

< 개발 완료 후 테스트 단계 >

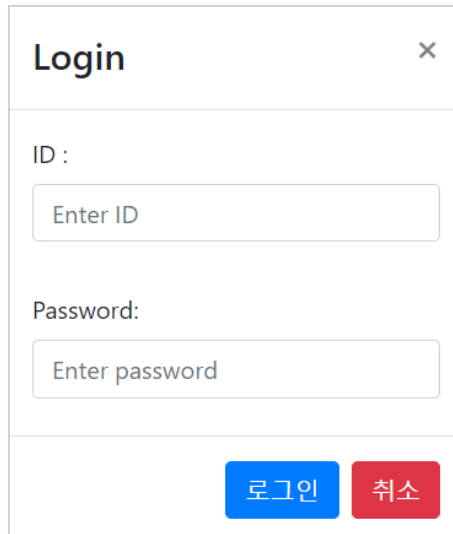
* 프로젝트 수행 단계에 따른 애플리케이션 테스트 명칭 및 순서

- 1) 단위 테스트 : 분리된 기능에 대한 검증으로 기능 단위마다 개발자가 테스트하는 것
- 2) 통합 테스트 : 컴포넌트 간의 상호 작용에 대한 검증으로 테스트 입력 값을 만들어 실행한 후
결과 확인하는 테스트
- 3) 시스템 테스트 : 전체 시스템 동작에 대한 검증으로 암호/복호화 데이터의 처리결과 확인 및
데이터 처리 시간을 통해 시스템 속도 측정, 정확도, 성공 및 실패율을
확인하는 테스트
- 4) 인수 테스트 : 사용자의 요구사항 처리에 대한 검증으로 사용자가 요구하는 기능들을
입력하고 정확하게 수행하는지 확인하는 테스트

* 기본적인 테스트 프로세스의 진행 순서

계획과 제어 -> 분석과 설계 -> 구현과 실행 -> 완료조건의 평가 및 리포팅 -> 테스트 마감활동

< 테스트1. 로그인 관련 테스트 >

A login form titled "Login" with a close button (X) in the top right corner. It contains two input fields: "ID :" with a placeholder "Enter ID" and "Password:" with a placeholder "Enter password". At the bottom, there are two buttons: a blue "로그인" (Login) button and a red "취소" (Cancel) button.

Login

ID :

Enter ID

Password:

Enter password

로그인 취소

- 로그인 기능이 정상적으로 작동되는지 단위테스트 하는 과정

- 1) WAS를 구동시킨다.
- 2) 로그인 페이지로 이동한다.
- 3) 로그인할 샘플 아이디와 비밀번호를 입력하고 로그인 버튼을 클릭한다.
- 4) 로그인 요청 처리용 컨트롤러가 정상적으로 구동되는지 확인한다.
- 5) 전달된 값들을 sql문에 적용한다.
- 6) sql문을 DB에 전달하여 실행하고 그 결과를 받는다.
- 7) 모델이 리턴한 결과를 가지고 성공/실패에 대한 뷰를 선택해 응답한다.
- 8) 처리 결과의 뷰가 클라이언트 브라우저에 출력되는지 확인한다.

- 테스트 수행의 목적 : 회원 로그인 기능이 올바르게 작동하는지 확인하기 위해 테스트 한다.

- 테스트 시나리오

테스트 케이스 작성하기	
테스트 케이스 ID	TEST-101
테스트 시나리오	로그인 화면에서 아이디와 비밀번호(user01, pass01)을 입력하여 로그인을 테스트 한다.
테스트 데이터	<pre>CREATE TABLE MEMBER (USER_NO NUMBER PRIMARY KEY, USER_ID VARCHAR2(30) NOT NULL, USER_PWD VARCHAR2(30) NOT NULL, USER_NAME VARCHAR2(15) NOT NULL, .. 이하 생략); INSERT INTO MEMBER VALUES(1, 'user01', 'pass01', '홍길동', .. 이하생략); INSERT INTO MEMBER VALUES(2, 'user02', 'pass02', '김말똥', .. 이하생략); INSERT INTO MEMBER VALUES(3, 'user03', 'pass03', '강개똥', .. 이하생략);</pre>
시작 조건	화면 개발 및 서버 로직 구현(모듈 연동)이 완료되었을 때
종료 조건	샘플 회원 (3명 정도)의 정보로 로그인 시 올바르게 로그인 성공했을 때
예상 결과	'로그인 성공했습니다' 와 같은 알림창 팝업됨
실제 결과	'로그인 성공했습니다' 와 같은 알림창 팝업됨 확인

- 테스트 수행 내용 (자바스크립트)

```
function loginTest(){  
    var userId = document.getElementById("userId").value;  
    var userPwd = document.getElementById("userPwd").value;  
    jQuery.ajax({  
        url:"member/loginTest.do",  
        data:"userId=" + userId + "&userPwd=" + userPwd,  
        success:function(data){  
            console.log(data);  
        }  
    });  
}
```

- 테스트 수행 내용 (서버 컨트롤러)

@WebServlet("member/loginTest.do")

```
Public class LoginController extends HttpServlet {  
    public void doGet(HttpServletRequest request, HttpServletResponse response)  
        throws IOException, ServletException {  
        String userId = request.getParameter("userId");  
        String userPwd = request.getParameter("userPwd");  
  
        Member m = new MemberService().selectOne(userId);  
  
        if(m != null) {  
            response.getWriter().append("true");  
        }  
    }  
}
```

- 로그인 요청시 수행할 SQL문 관련 mapper 파일 정보

[만일, 회원 로그인 기능을 통해 조회해 올 필수 정보가 다음과 같다면]

아이디(MEMBER.USER_ID), 비밀번호(MEMBER.USER_PWD), 이름(MEMBER.USER_NAME),
등급명(GRADE.GRADE_NAME), 권한명(AUTH.AUTH_NAME)

[현재 테스트 시 실행한 mapper 내용]

```
<select id="login" resultMap="memberResultSet">  
    SELECT * FROM MEMBER  
    JOIN GRADE ON (MEMBER.GRADE_ID = GRADE.GRADE_NO)  
    JOIN AUTH ON (MEMBER.AUTH_ID = AUTH.AUTH_NO)  
    WHERE USER_ID = #{userId}  
</select>
```

단, 위와 같이 실행하면 성능 저하 발생 그 이유는 다음 페이지에..

- 위와 같은 sql문 실행 시 지나치게 많은 정보를 조회해와 데이터 성능 저하를 야기시킨다.

원인 : 현재 로그인 서비스에서 요구하는 정보 외 부가적인 정보들도 모두 조회해오기 때문

조치내용 : 아래의 SQL문과 같이 필요한 정보만 조회해오도록 수정

```
<select id="login" resultMap="memberResultSet">
```

```
    SELECT MEMBER.USER_ID, MEMBER.USER_PWD, MEMBER.USER_NAME,
```

```
           GRADE.GRADE_NAME, AUTH.AUTH_NAME
```

```
    FROM MEMBER
```

```
    JOIN GRADE ON (MEMBER.GRADE_ID = GRADE.GRADE_NO)
```

```
    JOIN AUTH ON (MEMBER.AUTH_ID = AUTH.AUTH_NO)
```

```
    WHERE USER_ID = #{userId}
```

```
</select>
```

테스트2. 게시글 검색 관련 테스트

Previous1234Next

작성자

검색

- 테스트 수행의 목적 : 게시글 조회 기능이 올바르게 작동하는지 확인하기 위해 테스트 한다.
- 테스트 시나리오

테스트 케이스 작성하기	
테스트 케이스 ID	TEST-102
테스트 시나리오	데이터베이스에 저장된 샘플 데이터를 조회하여 게시글 목록을 출력되게 한다.
테스트 데이터	<div>CREATE TABLE BOARD (BOARD_NO NUMBER PRIMARY KEY, BOARD_TITLE VARCHAR2(300) NOT NULL, BOARD_WRITER VARCHAR2(30) NOT NULL, BOARD_CONTENT VARCHAR2(15) NOT NULL); INSERT INTO BOARD VALUES(1, '게시글 제목1', '홍길동', '게시글 내용1'); INSERT INTO BOARD VALUES(2, '게시글 제목2', '김말똥', '게시글 내용2'); INSERT INTO BOARD VALUES(3, '게시글 제목3', '강개똥', '게시글 내용3');</div>
시작 조건	화면 개발 및 서버 로직 구현(모듈 연동)이 완료되었을 때
종료 조건	샘플 데이터가 담겨있는 정보가 올바르게 조회되었을 때
예상 결과	게시글 목록에 샘플 데이터가 조회된다.
실제 결과	게시글 목록에 샘플 데이터가 조회됨을 확인