

SPACE STORE

- 1조 밥뜨랑 -

목 차

01. 팀원소개

02. 기획의도

03. 벤치마킹

04. 프로젝트소개

05. 팀원소개

06. 개발환경

07. 개발일정

08. ERD

09. 소스코드

10. 후기

팀원 소개



박태웅

메인페이지
예약 상세페이지
캘린더 위젯
검색기능
지도 API
Main엔지니어



윤종무

호스트 페이지
주소API
Oracle 스케줄러
BE총괄



최한길

게스트 페이지
DB 설계
DB 관리
서버 관리
Project Manager



김현우

관리자 페이지
호스트 페이지
DB 설계
FE총괄

기획 의도



춤, 연극, 공연 등의 연습 공간대여에 대한 수요가 늘어남에 따라 소비자가 쉽게 본인에게 필요한 공간을 찾고 예약할 수 있는 웹페이지를 구현하고자 하였고 호스트 역시 쉽게 본인의 연습실을 등록하고 판매할 수 있도록 하고자 하였다.

벤치 마킹

SpaceCloud. 지역 또는 공간유형을 검색해보세요!

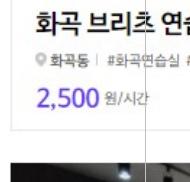
공간유형: 연습실 | 지역: 전체 | 이용일: 모든 날짜

전체 | 시간단위 | 패키지단위 | 월단위 | 지금 쿠폰 할인되는 공간 NEW

프리미엄존 ⓘ광고

- 홍대 M3스튜디오(2층5룸)**
동교동 | #개인연습실 #댄스연습 #소그룹레슨 #24시 #홍대...
2,000 원/시간 | 최대 6인 | 134 좋아요 | 1514 댓글


- 대형 댄스 촬영 스튜디오 연습실**
화곡동 | #댄스촬영 #커버댄스 #목동 #끼치산 #댄스연습실
7,000 원/시간 | 최대 25인 | 12 좋아요 | 560 댓글

- 화곡 브리츠 연습**
화곡동 | #화곡연습실 #5
2,500 원/시간 | 최대 10인 | 10 좋아요 | 10 댓글


mODO STAGE 대관, 홍보를 한 번에!

공연장 정보

위치: 서울특별시 용산구 용산동6가 168-6 국립중앙박물관
수용규모:

문의하기

문의하기

국립중앙박물관 극장 용
서울특별시 용산구 용산동6가 168-6 국립중앙박물관
국립중앙박물관 극장 용
서울특별시 용산구 용산동6가 168-6 국립중앙박물관

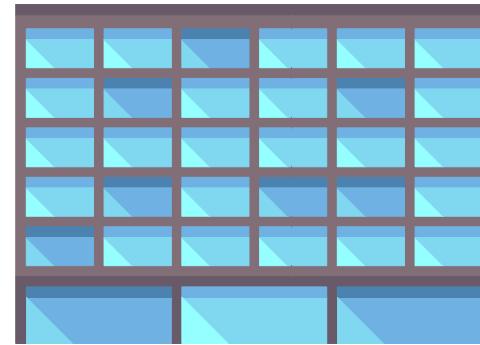


<스페이스 클라우드>
<https://www.spacecloud.kr/>
<모두의 스테이지>
<http://www.modoostage.com/venue/intro>

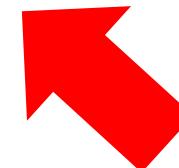
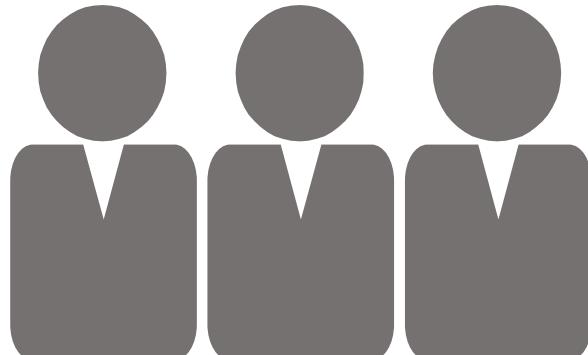
ACE STORE

프로젝트 소개

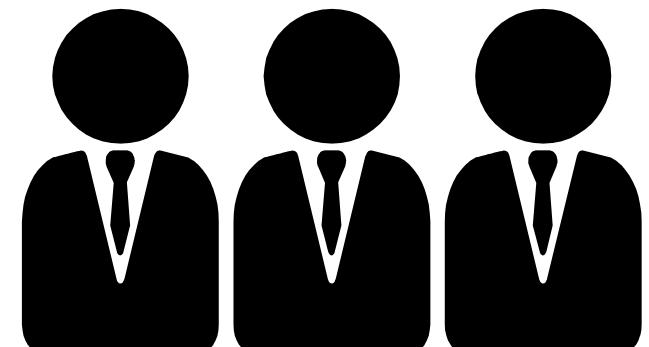
SPACE STORE



Guests



Hosts



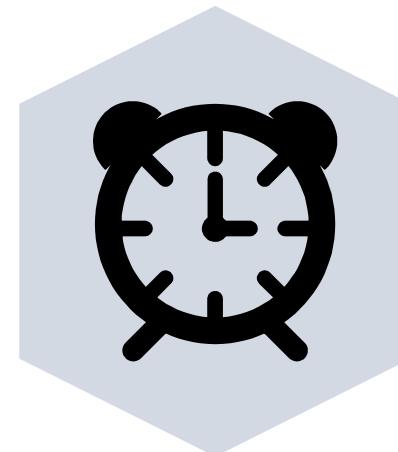
프로젝트 소개

Guest



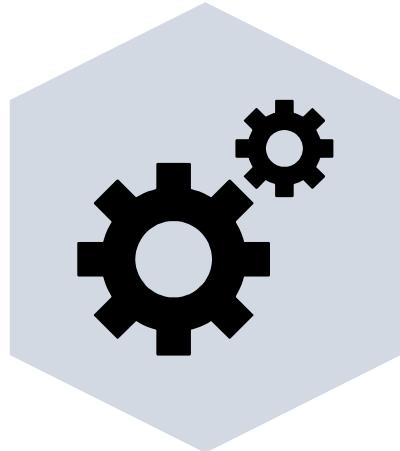
지하철, 지역 검색
구비시설 및 장비로 검색

연습실 예약
날짜별 예약가능 시간 확인



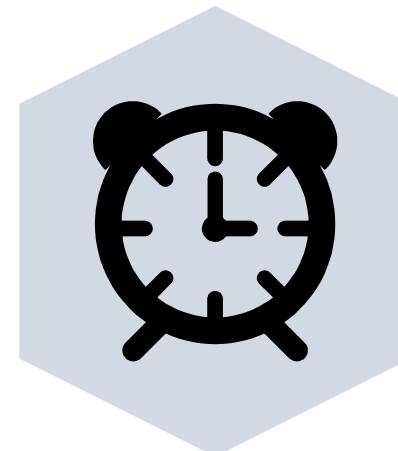
프로젝트 소개

Host



연습실 등록
연습실 관리

연습실 예약 현황파악
연습실 예약 결제



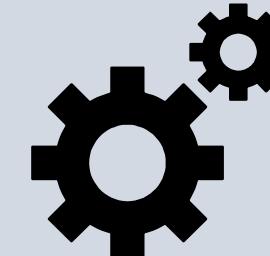
프로젝트 소개

Admin



연습실, 방, 게스트, 호스트,
예약 조회

연습실 신청 승인



개발환경



사용기술

JDK java 1.8
Servlet & JSP
Java Script
jQuery & Ajax
HTML5 & CSS3
Bootstrap4

DB

Oracle 11g

서버

Apache Tomcat v9.0

자료공유

GitHub

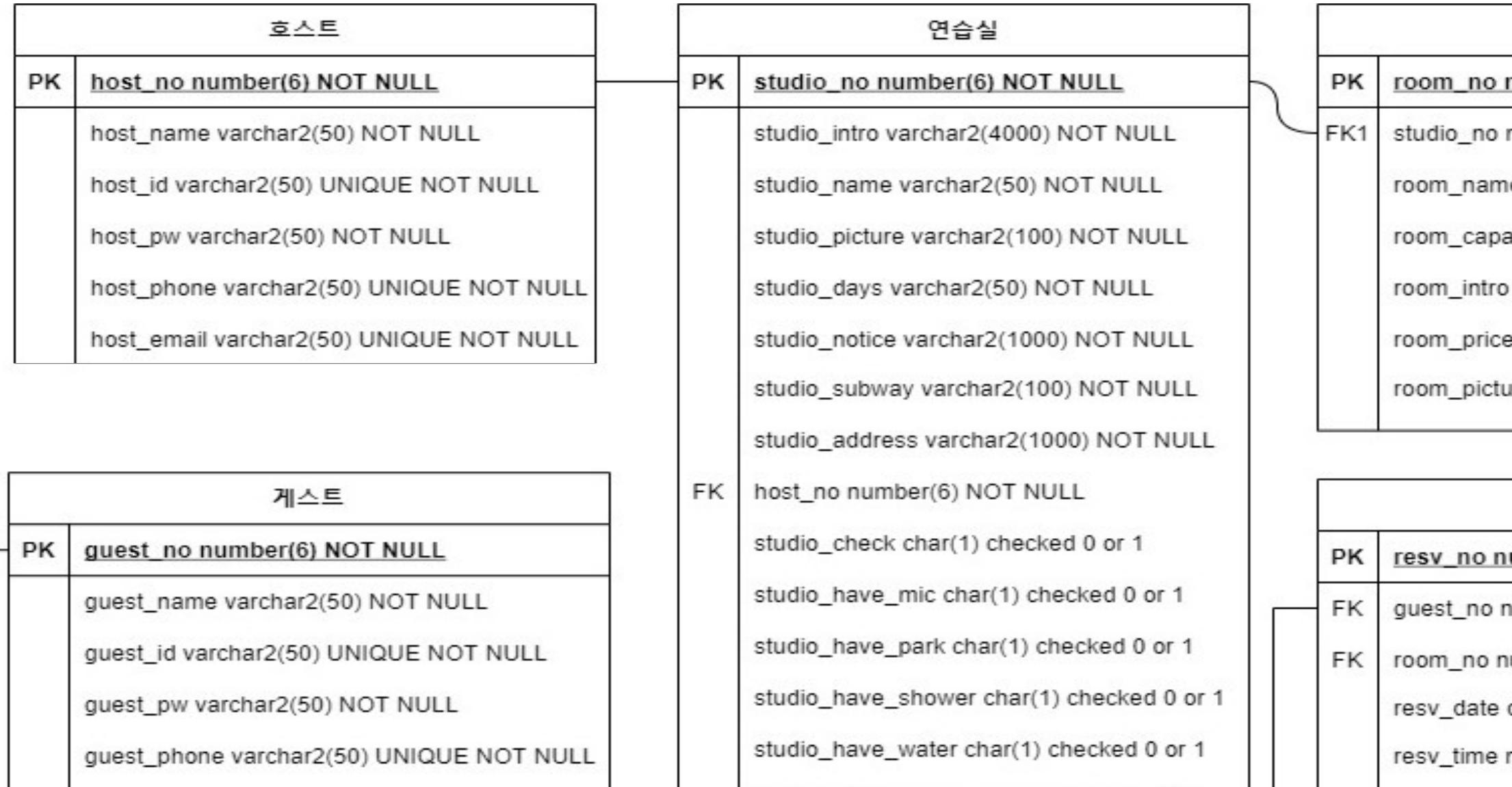
개발 일정

Github 테스트	DB & DAO 테스트	통합 테스트	
1. 요구사항분석	2. 설계	3. 프로그래밍	4. 디자인 및 최종테스트
주제 선정	DB 설계 및 구현 (test)	기능 구현	CSS, JavaScript 작성
벤치마킹 페이지 조사	ERD 작성 (test)	세부 링크 연결	최종 테스트
상세 요구 분석	서버 설계 및 구현 (test)	페이지 작성	
기능 분석			
구현기능 추가			
자료공유 Github 설정			

개발 일정

	DB & DAO 테스트	통합 테스트	
1. 요구사항분석	2. 설계	3. 프로그래밍	4. 디자인 및 최종테스트
주제 선정	DB 설계 및 ERD작성 (test)	기능 구현	CSS, JavaScript 작성
벤치마킹	서버 설계 및 구현 (test)	세부 링크 연결	최종 테스트
상세 요구 분석		페이지 작성	
기능 분석			
개발환경 설정			

ERD



애인 범색 페이지 핵심기능

메인 상세 검색 기능

지하철 : 지역 :

마이크 주차장 샤워장 정수기 에어컨 난방기 내부화장실



지역, 지하철, 옵션의 빠른 상세 검색이 가능하다.
사용자가 원하는 검색명과 옵션을 입력하면
해당 단어가 포함된 지역과 지하철을 조회하고
체크한 버튼에 따라 지정된 옵션을 조회하도록 했다.

메인 상세 검색 기능

```
String sqlResult = "select * from studios join hosts using (host_no) where 1=1 and studio_check = '1'";  
String sqlLoc="";  
String sqlSub="";  
String sqlopt="";  
  
if(!locOption.equals("")) {  
    sqlLoc += " and (studio_address like "+"%"+ "%"+locOption+"%"+")";  
}  
  
if(!subOption.equals("")) {  
    sqlSub += " and (studio_subway like "+"%"+ "%"+subOption+"%"+")";  
}  
  
if(detailOption != null) {  
    for(int i=0; i<detailOption.length; i++) {  
        int j = Integer.parseInt(detailOption[i]);  
  
        switch (j) {  
            case 1: sqlopt += " and studio_have_mic=1"; break;  
            case 2: sqlopt += " and studio_have_park=1"; break;  
            case 3: sqlopt += " and studio_have_shower=1"; break;  
            case 4: sqlopt += " and studio_have_water=1"; break;  
            case 5: sqlopt += " and studio_have_aircon=1"; break;  
            case 6: sqlopt += " and studio_have_heater=1"; break;  
            case 7: sqlopt += " and studio_have_toilet=1"; break;  
        default: break;  
    }  
}
```

각 조건문과 반복문은 값이 들어올 경우만 실행되도록 조건을 적용했다.

지역과 지하철은 값이 들어올 경우에는 사용자가 입력한 단어를 포함한 데이터를 DB에서 조회하도록 했다.

옵션은 사용자가 버튼을 체크했을 경우 버튼에 적용된 값이 넘어오고 해당 버튼 값이 의미하는 옵션을 조건문에 포함시켜 DB에서 조회하도록 했다.

상세 페이지 핵심기능

카카오 지도 API

수원연습실 MK

- 주소: 경기 수원시 팔달구 항교로 126
- 연락처: 010-1234-6554



호스트가 연습실을 등록할 때 입력한 주소가 DB에 등록되고 사용자가 연습실을 조회할 때 호스트가 등록한 주소 데이터를 사용해서 지도의 정확한 위치를 시각적으로 보여줄 수 있다.

카카오 지도 API

```

<script type="text/javascript" src="//dapi.kakao.com/v2/maps/sdk.js?appkey=806e8091967ec917e3572fad97eb1b9a&libraries=services"></script>
<script>
var mapContainer = document.getElementById('map'), // 지도를 표시할 div
    mapOption = {
        center: new kakao.maps.LatLng(33.450701, 126.570667), // 지도의 중심좌표
        level: 3 // 지도의 확대 레벨
    };
// 지도를 생성합니다
var map = new kakao.maps.Map(mapContainer, mapOption);

// 주소-좌표 변환 객체를 생성합니다
var geocoder = new kakao.maps.services.GeoCoder();

// 주소로 좌표를 검색합니다
geocoder.addressSearch('${studio.studio_address}', function(result, status) {
    // 정상적으로 검색이 완료 되었으면
    if (status === kakao.maps.services.Status.OK) {
        var coords = new kakao.maps.LatLng(result[0].y, result[0].x);

        // 결과값으로 받은 위치를 마커로 표시합니다
        var marker = new kakao.maps.Marker({
            map: map,
            position: coords
        });

        // 인포윈도우로 장소에 대한 설명을 표시합니다
        var infowindow = new kakao.maps.InfoWindow({
            content: '<div style="width:150px;text-align:center;padding:6px;">${studio.studio_name}</div>'
        });
        infowindow.open(map, marker);

        // 지도의 중심을 결과값으로 받은 위치로 이동시킵니다
        map.setCenter(coords);
    }
});
</script>

```

**연습실 위치 조회를 위한 지도 API는
<https://developers.kakao.com/>
 페이지를 통해 쉽게 사용이 가능하다**

**특별하게 수정한 부분이 있다면
 지도의 연습실 위치와 이름을 표시하기 위해**

**사용자가 선택한 연습실에 따라
 DB에서 필요한 값을
 자동으로 조회할 수 있도록 입력해두었다.**

예약페이지

- MK1 수용인원 : 20 | 가격 : 6000 /시간
- MK2 수용인원 : 15 | 가격 : 5000 /시간
- MK3 수용인원 : 25 | 가격 : 20000 /시간
- MK4 수용인원 : 20 | 가격 : 20000 /시간

일	월	화	수	목	금	토
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					

예약시간 최소 1시간부터

방번호와 날짜를 선택하시면
예약 가능한 시간 확인 가능합니다.

방, 날짜, 시간 정보를 통해 예약이 가능하다.

사용자가 연습실을 선택하면 예약 가능한 방이 상단에 표시되고
다음 칸에는 날짜를 정확하게 입력받기 위해 캘린더 위젯이 설치되어 있다.
그리고 가장 아래에는 예약 가능한 시간이 표시된다.

사용자가 날짜와 방 번호를 입력하면 화면 변동 없이
실시간으로 DB와 연동되어 조건에 맞는 예약 정보를 받아와서
사용자가 예약 가능한 시간을 손쉽게 볼 수 있도록 정보를 제공한다.

<input type="checkbox"/> 0:00	<input checked="" type="checkbox"/> 1:00	<input checked="" type="checkbox"/> 2:00	<input checked="" type="checkbox"/> 3:00
<input checked="" type="checkbox"/> 4:00	<input checked="" type="checkbox"/> 5:00	<input type="checkbox"/> 6:00	<input checked="" type="checkbox"/> 7:00
<input checked="" type="checkbox"/> 8:00	<input checked="" type="checkbox"/> 9:00	<input type="checkbox"/> 10:00	<input type="checkbox"/> 11:00
<input type="checkbox"/> 12:00	<input type="checkbox"/> 13:00	<input type="checkbox"/> 14:00	<input type="checkbox"/> 15:00
<input type="checkbox"/> 16:00	<input type="checkbox"/> 17:00	<input type="checkbox"/> 18:00	<input type="checkbox"/> 19:00
<input type="checkbox"/> 20:00	<input type="checkbox"/> 21:00	<input type="checkbox"/> 22:00	<input type="checkbox"/> 23:00

예약하기

예약페이지

```
var radioVal;
var dateVal;

function ajaxFn() {
    $.ajax({
        url : "searchByNoDate",
        data : {
            "radioVal" : radioVal,
            "dateVal" : dateVal
        },
        type : "get",
        success : function(responseData) {
            $("#resvTime").html(responseData);
        }
    });
}

function radioChk() {
    radioVal = $("input[name='roomno']:checked").val();
    console.log(radioVal);
    ajaxFn();
}

$(function() {
    $("#datepicker").datepicker({
        onSelect : function(dateText, inst) {
            dateVal = dateText;
            ajaxFn();
        }
    });
});
```

예약에는 방, 날짜, 시간, 게스트 번호 총 4가지 정보가 필요하다.
게스트 번호 이외 정보는 사용자가 입력한 정보로 구성되고
게스트 번호는 사용자가 로그인할때 저장된 세션데이터를 가져와서
예약신청에 사용된다.

예약을 위한 전체 과정은 사용자는 먼저 날짜와 방을 선택하면 입력
된 정보로 예약된 시간을 서버에서 조회하도록 요청한다.
이후에 입력된 데이터를 기준으로 예약 가능한 시간을 보여주게 되
고 사용자가 예약 시간까지 선택을 하면 예약 신청이 완료된다.

관리자 페이지 핵심기능

Main.jsp ajax 활용

SPACE STORE

전체호스트조회

전체게스트조회

전체방조회

전체연습실조회

전

비어있음

관리자 메인페이지(adminMain.jsp)

```
<body>
<jsp:include page="../common/adminHeader.jsp">/>

<nav id="nav">
  <ul>
    <li class = "menu" id="hostALL">전체호스트조회</li>
    <li class = "menu" id="guestALL">전체게스트조회</li>
    <li class = "menu" id="roomALL">전체방조회</li>
    <li class = "menu" id="studioALL">전체연습실조회</li>
    <li class = "menu" id="resvALL">전체예약조회</li>
  </ul>
</nav>
<br>
<br>
```

Main.jsp ajax 활용

관리자 메인페이지(adminMain.jsp)

클릭!!

SPACE STORE

전체모스트조회 전체게스트조회 전체방조회 전체연습실조회

호스트 전체

번호	아이디	패스워드	이름	전화번호	이메일
100	hogil2	znf123	최한길	010-8640-5078	chlgksrlf11@naver.com
101	hyk339	hyk	김현유	010-1234-6554	hyk339@naver.com
102	ptw7469	ptw351	박태웅	010-7708-7469	ilove7469@gmail.com
103	hostho	znf123	호오스	010-1234-123	qazxcv864@naver.com
104	mcyu	1234	유재석	010-5555-6554	mcyu@naver.com
105	choisanq11	1234	최상철	010-2222-5648	choisanq11@naver.com

Script

```

$(function(){
    $("#hostAll").on("click",function(){
        $("#content").load("adminSearchHost");
    });
    $("#guestAll").on("click",function(){
        $("#content").load("adminSearchGuest");
    });
    $("#roomAll").on("click",function(){
        $("#content").load("adminSearchRoom");
    });
    $("#studioAll").on("click",function(){
        $("#content").load("adminSearchStudio");
    });
    $("#resvAll").on("click",function(){
        $("#content").load("adminSearchReserve");
    });
    var work = "${work}";
    if(work=="approve"){
        // ...
    }
});

```

Main.jsp ajax 활용

관리자 메인페이지(adminMain.jsp)

클릭!!

SPACE STORE

전체호스트조회 전체게스트조회 전체방조회 전체연습실조회

번호	아이디	패스워드	이름	전화번호	이메일
100	hogil2	znf123	최한길	010-8640-5078	chlgksrlf11@naver.com
101	hyk339	hyk	김현유	010-1234-6554	hyk339@naver.com
102	ptw7469	ptw351	박태웅	010-7708-7469	ilove7469@gmail.com
103	hostho	znf123	호오스	010-1234-123	qazxcv864@naver.com
104	mcyu	1234	유재석	010-5555-6554	mcyu@naver.com
105	choisanq11	1234	최상철	010-2222-5648	choisanq11@naver.com

```
<script>
$(function(){
    $("#hostAll").on("click",function(){
        $("#content").load("adminSearchHostAll");
    });
});
```

1. 전체호스트 조회 클릭.
2. Script태그안에 함수 실행.
3. Div id="content"에 adminSearchHostAll() load.

```
$("#content").load("adminSearchRes");
});
var work = "${work}";
if(work=="approve"){
    // ...
}
```

Main.jsp ajax 활용2

SPACE STORE

전체호스트조회

전체게스트조회

전체방조회

전체연습실조회

전체예약조회

연습실 전체

번호	연습실이름	영업요일	주소	호스트아이디	등록일
111	반 어느 연습실	월화수목금토일	서울특별시 강남구 테헤란로 123	hostho	2023-10-10

Main.jsp ajax 활용2

SPACE STORE

호스트아이디	승인여부		
hostho	신청중	<input type="button" value="승인"/>	<input type="button" value="연습"/>

연습실 전체

번호	연습실이름	영업요일	주소	호스트아이디	등록일
111	반 어느 연습실	월화수목금토일	서울 강남구 테헤란로 123 14층 2호	hostho	2023-10-15

Main.jsp ajax 활용2

adminSearchStudioAll.jsp

```
<td>
<button onclick='approve(this, "${studio.stud:
승인
</button>
```

1. 승인 클릭시

2. Script태그에 approve함수실행

3. ajax실행 성공시 html을 '승인'

완료로 변경

```
<script>
function approve(obj, studio_no){
    $.ajax({
        url: "adminStudioApprove",
        data: {"studio_no": studio_no},
        success: function(responsedata){
            $(obj).parent().prev().html(
            })
    });
}
```

Main.jsp ajax 활용2

adminStudioApprove.servlet

```
protected void doGet(HttpServletRequest request, HttpServletResponse response) {
    StudioResDAO dao = new StudioResDAO();
    int studio_no = Integer.parseInt(request.getParameter("studio_no"));
    dao.updateStudioChkByNo(studio_no);
```

호스트아이디	승인여부		
jang33	승인완료	<input type="button" value="승인"/>	<input type="button" value="연습"/>
.....	<input type="button" value="승인"/>	<input type="button" value="연습"/>

dB에서 check_no를 1(승인완료)로 update

제스트 페이지 핵심기능

게스트 페이지 주요 사항

회원가입 페이지(GuestInsertServlet)

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    request.setCharacterEncoding("utf-8");
    String guest_id = request.getParameter("guest_id");
    String guest_pw = request.getParameter("guest_pw");
    String guest_name = request.getParameter("guest_name");
    String guest_email = request.getParameter("guest_email");
    String guest_phone = request.getParameter("guest_phone");

    GuestVO guest = new GuestVO(0, guest_id, guest_pw, guest_name, guest_phone, guest_email);
    StudioResDAO dao = new StudioResDAO();
    int result = dao.insertGuest(guest);
    if(result == 0) {
        response.setContentType("text/html; charset=UTF-8");
        PrintWriter writer = response.getWriter();
        writer.println("<script>alert('이미 있는 아이디 입니다.');" +
                      + "location.href='insert';</script>");
        writer.close();
    } else {
        response.sendRedirect("login");
    }
}
```

기존에 있는 아이디 일때 servlet에서 alert를 띄우기 위해 writer를 사용했습니다.
Writer.close()를 하면 sendRedirect를 할 수 없기 때문에 location.href를 사용했습니다.

게스트 페이지 주요 사항

로그인 페이지(GuestLoginServlet)

```
String before_address = request.getHeader("Referer");
session.setAttribute("before", before_address);

String scheme = request.getScheme();
String server = request.getServerName();
String port = request.getServerPort() + "";

String url = scheme + "://" + server + ":" + port;

if(session.getAttribute("guest_id")!=null) {
    if(before_address.equals(url + "/StudioRes/guest/login")) {
        response.sendRedirect("main");
        return;
    }
    response.sendRedirect(before_address);
    return;
}
```

로그인을 하였을 때 기존에 서비스 되던 화면으로 돌아가야 하기 때문에 **before_address**를 사용했습니다. Get에서 **setAttribute**를, Post에서 **removeAttribute**를 했습니다.

성격이 급해서 get 방식으로 2번 이상 호출되었을 때 **before_address**또한 **loginOk**이 때문에 무한 호출이 걸립니다. 그것에 대한 예외처리를 하였습니다.

게스트 페이지 주요 사항

로그인, 로그아웃 페이지(Login, Logout Servlet)

```
if(guest != null) {  
    session.setAttribute("guest_id", guest_id);  
    session.setAttribute("guest_pw", guest_pw);  
    session.setAttribute("guest_name", guest.getGuest_name());  
    session.setAttribute("guest_no", guest.getGuest_no());  
    session.removeAttribute("before");  
    response.sendRedirect(before);  
    return;  
}
```

```
HttpSession session = request.getSession();  
session.removeAttribute("host_id");  
session.removeAttribute("host_pw");  
session.removeAttribute("host_name");  
session.removeAttribute("host_no");  
session.removeAttribute("studiolist");  
response.sendRedirect("hostLogin");
```

저희 서비스는 하나의 세션에 guest와 host의 로그인 정보를 저장해야 합니다.
따라서 로그아웃 했을 때 세션이 유지되어야 합니다.

로그인 했을 때 세션에 guest, host의 정보를 각각 저장하고, 로그아웃 했을 때 세션에서 해당 정보를 삭제해주었습니다.
위는 guest login의 코드 조각이고 아래는 host logout의 코드 조각입니다.

호스트 페이지 핵심기능

스케줄러

만들어진 프로시저를 잡스케줄러로 시간대와 반복주기를 설정하여 해당시간에 작동하도록 하였습니다

--프로시저

```
create or replace procedure update_job_test
is
begin
update reservations set resv_check=2 where resv_date = TO_CHAR(SYSDATE , 'yyyy/mm/dd') and (resv_check=1 or resv_check=0);
commit;
end;
/
```

--스케줄러

```
begin
dbms_scheduler.create_job(
job_name =>'update_job_test1',
job_type =>'PLSQL_BLOCK',
job_action => 'BEGIN update_job_test;END;',
start_date => TO_TIMESTAMP_TZ('2021/04/26 23:50:00','yyyy/mm/dd hh24:mi:ss'),
repeat_interval => 'FREQ=DAILY'
);
end;
/
```

-----컨트롤러-----

```
begin dbms_scheduler.run_job('update_job_test1');end; --잡 시작
begin dbms_scheduler.enable('update_job_test1');end; --잡실행(사용가능)

BEGIN dbms_scheduler.disable ('update_job_test1');END; -- 일시중지
BEGIN dbms_scheduler.enable ('update_job_test1');END; -- 다시시작
```

스케줄러

The screenshot shows a Windows desktop environment. At the top, there is a blue header bar with the title "스케줄러". Below it, the main area contains two windows from SQL Server Management Studio (SSMS) showing tables of reservation data.

Left Window (Reservation Data):

RESV_NO	GUEST...	ROOM_NO	RESV_D...	RESV_TIME	RESV_CHECK
122	102	107	21/05/04		10
123	102	107	21/05/04		20
124	102	107	21/05/04		30
125	102	105	21/05/03		80
126	102	105	21/05/03		90
127	102	105	21/05/03		100
128	102	105	21/05/03		110
129	102	102	21/05/03		170
130	102	102	21/05/03		180
131	102	102	21/05/03		190
132	102	102	21/05/03		200
134	102	127	21/05/18		21

Right Window (Reservation Data):

RESV_NO	GUEST...	ROOM_NO	RESV_D...	RESV_TIME	RESV_CHECK
28	122	102	107	21/05/04	10
29	123	102	107	21/05/04	20
30	124	102	107	21/05/04	30
31	125	102	105	21/05/03	82
32	126	102	105	21/05/03	92
33	127	102	105	21/05/03	102
34	128	102	105	21/05/03	112
35	129	102	102	21/05/03	172
36	130	102	102	21/05/03	182
37	131	102	102	21/05/03	192
38	132	102	102	21/05/03	202
39	134	102	127	21/05/18	21

The taskbar at the bottom shows the date and time: "오전 9:05 2021-05-04". Red boxes highlight specific rows in both tables and the taskbar date/time.

하루가 지나자 resv_check가 0 or 1에서 2로 바뀐것을 볼 수 있습니다.

스케줄러

SPACE STORE

SPACE STORE

마이페이지

마이페이지

예약번호	연습실이름	방이름	게스트id	예약일	예약시간	현재상태	결제
125	호오오호 연습실	호	qazxcv	2021-05-03	8	결제완료	결제
126	호오오호 연습실	호	qazxcv	2021-05-03	9	결제완료	결제
127	호오오호 연습실	호	qazxcv	2021-05-03	10	예약	결제
128	호오오호 연습실	호	qazxcv	2021-05-03	11	예약	결제

오후 8:39
2021-05-03 ①

예약번호	연습실이름	방이름	게스트id	예약일	예약시간	현재상태	결제
125	호오오호 연습실	호	qazxcv	2021-05-03	8	사용완료	결제
126	호오오호 연습실	호	qazxcv	2021-05-03	9	사용완료	결제
127	호오오호 연습실	호	qazxcv	2021-05-03	10	사용완료	결제
128	호오오호 연습실	호	qazxcv	2021-05-03	11	사용완료	결제

오전 9:14
2021-05-04 ①

하루가 지나자 현재상태는 사용완료로 바뀌었습니다.

예약상태 확인 - jstl 활용

게스트센터
호스트센터

호스1님 환영합니다.
로그아웃

SPACE STORE

마이페이지
연습실등록
연습실조회

예약번호	연습실이름	방이름	게스트id	예약일	예약시간	현재상태	결제
17	연습실1	방1	guest3	2021-05-01	11	예약	결제

결제 버튼 누르면 현재상태를 바꾸기
위하여 jstl을 사용해 보았습니다.

게스트센터
호스트센터

호스1님 환영합니다.
로그아웃

SPACE STORE

마이페이지
연습실등록
연습실조회

예약번호	연습실이름	방이름	게스트id	예약일	예약시간	현재상태	결제
17	연습실1	방1	guest3	2021-05-01	11	결제완료	결제

예약상태 확인 - jstl 활용

```

<c:set var="number" value="1"/>
<c:forEach var="reservation" items="${reservationlist}">
  <tr>
    <td>${reservation.resv_no}</td>
    <td>${reservation.studio_name}</td>
    <td>${reservation.room_name}</td>
    <td>${reservation.guest_id}</td>
    <td>${reservation.resv_date}</td>
    <td>${reservation.resv_time}</td>
  <c:choose>
    <c:when test= "${reservation.resv_check==0}">
      <td>예약</td>
    </c:when>
    <c:when test= "${reservation.resv_check==1}">
      <td>결제완료</td>
    </c:when>
    <c:when test= "${reservation.resv_check==2}">
      <td>사용완료</td>
    </c:when>
    <c:when test= "${reservation.resv_check==3}">
      <td>예약취소</td>
    </c:when>
  </c:choose>
  <td>
    <button onclick="javascript:document.getElementById('studioPay${number}').submit()">결제</button>
    <form id="studioPay${number}" class="studioPay" action="resvPayByHost" method="get">
      <input type="hidden" name="resv_no" value="${reservation.resv_no}">
      <input type="hidden" name="studio_no" value="${reservation.studio_no}">
    </form>
  </td>
  <div style="display:none">${number = number + 1}</div>
</c:forEach>

```

```

StudioResDAO dao = new StudioResDAO();
int studio_no = Integer.parseInt(request.getParameter("studio_no"));
String studio_name= request.getParameter("studio_name");
request.setAttribute("studio_name", studio_name);
List<ReservationsVO> originReslist = dao.selectReservationsAll();
List<ReservationsVO> filteredReslist = new ArrayList<ReservationsVO>();
for(ReservationsVO vo:originReslist) {
  if(vo.getStudio_no()==studio_no) {
    filteredReslist.add(vo);
  }
}
request.setAttribute("reservationlist", filteredReslist);
RequestDispatcher rd;
rd = request.getRequestDispatcher("resvSearchByHost.jsp");
rd.forward(request, response);

```

카카오주소 API

```

script src="//t1.daumcdn.net/mapjsapi/bundle/postcode/prod/postcode.v2.js"></script>
function execDaumPostcode() {
    new daum.Postcode({
        oncomplete: function(data) {
            // 팝업에서 검색결과 항목을 클릭했을때 실행할 코드를 작성하는 부분.
            // 각 주소의 노출 규칙에 따라 주소를 조합한다.
            // 내려오는 변수가 값이 없는 경우엔 공백('')값을 가지므로, 이를 참고하여 분기 한다.
            var addr = ''; // 주소 변수
            var extraAddr = ''; // 참고항목 변수

            // 사용자가 선택한 주소 타입에 따라 해당 주소 값을 가져온다.
            if (data.userSelectedType === 'R') { // 사용자가 도로명 주소를 선택했을 경우
                addr = data.roadAddress;
            } else { // 사용자가 지번 주소를 선택했을 경우(J)
                addr = data.jibunAddress;
            }
            // 사용자가 선택한 주소가 도로명 태입일 때 참고항목을 조합한다.
            if(data.userSelectedType === 'R'){
                // 법정동명이 있을 경우 추가한다. (법정리는 제외)
                // 법정동의 경우 마지막 문자가 "동/로/가"로 끝난다.
                if(data.bname !== '' & /[동|로|가]/g.test(data.bname)){
                    extraAddr += data.bname;
                }
                // 건물명이 있고, 공동주택일 경우 추가한다.
                if(data.buildingName !== '' & data.apartment === 'Y'){
                    extraAddr += (extraAddr !== '' ? ' ' + data.buildingName : data.buildingName);
                }
                // 표시할 참고항목이 있을 경우, 괄호까지 추가한 최종 문자열을 만든다.
                if(extraAddr !== ''){
                    extraAddr = '(' + extraAddr + ')';
                }
                // 조합된 참고항목을 해당 필드에 넣는다.
                document.getElementById("extraAddress").value = extraAddr;
            }
            } else {
                document.getElementById("extraAddress").value = '';
            }
            // 우편번호와 주소 정보를 해당 필드에 넣는다.
            document.getElementById('postcode').value = data.zonecode;
            document.getElementById("address").value = addr;
            // 커서를 상세주소 필드로 이동한다.
            document.getElementById("detailAddress").focus();
        }
    }).open();
}

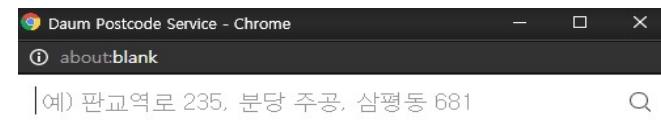
```

호스트 연습실상세페이지(studioDetailByHost.jsp)

```

<fieldset>
    <legend>주소</legend>
    <input type="text" id="postcode" placeholder="우편번호" readonly="readonly">
    <input type="button" onclick="execDaumPostcode()" value="주소 찾기"/><br>
    <input type="text" id="address" name="address" placeholder="주소" readonly="readonly"><br>
    <input type="text" id="detailAddress" name="detailAddress" placeholder="상세주소">
    <input type="text" id="extraAddress" placeholder="참고항목" readonly="readonly"/><br>
</fieldset>
<div id="inputBtn">
    <input type="submit" value="등록하기"/>
    <input type="reset" value="취소"/>
</div>

```



tip

아래와 같은 조합으로 검색을 하시면 더욱 정확한 결과가 검색됩니다.

- 도로명 + 건물번호
예) 판교역로 235, 제주 첨단로 242
- 지역명(동/리) + 번지
예) 삼평동 681, 제주 영평동 2181
- 지역명(동/리) + 건물명(아파트명)
예) 분당 주공, 연수동 주공3자
- 사서함명 + 번호
예) 분당우체국사서함 1~100

StudioInsert.jsp 체크박스데이터처리

시설안내

マイク □ 주차장 □ 샤워실 □ 정수기 □ 에어컨 □ 히터 □ 화장실 □

StudioInsert.jsp

```
<fieldset>
<legend>시설안내</legend>
マイク<input type="checkbox" name="have" value="studio_have_mic">
주차장<input type="checkbox" name="have" value="studio_have_park">
샤워실<input type="checkbox" name="have" value="studio_have_shower">
정수기<input type="checkbox" name="have" value="studio_have_water">
에어컨<input type="checkbox" name="have" value="studio_have_aircon">
히터<input type="checkbox" name="have" value="studio_have_heater">
화장실<input type="checkbox" name="have" value="studio_have_toilet">
</fieldset>
```

배열 넘어감

〈연습실 등록시 체크박스 DB에 입력하기〉

Check 박스의 name을 같게 하고 submit을 하면 배열로 넘어갑니다.

Value는 DB상의 칼럼이름으로 넣어줍니다.

//부대시설 체크

```
String[] facility = multipartRequest.getParameterValues("have");
Map<String, String> facilityChk = new HashMap<String, String>();
facilityChk.put("studio_have_mic", "0");
facilityChk.put("studio_have_park", "0"); (DB상에 0은 체크안함, 1은 체
facilityChk.put("studio_have_shower", "0");
facilityChk.put("studio_have_water", "0");
facilityChk.put("studio_have_aircon", "0");
facilityChk.put("studio_have_heater", "0");
facilityChk.put("studio_have_toilet", "0");
```

StudioInsertServlet.java

```
if(facility != null) {
    for(String s:facility) {
        if(facilityChk.containsKey(s)) {
            facilityChk.replace(s, "1");
        }
    }
}
```

facilityChk.containsKey(s)

facilityChk Map의 키값 중
s가 있는지 판단합니다.

Map으로 저장된 키값과 배열로 넘어온 값을 비교하여 같은 경우
값을 1로 고쳐줍니다.

StudioInsert.jsp 파일 저장

<StudioInsert.jsp>

```

form id="studiofrm" action="studioInsert" method="post" enctype="multipart/form-data">
<fieldset>
<legend>연습실 등록</legend>
<label>연습실 File 타입을 보낼때는 enctype='multipart/form-data' 를 사용합니다.
<input type="file" name="studio_picture"><br>

```

MultipartRequest를 생성하면 파일이 경로에 저장이 됩니다

DB에는 파일이름을 저장해주기 위하여 파일명을 알아왔습니다

15	연습실15	사진15
16	test1	%25B4%25F5_%25B1%25D7%25B7%25B9%25C0%25CC%25BD%25BA_%25BD%25BA%25C6%25A9%25B5%25F0%25BF%25C0_3.jpg
17	test12	01%25C8%25A6004.jpg

```

<div id="imagesizeDiv">
  <c:set var="pPath" value="${pageContext.request.contextPath}" />
  
</div>

```

이미지를 불러오기 위하여 경로와 파일 이름을 가져옵니다
\${pageContext.request.contextPath} >> /StudioRes

C:\eclipse-workspace\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps\StudioRes\imageUpload

<StudioInsertServlet.java>

```

//사진업로드
String upload_dir = "imageUpload";
int size = 1024*1024*10;
//서버경로
String path = getServletContext().getRealPath(upload_dir);

```

```

MultipartRequest multipartRequest = new MultipartRequest(request, path, size, "ut"
    new DefaultFileRenamePolicy());
Enumeration files = multipartRequest.getFileNames();
String str = (String)files.nextElement();
String originalFileName = multipartRequest.getOriginalFileName(str);

```

Q & A

감사합니다