프로젝트 진행 시 어려웠던 점

2024-09-27

프로젝트를 진행하면서 회원 가입 기능을 구현하였고 그 다음으로 회원 가입 시 아이디가 중복되는지 확인하는 기능을 구현하고 있었습니다.

정확하게 fetch 를 작성하였고 민감한 정보인 아이디였기 때문에 POST 형식으로 전송하였습니다. 하지만 데이터가 전송이 되지 않았습니다.

400 즉 주소를 못찾는 문제가 발생하였고 alert으로 계속 확인했을 때는 .value 시에 값이 제대로 읽히는 것 까지 확인하였습니다. 하지만

버튼을 클릭 시 데이터는 읽지 못해 주소가 문제가 발생하는 것을 알게 되었습니다.

해결 방법은 자바스크립트가 블록 스코프인 것을 깜빡하고 있었고 이벤트리스너 안에서는 초기값을 가진다는 것을 생각해냈습니다.

그래서 주소로 읽을 값을 이벤트리스너 안에서 정의하였고 스프링 부트에서 데이터가 정확하게 전송되는 것을 확인할 수 있었습니다.

2024-09-28

fetch 문제를 해결하였으나 @PostMapping를 사용하는 함수에 레포지토리에서 id 값을 못찾는 문제가 발생하였습니다. fetch에서 중복을

찾기위해 유저 아이디를 전송했고 레포지토리는 String 타입으로 저장되었기 때문에 String 타입으로 받았으나 JSON 형식으로 전송되기

때문에 문제가 발생하였습니다. 그래서 찾아본 결과 JSON으로 오기 때문에 한번 파싱 작업을 해야한다는 것을 알게 되었습니다. 그래서

finduser(@RequestBody UserDTO users) 로 DTO 형식으로 변경하자 아이디 값을 찾을 수 있었습니다.

2024-10-01

스프링 시큐리티를 구현하는 과정에서 스프링부트를 공부하며 배운 내용으로 구현하였는데 스프링 시큐리티가 제대로 동작하지 않아서

이틀 간 수정하였고

문제가 발생하게된 이유는 sign.html 페이지를 만들고 form 태그에서 action="/login" 으로 했고 찾아본 결과로도 sign.html로 하고

action은 "/login"을 해도 문제가 없다 였습니다. 해결한 것은 action="/sign"으로 변경하고 아이디와 패스워드를 member\_id, member\_password

로 하였는데 username, password로 변경하였습니다. 그리고 데이터베이스 user 테이블을 member로 테이블명을 바꿨는데 나머지 외래키를

수정하지 않아서 발생한 문제였습니다.

2024-10-04



처음 게시판 페이지를 만들 때는 Board 엔티티 클래스에 Id는 board\_id 이런식으로 작성하였는데 이런 경우에는 JPA가 매핑을 하지 못해 오류가 발생하였습니다. 이를 해결하기 위해 각 테이블의 기본키는 id로 선언하고 @Column(name=”데이터베이스 테이블 기본키명”)으로 변경하여 해결하였습니다.

2024-10-07



게시판에 글 들 중에서 하나를 클릭하면 디테일 페이지로 넘어가는 기능을 만들었는데 이미지가 있을 경우에는 정상적으로 페이지가 화면에 출력되었지만 이미지가 없는 경우에는 화면이 정상 출력되지 않는 현상이 발생하였습니다. 이를 해결하고자 사진과 같이 작성하였는데 발상을 바꿔 board는 image 와 연관 관계인 것을 생각해 board 만 model로 전해줘도 되겠다는 생각을 하게 되었습니다. 그래서 디버그 기능으로 확인한 결과 board 객체만 호출해도 image도 함께 불러와져서 board만 전송해도 되는 것을 알게되어 디테일 페이지를 코드를 줄일 수 있게 되었습니다.



게시판 디테일뿐만 아니라 게시판 목록도 이미지 객체를 만들어 전송하는 것이 아닌 board 객체 하나를 전송하면 기존 기능을 수행하도록 변경하였습니다.



2024-10-13

10월 7일 이후로는 대부분의 기능을 구현하여 코드의 가독성과 중복되는 기능을 제거하면서 수정 작업을 진행하였습니다. 코드를 정리하면서 이전에 대충 구현했던 기능 중에서 논리적으로 이상한 부분을 오늘 수정하였습니다. 로그인된 상태에서만 게시판 글 삭제 수정 기능이 동작하게 만들었는데 삭제 버튼을 누르면 사용자가 누구든간에 삭제가 진행되는 이상함을 발견하였습니다. 이전에 기능 구현에 초점을 두어 기능들을 만들다 보니 삭제 기능을 간단하게 window.location.href=`/delete`; 로 작성하고 컨트롤러에서도 사용자 식별 없이 바로 삭제하는 과정을 거치기 때문에 사용자 식별 후 삭제 기능을 추가하였습니다.

2024-10-14

댓글 수정 기능을 구현하는 중에 고민한 생각으로 댓글을 후 수정 기능을 넣을지 아니면 다른 사이트처럼 삭제 기능만 구현하고 수정 기능을 없애버릴지 고민하였습니다.

화장품 추천 기능을 구현하기 위한 뼈대 작업으로 버튼을 클릭하면 이미지가 될 부분이 슬라이더 형식으로 동작하는 기능을 구현하였습니다.

2024-10-15

댓글 수정 기능에 대한 고민하여 내린 결론으로 댓글 수정 기능은 빼는게 낫다라고 생각하였습니다. 대부분의 사이트에서는 게시글에 대한 수정 기능은 제공하지만 댓글 수정 기능은 제공하지 않습니다. 그리고 댓글의 경우 사이트에 중요 기능이 아니고 로직 또한 복잡해져서 포트폴리오를 완성하고 나중에 구현하는게 낫다라고 판단하였습니다. 그래서 구현하고자하는 스마트 메이크업 사이트에서도 댓글에는 수정 기능을 잠시 빼고 구현하는 것으로 결정하였습니다.

게시판 삭제 기능에서 삭제 후에 이미지 링크는 데이터베이스에서 삭제되었는데 이미지는 스프링 부트 static 폴더에 계속 남는 현상을 수정하여 게시판을 삭제하면 이미지도 함께 삭제하게 변경하였습니다.

이미지를 삭제할 때 파일의 경로가 상대 경로일 경우 삭제가 되지 않아 절대 경로로 삭제하였습니다.