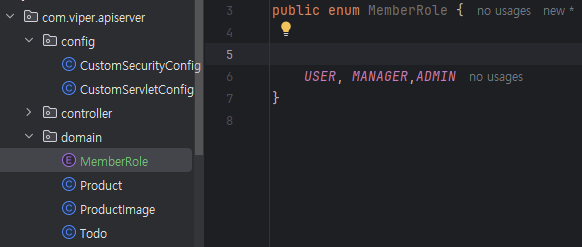
**#사용자 엔티티 처리**

user라는 이름은 DB에서 키워드로 사용될 확률이 매우 높다. 따라서 member로 생성한다.

**회원 권한**

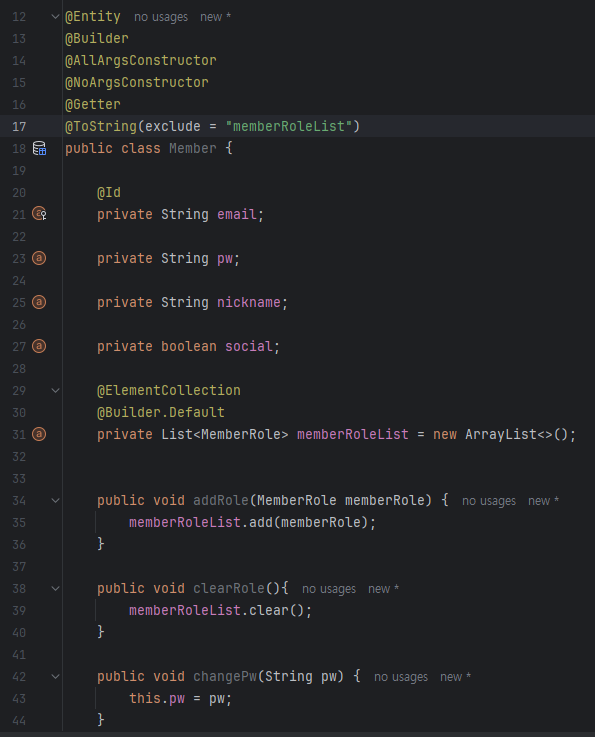


**회원 엔티티 클래스**

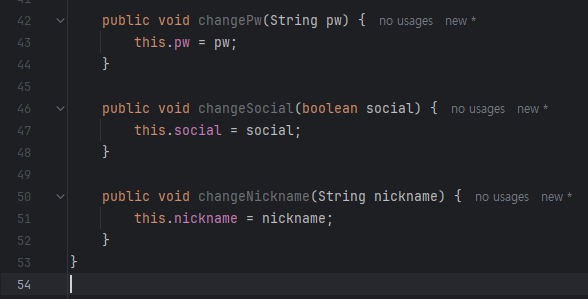
social 컬럼은 이후에 소셜로그인을 적용할 예정이기 때문에 소셜로그인한 사용자인지 구분하기 위한 컬럼이다.

**@ElementCollection을 사용하기 때문에 객체가 변경되면 영속성컨텍스트가 변경감지를 못한다.**

**따라서 Builder.Defualt를 사용해 빈 리스트로 초기화 해놓고 추가하는 방식으로 사용한다.**



수정 시 수정이 가능한 컬럼들에 한해 setter 메서드 추가 (요즘은 change라 많이씀.)



**JpaRepository 만들기**

이메일로 member테이블 조회해서 결과 가져오는 데 연관 테이블을 즉시 로딩함. (N+1문제 방지.)



**테스트 코드 작성 : Member 추가 , getWithRoles()**

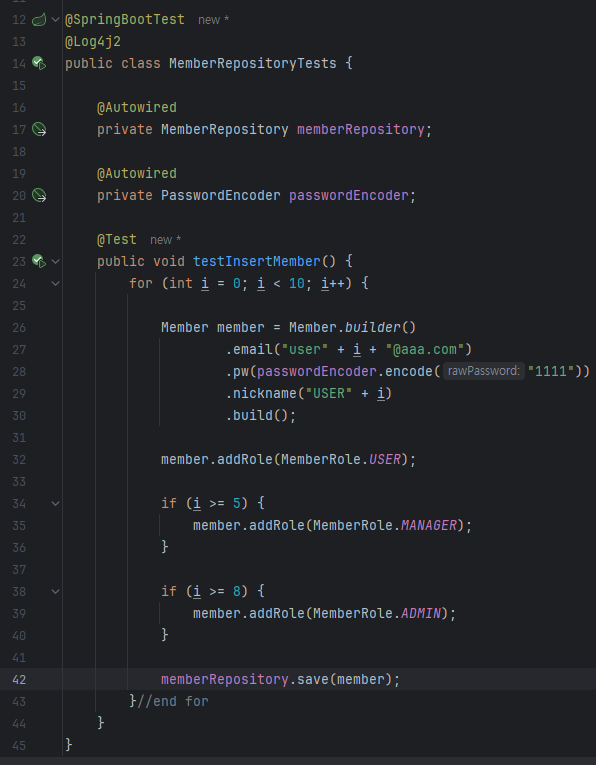
테스트를 위해서 member 튜플들을 생성 해줄텐데 password는 암호화해서 insert해야한다.

아니면 추후에 사용할 때 문제 된다.

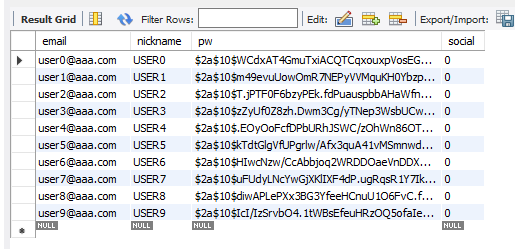
모든 유저들은 USER권한을 가진다.

5~10번은 MANAGER 권한도 가진다

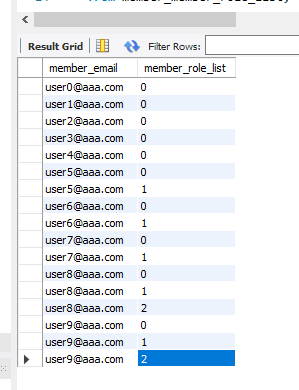
8~10번은 ADMIN 권한도 가진다



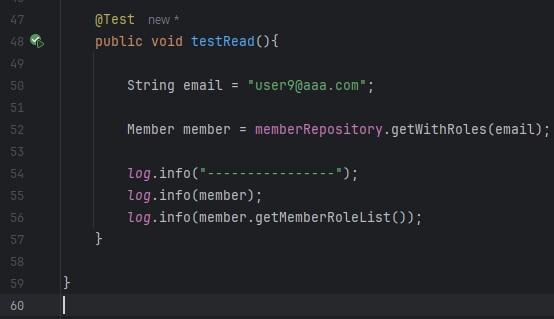
모든 password를 1111로 설정해줬지만 암호화되니 모두 값이 다른 모습.

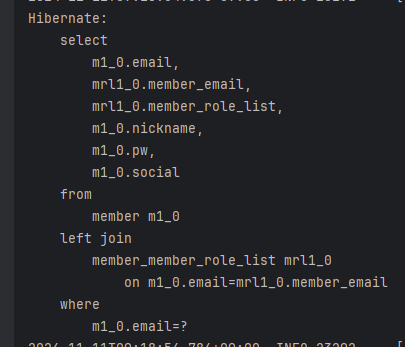


사용자별로 권한의 갯수가 다른 모습을 확인해볼 수 있다.



사용자 조회 테스트 : 권한 전부 조인하여 가져오는 지 확인해보기







이렇게 조인해서 잘 가져왔다는 걸 확인해볼 수 있다.