

# 김지민 | UI/UX에 대해 고민하는 개발자



## Contact & Channels

- Email | [zeem\\_m2@naver.com](mailto:zeem_m2@naver.com)
- Phone | 010-7369-0111
- Github | <https://github.com/wlals4264>
- Blog | <https://velog.io/@wlals4264/posts>
- Portfolio | <https://bluemin-portfolio.vercel.app/>



## Front-End Developer

- 사소하더라도 사람들의 경험을 바꾸는 서비스를 통해 가치를 만드는 개발자입니다.
- 소통을 통한 피드백을 성장의 원동력으로 삼아 함께 발전합니다.
- 어떤 일이든 도전과 실패를 두려워하지 않습니다.
- 한 번 시작한 일은 끝까지 완수하는 집요함이 있습니다.



## Stacks

### Front-End

- HTML, CSS, JavaScript(ES6)
- React.js
- TypeScript
- Tailwind, Scss, Styled Component, CSS Module
- TanStack Query
- Recoil, Zustand

### Collaboration & Tools

- Git, Github
- Notion, Figma



## Projects

### OlaOla

2024.11.18 ~ 2024.12

개인 프로젝트

- 소개 : 클라이언트들을 위한 커뮤니티 공간
- 사용 기술  
React, TypeScript, Recoil, Indexed DB, Tailwind CSS
- 배포 링크 : <https://ola-ola-nine.vercel.app/>
- Github : <https://github.com/wlals4264/olala>

### 프로젝트 주요 기능 및 트러블 슈팅

- 서버 없이 브라우저 DB로 구현
  - 처음으로 진행한 프로젝트로 서버 구현없이 프론트엔드 기술에 집중하기 위해 브라우저에서 제공하는 **indexedDB**를 활용  
→  
**indexedDB**를 해당 프로젝트에 도입한 이유 : **indexedDB**는 서버와의 통신처럼 비동기적으로 작동하며 이미지, 영상을 저장해야하는 프로젝트 특성상 브라우저에서 제공하는 로컬스토리지와 쿠키등의 저장소들에 비해 저장할 수 있는 용량이 크고 Blob과 File 객체와 같이 할 수 있다는 이점이 있어 도입
  - **indexedDB**를 사용하여 DB를 구현하면서 백에서 구현하는 CRUD 기능을 직접 구현해보는 경험
- 캐러셀, 스크롤 스냅 등을 라이브러리 없이 구현
- 외부 라이브러리 커스터마이징(React Quill)

- 커스터마이징 과정 중 라이브러리 내에서 id를 부여할 수 없어 특정 이미지를 수정할 수 없는 문제를 발견하였고 이를 해결하기 위해 `uuid` 와 `DOMParser` 를 사용하여 각 이미지의 `data-set` 속성으로 `data-img-id` 속성을 넣어서 저장하고 이를 `indexedDB`에 해당 피드 DB에도 추가하여 이미 존재하는 `data-img-id` 면 수정을 존재하지 않는다면 추가를 할 수 있도록 구현함

## MOMO

2024.12.16 ~ 02

팀 프로젝트

- 소개 : 혼밥이 싫은 사람들을 위한 '밥친구 구하기' 플랫폼

- 사용 기술

React, TypeScript, TanStack Query, Recoil, Tailwind


- 팀 구성

Back-End 2 / Front-End 2


- 발표 자료 :

- Github : <https://github.com/Team-momo-front/momo-front>

### 기여한 주요 기능 및 트러블 슈팅

- 프로젝트 디자인 시스템 기반 마련
- 회원가입 & 로그인
  - 프로필 검증 처리
    - 사이트를 이용하기 위해서는 프로필 생성이 필수 요소이기 때문에 회원가입 절차에 프로필 생성 단계가 있었으나, 백&프론트 간 프로세스에 대한 이해가 달라 프로필 생성은 회원가입 이후 동작하도록 구현됨(프로세스가 분리됨)
    - 백엔드에 리프레쉬 토큰 발급이 안되는 이슈가 있어 해당 로직 수정에 오랜 시간이 걸릴것이라 판단, 따라서 클라이언트단에서 프로필 검증 로직을 구현하도록 결정
    - 초기에는 403오류를 프로필 검증 오류로 통일하고 공용 컴포넌트인 `AxiosInstance`에서 403에러 처리를 `Interceptor`로 구현하려고 했으나, 팀회의를 통해 여러 코드만으로 처리하기에는 의미상 명확하지 않고 처리 범위가 광범위하다고 판단, 현재 상황에서는 각 API 호출별로 프로필 검증이 필요한 경우 403 에러 처리를 개별로 처리하기로 결정
    - 이후 리팩토링 과정에서, 최초 로그인 시 진입 페이지인 메인 페이지의 헤더에서 프로필 정보를 불러오므로. 이때, 해당 API가 403 검증 오류를 반환하는 경우, 이를 활용하여 프로필 생성 페이지로 이동하도록 구현함으로써 **최초의 한 번만 검증하도록 하여 유저 사용성을 높이는 방향으로 개선**
- 마이페이지 기능 구현
  - 주치한 모임 & 참여한 모임 구현
    - 활성화된 상태를 자식 컴포넌트인 `PostCard`에 전달해 필요한 버튼 및 승인 상태, 모집 상태 등이 렌더링 되도록 구현
  - 신청자 보기 페이지 구현
    - 신청자의 신청 상태를 변경하고 처리 결과를 구분해서 보여주도록 구현
  - 상태 업데이트 개선
    -  기존 방식

api 호출 성공 이후 리다이렉트 설정을 하여 페이지 이동 시, 이동한 페이지의 데이터 업데이트가 필요하였고, `useLocation` 을 사용하여 정의한 `navigate` 에 `state` 를 전달하여 이동시키고 해당 `state` 를 컴포넌트의 `key` 값으로 전달하여 해당 컴포넌트가 `state` 가 변경될 때 리마운트 될 수 있도록 구현함.

→ 간단하게는 페이지 새로고침을 할 수도 있었지만 그렇게 하면 UI/UX 면에서 좋지 않다고 생각했고, 리액트의 장점인 클라이언트 사이드 렌더링의 의미가 사라진다고 생각함. 여러 방법을 찾다가 상태 변경을 이용하여 구현. 해당 컴포넌트는 리액트 라우터를 사용하여 아울렛으로 구현하고 있는 부분이 라 더더욱 부분적으로 리렌더링을 구현하는 것이 좋다고 생각함.
  -  Refactoring

tanstack Query의 `queryClient.invalidateQueries` 를 사용하여 리팩토링.

→ 팀원분이 프로젝트에서 tanstack Query를 이용하고 있기 때문에 `useMutation`의 `onSuccess` 콜백에 `queryClient.invalidateQueries` 로 기존 쿼리를 무효화시키는 방법을 제안해주셨고 찾아보니 해당 방법이 이와같이 서버 상태 업데이트가 필요할 때 서버와 동기화를 쉽게 처리해주는 라이브러리 메서드로 굳이 인위적인 리렌더링 없이 쿼리를 자동으로 만료시키고 다시 가져오도록 요청함으로써 서버 상태를 바로 업데이트 할 수 있었음.
- 채팅
  - 채팅 UI
    - 채팅창 진입 버튼과 진입 효과를 커스텀 애니메이션을 사용해 동적으로 렌더링 되도록 구현
    - 채팅방 진입시 스크롤이 제일 하단으로 위치하도록 구현

- 채팅 기능 구현
  - 양방향 통신을 위해 웹소켓 사용
  - 웹소켓을 활용하여 메세지 전송을 처리하기 위해 STOMP 사용
- 채팅 삭제 및 내보내기 기능 구현
  - 주최자 입장에서 채팅 삭제, 즉 모임 삭제가 가능하도록 구현
  - 모임 API는 총 3개의 컴포넌트에서 호출해야했고 각각의 성공 이벤트를 개별적으로 처리하기 위해 성공 이벤트 핸들러 `onSuccess`를 props로 전달받아 처리할 수 있도록 구현
  - 승인된 멤버 강퇴 기능 구현

## MBTI Chat

2024.03월 중

개인 프로젝트

(  
토이 프로젝트)

- 소개 : MBTI 기반 채팅 사이트
- 사용 기술  
React, TypeScript, Node.js, WebSocket, Recoil, Tailwind
- 배포 링크 : <https://mbti-chat-gamma.vercel.app/>
- Github : <https://github.com/wlals4264/mbti-chat>
- Velog Series Link :  
<https://velog.io/@wlals4264/series/%ED%86%A0%EC%9D%B4%ED%94%84%EB%A1%9C%EC%A0%9D%ED%8A%MBTI-%EB%9E%9C%EB%8D%A4%EC%B1%84%ED%8C%85>

### 프로젝트 주요 기능 및 트러블 슈팅

- **WebSocketController 구현**
  - Polling, SSE, WebSocket의 차이에 대한 이해를 바탕으로 양방향 통신인 웹소켓을 사용하여 채팅 기능 구현
  - 큰 규모의 프로젝트는 다양한 이유에서 기능이 추가될 수도 수정될 수도 있고 한 번 구현하고 끝나는게 아니라 만일의 사태에 대비해서 언제든 어느 곳에서든 가져다 쓸 수 있도록 모듈화, 캡슐화하고 중앙화하여 재사용성과 확장성을 높이는 것이 중요하므로 간단한 기능 구현 테스트 코드가 완성된 이후 웹소켓 컨트롤러를 구현.
    - 모든 웹소켓 연결을 위한 하나의 핵심 컨트롤러(WebSocket Controller)역할을 하며, 관련 기능을 한 군데에서 관리하여 기능이 확장될 때도 이 컨트롤러를 기반으로 확장성을 가질 수 있음.
    - 가장 기본적인 웹소켓 생성 및 웹소켓 생성 이후 진행되는 하위 이벤트들인 `onopen`, `onmessage`, `onerror`, `onclose`를 제외한 기능들을 커스텀 훅으로 분리하여 컨트롤러의 순수성을 지키고 동시에 최소 단위의 기능들로 분리하여 커스텀 훅을 구현하여 재사용성 및 확장성을 높임
- **Render를 통한 간단한 서버 배포**
  - github과의 연동을 통해 간단히 서버를 배포할 수 있는 Render를 사용하여 배포된 서버를 통해 단순 프론트 정적 배포가 아닌 실제 사이트를 이용가능하도록 하며 토이 프로젝트를 마무리함.

## Flobby

2025.04. ~

7월 중 MVP 완료

(이후 배포 예정)


팀 프로젝트

- 소개 : 기존 소모임 앱의 불편함을 개선한 지역 기반 소모임 플랫폼
- 사용 기술  
React, TypeScript, Zustand, Scss
- 팀 구성  
🌟  
경력자 & 현업자 & 취업준비생들이 모여 구성된 팀  
  
Web PM 1 / App PM 1  
Back-End 3 / Web Front-End 3 / App Developer 2  
Designer 2
- Github : private 레포로 이미지 첨부로 대체

## 기여한 주요 기능 및 트러블 슈팅



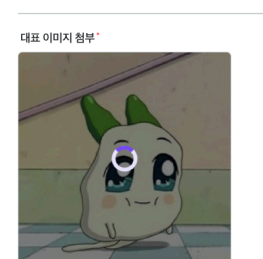
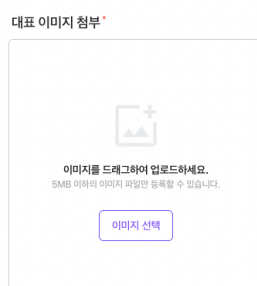
- 메인 페이지 - 지역 변경 셀렉터

- 지역 변경 상태 관리
  - 지역 변경을 위한 모델, 컨트롤러를 구현하여 모델 안에서 지역 리스트 api 요청을 통해 들어온 데이터를 저장
  - 지역 변경에 따른 데이터 요청을 위해 지역 id를 쿠키에 담아 저장, 백에서 쿠키 검증 → 필터링을 통해 데이터를 내려주면 해당 데이터를 메인과 모임 목록 api를 통해 각각 렌더링
-  트러블 슈팅
  - 문제 상황

지역 셀렉터는 메인 헤더에서 렌더링되며 페이지 전역에 사용되지만 그 안에 담기는 선택 지역 데이터 및 관심 지역 데이터는 메인 api에 종속되어 있어 어떤 페이지에서는 필연적으로 메인 api를 호출해야하므로 불필요하고 효율적이지 못한 데이터 흐름이 발생하였다.

- ### ■ 해결 방법
- 프론트 팀에게 종속된 데이터로 인해 각 페이지마다 불필요하게 메인 api를 호출해야하기 때문에 불필요한 데이터까지 요청하게 되므로 효율성이 떨어지기 때문에 지역 데이터 api 분리에 대한 의견을 공유하였고 QA기간동안 백엔드팀에게 데이터 의존성을 개선하고 불필요한 api 호출 구조를 개선하기 위해 지역 데이터 api 분리를 요청하였고 결과적으로 지역 리스트를 내려주는 api에 관심 지역, 선택 지역을 함께 보내주는 것으로 변경점을 협의함

- 모임 등록 - 이미지 업로더와 Drag & Drop 기능 구현



- 모임 등록페이지에서 사용되는 이미지 업로더를 구현하고 Drag and Drop 클래스형 유틸 함수를 구현
- 사용성 증대를 위해 기획팀과 협의하에 HEIC 파일과 HEIF 파일 변환 라이브러리를 추가하였고, 고압축된 파일인 HEIF 형식의 특성상 고해상도일수록 파일 변환에 소요되는 시간이 증가하였고 유저 경험에 영향을 미친다고 판단하여 기획팀과 디자인팀에 로딩 스피너 처리를 제안 및 요청하였고 이를 적용함

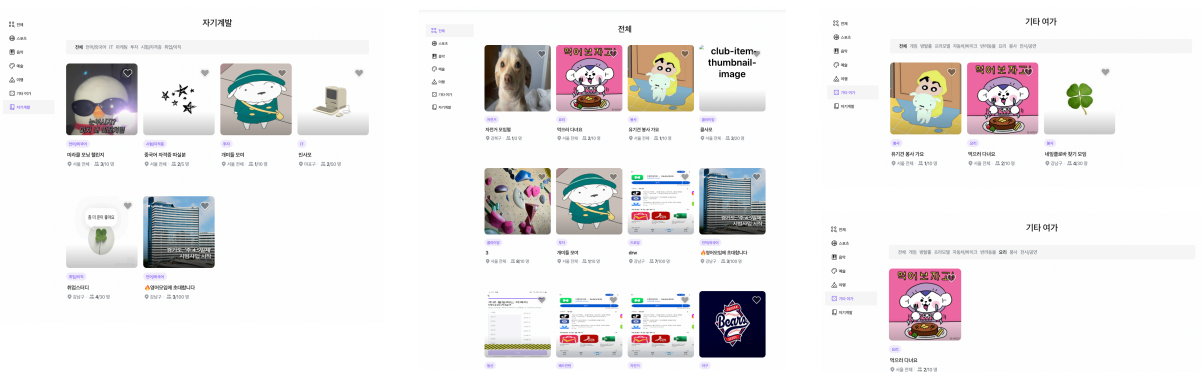
- 모임 등록 & 정기 모임 등록 - 공통 드롭다운 컴포넌트 구현

- 게시글 등록 페이지에서 빈번하게 사용되는 드롭다운 컴포넌트를 공용 컴포넌트로 구현
  - 모임 등록 페이지에 지역과 카테고리 드롭다운이 필요하여 가장 작은 단위의 드롭다운 컴포넌트를 구현 후 지역 드롭다운과 카테고리 드롭다운 각각에 활용.
  - 이후 추가된 정기 모임 등록 페이지에서 시간 드롭다운에서 활용하면서 잘 만들어진 공용 컴포넌트의 중요성을 느낀 케이스.

- 정기 모임 등록 - 데이트 피커(라이브러리 없이 구현)

- 데이트피커의 자유로운 커스터마이징과 애러 핸들링을 위해 외부 라이브러리 없이 직접 구현을 선택
- 기획팀과 이해가 달라 비활성화 및 활성화된 날짜에 대한 QA가 들어왔으나 라이브러리 사용 없이 직접 구현했기 때문에 해당 부분에 대한 수정 처리가 원활히 이루어짐

- 모임 목록 - 카테고리 컴포넌트 및 카테고리에 따른 모임 목록 렌더링 구현



- 카테고리 리스트 api 호출을 통해 불러온 메인 카테고리 데이터와 서브 카테고리 리스트를 분리하여 각각의 데이터 리스트를 렌더링할 수 있도록 구현.
- 메인 카테고리는 쿠키에 담아 관리, 서브 카테고리는 모임 목록 api 요청시 body에 담아 요청하기 위해 각각의 데이터를 분리하여 관리가 필요하였고 이에 따른 api 요청 로직과 쿠키 유효 함수 구현하여 쿠키값(메인 카테고리)과 요청값(서브 카테고리)에 따라 데이터가 렌더링 될 수 있도록 구현.
- 쿠키 유효 함수는 후에 지역 데이터 api 분리에 따른 리팩토링에 재사용함.

## Portfolio Site

2025.03 ~ 2025.05

개인 프로젝트

- 소개 : 포트폴리오용 웹사이트
- 사용 기술  
Next.js, TypeScript, Scss
- 배포 링크 : <https://bluemin-portfolio.vercel.app/>
- Github : <https://github.com/wlals4264/bluemin-portfolio>

### 프로젝트 주요 기능 및 트러블 슈팅

- Next.js 13에서 도입된 App Router 사용
- 초기 개발 속도 향상을 위해 라이브러리 적극 활용
- Terminal을 통해 해당 컴포넌트로 이동할 수 있도록 CodeSnap 컴포넌트 구현
  - 싱글 페이지에 컴포넌트들이 나열되어 있는 프로젝트 구조 특성상 Nav 기능이 중요하다고 생각하여 헤더에 포함된 Nav이외에 개발자들에게 친숙한 터미널 UI를 통해 해당 컴포넌트로 이동할 수 있는 기능을 제공하여 사용성을 높임과 동시에 유저에게 흥미로운 재미 요소를 제공하고자 구현함
  - 모바일은 다른 디바이스에 비해 비교적 텍스트 타이핑이 번거롭고 터미널이라는 UI가 어색하다고 판단하였기에 모바일에서는 해당 기능을 제외하고 more 버튼과 헤더의 nav 기능을 통해서만 진입할 수 있도록 함
- 반응형 웹 구현
  - 포트폴리오 특성상 채용 담당자들이 다양한 디바이스에서 진입할 수 있다고 생각하였고, 디바이스별 분기점을 비교적 타이트하게 잡아 사이트에 접근하는 유저들이 디바이스에 상관없이 원활히 사이트를 이용할 수 있도록 반응형을 구현함
- History API를 사용하여 뒤로가기 버튼 클릭시 모달이 닫히도록 구현
  - `popState` 이벤트에 모달 상태를 닫히도록 하는 이벤트 핸들러를 등록하여 뒤로가기 버튼을 눌렀을 때 history에서 pop되도록 구현하여 기존에는 닫힘 버튼을 통해서만 모달을 이탈할 수 있었지만 웹 유저가 습관적으로 뒤로가기 버튼을 통해 페이지 이탈을 한다는 점을 고려하여 뒤로가기 버튼을 통하여 모달을 이탈할 수 있도록 하여 유저 사용성을 높임



## Experiences

### 제로베이스 프론트엔드 스쿨

2024.03 ~ 2025.01.25

- 제로베이스에서 진행하는 프론트엔드 개발자 양성 교육 과정 수료
  - 웹프론트엔드 중심으로 배우는 자기 주도형 학습 과정
  - CS지식, 자료구조, 알고리즘 학습
  - HTML, CSS, JavaScript 학습을 통한 웹 기반 기술 습득
  - React.js, TypeScript, Recoil 등 프론트엔드 개발 기술 습득



## Education

### 숙명여자대학교

2014.03 ~ 2019.02.

- 음악대학 작곡 전공
- 문화예술기획 연계 전공