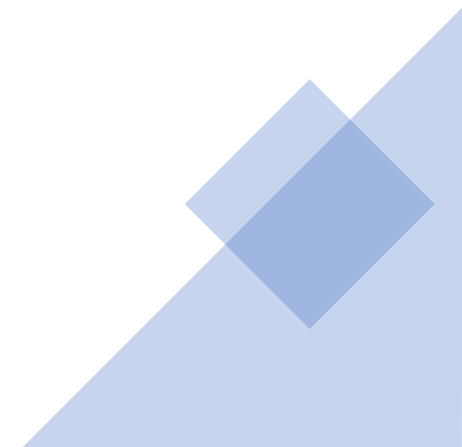


# 4월 4일 발표자료 - iOS

1871069 김진웅



## 목차

- 1. 한 주간의 진행 상황
    - 부딪힌 이슈 & 해결사항
    - 현재 캘린더 UI 진행상황
  - 2. 고민하고 있는 issue
    - 컬렉션 셀을 이용한 캘린더의 일정을 삽입하기
  - 3. 앞으로의 계획
- 

# 1. 한 주간의 진행상황

- 부딪힌 이슈 & 해결사항
  - 기존의 개발을 iPhone 11 Plus에서 진행을 해왔으나,
  - Xcode 업데이트로 인해, 강제로 14 pro로 개발 도중
  - 회원가입 Ui가 정상적으로 표현되지 않는 문제를 발견하여
  - Ui의 전체적인 레이아웃을 재수정

10:21

← 회원가입

이메일 \*

예) abc@hansung.ac.kr

비밀번호 \*

영문, 숫자 조합 8~16자

비밀번호 확인 \*

비밀번호를 한번 더 입력해주세요.

생년월일

생년월일을 골라주세요.

이름 \*

예) 홍길동

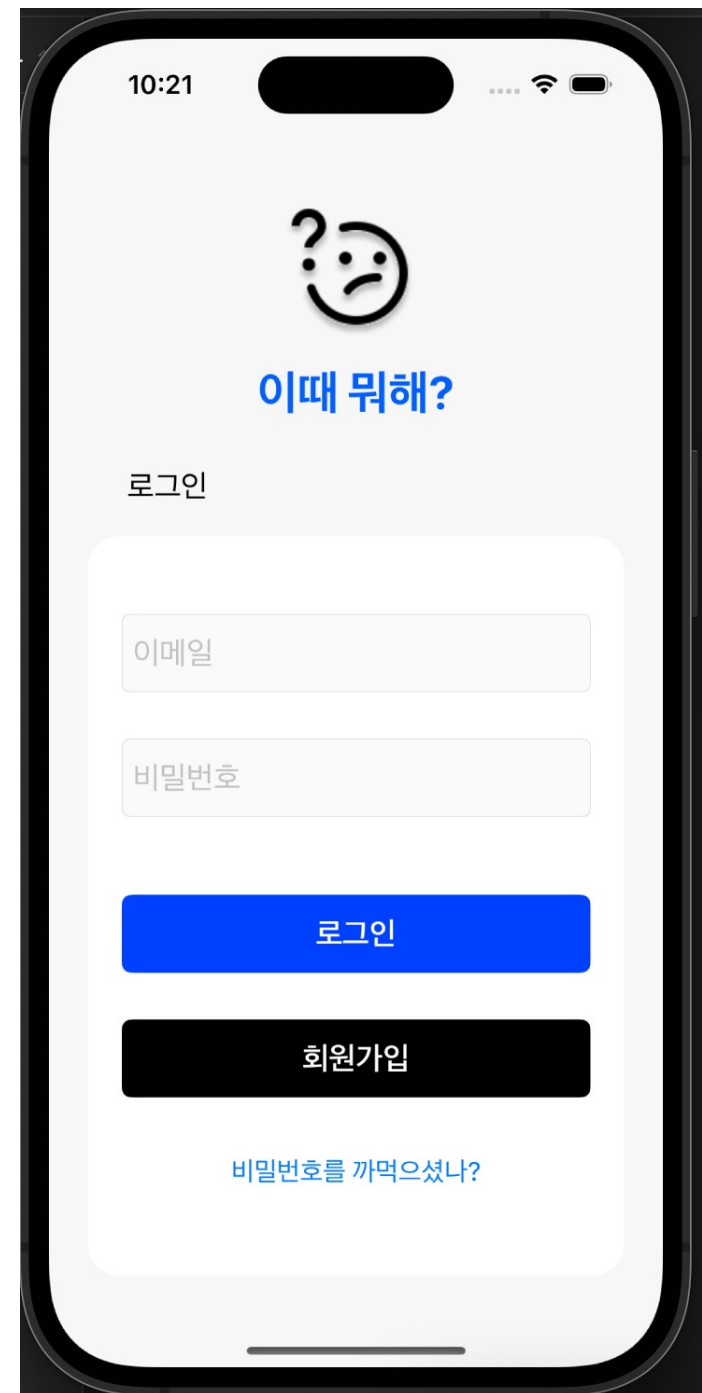
휴대폰 번호 인증 \*

인증하기

# 1. 한 주간의 진행상황

- 로그인 페이지 UI 재구성

- WEB의 로그인 페이지와 동일하게
- 전체적인 레이아웃을 재구성하였음.



# 1. 한 주간의 진행상황

- 컬렉션 셀을 이용한 캘린더 구현하기

- 아직 오늘 날짜에 대한 백그라운드 이미지 삽입과 관련한 로직을 해결하지 못하였음.

- FSCalendar를 이용한 캘린더 구현하기

- 현재 사이드 프로젝트에서 FSCalendar Delegate를 통해 커스텀 과정을 거치고 있으나, 미완성...

금주 예정 작업으로 했어야 하나, 지난 주말부터 몸상태가 좋지 않아 완료하지 못했습니다. 빠른 시일 내에 정상적으로 진행하겠습니다..

## 2. 고민하고 있는 이슈

- 컬렉션 셀을 이용한 캘린더에서 각 셀에 아이템을 어떻게 집어 넣어야 할까?
  - 현재 Horizontal Stack View로 각 버튼을 일정의 제목으로 삽입해야 할지 등에 대한 방식을 고민하고 있음.

### 3. 앞으로의 계획

3주차 (3/13~3/19)	세부UI 확정, 로그인 및 회원가입 UI 구현
4주차 (3/20~3/26)	캘린더 UI 구현 - 1
5주차 (3/27~4/2)	캘린더 UI 구현 - 2
6주차 (4/3~4/9)	캘린더 UI 구현 - 3
7주차 (4/10~4/16)	일정 생성 및 수정 UI 구현
8주차 (4/17~4/23)	그룹 일정 UI 구현 - 1
9주차 (4/24~4/30)	그룹 일정 UI 구현 - 2
10주차 (5/1~5/7)	서버 통신 연결 및 데이터 교환 확인 절차 - 1
11주차 (5/8~5/14)	서버 통신 연결 및 데이터 교환 확인 절차 - 2
12주차 (5/15~5/21)	테스팅 및 보완
13주차 (5/22~5/28)	최종 테스트

- FSCalendar 커스텀 하기
- 컬렉션 셀에서 해결할 수 있는 문제점들 해결하기

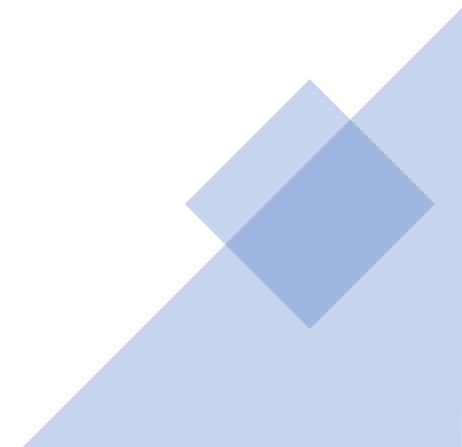
# 4월 4일 발표자료 - WEB

1871062 김정한





# 목차

- 1. 한 주간의 진행 상황
    - 1-1. 현재 진행 상황
    - 1-2. 부딪힌 이슈 & 해결사항
  - 2. 고민하고 있는 이슈들
  - 3. 앞으로의 계획
- 

# 1. 현재 진행 상황

- 1-1. 어떠한 것을 공부했는가
  - 캘린더 페이지 개발
    - React-big-calendar 라이브러리를 사용해 캘린더 페이지 개발

Today	Back	Next	4월 2023				Month	Week	Day	Agenda
일요일	월요일	화요일	수요일	목요일	금요일	토요일				
26	27	28	29	30	31	01				
02	03	04	05	06	07	08				
		ppt완성 Conference	아르바이트							
09	10	11	12	13	14	15				
16	17	18	19	20	21	22				
23	24	25	26	27	28	29				
30	01	02	03	04	05	06				

React-big-calendar Library

# 1. 현재 진행 상황

- 1-2. 부딪힌 이슈 & 해결사항
  - React-big-calendar 언어설정을 한글로 바꾸기
    - Locale element를 따로 만들지 않고 한글코드를 직접 import해서 가져옴.

Today	Back	Next	April 2023				Month	Week	Day	Agenda
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday				
26	27	28	29	30	31	01				
02	03	04	05	06	07	08				
09	10	11	12	13	14	15				
16	17	18	19	20	21	22				
23	24	25	26	27	28	29				
30	01	02	03	04	05	06				

이전 캘린더(영어로 되어있음)

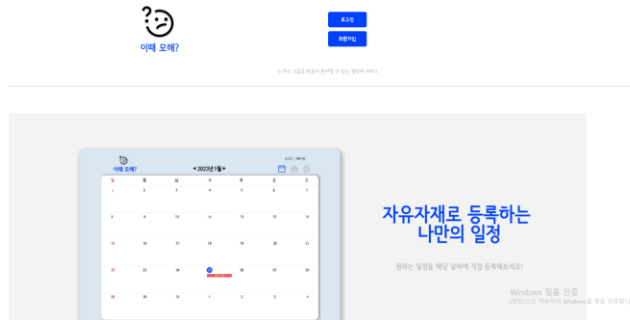
Today	Back	Next	4월 2023				Month	Week	Day	Agenda
일요일	월요일	화요일	수요일	목요일	금요일	토요일				
26	27	28	29	30	31	01				
02	03	04	05	06	07	08				
09	10	11	12	13	14	15				
16	17	18	19	20	21	22				
23	24	25	26	27	28	29				
30	01	02	03	04	05	06				

수정 캘린더(한글로 바뀜)

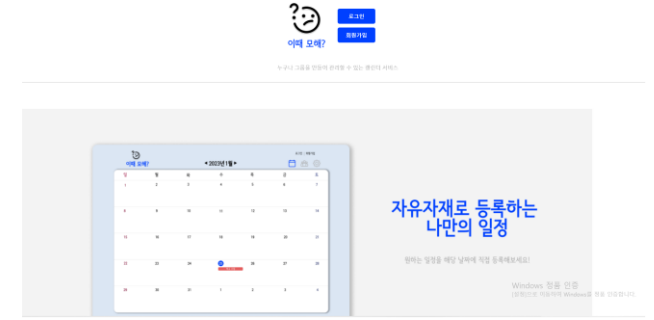
# 1. 현재 진행 상황

## • 1-2. 부딪힌 이슈 & 해결사항

- 이때 모해 아이콘이 중앙에 고정되어있지 않는 문제 해결
- Position: relative; 속성을 새로 학습하여 적용. 아이콘이 항상 중앙에 위치하도록 고정 완료



-이전 로그인 화면-



-변경된 로그인 화면-

```
position: relative;
left: 95%;
transform: translateX(-95%);
```

# 1. 현재 진행 상황

- 1-2. 부딪힌 이슈 & 해결사항
  - React-big-calendar 언어설정을 한글로 바꾸기
    - Locales element를 따로 만들지 않고 한글코드를 직접 import해서 가져옴.

```
const locales = {  
  "ko": require("date-fns/locale/ko")  
}  
  
const localizer = dateFnsLocalizer({  
  format,  
  parse,  
  startOfWeek,  
  getDay,  
  locales  
})
```

-수정 전 오류 코드-

```
import koLocale from 'date-fns/locale/ko';  
  
const localizer = dateFnsLocalizer({  
  format,  
  parse,  
  startOfWeek,  
  getDay,  
  locales: {  
    'ko' : koLocale  
  }  
})
```

-수정 후 완성 코드-

## 2. 고민하고 있는 이슈들

- React-big-calendar 를 커스텀하는 방법에 대하여 더욱 더 깊은 연구 필요
- Redux를 어떻게 사용해야 편리하게 React의 상태를관리할 수 있는지에 대한 연구 필요

### 3. 앞으로의 계획

- React-big-calendar custom 하기
- Redux 로직 이해하기

3주차 (3/12 ~ 3/18)	시작 메인 페이지 만들기, 로그인 form과 fetch 공부
4주차 (3/19 ~ 3/25)	UI 구성 완료, 시작 메인 페이지 개발 시작
5주차 (3/26 ~ 4/1)	로그인 폼, 캘린더 UI 구현 시작
6주차 (4/2 ~ 4/8)	캘린더 UI 구현
7주차 (4/9 ~ 4/15)	그룹 UI 구현
8주차 (4/16 ~ 4/22)	그룹 UI 구현
9주차 (4/23 ~ 4/29)	그룹 UI 구현
10주차 (4/30 ~ 5/6)	서버와의 연결 관리
11주차 (5/7 ~ 5/13)	서버와의 연결 관리
12주차 (5/14 ~ 5/20)	테스팅
13주차 ( 5/21 ~ 5/27)	테스팅

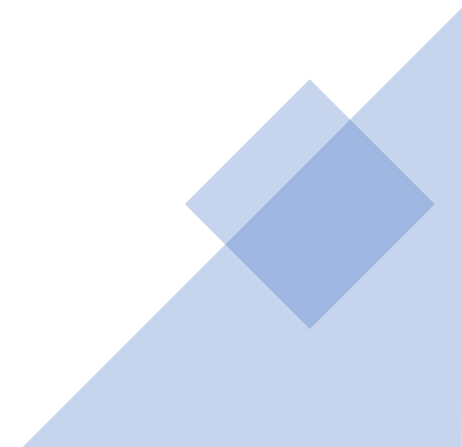
# 4월 4일 발표자료 - Server

1811072 유영재  
1871197 이윤재





# 목차

- 1. 한 주간의 진행 상황
    - 1-1. 어떠한 것을 공부했는가
    - 1-2. 부딪힌 이슈 & 해결사항
  - 2. 고민하고 있는 issue
  - 3. 앞으로의 계획
  - 4. 기타
- 

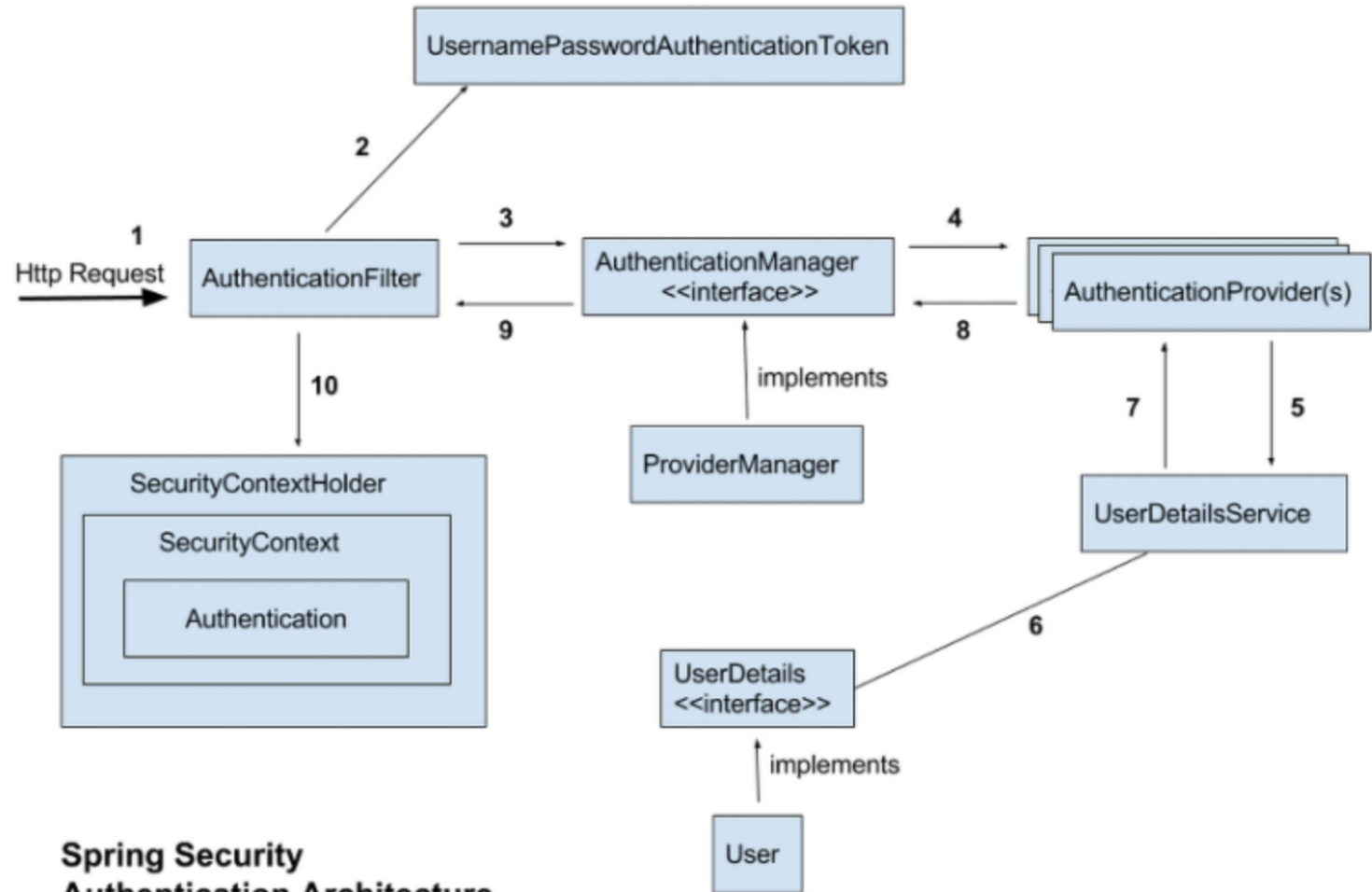
# 1. 한 주간의 진행상황

- 1-1. 어떠한 것을 공부했는가
  - Schedule CRUD 구현 시작
  - Schedule Create 동작 과정



# 1. 한 주간의 진행상황

- 1-1. 어떠한 것을 공부했는가
  - Spring Security
  - AuthenticationFilter를 통해 AuthenticationManager, AuthenticationProvider에게 JWT를 전달하고 UserDetailsService를 통해 실질적인 인증 로직을 수행함.
  - 인증 수행이 완료되면 SecurityContextHolder에 Authentication 정보를 담아 로그인 세션 정보를 유지하도록 함.



Spring Security  
Authentication Architecture

Chathuranga Tennakoon  
[www.springbootdev.com](http://www.springbootdev.com)



# 1. 한 주간의 진행상황

- 1-1. 어떠한 것을 공부했는가
  - 이메일 인증
    - JavaMailSender 라이브러리 이용
    - 인증메일 발송용 계정 생성(Gmail)

```
@Configuration
public class MailConfig {

    no usages  terry8408@gmail.com <96044823+terry8408@users.noreply.github.com>
    @Bean
    public JavaMailSender javaMailSender() {
        JavaMailSenderImpl mailSender = new JavaMailSenderImpl();
        mailSender.setHost("smtp.gmail.com"); // SMTP 서버 주소
        mailSender.setPort(587); // SMTP 포트
        mailSender.setUsername("hansung.wassup@gmail.com"); // SMTP 계정 이메일
        mailSender.setPassword("inhiorrhwsdpdlotc"); // SMTP 계정 비밀번호

        Properties props = mailSender.getJavaMailProperties();
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.starttls.enable", "true");

        return mailSender;
    }
}
```

테스트 메일 전송 성공

test ➤



hansung.wassup@gmail.com

나에게 ▼

메일 테스트 입니다

↩ 답장

➡ 전달

# 1-2. 부딪힌 이슈 & 해결사항

- 1-2. 부딪힌 이슈

- Spring Security Authentication Architecture에 따라서 기존의 인증 방식을 보완하는 과정으로 개발을 진행 중인데, 현재 프로젝트와 Spring Security 아키텍처가 패턴이 많이 달라서 개발에 난항을 겪음.
- UserDetails라는 새로운 객체 타입을 적용하는데 어려움을 겪음.
- 해결하지 못한 사항으로, 전체적인 인증 과정을 다시 돌아보며 해결할 예정.
- Gmail 계정 해킹...

## 2. 고민하고 있는 이슈들

### 2-1. Authentication 방법

- 테이블마다 `origin_key` 필드를 기본키로 지정해 테이블 검색 시 식별하는 용도로 사용.
- 하지만 이런 방식을 채택한다면, 클라이언트의 로그인 정보 (세션)을 유지하기 위해 클라이언트로부터 받은 JWT를 해석하고, `originKey`를 DB에서 찾는 불필요한 DB접근이 많이 이루어지게 됨.
- 스프링 시큐리티에서는 `userId`를 `@AuthenticationPrincipal`로 사용자의 로그인 정보를 유지하는데, 이를 사용하면 불필요한 DB접근을 줄이고 효율을 높일 수 있을거라 생각됨.

## 2. 고민하고 있는 이슈들

```
@PostMapping
public ResponseEntity<?> createSchedule(@AuthenticationPrincipal String userId, @RequestBody ScheduleDTO dto) {
    try {
        ScheduleEntity entity = ScheduleDTO.toEntity(dto);

        // dto에서 token 가져온 후에
        String token = dto.getToken();

        // tokenProvider로 토큰 해석해서 userId 알아냄
        String tokenizedUserId = tokenProvider.validateAndGetUserId(token);

        entity.setUserId(userId);

        List<ScheduleEntity> entities = service.create(entity);

        List<ScheduleDTO> dtos = entities.stream().map(ScheduleDTO::new).collect(Collectors.toList());

        ResponseDTO<ScheduleDTO> response = ResponseDTO.<ScheduleDTO>builder().data(dtos).build();

        return ResponseEntity.ok().body(response);
    } catch (Exception e) {
        String error = e.getMessage();
        ResponseDTO<ScheduleDTO> response = ResponseDTO.<ScheduleDTO>builder().error(error).build();
        return ResponseEntity.badRequest().body(response);
    }
}
```

Spring Security에서 지원하는 `@AuthenticationPrincipal` Annotation을 사용해 HTTP 통신 상에서 사용자의 로그인 정보를 주고 받지 않고도 로그인 세션을 유지하면서 그 정보를 빼낼 수도 있음.

## 2. 고민하고 있는 이슈들

### 2-2. 이메일 인증

- Gmail 해킹 방지대책

- 아니면 아예 다른 인증방법?



## 3. 앞으로의 계획

- 3-1. 계획
  - Spring Security 개발
  - Authenticate 방식 확정
  - Schedule CRUD