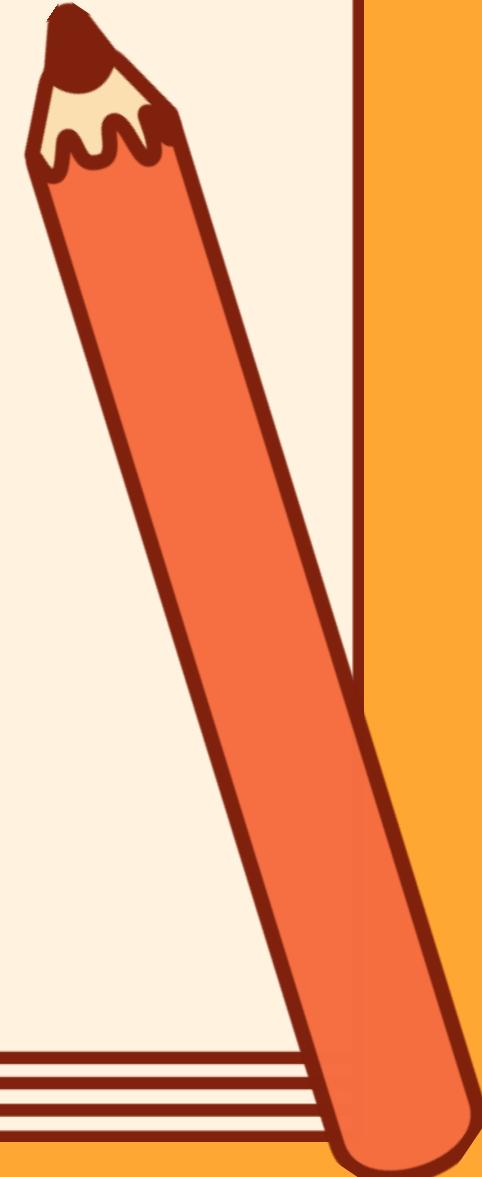


공부인증 사이트

영공 10.0



## 목차

01

개발 목적

03

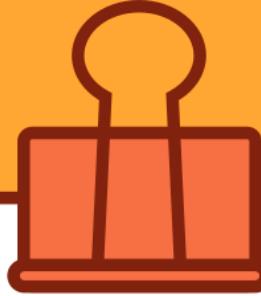
개발 환경

02

프로젝트 기획

04

기능 설명



## ■ 조원소개



김태양

개발 환경  
구축/관리

문유정

UI,UX  
전반적인 디자인

동선미

DB 관리 설계

박유나

PTJ 매니저

# 개발 목적

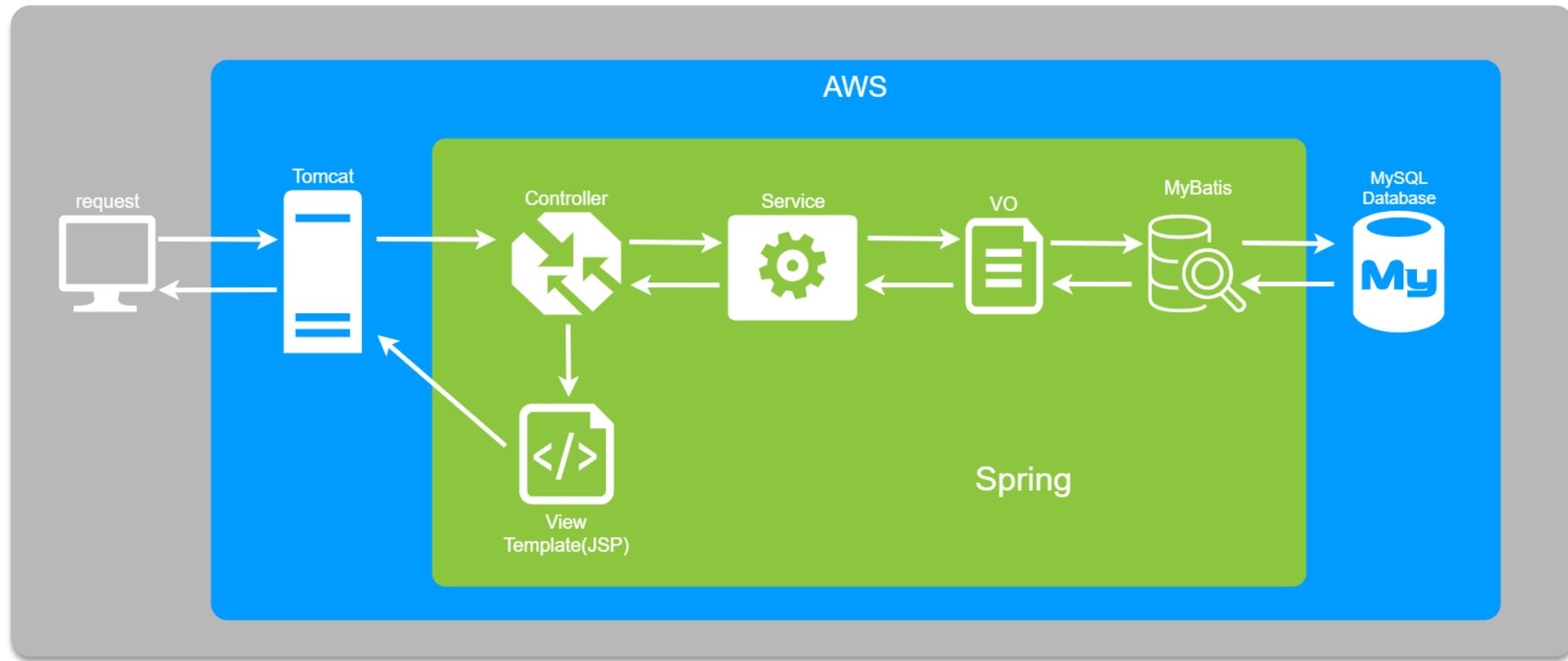


열공 사이트는  
원하는 목표를 이루기 위한 공부기록 사이트입니다.

타이머를 이용해서 공부시간을 기록하고  
캘린더를 통해 일별스케줄을 기록과 todo리스트 진행이 가능해 간편하게 일정을 관리할 수 있  
습니다.  
또한, 차트기능을 제공 일주일간의 공부시간 데이터를 한눈에 확인이 가능합니다.

그룹기능을 통해  
같은 목표를 가진사람들과 서로 공부상태를 확인하면서  
경쟁과 동시에 함께 목표를 이루기 위해 힘이되고 자극 받을 수 있습니다!

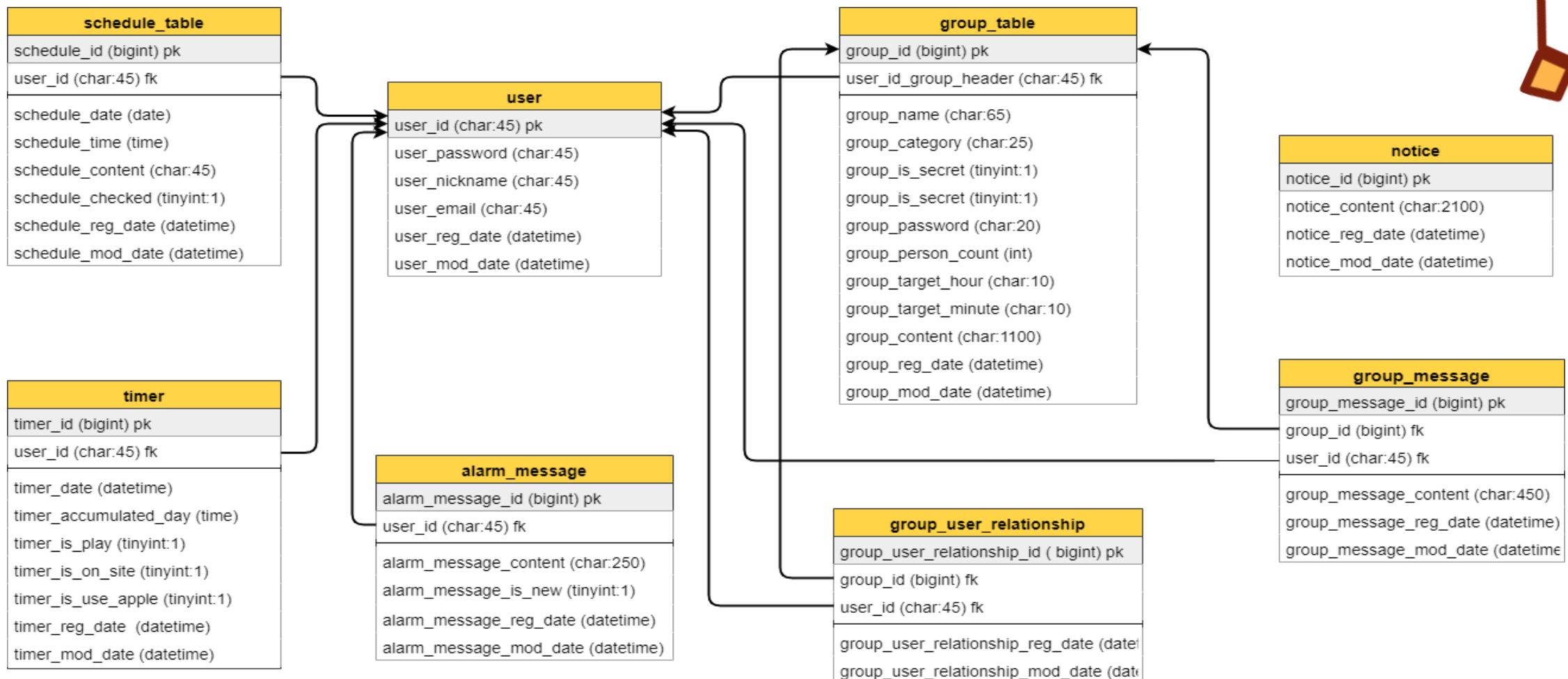
# Project 구조



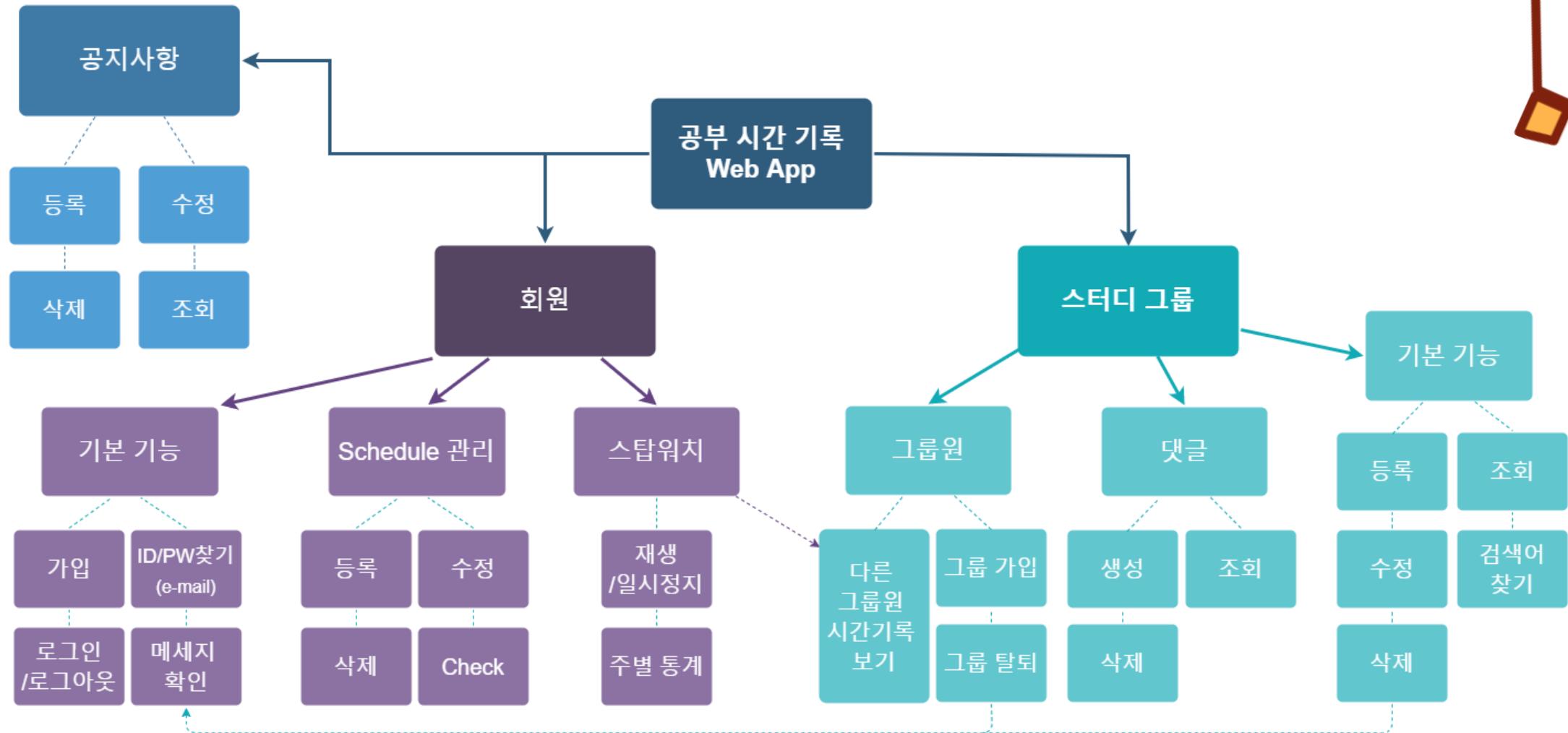
# 개발 일정



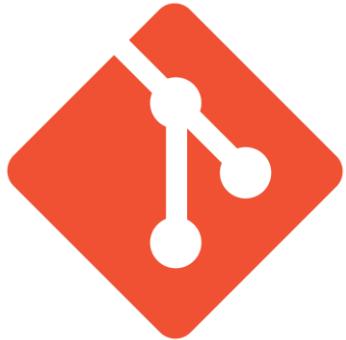
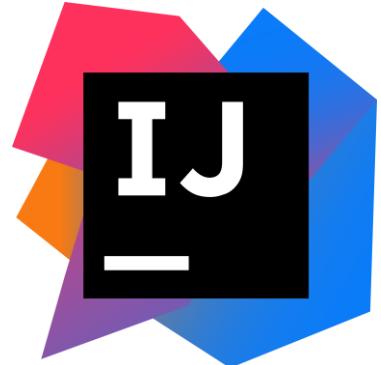
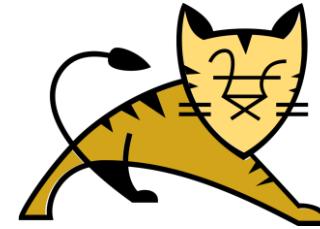
# DB 구조



# 기능 구조도

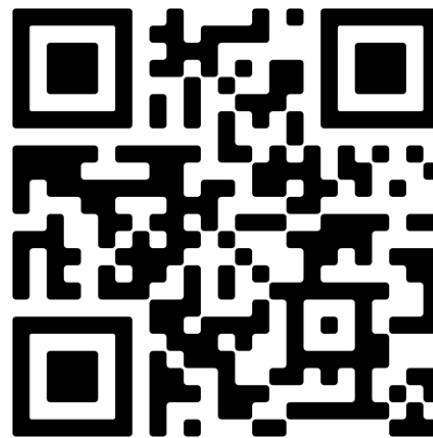


# 개발 환경





# 기능 설명



# 로그인 Session & Cookie

```
<interceptor>
    <mapping path="/*/*"/>
    <exclude-mapping path="/" />
    <exclude-mapping path="/user/login"/>
    <exclude-mapping path="/user/findID"/>
    <exclude-mapping path="/user/findPassword"/>
    <exclude-mapping path="/user/create"/>
    <exclude-mapping path="/user/logout"/>
    <exclude-mapping path="/resources/**"/>
    <exclude-mapping path="/WEB-INF/**"/>
    <exclude-mapping path="/favicon.ico"/>
    <beans:bean class="com.ball.interceptor.SessionAuthCheckInterceptor"/>
</interceptor>
```

## Interceptor URL 설정

```
public class SessionAuthCheckInterceptor implements HandlerInterceptor {
    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception {
        ...
        // JSESSIONID 의 값이 session ID와 일치하고,
        // session의 user_id와 cookie의 user_id가 일치하면
        // auth는 통과
        if( JSESSIONID != null && userCookie !=null ) {
            if (userCookie.getValue().equals(String.valueOf(obj))
                && JSESSIONID.getValue().equals(session.getId() )) {
                log.info("user Session is existed.....");
                return true;
            }
        }
    }
}
```

## Interceptor 통과 조건

```
//add user info to session
session.setAttribute( name: "userID", userVO.getUser_id());

userCookie
    = new Cookie( name: "userCookie", userVO.getUser_id());
userCookie.setSecure(true);
 다른 앤트포인트에 쿠키전달이 안되서 true를 하면 안됨
userCookie.setPath("/");
userCookie.setMaxAge(60*60*24);
if(user_remember) { //로그인 상태 유지하면 userid를 쿠키에 저장함
    userCookie.setMaxAge(60*60*24*365*10); //set cookie 10 years
    JSESSIONID.setMaxAge(60*60*24*180);
    // session은 6개월로 기본 세션 시간은 24시간 (web.xml 에 기술함)
    session.setMaxInactiveInterval(60*60*24*180);
}
res.addCookie(userCookie);
res.addCookie(JSESSIONID);
```

## Login시 session&cookie 발행

- Request URL Check하여 Interceptor 실행
  - home, resource 등을 제외한 모든 url에 실행
- Login시 Controller에서 Cookie 발행
  - 새롭게 발행한 세션 id를 JSESSIONID 쿠키 값으로 지정
  - userID 값을 저장한 쿠키 발행
  - 로그인 유지기간을 추가하여 세션, 쿠키 기간을 변경
- SessionId정보와 Cookie정보를 비교하여 통과 여부 설정

# 스탑워치 Cookie & Event

열공하는 사람들의 오늘의 공  
부량

02:44:36

공부시작하기

User page의  
Main 스탑워치

Controller에서  
쿠키발행

```
if(resetCookie) {
    if(timerCookie != null){
        timerCookie.setMaxAge(0);
    }
    System.out.println("cookie reset");
    timerCookie = new Cookie(name: "timerCookie",
        timerCookie.setMaxAge(remainSecondsFrom3AM());
    timerCookie.setSecure(false);
    timerCookie.setPath("/");
}

else { //오늘 처음 접속해서 타이머 정보가 없는 경우
    timerCookie.setValue(timerVO.getTimer_id()
        + "-0-00:00:00");
}

response.addCookie(timerCookie);
```



지금은 열공시간  
02:18:21  
00

그룹 page의  
Sub 스탑워치

Browser에서  
쿠키상태 전송

```
const saveTimerToDB = function(data, resultFunc){
    $.ajax({
        type: "PUT",
        url: "/ajax/timer/" + timerID,
        data: JSON.stringify(data),
        contentType: "application/json; charset=UTF-8;",
        success: function(result, status, xhr){
            if(timerIsPlay == "1"){
                viewTimerStartInterval();
            }
            if(resultFunc != null){
                resultFunc(timerID + "-" + timerIsPlay + "-" + accumulatedTimeStr);
                [ cookieHour, cookieMinute, cookieSecond ]
                = accumulatedTimeStr.split(':').map(i=>Number(i));
                // console.log(cookieHour, cookieMinute, cookieSecond);
            }
        },
    });
},
```

```
$("#time-toggle").click(function(e){
    if(timerPlayFlag){
        $(this).html('공부시작하기');
        timerPlayFlag = false;
        timerStop(function(resultCookieTimer){
            //타이머정보가 db에 저장되면 타이머의 정보를 쿠키에 저장
            document.cookie = "timerCookie=" + resultCookieTimer
            + ";path=/; expires=" + getDateStringToNextMorning3AM() + ";";
        });
    }
});
```

DB와 쿠키에  
Timer정보를 일치화

- 매일 처음 웹에 접속시 스탑워치(타이머)쿠키 실행
  - 실행되는 쿠키의 최대 지속시간은 다음날 새벽3시까지임
  - 처음 접속 후에는 DB정보와 비교하여 쿠키의 Reset여부를 결정함
- 스탑워치 버튼 Click이벤트로 상태정보로 DB에 저장
- Group List/Read Page에서 Sub 타이머 동작
  - View 템플릿을 분리하여 다른 페이지에도 쉽게 확장 가능
  - JavaScript의 beforeunload 이벤트 함수를 활용하여 유저가 페이지 이동시에도 타이머 상태를 쿠키 및 DB에 업데이트하여 유지함
- setInterval함수를 사용하여 1초에 한번씩 동작
  - setInterval이 Start할때의 시간을 저장하여 실행할 때의 시간차이를 계산하여 타이머를 동작시킴(event loop와 thread 관련)

# 평균시간 View (MySQL)

취업

취업을 하는 스터디 그룹

하기

목표시간 : 8시간 00분 그룹인원 : 1/2명 그룹장 : 열공하는 사람

공부량 : 2시간 44분 시작일 : 2021-07-08

개발자로 취업하는 사람들의 모임입니다.

자유 스터디

멤버 7

목표시간 : 8시간 00분 그룹인원 : 3/10명 그룹장 : 열공@-@

공부량 : 0시간 23분 시작일 : 2021-06-15

아무거나 공부합시다.

토익

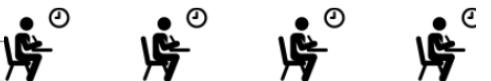
영渣들의 토익도전

| 현황

목표시간 : 2시간 20분 그룹인원 : 2/5명 그룹장 : 농담곰

공부량 : 1시간 22분 시작일 : 2021-06-26

2021년 토익 800목표 하루 한번 출석 체크 해주세요



## 그룹별 평균 공부시간 표시

- MySQL의 AVG함수를 사용하여 평균시간 계산
  - 그룹별 유저의 타이머 정보를 조인하여 Time데이터를 Second로 변환하여 평균을 계산
- MySQL의 Case문 활용
  - 유저의 타이머 정보를 구분하여 return값을 달리함
  - 타이머가 play상태이면 마지막 저장시간과 현재 시간의 차이를 더하여 return함
- Group List / Read 페이지에서 View를 활용

View 생성 SQL문

```
create view accumulated_time_per_group as
select `g`.`group_id` AS `group_id`,
       SEC_TO_TIME(ROUND(AVG(TIME_TO_SEC(
CASE
        WHEN (`t`.`timer_accumulated_day` IS NULL) THEN '00:00:00'
        WHEN (`t`.`timer_is_on_site` = 0) THEN `t`.`timer_accumulated_day`
        WHEN
          ((`t`.`timer_is_play` = 1)
           AND (`t`.`timer_is_on_site` = 1)
           AND (`t`.`timer_is_use_apple` = 0))
        THEN
          ADDTIME(`t`.`timer_accumulated_day`, TIMEDIFF(NOW(), `t`.`timer_mod_date`))
        ELSE `t`.`timer_accumulated_day`
      END)),
      0)) AS `group_accumulated_avg_time`

from group_table g
  left join group_user_relationship gr on g.group_id = gr.group_id
left join (
  select * from timer
  where timer_date = if((HOUR(NOW()) < 3), CAST(NOW() - INTERVAL 1 DAY), CAST(NOW() AS DATE))
) t on gr.user_id = t.user_id
GROUP BY g.group_id
order by g.group_id desc;
```

View 사용

```
<select id="groupRead" resultType="com.ball.vo.GroupVO">
  select g.group_id, g.user_id_group_header, u.userNickname as user_nickname_group_header,
         g.group_name, g.group_category, g.group_password, g.group_target_hour, g.group_
         g.group_person_count, g.group_reg_date, atpg.group_accumulated_avg_time
  from group_table g
    left outer join user u on g.user_id_group_header = u.user_id
    left outer join group_user_relationship gr on gr.group_id = g.group_id
    left outer join accumulated_time_per_group atpg on g.group_id = atpg.group_id
  where g.group_id = #{group_id}
  group by g.group_id
</select>
```

# User 알람서비스

알람 확인하기		
날짜	내용	확인
12:44:23	■개발자 취업 준비■ 그룹에서 탈퇴하셨습니다...	✉
12:44:04	■개발자 취업 준비■ 그룹에 가입하셨습니다...	✉
11:54:47	유저닉네임11님께서 efwegweg 그룹에 탈퇴...	✉
11:42:09	유저닉네임11님께서 wewe 그룹에 가입하셨...	✉
2021/07/08	유저닉네임11님께서 efwegweg 그룹에 가입...	✉
2021/07/01	10-0에 합류한 것에 감사드립니다. 저희와 함...	✉
2021/06/24	열공합시다~!!	✉
2021/06/20	홧팅~~!!	✉
2021/06/20	홧팅~~!!	✉
2021/06/20	홧팅~~!!	✉

[더보기](#)

이벤트 발생시 전달하는 시스템메세지로  
Controller와 Service에서 insert처리

```
@PostMapping("groupRemove")
public String groupRemove (Long group_id){
    String groupDestroyMessage = " 그룹이 파괴되었습니다. 다른 그룹에 가입하셔서 열공해주세요!! ";
    groupService.groupRemove(group_id,groupDestroyMessage);
    return "redirect:/group/list";
}

alarmVO.setUser_id(groupVO.getUser_id_group_header());
alarmVO.setAlarm_message_is_new((byte)1);
alarmVO.setAlarm_message_content(newUserVO.getUserNickname()+"님께서 "+groupVO.getGroupName()
    +" 그룹에 가입하셨습니다.🎉 환영의 메세지를 남겨보세요! 😊");

alarmVOjoin.setUser_id(join.getUserId());
alarmVOjoin.setAlarm_message_is_new((byte)1);
alarmVOjoin.setAlarm_message_content(groupVO.getGroupName()
    +" 그룹에 가입하셨습니다.🎉 오늘도 열공!열공! 😊");
```

2021/06/20 첫정~!!

2021/06/20 핫팅~!!

더보기

□ ajax로 최근 알람 10개를 가져와 화면에 출력해준다  
더보기버튼을 누르면 추가10개를 가져온다

```
model.addAttribute( attributeName: "firstCriterionNumber", criterionNumber==null? 0: criterionNumber+1);
```

```
let changeCriterionNumber=${firstCriterionNumber};  
$(document).ready(function (){
```

초기값

```
const moreList = (criterionNumber)=>{  
    $.ajax({  
        type:"post",  
        url:"/ajax/user/alarmMessage",  
        data:{  
            userID:${userID},  
            criterionNumber:criterionNumber  
        },  
        dataType:"json",  
        success : function (res){  
            const list = res['list'];  
            let data = "";  
  
            for (let i = 0; i < list.length; i++) {  
                data += "<tr class='itemTitle'>";  
                data += "  <input type='hidden' value='"+ list[i].alarm_message_id + "'></input>";  
                data += "  <td style='font-size: 12px; class='align-middle'>" + displayTime(list[i].alarm_message_reg_date) + "</td>";  
                data += "  <td>";  
                data += "    <div id='alarm-content'>" + list[i].alarm_message_content + "</div>";  
                data += "  </td>";  
                data += "  <td id='new' style='padding-left: 15px;'>" + changeImg(list[i].alarm_message_is_new) + "</td>";  
                data += "</tr>";  
                data += "<tr class='hideContent' style='pointer-events: none; '>";  
                data += "  <td></td>";  
                data += "  <td>" + list[i].alarm_message_content + "</td>";  
                data += "  <td></td>";  
                data += "</tr>";  
            }  
            $('#dataSection').append(data);  
            changeCriterionNumber = $("#dataSection input:last").val();  
  
            //더보기할 내용없을시 더보기 버튼 삭제  
            if(list.length<10){  
                $("#addBtn").remove();  
            }  
        }  
    })  
}
```

```
@PostMapping (value = "/alarmMessage") //userVo vo  
public ResponseEntity<HashMap<String, Object>> userAlarmList(Long criterionNumber, String userID) throws Exception {  
  
    Criteria cri = new Criteria(criterionNumber, amount: 10);  
    HashMap<String, Object> result = new HashMap<>();  
  
    result.put("list", alarmService.getListWithPage(cri,userID));  
    return ResponseEntity.ok(result);  
}
```

알람 확인하기

날짜	내용	확인
12:44:23	■개발자 취업 준비■ 그룹에서 탈퇴하셨습니다...	

알람 확인하기

날짜	내용	확인
12:44:23	■개발자 취업 준비■ 그룹에서 탈퇴하셨습니다...	
	■개발자 취업 준비■ 그룹에서 탈퇴하셨습니다.	

읽지 않은 알람메세지를 표현하고 확인시에  
읽지 않은 갯수를 ajax로 반영이 되게 작성

```

$.ajax({
  type:"post",
  url:"/ajax/user/alarmCount",
  data:{
    alarmID:alarmID,
    userID:"${userID}"
  },
  dataType:"json",
  success : function (res){
    const count = res['alarmCount'];
    $('#alarm-count').text(count);
  },
}

@PostMapping (value = "/alarmCount") //userVo vo
public ResponseEntity<HashMap<String, Object>> userAlarmCount(Long alarmID, String userID) throws Exception {
  HashMap<String, Object> result = new HashMap<>();

  // 알람 읽음 처리 및 DB 저장
  alarmService.modify(alarmID);
  result.put("alarmCount", alarmService.alarmCount(userID));
  return ResponseEntity.ok(result);
}

```

# User 가입한 그룹 리스트

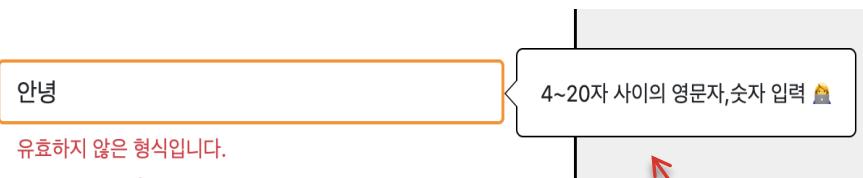
취업하자!!🔥님의 속한 그룹 😊



- 가입한 그룹이 있을시에는 그 그룹들에 대한 리스트를 불러오고  
없을시에는 배너를 띄움

```
<c:if test="${empty userJoinGroupList}">
    <div style="...">
        <img src='/resources/img/group_empty.jpg' class="group-empty" style="..." onclick="location.href='/group/list'" />
    </div>
</c:if>
<c:forEach var="groupList" items="${userJoinGroupList}">
    <div class="card user-card-group" style="..." onclick="location.href='/group/read?group_id=${groupList.group_id}'">
        <div class="card-body">
            <div class="row">
                <div class="col-10 group-category">${groupList.group_category}</div>
                <div class="col-2 text-end groupSecret">
                    <c:if test="${groupList.group_is_secret==1}">
```

# 회원가입, 수정

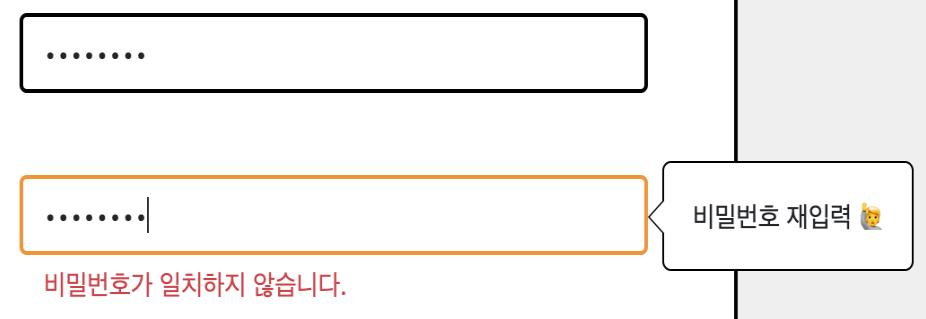


popover 기능으로 사용 가능한 형식을 보여주고  
input에 유효하지 않은 형식이 들어올 경우 오류메세지 반환

```
//input popover 처리
$('#user_id').popover({content:"4~20자 사이의 영문자, 숫자 입력 🎉",placement:'right',trigger:'focus'});
$('#inputPassword1').popover({content:"4~20자 사이의 영문자, 숫자 입력 🎉",placement:'right',trigger:'focus'});
$('#inputPassword2').popover({content:"비밀번호 재입력 🎉",placement:'right',trigger:'focus'});
$('#user_nickname').popover({content:"홈페이지내에서 사용될 닉네임 😊",placement:'right',trigger:'focus'});
$('#user_email').popover({content:"실제 사용중인 이메일 입력 🎉",placement:'right',trigger:'focus'});

//아이디, 비밀번호 체크할 정규식 표현
let re = /^[a-zA-Z0-9]{4,20}$/;

$('#user_id').keyup(function (){
    $('#user_id').val($('#user_id').val().replace(/\s/gi, ""));
    if(!re.test($('#user_id').val())){
        $('#idHelpInline').text("유효하지 않은 형식입니다.");
    }
})
```



□ 비밀번호는 2개의 데이터값을 비교해 일치확인

```
const pwCheck = ()=> {
  if($('#inputPassword2').val() != $('#inputPassword1').val()) {
    $('#passwordHelpInline2').text("비밀번호가 일치하지 않습니다.");
  }
}
```

```
@PostMapping("/create")
public String create(UserVO vo, RedirectAttributes rAttr){

  //아이디 중복 조회
  boolean Idcheck = userService.idCheck(vo.getUser_id());
  if(! Idcheck){ // id가 존재할때 다시 회원가입 페이지로
    rAttr.addFlashAttribute( attributeName: "idFail", attributeValue: "fail");
    rAttr.addFlashAttribute( attributeName: "writing", vo);
    return "redirect:/user/create";
  }

  //이메일 중복 조회
  boolean emailcheck = userService.emailCheck( user_id: null,vo.getUser_email());
  if(! emailcheck){ // 이메일이 존재할때 다시 회원가입 페이지로
    rAttr.addFlashAttribute( attributeName: "emailFail", attributeValue: "fail");
    rAttr.addFlashAttribute( attributeName: "writing", vo);
    return "redirect:/user/create";
  }
}
```

□ 아이디와 이메일은 작성품을 Controller에서 DB와의 대조를 통해 중복이 아닌경우 insert 중복일 경우 다시 회원가입페이지에서 오류메세지를 반환

dbwjd312@naver.com

이미 사용중인 이메일입니다.

```
userService.userCreate(vo, alarmMessageVO);
return "redirect:/user/login";
```

# 그룹리스트

**자격증**  
**그룹이름8**

목표시간 : 3시간 30분 그룹인원 : 6/5명 그룹장 : 유저닉네임8  
공부량 : 0시간 00분 시작일 : 2021-06-17

그룹 내용

**자격증**  
**그룹이름7**

목표시간 : 3시간 30분 그룹인원 : 9/5명 그룹장 : 유저닉네임7  
공부량 : 0시간 00분 시작일 : 2021-06-17

그룹 내용

**이직**  
**그룹이름6**

목표시간 : 3시간 30분 그룹인원 : 7/5명 그룹장 : 유저닉네임6  
공부량 : 0시간 00분 시작일 : 2021-06-17

그룹 내용

**취업**  
**테스트 그룹 이름**

목표시간 : 3시간 30분 그룹인원 : 6/20명 그룹장 : 유저닉네임5  
공부량 : 0시간 00분 시작일 : 2021-06-17

수정이 되나

**취업**  
**그룹이름4**

목표시간 :  
공부량 : 0시간 00분

**더보기 버튼 클릭시 그룹 출력**

그룹 내용

**스터디 그룹**

✓ --- 토익 취업 자격증 이직

상단의 카테고리 검색 조건에 따라 가장 최근에 생성된 그룹들을 20개씩 화면에 출력해준다. 출력된 그룹 외에 남은 그룹이 20개 이상일 경우 하단의 더보기 버튼이 생성되며, 그 버튼을 눌렀을 경우에만 20개씩 추가로 출력이 가능하다.  
더보기 버튼을 눌러서 생성된 그룹들은 ajax처리하여 가져온다.

**이직**  
**댕댕곰**

목표시간 : 3시간30분 그룹인원 : 5/5명 그룹장 : 유저닉네임3  
공부량 : 0시간 00분 시작일 : 2021-06-17

농담곰을 위하여

**공부카테고리**  
**테스트 그룹 이름**

목표시간 : 3시간30분 그룹인원 : 5/7명 그룹장 : 유저닉네임2  
공부량 : 0시간 00분 시작일 : 2021-06-17

수정이 되나

**취업**  
**그룹이름1**

목표시간 : 3시간30분 그룹인원 : 5/5명 그룹장 : hana  
공부량 : 0시간 00분 시작일 : 2021-06-17

그룹 내용

더보기

- Ajax를 이용하여 출력되지 않은 그룹을 groupText에 담아 버튼 클릭시에 출력해준다.

## 추가적인 그룹 가져오는 ajax

```

const moreList = (criterionNumber, keyword, type) => {
  var startNum = $(".user-card-group input:last").val();
  $.ajax({
    type: "POST",
    url: "/ajax/addList",
    dataType: "json",
    data: {
      criterionNumber : criterionNumber,
      keyword: keyword,
      category:type
    },
    success(data){
      const criNum = data['criNumber'];
      if(criNum.length < 20){
        $("#addBtn").remove();
      }
      var groupText = "";
      $(".center-block").append(groupText);
      changeCriterionNumber = $(".user-card-group input:last").val();
    }
  })
}

```

## groupAjaxController

```

@PostMapping(value = "/addList")
public ResponseEntity<HashMap<String, Object>> getList (Criteria cri) throws Exception{
  HashMap<String, Object> result = new HashMap<>();
  result.put("criNumber", groupService.allRead(cri));
  return ResponseEntity.ok(result);
}

for(let i = 0; i < criNum.length; i++) {
  groupText += "<div class='card user-card-group' style='cursor: pointer;'";
  groupText += "  <input type='hidden' name='group_id' value='"+criNum[i].";
  groupText += "  <div class='card-body'>";
  groupText += "    <div class='row'>";
  groupText += "      <div class='col-10 group-category'>"+criNum[i].";
  groupText += "      <div class='col-2 text-end groupSecret'>";
  if(criNum[i].group_is_secret == 1){
    groupText += "        <div style='text-align: center; margin-top: 10px'";
    groupText += "          <img src='/resources/img/lock.png' style='width: 20px; height: 20px;'";
    groupText += "        </div>";
  }
  groupText += "      </div>";
  groupText += "    </div>";
  groupText += "    <div class='group-list-margin'>";
  groupText += "      <span class='group-title'> "+criNum[i].group_name+" </span>";
  groupText += "    </div>";
  groupText += "    <div>";
  groupText += "      <span class='group-list-title'>목표시간 : </span><span>"+criNum[i].group_target_time+"</span>";
  groupText += "    </div>";
  groupText += "    <div>";
  groupText += "      <span class='group-list-title'>공부량 : </span><span>"+(criNum[i].group_accumulated_avg_time.length)+"</span>";
  groupText += "    </div>";
  groupText += "    <span class='group-list-title'> 시작일 : </span><span>"+criNum[i].group_start_date+"</span>";
  groupText += "  </div>";
  groupText += "  <ul class='list-group list-group-flush'>";
  groupText += "    <li class='list-group-item group-content'>"+criNum[i].group_content+"</li>";
}

```

# 그룹 시스템

비밀방일 경우 비밀번호 입력 후 진입 가능

그룹장 Seven  
멤버 2  
목표시간 : 3시간 30분 그룹인원 : 2/2명 그룹장 : 유저닉네임7  
공부량 : 0시간 00분 시작일 : 2021-06-24

그룹원 Seven  
멤버 2  
목표시간 : 3시간 30분 그룹인원 : 2/2명 그룹장 : 유저닉네임7  
공부량 : 0시간 00분 시작일 : 2021-06-24

둘다 아님 Seven  
멤버 2  
목표시간 : 3시간 30분 그룹인원 : 2/2명 그룹장 : 유저닉네임7  
공부량 : 0시간 00분 시작일 : 2021-06-24

- groupList에서 출력된 그룹중 원하는 그룹을 클릭시에 조회가 가능하다.  
그룹장일 경우 좌측 상단의 파괴하기, 수정하기 버튼이 보이고  
그룹원일 경우 탈퇴하기 버튼이 보이며,  
둘 다 아닐 경우에는 가입하기 버튼이 보인다.

```
if(${group.user_id_group_header eq user_id}) {  
    $('.btn-warning').attr('hidden', true)  
    $('.btn-block').attr('hidden', true)  
} else{  
    $('.btn-danger').attr('hidden', true)  
    $('.btn-default').attr('hidden', true)  
}  
if(${join >=1}){  
    $('.btn-warning').attr('hidden',true);  
} else{  
    $('.btn-block').attr('hidden', true)  
}  
  
$('#joinGroup').click(function (){  
    if(${group.group_join_person_number >= group.group_person_count}){  
        alert('인원수가 초과되었습니다.')  
        return false;  
    }  
    $('#openForm').attr("action", "/group/read").attr("method", "post").submit();  
})
```

그룹인원이 초과하면 가입 불가

# 그룹생성

## 그룹 생성

그룹이름

그룹 카테고리

비밀방

비밀번호

그룹 인원

그룹 공지

목표시간

```
@GetMapping("create")
public String register(HttpServletRequest request, Model model){
    String userID = String.valueOf(request.getSession().getAttribute("userID"));
    model.addAttribute("user_id", userID);
    return "group/groupCreate";
}
```

- 세션에 로그인 되어있는 user\_id를 가져온 후에  
로그인 한 사용자가 groupList 좌측 상단에  
등록 버튼을 누르면  
그 유저가 그룹장이 된다.

```
@PostMapping("create")
public String register(GroupVO group, HttpServletRequest request){
    String userID = String.valueOf(request.getSession().getAttribute("userID"));
    groupService.register(group, userID);
    return "redirect:/group/read?group_id=" + group.getGroup_id();
}
```

- 그룹 생성 버튼을 누르면 그룹이 추가되어  
groupRead 페이지로 이동한다.

# 그룹 수정

## 그룹 수정

LuckySeven

취업

비밀방

7777

2

Lucky

목표 시간 3시간 ▾ 30분 ▾

수정하기 취소 버튼

```
@GetMapping("/modify")
public String modify(Long group_id, Model model, @ModelAttribute("cri") Criteria cri){
    model.addAttribute("group", groupService.oneRead(group_id));
    return "group/groupModify";
}
```

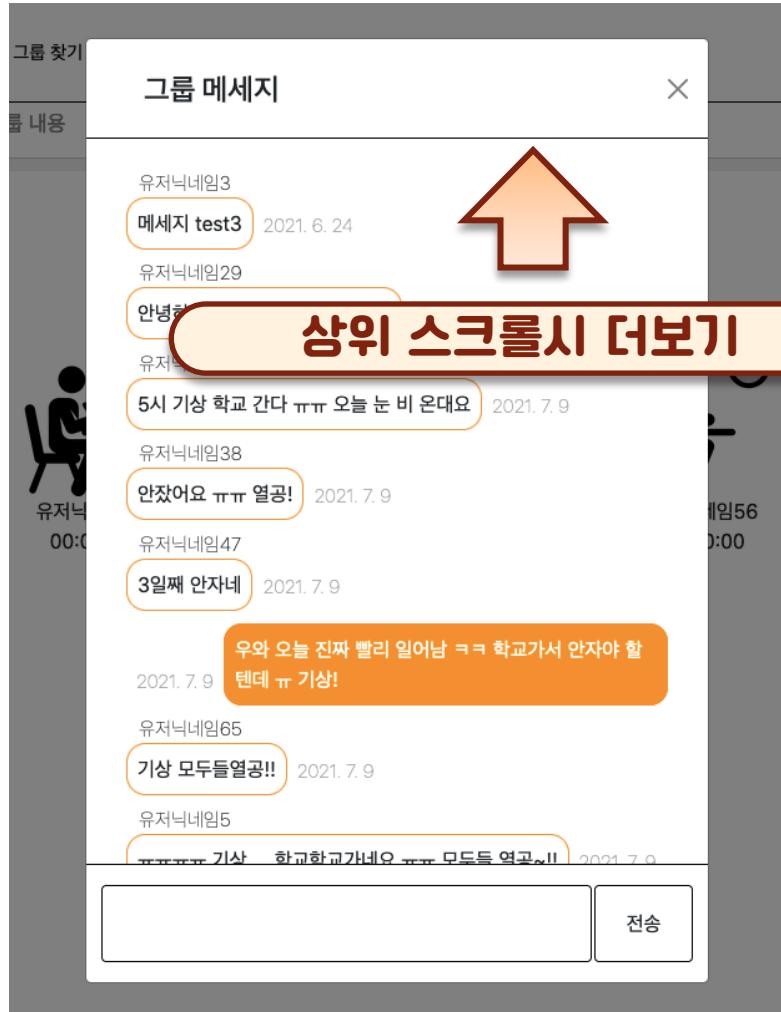
- Group\_id를 Controller에 Model에 실어서 해당하는 그룹 정보를 가져올 수 있다.

```
$("#group_category").val("${group.group_category}").attr("selected", true);
$("#group_person_count").val("${group.group_person_count}").attr("selected", true);
$("#group_target_hour").val("${group.group_target_hour}").attr("selected", true);
$("#group_target_minute").val("${group.group_target_minute}").attr("selected", true);
```



DB에 저장되어 있는 값을 select option에 선택된 상태로 가져올 수 있다

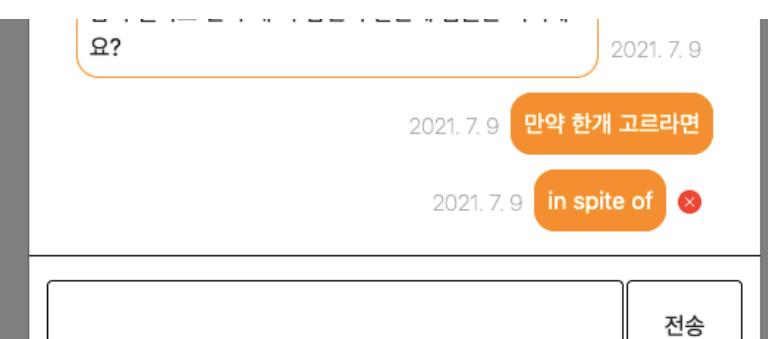
# 그룹 메시지



## 메시지(댓글) 입력 (ajax 사용)



## 선택한 메시지(댓글) 삭제



그룹에 가입한  
사용자에게  
메시지 보기, 입력, 삭제를  
Ajax를 활용하여 구현.

## ▢ 그룹 메시지를 가져오는 ajax

```
function getList(group_id,limit,callback){  
    $.ajax({  
        url:'./group/read/ajax/list/?group_id='+group_id,  
        type:'GET',  
        data:{  
            limit:limit  
        },  
        dataType: "json",  
        success:function (data){  
            text = ""  
            var list = data['list'];  
            const userId = data['userID'];  
            for(var i=list.length-1; i >= 0; i--){  
                if(list[i].user_id == userId){  
                    text += "<div class='d-flex flex-row-revers  
                    text += "<div class='p-2 remove_message' st  
                    // text += "<button class='remove_message b  
                    text += "<div class='border rounded-3 p-2'  
                    text += "<div style='font-weight: lighter;  
                    text += "</div>"  
                }else{  
                    text += "<div style='font-weight: lighter;  
                    text += "<div class='d-flex flex-row >"  
                    text += "<div class='p-2 border rounded-3'  
                    text += "<div style='text-align:end; font-w  
                }  
            }  
            callback(text);  
        },  
        error: (log)=>{alert(log)}  
    });  
}
```

## ▢ groupMessageAjax Controller

```
@GetMapping("group/list")  
public ResponseEntity<HashMap<String, Object>> readMessage(@RequestParam("group_id") Long group_id, int limit, HttpSession session){  
    String userID = String.valueOf(session.getAttribute("userID"));  
    HashMap<String, Object> hashMap = new HashMap<>();  
    hashMap.put("list", messageService.groupMessageMoreRead(limit, group_id));  
    hashMap.put("userID", userID);  
    return ResponseEntity.ok(hashMap);  
}
```

## ▢ Ajax 함수 구현 부분

```
// 처음 메세지 가져오기  
messageService.getList(group_id, limit, function (result) {  
    $('#readGroupMessage').html(result);  
})
```

Ajax로 Controller에 가져올 댓글 개수를 보내준 후,  
Controller에서 받은 파라미터로 Service 처리.  
콜백을 활용하여 데이터를 jsp에서 사용

# 캘린더

July 2021

← Today →

당일날짜를 기준으로 캘린더 출력

일	월	화	수	목	금	토
27	28	29	30	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

```
loadYYMM(init.today);
loadDate(init.today.getDay(), init.today.getMonth(), getWeekNo(init.today.toDateString()), init.today.toDateString());
scheduleList(init.today);
```

← Today →

□ 전월과 익월로 전환이 가능하며  
today버튼으로 당일로 이동 가능

```
<div class="calendar-nav">
  <a href="javascript:loadYYMM(init.prevMonth());"><i class="fas fa-arrow-left"></i></a>
  <a href="javascript:goToday()">Today</a>
  <a href="javascript:loadYYMM(init.nextMonth());"><i class="fas fa-arrow-right"></i></a>
</div>
```

```
nextMonth: function () {
  let d = new Date();
  d.setDate(1);
  d.setMonth(++this.monForChange);
  this.activeDate = d;
  return d;
},
prevMonth: function () {
  let d = new Date();
  d.setDate(1);
  d.setMonth(--this.monForChange);
  this.activeDate = d;
  return d;
},
```

## □ 주말/전월,익월의 데이터에는 클래스를 추가 다른 디자인

```
//전월,익월 데이터
const prevDate = new Date(firstDay);
const nextDate = new Date(lastDay);
const PLDay = prevDate.getDay();
const NLDay = nextDate.getDay();

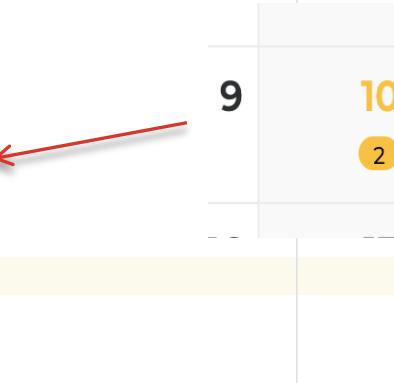
const prevDates = [];
const nextDates = [];
//전월 데이터
if (PLDay !== 5) {
  for (let i = 0; i < PLDay; i++) {
    let yesterday = new Date(prevDate.setDate(prevDate.getDate() - 1));
    prevDates.unshift(yesterday);
  }
}
//익월 데이터
for (let i = 1; i < 7 - NLDay; i++) {
  let tomorrow = new Date(nextDate.setDate(nextDate.getDate() + 1));
  nextDates.push(tomorrow);
}
const notThis = prevDates.concat(nextDates);
```

일	월	화
27	28	
4	5	

## □ 각각의 날짜는 다양한 이벤트 처리를 위해 VO에 LocalDateTime 객체변수 사용

```
<input type="hidden" name="count" value="${fullDate}">
```

```
$ajax({
    type:"post",
    url:"/ajax/schedule/list",
    date:{
        date:displayTime(date).toString(),
    },
    success : function (res){
        let list = res['list'];
        let count = list.length;
        let data = "";
        for (let i = 0; i < count; i++) {
            const checkValue = list[i].schedule_checked ? "checked" : '';
            const checkLink = list[i].schedule_content ? "style='text-decoration: underline'" : '';
            data += `<li ${checkValue} ${checkLink}>`;
            data += `    <p style='cursor: pointer;'>`;
            data += `        <span class="timeStrong">${timeChange(String(list[i].schedule_time))}</span><br/>`;
            data += `        ${list[i].schedule_content}</p></li>`;
            data += `<div class="form-check" style="font-size: 13px; margin-top: -13px; margin-bottom: 15px;">`;
            data += `<label><input class="form-check-input" ${checkValue} type="checkbox" name="todo" value=${list[i].schedule_checked}>&nbsp;check</label></div>
        }
        document.querySelector( selectors: '#calendar-events').innerHTML = data;
        document.querySelector( selectors: '.primary-color span').textContent = count;
    }
})
```



SAT JUL 10 2021

Saturday,  
July 2nd

2 Items

10:00 AM

토익공부하기

check

2:30 PM

인강시청

check

+ Add new item

## □ ajax로 Date객체를 전달하고 insert처리 후 결과 화면출력

Saturday, July 2nd

2 Items

10:00 AM 토익 공부하기  check

2:30 PM 인강시청  check

+ Add new item

JULY 2021

2021년 7월 10일

X

스케줄 등록

몇시

몇분

스케줄 내용

확인 취소

The screenshot shows a calendar interface for July 2021. A modal dialog is open for adding a new event on July 10th. The dialog has fields for time (Hour and Minute dropdowns), content (Text area), and confirmation buttons (확인, 취소). Red arrows point from the 'schedule\_date' variable in the JavaScript code to the date input field in the modal and from the 'data' parameter in the jQuery append method to the content area of the modal.

```
$('#calendar-events').append(data);  
let day = new Date(schedule_date);  
scheduleList(day);  
loadYYMM(day);
```

## □ 스케줄이 추가되면 캘린더의 해당일에 스케줄갯수 출력

SAT JUL 10 2021

Saturday, July 2nd

3 Items

10:00 AM 토익 공부하기  check

2:30 PM 인강시청  check

7:30 PM 영어회화  check

+ Add new item

The screenshot shows the same calendar interface after the event was added. The '3 Items' message is displayed above the event list, indicating that the count has been updated. The event details remain the same as in the previous screenshot.

July 2021

일	월	화	수	목	금	토
27	28	29	30	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

The screenshot shows the full monthly calendar for July 2021. Each day cell contains a number representing the count of events scheduled for that day. For example, July 10th (Saturday) has a value of 3, while other days have values of 0 or 1.

# Todo 리스트 구현

```
//todoList 처리
$("#calendar-events").on("click","div input",function () {
    let checkToggle = $(this).val();
    let selectTodo = $(this).closest("div").prev();
    let scheduleID = selectTodo.find("input").val();
    if(checkToggle==0){
        $(this).val(1);
        selectTodo.css('text-decoration','line-through');
        todoCheck(scheduleID);
    }
    else {
        $(this).val(0);
        selectTodo.css('text-decoration','');
        todoCheck(scheduleID);
    }
})
```

10:00 AM

토익 공부하기

check

10:00 AM

토익 공부하기

check

## DB반영

```
function todoCheck(scheduleID){
    $.ajax({
        type:"post",
        url:"/ajax/schedule/todo",
        data:{
            scheduleID:scheduleID,
        },
        success : function (res){
        },
        error : ()=>{}
    })
}
```

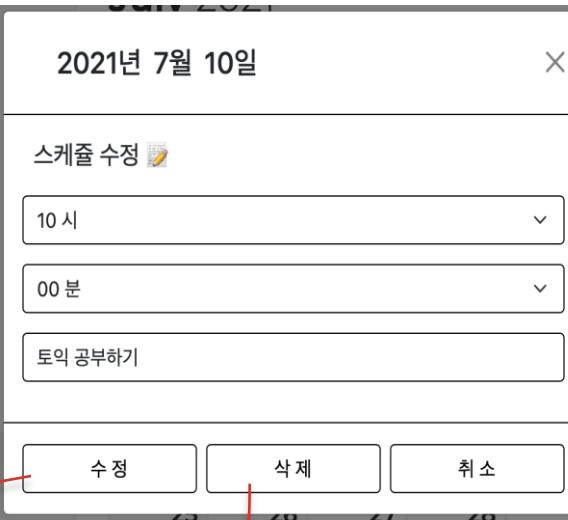
scheduleService.todoScheduleChecked(scheduleID);

## □ Ajax 부분

스케줄하나 선택시 ajax로 해당정보를 출력

```
$('#schedule_id').val(scheduleVO.schedule_id);
$('#schedule_date').val(scheduleVO.schedule_date);
```

```
//스케줄 수정
$('#modify_submit').click(function () {
    let queryString = $("form[name=scheduleModify]").serialize();
    $.ajax({
        type: "post",
        url: "/ajax/schedule/modify",
        data: queryString,
        success: function (res) {
            let date = res['date'];
            let day = new Date(date);
            scheduleList(day);
            loadYYMM(day);
            $('#modifySchedule').find('form')[0].reset();
            $('#modifySchedule').modal("hide");
        },
        error: () => {}
    });
});
```



```
//스케줄 삭제
$('#delete_submit').click(function () {
    let queryString = $("form[name=scheduleModify]").serialize();
    $.ajax({
        type: "post",
        url: "/ajax/schedule/delete",
        data: queryString,
        success: function (res) {
            let date = res['date'];
            let day = new Date(date);
            scheduleList(day);
            loadYYMM(day);
            $('#modifySchedule').find('form')[0].reset();
            $('#modifySchedule').modal("hide");
        },
        error: () => {}
    });
});
```

□ 해당 스케줄을 클릭시  
수정과 삭제 가능  
각각 ajax 구현



# 총평

프로젝트를 마치며

- 5개월동안 배웠던 내용을 활용하여 프로젝트를 완성시켰다는 성취감을 느꼈음
- 어려울 것 같아, Spring Security와 OAuth2 API를 포기하고 Session과 Cookie만을 사용한 것에 아쉬움
- 여러가지 버그로 인해 초반 여유 있을 일정이라 생각한 부분이 많이 엉나갔었음. 그래도 함께 프로젝트를 진행하면서 팀의 단합으로 잘 해결되었고, 각자 맡은 역할을 잘 수행하는게 중요하다는 것을 알게 됨.



## 스탑워치 기능

- a. 아이폰과 맥의 OS에서 브라우저의 동작방식이 달라 JavaScript의 beforeunload의 이벤트 함수(타이머정보를 DB에 동기화함)가 동작이 안되는 문제점을 프로젝트 중간에 발견함. 또한 애플 OS의 브라우저에서 cache를 지속시키는 이유로, page가 hide된 상태에서도 SetInterval함수가 계속 발생한다는 것을 알게됨.
- b. 첫번째 해결 방법으로는 Pagehide 이벤트 함수를 사용하여, SetInterval을 클리어 했지만 시행시간이 짧아, DB에 타이머 상태를 저장할 수 없었음
- c. 타이머의 테이블에 애플사 기기의 유저 여부를 저장하는 컬럼을 생성하였고, Front에서 유저의 기기 상태를 확인하는 JavaScript명령문(`var isIOS = /Mac|iPad|iPhone|iPod/.test(navigator.userAgent);`)을 사용하여 유저 기기 정보를 저장하고, 1분마다 타이머 상태를 업데이트 하는 Interval함수를 실행함
- d. 유저의 현재 상태를 웹에서 체크하는 기능들이 있어서, 타이머정보를 업데이트 하는 기능은 웹소켓을 활용하여 더 컴팩트하게 구현할 수 있을것이라고 사료됨. 하지만, 시간과 능력 부족으로 구현할 수 없단 점에서 많은 아쉬움을 느끼고 있음.



## Toast UI

- a. 타이머 통계를 확인하기 위해 처음 선택한 라이브러리는 NHN사의 Toast UI의 Chart 라이브러리였음
- b. 하지만, 웹팩이라는 기능이 있으며 Toast UI는 웹팩 기능을 사용한다는 것을 처음 알았고, 스프링MVC와 JSP에 적용한 case가 많이 없다는 것을 발견함.
- c. 다양하고 디자인 변경이 용이한 라이브러리지만, 짧은 시간안에 웹팩에 대한 기능을 알고 프로젝트에 적용하여 사용하는 것에 어려움을 느껴 다른 라이브러리를 사용함. 미리 웹개발 분야의 다양한 기능들을 알지 못했다는 것에 아쉬움을 느낌
- d. 툭툭히 모르는 기능들을 탐색하는 시간이 필요하다는 것을 깨달음

# 감사합니다



home page



Git hub