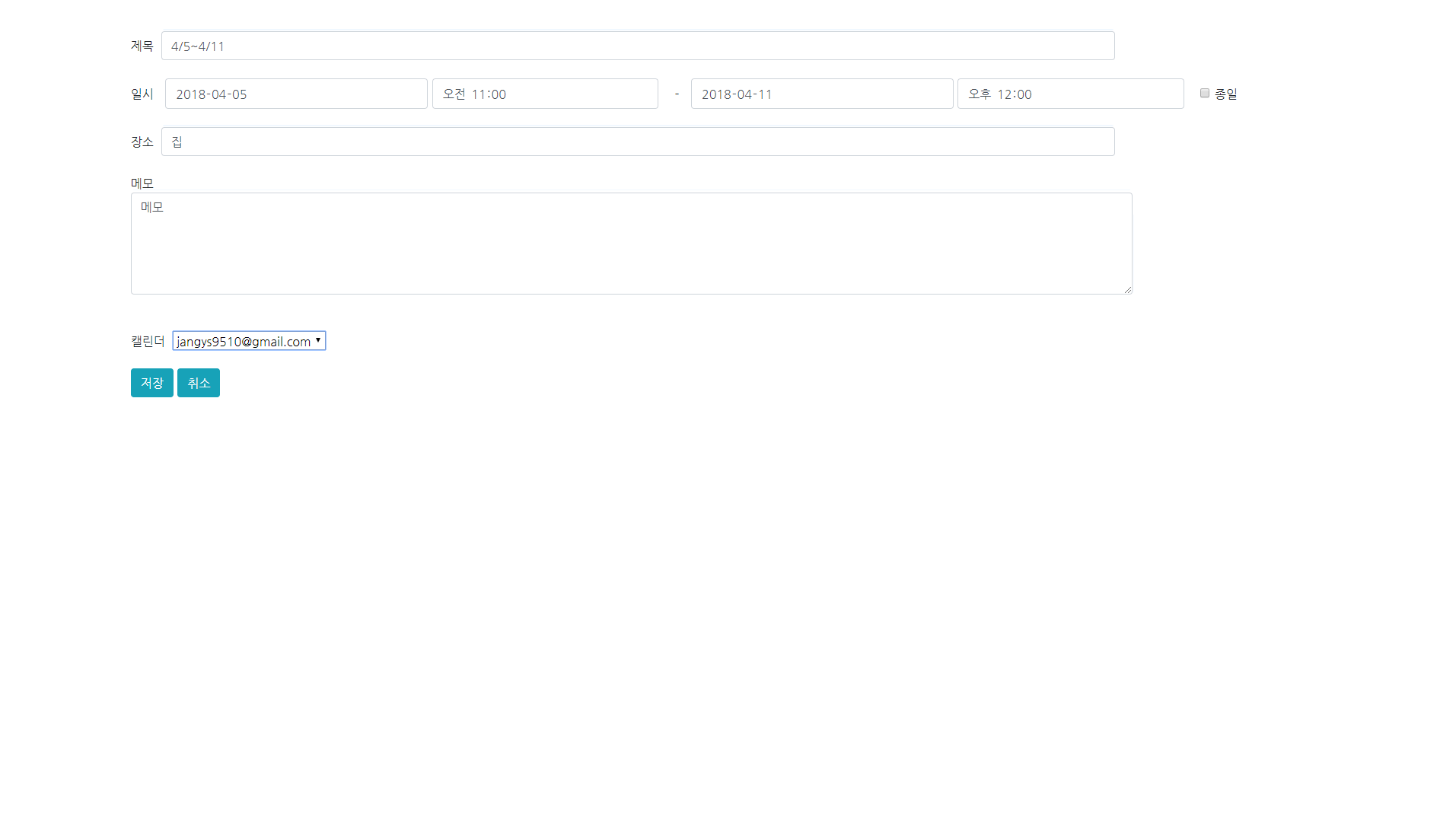
8주차(5/7~5/11)->캘린더 테스트 하며 오류 해결. 일정 상태 변경 구현. 일정 공개 여부 변경 구현. 테스트 코드 일부 작성. 일정에 업로드 된 파일 표시

9주차(5/14~)->캘린더 내보내기 기능 구현

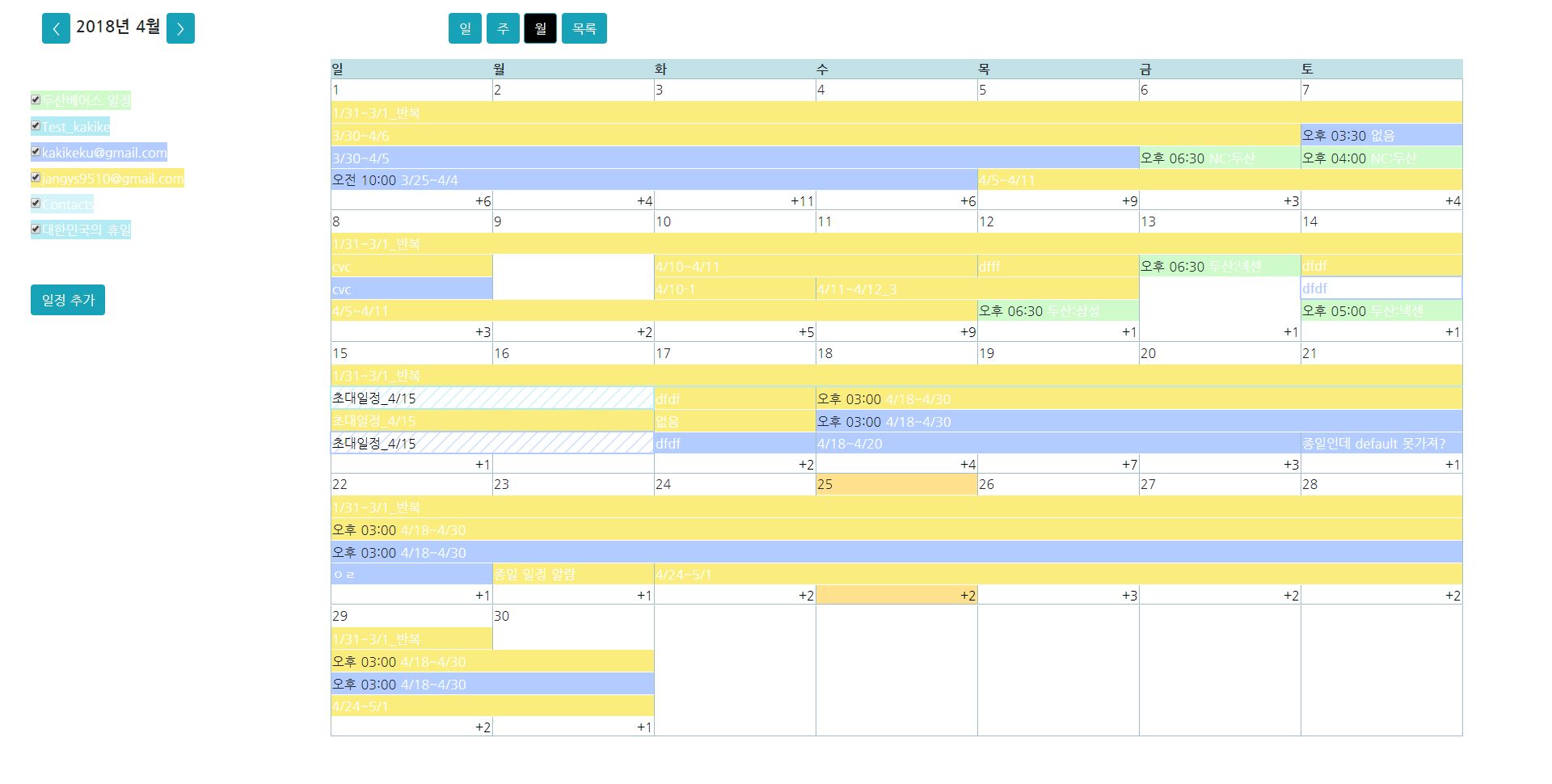


* 3/30 UI 모습

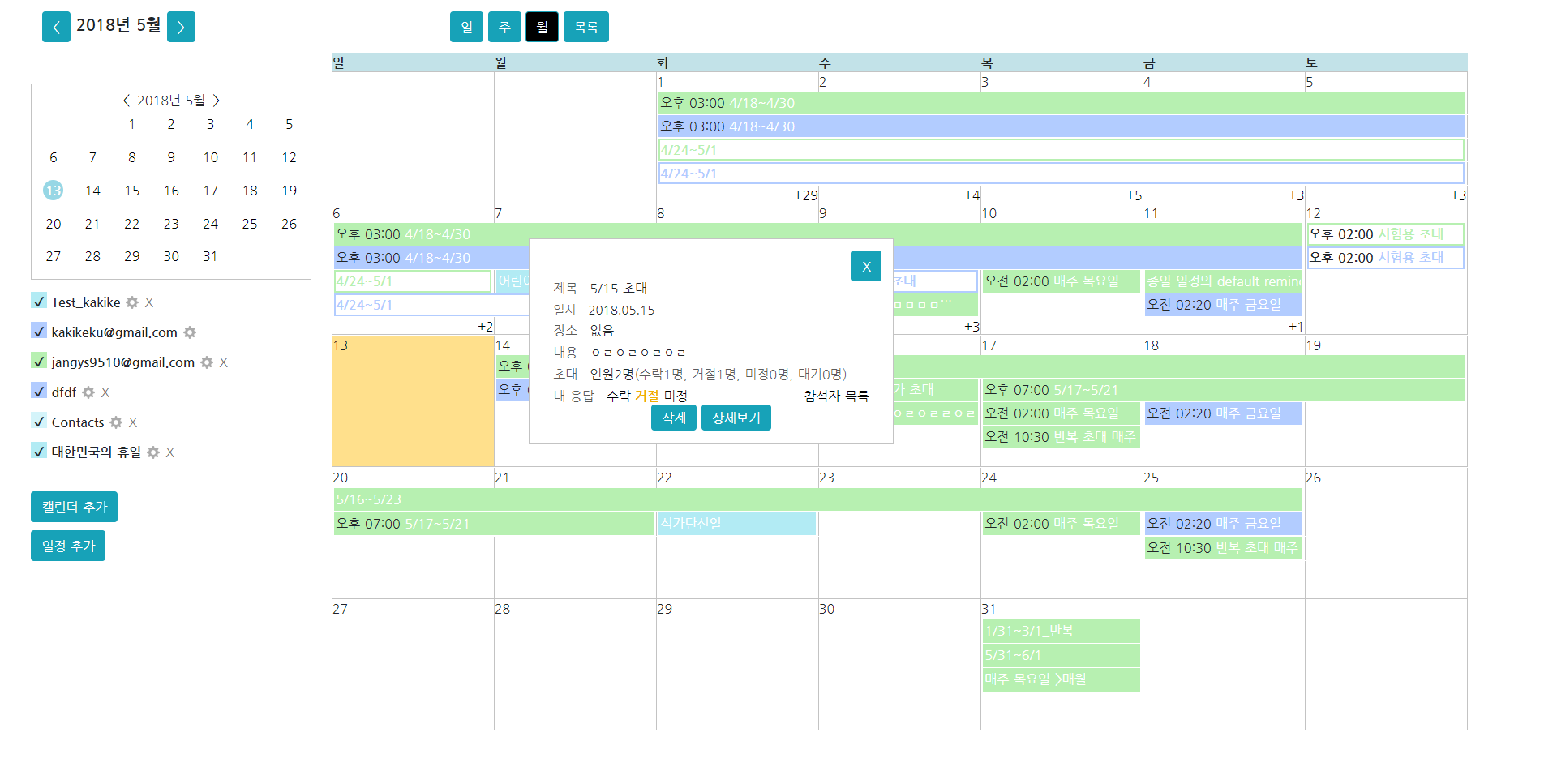
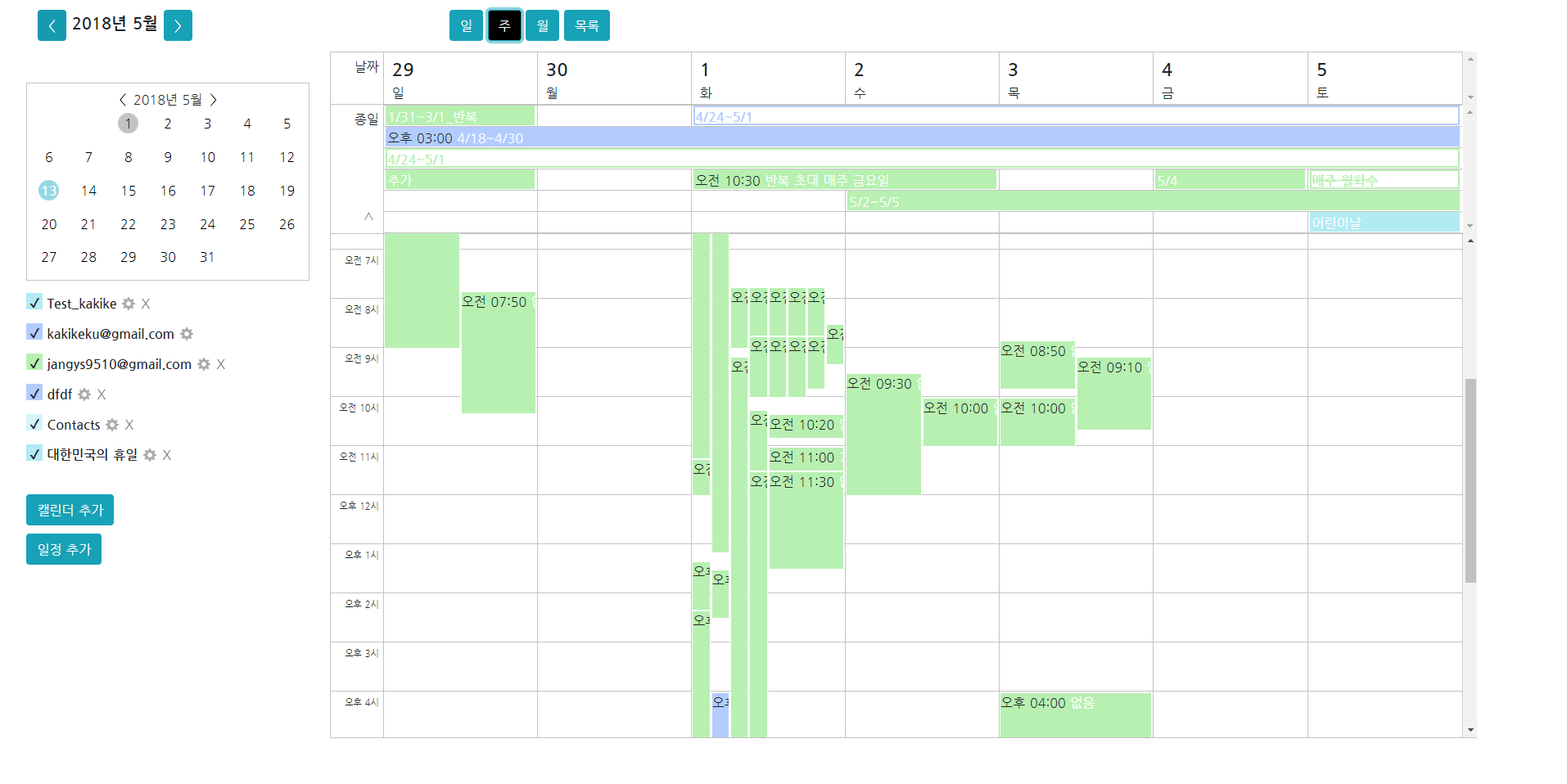
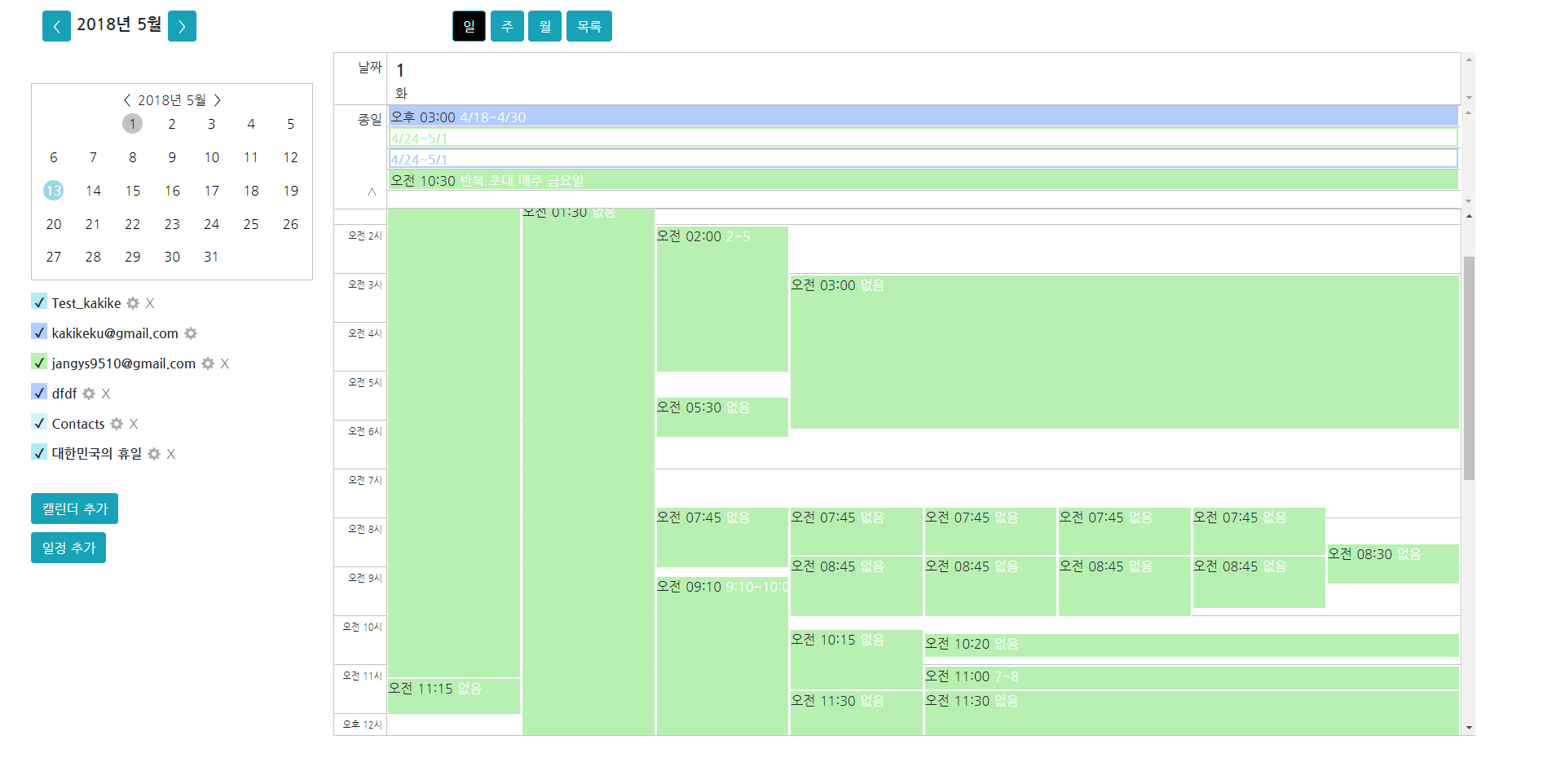


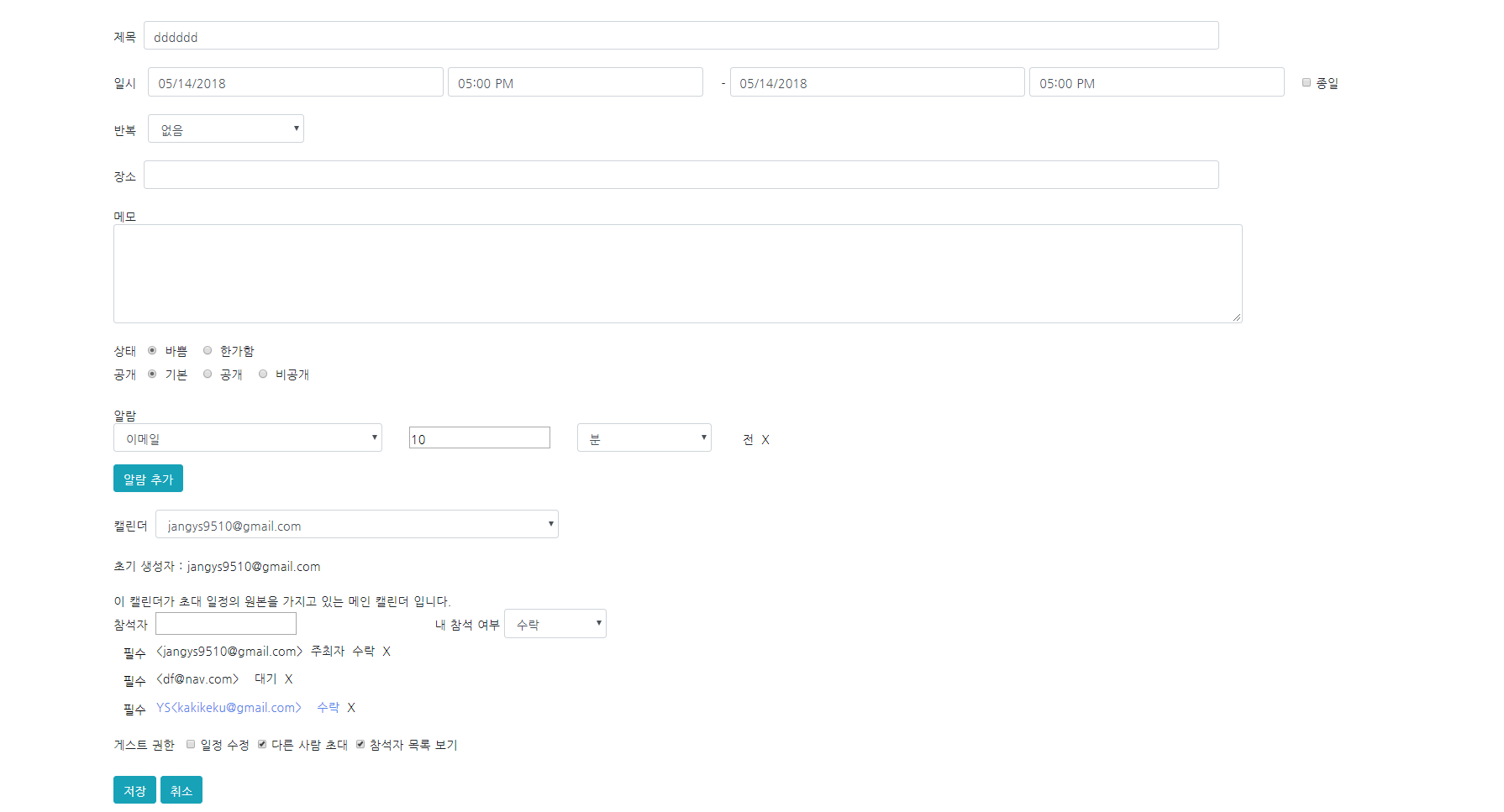


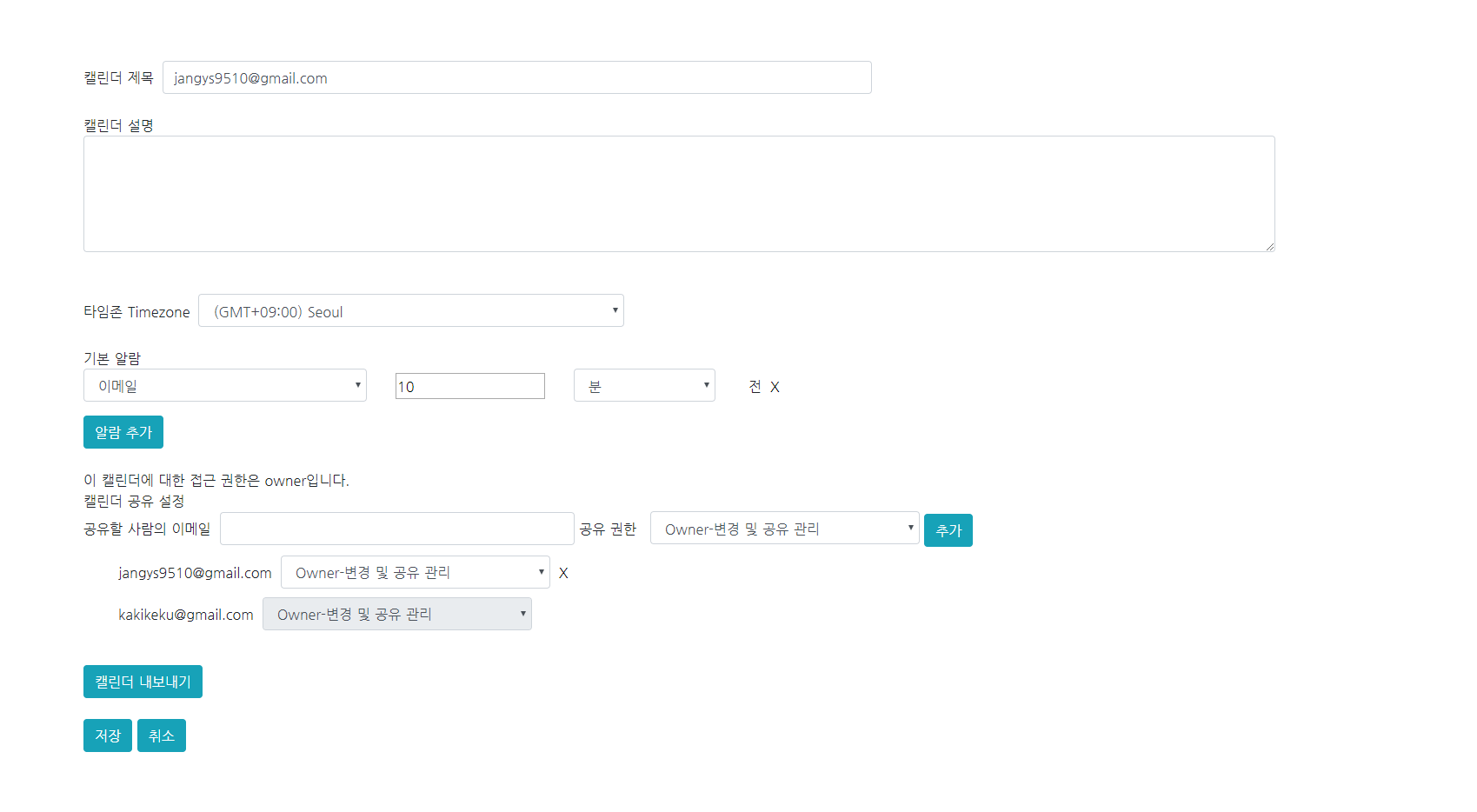
* 4/4 까지 UI



* 4/25까지 UI







* 최종 UI

2) Google Calendar API 소개

Google Calendar API를 쉽게 사용할 수 있도록 라이브러리를 제공

구글 계정 내에 있는 캘린더를 관리할 수 있게 해 줌.

OAuth 2.0 인증 이용

3) 구현한 기능

daily, weekly, monthly, list 캘린더에 일정 표시, 생성, 수정 기능

* 일정 클릭 시 요약창
  + 요약창에서 일정 삭제 가능

일정 상세보기 페이지에서 일정 생성, 수정

* 일정 초대
* 반복 일정 생성, 수정

캘린더 생성, 수정, 삭제

* 캘린더 공유 기능
* 캘린더 내보내기 기능

4) 기능 구현 방법

API 사용을 위한 사전 준비

Pom.xml에 Google Calendar API 라이브러리를 쓰기 위한 라이브러리들을 추가함.

* com.google.api-client -> Google api를 더 쉽게 사용하기 위해 제공. 인증을 쉽게 만들어 줌. 구글 로그인 기능 제공.
* org.springframework.social -> Spring Social extension. 소셜 로그인을 해줌.
* com.google.oauth-client -> OAuth 인증 방식을 이용한 구글 인증.
* com.google.apis -> Calendar API v3-rev235-1.22.0 Calendar API를 사용하기 위한 라이브러리 제공.
* org.apache.httpcomponents -> Apache HttpComponents Client HTTP 관련 low level의 java components를 생성하고 유지할 수 있도록 함.

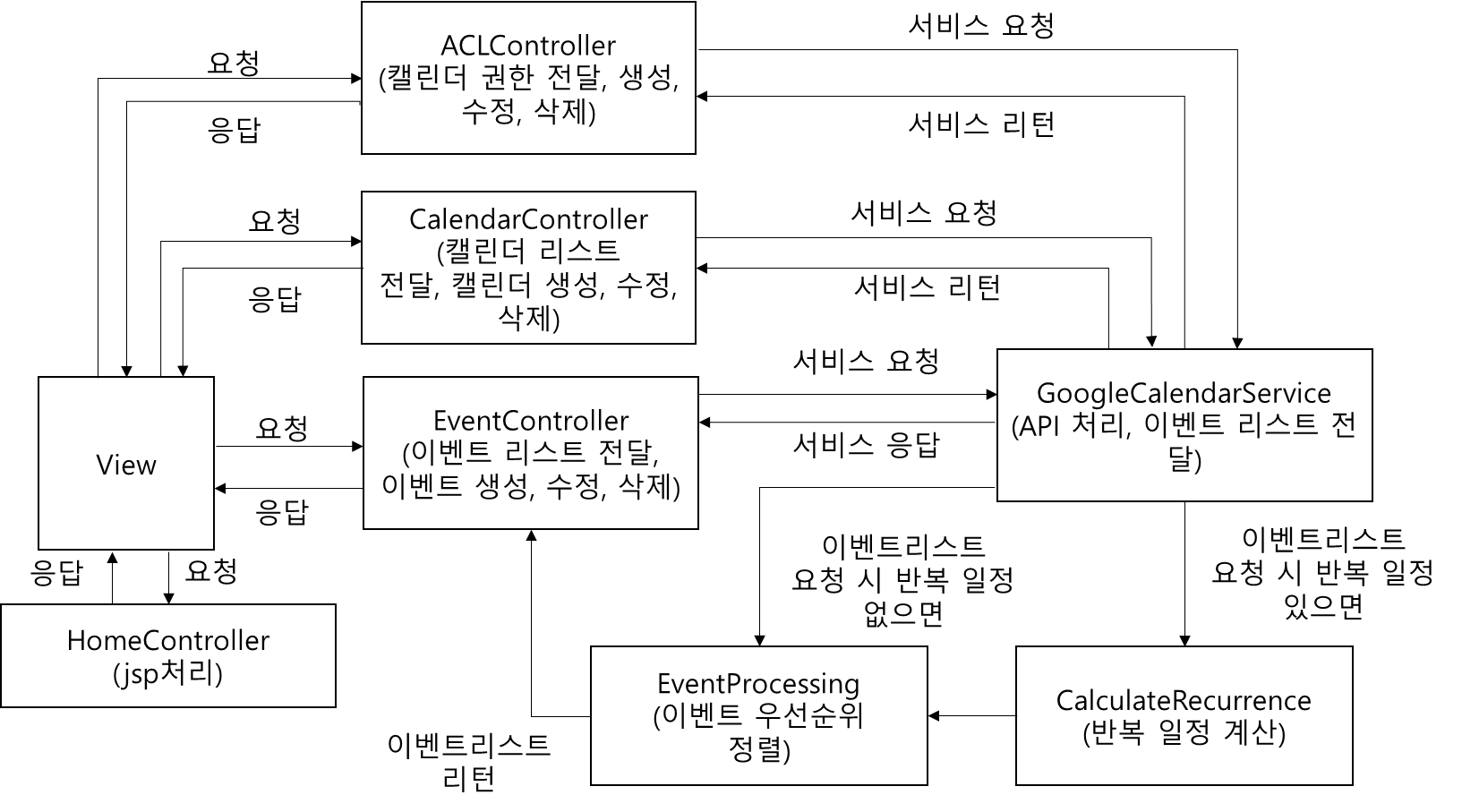
Google Cloud Platform에 접속.

* 사용자 인증 정보를 만들어서 Oauth 클라이언트 ID(특정 사용자의 구글 안에서의 정보를 가져오는 경우)를 받음.
* client\_secret.json 파일을 다운 받아 프로젝트에 저장.
* 프로젝트를 생성하고 사용할 API 설정.

캘린더 공유 설정

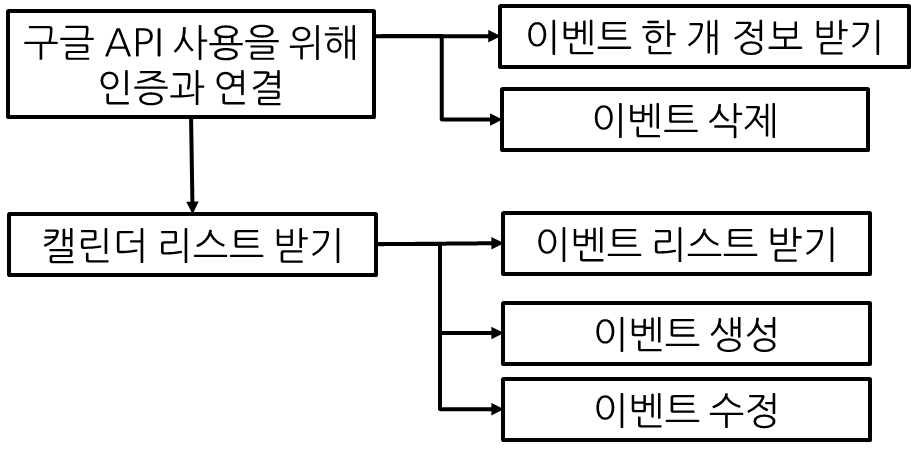
* 캘린더 공개 설정 체크

전체 클래스 구조

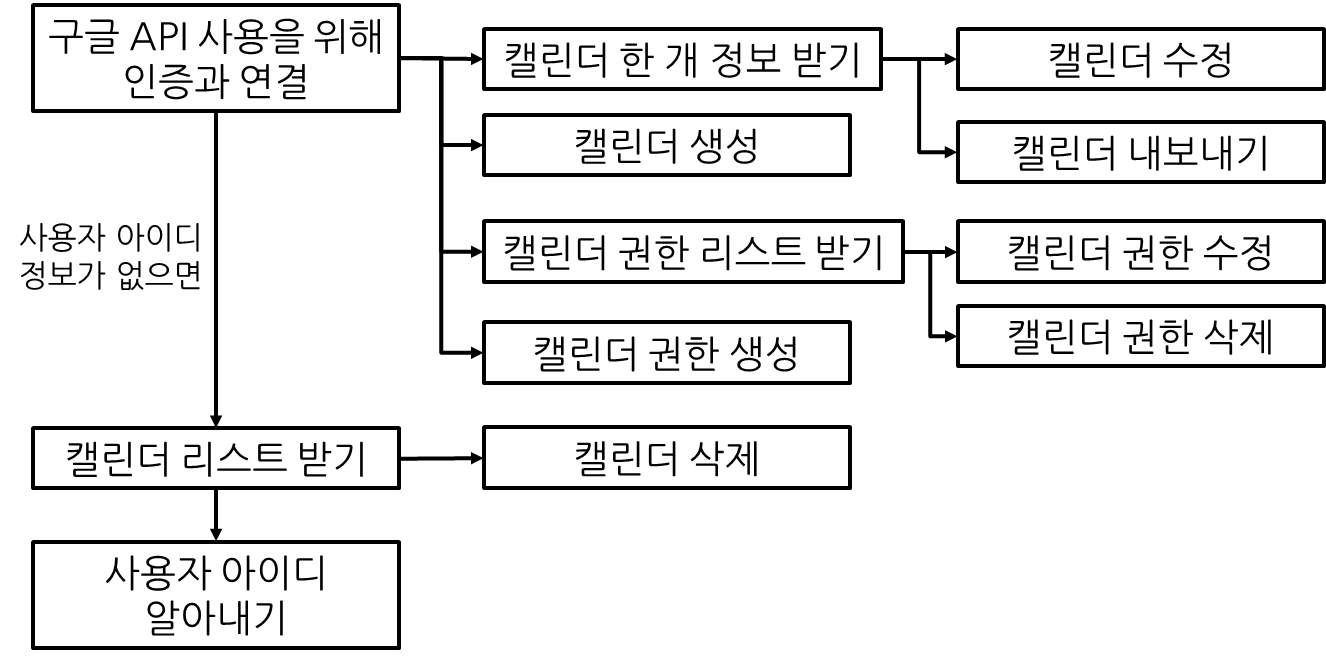


* 전체적인 동작은 View에서 정보를 요청하거나 생성, 수정, 삭제를 요청하면 그에 맞는 Controller에서 GoogleCalendarService를 통해 서비스(인증과 연결 정보가 있는 Calendar 객체)를 받아 API를 사용하여 처리한 후 응답해 줌.
* View의 jsp 파일 처리는 HomeController에서 해 줌.
* 이벤트 리스트를 요청 받으면 이벤트를 그리기 위한 편의성을 위해 정해 놓은 우선 순위에 맞게 정렬하여 이벤트를 보냄.
  + 이벤트 리스트에 반복 일정이 있으면 반복 일정을 계산해 반복 일정 리스트를 얻어낸 후 이벤트 리스트에 추가하고 정렬함.

기능 동작 구조



* 여러 캘린더들에 대해 이벤트를 받거나 생성, 수정이 가능하기 때문에 캘린더 리스트를 받고 이벤트 리스트를 받거나 생성, 수정할 수 있음.
* 이벤트 한 개 정보를 받거나 이벤트를 삭제할 때도 캘린더의 정보(캘린더 아이디)가 필요로 하지만 이벤트 한 개에 대한 캘린더 정보는 바뀌지 않기 때문에 태그에 속성 값으로 저장하여 사용함.

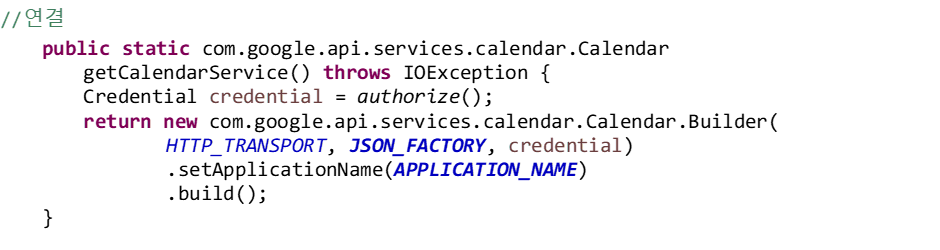


* 캘린더 권한 수정, 삭제, 캘린더 내보내기 기능을 수행하기 위해서는 사용자의 아이디를 알아야 함.
  + 사용자 아이디가 가지고 있는 캘린더 권한은 변경 불가
  + 캘린더 내보내기 시 ics 파일을 작성할 때 사용자의 아이디 정보가 필요
* 캘린더 한 개에 대한 정보를 받아서 캘린더를 수정하고 내보내기 기능 수행.
* 캘린더 권한 리스트를 받아 캘린더 권한 수정, 삭제 기능 수행
* 캘린더 삭제는 캘린더 리스트를 받아온 후에만 가능. 캘린더 리스트 체크 박스 부분에 삭제 기능을 추가했기 때문

구글 API 사용을 위한 인증



* HttpTransport는 GET, POST, PUT, DELETE만 지원해 줌.
* 프로젝트에 저장된 client\_secret.json 파일을 통해 client secret 값을 얻음.
* SCOPES = 캘린더의 읽기/쓰기 권환
* Access type = offline으로 해서 access token이 만료 되어 사용하지 못하게 되면 리소스 서버에 refresh token으로 다시 요청해 access token을 받을 수 있음.
* LocalServerReceiver에서 redirectUri 생성. Host(localhost), port는 -1로 해서 사용하지 않은 port 선정 verification code receiver
* authorize(“user”) 이 부분에서 브라우저를 열어서 승인을 받고 code를 받아서 access token을 받음. 그리고 credential 저장.

JsonFactory -> JSON 파싱 기능 제공.(input stream에서 파싱)

* builder안에 캘린더 정보 요청할 때 사용하는 url 가지고 있음.
* 기본 url: https://www.googleapis.com/calendar/v3
* 만약 캘린더 리스트에 대한 요청을 했다면 위의 url 뒤에
  + - /users/me/calendarList 붙어서 요청.

Feature List

|  |  |
| --- | --- |
| feature | method |
| 캘린더 리스트 받기 | calendarList().list() |
| 이벤트 리스트 받기 | events().list(calendarId)  (.setTimeMin(DateTime).setTimeMax(DateTime)) |
| 이벤트 생성 | events().insert(calendarId, Event) |
| 이벤트 수정 | events().update(calendarId, eventId, Event) |
| 이벤트 삭제 | events().delete(calendarId, eventId) |
| 이벤트 캘린더 옮기기 | events().move(originalCalendarId, eventId, newCalendarId) |
| 캘린더 생성 | calendars().insert(Calendar) |
| 캘린더 수정 | calendars().update(calendarId, Calendar) |
| 캘린더 삭제 | calendars().delete(calendarId) |
| 캘린더 정보 받기 | calendarList().get(calendarId) |
| 캘린더 정보 수정 | calendarList().update(calendarId, calendarListEntry) |
| 캘린더의 권한 리스트 받기 | Acl().list(calendarId) |
| 캘린더의 권한 생성 | Acl().insert(calendarId,rule) |
| 캘린더의 권한 수정 | Acl().update(calendarId, ruleId, rule) |
| 캘린더의 권한 삭제 | Acl().delete(calendarId, ruleId) |

캘린더 리스트 받기

CalendarList calendarList = service.calendarList().list().setPageToken(pageToken).execute();

* 위의 코드로 리스트를 얻어 옴.
* CalendarDTO에 맞게 데이터 전처리
* DTO를 사용하는 이유는 전송하는 데이터의 사이즈를 작게 해서 전송 시간을 줄이기 위함
  + API에 요청하고 받아오는 시간이 최소 500ms 걸림
* CalendarDTO

|  |  |  |  |
| --- | --- | --- | --- |
| 자료명 | 자료형 | 설명 | 값 |
| id | String | 사용자의 캘린더 아이디 | CalendarListEntry.getId() |
| summary | String | 사용자의 캘린더 제목 | CalendarListEntry.getSummary() |
| check | boolean | 사용자가 캘린더의 체크박스 체크 여부 | Calendar의 id를 name으로 갖는 세션의 value  (초기는 true) |
| colorId | String | 사용자의 캘린더 색 아이디 | CalendarListEntry.getColorId() |
| primary | Boolean | 캘린더가 사용자의 메인  캘린더인지에 대한 여부 | calendarListEntry.getPrimary() |
| accessRole | String | 캘린더의 Access Role(일정에 대한 접근 권한 정의) | calendarListEntry.getAccessRole() |
| defaultReminders | List<EventReminder> | 캘린더의 알림 기본 값 | caelndarListEntry.getDefaultReminders() |

* Access Role = freeBusyReader < reader < writer < owner
* writer, owner만이 일정을 쓸 수 있는 권한이 있음.
* 캘린더마다 기본 알림 값을 설정할 수 있음.
  + 일정이 캘린더의 기본 알림을 사용했으면 이 알림 값을 보여주면 됨.
  + Google Calendar에서 시간이 있는 일정에 대한 기본 알림 값만 API로 알 수 있음.
  + Google Calendar에서는 종일 일정에 대한 기본 알림 값 설정도 제공해주지만 API를 통해서는 알 수 없음.