KB금융그룹 | 국민의 평생 금융파트너

2024년 상반기 K-디지털 트레이닝

SPRING JWT

[KB] IT's Your Life

KB 국민은행

Q1 - SPRING

- spring_JWT_문제.zip 프로젝트 파일을 다운로드(클릭) 받아 본인의 환경에 맞게 설정을 변경한 후, 프로젝트를 완성하시오.
- JUnit을 이용하여 다음과 같이 출력되도록 <코드 구현 부분>을 각 요구사항에 맞게 구현하시오.
- 실습을 위한 db 및 table, data는 다음 페이지의 스크립트를 실행하여 준비한다.

```
INFO : org.scoula.security.PasswordEncoderTest - password: $2a$10$9zdbCkzigFgfX3Kargxaqeh0j/tu7SyiECL0g7FP7bYcgRTEDabu6
INFO : org.scoula.security.PasswordEncoderTest - password: $2a$10$sWldK4sBbdjhMVP/6ac/TeOd8PEfcwhoKvgSJiuFw1VzOVYUR
INFO : org.scoula.security.PasswordEncoderTest - match :true
INFO : org.scoula.security.PasswordEncoderTest - match :true
```

- PasswordEncoderTest.java

```
join tbl_member_auth a on m.username = a.username where m.username = 'admin'

INFO : org.scoula.security.account.mapper.UserDetailsMapperTest - ----> MemberVO(username=admin, password=$2a$10$EsIMfxbJ6NuvwX7MDj4WqOYFzLU9U/lddCvn0nic5dFo3VfJYrXYC, email=admin@galapgos.
INFO : org.scoula.security.account.mapper.UserDetailsMapperTest - AuthVO(username=admin, auth=ROLE_ADMIN)
INFO : org.scoula.security.account.mapper.UserDetailsMapperTest - AuthVO(username=admin, auth=ROLE_MANAGER)
INFO : org.scoula.security.account.mapper.UserDetailsMapperTest - AuthVO(username=admin, auth=ROLE_MEMBER)
```

- UserDetailsMapperTest.java

- JwtProcessorTest.java

```
INFO : org.scoula.security.util.JwtProcessorTest - eyJhbGciOiJIUzM4NCJ9.eyJzdWIiOiJ1c2VyMCIsImlhdCI6MTcyNDY0NjM3Mywi
```

```
INFO : org.scoula.security.util.JwtProcessorTest - user0
```

발급 받은 토큰 복사 후
사용자id출력(발급 후 10분 전 실행)

```
INFO : org.scoula.security.util.JwtProcessorTest - --->> true
```

발급 받은 토큰 복사 후 유효한 토큰 여부
출력(발급 후 10분 전 실행)

```
JWT expired at 2024-08-26T04:36:13Z. Current time: 2024-08-26T04:37:17Z, a difference of 64430 milliseconds
io.jsonwebtoken.ExpiredJwtException Create breakpoint : JWT expired at 2024-08-26T04:36:13Z. Current time: 2024
```

발급 받은 토큰 복사 후 유효한 토큰 여부
출력(발급 후 10분 후 실행)

db table - member

- **테이블 생성**

```
-- 사용자 정보 테이블
drop table if exists tbl_member;

create table tbl_member
(
        username        varchar(50) primary key,        -- 사용자 id
        password        varchar(128) not null,                  -- 암호화된 비밀번호
        email                   varchar(50) not null,
        reg_date        datetime default now(),
        update_date datetime default now()
);

-- 사용자 권한 테이블
drop table if exists tbl_member_auth;

create table tbl_member_auth
(
        username        varchar(50) not null,           -- 사용자 id
        auth                    varchar(50) not null,           -- 권한 문자열 ROLE_ADMIN, ROLE_MANAGER, ROLE_MEMBER 등
        primary key(username, auth),                            -- 복합키
        constraint fk_authorities_users foreign key (username) references tbl_member(username)
);
```

db table - member

- 데이터 추가

```
-- 테스트 사용자 추가
insert into tbl_member(username, password, email)
values
        ('admin', '$2a$10$EsIMfxbJ6NuvwX7MDj4WqOYFzLU9U/lddCyn0nic5dFo3VfJYrXYC', 'admin@galapgos.org'),
        ('user0', '$2a$10$EsIMfxbJ6NuvwX7MDj4WqOYFzLU9U/lddCyn0nic5dFo3VfJYrXYC', 'user0@galapgos.org'),
        ('user1', '$2a$10$EsIMfxbJ6NuvwX7MDj4WqOYFzLU9U/lddCyn0nic5dFo3VfJYrXYC', 'user1@galapgos.org'),
        ('user2', '$2a$10$EsIMfxbJ6NuvwX7MDj4WqOYFzLU9U/lddCyn0nic5dFo3VfJYrXYC', 'user2@galapgos.org'),
        ('user3', '$2a$10$EsIMfxbJ6NuvwX7MDj4WqOYFzLU9U/lddCyn0nic5dFo3VfJYrXYC', 'user3@galapgos.org'),
        ('user4', '$2a$10$EsIMfxbJ6NuvwX7MDj4WqOYFzLU9U/lddCyn0nic5dFo3VfJYrXYC', 'user4@galapgos.org');


select * from tbl_member;
```

db table - member

- **권한 데이터 추가**

```
insert into tbl_member_auth(username, auth)
values
                ('admin','ROLE_ADMIN'),
                ('admin','ROLE_MANAGER'),
                ('admin','ROLE_MEMBER'),
                ('user0','ROLE_MANAGER'),
                ('user0','ROLE_MEMBER'),
                ('user1','ROLE_MEMBER'),
                ('user2','ROLE_MEMBER'),
                ('user3','ROLE_MEMBER'),
                ('user4','ROLE_MEMBER');

select * from tbl_member_auth order by auth;
```

  - ADMIN(최고 관리자)
    - ROLE_ADMIN, ROLE_MANAGER, ROLE_MEMBER
  - MANAGER(일반 관리자)
    - ROLE_MANAGER, ROLE_MEMBER
  - MEMBER(일반 회원)
    - ROLE_MEMBER

Q1 - SPRING

- **요구사항1 : PasswordEncoderTest.java**
  - spring Security를 활용한 비밀번호 인코딩 테스트를 수행하는 JUnit 테스트 클래스
- **클래스 설정:**
  - @ExtendWith(SpringExtension.class): Spring의 확장 기능을 통해 테스트 클래스에서 Spring의 기능을 사용할 수 있게 함.
  - @ContextConfiguration(classes = {RootConfig.class, SecurityConfig.class}): 테스트에 필요한 Spring 설정 파일(RootConfig 및 SecurityConfig)을 로드
  - @Log4j: Log4j를 사용하여 로그를 출력할 수 있게 함.
- **의존성 주입:**
  - @Autowired 어노테이션을 통해 PasswordEncoder 객체를 주입
- **테스트 메서드:**
  - testEncode 메서드는 PasswordEncoder를 사용하여 "1234"라는 문자열을 인코딩한 후, 그 결과를 로그에 출력
  - pwEncoder.encode(str) 메서드를 통해 두 번 인코딩된 결과를 각각 로그에 기록
  - pwEncoder.matches(str, enStr) 메서드를 통해 원본 문자열과 인코딩된 문자열이 일치하는지를 확인하여, 그 결과를 로그에 기록
- **결과 출력:**
  - 인코딩된 문자열은 매번 다르게 출력됨(BCrypt 등의 인코딩 방식 사용 시).
  - matches 메서드를 통해 원본 문자열과 인코딩된 문자열이 일치하는지 검증

- **요구사항2 : UserDetailsMapperTest.java**
  - spring 환경에서 UserDetailsMapper를 사용하여 특정 사용자의 정보를 조회하는 테스트를 수행하는 JUnit 테스트 클래스
- **클래스 설정:**
  - @ExtendWith(SpringExtension.class): Spring의 테스트 확장 기능을 사용하여 Spring 컨텍스트를 초기화
  - @ContextConfiguration(classes = {RootConfig.class, SecurityConfig.class}): 테스트에서 필요한 Spring 설정 파일(RootConfig, SecurityConfig)을 로드
  - @Log4j: Log4j를 사용하여 로그를 출력할 수 있도록 설정
- **의존성 주입:**
  - @Autowired 어노테이션을 통해 UserDetailsMapper 객체를 주입
- **테스트 메서드:**
  - testGet 메서드는 UserDetailsMapper를 사용하여 사용자 아이디가 "admin"인 사용자의 정보를 조회
  - 조회된 MemberVO 객체를 로그에 출력
  - 해당 사용자에게 부여된 권한(AuthVO 리스트)을 순회하며 각 권한을 로그에 출력

- **요구사항3 : JwtProcessorTest.java**
  - Spring 환경에서 JWT(Json Web Token) 처리 기능을 테스트하기 위한 JUnit 테스트 클래스
- **클래스 설정:**
  - @ExtendWith(SpringExtension.class): Spring의 테스트 확장 기능을 사용하여 Spring 컨텍스트를 초기화
  - @ContextConfiguration(classes = {RootConfig.class, SecurityConfig.class}): 테스트에 필요한 Spring 설정 파일(RootConfig, SecurityConfig)을 로드
  - @Log4j: Log4j를 사용하여 로그를 출력할 수 있도록 설정
- **의존성 주입:**
  - @Autowired 어노테이션을 통해 JwtProcessor 객체를 주입
- **테스트 메서드:**
  - generateToken 메서드: 특정 사용자 이름(예: "user0")을 사용하여 JWT 토큰을 생성하고, 해당 토큰이 null이 아닌지 확인. 생성된 토큰은 로그에 출력
  - getUsername 메서드: 주어진 JWT 토큰에서 사용자 이름을 추출하고, 추출된 사용자 이름이 null이 아닌지 확인.추출된 사용자 이름은 로그에 출력
  - validateToken 메서드: 주어진 JWT 토큰의 유효성을 검사하고, 유효한 경우 true를 반환하며, 그 결과를 로그에 출력. 5분 이상 경과한 토큰은 무효로 처리

수고하셨습니다!

KB 국민은행