**주택 실거래 정보 제공 웹 제작**

3조 개발자국

이병학

김동호

곽주영

윤영준

◎ 프로젝트 목표

- API를 이용하여 지도를 띄우고 건물과 거래정보 데이터를 표시

- thymeleaf를 사용하여 댓글과 대댓글 기능 제작

- ajax를 써서 페이지 내에 비동기 갱신되는 좋아요(즐겨찾기) 기능 제작

◎ 팀원 역할

이병학 : 백엔드 파일 제작, ppt/보고서 작성

김동호 : 프론트엔드 제작, 제작 과정 전체적인 검수

곽주영 : 프론트엔드, 웹 디자인 제작

윤영준 : DB 데이터 정리, 제작에 필요한 자료 조사

◎ 제작 과정

|  |
| --- |
| 03-13 프로젝트 시작, 전체적인 기능 구상 및 타 사이트의 자료 조사  03-14 페이지 구성 구상, 객체 제작, 조사한 데이터 정리  03-15 DAO, DB, Service, Controller 기초 구조 완성, 카카오 API를 이용하여 지도 제작  03-18 메인 페이지, 검색 페이지, 세부 정보 페이지 제작  03-19 메인 페이지 디자인 제작, 거래내역을 이용하여 차트 페이지 제작  03-20 댓글 / 좋아요 기능 구현, 페이지 기능 완성  03-21 사이트 디자인 완성 및 마무리 |

◎ 라우팅 테이블

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| method | PathVariable | input parameter | | viewer | | | model parameter | | session parameter |
| /main | | | | | | | | | |
| GET |  |  | | common/main.html | | |  | |  |
| /user/register | | | | | | | | | |
| GET |  |  | | user/register.html | | |  | |  |
| POST |  | uid, pwd, pwd2, uname, email, location, tel | | common/alertMsg.html | | | msg, url | |  |
| /user/login | | | | | | | | | |
| GET |  |  | | user/login | | |  | |  |
| POST |  | uid, pwd | | common/alertMsg.html | | | msg, url | | uid, uname, email, location |
| /user/logout | | | | | | | | | |
| GET |  |  | | /user/login | | |  | |  |
| /user/update | | | | | | | | | |
| GET | uid |  | | user/update.html | | | User | |  |
| POST |  | uid, pwd, pwd2, uname, email, location, tel | | common/alertMsg.html  /main | | |  | | uname, email, location |
| /user/mypage | | | | | | | | | |
| GET | uid |  | | user/mypage.html | | | User | |  |
| /user/detail | | | | | | | | | |
| GET | uid | |  | | user/userDetail.html | User | |  | |
| /building/map | | | | | | | | | |
| GET |  | |  | | search.html | buildingList | |  | |
| /building/search | | | | | | | | | |
| GET |  | | field, query | | search.html | buildingList, field, query | |  | |
| /building/detail | | | | | | | | | |
| GET |  | | bid, field, query | | detail.html | buildingList, building, lat, lon, areaList, commentList, replyList, favoriteCount | | favoriteCheck | |
| /building/comment | | | | | | | | | |
| POST |  | | bid, uid, content | | /building/detail |  | |  | |
| /building/comment | | | | | | | | | |
| POST |  | | replycid, replybid, replyuid, replycontent | | /building/detail |  | |  | |
| /building/favorite | | | | | | | | | |
| GET | bid | |  | | detail.html(ajax) | favoriteCheck | |  | |

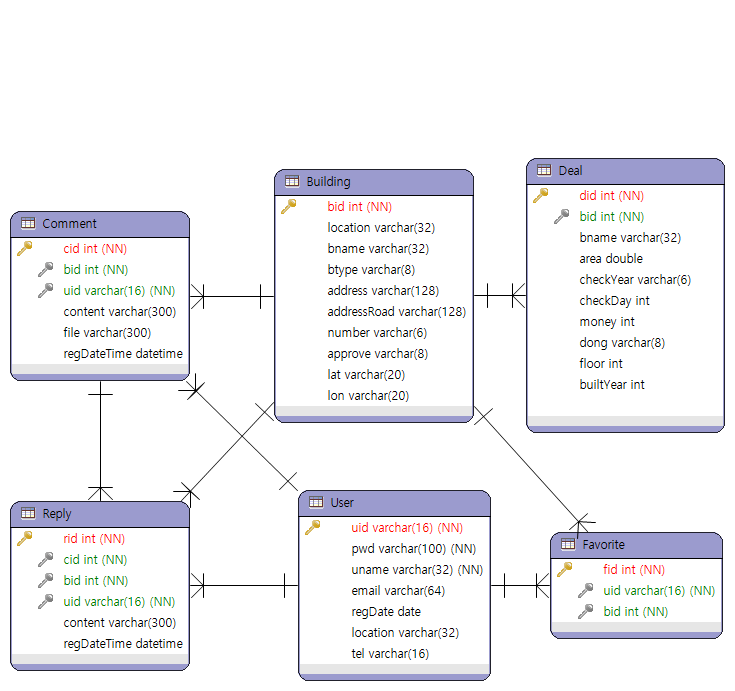
◎ 사용된 객체와 DAO

|  |  |
| --- | --- |
| User  사용자 | **private** String uid : ID  **private** String pwd : 비밀번호  **private** String uname : 이름  **private** String email : 이메일  **private** LocalDate regDate : 등록일자  **private** String location : 위치  **private** String tel : 전화번호 |
| Building  건물 | **private** **int** bid : 건물번호  **private** String location : 위치(시군구)  **private** String bname : 건물이름  **private** String btype : 건물유형(아파트 등)  **private** String address : 지번주소  **private** String addressroad : 도로명주소  **private** String number : 우편번호  **private** String approve : 승인날짜  **private** String lat : 위도  **private** String lon : 경도  **private** **boolean** hasDeal : 거래내역 존재 여부 - DB 미포함 변수 |
| Deal  거래정보 | **private** **int** did : 거래정보번호  **private** **int** bid : 건물번호(building.bid)  **private** String bname : 건물이름  **private** **double** area : 넓이(m2 단위)  **private** String checkYear : 거래년월  **private** String checkday : 거래일  **private** **int** money : 금액(만 원 단위)  **private** String dong : 동  **private** **int** floor : 층  **private** **int** builtYear : 건축일자 |
| Comment  댓글 | **private** **int** cid : 댓글번호  **private** **int** bid : 건물번호(building.bid)  **private** String uid : 사용자ID(User.uid)  **private** String content : 댓글내용  **private** String file : 파일  **private** LocalDateTime regDateTime : 작성일시  **private** String uname : 사용자이름 - DB 미포함 변수 |
| Reply  대댓글 | **private** **int** rid : 대댓글번호  **private** **int** cid : 댓글번호(Comment.cid)  **private** **int** bid : 건물번호(building.bid)  **private** String uid : 사용자 ID(User.uid)  **private** String content : 대댓글내용  **private** LocalDateTime regDateTime : 작성일시  **private** String uname : 사용자이름 - DB 미포함 변수 |
| Favorite  좋아요 | **private** **int** fid : 좋아요번호  **private** String uid : 사용자 ID(User.uid)  **private** **int** bid : 건물번호(building.bid)  **private** String bname : 건물이름 - DB 미포함 변수 |

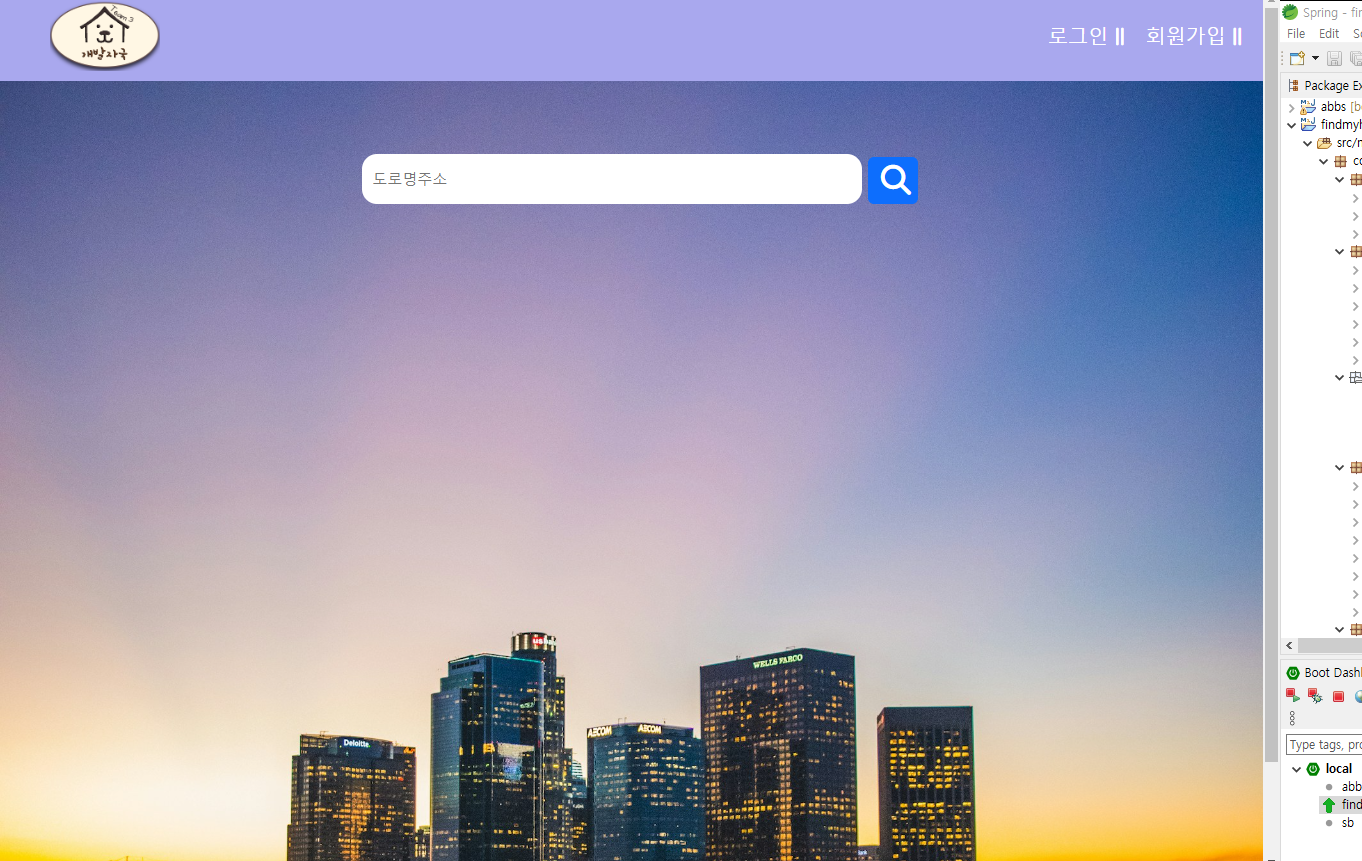
|  |  |
| --- | --- |
| UserDao | User getUser(String uid)  List<User> getUserList()  **void** insertUser(User user)  **void** updateUser(User user)  **void** deleteUser(String uid) |
| BuildingDao | Building getBuilding(**int** bid)  List<Building> getBuildingList(String field, String query)  **void** updateBuidingLatLon(Building buidling) - 건물의 위도 경도 정보만 갱신, 처음 건물 리스트를 받을 때 위도와 경도 값만 갱신하기 위해 사용  **void** deleteBuilding(**int** bid)  **float** getAvgLat(String field, String query) - 특정 범위의 건물의 위도 평균값  **float** getAvgLon(String field, String query) - 특정 범위의 건물의 경도 평균값  **float** getLatBid(**int** bid)  **float** getLonBid(**int** bid) |
| DealDao | Deal getDeal(**int** did)  List<Deal> getDealList(String field, String query)  List<Deal> getDealListBybid(**int** bid) - 특정 건물의 거래내역만 얻기  List<Integer> getAreaList(**int** bid) - 특정 건물의 거래내역 중 area 부분만 리턴, 특정 건물의 거래내역 중 넓이(소숫점 버림)가 같은 것끼리 묶기 위해 사용  List<Deal> getDealListBybidArea(**int** bid, **int** area) - 특정 건물에서 특정 넓이의 거래내역만 얻기  **int** getDealCount(**int** bid) - 특정 건물의 거래내역 수 |
| CommentDao | Comment getComment(**int** cid)  List<Comment> getCommentList(**int** bid, String query)  **void** insertComment(Comment comment)  **void** updateComment(Comment comment)  **void** deleteComment(**int** cid) |
| ReplyDao | Reply getReply(**int** rid)  List<Reply> getReplyList(**int** bid)  **void** insertReply(Reply reply);  **void** updateReply(Reply reply)  **void** deleteReply(**int** rid) |
| FavoriteDao | Favorite getFavorite(**int** fid)  Favorite getFavoriteByBid(**int** bid, String uid) - 특정 건물의 특정 유저의 좋아요 정보 찾기. 중복 좋아요 시 좋아요를 해제하는 기능을 만들기 위해 사용  List<Favorite> getFavoriteList(String uid)  **int** getFavoriteCount(**int** bid)  **void** insertFavorite(Favorite favorite)  **void** deleteFavorite(**int** fid) |

◎ DB 구성

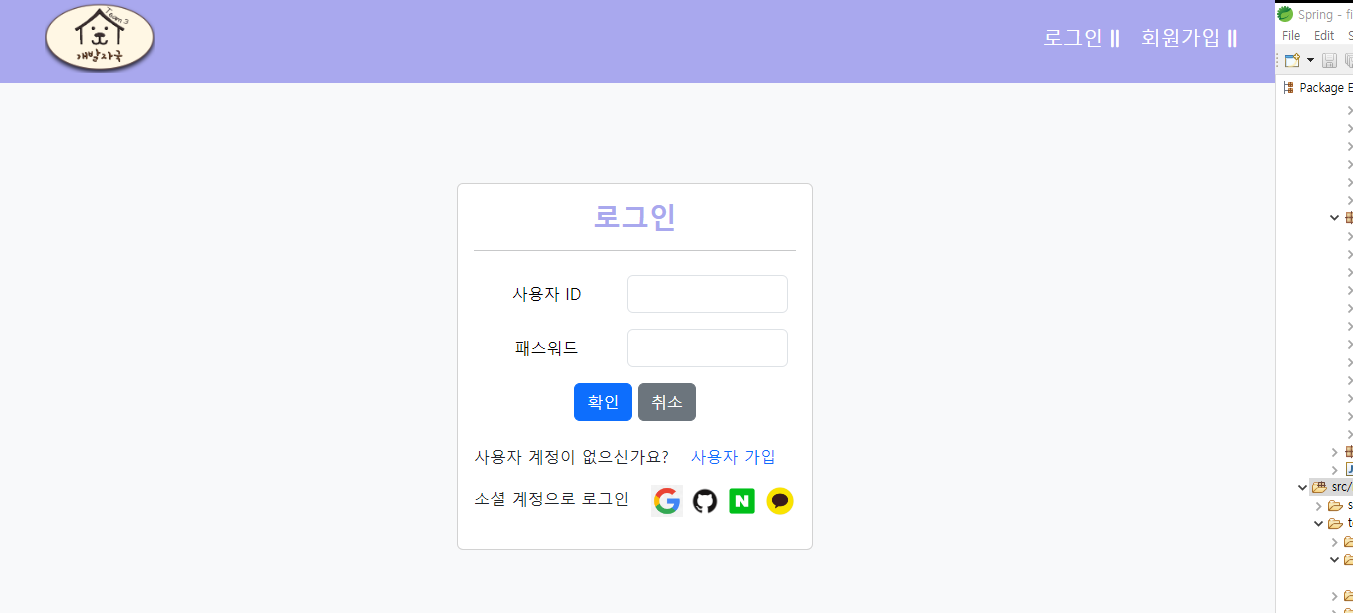
◎ 페이지 제작 결과



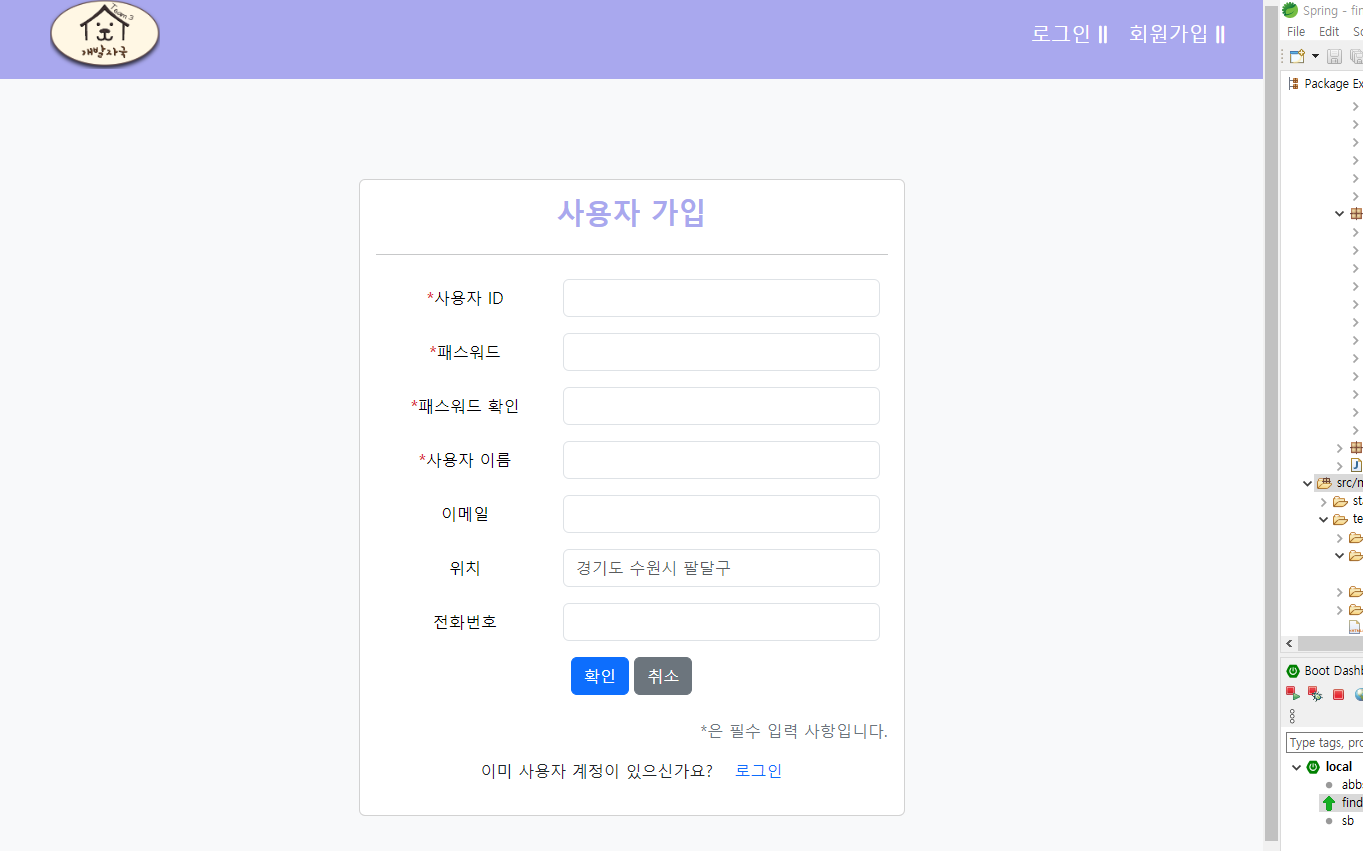
1. 메인페이지



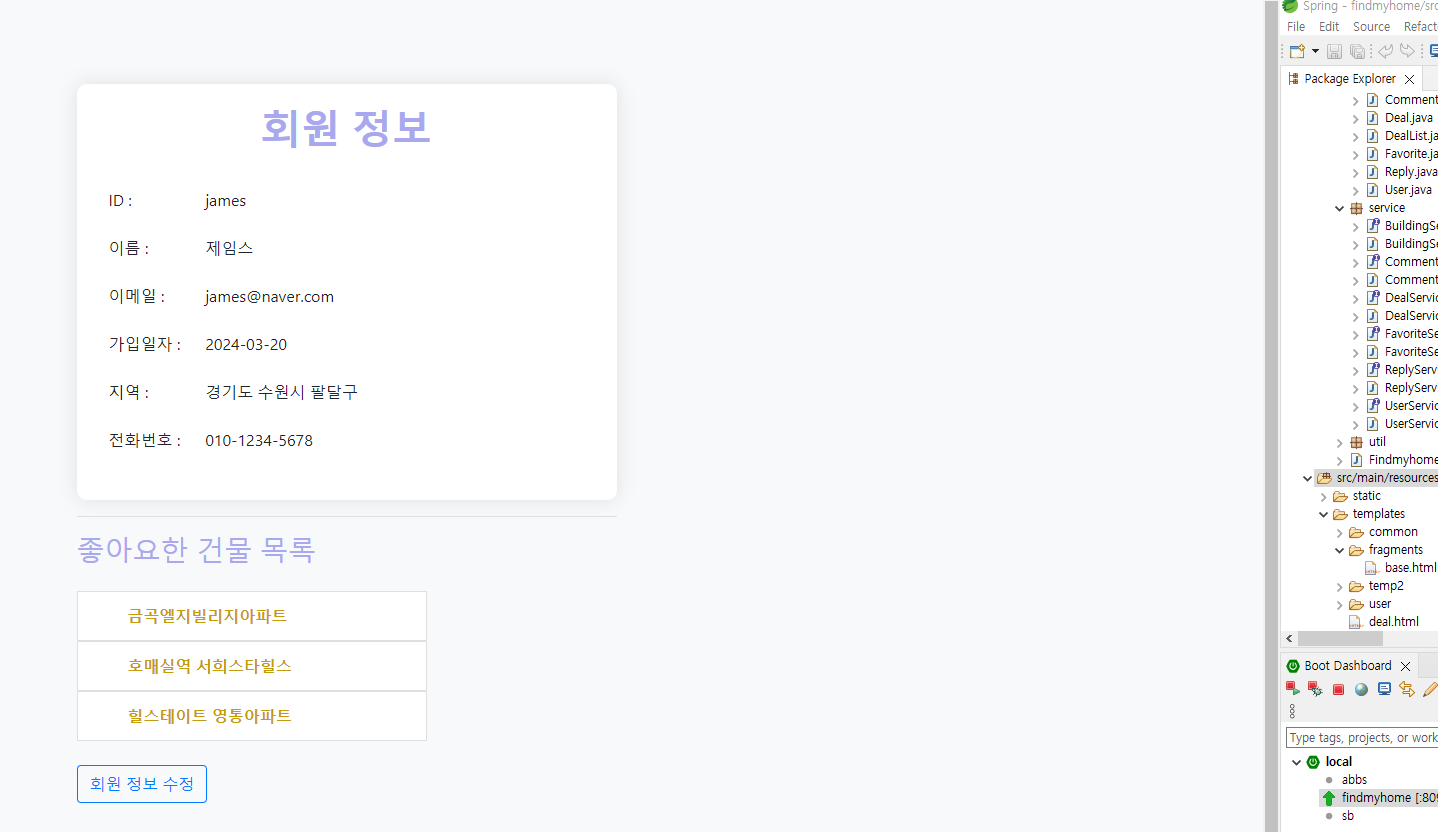
2. 로그인



3. 회원가입



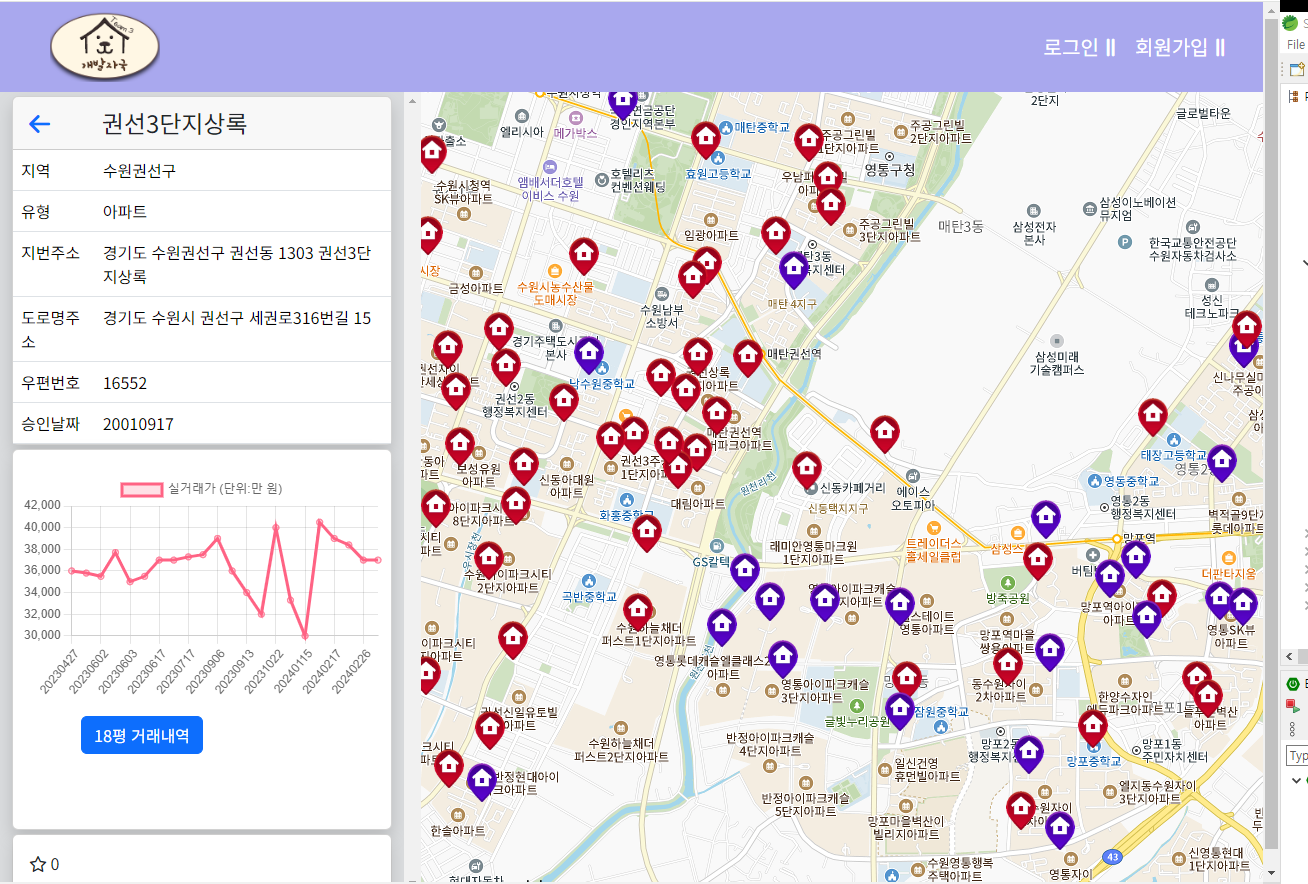
4. 마이페이지



5. 검색

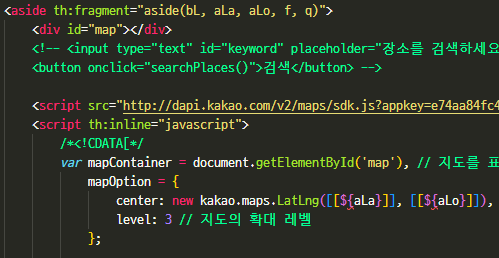


6. 건물 상세 페이지

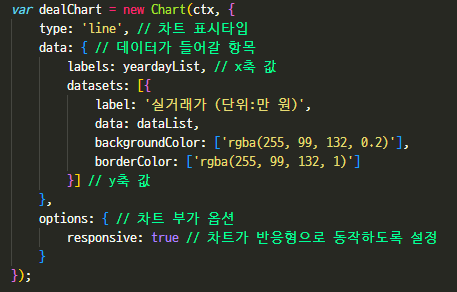
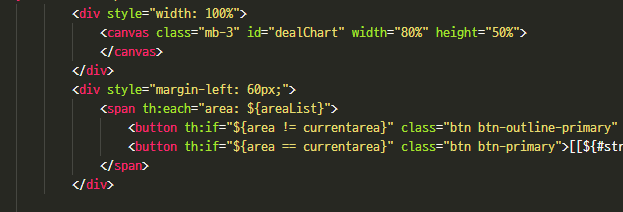


◎ 기능 구현

- 지도는 카카오 API를 사용해서 구현(base.html)



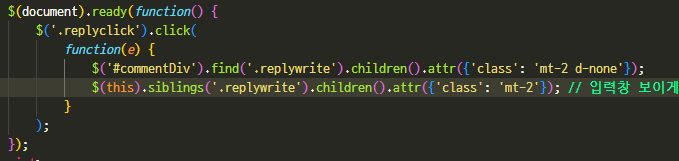
- 차트는 chart.js를 사용해서 구현(deal.html)



- 차트를 건물 상세정보에 지도랑 같이 표시하기 위해 iframe 사용 (detail.html)



- 댓글, 대댓글 기능은 thymeleaf, jQuery를 이용해서 구현(detail.html)



◎ 개선할 점

- iframe은 보안 취약점 등 현재는 잘 사용되지 않는 기능이라 다른 것으로 대체하는 것이 권장

- 댓글에 이미지 관련 변수가 있으나 실제로 사용되지는 않아 사용되게 하는 것이 필요

- DB에 들어간 정적인 데이터를 반영하는 것이 아닌 실제 데이터를 API로 반영하는 것이 좀 더 발전된 방향

◎ 참고한 사이트

아실

https://asil.kr/asil/index.jsp

국토교통부

https://www.data.go.kr/tcs/dss/selectApiDataDetailView.do?publicDataPk=15057511#/tab\_layer\_detail\_function

주소기반산업지원서비스

https://business.juso.go.kr/addrlink/openApi/apiExprn.do

호갱노노

https://hogangnono.com/