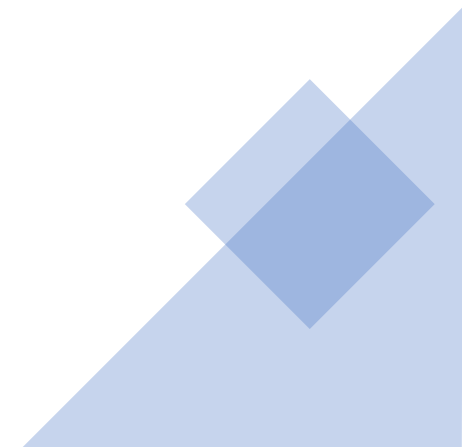


# 3월 21일 발표자료 - iOS

1871069 김진웅



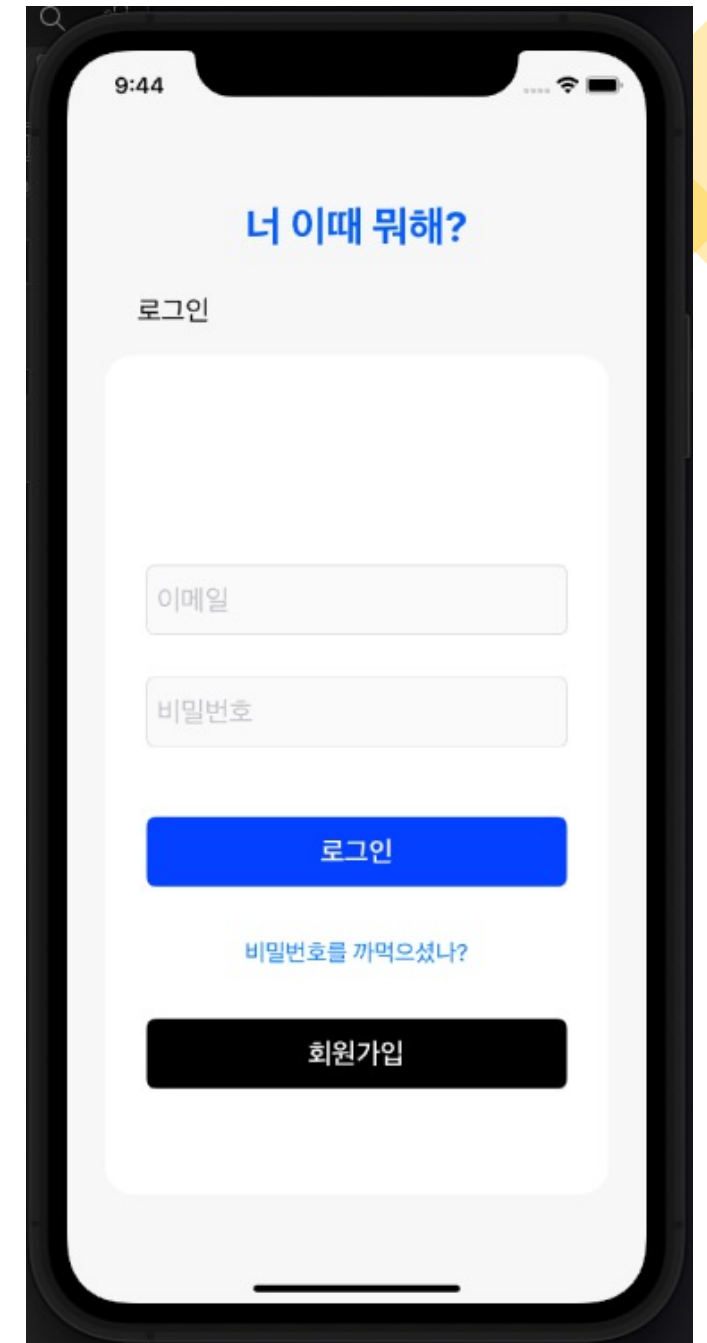
# 목차

- 1. 한 주간의 진행 상황
    - 로그인 및 회원가입 창 개발
    - 개발 과정에서 부딪힌 이슈 및 해결사항
  - 2. 고민하고 있는 issue
    - 휴대폰 인증 방식
    - 비밀번호 찾기
  - 3. 앞으로의 계획
- 

# 1. 한 주간의 진행상황

## • 로그인 화면 개발

- 기존 Figma 설계에 비해 상단 제목 글씨가 작아 크기를 각 20->30 / 15->20으로 키웠음
- 이메일 위에 남는 빈 여백에 Figma 설계 상으로 메인 아이콘 자리를 두었으나, 아이콘의 모형 등 정해진 바가 없어 우선적으로 빈 공간으로 두었음.
- 에뮬레이터 사용 기기는 iPhone 11을 사용
- Auto-layout 기능을 활용하여 다양한 기기에서도 해당 UI가 원활히 보여짐.



# 1. 한 주간의 진행상황

- Auto-layout이 무엇인가?

- iOS 등 사용되는 레이아웃 엔진으로써, 앱 내의 모든 요소를 배치하기 위한 규칙을 만들어, 뷰 계층 구조 내에서 상대적인 위치를 지정함.
- 뷰의 크기나 화면 크기가 바뀌더라도 뷰 간의 관계를 유지할 수 있게 도와줌.
- 제약조건 (constraints)를 사용하여 뷰의 크기와 위치를 제어하는데, 각 뷰는 슈퍼 뷰와의 상대적인 위치나 크기에 대한 제약 조건을 가질 수 있으며, 시스템은 이를 해석하여 뷰를 올바른 위치와 크기로 재배치함.
- 화면 크기와 비율을 자동으로 처리하기 때문에 개발자로서 앱을 재설계할 필요 X

# 1. 한 주간의 진행상황

## • 회원가입 화면 개발

- 페이지를 띄우는 방식은 모달을 통해 띄우도록 하였음.
- 이메일, 비밀번호, 확인, 생년월일, 이름, 휴대폰 본인 인증의 내용을 가짐.
- 생년월일의 경우 DatePicker를 사용하여 사용자가 고를 수 있게끔 방식을 변경할 예정
- 모달을 닫는 방식 (<- 버튼, 모달 내리기, 완료)

9:53

←  
회원가입

이메일 \*  
예) abc@naver.com

비밀번호 \*  
영문, 숫자 조합 8~16자

비밀번호 확인 \*  
비밀번호를 한번 더 입력해 주세요.

생년월일  
생년월일을 골라주세요.

이름 \*  
예) 홍길동

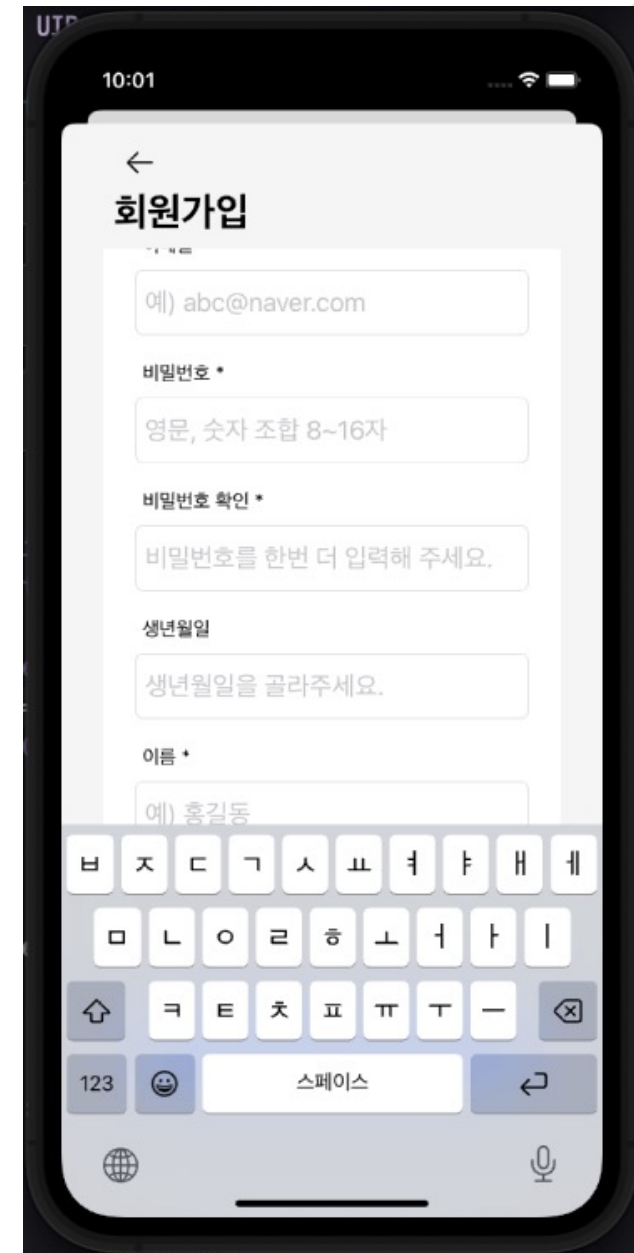
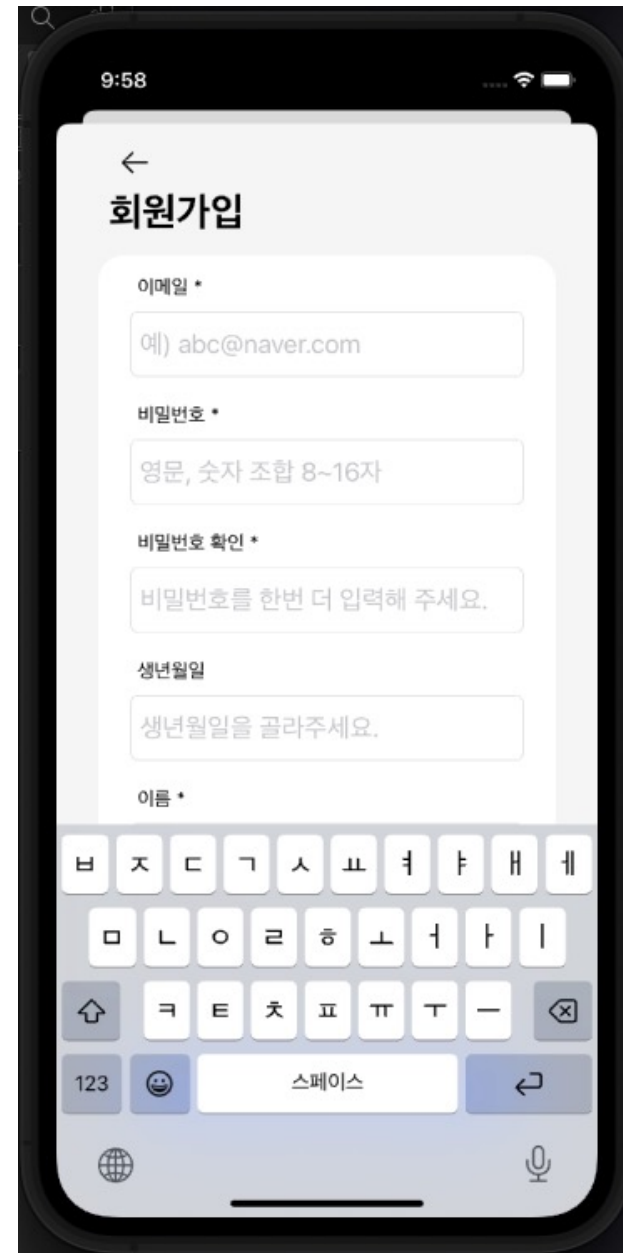
휴대폰 본인 인증 \*  
인증하기

완료

# 1. 한 주간의 진행상황

## • 부딪힌 이슈 & 해결 사항

- 회원가입 화면에서 이름 TextField 선택 시, 키보드에 의해 가려짐. -> 사용자가 입력한 내용을 눈으로 확인할 수 없게 됨.
- 해결과정으로, iOS의 ScrollView를 활용하여 최소한의 영역 내에서 스크롤이 가능하게끔 하였음.
- View > ScrollView > View > 각 컴포넌트



## 2. 고민하고 있는 이슈들

- 휴대폰 본인 인증에 대한 방식

- 휴대폰 번호를 인증하기 위해서는 문자를 전송하는 등에 방법 구현이 필요한데 기타 API를 활용해야 하는 것인지,,
- 예를 들어 PASS 앱을 활용한다든지 또는 독자적인 문자 전송 및 인증을 구현해야 하는 것인지,,

- 비밀번호를 사용자가 잊어버렸을 때

- 비밀번호를 사용자가 잊어버렸을 때, 어떻게 방법을 구현해 나가야 하는 것인지에 대한 고민
- 비밀번호 까먹음 -> 비밀번호 찾기를 위한 email 입력 -> email을 DB와 비교하여 사용자를 찾고 -> 바로 새 비밀번호 입력 또는 email로 인증 코드 전송 등
- 구체적인 방법이 필요하나, 이러한 과정까지 거치게 되면 복잡성이 증가될 것으로 생각됨.

### 3. 앞으로의 계획

3주차 (3/12~3/18)	세부UI 확정, 로그인 창 및 회원가입 창 만들기
4주차 (3/19~3/25)	캘린더 UI 구현하기 - 1
5주차 (3/26~4/1)	캘린더 UI 구현하기 - 2
6주차 (4/2~4/9)	캘린더 UI 구현하기 - 3
7주차 (4/9~4/15)	일정 생성 및 수정 UI 구현
8주차 (4/16~4/22)	그룹 일정 UI 구현 - 1
9주차 (4/23~4/29)	그룹 일정 UI 구현 - 2
10주차 (4/30~5/6)	서버와의 통신 연결 확인 및 데이터 교환 간 오류 확인
11주차 (5/7~5/13)	서버와의 통신 연결 확인 및 데이터 교환 간 오류 확인
12주차 (5/14~5/20)	테스팅
13주차 (5/21~5/27)	테스팅

- 회원가입에서 생년월일을 DatePicker로,,
- 캘린더 UI를 구현하기 위한 Collection Cell 구성 및 꾸미기

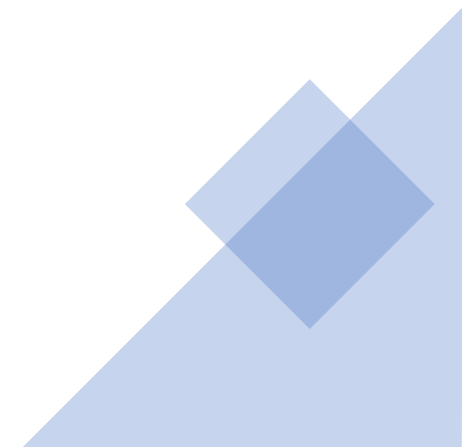


# 3월 21일 발표자료 - WEB

1871062 김정한

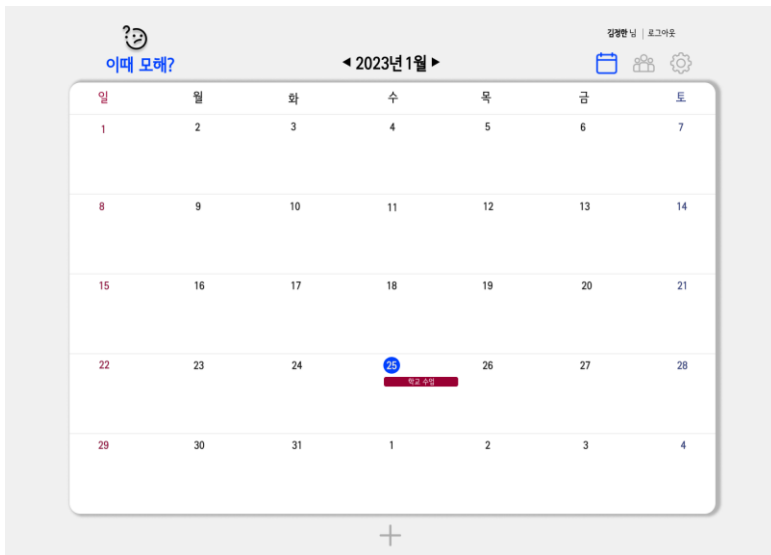


# 목차

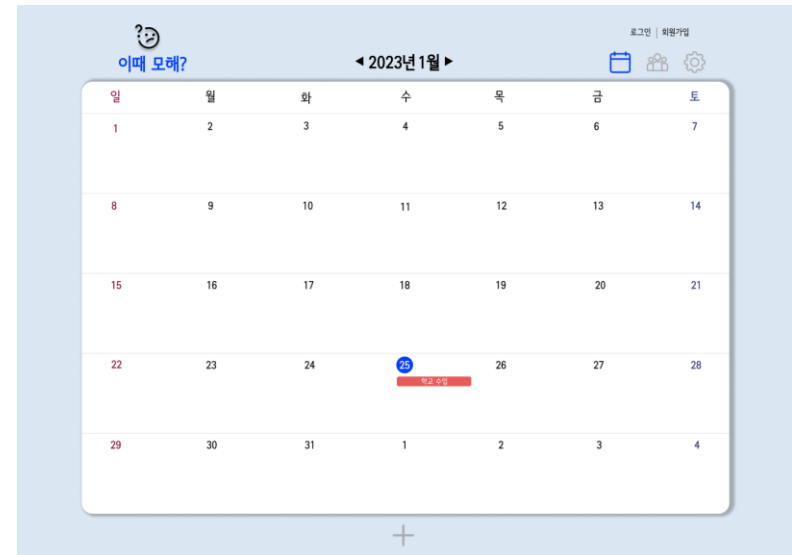
- 1. 한 주간의 진행 상황
    - 1-1. 어떠한 것을 공부했는가
    - 1-2. 부딪힌 이슈 & 해결사항
  - 2. 고민하고 있는 이슈들
  - 3. 앞으로의 계획
- 

# 1. 한 주간의 진행상황

- 1-1. 어떠한 것을 공부했는가
  - 캘린더 배경화면을 회색에서 흐린 하늘색으로 변경 완료



-이전 배경화면-



-변경된 배경화면-

# 1. 한 주간의 진행상황

- 1-1. 어떠한 것을 공부했는가
  - 시작 페이지 피그마로 디자인 완료



-시작 페이지-

# 1. 한 주간의 진행상황

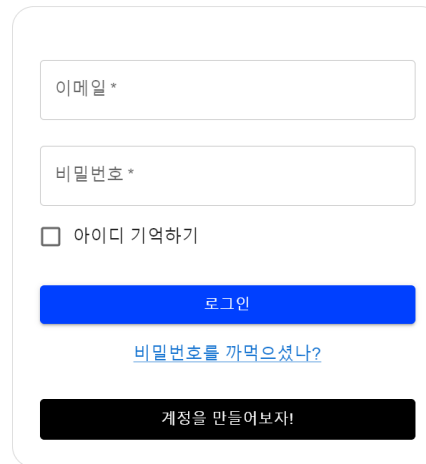
- 1-1. 어떠한 것을 공부했는가
  - 리액트 개발환경 세팅 및 로그인 화면 개발



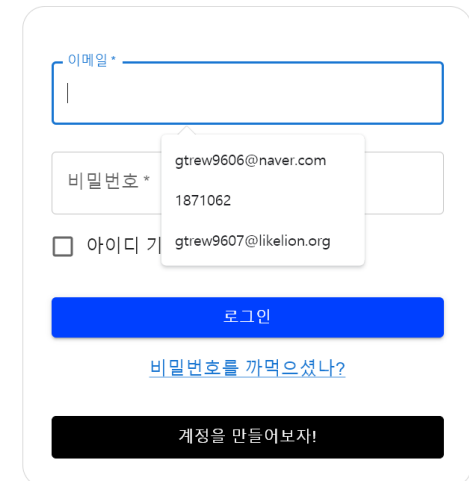
MUI 라이브러리를 활용하여 더욱 더 편리한 UI 개발



이때 모해?

A mockup of a login form. It features two input fields: '이메일 \*' (Email) and '비밀번호 \*' (Password). Below the password field is a checkbox labeled '아이디 기억하기' (Remember ID). A blue button labeled '로그인' (Login) is positioned below the checkbox. Below the button is a link that says '비밀번호를 까먹으셨나?' (Forgot your password?). At the bottom is a black button with white text that says '계정을 만들어보자!' (Let's create an account!).

이때 모해?

A mockup of a login form, similar to the one on the left but with an additional feature. It has '이메일 \*' and '비밀번호 \*' input fields. A dropdown menu is open below the password field, showing two options: 'gtrew9606@naver.com' and '1871062'. Below the dropdown is a checkbox labeled '아이디 기억하기'. A blue '로그인' button is below the checkbox, followed by a link '비밀번호를 까먹으셨나?'. At the bottom is a black button with white text '계정을 만들어보자!'.

-로그인 화면-

# 1. 한 주간의 진행상황

- 1-2. 부딪힌 이슈 & 해결사항
  - Npm start 하는 과정에서  
문제는 없지만 화면에 아무것도  
나타나지 않는 오류 발생.
  - 해결사항 : `component = {Home}`이  
아닌 `element = {<Home />}`의  
방식으로 코드를 수정하였다.

컴포넌트를 생성하는 함수가 아닌 JSX 엘리먼트를 사용하여 **Route**에 전달하면 **element** prop을 사용하고, 컴포넌트를 생성하는 함수를 전달하려면 **component** prop을 사용해야 합니다.

```
1 import React from 'react';
2 import {BrowserRouter, Routes, Route } from 'react-router-dom';
3 import Home from './pages/Home';
4 import Signin from './pages/Signin';
5 import NotFound from './pages/NotFound';
6
7 function App() {
8   return (
9     <BrowserRouter>
10     <Routes>
11       <Route path="/" component={Home} />
12
13       <Route path="/signin" component={Signin} />
14       <Route component = {NotFound} />
15     </Routes>
16   </BrowserRouter>
17 )
18 }
19
20 export default App;
21
```

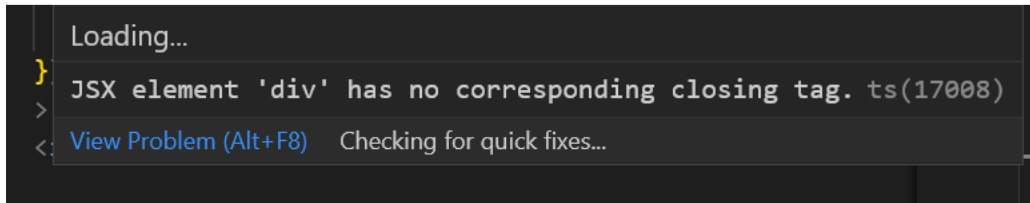
이전 코드

```
function App() {
  return (
    <BrowserRouter>
    <Routes>
      <Route exact path="/" element= {<Home />} />
      <Route path="/signin" element= {<Signin />} />
      <Route element= {<NotFound />} />
    </Routes>
  </BrowserRouter>
)
}
```

수정한 코드

# 1. 한 주간의 진행상황

- 1-2. 부딪힌 이슈 & 해결사항
  - Div 태그가 닫히지 않았다는 오류 재발생 (실행은 잘 됨.)

A screenshot of a VS Code error message. The error message is displayed in a dark-themed window. It starts with "Loading..." in a light gray font. Below it, a yellow bracket icon is on the left, followed by the text "JSX element 'div' has no corresponding closing tag. ts(17008)" in a light gray font. At the bottom, there is a blue link "View Problem (Alt+F8)" and the text "Checking for quick fixes..." in a light gray font.

```
Loading...
} JSX element 'div' has no corresponding closing tag. ts(17008)
>
< View Problem (Alt+F8) Checking for quick fixes...
```

## 2. 고민하고 있는 이슈들

- MUI라이브러리와 CSS와의 조화에 대한 고민 - MUI라이브러리에 설정되어 있는 기본 디자인을 입맛에 맞게 수정하는 방법에 대한 공부 필요
- 회원가입 시 user들의 정보들 어떠한 방식으로 서버에 전달해야 할지에 대한 고민 - 여러 자료들과 패스트캠퍼스를 이용한 공부 필요
- 휴대폰 인증 시 어떠한 방식으로 문자를 전송되게 할 것인지에 대한 고민 - 다른 API 사용시 복잡해질것으로 예상됨



### 3. 앞으로의 계획

- 회원가입 페이지 개발
- 초기 페이지 개발

# 3월 21일 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
  - 2-1. 인증 관련 문제
- 3. 앞으로의 계획

# 1. 한 주간의 진행상황

- 1-1. 어떠한 것을 공부했는가
  - 클라이언트-서버 간의 프로토콜
  - 회원가입, 로그인, 일정 생성 등 사용자가 작업을 수행할 때 클라이언트와 서버 간에 주고받아야하는 데이터의 형태를 미리 정의



# 1. 한 주간의 진행상황

```
// 로그인 요청 c->s
User {
  "id" : "cross_man@naver.com"
  "password" : "abc123"
}
```

```
// id,pw가 맞으면 로그인 성공, 토큰을 붙여서 다시 넘기기 s->c
User {
  "id" : "cross_man@naver.com"
  "password" : null
  "token" : "aooalskfnaalkfnkjbfaklksdpoalksfnaalksnf"
}
```

```
// 일정 생성 요청 c->s
Schedule {
  "name" : "밥먹기"
  "start" : "2023-03-17-19:11"
  "end" : "2023-03-17-19:11"
  "memo" : "오늘 저녁은 고기!"
  "notification" : "true"
  "all_day_toogle" : "true"
  "token" : "aooalskfnaalkfnkjbfaklksdpoalksfnaalksnf"
  "created_at" : "2023-03-17-19:11"
  "last_modified_at" : "2023-03-17-19:11"
}
```

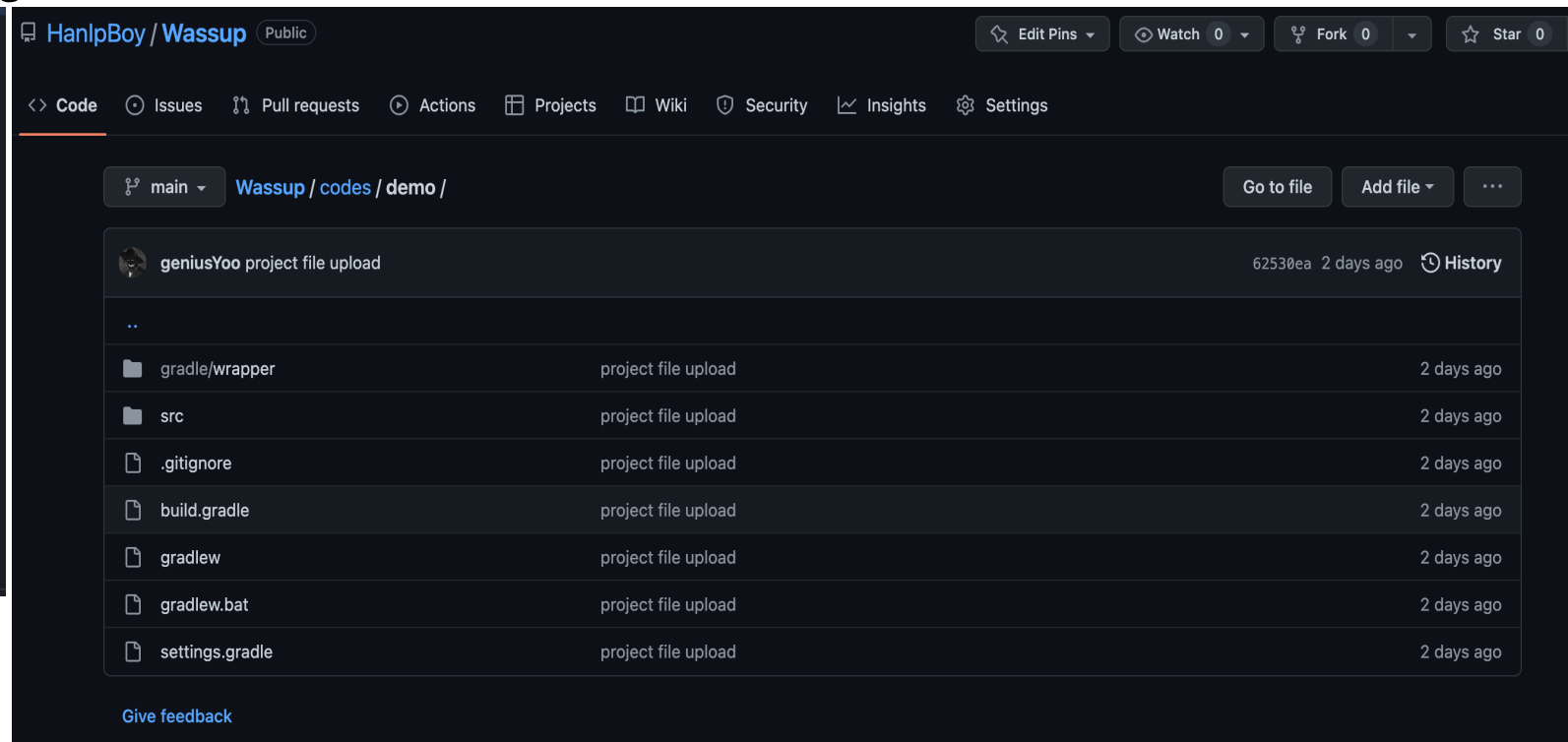
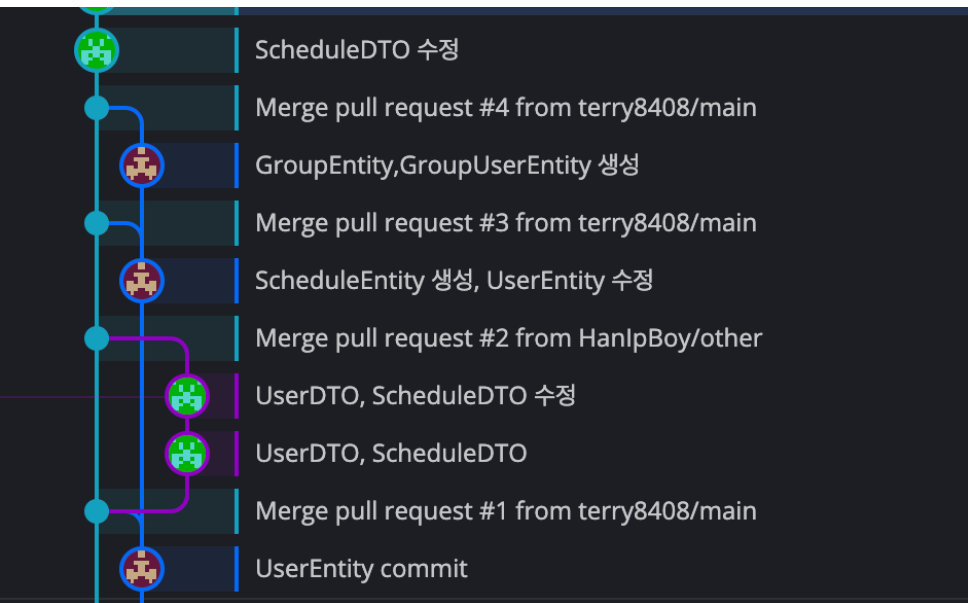
로그인

일정 생성

# 1. 한 주간의 진행상황

## • 1-1. 어떠한 것을 공부했는가

- 영재 - 사용자의 요청을 처음으로 받게 되는 Controller Layer 구현 시작
- 윤재 - 데이터들을 DB에 넣는 형태를 구성하는 Entity 구현 시작
- 프로젝트 파일을 Github Organization에 올려 협업 환경 생성.



# 1. 한 주간의 진행상황

## • 1-2. 결정사항

- Id를 생성할 때, 이메일의 형태로 한 이유는 Firebase Authentication을 이용해 이메일 인증으로 사용자를 생성할 예정이기 때문. 스프링 시큐리티를 사용하려 했으나 스프링 시큐리티에서 사용하는 JWT를 이용하는 것만 채택하고, 소셜 로그인 구현이 아닌 이메일 인증으로 단순화하기로 결정.
- Birth 필드나 created\_at, last\_modified\_at 필드들은 Date객체가 아닌 규격화된 String 타입으로 데이터를 주고받아 TimeZone의 영향으로 데이터의 변질을 막기로 결정.
- origin\_key의 역할은 DB에서 테이블을 식별할 때(테이블을 검색할 때) 기본키의 역할을 함. 또한 테이블의 기본키는 클라이언트와의 통신에서 외부로 공유되면 안되는 정보이기 때문에 보호하기로 결정.
- origin\_key 생성은 JPA로 uuid 생성.

# 1. 한 주간의 진행상황

## • 1-3. 결정사항에 대한 구체적인 설명

### • 로그인 시

1. 클라이언트로부터 로그인 요청이 들어오면, 서버는 id pw를 확인해서 로그인 인증
2. 사용자의 id를 JWT로 암호화해서 클라이언트에게 토큰을 붙여 다시 넘김과 동시에 pw필드는 null로 비워서 보냄.

### • 작업 요청 시

1. 클라이언트가 서버에게 작업을 요청할 때, 서버는 토큰을 받아 해석해 사용자의 id를 알아냄.
2. 알아낸 id로 해당 사용자의 origin\_key를 알아내고
3. 이를 이용해 테이블 검색 (CRUD 전 과정이 마찬가지로 진행됨)

```
// 로그인 요청 c->s
User {
  "id" : "cross_man@naver.com"
  "password" : "abc123"
}

// id,pw가 맞으면 로그인 성공, 토큰을 붙여서 다시 넘기기 s->c
User {
  "id" : "cross_man@naver.com"
  "password" : null
  "token" : "aooalskfnaalkfnkjbfaklksdpoalksfnaalksnf"
}
```



# 1. 한 주간의 진행상황

## • 1-2. 부딪힌 이슈 & 해결사항

### • 보안 문제

- 프로토콜을 구성하던 중, 사용자를 구별하기 위해서는 토큰을 주고받으면서 로그인 상태를 유지하고 작업을 수행하는데, origin\_key가 테이블마다의 기본키이기 때문에 이를 이용해 테이블 검색을 수행해야 한다고 생각함.
- 하지만 클라이언트로부터 요청이 들어올 때, 계속 origin\_key를 주고받는다면 보안의 측면에서 매우 좋지 않기 때문에, 사용자의 id를 JWT로 암호화한 뒤 토큰을 해석해 사용자의 id를 알아내고, id를 통해 DB에서 쿼리로 origin\_key를 알아내고, 알아낸 기본키로 다시 테이블을 검색하는 두번의 DB 접근이 이루어지게 됨.
- 두번 씩 이루어지긴 하지만, 보안적인 측면에서는 이 방법이 더 좋다고 생각하여 이 방법을 채택함.

## 2. 고민하고 있는 이슈들

### • 2-1. 알림

- Notification을 구현하는 데는 여러가지 방법들이 있지만, Firebase Cloud Messaging을 이용해 알림을 구현할지, 스프링 내 라이브러리를 찾아서 구현할 지 고민.

### • 2-2. 앞으로의 개발 환경

- 로컬에서 로직을 구현하고 로컬 DB를 사용해서 개발을 하는 것은 위험한 선택이라고 생각됨.
- 나중에 연결했을 때 문제가 생겨버리면 프로젝트 전체에 문제가 생겨버리기 때문에, 처음부터 스프링과 EC2, MySQL DB까지 연결해서 개발하는 방향이 초반에는 시간이 좀 더 걸리더라도 안정적으로 프로젝트를 수행할 수 있을 것으로 생각됨.

### 3. 앞으로의 계획

- Presentation Layer의 DTO와 Service Layer의 Entity는 다음주까지 계속 구현 예정
- 동시에, EC2와 DB연결 다시 시작해 환경 구축
- 프로토콜도 계속 완성 예정 ..