SKALA

Mini Project

웹 서비스 개발

2025.5

SK 주식회사 AX

웹 서비스(Web Service)

정의 & 개념

- **인터넷을 통해 서로 다른 시스템 간에 데이터를 주고받고 기능을 제공하는 기술 및 아키텍처**
- **주로 클라이언트(Client)의 요청에 의해 서버(Server)가 데이터를 제공하는 방식의 서비스**
- **HTTP, XML, JSON, SOAP, REST API 등의 기술을 사용하여 구현**

특징

- **네트워크를 통해 접근 가능**
- **표준 프로토콜(HTTP{S})과 데이터 형식(JSON, XML) 사용**
- **서로 다른 플랫폼(Windows, Mac, Linux)과 프로그램 언어 간 호환성**
- **클라이언트-서버 모델**

종류

- **RESTful API (Representational State Transfer)**
  - REST 원칙을 따르며, HTTP 메서드(GET, POST, PUT, DELETE)를 활용
- **SOAP (Simple Object Access Protocol)**
  - XML을 기반으로 한 메시징 프로토콜, 레거시 엔터프라이즈 시스템에서 주로 사용
- **GraphQL**
  - 클라이언트가 원하는 데이터를 직접 지정하여 요청할 수 있는 쿼리 언어, 페이스북에서 개발

프로젝트 진행 순서

서비스 기획 및 기능(요구사항) 정의

- 작업 목표 : 웹 서비스의 목표와 주요 기능 정의
  - 서비스 개요 및 배경
  - 요구 사항 / Actor별 기능 목록 정리
  - UI 흐름도

시스템 설계 및 데이터 모델링

- 작업 목표 : 데이터 구조 설계 및 API 명세화
  - 데이터 모델링 (ERD 작성): 주요 엔티티(Entity) 및 관계 정의
  - API 설계 (RESTful API): API 명세 작성

과정 범위

어플리케이션 개발

- 작업 목표 : Backend 서비스 기능 구현
  - 프로젝트 패키지 구성
  - 데이터 엔터티 및 레파지토리 구성
  - 서비스 로직 및 컨트롤러 개발

테스트 및 디버깅

- 작업 목표 : 서비스 안정성 검증 및 버그 수정
  - 단위/통합/성능/보안 테스트 → API 테스트 위주로 진행

프로젝트 수행 일정 가이드 라인

1일차

- 서비스 기획/요구사항 분석
  → UI 흐름도,
  요구사항 정의서 작성

서비스 구축 방향

2일차

- 시스템 설계/데이터 모델링
  → ERD 및 API 명세서 작성

서비스 설계

3일차

- 프로젝트 정의서
  (서비스 기획 및 시스템 설계)
  및 API 명세서 보완
- ***오후 2시~5시***
  → 프로젝트 수행 결과 발표
  (개인별 5분)

프로젝트 착수 준비

웹 서비스 프로젝트 정의서 (목차)

서비스 개요	• 서비스 이름, 서비스 목적, 주요 기능 요약 및 핵심 가치(사용자 문제 해결)
시스템 액터 정의	• 주요 액터 식별(일반 사용자, 관리자, 외부 시스템 등), 액터별 주요 역할 및 행위
화면 및 UI 와이어프레임	• 주요 화면 목록(로그인, 메인화면, 상세화면 등), 각 화면의 와이어프레임 (스케치/툴 무관)
데이터 모델 설계	• 주요 개체(Entity) 정의, Entity 간 관계 (ERD), 속성 목록 (각 Entity의 주요 속성 및 설명)
API 명세 정의	• API 목록 (OAS 기반, 각 API의 입력 및 출력 예시)
고려사항	• 설계 상의 고민, 다음 단계 개발 시 고려 사항

평가 기준

납기 (30점)

- **3일차 오후 2시 마감 – 미준수 0점**

목표 시스템 정의 (20점)

- **기능 명확성(O), 창의/독창성(X)**
  - 서비스 개요 및 배경
  - 사용자(actor) 별 시스템 기능(요구사항) 정의
  - 서비스 UI 흐름 – 제공하고자 하는 전체 화면 와이어프레임

시스템 설계 (40점)

- **데이터 모델 (20점)**
  - API를 제공하기 위해 필요한 모든 데이터
- **API 명세 (20점)**
  - 화면을 구현할 때 호출되는 API 전체 목록 (요청/응답/예외처리, **Security Schemes 제외**)

발표 (10점)

- **전달 능력, 시간 엄수**

✓산출물

프로젝트명_O반_이름_정의서 (서비스 개요 및 기능, UI 흐름, ERD) – PDF 또는 PPT

프로젝트명_O반_이름_API – YML 또는 JSON

활용 도구

구분		설명
UI 흐름도	와이어프레임	Balsamiq: https://balsamiq.com
	AI 기반 UI 설계	creatie: https://creatie.ai
	파워포인트	Microsoft Powerpoint
ERD		dbdiagram.io: https://dbdiagram.io/home
		DBeaver
API명세서		Swagger Editor : https://editor-next.swagger.io OAS Editor: https://oas-editor.web.app (불안정)

[참고] OAS (OpenAPI Specification)

개념

"RESTful API를 기술하고 문서화하기 위한 표준 포맷"

- 예전 이름: Swagger Specification
- 주요 형식: YAML 또는 JSON
- 현재 버전: OpenAPI 3.x (3.0.0, 3.0.3, 3.1.0 등)

목적

- **API 명세를 명확히 하여 팀 간 협업 효율화**
- **API 문서 자동 생성 및 테스트 도구 활용 가능**
- **서버/클라이언트 코드 자동 생성**
- **Mock 서버 및 검증 자동화 기반 구축**

[참고] OAS 기반 API 명세서

OAS (OpenAPI Specification)

RESTful API를 정의 하는 표준 형식

- OAS History
  - Swagger (2010, Tony Tam)
    → SmartBear 가 인수
  - OpenAPI Initiative (2015)
    : Google, Microsoft, IBM, Adobe,
    Red Hat 등에 의해 추진된
    OAS 표준화를 위한 프로젝트
  - 2017년 OpenAPI Specification 3.0 발표

【 Open API vs OpenAPI 】

Open API	OpenAPI
개방형API (ex.공공데이터조회)	**OpenAPI Specification** **RESTful API**에 대한 명세서를 작성하는 표준 규격

【 OAS vs Swagger 】

OAS	Swagger
OPEN API INITIATIVE **OAI**에 의해 추진된 **RESTful API** 표준규격	Swagger Supported by SMARTBEAR **SmartBear** 에서 제공 하는 **OAS** 작성 도구 (Swagger Editor, Swagger UI, Swagger Codegen 등)

[참고] OAS (OpenAPI Specification) 필요 이유

【 전통적인 API 문제점 】

- 문서가 항상 최신이 아님
- API 응답 구조가 다르게 구현
- 프론트엔드-백엔드 협업 지연

【 OAS 기반의 장점 】

- 문서화와 코드 일관성 유지
- 명세 기반 자동화 (코드 생성, 문서, Mock 서버 등)
- API 테스트 자동화 및 오류 감소
- 다양한 도구와 생태계 연계

[참고] OAS (OpenAPI Specification)

OAS 문서 구조

```yaml
openapi: 3.0.0
info:
  title: API 이름
  version: 버전
servers:
  - url: https://example.com/v1
paths:
  /users:
    get:
      summary: 사용자 조회
      responses:
        '200':
          description: 성공
components:
  schemas:
    User:
      type: object
      properties:
        id:
          type: integer
        name:
          type: string
```

【 주요 섹션 】

구분	설명
openapi	명세서 버전 (예 : 3.0.0)
info	API 정보 (제목, 설명, 버전 등)
servers	API 호출이 이루어질 기본 URL
paths	각 앤드포인트에 대한 설명 (GET /users, POST /users/{id} 등)
components	재사용 가능한 객체 정의 공간 (schemas, parameters, reponses 등)
schemas	객체 데이터 구조 정의

[참고] OAS (OpenAPI Specification)

【 HTTP 메서드별 구조 표현 】

메서드	설명
get	데이터 조회
post	새로운 리소스 생성
put	전체 리소스 갱신
patch	리소스 부분 업데이트
delete	리소스 삭제

각 메서드는 OAS 내 paths 아래 다음과 같이 정의

```
/users/{id}:
  get:
    summary: 특정 사용자 조회
    parameters:
      - name: id
        in: path
        required: true
        schema:
          type: integer
    responses:
      '200':
        description: 사용자 반환 성공
```

[참고] OAS (OpenAPI Specification)

【 데이터 타입 및 구조 정의 (schemas) 】

타입	설명	예시
string	문자열	이름, 이메일
integer	정수	나이, 아이디
boolean	True/False	상태값
array	리스트	태그 리스트 등
object	복합 구조체	사용자, 주소 등

예시 – 배열 포함 구조

```
UserList:
  type: array
  items:
    $ref: '#/components/schemas/User'
```

[참고] OAS (OpenAPI Specification)

【 파라미터 정의 방식 】

위치	키워드	설명
경로	in: path	URL 변수(/users/{id} 등)
쿼리	in: query	/users?age=20
헤더	in: header	사용자 인증 등의 헤더
바디	requestBody	POST/PUT 요청의 본문

예시

```
parameters:
  - name: age
    in: query
    schema:
      type: integer
    required: false
```

[참고] OAS (OpenAPI Specification)

【 Swagger 생태계 도구 】

도구	설명
Swagger Editor	YAML 기반 API 문서 작성 및 미리보기
Swagger UI	API 스팩 기반 인터랙티브 문서
Swagger Codegen	API 명세서로 서버/클라이언트 코드 자동 생성
OpenAPI Generator	Swagger Codegen 확장판, 언어/프레임워크 다양
SwaggerHub	API 명세 협업 플랫폼 (SaaS)

[참고] OAS (OpenAPI Specification) - 백엔드 프레임워크 연계

【 FastAPI (Python) 예시 】

- **FastAPI는 OAS 기반 문서 자동 생성을 지원**
- **/docs → Swagger UI 제공**
- **/redoc → Redoc 문서 제공**

```python
@app.get("/items/{item_id}", response_model=Item)
def read_item(item_id: int):
    ...
```

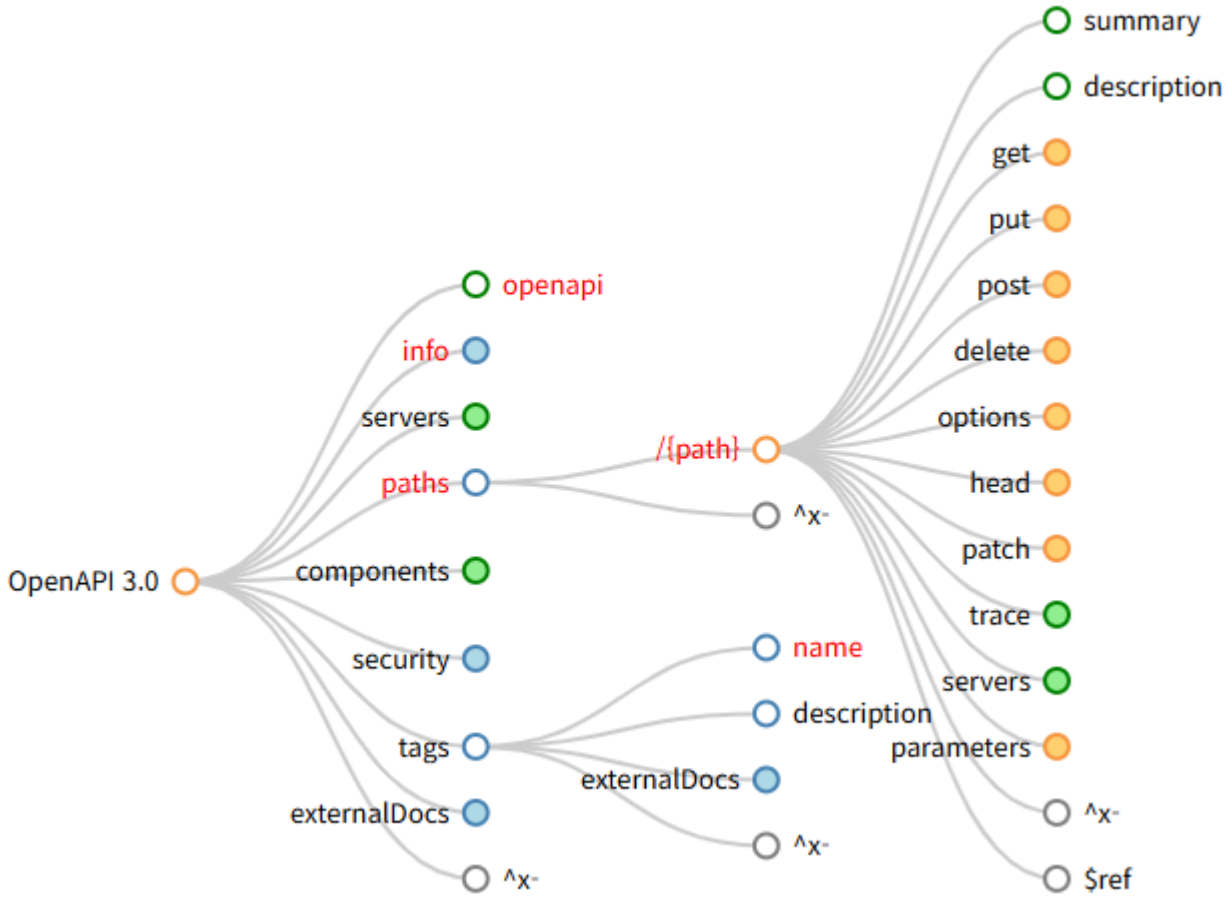
이 코드만으로도 자동으로 Swagger 문서가 생성

【 SpringBoot 예시 】

```xml
<dependency>
    <groupId>org.springdoc</groupId>
    <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
    <version>2.8.11</version>
</dependency>
```

- Spring Boot 3.0 이상 버전에서는 springdoc-openapi 라이브러리를 사용하여 API 문서 자동화 지원
- OpenAPI 3 명세 기반의 문서와 Swagger UI를 자동 생성

[참고] OAS 컴포넌트 맵



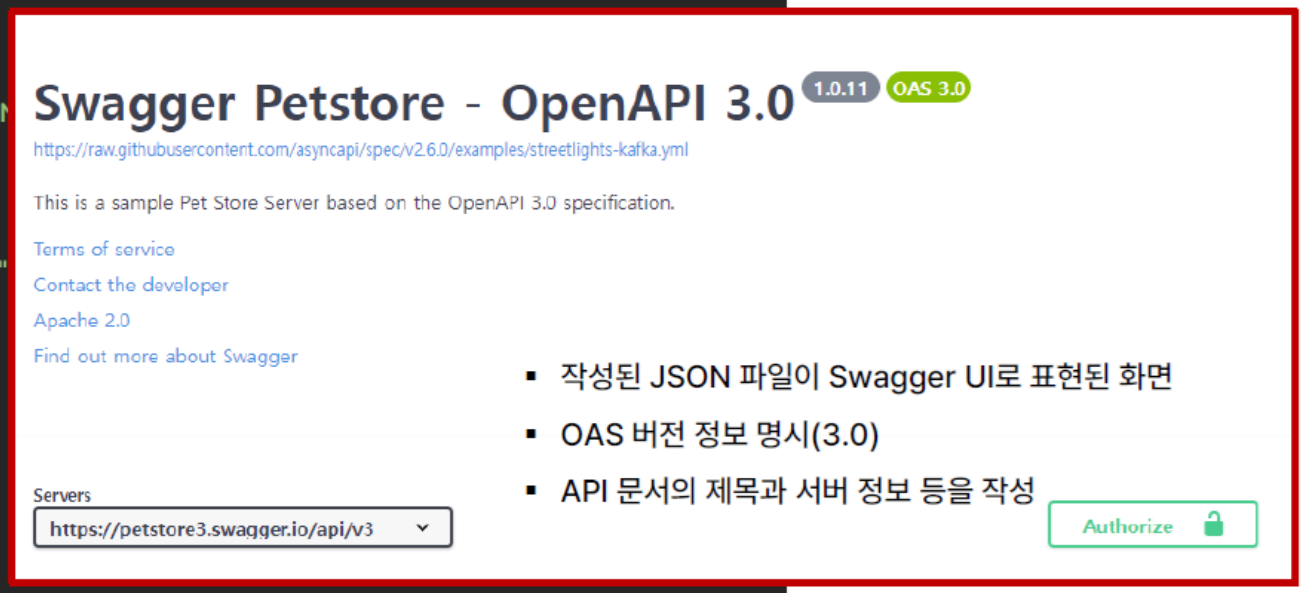
https://openapi-map.apihandyman.io/

[참고] OAS 기반 API 명세서 구조

- **OAS API 명세서는 YML 또는 JSON 파일로 작성 :**

  Text Editor(IDE에 플러그인)로 작성할 수 있으며 Swager UI로 확인 가능


```
{
 "openapi": "3.0.3",
 "info": {
  "title": "Swagger Petstore - OpenAPI 3.0",
  "description": "This is a sample Pet Store Server based on the OpenAPI 3.0 specification.",
  "termsOfService": "http://swagger.io/terms/",
  "contact": {
   "email": "apiteam@swagger.io"
  },
  "license": {
   "name": "Apache 2.0",
   "url": "http://www.apache.org/licenses/LICEN
  },
  "version": "1.0.11"
 },
 "externalDocs": {
  "description": "Find out more about Swagger"
  "url": "http://swagger.io"
 },
 "servers": [
  {
   "url": "https://petstore3.swagger.io/api/v3"
  }
 ],
...
}
```

https://openapi-map.apihandyman.io/

[참고] OAS 실무 활용 시나리오

【 실무 활용 시나리오 】

활용 상황	기대 효과
Front-End/Back-End 협업	명세 기반 개발로 의사소통 개선
팀원 교체/온보딩	잘 정리된 문서로 빠른 이해 가능
API 변경 시 버전 관리	/v1, /v2 등 서버 정의로 안정적 이전
자동테스트	명세 기반 테스트 스크립트 생성 가능
문서 자동화	실시간 API 미리 보기 및 문서화 지원

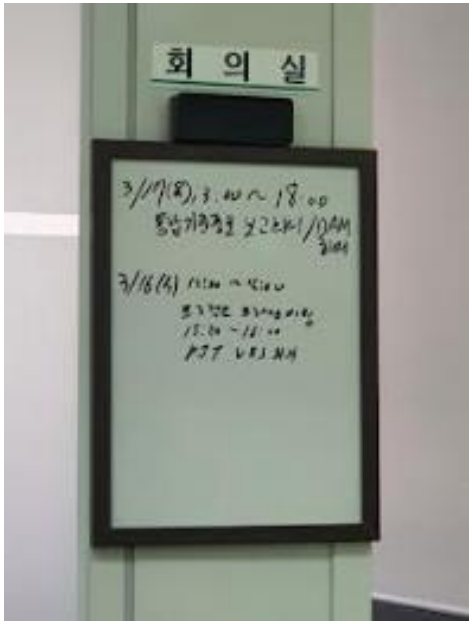
https://openapi-map.apihandyman.io/

[예시] 회의실 예약 서비스

(https://sk-rent-a-room.web.app)

회의실 예약 서비스 개요 및 목적

- **프로젝트 수행을 위한 건물이 임시 대여 공간**인 경우, **사내 회의실 예약 시스템의 범위에 포함되지 못함**.
- 회의실 예약을 위해 **화이트보드에 수기 작성** 또는 **협업 도구의 페이지에 공지를 통해 회의실 예약**.
- **고객사, 수행사, 개발자가 함께 접근 가능한 QR코드 기반의 회의실 예약 서비스**를 제공.

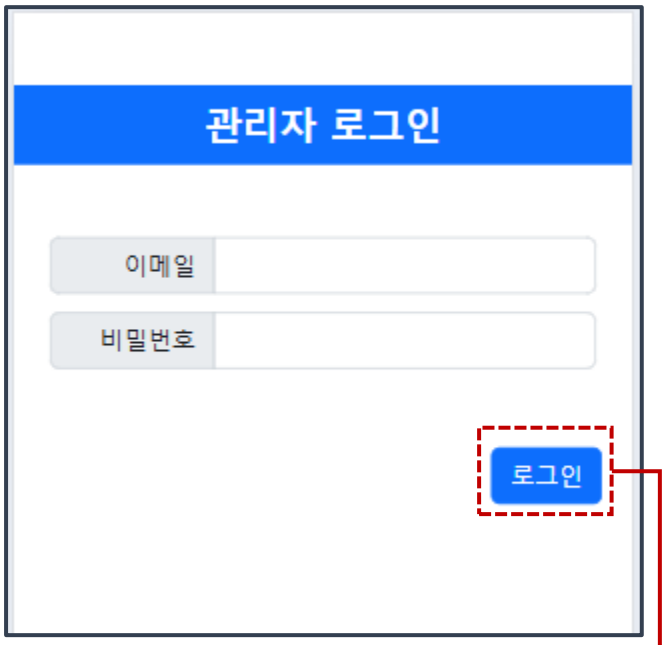


시스템 액터 정의

Actor 구분	주요 역할/행동	상세 설명
서비스 관리자	- 회원 가입 / 로그인 - 건물 관리 - 회의실 관리	- 회의실이 위치한 건물을 추가/삭제/변경 - 각 건물에 속한 회의실을 등록 및 수정 - 회의실 정보는 이름, 좌석 수, 보유 장비 포함
서비스 사용자	- QR코드로 로그인 없이 접근 - 회의실 예약 현황 조회 - 예약 추가/수정/취소	- 제공된 QR코드를 통해 회의실 예약 현황을 조회 - 건물의 모든 회의실 현황 조회 및 예약 관리 가능 - 회의실 예약 추가/수정/취소 (예약 수정/취소를 위해 PIN 설정 필요) - 현재 시간 이전 예약은 변경 불가

서비스 관리자 UI 흐름도 – 회원가입/로그인

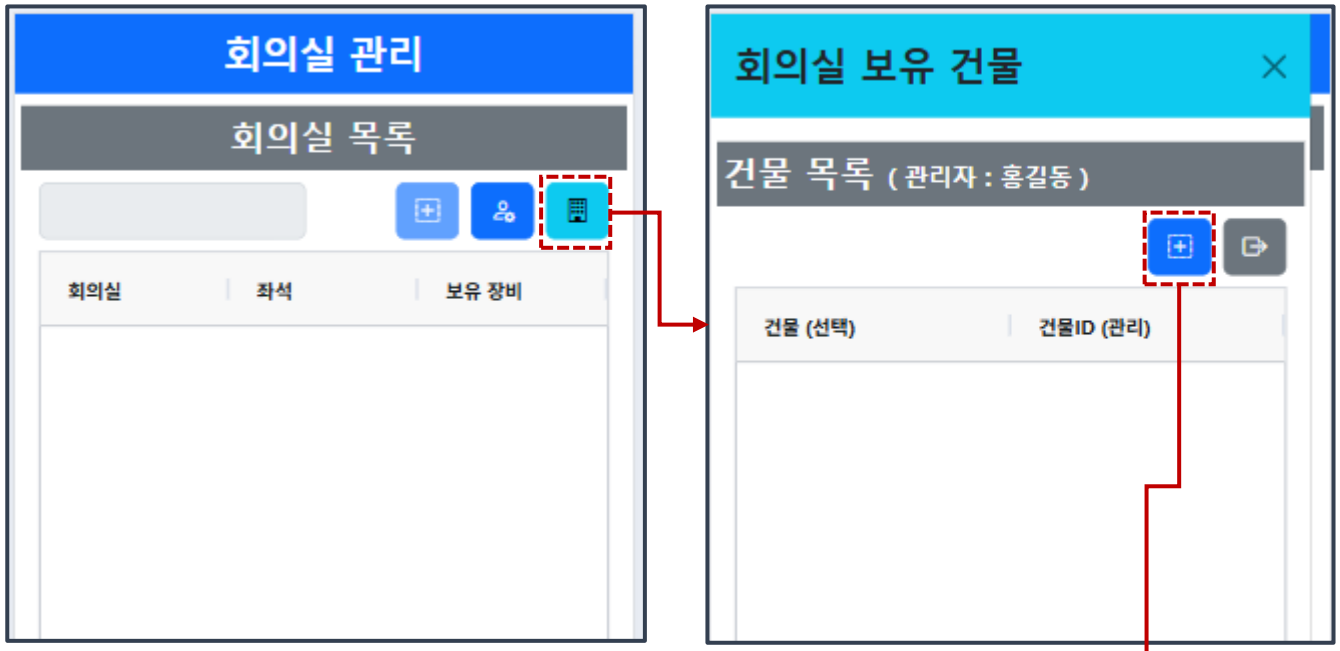
- 회원 가입 필요 정보 : 이름, 이메일, 비밀번호
- 비밀번호는 6자 이상 동일하게 입력
- 회원가입 버튼을 눌러 가입 진행



- 로그인에 필요한 정보는 이메일, 비밀번호
- 로그인 버튼을 눌러 서비스 진입

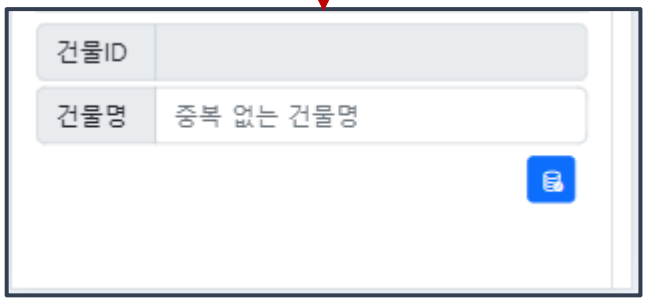
"회의실 관리" 화면으로 이동

서비스 관리자 UI 흐름도 – 회의실 관리 (건물)



- 관리자가 로그인하면 회의실 관리 화면으로 진입.
- 관리자가 등록한 건물이 있는 경우, 첫번째 등록 건물의 회의실 목록이 노출된다.
- 건물 아이콘 버튼을 클릭하면 건물 목록 관리 팝업이 노출되어 신규 건물을 등록할 있다.

❖ 한 명의 관리자는 다수의 건물을 등록할 수 있고, 하나의 건물은 다수의 회의실을 등록할 수 있다.

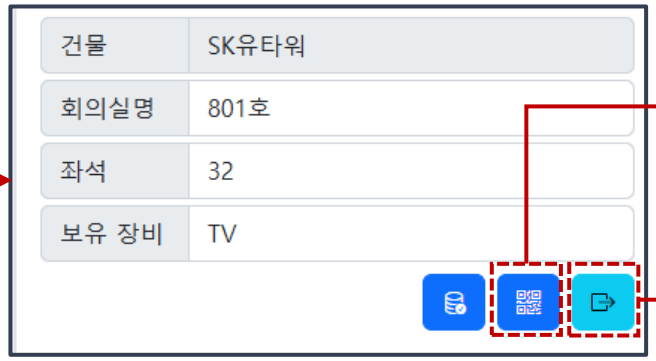


- 건물 목록 관리 팝업에서 + 아이콘 버튼을 클릭하면 건물명을 입력할 수 있는 패널이 하단에 노출된다.
- 중복되지 않는 건물명을 입력하고 저장 버튼을 클릭하면 건물ID가 생성

서비스 관리자 UI 흐름도 – 회의실 관리 (회의실)

- 건물을 선택하면 건물의 회의실 목록이 노출된다.
- 상단 플러스 버튼을 클릭하면 목록 하단에 새로운 회의실을 등록할 수 있는 패널이 노출된다.
- 목록에 있는 회의실을 클릭하면 하단에 해당 회의실 정보를 변경할 수 있는 패널이 노출된다.

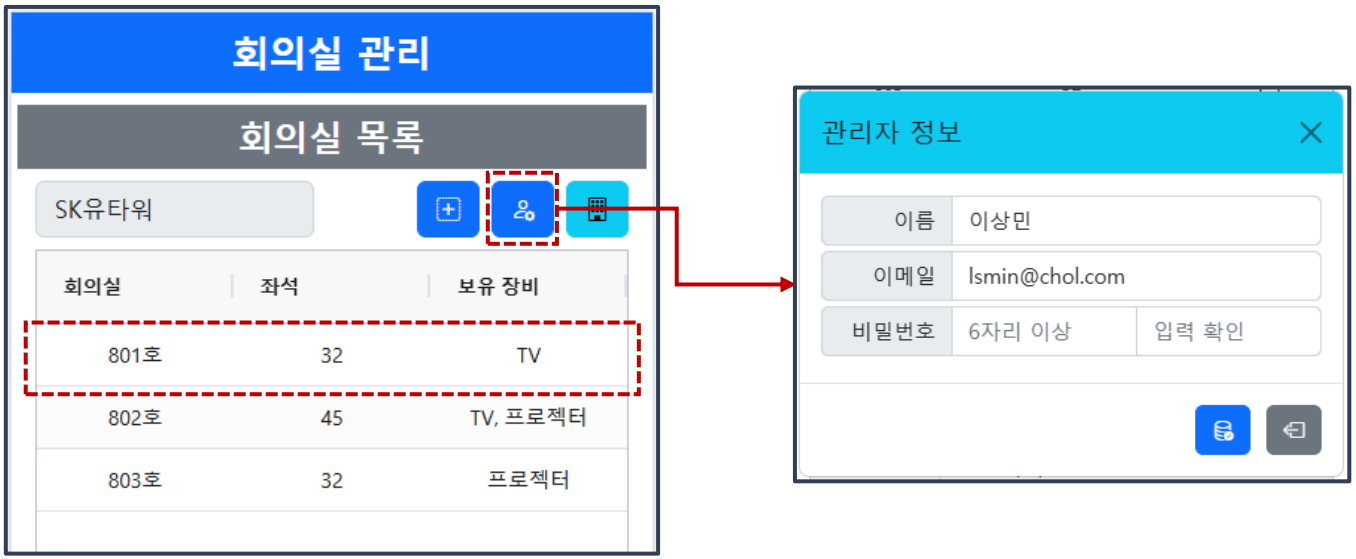
- 회의실 명, 좌석 수, 보유 장비를 입력하고 저장 버튼을 클릭한다.



- 회의실 정보를 수정 후 저장 버튼을 클릭한다.
- QR코드 버튼을 클릭하여 회의실 QR 코드를 출력한다.

"회의실 예약" 화면으로 이동

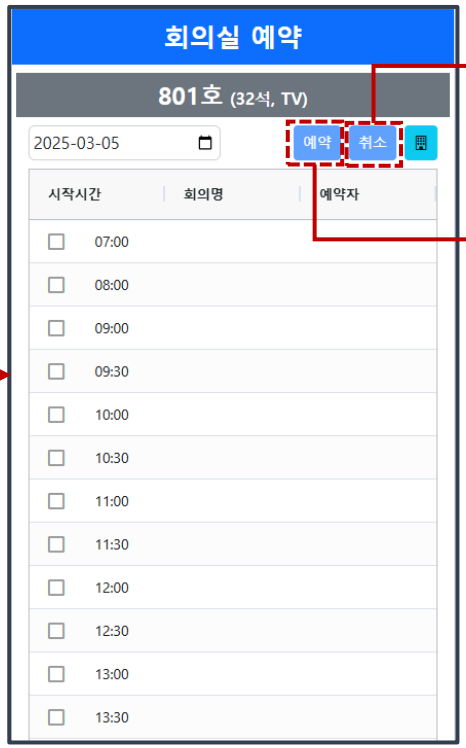
서비스 관리자 UI 흐름도 – 회의실 관리 (관리자 정보)



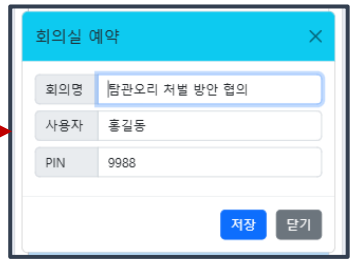
- 계정 아이콘 버튼을 클릭하면 관리자 정보 팝업이 노출된다.
- 비밀번호를 변경할 경우
  새로운 비밀번호를 입력하고 저장 버튼을 클릭

서비스 사용자 UI 흐름도 – 회의실 예약

- 회의실 QR코드 또는 URL을 통해 회의실 예약 화면으로 이동



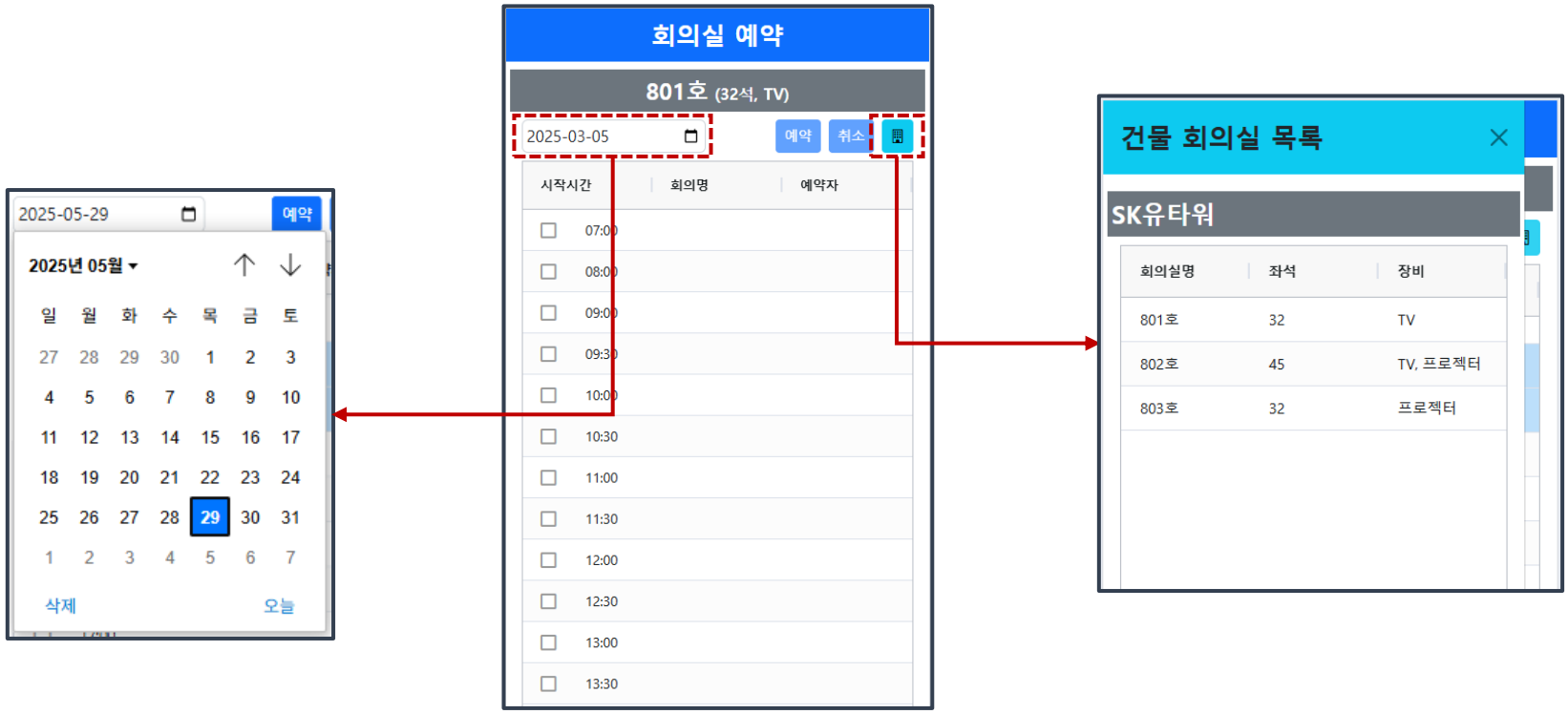
- 회의실 예약 시작 시간(30분 단위 슬롯)을 선택하면 예약 버튼이 활성화 됨.
- 예약 버튼을 클릭하면 회의명, 사용자, 예약 변경을 위한 PIN을 입력할 수 있는 팝업을 노출



- 회의 정보와 PIN을 입력하고 저장 버튼을 클릭
- 해당 슬롯에 예약된 회의가 있는 경우 오류 팝업을 노출

❖ 예약 편의성을 위해 사용자와 PIN은 브라우저 localStorage를 통해 캐시 처리

서비스 사용자 UI 흐름도 – 일자 및 회의실 변경



- 예약 일자 변경은 날짜 선택창을 클릭하여 선택한다.
- 다른 회의실은 건물 아이콘 버튼을 클릭하여 목록에서 선택한다.

데이터 모델 설계

서비스 관리자 - 건물 및 회의실 관리 권한

회의실이 존재하는 건물

건물 내 회의실 정보

보유 장비 종류 예: TV, 프로젝터 등

회의실의 보유 장비 (Many-to-Many)

회의실 예약 정보

API 명세 정의

```yaml
openapi: 3.0.0
info:
  title: SK-RENT-A-ROOM 회의실 예약 서비스 API
  description: SK-RENT-A-ROOM 웹 서비스 개발을 위해 작성된 회의실 예약 백엔드 API.
servers:
  - url: http://localhost:8080/api
    description: DEV
tags:
  - name: admins
    description: 회의실 관리자
  - name: bookings
    description: 예약
  - name: buildings
    description: 건물
  - name: rooms
    description: 회의실
paths:
  /admins/signup:
    post:
      tags:
        - admins
      summary: 관리자 회원가입
      description: 관리자 회원가입
      operationId: postAdminsSignup
      requestBody:
        description: 관리자 정보
        required: false
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/Admin"
      responses:
        "200":
          description: 200 OK - Response Success
```