7강 로그인 만들기

1. FORM 태그 정리

```
<form action="/login" method="post" enctype="application/x-www-form-urlencoded">
    <input type="text" name="username" placeholder="Enter username" /><br />
    <input type="password" name="password" placeholder="Enter password" /><br />
    <button>로그인</button>
</form>
```

> POST (BODY)
> key=value&key=value (x-www)
>
> GET (URL)
> ?key=value&key=value (QueryString)

Get 요청과 Post 요청 모두 x-www-form-urlencoded 타입으로 데이터가 전달되지만 Get요청을 하게되면 주소에 데이터가 실려가기 때문에 흔히 QueryString이라고 부른다.

기본적으로 form 태그는 데이터를 application/x-www-form-urlencoded로 보낸다. 이것이 디폴트이다.

기본적으로 form 태그는 method 정의를 안하면 디폴트가 GET이다.

application/x-www-form-urlencoded 이 친구의 key값은 input태그의 name값이다.

form태그의 button의 타입은 디폴트가 submit이다.

form태그의 validation 체크는 onSubmit 속성으로 한다.

https://penguingoon.tistory.com/188

2. Controller 만들기

UserController.java

```
    @Autowired
    private UserRepository userRepository;

    @Autowired
    private UserService userService;

    @Autowired
    private HttpSession session;

    // SELECT 요청이지만 로그인만 post로 한다. (예외임!!)
    @PostMapping("/login")
    public String login(LoginReqDto loginReqDto) {
        if (loginReqDto.getUsername() == null ||
loginReqDto.getUsername().isEmpty()) {
```

```
            throw new CustomException("username을 입력해주세요",
HttpStatus.BAD_REQUEST);
        }
        if (loginReqDto.getPassword() == null ||
loginReqDto.getPassword().isEmpty()) {
            throw new CustomException("password를 입력해주세요",
HttpStatus.BAD_REQUEST);
        }

        // 레파지토리 호출 (조회)
        // User principal = userRepository.findByUsernameAndPassword(loginReqDto);
        User principal = new User();
        principal.setId(1);
        principal.setUsername("ssar");
        if (principal != null) {
            throw new CustomException("아이디 혹은 비번이 틀렸습니다",
HttpStatus.BAD_REQUEST);
        }

        // ------ HTTP 최초 로직!!!!!!!!
        // HTTP로 만들어진 서버는 stateless 이다.
        // stateless 상태를 저장하지 않는 서버!!
        // 클라이언트가 request(get,post,put,delete) 요청을 하면!!
        // 서버는 거기에 해당하는 처리를 하고, response 응답을 한다.
        // 서버는 클라이언트의 정보를 저장하지 않는다.
        // 서버는 멀티쓰레드 프로세스이다. 서버는 다수의 클라이언트의 요청 동시에 받을 수 있다.
        // 부하를 방지하기 위해서 클라이언트 정보를 기억안함.

        // 클라이언트의 정보를 기억해야하는 stateful 한 서버가 필요해진다.
        // 그 저장공간에 session이다.
        // 그래서 세션에 저장해야한다.

        // 아파치/톰켓의 저장 영역
        // (request - 응답이 되는 순간 사라짐, session - 브라우저가 켜져 있는 동안)
        session.setAttribute("principal", principal);

        return "redirect:/";
    }
```

AccountController.java

```
 @GetMapping({ "/", "/account" })
public String main() {

    // throw new CustomException("인증되지 않았습니다", HttpStatus.UNAUTHORIZED);
    return "account/main";
}
```

> User객체를 login 메서드에서 강제로 만든 이유는 지금은 컨트롤러를 확인하는 것이기 때문에, User를 그냥 가짜로 만들어준 것

3. UserRepository 추상 메서드 만들기

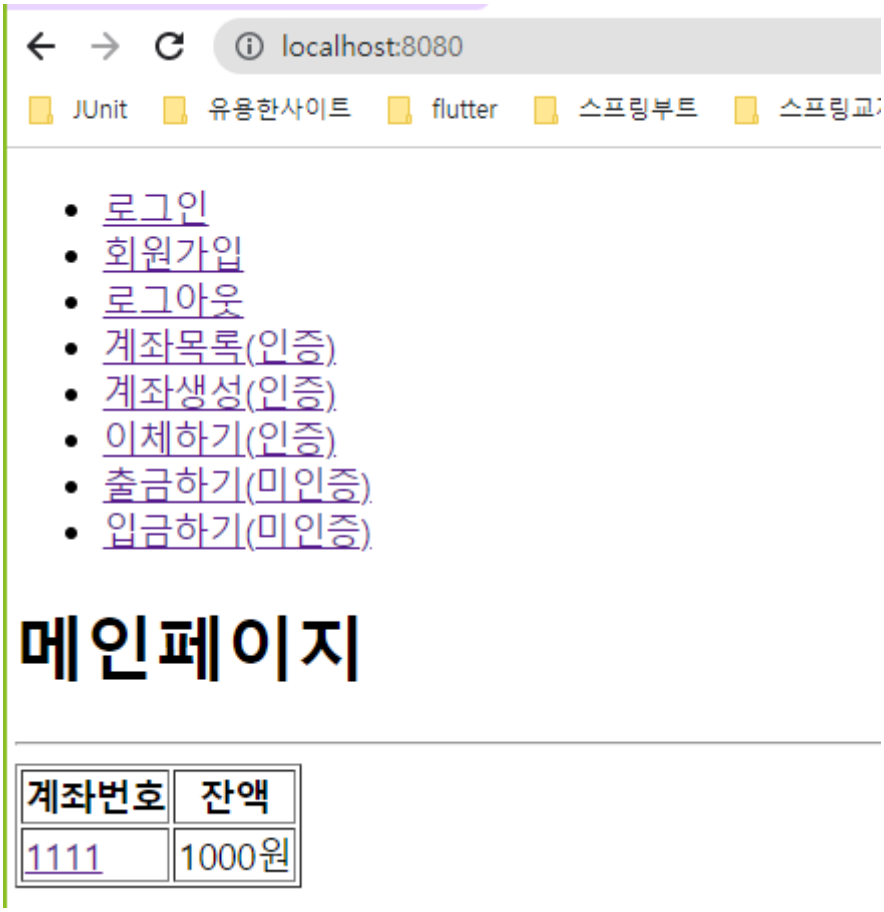
```
public User findByUsernameAndPassword(LoginReqDto loginReqDto);
```

4. Controller 코드 수정

```
// 레파지토리 호출 (조회)
User principal = userRepository.findByUsernameAndPassword(loginReqDto);

if (principal == null) {
    throw new CustomException("아이디 혹은 비번이 틀렸습니다", HttpStatus.BAD_REQUEST);
}
```

기존에 가짜 코드를 지우고 진짜 코드로 변경함... 이제 레포지토리 만들 것이기 때문에!!



5. 세션 체크해서 화면(네비게이션) 동기화

layout/header.jsp

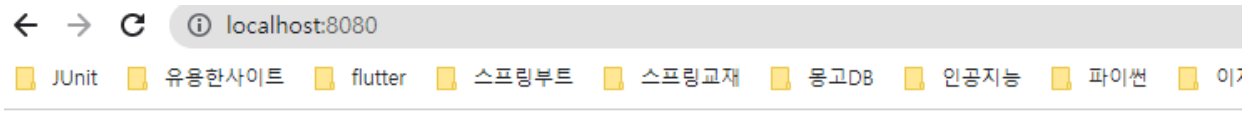
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
%>
    <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
        <!DOCTYPE html>
        <html lang="en">

        <head>
            <meta charset="UTF-8">
            <meta http-equiv="X-UA-Compatible" content="IE=edge">
            <meta name="viewport" content="width=device-width, initial-scale=1.0">
            <title>Bank 애플리케이션</title>
            <link rel="stylesheet" href="/css/style.css">
        </head>

        <body>
            <ul>
                <c:choose>
                    <c:when test="${principal != null}">
                        <li><a href="/logout">로그아웃</a></li>
                        <li><a href="/account">계좌목록(인증)</a></li>
                        <li><a href="/account/saveForm">계좌생성(인증)</a></li>
                        <li><a href="/account/transferForm">이체하기(인증)</a></li>
                        <li><a href="/account/withdrawForm">출금하기(미인증)</a></li>
                        <li><a href="/account/depositForm">입금하기(미인증)</a></li>
                    </c:when>

                    <c:otherwise>
                        <li><a href="/loginForm">로그인</a></li>
                        <li><a href="/joinForm">회원가입</a></li>
                        <li><a href="/account/withdrawForm">출금하기(미인증)</a></li>
                        <li><a href="/account/depositForm">입금하기(미인증)</a></li>
                    </c:otherwise>
                </c:choose>


            </ul>
```



- 로그아웃
- 계좌목록(인증)
- 계좌생성(인증)
- 이체하기(인증)
- 출금하기(미인증)
- 입금하기(미인증)

메인페이지

계좌번호	잔액
1111	1000원