수학과제 탐구 최종보고서

석관고등학교

학번, 이름	30102 김도연
지도교사	김민형

Ⅰ. 탐구 주제

공간 구문론을 이용한 교내 최단 동선 안내 시스템 구현

Ⅱ. 탐구 동기 및 목적

우리 학교는 여러 건물이 복잡하게 이어져 있고, 구조가 생소하여 신입생이나 외부인뿐만 아니라 재학생들도 특정 장소로 이동할 때 비효율적인 동선으로 이동하는 경우가 많음. 이를 해결하기 위해 2학년 정보과학 시간에 학습한 파이썬 노드와 간선을 활용하고 2학년 사제동행 시간에 학습한 HTML과 CSS를 기반으로 웹프로그래밍을 하여 학교 건물 구조의 최단 거리 알고리즘을 개발하고자 함.

Ⅳ. 탐구 과정

1. 이론적 배경

1-1. 공간 구문론

이번 프로젝트에서는 보행자의 통행 효율성을 높이기 위해 이 이론을 사용하였다.
공간 구문론이란, 공간을 인간의 행위와 사회적 상호작용이 일어나는 논리적 체계이다.
이 이론에서는, 공간의 배치가 사람들의 움직임과 만남의 패턴을 결정한다. 이 이론의 분석 과정은 2가지로 나뉜다.

1) 축선 (공간 내에서 길게 뻗은 시각적 통로) 과 볼록 공간(내부의 모든 점에서 서로를 볼 수 있는 최소 단위 공간) 으로 연속된 공간을 나눈다.
2) 각 공간을 점으로, 공간 사이 연결성을 선으로 치환하여 위상학적 거리로 수치화한다.

공간 구문론에서 다루는 지표로는 통합도, 연결도, 통제도가 있다. 공간 구문론을 사용하면 보행자의 흐름을 예측하고, 상권을 분석할 수 있다는 의의가 있다.
본 프로젝트에서는 석관고등학교의 각 교실, 계단, 화장실 등을 노드로 설정하고, 복도를 통해 이동할 수 있는 공간들을 간선으로 연결함으로써 학교 공간을 그래프 구조로 모델링하였다.

1-2. 다익스트라 알고리즘 (Dijkstra's Algorithm)

다익스트라 알고리즘은 가중치가 있는 그래프에서 특정 출발 노드로부터 목적지 노드까지의 최단 경로를 탐색하는 알고리즘이다. 본 프로젝트에서는 dijkstra() 함수를 통해 이를 구현하였다.

모든 노드의 거리를 무한대, 출발 노드만 0으로 설정한다. 그리고 큐에서 거리가 가장 짧은 노드를 꺼내 탐색하고 꺼낸 노드가 목적지면 즉시 종료한다. 현재 노드를 거쳐 인접 노드로 이동하는 거리가 기존보다 짧으면 갱신하고 큐에 삽입한다. 탐색 완료 후 지도를 목적지부터 역추적하여 최단 경로를 복원한다.

```
export function dijkstra(
  startId: string, -> 출발 노드
  endId: string, -> 도착 노드
  nodes: Map&string, Node&, -> 전체 노드 목록
  adjacency: Map&string, { to: string; weight: number }[]& -> 인접 리스트
): PathResult | null {
  const dist = new Map&string, number&(); -> 각 노드까지의 최단거리
  const prev = new Map&string, string | null&(); -> 경로 역추척용 이전 노드
  const visited = new Set&string&(); ->방문 여부 기록
  const pq: { id: string; dist: number }[] = []; -> 우선순위 큐
  nodes.forEach((_, id) =& {
    dist.set(id, Infinity); -> 모든 노드의 초기 거리를 무한대로 설정
    prev.set(id, null);
  });
  dist.set(startId, 0);
  pq.push({ id: startId, dist: 0 });
  function pqPop() {
    let minIdx = 0;
    for (let i = 1; i & pq.length; i++) {
      if (pq[i].dist & pq[minIdx].dist) minIdx = i;
    }
    return pq.splice(minIdx, 1)[0]; -> 큐에서 현재까지 발견된 거리가 가장 짧은 노드 선택하여 반환
  }
  while (pq.length & 0) {
    const { id: u } = pqPop();
    if (visited.has(u)) continue;
    visited.add(u);
    if (u === endId) break; -> 큐가 빌 때까지 반복, 꺼낸 노드가 이미 방문한 노드라면 건너뜀. 꺼낸 노드를 방문 처리하고 해당 노드가 목적지라면 탐색을 즉시 종료
    const neighbors = adjacency.get(u) ?? [];
    for (const { to: v, weight } of neighbors) {
      if (visited.has(v)) continue;
      const newDist = dist.get(u)! + weight;
      if (newDist & dist.get(v)!) {
        dist.set(v, newDist);
        prev.set(v, u);
        pq.push({ id: v, dist: newDist });
      }
    }
  } -> 현재 노드(u)와 인접한 노드(v)에 대해, u에서 v로 이동하는 거리(newDist)계산. 이 값이 기존에 저장된 v까지의 거리보다 짧으면 거리를 갱신하고, v의 이전 노드를 u로 기록하며 큐에 삽입한다. 이 과정을 통해 더 짧은 경로가 발견될 때마다 거리 정보가 갱신된다.
```

다익스트라 알고리즘은 모든 간선의 가중치가 음수가 아닐 때 항상 최적해를 보장하며, 본 프로젝트에서 사용한 유클리드 거리 기반의 가중치는 항상 양수이므로 이 알고리즘의 적용이 적합하다.

1-3 벡터 기반 이동 방향 판별

경로 안내문 생성 시 두 노드 사이의 이동 방향을 다음과 같이 판별할 수 있다.

```
const dx = next.x - cur.x
const dy = next.y - cur.y
const dir = Math.abs(dx) & Math.abs(dy)
? (dx & 0 ? '오른쪽' : '왼쪽')
: (dy & 0 ? '위쪽' : '아래쪽')
```

현재 노드 A(x_1, y_1)에서 다음 노드 B(x_2, y_2)로 이동할 때, 두 점을 잇는 변위 벡터를 구한다.

$\bar{v}$ = (dx, dy) = ($x_2 - x_1$, $y_2 - y_1$)

예를 들어 3-1반 (38.0, 5.0) 에서 3-2반 (46.5, 5.0) 으로 이동하는 경우

dx = 46.5 − 38.0 = 8.5
dy = 5.0 − 5.0 = 0

로 계산할 수 있다.

변위 벡터 (dx, dy)에서 절댓값이 더 큰 성분이 이동의 주된 방향이다.

| dx | > | dy | -> 가로 방향 이동
| dx | ≤ | dy | -> 세로 방향 이동

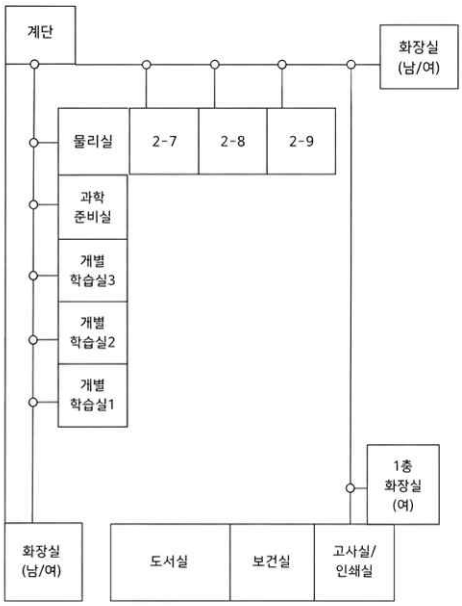
가로 방향이면
    dx > 0 -> 오른쪽
    dx < 0 -> 왼쪽
세로 방향이면
    dy > 0 -> 위쪽
    dy < 0 -> 아래쪽

위 예시에서 |dx| = 8.5, |dy| = 0 이므로 |dx| > |dy|, dx > 0 이므로 오른쪽으로 이동으로 안내된다.

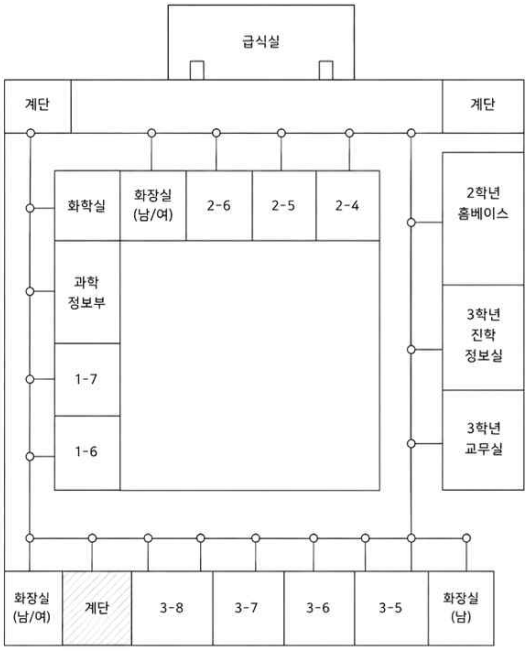
2. 시스템 설계 및 연구 방법

2-1. 공간의 그래프화

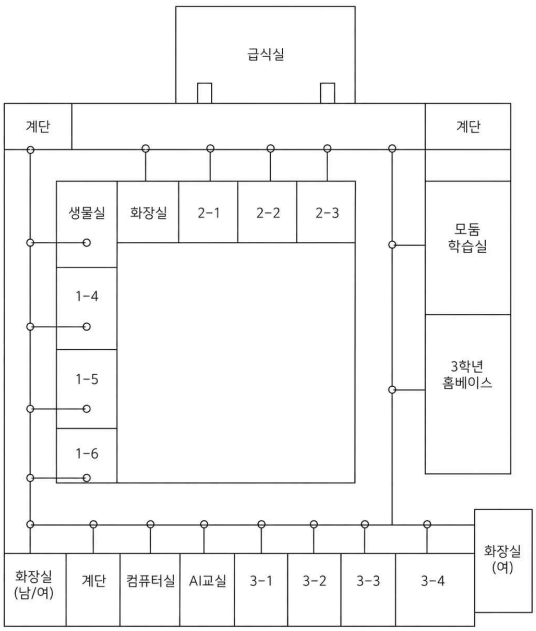
먼저 학교의 메인이 되는 ㅁ 자 건물 그래프화를 목표로 잡고, 층별 전개도를 그렸다. (그림 1~그림4) 이 전개도는 학교에서 제공하는 시설 배치도를 참고했으나, 미흡한 부분은 직접 답사를 하여 보완하였다. 그림은 손 그림으로 직접 그렸으며 생성형 인공지능이 디지털 그림으로 변환했다. 또한 사용자가 지나갈 수 있는 경로를 선으로 표현하였다.



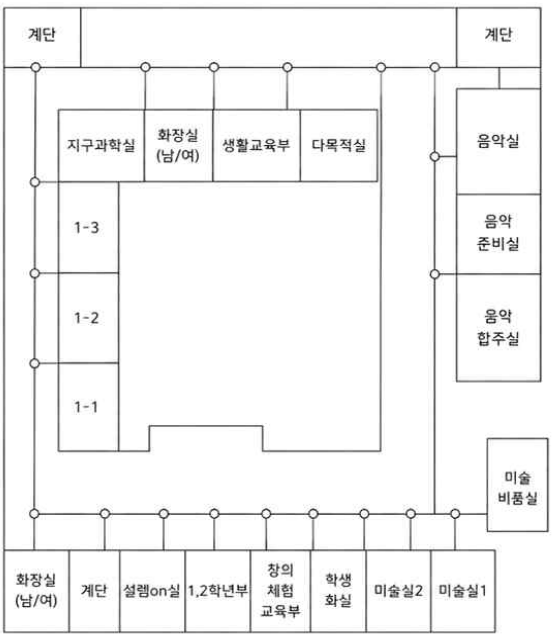
<그림 1 - 1층 배치도>



<그림 2 - 2층 배치도>



<그림 3 - 3층 배치도>



<그림 4 - 4층 배치도>

그다음으로 학교의 공간 곳곳을 측정하였다. 측정은 아이폰의 기본 앱인 '측정'을 사용하였다. 측정된 데이터를 기반으로 생성형 인공지능의 도움으로 공간 이름, x좌표, y좌표로 구성된 총 데이터를 생성하였다. (표 1~ 표 4)

공간 이름	X 좌표	Y 좌표
화장실(여,남)	4.25	5.0
계단(좌)	12.75	5.0
계단(2층행)	29.75	5.0
도서실	46.5	5.0
보건실	55.25	5.0
화장실(여)	71.5	5.0
계단(우)	71.5	15.0
물리실	4.25	43.2
화장실(상)	21.5	43.2
2-7반	30.4	43.2
2-8반	38.3	43.2
3-9반	46.75	43.2
계단(상)	63.75	43.2
과학준비실	4.25	34.2
개별학습실	4.25	25.8
계단(상단 좌)	21.5	51.0

<표 1 - 1층 노드 좌표 목록>

공간 이름	X 좌표	Y 좌표
화장실(여,남)	4.25	5.0
계단(좌·전 층)	12.75	5.0
계단(1층 행)	29.75	5.0
3-8반	38.0	5.0
3-7반	46.5	5.0
3-6반	55.25	5.0
3-5반	63.75	5.0
화장실(남)	71.5	5.0
계단(우)	71.5	15.0
화학실	4.25	43.2
화장실 남/여	21.5	43.2
2-6반	30.4	43.2
2-5반	38.3	43.2
2-4반	46.75	43.2
계단(상)	63.75	43.2
과학정보부	4.25	34.2
1-7반	4.25	25.8
1-6반	4.25	17.4
2학년 홈베이스	63.75	34.2
진학정보실	63.75	25.8
3학년부	63.75	17.4
급식실	34.1	51.0
계단(상단 좌)	21.5	51.0

<표 3 - 3층 노드 좌표 목록>

공간 이름	X 좌표	Y 좌표
화장실(여,남)	4.25	5.0
계단(좌)	12.75	5.0
컴퓨터실	21.25	5.0
AI실	29.75	5.0
3-1반	38.0	5.0
3-2반	46.5	5.0
3-3반	55.25	5.0
3-4반	63.75	5.0
화장실(여)	71.5	5.0
계단(우)	71.5	15.0
생물실	4.25	43.2
화장실(상)	21.5	43.2
2-1반	30.4	43.2
2-2반	38.3	43.2
2-3반	46.75	43.2
계단(상)	63.75	43.2
1-4반	4.25	34.2
1-5반	4.25	25.8
1-6반	4.25	17.4
모둠학습실	63.75	34.2
3학년 홈베이스	63.75	25.8
급식실	34.1	51.0
계단(상단 좌)	21.5	51.0

<표 2 - 2층 노드 좌표 목록>

공간 이름	X 좌표	Y 좌표
화장실(여,남)	4.25	5.0
계단(좌)	12.75	5.0
설렘 온실	21.25	5.0
1,2학년부	29.75	5.0
창의체험교육부	38.0	5.0
학생회실	46.5	5.0
미술실 2	55.25	5.0
미술실 1	63.75	5.0
미술비품실	71.5	5.0
계단(우)	71.5	15.0
지구과학실	4.25	43.2
화장실(상)	21.5	43.2
생활교육부	30.4	43.2
다목적실	46.75	43.2
계단(상)	63.75	43.2
1-3반	4.25	34.2
1-2반	4.25	25.8
1-1반	4.25	17.4
음악실	63.75	34.2
음악 준비실	63.75	25.8
음악합주실	63.75	17.4
계단(상단 좌)	21.5	51.0

<표 4 - 4층 노드 좌표 목록>

학교 공간을 그래프로 변환하기 위해 각 층의 노드를 정의하였다. 각 노드는 고유 키(key), 이름(name), x·y 좌표, 층(floor), 공간 유형(type)으로 구성된다. 공간 유형은 room(교실/교무실), stairs(계단), cafeteria(급식실)의 세 가지로 분류하였다. 고유 key와 type 생성은 단순한 반복 작업이므로 생성형 인공지능의 도움을 받았다.

```
type RawNode = { key: string; name: string; x: number; y: number; type: Node["type"] };
const FLOOR_NODES: Record&Floor, RawNode[]& = {
  1: [
    { key: "TL",  name: "화장실(여,남)",  x: 4.25, y: 5.0,  type: "room" },
    { key: "SL",  name: "계단(좌)",       x: 12.75,y: 5.0,  type: "stairs" },
    { key: "SL2", name: "계단(2층행)",    x: 29.75,y: 5.0,  type: "stairs" },
    { key: "LIB", name: "도서실",         x: 46.5, y: 5.0,  type: "room" },
    { key: "HLT", name: "보건실",         x: 55.25,y: 5.0,  type: "room" }…(이하 생략) ]
  2: [ … ],
  3: [ … ],
  4: [ … ],
}
```

간선은 실제로 복도를 통해 이동할 수 있는 공간들만을 연결하였다. 벽으로 막혀 있는 경로 (예: 2층 B동 3-8반 옆 복도가 막혀 있는 구조, 도서실 옆 계단은 1층과 2층만을 이동할 수 있는 구조) 는 간선에서 제외하였다.

```
const STAIR_CONNECTIONS: { key: string; floors: Floor[] }[] = [
  { key: "SL",  floors: [1, 2, 3, 4] },   // 계단(좌) 전층
  { key: "SL2", floors: [1, 2] },          // 계단(2층행/1층행) 1↔2만
  { key: "SU",  floors: [1, 2, 3, 4] },   // 계단(상) 전층
  { key: "SR",  floors: [1, 2, 3, 4] },   // 계단(우) 전층
  { key: "SX", floors: [1, 2, 3, 4] },   // 계단(상단좌) 전층
];
```

2-2. 가중치 설정

간선의 가중치는 두 노드 사이의 유클리드 거리를 사용하여 자동으로 계산하였다. 유클리드 거리는 평면 위 두 점 사이의 직선거리를 의미하며, 다음 공식으로 계산된다.

```
function dist(a: Node, b: Node) {
    return Math.sqrt((a.x - b.x) ** 2 + (a.y - b.y) ** 2);
  }
```

층간 이동의 경우 계단을 오르내리는 행위 자체의 이동 비용을 반영하기 위해 별도의 고정 가중치를 부여하였다.

```
const FLOOR_CHANGE_WEIGHT = 20;
```

이 내용은 프로그래밍 코드의 극히 일부분이며, 전체 코드는
GitHub Pages https://github.com/doyeonkimmm/seokgwan-map 에서 열람할 수 있다.

3. 웹 시스템 구현

3-1. 개발 환경 소개

- 정적 페이지 (홈 화면, 시설 안내, 메모, 제작 과정 등) : HTML, CSS, JavaScript
- 길 찾기 시스템 페이지 : React 18
- 커스텀 폰트 : 온글잎 의연체 TTF
- 반응형 처리 : 초기에는 px를 사용하다가 모바일 형식과 맞지 않다는 것을 깨달아 모바일 기기에 대응하는 단위인 vw/vh로 수정하였다.
- 배포 : GitHub Pages (https://doyeonkimmm.github.io/seokgwan-map/index/home.html)

- **HTML** : 웹 페이지의 구조와 콘텐츠를 정의하는 마크업 언어로, 제목·단락·이미지·링크 등의 요소를 태그로 표현한다.
- **CSS** : HTML 요소의 색상·크기·배치 등 시각적 스타일을 지정하는 언어이다.
- **JavaScript** : 웹 페이지에 동적인 기능을 부여하는 프로그래밍 언어로, 버튼 클릭, 팝업 표시, localStorage를 활용한 메모 저장 등의 기능을 구현하는 데 사용하였다.
- **React 18** : Meta에서 개발한 JavaScript 라이브러리로, 컴포넌트 기반으로 UI를 구성하며 본 프로젝트에서는 배치도 렌더링, 층 선택, 경로 탐색 등 길 찾기 핵심 기능을 구현하는 데 사용하였다.

3-2. 디자인 과정 (그림 5 ~ 그림 9)

- 페이지 디자인 : Canva, Figma로 직접 디자인
- 각종 버튼 디자인 : ibis paint로 그린 손 그림
- 포스트잇, 제작 과정 배경 : Pinterest에서 다운로드

- **Figma** : 웹 기반 UI/UX 디자인 도구로, 본 프로젝트에서는 각 페이지의 레이아웃과 디자인을 미리 설계하고 색상·폰트·크기 등의 수치를 확인하는 데 사용하였다.
- **Canva** : 웹 기반 그래픽 디자인 플랫폼으로, 본 프로젝트에서는 도움말 이미지 등 각종 시각 자료를 제작하는 데 사용하였다.


시작화면
길찾기
석관고등학교 길찾기
석관고등학교
길찾기
길찾기
시설안내
제작 과정
1층
2층
3층
4층
출발지 추가
목적지 추가

<그림 5, 6>


시설안내_특별실
시설안내_편의시설
시설안내_교무실
시설 안내
시설 안내
시설 안내
편의시설
특별실
교무실
편의시설
특별실
교무실
편의시설
특별실

<그림 7, 8, 9>

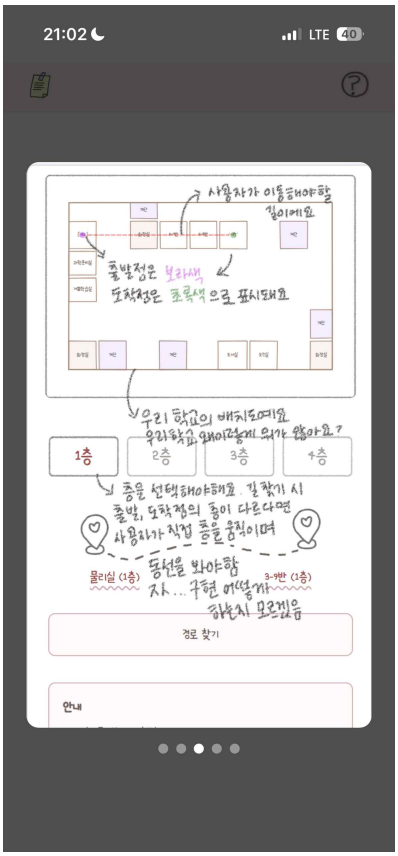
3-3. 주요 기능 및 UI 소개

1) 최초 접속 화면

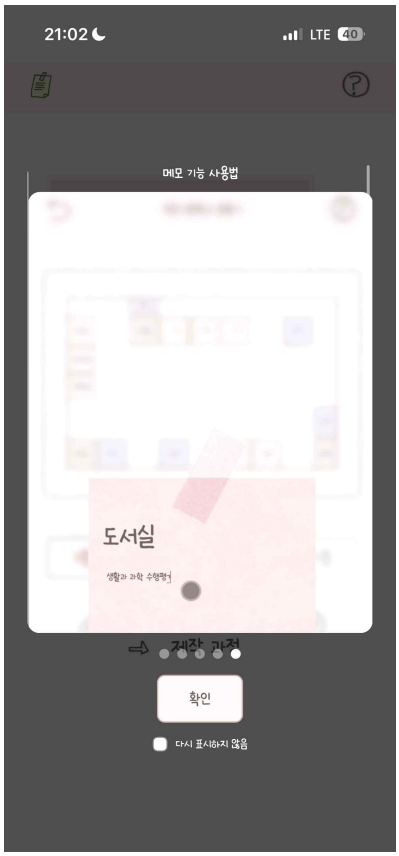
도움말은 사이트 최초 접속 시 자동으로 팝업으로 표시되며, 사용자는 화면을 옆으로 스와이프하며 도움말을 확인 할 수 있다. 도움말을 끝까지 확인하기 전에는 도움말을 끌 수 없게 설정해 놓았다. (그림 10) 도움말의 끝에는 사용 방법 영상을 첨부하였고, 자동으로 재생되게 설정하였다. localStorage와 sessionStorage를 활용하여 사용자가 "다시 표시하지 않음"을 선택하면 이후 접속 시 자동 표시되지 않도록 구현하였다. (그림 11)

2) 홈 화면

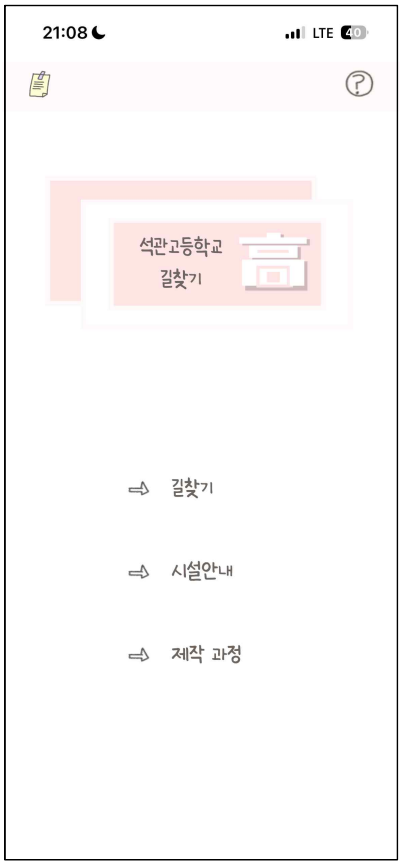
가운데에 석관고등학교 마크를 분홍색 계열로 재디자인하였다. 길 찾기, 시설 안내, 제작 과정의 세 가지 메뉴로 구성된다. 상단 바 좌측에는 메모 목록으로 이동하는 버튼, 우측에는 도움말 버튼이 배치된다. 도움말 버튼을 클릭하면 방금 표시되었던 도움말을 다시 열람할 수 있다. (그림 12)



<그림 10 - 최초 접속 화면>



<그림 11 - 다시 표시하지 않음>

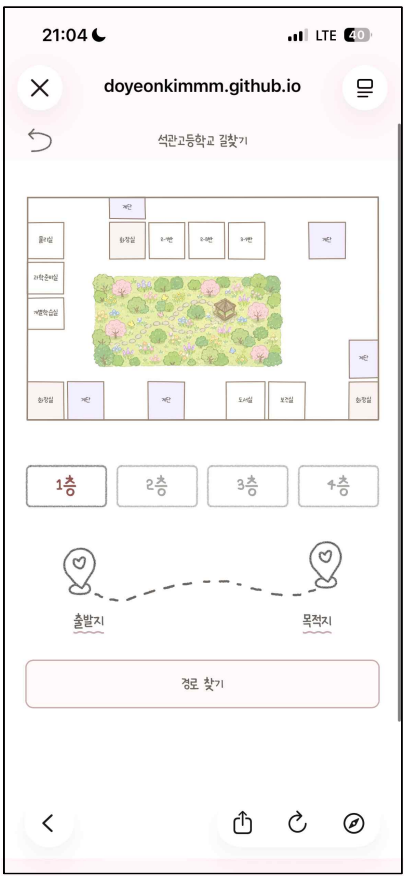


<그림 12 - 홈 화면>

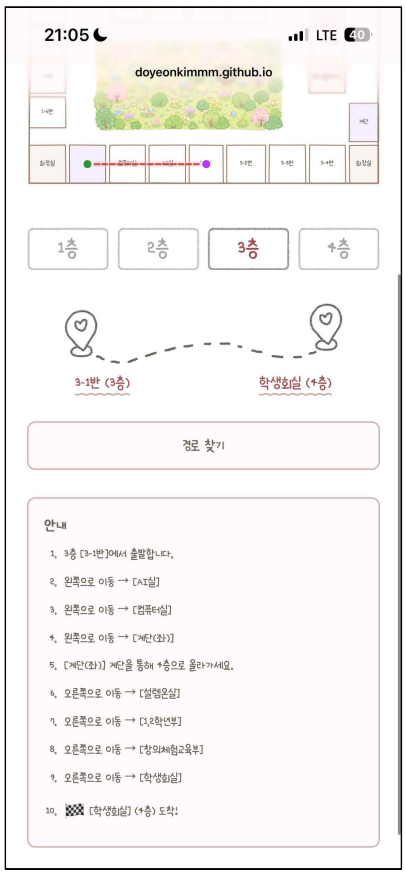
3) 길 찾기 화면

React로 구현된 길 찾기 페이지는 학교 배치도, 층 선택 버튼, 출발지/목적지 입력창, 경로 안내 영역으로 구성된다. (그림 13) 출발지와 목적지를 선택하면 다익스트라 알고리즘이 실행되어 최단 경로가 배치도에 시각화되고, 단계별 이동 안내문이 표시된다. (그림 14) 또한 배치도의 각 방을 클릭하면 메모를 작성할 수 있는 팝업이 표시되어 해당 장소에 메모를 남길 수 있다. 저장된 메모는 localStorage에 장소별로 저장되어 다음 접속 시에도 유지된다. 홈 화면 상단의 메모 버튼을 클릭하면 작성한 모든 메모를 목록으로 확인할 수 있다. (그림 15)

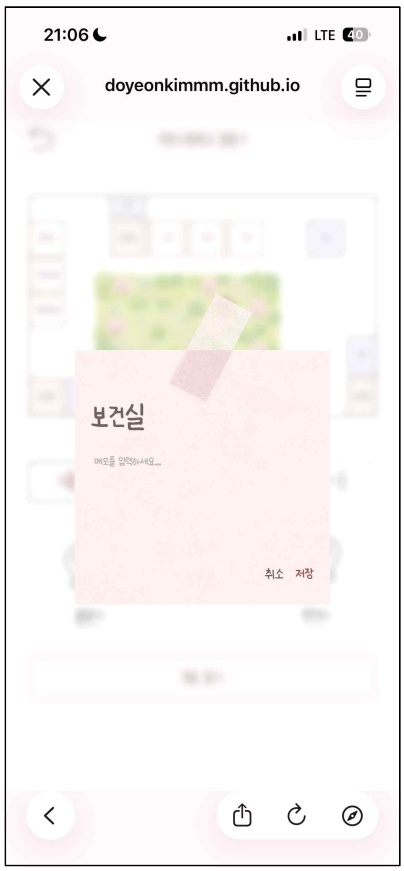
여기서 주목할 점은 배치도 중앙에 폭풍의 언덕 (정원)을 배치하였다는 점이다. 실제로 폭풍의 언덕에 가면 정자가 있는데, 이 정자까지 그렸다.



<그림 13 - 길 찾기 초기 화면>



<그림 14 - 경로 안내문>



<그림 15 - 메모지 팝업 창>

5) 시설 안내 화면

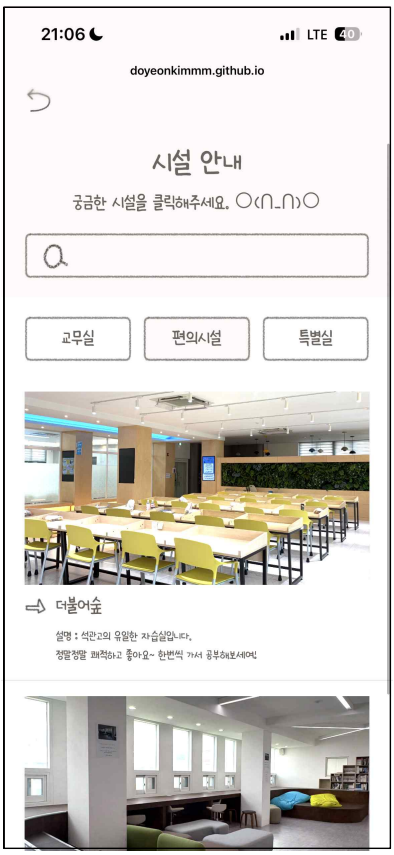
교무실, 편의시설, 특별실의 세 카테고리로 구분하여 각 시설의 사진, 설명, 위치 정보를 제공한다. 검색창을 통해 원하는 시설을 검색할 수 있으며, 위치 핀 버튼을 클릭하면 해당 시설이 위치한 층의 배치도로 자동 이동한다. 교내를 돌아다니며 직접 찍은 사진을 첨부하였고, 설명란에는 비록 쓸데없을 수 있지만 나의 TMI를 넣어보았다. (그림 16)

6) 제작 과정 팝업 창

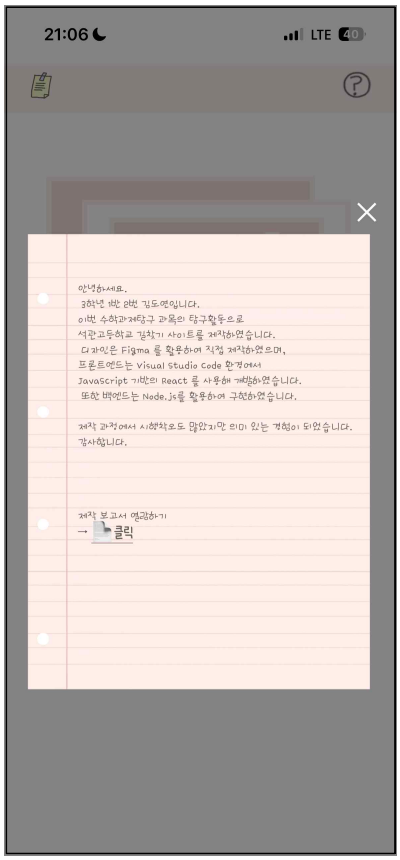
홈 화면에서 제작 과정을 클릭하면 팝업창이 뜬다. 여기에는 간단하게 이 프로그램의 개발 환경과 나의 짧은 소감이 담겨있다. 또한 아래쪽에 pdf를 내려받을 수 있는 버튼이 있는데, 이를 클릭하면 본 보고서를 열람할 수 있다. (그림 17)

7) 메모 화면

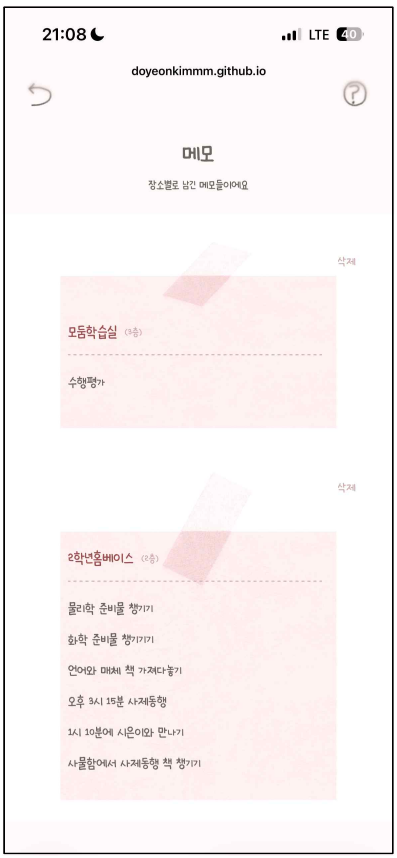
저장된 메모를 열람할 수 있는 화면으로, 홈 화면에서 메모 그림을 클릭하여 실행한다. 이곳에서는 사용자가 적은 모든 메모를 확인할 수 있으며, 삭제 기능이 포함되어 있다. 여기서 주목할 점은 공간 활용의 효율성을 위해 텍스트의 길이에 따라 메모지의 길이도 달라진다는 것이다. 따라서 남는 메모지 없이 화면에 메모를 나타낼 수 있다. (그림 18)



<그림 16 - 시설 안내 화면>



<그림 17 - 제작 과정 팝업 창>



<그림 18 - 메모 화면>

Ⅴ. 탐구 결과 및 분석

4. 탐구 결과 및 분석

본 시스템을 통해 학교 내 임의의 두 공간 사이의 최단 경로를 성공적으로 탐색할 수 있었다. 같은 층 내의 경로뿐만 아니라 다른 층 사이의 경로도 계단을 통해 자동으로 연결되어 안내된다.
예를 들어 내가 속한 반인 3-1반에서 이 프로젝트를 하는 장소인 학생회실로 이동하는 경우, 시스템은 다음과 같은 경로를 탐색한다.

- 3층 [3-1반]에서 출발합니다.
- 왼쪽으로 이동 → [AI실]
- 왼쪽으로 이동 → [컴퓨터실]
- 왼쪽으로 이동 → [계단(좌)]
- [계단(좌)] 계단을 통해 4층으로 올라가세요.
- 오른쪽으로 이동 → [설렘 온실]
- 오른쪽으로 이동 → [1,2학년부]
- 오른쪽으로 이동 → [창의체험교육부]
- 오른쪽으로 이동 → [학생회실]
- □ [학생회실] (4층) 도착!

이처럼 층간 이동이 포함된 복잡한 경로도 다익스트라 알고리즘을 통해 최단 경로로 올바르게 탐색 된다. 경로는 배치도 위에 점선으로 시각화되어 사용자가 직관적으로 이동 경로를 파악할 수 있다.
그렇다면 이 예시 경로에서 정말 좌측 계단을 사용하는 것이 우측 계단을 사용하는 것보다 빠를까?
이는 일정한 보폭으로 직접 걸으며 발자국 계산을 해보았다.
우측 계단을 사용하는 경우 -> 약 111보
좌측 계단을 사용하는 경우 -> 약 105보
따라서 6보라는 아주 미세한 차이지만 좌측 계단을 사용하는 경우가 실제로 최단 거리임을 증명하였다.

다만 현재 구현에서는 복도의 실제 형태가 아닌 노드 간의 직선 연결로 경로를 표현하므로, 실제 복도의 굴곡이 반영되지 않는 한계가 있다. 또한 노드의 수가 제한적이므로 일부 공간은 가장 가까운 노드로 근사하여 경로를 탐색한다.

Ⅵ. 결론 및 느낀 점

5-1. 결론

본 연구에서는 공간 구문론과 다익스트라 알고리즘을 결합하여 석관고등학교 길 찾기 웹 시스템을 개발하였다. 학교의 복잡한 공간 구조를 그래프로 모델링하고, 최단 경로 탐색 알고리즘을 적용함으로써 사용자가 원하는 장소를 쉽고 빠르게 찾을 수 있는 시스템을 구현하였다.

5-2. 어려웠던 점과 해결 방안

개발 과정에서 가장 어려웠던 점은 크게 세 가지였다.

첫째, GitHub Pages 배포 시 흰 화면만 표시되는 문제가 발생하였다. 이는 배포 페이지에서 F12로 관리자 화면을 들어가 console에 있는 오류 메시지를 사용해 해결하였다. 이는 Vite가 로컬 환경 기준으로 파일 경로를 설정하기 때문에, 배포 환경에서는 JavaScript와 CSS 파일을 찾지 못하는 것이 원인이었다. vite.config.ts에서 base: '/seokgwan-map/index/map/'으로 경로를 명시함으로써 해결하였으며, 이를 통해 배포 환경에서의 경로 관리의 중요성을 이해하게 되었다.

둘째, 배치도에 방 이름을 추가하는 과정에서 TypeScript 타입 오류가 대량으로 발생하였다. 기존에 number[][]로 정의된 FLOOR_ROOMS 배열에 문자열 이름을 추가하려 했기 때문이었다. (number | string)[][]로 타입을 변경함으로써 해결하였으며, TypeScript에서 정확한 타입 정의의 중요성을 배울 수 있었다.

셋째, 메모 기능 추가 후 Cannot read properties of null (reading 'useState') 오류가 발생하였다. 원인 생성형 인공지능을 통해 분석한 결과, home.html 파일에서 &body& 태그가 중복으로 선언되고 팝업 div가 &body& 태그 바깥에 위치하는 등 HTML 구조가 잘못되어 있었다. 구조를 올바르게 정리함으로써 해결하였으며, HTML 문서 구조의 중요성을 다시 한번 깨달았다.

5-3. 향후 개선 방향

본 프로젝트는 실제 복도의 형태를 세밀하게 경로로 표현하지 못하였다는 한계가 있다. 경로를 세밀하게 표현할수록 좌표 계산이 복잡해지고 노드의 수가 기하급수적으로 증가하기 때문이다. 추후 노드 밀도를 높여 실제 복도 형태에 가까운 경로를 구현한다면 시스템의 완성도가 더욱 높아질 것으로 기대된다.
또한 본 시스템은 석관고등학교의 중심 건물만을 대상으로 구현하였다. 향후 학교 전체 건물로 범위를 확장한다면 더욱 활용성이 높은 시스템이 될 것이다.
아울러 석관고등학교에는 엘리베이터가 한 대 설치되어 있으나, 본 시스템에서는 이를 반영하지 못하였다. 현재 엘리베이터를 이용하더라도 접근할 수 없는 장소가 일부 존재하는데, 이를 길 찾기 시스템에 반영하여 이동 가능 여부를 안내할 수 있다면 모든 사용자에게 편의를 제공함과 동시에 교통약자를 배려하는 시스템으로 발전할 수 있을 것이다.

Ⅶ. 참고 자료

문지혜, 위세영, 유시환. (2018-06-20). 다익스트라[*] 알고리즘: 최소검색비용 최적경로 탐색. 한국정보과학회 학술발표논문집, 제주.

박경록 (2023). 코딩 테스트 합격자 되기 (파이썬 편). 골든래빗. pp. 404-409.