[Project Final Presentation]

SKKU MAP

: A Solution for Efficient Building and Classroom Navigator

Team 11

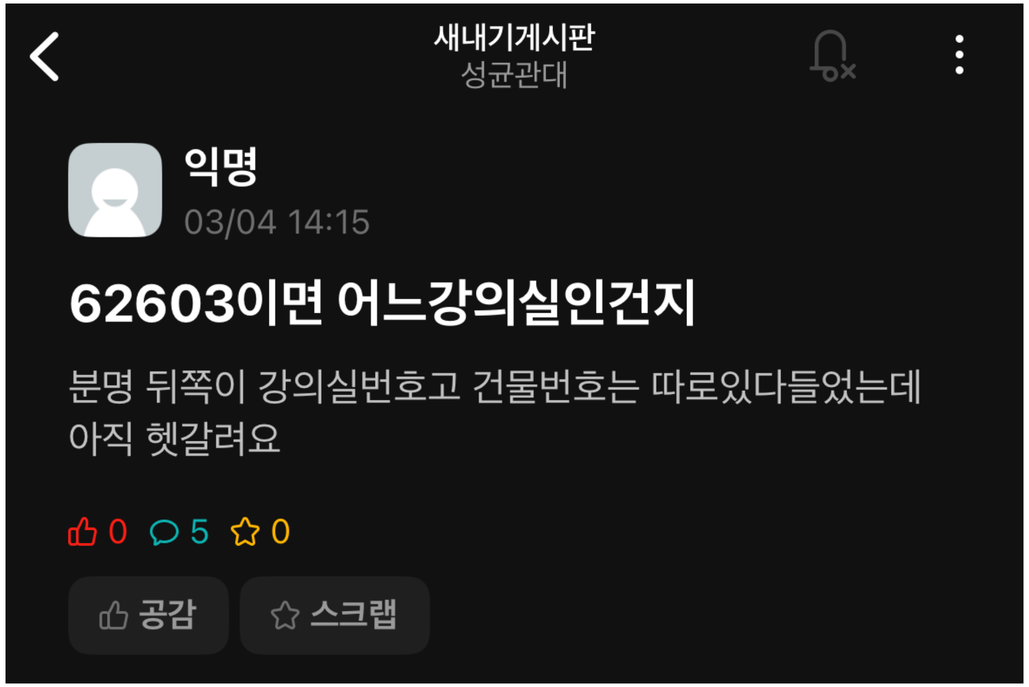
2020312552 김민수　2021315064 나리만　2019312522 박정은　2020313266 이주형　2022314221 정승훈

INDEX

Section 1

Motivation

01_Motivation



새내기게시판
성균관대

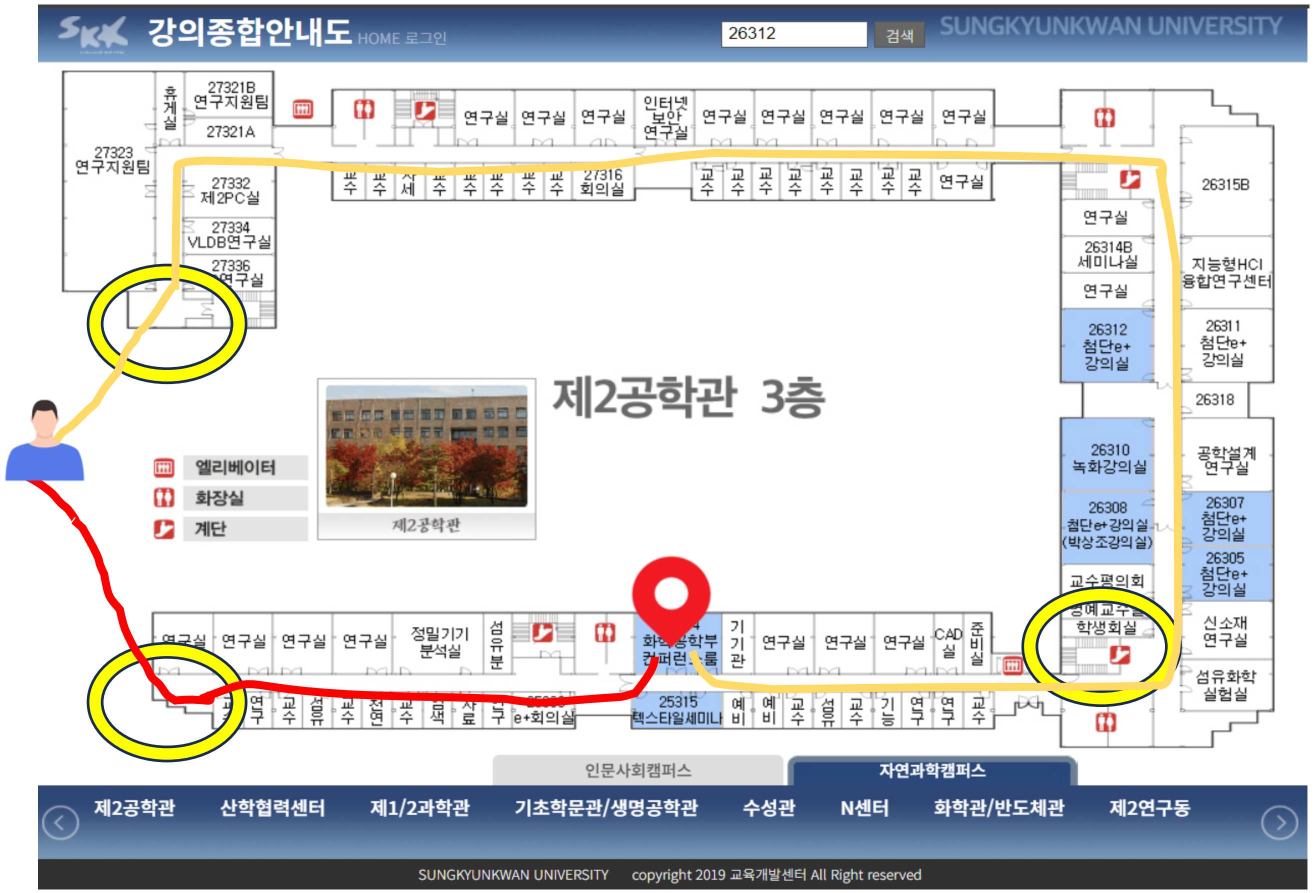
익명
03/04 14:15

626030|면 어느강의실인건지

분명 뒤쪽이 강의실번호고 건물번호는 따로있다들었는데
아직 헷갈려요

0 5 0

공감 스크랩



Section 2

Goals & Overview

02_Goals & Overview

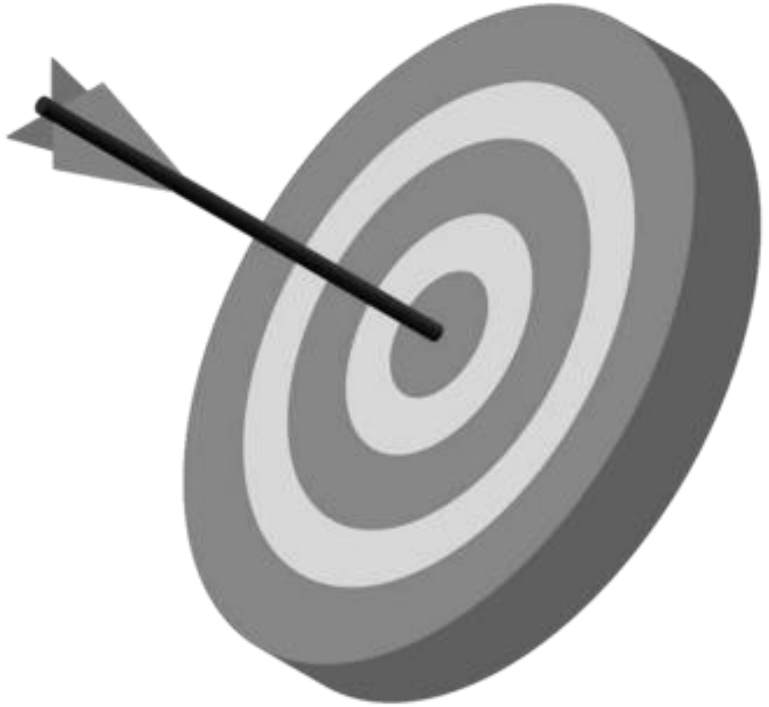
* Goals

Help students/visitors move to the classroom they want **without wandering** around looking for a room number.

Save time by showing entrances, elevators, and stairs for **the shortest route** to the classroom in the building.

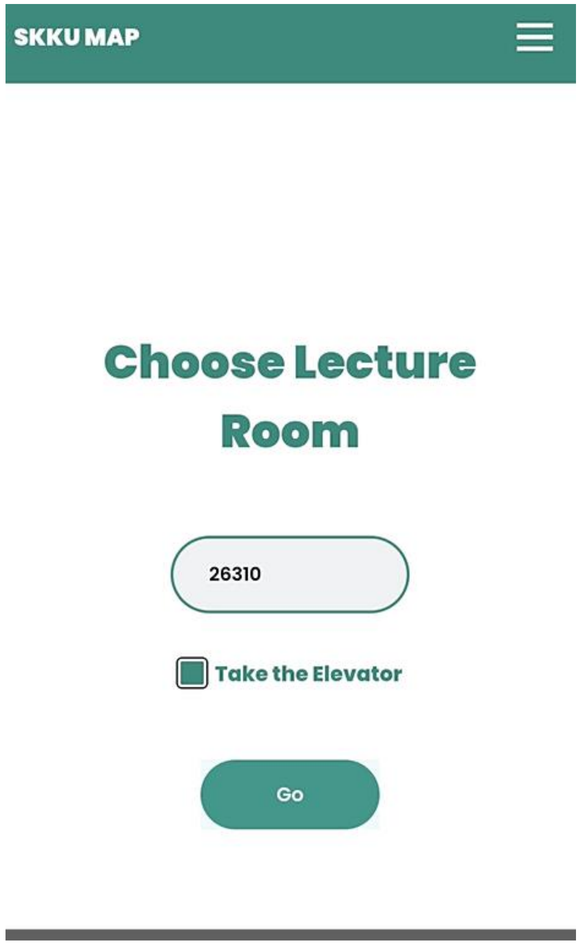
Don't need to understand complex building layouts simply by following the specified path.

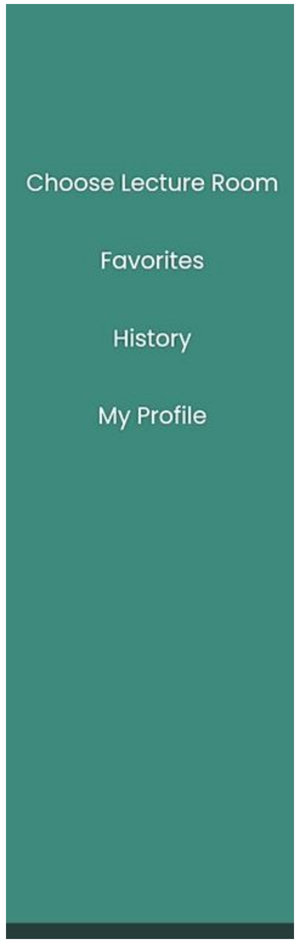
Easy to understand by visualizing the exact route in a **user-friendly way**.



02_Goals & Overview

* Main Functions





Main Functions

1 Outside + Inside Navigation

2 Favorites

3 History

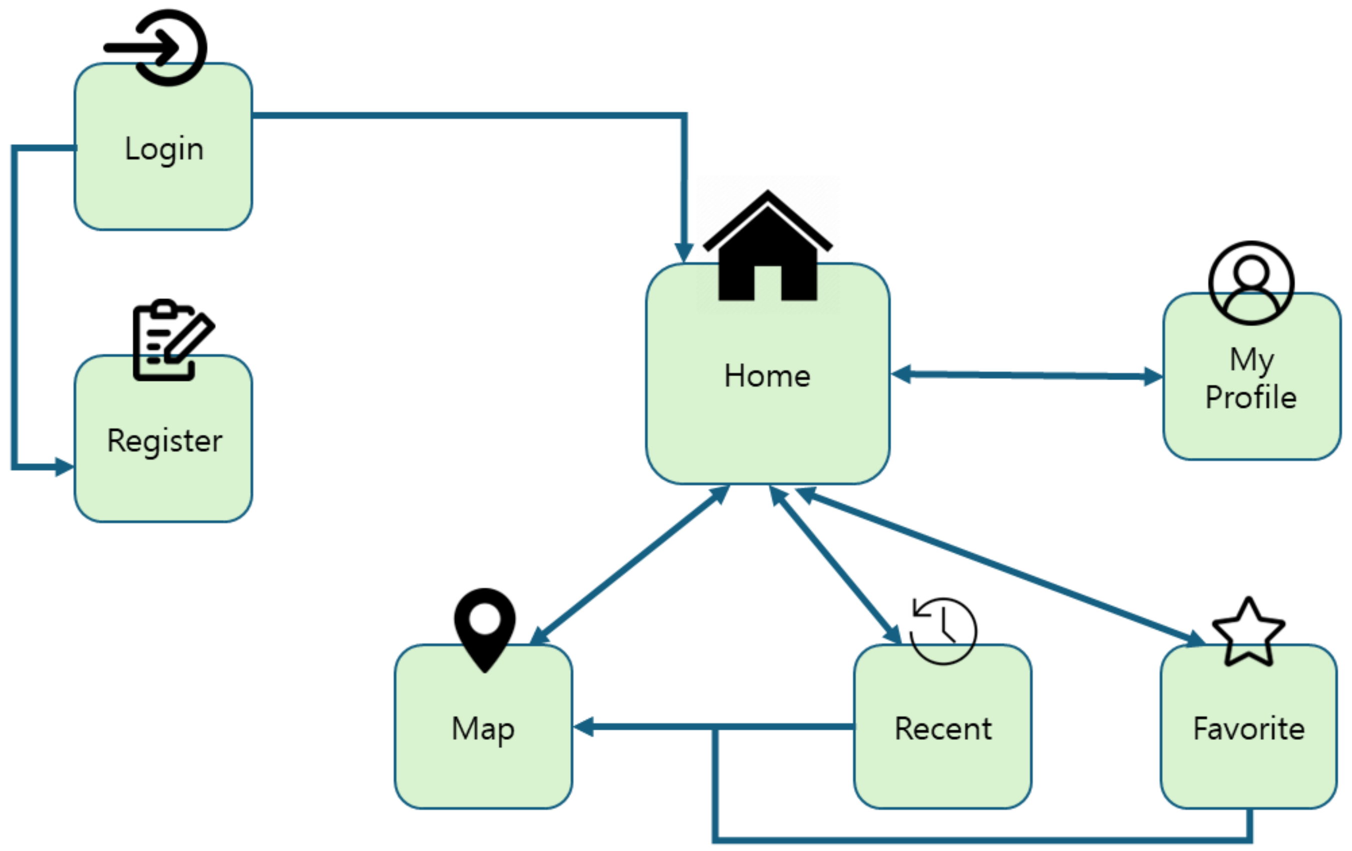
Section 3

Implementation & Demo Video

- Frontend
- Backend

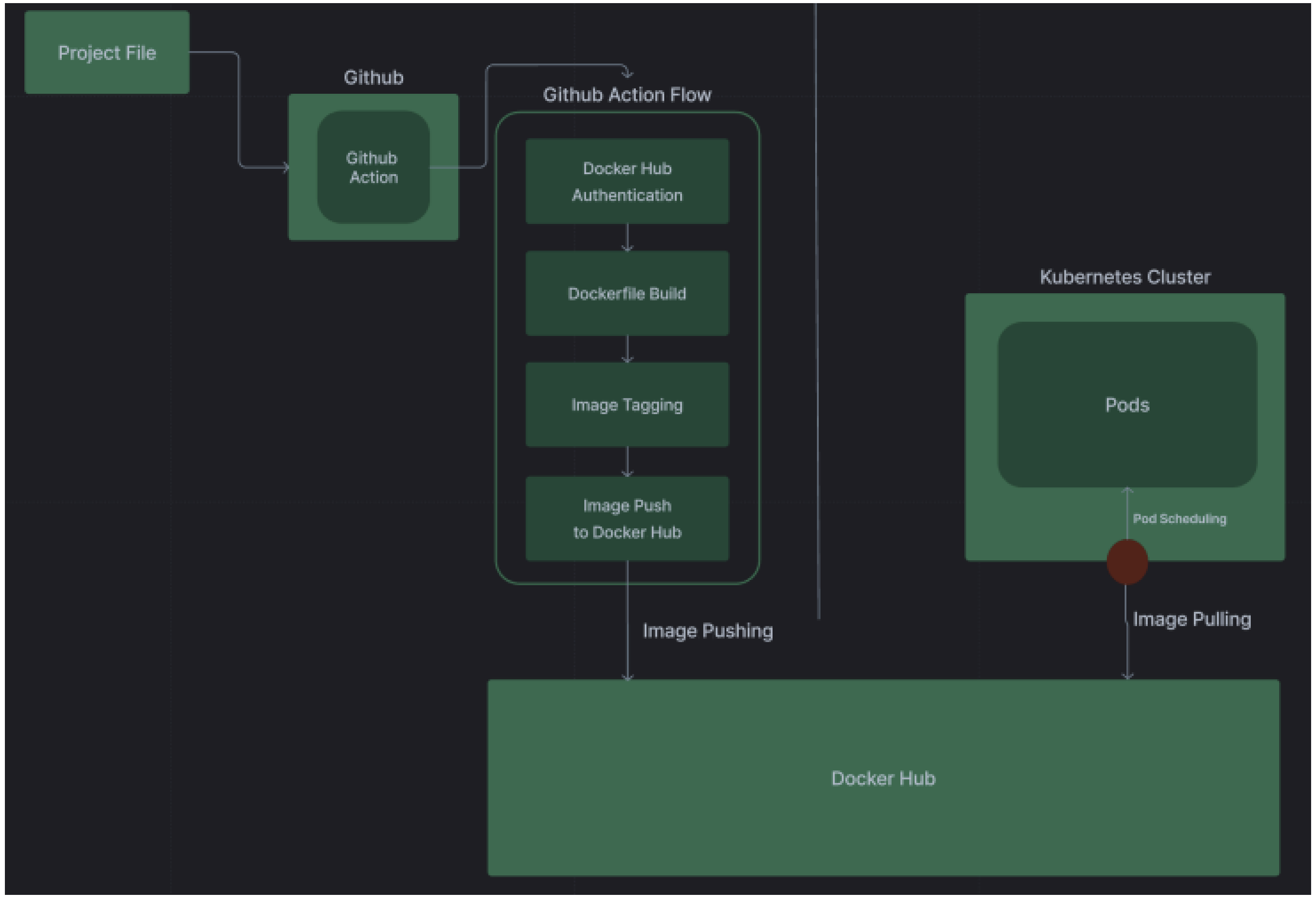
03_Implementation

❖ Frontend



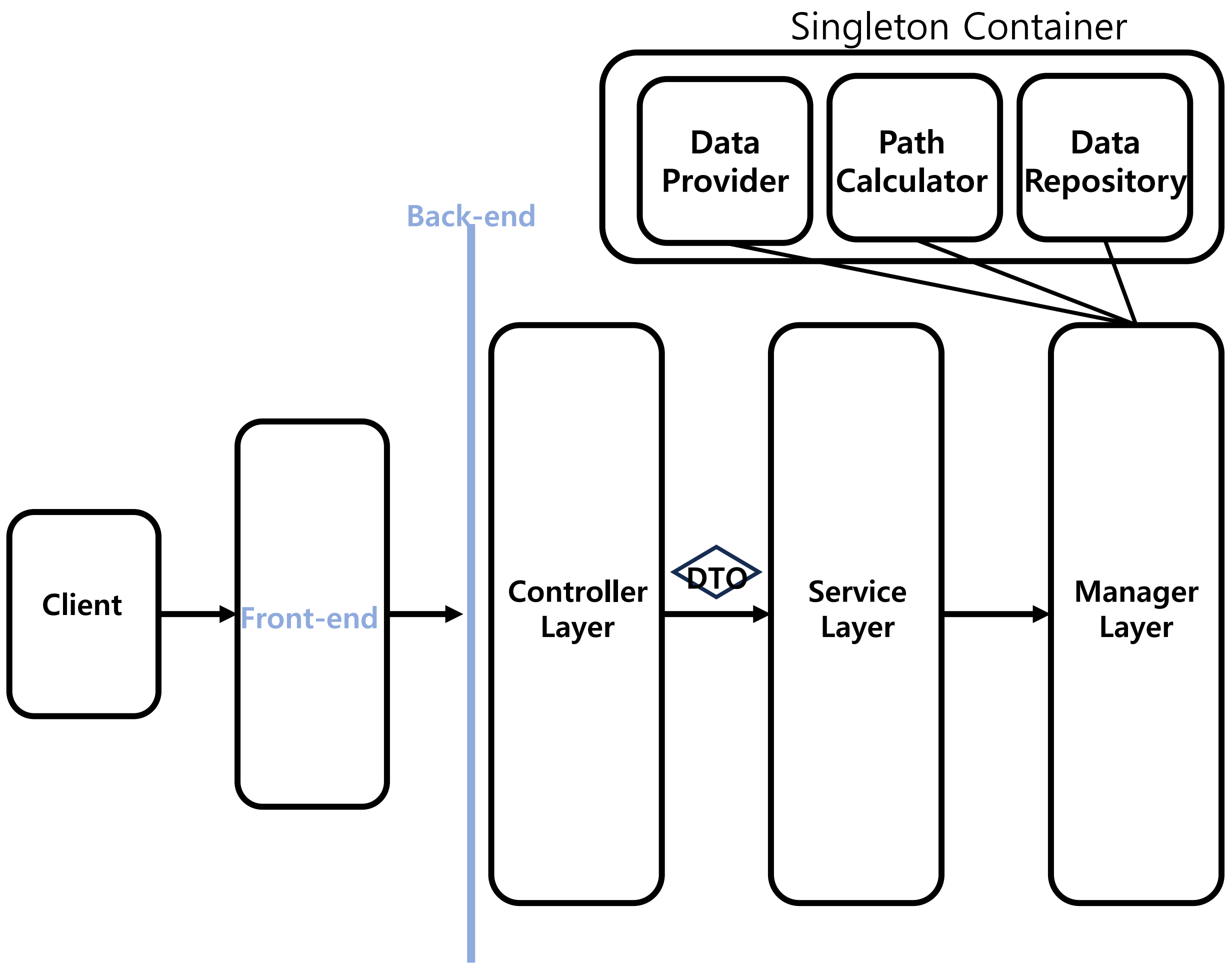
03_Implementation

❖ Backend – Deployment Flow



03_Implementation

❖ Backend – Structure



Database

MongoDB

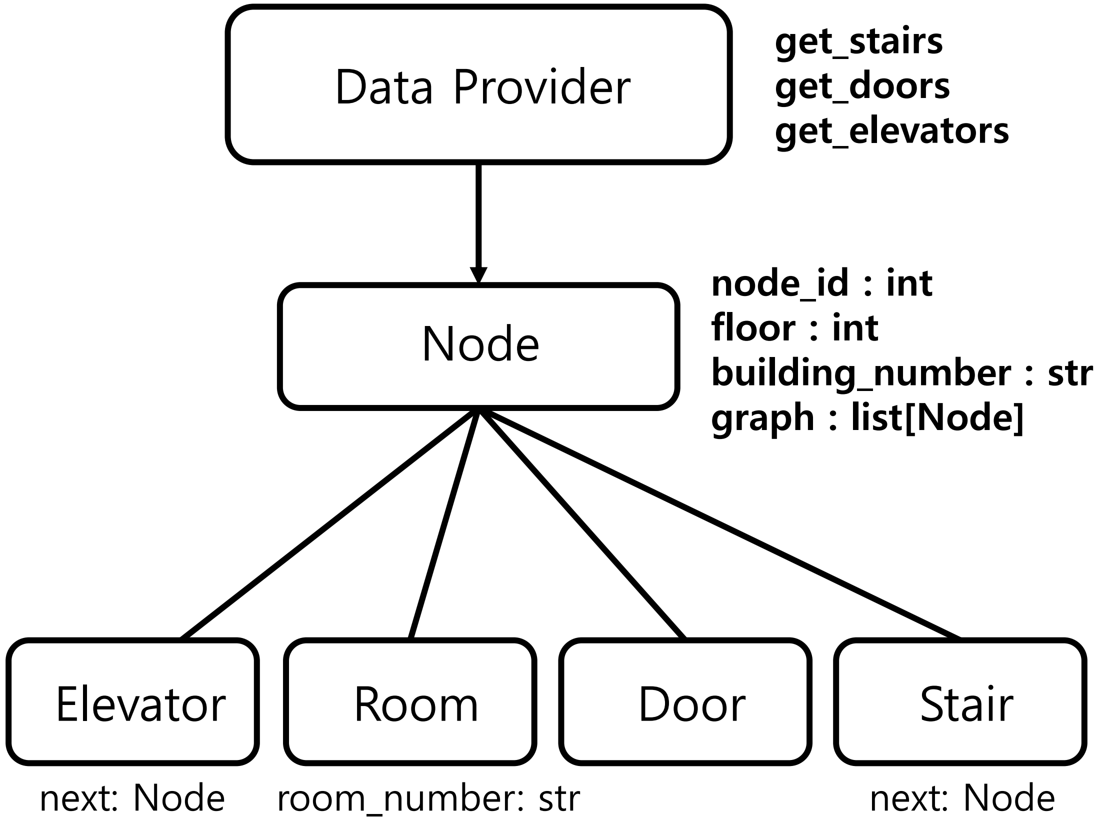
AWS S3 + EFS storage

Web Server Framework

FastAPI

03_Implementation

❖ Backend – Data Structure



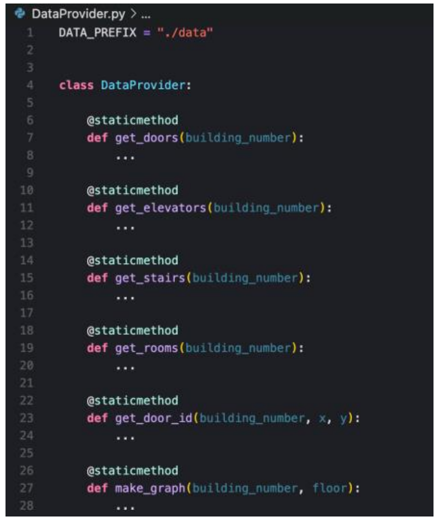
Nodes

- **Elevator**
- **Room**
- **Door**
- **Stair**

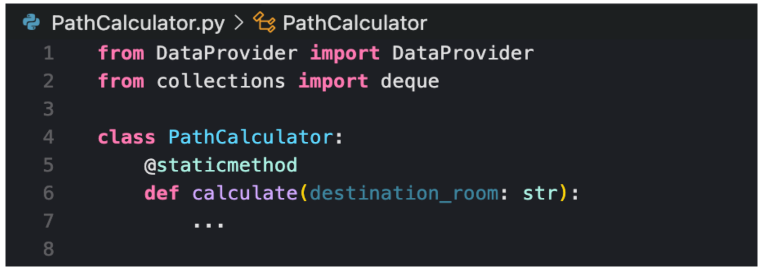
03_Implementation

❖ Backend – Implementation Result

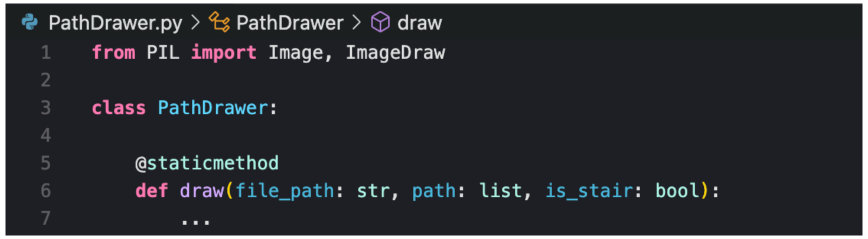
DataProvider	**Building Structure → Graph Data Structure**
PathProvider	**Calculate the shortest path from class 'DataProvider'**
PathDrawer	**Draw the shortest path visually**


DataProvider.py > ...

```python
DATA_PREFIX = "./data"


class DataProvider:

    @staticmethod
    def get_doors(building_number):
        ...

    @staticmethod
    def get_elevators(building_number):
        ...

    @staticmethod
    def get_stairs(building_number):
        ...

    @staticmethod
    def get_rooms(building_number):
        ...

    @staticmethod
    def get_door_id(building_number, x, y):
        ...

    @staticmethod
    def make_graph(building_number, floor):
        ...
```


PathCalculator.py > PathCalculator

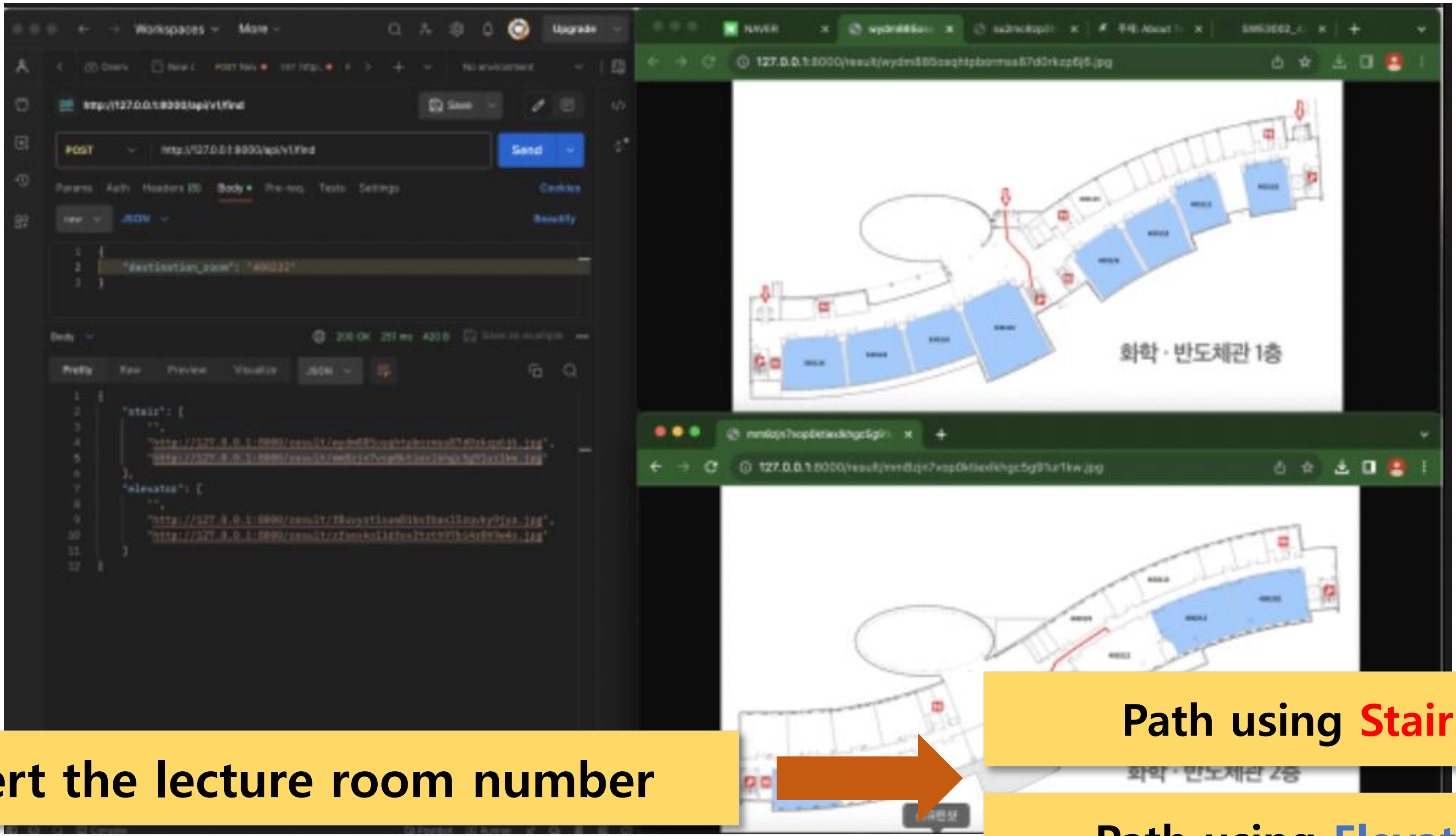
```python
from DataProvider import DataProvider
from collections import deque

class PathCalculator:
    @staticmethod
    def calculate(destination_room: str):
        ...

```


PathDrawer.py > PathDrawer > draw

```python
from PIL import Image, ImageDraw

class PathDrawer:

    @staticmethod
    def draw(file_path: str, path: list, is_stair: bool):
        ...
```

03_Implementation

❖ Backend – Implementation Result



```
POST http://127.0.0.1:8000/api/v1/find
```

```
{
    "destination_room": "400222"
}
```

200 OK

```
{
    "stair": [
        "",
        "http://127.0.0.1:8000/result/[illegible].jpg",
        "http://127.0.0.1:8000/result/[illegible].jpg"
    ],
    "elevator": [
        "",
        "http://127.0.0.1:8000/result/[illegible].jpg",
        "http://127.0.0.1:8000/result/[illegible].jpg"
    ]
}
```

Insert the lecture room number

Path using Stairs

Path using Elevator

Section 4

Test Plan & Result

- Unit test
- Integration test
- System test

04_Test Plan & Result

❖ Unit Test – Frontend (1)

Use - Case	Test
Register	If the user enters an already registered ID
	If the user tries to sign up with a blank space
	If the password re-entry box and the password entry box are different
	If the user creates an incorrect or correct type of email
Login	If the user enters the correct ID and password
	If the user enters with an ID that does not exist in the database
	If ID and password do not match
Change Password	<Similar with register>

04_Test Plan & Result

❖ Unit Test – Frontend (2)

Use - Case	Test
Insert lecture room number	If the room number entered correctly
	If the classroom number is correctly delivered to the server
	If invalid lecture room number has entered
Explore external, internal routes	If it gets the correct map data based on the input
Recent	If the most recent directions information is updated correctly
	If the page change working properly when the room number is clicked
Favorites	If the user's click to register/delete favorites works correctly

04_Test Plan & Result

❖ Unit Test - Backend

Used Pytest (732 test cases)

Input	Expected Output	
Lecture Room Number	test_graph.py	Graph of building with a input lecture room
	test_path.py	Shortest path to the input lecture room
	test_drawer.py	Name of the resulting internal building map

```
> pytest test_graph.py
======================= test session starts =======================
test_graph.py .                                              [100%]
======================== 732 passed in 1.21s ======================

> pytest test_path.py
======================= test session starts =======================
test_path.py .                                               [100%]
======================== 732 passed in 2.51s ======================

> pytest test_drawer.py
======================= test session starts =======================
test_drawer.py .                                             [100%]
======================== 732 passed in 6.52s ======================
```

04_Test Plan & Result

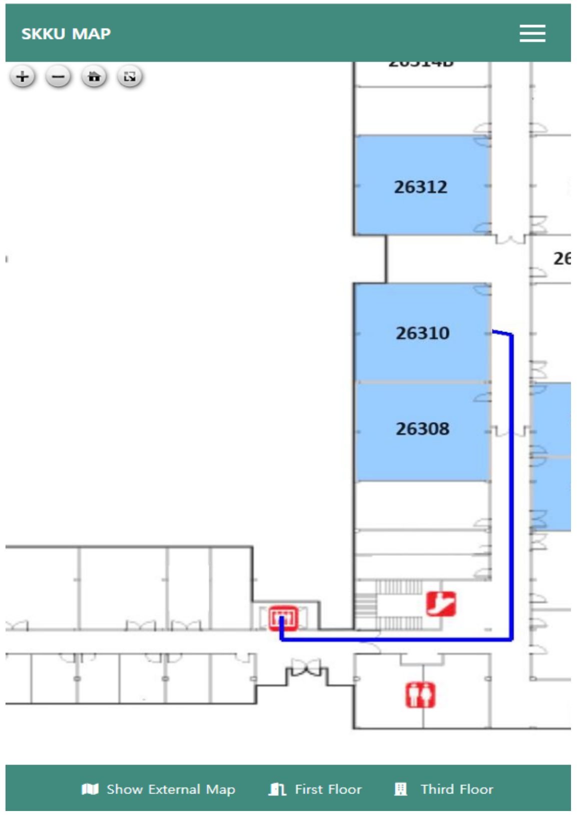
❖ Integration Test

1

Testcase	Path calculation and database integration testing

When the user enters the destination,

the application calculates the correct path and checks to return it to the picture.



04_Test Plan & Result

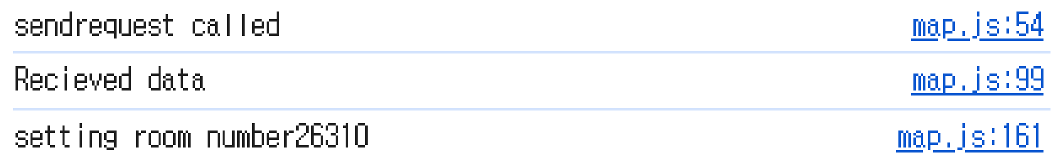
❖ Integration Test

2

Testcase	Integrate API endpoints with backend services

When a client sends a path request through an API endpoint,

verify that the backend service handles the request correctly and returns a response



```
sendrequest called                                   map.js:54
Recieved data                                        map.js:99
setting room number26310                            map.js:161
```

04_Test Plan & Result

❖ System Test

Resolving CORS issues	Allowed only the frontend server's Origin to block unnecessary API calls
	Restricted access using a Master API Key known only to the backend server
SQL Injection Vulnerability Assessment	Validated user input data
	Used Prepared Statements to prevent SQL Injection attacks
Authentication Security Review	Checked security setting for cookie usage
	Tested for potential authentication bypass

THANK YOU