비트캠프 네이버 클라우드 기반 웹 데브옵스 과정 9기

Team. DevHomes

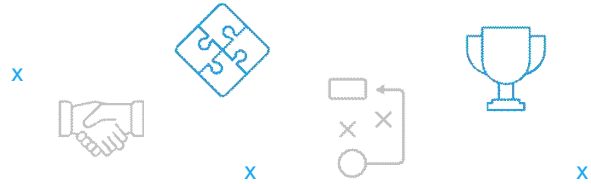
사용자를 위한 부동산 매물 중개 서비스

BANGEZ

5조 고준호 노태호 엄현지 정우석

지도강사 박정관 강사님

Contents



01 팀원 소개

자랑스럽고 훌륭한 방이지의 팀원을 소개합니다.





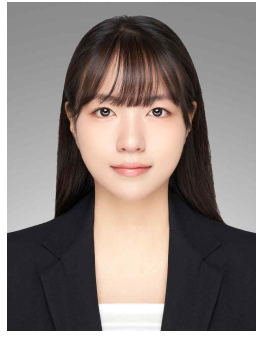

01

TEAM BANGEZ

팀원 소개

고준호

노태호



엄현지

정우석

02 프로젝트 소개

방이지는 무슨 프로젝트인가요?






02

ABOUT BANGEZ

프로젝트 기획 의도

1 **문제 인식**

부동산 거래 전 과정의 복잡성과 정보의 비대칭성으로 인한 사용자들의 어려움 파악

2 **해결책 구상**

사용자 경험을 개선하고 신뢰성을 높일 수 있는 부동산 통합 플랫폼 기획

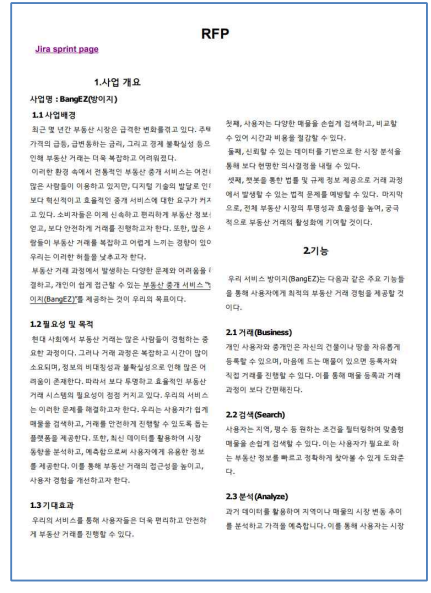
3 **기술 적용**

최신 웹 기술과 클라우드 인프라를 활용한 혁신적인 서비스 설계

4 **목표 설정**

부동산 매칭의 모든 단계를 효율적으로 지원하는 원스톱 솔루션 개발

방이지 프로젝트는 부동산 거래 매칠 과정에서 발생하는 다양한 문제점들을 해결하고자 기획되었습니다.
정보의 비대칭성, 신뢰성 부족, 복잡한 절차 등으로 인해 사용자들이 겪는 어려움을 인식하고, 이를 해결할 수 있는 플랫폼을 만들어 보고자 했습니다.



RFP

Jira sprint page

1.사업 개요

사업명 : BangEZ(방이지)

1.1 사업배경

최근 몇 년간 부동산 시장은 급격한 변화를겪고 있다. 주택 가격의 급등, 금변동하는 금리, 그리고 경제 불확실성 등으로 인해 부동산 거래는 더욱 복잡하고 어려워졌다.

이러한 환경 속에서 전통적인 부동산 중개 서비스는 여전히 많은 사람들이 이용하고 있지만, 디지털 기술의 발달로 인해 보다 혁신적이고 효율적인 중개 서비스에 대한 요구가 커지고 있다. 소비자들은 이제 신속하고 편리하게 부동산 정보를 얻고, 보다 안전하게 거래를 진행하고자 한다. 또한, 많은 사람들이 부동산 거래를 복잡하고 어렵게 느끼는 경향이 있어 우리는 이러한 허들을 낮추고자 한다.

부동산 거래 과정에서 발생하는 다양한 문제와 어려움을 해결하고, 개인이 쉽게 접근할 수 있는 부동산 중개 서비스 "방이지(BangEZ)"를 제공하는 것이 우리의 목표이다.

1.2 필요성 및 목적

현대 사회에서 부동산 거래는 많은 사람들이 경험하는 중요한 과정이다. 그러나 거래 과정은 복잡하고 시간이 많이 소요되며, 정보의 비대칭성과 불확실성으로 인해 많은 어려움이 존재한다. 따라서 보다 투명하고 효율적인 부동산 거래 시스템의 필요성이 점점 커지고 있다. 우리의 서비스는 이러한 문제를 해결하고자 한다. 우리는 사용자가 쉽게 매물을 검색하고, 거래를 안전하게 진행할 수 있도록 돕는 플랫폼을 제공한다. 또한, 최신 데이터를 활용하여 시장 동향을 분석하고, 예측함으로써 사용자에게 유용한 정보를 제공한다. 이를 통해 부동산 거래의 접근성을 높이고, 사용자 경험을 개선하고자 한다.

1.3 기대효과

우리의 서비스를 통해 사용자들은 더욱 편리하고 안전하게 부동산 거래를 진행할 수 있다.

첫째, 사용자는 다양한 매물을 손쉽게 검색하고, 비교할 수 있어 시간과 비용을 절감할 수 있다.

둘째, 신뢰할 수 있는 데이터를 기반으로 한 시장 분석을 통해 보다 현명한 의사결정을 내릴 수 있다.

셋째, 챗봇을 통한 법률 및 규제 정보 제공으로 거래 과정에서 발생할 수 있는 법적 문제를 예방할 수 있다. 마지막으로, 전체 부동산 시장의 투명성과 효율성을 높여, 궁극적으로 부동산 거래의 활성화에 기여할 것이다.

2.기능

우리 서비스 방이지(BangEZ)는 다음과 같은 주요 기능들을 통해 사용자에게 최적의 부동산 거래 경험을 제공할 것이다.

2.1 거래(Business)

개인 사용자와 중개인은 자신의 건물이나 땅을 자유롭게 등록할 수 있으며, 마음에 드는 매물이 있으면 등록자와 직접 거래를 진행할 수 있다. 이를 통해 매물 등록과 거래 과정이 보다 간편해진다.

2.2 검색(Search)

사용자는 지역, 평수 등 원하는 조건을 필터링하여 맞춤형 매물을 손쉽게 검색할 수 있다. 이는 사용자가 필요로 하는 부동산 정보를 빠르고 정확하게 찾아볼 수 있게 도와준다.

2.3 분석(Analyze)

과거 데이터를 활용하여 지역이나 매물의 시장 변동 추이를 분석하고 가격을 예측합니다. 이를 통해 사용자는 시장

02

WHY BANGEZ

프로젝트 차별점

사용자 경험 극대화

BangEZ는 직관적인 인터페이스와 고급 필터링 시스템을 통해 사용자가 원하는 조건에 맞는 매물을 쉽고 빠르게 찾을 수 있도록 설계되었습니다. 지도 기반 검색과 카드형 매물 정보 제공으로 사용자의 선호에 맞는 다양한 탐색 방식을 지원합니다.

종합적인 부동산 정보 제공

단순한 매물 정보를 넘어 주변 교육 시설, 생활 편의 시설, 교통 정보 등 종합적인 입지 정보를 제공합니다. 이를 통해 사용자는 더욱 합리적인 결정을 내릴 수 있습니다.

신뢰성 있는 매칭 시스템

포인트 보증금 기능을 도입하여 허위 매물을 방지하고 거래의 신뢰성을 높였습니다. 거기에 더해 매물을 기준으로 한 실시간 채팅 기능으로 즉각적인 소통을 가능하게 합니다. 이를 통해 사용자 간의 신뢰를 구축하고 거래 과정의 투명성을 확보합니다.

최신 기술 활용

OAuth 로그인, 서버 기반 인증, 실시간 데이터 처리 등 최신 웹 기술을 적극 활용하여 안전하고 효율적인 서비스를 제공합니다. 클라우드 기반 인프라를 통해 확장성과 안정성을 확보했습니다.

02

WHY BANGEZ

프로젝트 주요 기능 및 시스템

AOuth 로그인

서버 기반 로그인

매물 등록 및 검색

실시간 채팅

<MSA 기반의 개발 및 모듈화된 시스템 설계>
각 서비스(게이트웨이, 유레카 서버, 서비스 서버 등)를 독립적으로 개발하고 멀티모듈로 유지 관리할 수 있도록 구현하였습니다.

<실시간 데이터 처리>
Spring Servlet외에도 Spring WebFlux와 FastAPI를 사용해 대규모 트래픽을 효과적으로 처리할 수 있는 비동기 처리와 실시간 데이터 전송을 구현하였습니다.

<보안 강화>
Spring Security와 JWT, OAuth를 활용해 사용자의 인증과 권한 관리를 강화했습니다.

<클라우드 인프라 활용>
Naver Cloud Platform(NCP)을 활용해 서버 인프라를 구축하고 관리했습니다.

포인트 충전 및 소비

지도 기반 매물 제공

통계 제공

입지 정보 제공

02

KEY FEATURES OF BANGEZ

핵심 기능 소개

- 매물 검색 시간 단축 및 사용자 만족도 향상

지도 기반 매물 검색

- 부동산 거래 과정의 효율성 증대

실시간 채팅 시스템

- 안전한 거래 환경 구축

포인트 보증금 시스템

- 부동산 시장 데이터에 기반한 트렌드 분석 제공

입지 정보 제공

03 기획 및 설계

방이지는 어떻게 시작되었나






03

WHY BANGEZ

프로젝트 개발 과정 및 일정

1 개발 환경 구축 (7/10 ~ 7/14)

프로젝트의 기초를 다지는 단계로, 팀원들의 개발 환경을 통일하고 필요한 도구와 라이브러리를 설치했습니다. Docker 컨테이너 설정, 버전 관리 시스템 구축, 그리고 개발 서버 세팅을 완료했습니다.

2 본격적인 개발 (7/15 ~ 8/2)

백엔드와 프론트엔드 개발이 본격적으로 진행되었습니다. RESTful API 설계 및 구현, 데이터베이스 설계 및 구축, 그리고 핵심 기능들의 개발이 이루어졌습니다.

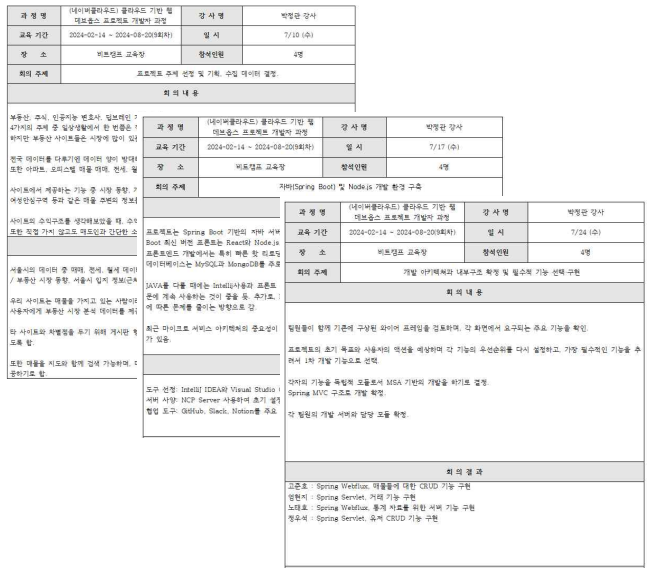
3 오류 수정 및 멀티모듈 구축 (8/3 ~ 8/6)

8/3 ~ 8/6: 발견된 버그를 수정하고 멀티모듈을 구축하여 실제 서비스 배포를 준비했습니다.

4 최적화 및 CI/CD 구축 (8/7 ~ 8/10)

성능 개선과 사용자 경험 향상을 위한 최종 최적화 작업을 진행하고, 지속적인 통합 및 배포를 자동화 하였습니다.

프로젝트 회 의 록



과 정 명	(네이버클라우드) 클라우드 기반 웹 미트숍스 프로젝트 개발자 과정	강 사 명	박정근 강사
교육 기간	2024-02-14 ~ 2024-08-20(94회차)	일 시	7/10 (수)
장 소	비트캠프 교육장	참석인원	4명
회의 주제	프로젝트 주제 선정 및 기획, 수집 데이터 결정.		
회 의 내 용			

과 정 명	(네이버클라우드) 클라우드 기반 웹 미트숍스 프로젝트 개발자 과정	강 사 명	박정근 강사
교육 기간	2024-02-14 ~ 2024-08-20(94회차)	일 시	7/17 (수)
장 소	비트캠프 교육장	참석인원	4명
회의 주제	자바(Spring Boot) 및 Node.js 개발 환경 구축		
회 의 내 용			

프로젝트는 Spring Boot 기반의 자바 서버 Boot 최신 버전 프론트는 React와 Node.js 프론트엔드 개발에서는 특히 빠른 핫 리로딩 데이터베이스는 MySQL과 MongoDB를 주로

JAVA를 다룰 때에는 IntelliJ사용과 프론트 문에 계속 사용하는 것이 좋을 듯. 추가로, 의 따른 문제를 줄이는 방향으로 갑.

최근 마이크로 서비스 아키텍처의 중요성이 가 있음.

도구 선정: IntelliJ IDEA와 Visual Studio 서버 사양: NCP Server 사용하여 초기 셀 협업 도구: GitHub, Slack, Notion을 주로

과 정 명	(네이버클라우드) 클라우드 기반 웹 미트숍스 프로젝트 개발자 과정	강 사 명	박정근 강사
교육 기간	2024-02-14 ~ 2024-08-20(94회차)	일 시	7/24 (수)
장 소	비트캠프 교육장	참석인원	4명
회의 주제	개발 아키텍처와 내부구조 확정 및 필수적 기능 선택 구현		
회 의 내 용			

팀원들이 함께 기존에 구상된 와이어 프레임을 검토하여, 각 화면에서 요구되는 주요 기능을 확인.

프로젝트의 초기 목표와 사용자의 액션을 예상하여 각 기능의 우선순위를 다시 설정하고, 가장 필수적인 기능을 추려서 1차 개발 기능으로 선택.

각자의 기능을 독립적 모듈로서 MSA 기반의 개발을 하기로 결정.
Spring MVC 구조로 개발 확정.

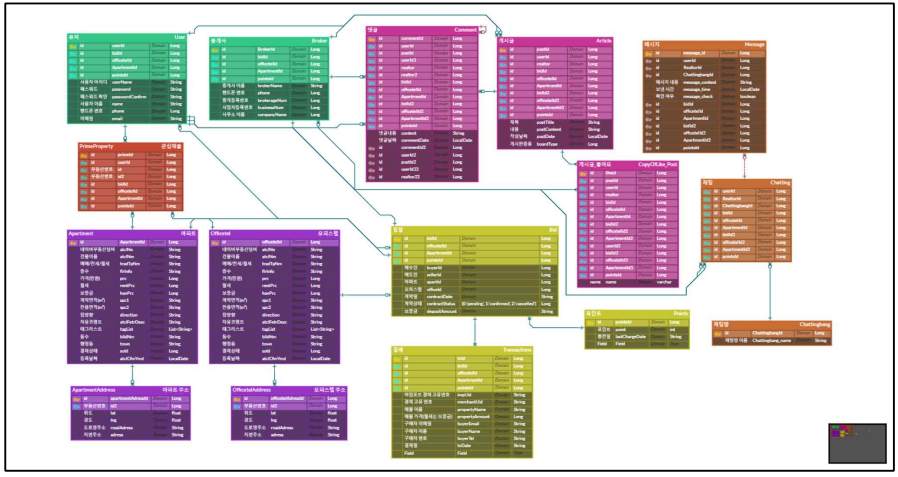
각 팀원의 개발 서버와 담당 모듈 확정.

회 의 결 과

고준호 : Spring Webflux, 매물들에 대한 CRUD 기능 구현
임현지 : Spring Servlet, 거래 기능 구현
노태호 : Spring Webflux, 통계 지표를 위한 서버 기능 구현
정우석 : Spring Servlet, 유저 CRUD 기능 구현

03

네이밍 컨벤션 가이드 & ERD



종류	형식	규칙	예시
클래스(Class)	PascalCase	클래스 이름은 항상 대문자로 시작하고, 각 단어의 첫 글자도 대문자로 작성해야 함. 클래스 이름은 명사로 작성해야 하며, 코드에서 나타내려는 객체의 의미를 잘 반영해야 함.	PropertyListing UserAccount TransactionHistory
인터페이스 (Interface)	PascalCase	인터페이스 이름은 클래스와 동일하게 PascalCase 형식을 사용해야 함.	Searchable Persistable Sortable
메서드(Method)	camelCase	메서드 이름은 소문자로 시작하고, 그 다음 단어부터는 각 단어의 첫 글자를 대문자로 작성해야 함. 메서드 이름은 동사로 시작하며, 메서드가 수행하는 동작을 명확히 나타내야 함.	getPropertyDetails sendNotification
변수(Variable)	camelCase	변수 이름은 소문자로 시작하고, 이후 단어는 각 단어의 첫 글자를 대문자로 작성해야 함. 변수 이름은 의미 있고 명확하게 작성해야 하며, 단어를 줄이거나 약어를 사용하는 것은 피해야 함.	propertyPrice userEmail listingDate
상수(Constant)	UPPER_SNAKE_CASE	상수는 모든 문자를 대문자로 작성하고, 단어 사이는 밑줄(_)로 구분해야 함. 상수는 변경되지 않는 값을 표현해야 하며, 명확하고 직관적인 이름을 사용해야 함.	MAX_LISTINGS DEFAULT_COMMISSION_RATE DATABASE_URL
패키지 (Package)	소문자와 도메인 네임을 기반으로 한 구조 사용	패키지 이름은 모두 소문자로 작성해야 하며, 도메인 네임의 역순으로 구성해야 함. 일반적으로 회사 이름과 프로젝트 이름을 포함하며, 하위 패키지로 기능이나 모듈을 구분함.	com.bangez.property
기타 네이밍 규칙		약어: 약어는 가능하면 사용하지 말아야 함. 약어를 사용해야 하는 경우, 일관성 있게 사용해야 함. 불필요한 단어: 변수명이나 메서드명에서 get, set, is와 같은 접두사는 필요할 때만 사용하고, 불필요한 단어는 생략해야 함. 명확성: 이름은 가능하면 짧지만 명확하게 작성해야 하며, 의미를 쉽게 이해할 수 있도록 작성해야 함.	

03

API 명세서

API 명세서

서버	Controller	Aa 기능	HTTP 메서드	API Path	Request	Response
analysis	officetel_rent	월별 오피스텔 평균 전세 가격 조회	GET	/api/officetel_rent/statistics	officetel_rent-Request	officetel_rent_Response
analysis	officetel_rent	날짜, 지역에 따른 오피스텔 전, 월세 거래량 조회	GET	/api/officetel_rent/statistics	officetel_rent-Request	officetel_rent_Response
analysis	city_park	공원 정보 조회	GET	/api/city_park/statistics		city_park_Response
analysis	school	학교 정보 조회	GET	/api/school/statistics		school_Response
land	apartment	모든 아파트 목록 조회	GET	/api/apartments		apartment_GET_Response
land	apartment	특정 ID의 아파트 정보 업데이트	PUT	/api/apartments/{id}	apartment_PUT_Request	apartment_PUT_Response
land	officetel	새로운 오피스텔 생성	POST	/api/officetels	officetel_Post_Request	
land	officetel	특정 ID의 오피스텔 정보 업데이트	PUT	/api/officetels/{id}	officetel_PUT_Request	officetel_PUT_Response
land	officetel	특정 ID의 오피스텔 정보 삭제	DELETE	/api/officetels/{id}		officetel_Delete_Response
TX	tx	포인트 적립 및 결제 정보 저장	POST	/add/{imp_uid}/{userId}		tx_add_Response
TX	tx	사용자 결제 리스트	GET	/list/{userId}		tx_list_Response
TX	tx	결제 정보 조회	GET	/detail/{id}		tx_detail_Response
TX	point	잔액 포인트 조회	GET	/point/detail/{userId}		point_detail_Response
TX	point	포인트 차감	PUT	/point/deduction/{userId}		point_deduction_Response
chat	room	채팅방 생성	POST	/room/open-room	open_room_Request	room_open_Response
chat	room	채팅방 목록	GET	/room/get-room-list/{userId}		room_get_list_Response
chat	room	채팅방 삭제	DELETE	/room/delete-room/{roomId}		room_delete_Response

apt_trade_Request

API 분류: API Response

태그: analysis

Query 월별 아파트 평균 가격 조회

```
?select=1
```

Query 날짜, 지역에 따른 아파트 거래량 조회

chat_save_Response

API 분류: API Response

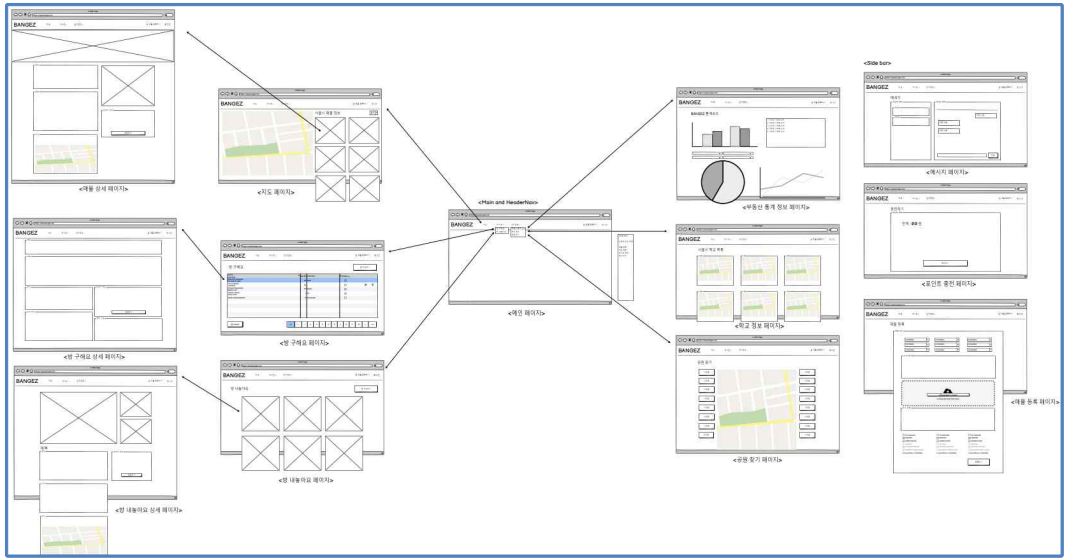
태그: [chat]

Body

```
{
    "id": "66bc7491f293056833fe92fd",
    "chatRoomId": "66bc735ef293056833fe92fb",
    "senderId": "300",
    "receiverId": "1",
    "message": "테스트입니다",
    "timeStamp": "2024-08-14T18:10:41.385716532",
    "read":

    }
```

03

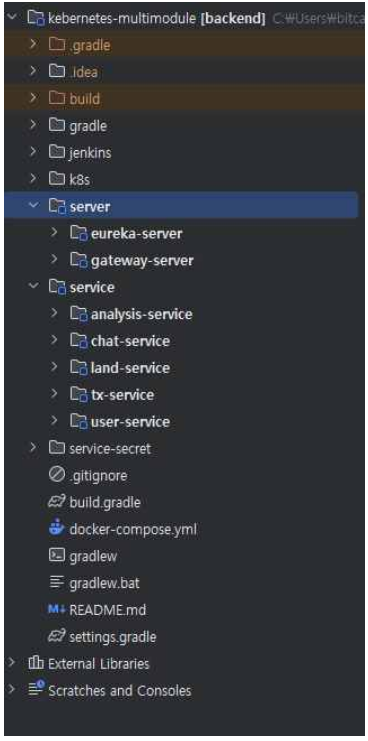
와이어 프레임

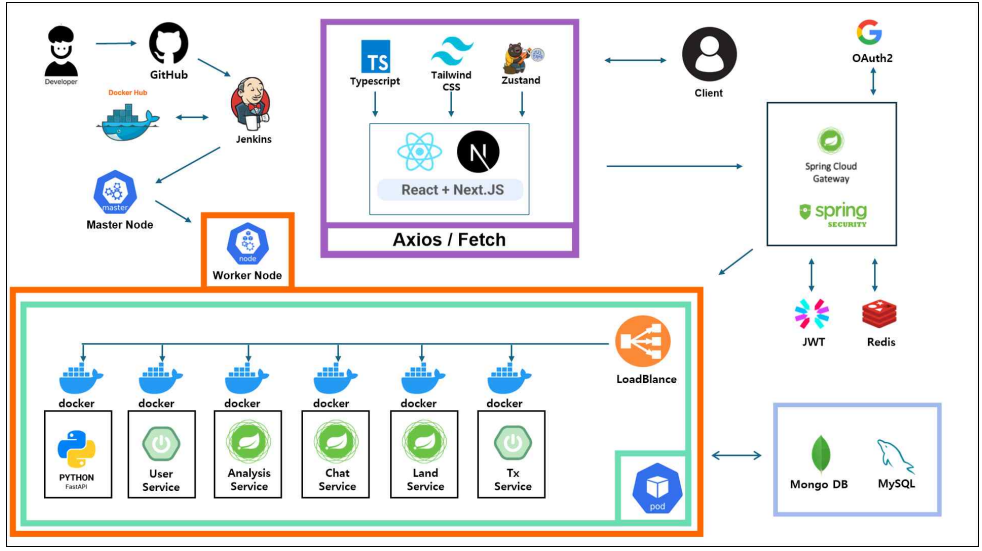


<웹 페이지 개발에 앞서 와이어 프레임을 만들어 UI 및 핵심 기능을 나타냄>

03

시스템 아키텍처 & 멀티모듈





04

ETL 과정
(Extract - Transform - Load)

데이터를 추출하고 전처리하여 데이터 베이스에 적재하는 과정

데이터 크롤링 및 정규화

```python
soup = str(BeautifulSoup(res.text, 'lxml'))

# filter 값 추출
value = soup.split("filter: {")[1].split("}")[0].replace(" ","").replace("'","")

print(value)

lat = (float)(value.split("lat:")[1].split(",")[0])
lon = (float)(value.split("lon:")[1].split(",")[0])
z = (int)(value.split("z:")[1].split(",")[0])
cortarNo = value.split("cortarNo:")[1].split(",")[0]
rletTpCds = value.split("rletTpCds:")[1].split(",")[0]
tradTpCds = value.split("tradTpCds:")[1].split()[0]

btm, lft, top, rgt = calculate_bounding_box(lat, lon, z)
print(f"btm: {btm}, lft: {lft}, top: {top}, rgt: {rgt}")

clusterList_URL = "https://m.land.naver.com/cluster/clusterList?view=atcl&cortarNo={}&
{}&btm={}&lft={}&top={}&rgt={}"\
    .format(cortarNo, rletTpCds, tradTpCds, z, lat, lon, btm, lft, top, rgt)
print(clusterList_URL)

res2 = requests.get(clusterList_URL, headers=headers)
res2.raise_for_status()

try:
    data = res2.json()
    # print(json.dumps(data, indent=2, ensure_ascii=False))
except json.JSONDecodeError as e:
    print("JSON 데이터를 파싱할 수 없습니다:", e)
    print("응답 내용:", res2.text)
```

<네이버 부동산에서 매물 데이터 크롤링 및 정규화 진행>

파이썬의 BeautifulSoup과 Selenium을 사용하여 동적이고 복잡한 웹사이트에서 필요한 데이터 수집

지오코더를 활용하여 주소와 위,경도 변환

결측값을 제거하고 데이터 정제 및 전처리

atclNm	rletTpNm	tradTpNm	flrInfo	prc	rentPrc	hanPrc	spc1	spc2	direction	atclCfmYmd	lat	lng	atclFetrDesc	tagList	bildNm	towr
광진캠퍼스시티	오피스텔	매매	17/20	15000.0	0.0	1억 5,000	32	16.08	북향	2024-08-09T00:00:00	37.547831	127.067537	NaN	['10년이내', '화장실한개', '소형평수', '방한개']	1동	광진 구 화 양동
아르페온1	오피스텔	매매	3/10	9000.0	0.0	9,000	24	14.96	북향	2024-08-07T00:00:00	37.565060	127.173084	갭투자가,상업지역소재생활편리,깨끗한풀옵션,빠른교통환경강남외곽고속뷰양호	['10년이내', '소형평수', '방한개', '화장실한개']	1동	강동 구 강 일동
건대트레비앙	오피스텔	매매	15/18	28900.0	0.0	2억 8,900	36	17.46	북향	2024-08-07T00:00:00	37.547654	127.071395	세안고 매매, 풀옵션, 시원한뷰, 강남진입전철10분	['2년이내', '역세권', '복층', '소형	1동	광진 구 화 양동

04

Extract - Transform - Load 구조 설계

DATA 공공데이터포털 .GO.KR

Open API

나이스 교육정보개방 서비스

Open API

데이터 파싱

FastAPI

PYTHON

```
@app.on_event("startup")
async def startup():
    start_scheduler()
    await save_mongodb()
```

데이터 전처리

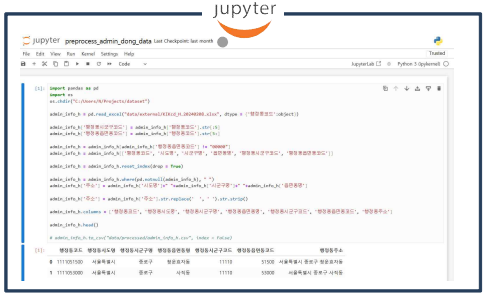
static_data

legal_address.csv

legal_info_b_seoul.csv

Mongo DB

jupyter

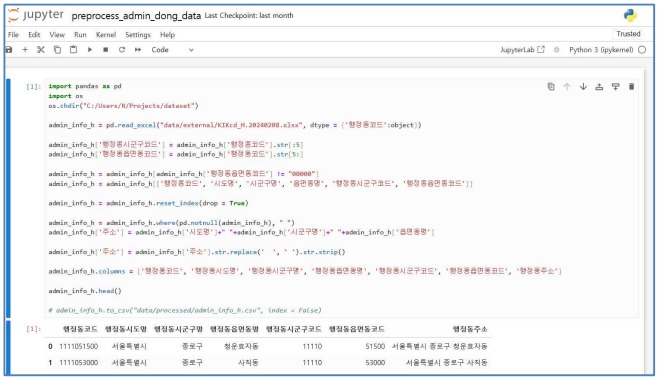


<법정동 데이터 전처리 후 파일 저장>

04

데이터 자동업데이트서버 구축과정(1)

1 업데이트가 필요 없는 법정동 데이터 전처리 후 저장



2 데이터 파싱(공공 데이터 API, Neis API)

```
async def officetel_rent_parsing():

    df = pd.read_csv(data_path)

    current = datetime.datetime.now()
    deal_ymd = current.strftime('%Y%m')

    LAWD_CD_list = df['법정동시군구코드'].unique()

    api_key = API_KEY

    column_nm = ['갱신요구권사용', '건축년도', '계약구분', '계약기간', '년', '단지', '법정동',
                 '보증금', '시군구', '월', '월세', '일', '전용면적', '종전계약보증금',
                 '종전계약월세', '지번', '지역코드', '층']

    total = pd.DataFrame()

    for i in range(len(LAWD_CD_list)):
        url = 'http://openapi.molit.go.kr/OpenAPI_ToolInstallPackage/service/rest/RTMSOBJSvc/getRTMSDataSvcOffiRent'
        params = {'serviceKey': api_key, 'LAWD_CD': LAWD_CD_list[i], 'DEAL_YMD': deal_ymd}

        res = requests.get(url, params)
        soup = bs(res.text, features='xml')
        items = soup.find_all('item')

        for k in range(len(items)):
            df_raw = []
            for j in column_nm:
                try:
                    df_raw.append(items[k].find(j).text)
```

04

데이터 자동 업데이트 서버 구축 과정(2)

3 데이터 전처리 후 MongoDB에 저장

```python
async def officetel_rent_preprocess(parsing_data: pd.DataFrame):

    legal_info_b_seoul = pd.read_csv(data_path).astype({'법정동코드': str, '동리명': str})

    officetel_rent = parsing_data

    officetel_rent = officetel_rent[
        ['건축년도', '계약기간', '법정동', '보증금', '시군구', '월세', '전용면적', '지번',
         '층', '지역코드', '년', '월', '일', '단지']]
    officetel_rent = officetel_rent.copy()
    officetel_rent.rename(columns={'지역코드': '법정동시군구코드'}, inplace=True)
    officetel_rent.rename(columns={'법정동': '읍면동명'}, inplace=True)

    officetel_rent = officetel_rent[officetel_rent['건축년도'].notnull()]
    officetel_rent = officetel_rent.astype({'읍면동명': str})
```

```python
async def start_save_officetel_rent(parsing_data: list):
    await officetelRents_collection.insert_many(parsing_data)
```

<이 후 설계 내용>
서버 실행 시
아파트 매매 / 아파트 전·월세
오피스텔 매매 / 오피스텔 전·월세
거래 데이터가 현재 시점으로 1년치 각 Collection으로 저장

학교 / 공원 정보 저장

04

데이터 자동 업데이트 서버 구축 과정(3)

4 Scheduler를 이용한 배치 시스템 구축

```python
app = FastAPI()


@app.get("/")
async def read_root():
    return {"Hello": "World"}


@app.on_event("startup")
async def startup():
    start_scheduler()
    await save_mongodb()


@app.on_event("shutdown")
def shutdown():
    shutdown_scheduler()


if __name__ == '__main__':
    import uvicorn

    uvicorn.run(app:"your_script:app", reload=True, host="0.0.0.0", port=8086, log_level="info")
```

매일 새벽에 1시부터 작동되도록 했으며,
특히 중복 데이터를 제외하고 업데이트 되도록 구축

```python
def start_scheduler():
    scheduler.add_job(schedule_apt_rent, trigger:'cron', hour=16, minute=0)
    scheduler.add_job(schedule_apt_trade, trigger:'cron', hour=16, minute=5)
    scheduler.add_job(schedule_officetel_rent, trigger:'cron', hour=16, minute=10)
    scheduler.add_job(schedule_officetel_trade, trigger:'cron', hour=16, minute=15)
    scheduler.start()
    print("스케줄러 시작")
```

```python
async def schedule_save_apt_rent(parsing_data: list):
    for data in parsing_data:
        existing_data = await aptRents_collection.find_one({
            "apt_name": data["apt_name"],
            "contract_date": data["contract_date"],
            "floor": data["floor"],
            "security_deposit": data["security_deposit"]
        })
        if not existing_data:
            await aptRents_collection.insert_one(data)
```

04

데이터 자동 업데이트 서버 구축 과정(4)

5 서버 동작 확인

도커허브 이미지를 이용하여 네이버 클라우드에 도커 컨테이너로 구축

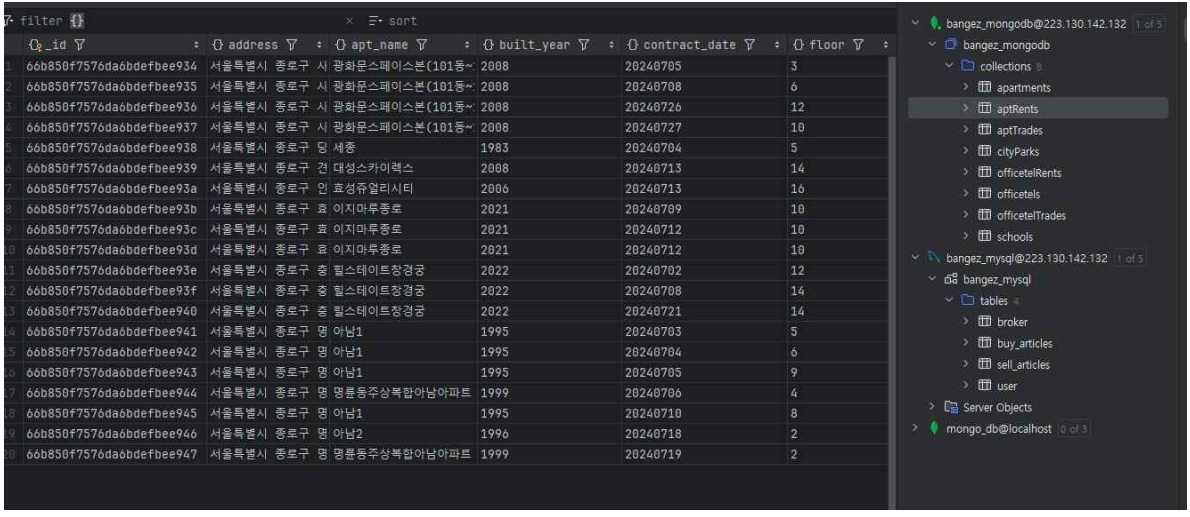
```
INFO:     Will watch for changes in these directories: ['/app']
INFO:     Uvicorn running on http://0.0.0.0:8086 (Press CTRL+C to quit)
INFO:     Started reloader process [1] using StatReload
INFO:     Started server process [15]
INFO:     Waiting for application startup.
스케줄러 시작
save_mongodb 시작
collections 읽어들임
202408 apt_rent save success
202407 apt_rent save success
202406 apt_rent save success
202405 apt_rent save success
202404 apt_rent save success
202403 apt_rent save success
202402 apt_rent save success
202401 apt_rent save success
```

```
202306 officetel_trade save success
202307 officetel_trade save success
202308 officetel_trade save success
202309 officetel_trade save success
202310 officetel_trade save success
202311 officetel_trade save success
202312 officetel_trade save success
city_park save success
school save success
save_mongodb 완료
INFO:     Application startup complete.
schedule 시작
schedule_apt_trade 데이터 업데이트 완료:  2024-08-11 16:05:08.280049
schedule_apt_rent 데이터 업데이트 완료:  2024-08-11 16:08:04.419618
schedule_officetel_rent 데이터 업데이트 완료:  2024-08-11 16:10:09.308512
schedule_officetel_trade 데이터 업데이트 완료:  2024-08-11 16:15:04.817292
```

04

데이터 자동 업데이트 서버 구축 과정(5)

6 데이터 베이스 데이터 업데이트 확인



05 코드 설명 세션

방이지 프로젝트에 사용된 실제 구현 코드를 소개하겠습니다.

05

입지 정보 및 통계정보 (백엔드)

- **특징 : 사용자에게 정보 제공**

백엔드 코드 설명

- 보안보다는 속도나 트래픽이 더 중요한 부분으로 판단하여 Controller 에서 라우터 구현
  Spring Webflux에 MongoReactiveOperations를 이용해 서버의 처음부터 끝까지 비동기를 구현

```
@GetMapping(path = "/statistics")
public ResponseEntity<?> searchPlayer(
        @RequestParam(value = "select", required = true) String select,
        @RequestParam(value = "date", required = false) String date,
        @RequestParam(value = "region", required = false) String region
) {
    Mono<?> monoMap = router.execute(select, date, region);

    return ResponseEntity.ok(monoMap);
}
```

```
@Component
@RequiredArgsConstructor
public class AptTradeRouter {
    private final AptTradeDAORepositoryImpl repository;

    1 usage   "noh_taeho" +1
    public Mono<?> execute(String select, String date, String region){

        return switch (select){
            case "1" -> repository.plotGraphAvgCostByDate();
            case "3" -> repository.plotGraphSalesCountByRegionForMonth(date, region);
            case "7" -> repository.tradeCountRaiseTop5ForMonth();

            default -> null;
        };
    }
}
```

```
@Repository
@RequiredArgsConstructor
public class AptTradeDAORepositoryImpl implements AptTradeDAORepositoty {

    private final ReactiveMongoOperations operations;

    "noh_taeho"
    public Mono<Map<String, Long>> plotGraphAvgCostByDate() {
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyyMM");

        Criteria criteria = Criteria.where( key: "contract_date")
                .gte( value: YearMonth.now().minusYears( yearsToSubtract: 1).format(formatter)+"01").lte( value: YearMonth.now().format(

        Query query = new Query(criteria);
        query.fields().include( field: "contract_date").include( field: "trade_price");

        return operations.find(query, AptTrade.class)
                .flatMap(i-> Mono.just(AptTrade.builder()
                                .contractDate(i.getContractDate().substring(0, 6))
                                .tradePrice(i.getTradePrice())
                                .build()))
                .collect(Collectors.groupingBy(i->i.getContractDate(),
                        Collectors.collectingAndThen(Collectors.averagingLong(i->i.getTradePrice()), i-> i.longValue())));
    }
```

05

입지 정보 및 통계정보(프론트엔드)

▶ **목적 : 사용자에게 정보 제공**

- Nextjs 14의 Fetch를 이용한 통신
- React-Chartjs 2 이용하여 그래프 표시

서울시 학교 목록

현재 용산구의 고등학교를 보는 중 입니다.

행정구 ⌄ 학교종류 ⌄

배문고등학교

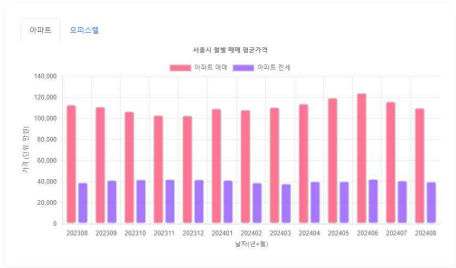
서울특별시 용산구 효창원로 258

홈페이지 바로가기

```
export async function SalesCountByRegionForMonth(date:string, propertyType:string, region:string) {
    const url = `${API.ANALYSERVER}/api/${propertyType}/statistics?select=3&date=${date}&region=${region}`
    try {
        const response = await fetch(url, {
            method: "GET",
        });
        if (!response.ok){
            throw new Error(`HTTP error! Status: ${response.status}`);
        }
        const data = await response.json();
        return data;
    } catch (error){
        console.error("fetch select 3 실패 :", error);
    }
}
```

BANGEZ 통계보드

최종 업데이트 날짜 : 2024년 8월 12일



최근 1년간 아파트 거래량 순위

순위	아파트	거래량
1등	힐리오시티 (송파구)	346
2등	파크리오 (송파구)	269
3등	고덕그라시움 (강동구)	263
4등	리센츠 (송파구)	176
5등	고덕아르테온 (강동구)	174

05

User CRUD 서비스 (백엔드)

▶ 목적 : 사용자가 직접 게시판에 구매/판매 글을 올려 직거래를 유도

```java
@PostMapping("/save")  JWOOSAGI
public ResponseEntity<MessengerVO> save(@RequestBody BuyArticleDTO dto) {
    log.info("입력받은 정보 : {}", dto);
    return ResponseEntity.ok(service.save(dto));
}

@DeleteMapping(path = "/delete")  JWOOSAGI
public ResponseEntity<MessengerVO> deleteById(@RequestParam Long id)  {
    log.info("입력받은 정보 : {}", id );
    return ResponseEntity.ok(service.deleteById(id));
}

@GetMapping(path = "/list")  JWOOSAGI *
public ResponseEntity<List<BuyArticleDTO>> findAll() {
    return ResponseEntity.ok(service.findAll());
}

@PatchMapping(path = "/update/{id}")  JWOOSAGI
public ResponseEntity<BuyArticle> modify(@PathVariable Long id, @RequestBody BuyArticle newBuyArticle){
    log.info("입력받은 정보 : {}", newBuyArticle);
    return ResponseEntity.ok(service.modify(id, newBuyArticle));
}

@GetMapping(path = "/detail")  JWOOSAGI
public ResponseEntity<Optional<BuyArticleDTO>> findById(@RequestParam Long id) {
    log.info("입력받은 정보 : {}", id );
    return ResponseEntity.ok(service.findById(id));
}
```

```java
@Override  1 usage  JWOOSAGI *
public Optional<BuyArticleDTO> findById(Long id) {
    // 게시글 조회 시 조회수 증가
    repository.incrementBoardHits(id);
    return repository.findById(id).map(this::entityToDTO);
}
```

```java
@Modifying  1 usage  new *
@Query("UPDATE buyArticles b SET b.boardHits = b.boardHits + 1 WHERE b.id = :id")
void incrementBoardHits(Long id);
}
```

백엔드 코드 설명

- 좌측 코드의 위에서부터 Create, Delete, Read, Update 기능 controller.
- 우측 상단 코드는 게시글 클릭해 상세하기 조회시, 조회수가 증가하는 로직.
- 우측 2번째 코드는 Repository에서 jpql을 통한 조회수 증가

05

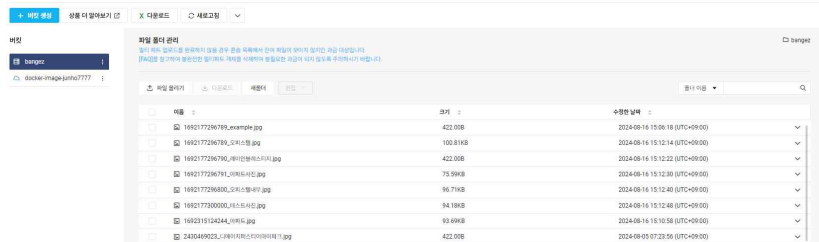
NCP Object Storage 파일 업로드 서비스

▶ 목적 : 사용자에게 매물 이미지 업로드 기능 제공

```java
@Service
public class S3Service {

    private final AmazonS3 s3Client;
    private final String bucketName;
    final String endPoint = "https://kr.object.ncloudstorage.com";
    final String regionName = "kr-standard";

    public S3Service(
            @Value("${cloud.ncp.credentials.accessKey}") String accessKey,
            @Value("${cloud.ncp.credentials.secretKey}") String secretKey,
            @Value("${cloud.ncp.s3.bucket}") String bucketName

    ) {
        this.s3Client = AmazonS3ClientBuilder.standard()
                .withEndpointConfiguration(new AwsClientBuilder.EndpointConfiguration
                (endPoint, regionName))
                .withCredentials(new AWSStaticCredentialsProvider(new BasicAWSCredentials
                (accessKey, secretKey)))
                .build();
        this.bucketName = bucketName;
    }

    public String uploadFile(MultipartFile file) throws IOException {
        File convertedFile = convertMultipartFileToFile(file);
        String fileName = System.currentTimeMillis() + "_" + file.getOriginalFilename();

        s3Client.putObject(new PutObjectRequest(bucketName, fileName, convertedFile));
        convertedFile.delete();

        return s3Client.getUrl(bucketName, fileName).toString();
    }
```

```java
@PostMapping("/upload")
public Mono<String> uploadImage(@RequestParam("file") MultipartFile file) {
    try {
        String imageUrl = s3Service.uploadFile(file);
        return Mono.just(imageUrl);
```

Bucket Management



AWS S3에서 제공하는 Java용 SDK를 이용해 네이버 클라우드 플랫폼의 Object Storage를 사용

파일 업로드가 성공적으로 완료되면, 해당 파일의 URL을 반환

05

KAKAO MAP API 시스템

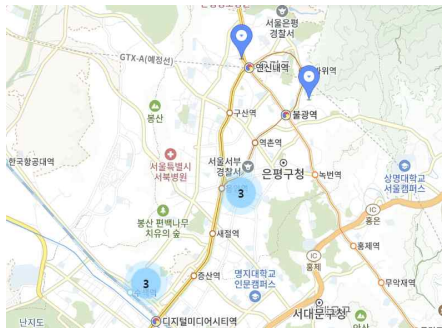
▶ **목적 : 사용자에게 매물 위치를 지도를 통해 제공**

```
const Map: React.FC<MapProps> = ({ properties }) => {
  useEffect(() => {
    const script = document.createElement('script');
    script.id = 'kakao-map-script';
    script.src = `//dapi.kakao.com/v2/maps/sdk.js?appkey=${process.env.NEXT_PUBLIC_MAP_KEY}&
    libraries=clusterer&autoload=false`;
    script.async = true;

    script.onload = () => {
      window.kakao.maps.load(initializeMap);
    };

    script.onerror = () => {
      console.error('Failed to load the Kakao map script.');
    };

    document.head.appendChild(script);
  } else {
    window.kakao.maps.load(initializeMap);
  }
}, []);

useEffect(() => {
  if (mapInstance.current && clustererInstance.current) {
    const map = mapInstance.current;
    const clusterer = clustererInstance.current;

    overlayInstance.current.forEach((overlay) => overlay.setMap(null));
    overlayInstance.current = [];

    const markers = properties.map((property, index) => {
      const markerPosition = new window.kakao.maps.LatLng(parseFloat(property.lat), parseFloat(property.
      lng));
      const marker = new window.kakao.maps.Marker({
```

카카오 맵 API를 이용한 동적 지도 로딩 및 마커 표시

React의 useEffect를 사용하여 컴포넌트가 마운트되면 지도에 마커를 동적으로 추가. 각 마커는 경도와 위도 데이터를 기반으로 생성.

지도 상의 마커들이 밀집되어 있을 때 이를 그룹화 (클러스터링)



05

MSA, k8s, Jenkins 설계 및 구축

▶ **목적 :** Ncloud Kubernetes 와 Jenkins를 통한 자동화 CI/CD 파이프라인 구축

Subnet 이름	Subnet ID	상태	VPC 이름	IP 주소 범위	Zone	Internet Gateway 전용 여부	용도
bangez-private-l...	165538	운영중	bangez	192.168.254.0/...	KR-1	N (Private)	LoadBalancer
bangez-pub-lb-...	165505	운영중	bangez	192.168.255.0/...	KR-1	Y (Public)	LoadBalancer
bangez-k8s-sub...	165504	운영중	bangez	192.168.1.0/24	KR-1	Y (Public)	일반
bangez-vpc-sub...	165503	운영중	bangez	192.168.0.0/24	KR-1	Y (Public)	일반

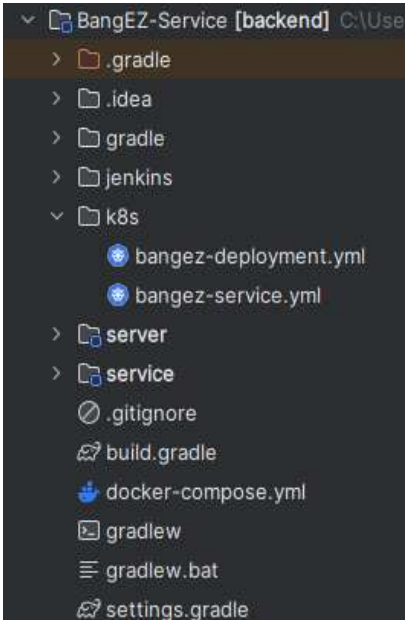
서버 이름 (Instance ID)	하이퍼바이저	서버 세대	서버 이미지 이름	서버 이미지 번호	서버 스펙	상태	비공인 IP	공인 IP	VPC	Subnet
bangez-node...	KVM	G3	ubuntu-...	23215604	s2-g3	...	192.168.1.6	223....	ban...	bangez...
bangez-confi...	KVM	G3	ubuntu-...	23214590	c2-g3	...	192.168.0.6	223....	ban...	bangez...

Ncloud에서 Public, Private, k8s, VPC 서브넷을 생성하여 네트워크 환경을 구성
각각의 서브넷에서 필요한 서버와 서비스들이 독립적으로 운영
Config 서버와 Kubernetes 서비스 클러스터를 구성.
노드풀 서버와 Config 서버로 구성

05

MSA, k8s, Jenkins 설계 및 구축

▶ **목적 : Ncloud Kubernetes 와 Jenkins를 통한 자동화 CI/CD 파이프라인 구축**



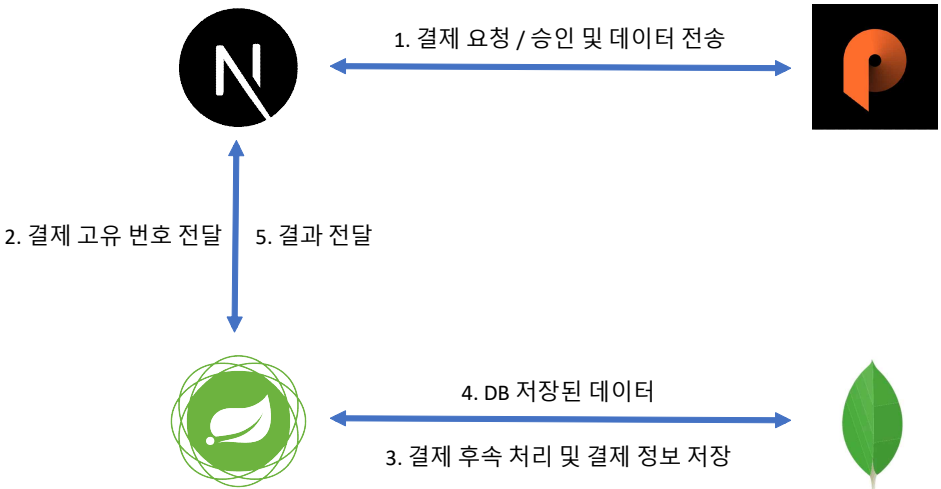
Jenkins와 GitHub repository 연동하여 코드 변경 사항이 발생하면, Jenkins가 자동으로 빌드 및 테스트를 수행함.
Webhook을 활용해 GitHub에 코드가 push되면 Jenkins에서 CI/CD 파이프라인이 실행되고, 코드 변경이 성공적으로 통과 되면 Kubernetes 클러스터에 변경 사항을 자동으로 배포함.

각 MSA 서비스에 대해 kubernetes deployment파드를 구성했음. 이를 통해 각 서비스는 kubernetes 클러스터에서 독립적인 배포 및 관리가 가능.

05

IamPort API 카카오 결제 시스템 (결제 Flow)

▶ **목적 :** 사용자가 카카오톡 간편 결제로 결제할 수 있는 기능 제공



05

IamPort API 카카오 결제 시스템 (프론트엔드)

▶ **목적 :** 사용자가 카카오톡 간편 결제로 결제할 수 있는 기능 제공

```
IMP.request_pay(
  {
    pg: 'kakaopay',
    merchant_uid: `mid-${merchant_uid}`,
    name: `BangEZ 포인트 충전`,
    amount: `${data.amount}`,
    customer_uid: `cuid-${merchant_uid}`,
    buyer_email: user?.email,
    buyer_name: user?.name,
    buyer_tel: user?.phone,
  },
  async (rsp: any) => {
    console.log('결제 후 rsp', rsp);
    if (rsp.success) {
      alert('결제 성공');
      await fetch(`${API.TXSERVER}/add/${rsp.imp_uid}/${decodedToken.id}`, {
        method: "POST",
      }).then(res => res.json())
        .then(res => {
          console.log(res.response.amount);
          if (rsp.paid_amount === res.response.amount) {
            alert('결제 성공');
            setPoint(prev => prev + (res.response.amount / 1000) * 10);
          } else {
            alert('결제 실패: 금액 불일치');
          }
        }).catch(error => {
          console.error("결제 검증 실패 : ", error);
        });
    } else {
      alert(`결제 실패: ${rsp.error_msg}`);
    }
  });
```

프론트엔드 핵심 코드와 설명

사용자가 결제 요청 시 필요한 파라미터를 API로 전송 후,
결제가 성공됨에 따라 후속 처리를 진행함.

05

IamPort API 카카오 결제 시스템 (백엔드)

▶ **특징 :** 사용자가 카카오톡 간편 결제로 결제할 수 있는 기능 제공

```
@PostMapping("/add/{imp_uid}/{userId}")
public IamportResponse<Payment> addIamport(@PathVariable("imp_uid") String imp_uid,
                                           @PathVariable("userId") Long userId) {
    IamportResponse<Payment> payment = iamportClient.paymentByImpUid(imp_uid);
    txService.saveTx(payment, userId);
    pointService.savePoint(payment.getResponse().getAmount(), userId);
    return payment;
}
```

백엔드 핵심 코드와 설명

결제 성공 후 프론트엔드로부터 받은 결제 정보로
결제 검증 후 사용자에게 포인트를 적립.
IamPort에서 제공하는 Payment 객체로 결제 정보로
결제 ID, 상태, 금액 등을 포인트 및 결제 정보를
MySQL에 저장.

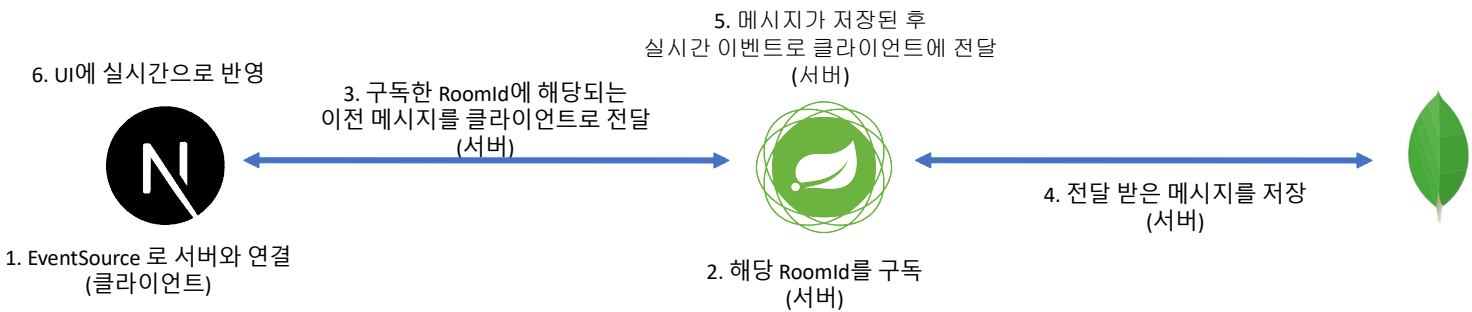
WHERE ORDER BY

apartment_id	id	officetel_id	property_amount	user_id	buyer_email	buyer_name	imp_uid	merchant_uid	p...	tx_date
<null>	9	<null>	1000	5	hyunji1443@gmail.com	엄현지	imp_420666378092	mid-1723363665567	BangEZ 포...	Sun Aug 11 17:08:10 KST 2024

05

SSE 활용 단방향 채팅 서비스 (채팅 Flow)

목적 : 단방향 채팅 서비스를 제공해 실시간으로 사용자 간 대화를 할 수 있도록 지원



05

SSE 활용 단방향 채팅 서비스 (프론트엔드)

▶ **특징 :** 단방향 채팅 서비스를 제공해 실시간으로 사용자 간 대화를 할 수 있도록 지원

```
const eventSource = new EventSourcePolyfill(`${API.CHATSERVER}/sse/${roomId}`, {
  headers: getAuthHeader(),
  withCredentials: true
});
eventSource.onopen = (event) => {
  console.log('Connection opened:', event);
};
eventSource.onmessage = (event) => {
  console.log('Message received:', event.data);
  setPrevMessages(prevMessage => [...prevMessage, JSON.parse(event.data)]);
};
eventSource.addEventListener('error', (e: any) => {
  console.log("An error occurred while attempting to connect.");
  console.error("Error event details:", e);
  console.error("EventSource readyState:", e.target.readyState);
  console.error("EventSource URL:", e.target.url);
  eventSource.close();
});
return () => {
  eventSource.close();
}
```

프론트엔드 핵심 코드와 설명

EventSource를 사용해 서버와 연결을 유지하고,
서버에서 발생하는 이벤트를 실시간으로 수신 받음

onmessage를 통해 서버에서 전송된 메시지를
클라이언트에 반영해 실시간 채팅을 구현함

addEventListener로 에러 처리,
에러 발생 시 연결을 종료

05

SSE 활용 단방향 채팅 서비스 (백엔드)

▶ **특징 :** 단방향 채팅 서비스를 제공해 실시간으로 사용자 간 대화를 할 수 있도록 지원

```java
@Override
public Flux<ServerSentEvent<ChatDto>> connectChat(String roomId) {
    log.info("connectChat service roomId : {}", roomId);

    Sinks.Many<ServerSentEvent<ChatDto>> sink = chatSinks.computeIfAbsent(roomId, key -> {
        log.info("Creating new sink for roomId : {}", roomId);
        Sinks.Many<ServerSentEvent<ChatDto>> chatSink = Sinks.many().replay().all();
        chatRepository.findByChatRoomId(roomId) Flux<ChatModel>
                .map(this::convertToDto) Flux<ChatDto>
                .map(chat -> ServerSentEvent.builder(chat).build()) Flux<ServerSentEvent<...>>
                .doOnNext(chatSink::tryEmitNext)
                .subscribe();
        return chatSink;
    });

    Flux<ServerSentEvent<ChatDto>> heartbeatFlux = Flux.interval(Duration.ofSeconds(30))
            .map(tick -> ServerSentEvent.<ChatDto>builder()
                    .event("heartbeat")
                    .data(new ChatDto())
                    .build());

    Flux<ServerSentEvent<ChatDto>> chatFlux = sink.asFlux()
            .mergeWith(heartbeatFlux);

    log.info("Existing sink for roomId : {}", roomId);
    return chatFlux.doOnCancel(() -> handleCancel(roomId));
}
```

백엔드 핵심 코드와 설명

Spring WebFlux를 사용해 SSE를 구현
클라이언트로부터 채팅방 연결 요청을 받으면 해당 채팅방 메시지를 실시간으로 스트리밍함.
Sinks.many().replay().all()로 새로운 구독자가 구독할 때, 이전 모든 메시지를 모두 받을 수 있도록 함

30초 간격으로 이벤트를 전송해 클라이언트와 서비스 간 연결 상태를 확인 및 유지 함

클라이언트 연결을 종료 시 doOnCancel을 통해 연결을 종료함.

05

SSE 활용 단방향 채팅 서비스 (백엔드)

▶ **특징 :** 단방향 채팅 서비스를 제공해 실시간으로 사용자 간 대화를 할 수 있도록 지원

```
67      @Override
68      public Mono<ChatDto> saveMessage(ChatDto chatDto) {
69          chatDto.setTimeStamp(LocalDateTime.now(ZoneId.of( zoneId: "Asia/Seoul")));
70          return chatRepository.save(convertToEntity(chatDto)) Mono<ChatModel>
71                  .map(this::convertToDto) Mono<ChatDto>
72                  .doOnSuccess(savedMessage -> {
73                      Sinks.Many<ServerSentEvent<ChatDto>> sink = chatSinks.get(savedMessage.getChatRoomId());
74                      if (sink != null) {
75                          sink.tryEmitNext(ServerSentEvent.builder(savedMessage).build());
76                      }
77                      if (!savedMessage.isRead()) {
78                          notifyUnreadMessage(savedMessage);
79                      }
80                  });
81      }
```

백엔드 핵심 코드와 설명

Spring WebFlux를 사용해 SSE를 구현
클라이언트로부터 채팅방 연결 요청을 받으면 해당 채팅방 메시지를 실시간으로 스트리밍함.

06 사용 기술

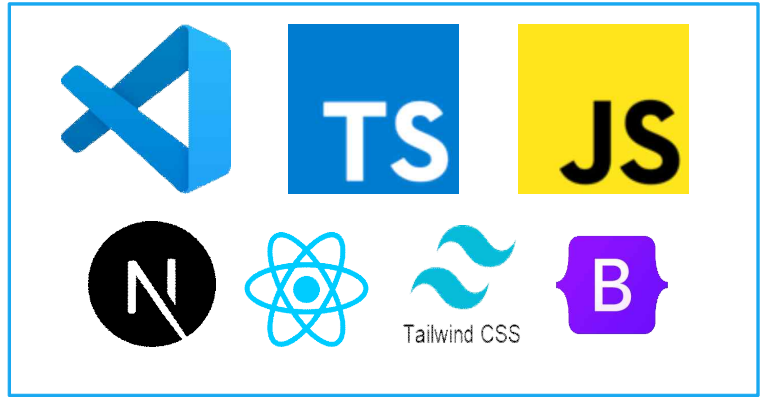
방이지는 어떠한 기술로 만들어 진 프로젝트인가요?

BangEZ Project

기술 소개

BE / Tools

Intellij , PyCharm , JAVA , Python , Gradle , Ubuntu , Kafka , Redis , MySQL , Mongo DB

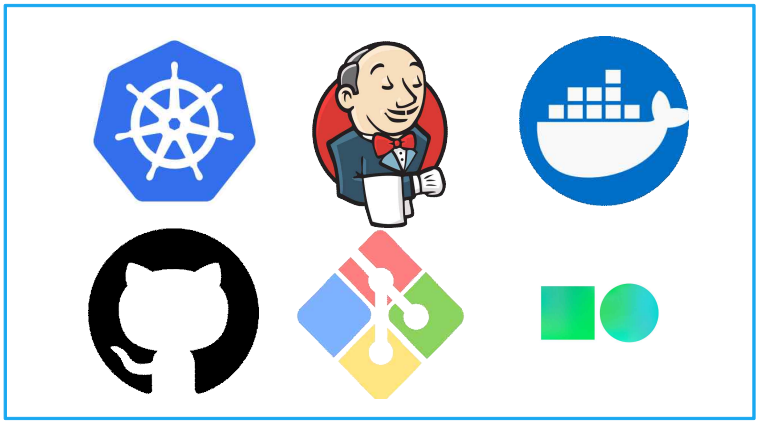


FE / Tools

VS Code , TypeScript , JavaScript , Next JS , React , Tailwind CSS , Bootstrap

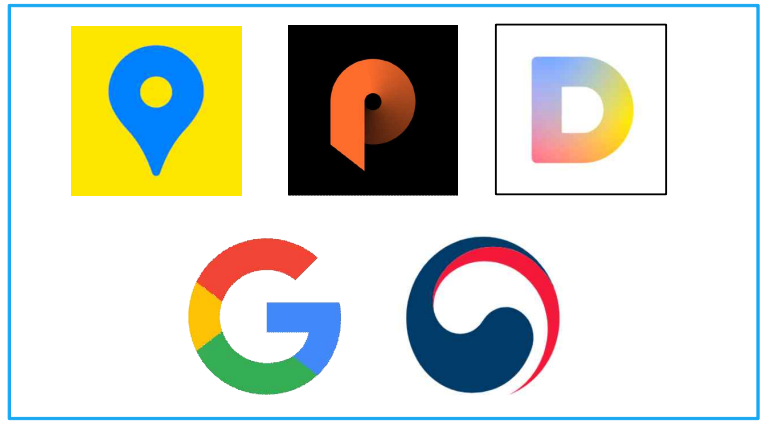
BangEZ Project

기술 소개



CI / CD

Kubernetess , Jenkins , Docker , GitHub , Git , Naver Cloud



API

KakaoMap, IamPort, Daum주소변환, Google login , 공공데이터포털

BangEZ Project

기술 소개

서버 개발	개발 도구 및 인프라	프론트엔드 기술 및 UI/UX
Spring Boot 3.3.1 + JDK 17 FastAPI 0.112.0 + Python 3.12.2 Uvicorn IntelliJ IDEA Ultimate PyCharm Professional VSCode Jupyter Notebook	Jenkins Docker Gradle Yarn Git GitHub Kubernetes Ubuntu SSH Gabia Notion NCP PostMan Swagger	Next.js 14 React 18 TypeScript JavaScript Sass Bootstrap Tailwind CSS **API 연동** IamPortAPI KakaoMapAPI 공공데이터포털API NeisAPI Daum주소변환API

07 시연 영상 시청

그렇게 만들어진 방이지.






07

BangEZ 시연 영상

08 프로젝트 후기

나의 방이지는 이러했다.






BangEZ Project

후기

고준호

프로젝트를 수행하면서 팀워크의 중요성을 깊이 깨달았다. 프로젝트의 성공은 개개인의 역량도 중요하지만 팀원 간의 원활한 소통과 협업이 크게 작용한다는 것을 경험하였다. 주기적인 회의와 피드백 세션을 통해 프로젝트 진행 상황을 공유하고, 문제점을 신속하게 해결할 수 있었다. 많은 경험과 경험과 교훈을 얻을 수 있는 프로젝트였다. 우리 팀원들에게 정말 고맙다.

노태호

경험이 있었다면 더 효율적인 개발 과정을 진행할 수 있었을거 같아 그 부분에 아쉬움이 있다. 이러한 상황에서도 함께 달려온 팀원들에게 고맙다. 이번 프로젝트를 통해 다양한 기술과 도구를 폭넓게 경험할 수 있었고, 이는 앞으로 큰 자산이 될 것이라고 생각한다. 다양한 프레임워크와 언어를 다뤄보면서 각 기술의 장단점을 직접 체감할 수 있었던 점이 매우 유익했다.

엄현지

프로젝트 제작 기간에 비해 처음 기획 의도와 비슷하게 결과물이 나왔다고 생각든다. 이번 프로젝트를 통해서 소통방식과 개발 과정에 대해서 많이 배웠다. 만약 프로젝트 기간이 길었다면 조금 더 완성도 높은 결과물이 나왔을거라고 생각해서 그 부분에 대해서는 조금 아쉽고, 1-2개월 가까이 함께 고생하고 힘써준 팀원들에게 정말 감사하다고 말하고싶다.

정우석

일단 마무리를 했다는 부분에서 우리 팀은 정말 잘했다. 다만 개인적으로는 내가 좀 더 빨리 완성할 수 있었고, 그랬다면 팀에게 도움을 주거나, 또 다른 기능을 추가 할 수 있었다는 부분에서 아쉬움이 남는다. 확실히 프론트 부분에서 부족함이 느껴졌다. 추후 프론트에서 입력 방식이나, 디자인 부분 등 에서 보완이 있었으면 좋겠다. 기획 의도와는 정확히 맞아 떨어지는 프로젝트 였으나 완성도 부분에서 다소 떨어진다 생각한다. 팀 프로젝트와 개인 프로젝트와는 다른 기술 등도 사용함에 있어 배움도 있었다.

THANK YOU FOR ATTENTION