

#K-Collections
ARTS ONLY FOR YOU

Spring Framework Project

이다영, 문태식, 김영래,
이한얼, 이상민

목차

1. ABOUT PROJECT
2. 개발환경
3. DB ERD
4. Page Layout
5. Spring Framework 패키지 구성
6. Security 구성도 - 가시성 / 접근성 / CSRF
7. CRUD & LIST
8. 영상시연
9. 핵심코드

ABOUT PROJECT

온라인 쇼핑몰

미술품 시장의
급성장

재고의 유일성

#K-Collections
ARTS ONLY FOR YOU



[국어사전]

컬렉션 (collection) ↗

1 미술품이나 우표, 화폐, 책, 골동품 따위를 모으는 일. 또는 그 모아진 물건들.

2 관련된 물건들이나 상품들의 집합.

☞ 미술품 판매 웹사이트

이미지 협찬: 한국미술협회 당진지부 회원 일동

개발환경



- 운영체제: Windows 10
- IDE : Eclipse 4.25
- JDK : Amazon Correto 11
- 웹서버 : Apache Tomcat 9.0
- 프론트엔드: Bootstrap 4 , jQuery@3.6.1, CSS
- Database : MySQL Workbench 8.0
- WAS application : Spring Framework 5
- 프로젝트 관리도구: Maven

개발환경 - Maven <property>

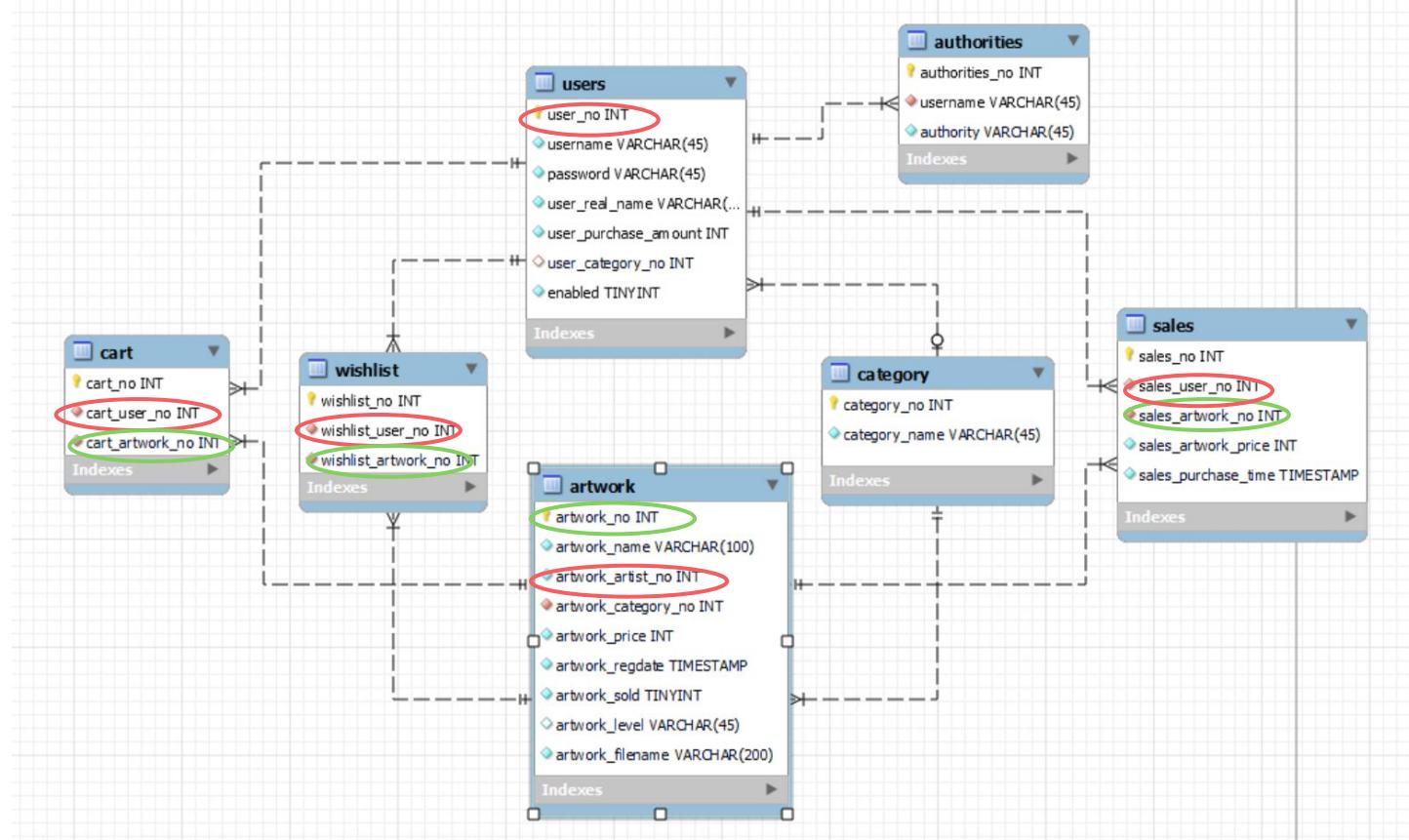


```
<properties>
    <springframework.version>5.0.2.RELEASE</springframework.version>
    <springsecurity.version>5.0.0.RELEASE</springsecurity.version>
    <hibernate.version>5.4.4.Final</hibernate.version>
    <mysql.connector.version>8.0.17</mysql.connector.version>
    <c3po.version>0.9.5.2</c3po.version>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
</properties>
```

- mysql.connector : MySQL 연결
- c3p0 : 커넥션 풀링 라이브러리. 어떤 jdbc 데이터베이스와도 연동, 멀티쓰레드 환경에서 유용
- 쿼리문: Hibernate 문법, 일부 Native query로 직접 sql문 작성

DB ERD

#K-Collections
ARTS ONLY FOR YOU



Page Layout

1. 페이지 전반 디자인: Bootstrap 4
2. 추가 CSS: 각 페이지마다 <style> 양식 적용
3. 페이지 공통 적용: header.jsp & footer.jsp
 - <jsp:include page = "header.jsp"></jsp:include>
4. 홈 메인 페이지
 - 짐 개수 기준 'Best Wish 템' 코너
 - 등록일 기준 '새로 들어온 작품' 코너
5. ARTWORKS (카테고리별), ARTISTS, VIP ZONE
6. 권한에 따른 MYPAGE, 짐 목록, 장바구니, 로그인 버튼

#K-Collections

ARTS ONLY FOR YOU



MENU

ARTWORKS ARTISTS VIP ZONE

MYPAGE



Login



Spring Framework 구성



1) Config

- DemoAppConfig
- DemoSecurityConfig
- MySpringMvcDispatcherServletInitializer
- SecurityWebApplicationInitializer

2) 컨트롤러

- DemoController
- LoginController

3) Entity 7개

4) Service

5) DAO



Spring Framework Configuration

```
public class SecurityWebApplicationInitializer extends AbstractSecurityWebApplicationInitializer {
```

➤ 우리 프로젝트 모든 URL에 자동으로 스프링시큐리티 필터

```
public class MySpringMvcDispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {
```

```
    @Override  
    protected Class<?>[] getRootConfigClasses() {  
        return null;  
    }
```

```
    @Override  
    protected Class<?>[] getServletConfigClasses() {  
        return new Class[] { DemoAppConfig.class };  
    }
```

➤ 최초로 불러오는 설정 클래스 Initialize

```
    @Override  
    protected String[] getServletMappings() {  
        return new String[] { "/" };  
    }
```

➤ 서블릿 맵핑은 /붙여서

Spring Framework Configuration

```
@Configuration
@EnableWebMvc
@EnableTransactionManagement
@ComponentScan(basePackages="com.code.springdemo")
@PropertySource({ "classpath:persistence-mysql.properties" })
public class DemoAppConfig implements WebMvcConfigurer {
    > .properties 파일에 있는 JDBC 관련 환경변수들을 Environment 객체에 담아서 가져옴
    @Autowired
    private Environment env;
    ...
    @Bean
    public ViewResolver viewResolver() {
        > jsp 로 끝나는 뷰단 페이지로 매팅
```

Security 구성

Spring Security

- 1) 접근성 Access Restriction → securityConfig.java
- 2) 가시성 Visibility Restriction → view 단

- View단 - header.jsp



<security:authorize access = "hasRole('000')">

A screenshot of a website's header navigation bar. It includes a 'MENU' button, 'ARTWORKS' link, 'ARTISTS' link, a 'VIP ZONE' link which is highlighted with a green oval, and a '접근성' (Access) link. To the right are 'MYPAGE', a heart icon, a shopping cart icon, and a 'Login' button. Below the bar are two small thumbnail images of artworks.

#K-Collections

ARTS ONLY FOR YOU



MENU

ARTWORKS ARTISTS VIP ZONE

MYPAGE



Logout



Security 구성도 - Visibility



- ▶ 권한에 따라 보여지는 내용이 다르거나, 같은 링크로 보여도 권한에 따라 매핑되는 페이지가 다름

MENU ARTWORKS ARTISTS VIP ZONE MYPAGE Login

```
<ul class="navbar-nav" id="nav5" >
<li class="nav-item">
<security:authorize access = "isAnonymous()">
  <a class="nav-link" onclick="window.location.href='${pageContext.request.contextPath}/showMyLoginPage'; return false;">MYPAGE</a>
</security:authorize>
</li>
<li class="nav-item">
<security:authorize access = "hasAnyRole('USER', 'VIP', 'VVIP', 'ARTIST')">
  <a class="nav-link" onclick="window.location.href='${pageContext.request.contextPath}/userPage'; return false;">MYPAGE</a>
</security:authorize>
</li>
```

MENU ARTWORKS ARTISTS VIP ZONE MYPAGE Logout

- ▶ 마이페이지: 비로그인 상태 시 로그인 창으로, 로그인되었다면 유저 마이페이지로

Security 구성도 - Visibility

#K-Collections
ARTS ONLY FOR YOU

MENU

ARTWORKS

ARTISTS

VIP ZONE

MYPAGE

작가페이지



Logout

MENU

ARTWORKS

ARTISTS

VIP ZONE

관리자페이지

Logout

```
<li class="nav-item">
<security:authorize access = "hasRole('ARTIST')">
    <a class="nav-link" onclick="window.location.href='${pageContext.request.contextPath}/artistPage'; return false;">작가페이지</a>
</security:authorize>
</li>
<li class="nav-item">
<security:authorize access = "hasRole('ADMIN')">
    <a class="nav-link" onclick="window.location.href='${pageContext.request.contextPath}/adminPage'; return false;">관리자페이지</a>
</security:authorize>
</li>
```

▶ 로그인 후, 작가 권한을 가진 자에게는 작가 마이페이지 / 관리자 권한을 가진 자에게는 관리자 페이지가 보여지도록

Security 구성도 - Visibility

#K-Collections
ARTS ONLY FOR YOU

```
<li class="nav-item">
<security:authorize access = "isAnonymous()">
    <a class="nav-link" onclick="window.location.href='${pageContext.request.contextPath}/showMyLoginPage'; return false;">
        <i class='fas fa-heart' style='font-size:24px'></i></a>
    </security:authorize>
</li>
<li class="nav-item">
<security:authorize access = "hasAnyRole('USER', 'VIP', 'VVIP', 'ARTIST')">
    <a class="nav-link" href="${pageContext.request.contextPath}/showWish"><i class='fas fa-heart' style='font-size:24px'></i></a>
    </security:authorize>
</li>
<li class="nav-item">
<security:authorize access = "isAnonymous()">
    <a class="nav-link" onclick="window.location.href='${pageContext.request.contextPath}/showMyLoginPage'; return false;">
        <i class='fas fa-shopping-cart' style='font-size:24px'></i></a>
    </security:authorize>
</li>
<li class="nav-item">
<security:authorize access = "hasAnyRole('USER', 'VIP', 'VVIP', 'ARTIST')">
    <a class="nav-link" href="${pageContext.request.contextPath}/showcart"><i class='fas fa-shopping-cart' style='font-size:24px'></i>
    </security:authorize>
</li>
```

찜

카트



MENU

ARTWORKS

ARTISTS

VIP ZONE

MYPAGE



Logout

▶ 찜, 장바구니: 비로그인 상태 시 로그인 창으로, 로그인 후 해당 유저에 따라 저장된 찜목록, 장바구니 목록으로

Security 구성도 - Visibility



```
-->
<li class="nav-item">
<security:authorize access = "isAnonymous()">
    <a class="nav-link" onclick="window.location.href='${pageContext.request.contextPath}/showMyLoginPage'; return false;">
        <button type="submit" class="btn btn-outline-dark">Login</button></a>
    </security:authorize>
</li>
<li class="nav-item ml-auto">
<security:authorize access = "isAuthenticated()">
    <form:form action="${pageContext.request.contextPath}/logout" method="POST">
        <button type="submit" class="btn btn-outline-dark">Logout</button>
    </form:form>
</security:authorize>
</li>
```

| | | | | | | | |
|-------------|----------|---------|----------|---------------|--|--|---------------|
| MENU | ARTWORKS | ARTISTS | VIP ZONE | MYPAGE | | | Login |
| MENU | ARTWORKS | ARTISTS | VIP ZONE | MYPAGE | | | Logout |

▶ 로그인, 로그아웃 버튼: 비로그인 상태 시 로그인창으로, 로그인 상태인 경우에는 로그아웃 버튼이 보여지도록

#K-Collections

ARTS ONLY FOR YOU

MENU

ARTWORKS ARTISTS VIP ZONE

MYPAGE



Login



- ▶ **Visibility**를 제한했을 경우, 허용되는 권한을 가진 자에게만 보여지므로 다른 권한을 가진 자가 접근을 할 수 없음
- ▶ 접근제한이 되어있는데 접근을 한다면, 403 에러 → 이러한 접근제한은 **securityConfig** 파일에서 설정



고객님의 등급은 "일반"입니다.
해당 페이지에 접근이 불가합니다.

[홈으로](#)

Security 구성도 - Access Restriction

#K-Collections
ARTS ONLY FOR YOU

```
@Configuration // 설정파일을 만들기 위한 애노테이션
```

```
@EnableWebSecurity
```

```
public class DemoSecurityConfig extends WebSecurityConfigurerAdapter {
```

▶ 스프링시큐리티 Configuration Class 만들기: `WebSecurityConfigurerAdapter`를 상속하여 클래스를 생성하고 `@EnableWebSecurity` 애노테이션을 추가

```
@Autowired // 의존성 주입 DI
```

```
private DataSource securityDataSource;
```

▶ 스프링시큐리티의 유연함: 어떤 데이터 저장소에 대해서도 사용자 인증 가능

```
@Override // 사용자 저장소 설정
```

수업시간에는 `.inMemoryAuthentication` 메서드, 여기서는 관계형 데이터베이스에 직접 접근

```
protected void configure(AuthenticationManagerBuilder auth) throws Exception{  
    auth.jdbcAuthentication().dataSource(securityDataSource);
```

} ▶ 위 메소드로 JDBC 지원 사용자 저장소에서 인증, Data source만 설정해주면 MySQL 내로 접근

Security 구성도 - Access Restriction

```
@Override  
protected void configure(HttpSecurity http) throws Exception { ➤ 스프링 시큐리티의 각종 설정: HttpSecurity로  
    http  
        .authorizeRequests()  
            .antMatchers("/css/**").permitAll()  
  
            .antMatchers("/userPage/**").hasAnyRole("USER", "VIP", "VVIP", "ARTIST")  
            .antMatchers("/artistPage/**").hasRole("ARTIST")  
            .antMatchers("/adminPage/**").hasRole("ADMIN")  
  
            .antMatchers("/vipPage/**").hasAnyRole("VIP", "VVIP", "ADMIN")  
  
        .anyRequest().permitAll()  
  
        .and()  
        .formLogin()  
            .loginPage("/showMyLoginPage") //로그인창 Url 설정  
            .loginProcessingUrl("/authenticateTheUser")  
            .permitAll()  
        .and()  
        .logout()  
            .logoutSuccessUrl("/logout") //로그아웃 성공 시 페이지  
            .permitAll()  
  
        .and()  
        .exceptionHandling().accessDeniedPage("/access-denied");  
}
```

1) URL마다 접근권한 설정

2) 인증 실패 시 이동페이지 연결
(권한 확인을 위해 자동으로
로그인창으로 유도)

3) 토큰을 통해 csrf로부터 보호
(default로 설정되어 있음)

CSRF (사이트 간 위조 요청)



이상옥 작가님의
마이
페이지입니다.

작가님의 판매 중인 작품 목록을 관리하고, 판매된 작품 목록을 볼 수 있는 페이지입니다.

새로운 작품을 업로드해보세요!

작품 업로드

판매하고자 하는 새로운 작품의 이미지를 업로드하고 상세 정보를 설정합니다.

```
DevTools is now available in Korean! Always match Chrome's language Switch DevTools to Korean Don't show again
Elements Console Sources Network Performance Memory Application Security Light
<li class="nav-item"> </li>
<li class="nav-item"> </li>
▶<li class="nav-item">..</li>
<li class="nav-item"> </li>
▶<li class="nav-item">..</li>
<li class="nav-item"> </li>
▼<li class="nav-item ml-auto">
  ▶<form id="command" action="/k-collections/logout" method="POST">
    <button type="submit" class="btn btn-outline-dark">Logout</button>
  ▶<div>
    <input type="hidden" name="_csrf" value="0c7b40e4-9645-46b9-a5c8-ebf8791b7530"> == $0
  </div>
  </form>
</li>
</ul>
</div>
</nav>
</header>
▼<div class="container mt-3">
  ▶<div class="jumbotron" style="background-image: url('resources/image/artistmypagefront.png'); for
    <h1>이상옥 작가님의 마이페이지입니다.</h1>
    <br>
    <p>작가님의 판매 중인 작품 목록을 관리하고, 판매된 작품 목록을 볼 수 있는 페이지입니다. </p>
    <p>새로운 작품을 업로드해보세요!</p>
  </div>
```

▶ Spring security <form:form> 태그를 통해 csrf 토큰값을 넘겨줌

- ▶ 사용자가 일반유저인지, 악용된 공격인지 구분할 수 없을 때를 대비해 csrf 토큰이 포함되어야 요청을 받아들이게 함으로써 위조요청을 방지

CRUD



회원 관련 CRUD - 회원가입, 회원정보 수정(비밀번호 변경, 회원탈퇴)

작가 마이페이지 CRUD - 그림 업로드, 상세정보 수정, 작품삭제, 조건별 화면구성

관리자 페이지 CRUD - 회원, 작품, 판매내역 Table 전반을 관리하는 CRUD

찜, 장바구니 CRUD - 원하는 작품 추가 및 삭제, 구매하기로 연결

LIST



작품 List - 카테고리별로 보여주기 (서양화, 동양화 등)

작가별 List - 새로 생성되는 작가 수 만큼의 탭 생성, 각각의 작품 보여줌

VIP List - vip/vvip 등급에만 보여지며 권한이 없는 자는 access-denied 페이지

영상시연

#K-Collections

ARTS ONLY FOR YOU

MENU

ARTWORKS ARTISTS VIP ZONE

MYPAGE



Login



#K-Collections

ARTS ONLY FOR YOU



MENU

ARTWORKS ARTISTS VIP ZONE

MYPAGE



Login



#K-Collections

ARTS ONLY FOR YOU

MENU

ARTWORKS ARTISTS VIP ZONE



MYPAGE



Logout



#K-Collections

ARTS ONLY FOR YOU

MENU

ARTWORKS ARTISTS VIP ZONE

관리자페이지

Logout



핵심코드 - 회원삭제

➤ View단 : admin-mypage.jsp

```
<c:url var="DeleteUrl" value="/adminPage/DeleteUser">
    <c:param name="userNo" value="${User.userNo}"></c:param>
</c:url>
```

➤ DemoController

```
//어드민 마이페이지 - 유저 관리 - 유저를 삭제하는 메서드
@GetMapping("/adminPage/DeleteUser")
public String deleteUser(@RequestParam("userNo") int theId) {
    artshopservice.deleteUser(theId);
    return "redirect:/adminPage/UserList";
}
```

회원삭제

DAOLimpl

```
//유저no를 가져와서 유저에 관한 모든 테이블 정보 삭제
@Override
public void deleteUser(int theUserId) {
    Session currentSession = sessionFactory.getCurrentSession();

    String userName = getUsers(theUserId).getUserName();
                                                판매내역테이블에서 본인 구매내역 수정
    Query theQuery = currentSession
        .createQuery("Update Sales " + "Set salesUserNo = 1" + "Where salesUserNo = :SalesUserNo");
    theQuery.setParameter("SalesUserNo", theUserId);
    theQuery.executeUpdate();           구매내역의 기록은 제거되면 안되기 때문에 유저의 ID를 알수없음으로 수정

    theQuery = currentSession.createQuery("Delete from Cart " + "Where cartUserNo = :UserId");
    theQuery.setParameter("UserId", theUserId);
    theQuery.executeUpdate();           본인찜목록과장바구니에 임여데이터가 남지 않도록 삭제

    theQuery = currentSession.createQuery("Delete from Wishlist " + "Where wishlistUserNo = :UserId");
    theQuery.setParameter("UserId", theUserId);
    theQuery.executeUpdate();

    theQuery = currentSession.createQuery("from Artwork " + "Where artworkArtistNo = :UserId");
    theQuery.setParameter("UserId", theUserId);
                                                본인의 작품을 제거하기 위해 본인의 ID를 가진 아트워크의 ID를 저장하여 메서드를 통해 삭제
    theQuery = currentSession
        .createQuery("Select artworkNo from Artwork " + "Where artworkArtistNo = " + theUserId);

    List<Integer> theArtworkIds = theQuery.getResultList();
    for (Integer theArtworkId : theArtworkIds)
        deleteArtwork(theArtworkId);
                                                권한테이블에서도 임여데이터가 남지 않도록 본인 이름권한 삭제
    theQuery = currentSession.createQuery("Delete from Authorities " + "Where userName = :UserName");
    theQuery.setParameter("UserName", userName);
    theQuery.executeUpdate();

    Users User = currentSession.get(Users.class, theUserId);
    currentSession.delete(User);      마지막으로 유저 데이터를 삭제
}
```

category, artwork, users 테이블 1번

| category_no | category_name |
|-------------|---------------|
| 1 | Non |

| artwork_no | artwork_name | artwork_artist_no | artwork_category_no | artwork_price | artwork_regdate | artwork_sold | artwork_level | artwork_filename |
|------------|-----------------|-------------------|---------------------|---------------|---------------------|--------------|---------------|---------------------|
| 1 | NotFoundArtwork | 1 | 1 | 0 | 2023-01-17 02:23:52 | 0 | normal | NotFoundArtwork.jpg |

| user_no | username | password | user_real_name | user_purchase_amount | user_category_no | enabled |
|---------|--------------|--------------|----------------|----------------------|------------------|---------|
| 1 | NotFoundUser | NotFoundUser | NotFoundUser | 0 | 1 | 0 |

핵심코드 - 작품구매 View단



► cart-list.jsp 장바구니에서 구매하기 버튼 클릭 시 구매하기 창으로 이동

```
<td>
    <c:url var="Buylink" value="/BuyArtwork">
        <c:param name="artworkNo" value="${Artwork.artworkNo}"></c:param>
    </c:url>
    <a href = "${Buylink}"><input type ="button" value = "바로 구매" class="btn btn-primary"></a>
    <a href = "${deletelink}"><input type ="button" value = "삭제" class="btn btn-secondary"
        onclick="if(!confirm('해당 상품을 장바구니에서 삭제하시겠습니까?')) return false"></a>
</td>
```

► buypage.jsp 구매 페이지에서 주문 버튼 클릭 시 작품 구매를 진행하는 컨트롤러로 매팅

```
<div class="buy_button_box">
    <c:url var="BuyArtwork" value="/BuyArtwork/Buy">
        <c:param name="artworkNo" value="${Artwork.artworkNo}"></c:param>
        <c:param name="artworkPrice" value="${Artwork.artworkPrice}"></c:param>
    </c:url>
    <a href="${BuyArtwork}" id="buy_button" class="btn btn-primary">구매하기</a>
</div>
```

```
//구매하기 =====  
//구매하기 - 장바구니에서 결제창으로 정보 넘기기  
@GetMapping("/BuyArtwork")  
public String getBuyArtworkPage(@RequestParam("artworkNo") int artworkNo, Model theModel) {  
    theModel.addAttribute("Artwork", artshopservice.getArtwork(artworkNo));  
    theModel.addAttribute("Users", artshopservice.getUsers());  
  
    return "buypage";  
}
```

핵심코드 - 작품구매 DemoController

```
//구매하기 - 최종결재내역  
@GetMapping("/BuyArtwork/Buy")  
public String buyArtwork(@RequestParam("artworkNo") int theArtworkNo,  
    @RequestParam("artworkPrice") int theArtworkPrice, Authentication auth, Model theModel) {  
    String username = auth.getName();  
    int theId = artshopservice.getIntUsers(username);  
    Users users = artshopservice.getUsers(theId);  
  
    우선 거래내역을 기록하는데 필요한 유저의 개인정보를 불러와서 저장합니다.  
  
    Sales theSales = new Sales();  
    theSales.setSalesUserNo(users.getUserNo());  
    theSales.setSalesArtworkNo(theArtworkNo);  
    theSales.setSalesArtworkPrice(theArtworkPrice);  
    theSales.setSalesPurchaseTime(LocalDateTime.now());  
    artshopservice.buyArtwork(theSales, users);  
  
    return "redirect:/";  
}
```

이후 새로운 판매내역 객체를 생성하여 jsp에서 받아온 아트워크 ID, 가격, 유저의 ID를 입력해준 뒤 유저 정보와 함께 판매를 진행하는 메서드로 보내줍니다.

핵심코드 - 작품구매 DAOImpl



```
//그림을 구매하고 나서 그 그림과 관련된 테이블 항목 지우기
@Override
public void buyArtwork(Sales theSales, Users user) {
    Session currentSession = sessionFactory.getCurrentSession();
    currentSession.save(theSales); 완성된 구매 내역을 테이블에 추가

    Query theQuery = currentSession
        .createQuery("Delete from Cart " + "Where cartArtworkNo = " + theSales.getSalesArtworkNo());
    theQuery.executeUpdate();

    theQuery = currentSession 관련 상품은 존재하지 않기 때문에 짐목록과 장바구니에 아트워크 ID를 넘겨 완전히 삭제를 하고
        .createQuery("Delete from Wishlist " + "Where wishlistArtworkNo = " + theSales.getSalesArtworkNo());
    theQuery.executeUpdate();

    theQuery = currentSession.createQuery(
        "Update Artwork " + "Set artworkSold = 1 " + "Where artworkNo = " + theSales.getSalesArtworkNo());
    theQuery.executeUpdate(); 아트워크를 판매 완료 상태로 전환
```

핵심코드 - 작품구매 DAOImpl



```
int price = user.getUserPurchaseAmount() + theSales.getSalesArtworkPrice();
theQuery = currentSession.createQuery(
    "Update Users " + "Set userPurchaseAmount = " + price + " Where userNo = " + theSales.getSalesUserNo());
theQuery.executeUpdate();

이후 유저의 총 구매액에 아트워크 가격을 합쳐서 유저의 총 구매액을 갱신 시켜주고 해당 유저가 가진 권한정보를 가져옴
theQuery = currentSession.createQuery("Select authority " + "From Authorities " + "Where userName = :username");
theQuery.setParameter("username", user.getUserName());
List<String> theauthorities = theQuery.getResultList();

if (price >= 3000000)
    if (!theauthorities.contains("ROLE_VIP")) {
        Authorities theAuthority = new Authorities();
        theAuthority.setuserName(user.getUserName());
        theAuthority.setAuthority("ROLE_VIP");
        currentSession.save(theAuthority);
    }

해당유저의 총 구매액이 300만원을 넘었을 때는 VIP권한을 부여,

if (price >= 6000000)
    if (!theauthorities.contains("ROLE_VVIP")) {
        Authorities theAuthority = new Authorities();
        theAuthority.setuserName(user.getUserName());
        theAuthority.setAuthority("ROLE_VVIP");
        currentSession.save(theAuthority);
    }

600만원을 넘었을 경우 VVIP권한을 부여,

이미 각 권한이 존재할 경우, 과정 없이 매서드를 종료
```

Controller 구성도

LoginController 전반적인 동작 메서드

```
1 package com.code.springdemo.controller;
2
3 import java.util.List;
4
5 @Controller
6 public class LoginController {
7     @Autowired
8     private ArtshopService artshopservice;
9
10    //로그인
11    @GetMapping("/showMyLoginPage")
12    public String showMyLoginPage() {
13        return "fancy-login";
14    }
15
16    //로그아웃
17    @GetMapping("/logout")
18    public String logout() {
19        return "redirect:/";
20    }
21
22    //권한제한 페이지
23    @GetMapping("/access-denied")
24    public String showDenied(Authentication auth, Model theModel) {
25        String username = auth.getName();
26        String authority = artshopservice.getAuth(username);
27
28        theModel.addAttribute("Role", authority);
29
30        return "access-denied";
31    }
32
33    //기존 회원 정보 보여주기
34    @GetMapping("/change-pw")
35    public String change_pw(Authentication authentication, Model theModel) {
36        String userName = authentication.getName();
37        int theId = artshopservice.getIntUsers(userName);
38        Users theUser = artshopservice.getUsers(theId);
39
40        theModel.addAttribute("Users", theUser);
41
42        return "change-pw";
43    }
44
45    //변경할 개인정보 수정하기
46    @GetMapping("/changepw")
47    public String changepw(Authentication authentication, @ModelAttribute("Users") Users theusers, Model theModel) {
48        String userName = authentication.getName();
49        int theId = artshopservice.getIntUsers(userName);
50        artshopservice.updateUsers(theId, theusers);
51
52        theModel.addAttribute("unregister", "logout");
53
54        return "home";
55    }
56
57    //일반유저 회원가입
58    @GetMapping("/common_register")
59    public String common_register() {
60        return "common-register";
61    }
62
63    //아티스트 회원가입
64    @GetMapping("/artist_register")
65    public String artist_register() {
66        return "artist-register";
67    }
68
69    //회원가입 처리하기
70    @GetMapping("/register")
71    public String register(Users theuser, Authorities theauthorities, Model theModel) {
72
73        artshopservice.register(theuser, theauthorities);
74
75        return "redirect:/showMyLoginPage";
76    }
77
78    //회원 탈퇴하기
79    @GetMapping("/unregister")
80    public String unregister(Authentication authentication, Model theModel) {
81        String userName = authentication.getName();
82        int theId = artshopservice.getIntUsers(userName);
83        System.out.println(userName);
84        artshopservice.unregister(theId);
85
86        theModel.addAttribute("unregister", "logout");
87
88        return "home";
89    }
90
91    //변경할 개인정보 수정하기
92    @GetMapping("/changepw")
93    public String changepw(Authentication authentication, Model theModel) {
94        String userName = authentication.getName();
95        int theId = artshopservice.getIntUsers(userName);
96
97        theModel.addAttribute("unregister", "logout");
98
99        return "home";
100    }
101}
```

Controller 구성도

홈 화면 컨트롤러 및 카테고리 화면 컨트롤러

```
DemoController.java ×
1 package com.code.springdemo.controller;
2
3 import java.sql.SQLException;□
4
5 @Controller
6 public class DemoController {
7     @Autowired
8     private ArtshopService artshopservice;
9
10    //홈 =====
11    @GetMapping("/")
12    public String showHome(Model theModel) throws SQLException {
13        //새로 들어온 작품(입고일자 기반) - 캐로셀 슬라이드
14        List<Artwork> newArtworks = artshopservice.getNewArtworks();
15        theModel.addAttribute("newArtworks", newArtworks);
16        //아티스트 정보
17        List<Users> Users = artshopservice.getUsers();
18        theModel.addAttribute("Users", Users);
19        //작품 정보
20        List<Artwork> theArtworks = artshopservice.getArtworkSoldlist();
21        theModel.addAttribute("artworks", theArtworks);
22        //베스트 위시 작품(찜 개수 기반)
23        List<Wishlist> bestWish = artshopservice.getBestWish();
24        theModel.addAttribute("bestWish", bestWish);
25        return "home";
26    }
27    //홈 - 카테고리별 페이지
28    @GetMapping("/byCategory")
29    public String showByCategory(Model theModel) {
30        List<Category> category = artshopservice.getCategory();
31        theModel.addAttribute("category", category);
32        List<Artwork> theArtworks = artshopservice.getArtworkSoldlist();
33        theModel.addAttribute("artworks", theArtworks);
34        List<Users> users = artshopservice.getUsers();
35        theModel.addAttribute("users", users);
36        return "by-category";
37    }
38}
```

작가별 페이지 및 vip 페이지

```
DemoController.java ×
58
59    //홈 - 작가별 소개 페이지
60    @GetMapping("/byArtist")
61    public String showByArtist(Model theModel) {
62        List<Users> artists = artshopservice.getArtists();
63        theModel.addAttribute("artists", artists);
64
65        List<Artwork> theArtworks = artshopservice.getArtworkSoldlist();
66        theModel.addAttribute("artworks", theArtworks);
67
68        return "by-artist";
69    }
70
71    //홈 - Vip 페이지
72    @GetMapping("/vipPage")
73    public String showVip(Authentication auth, Model theModel) {
74        String username = auth.getName();
75        int theId = artshopservice.getIntUsers(username);
76        Users user = artshopservice.getUsers(theId);
77        theModel.addAttribute("User", user);
78
79        String authority = artshopservice.getAuth(username);
80        theModel.addAttribute("Role", authority);
81        theModel.addAttribute("Users", artshopservice.getUsers());
82
83        List<Artwork> artworksVip = artshopservice.getArtworksVip();
84        theModel.addAttribute("artworksVip", artworksVip);
85
86        List<Artwork> artworksVvip = artshopservice.getArtworksVvip();
87        theModel.addAttribute("artworksVvip", artworksVvip);
88
89        return "list-vip";
90    }
```

Controller 구성도

찜 목록, 장바구니 및 유저 마이페이지

```
103 //찜 - 찜목록 페이지
104 @GetMapping("/showWish")
105 public String showWish(Authentication auth, Model theModel) {
106     List<Wishlist> wishes = artshopservice.wishlists();
107     List<Artwork> theArtworks = artshopservice.getArtworkSoldlist();
108     String username = auth.getName();
109     int theId = artshopservice.getIntUsers(username);
110     Users users = artshopservice.getUsers(theId);
111     theModel.addAttribute("Users", artshopservice.getUsers());
112     theModel.addAttribute("Artworks", theArtworks);
113     theModel.addAttribute("Wishes", wishes);
114     theModel.addAttribute("User", users);
115     return "wish-list";
116 }
117 //찜 - 장바구니 페이지
118 @GetMapping("/showcart")
119 public String showCart(Authentication auth, Model theModel) {
120     List<Cart> carts = artshopservice.Cartslists();
121     List<Artwork> theArtworks = artshopservice.getArtworkSoldlist();
122     String username = auth.getName();
123     int theId = artshopservice.getIntUsers(username);
124     Users users = artshopservice.getUsers(theId);
125
126     theModel.addAttribute("Carts", carts);
127     theModel.addAttribute("User", users);
128     theModel.addAttribute("Users", artshopservice.getUsers());
129     theModel.addAttribute("Artworks", theArtworks);
130     return "cart-list";
131 }
132 //유저 마이페이지 =====
133 @GetMapping("/userPage")
134 public String showUser(Authentication auth, Model theModel) {
135     String username = auth.getName();
136     int theId = artshopservice.getIntUsers(username);
137     Users users = artshopservice.getUsers(theId);
138     theModel.addAttribute("Users", users);
139
140     String authority = artshopservice.getAuth(username);
141     theModel.addAttribute("Role", authority);
142     return "user-mypage";
143 }
144 }
```

유저 마이페이지 주문내역 및 작가 마이페이지

```
145 //유저 마이페이지 - 주문내역
146 @GetMapping("/order-list")
147 public String orderList(Authentication auth, Model theModel) {
148
149     String username = auth.getName();
150     int theId = artshopservice.getIntUsers(username);
151     Users user = artshopservice.getUsers(theId);
152     theModel.addAttribute("user", user);
153     theModel.addAttribute("Users", artshopservice.getUsers());
154     theModel.addAttribute("Artworks", artshopservice.getArtworks());
155     theModel.addAttribute("Sales", artshopservice.getSales());
156
157     return "user-order-list";
158 }
159
160 //작가 마이페이지 =====
161 @GetMapping("/artistPage/artist")
162 public String showArtist(Authentication auth, Model theModel) {
163
164     theModel.addAttribute("thisPage", "ArtworkShow");
165
166     String username = auth.getName();
167     int theId = artshopservice.getIntUsers(username);
168     Users users = artshopservice.getUsers(theId);
169     theModel.addAttribute("Users", users);
170
171     int userNo = users.getUserNo();
172
173     List<Artwork> artworksSold = artshopservice.getArtworksSold(userNo, true);
174     theModel.addAttribute("artworksSold", artworksSold); //true => 판매완료
175
176     List<Artwork> artworksSaleNormal = artshopservice.getArtworksSaleNormal(userNo, false, "normal");
177     theModel.addAttribute("artworksSaleNormal", artworksSaleNormal); //false => 일반 작품 판매중
178
179     List<Artwork> artworksSaleVip = artshopservice.getArtworksSaleVip(userNo, false, "vip");
180     theModel.addAttribute("artworksSaleVip", artworksSaleVip); //false => vip 혹은 vvip 작품 판매중
181
182     return "artist-mypage";
183 }
184 }
```

Controller 구성도

작가マイ페이지Update&Delete, 작품업로드기능 및 작품정보수정

```
(LoginController.java) * DemoController.java ×
184
185     //작가 마이페이지 - update & delete
186     @GetMapping("/artistPage/showForUpdate")
187     public String showUpdateArtworkList(@RequestParam("artworkNo") int theId, Authentication auth, Model theModel) {
188         theModel.addAttribute("saveOrUpdate", "ArtworkUpdate");
189         theModel.addAttribute("thisPage", "ArtworkUpdate");
190         theModel.addAttribute("Artwork", artshopservice.getArtwork(theId));
191
192         String username = auth.getName();
193         int theId2 = artshopservice.getIntUsers(username);
194         Users users = artshopservice.getUsers(theId2);
195         theModel.addAttribute("Users", users);
196
197         return "artist-mypage";
198     }
199
200     //작가 마이페이지 - 작품 업로드
201     @GetMapping("/artistPage/showForUpload")
202     public String showUploadArtworkList(Authentication auth, Model theModel) {
203         String username = auth.getName();
204         int theId = artshopservice.getIntUsers(username);
205         Users users = artshopservice.getUsers(theId);
206         theModel.addAttribute("Users", users);
207         theModel.addAttribute("saveOrUpdate", "ArtworkUpload");
208         theModel.addAttribute("thisPage", "ArtworkUpdate");
209
210         Artwork artwork = new Artwork();
211         artwork.setArtworkLevel("none");
212         artwork.setArtworkArtistNo(users.getUserNo());
213         artwork.setArtworkSold(false);
214         artwork.setArtworkRegdate(LocalDateTime.now());
215         theModel.addAttribute("Artwork", artwork);
216
217         return "artist-mypage";
218     }
219
220     //작가 마이페이지 - 작품정보 수정
221     @GetMapping("/artistPage/UpdateArtwork")
222     public String saveOrUpdateArtwork(@ModelAttribute("Artwork") Artwork theArtwork, Model theModel) {
223         artshopservice.updateArtwork(theArtwork);
224
225         return "redirect:/artistPage/artist";
226     }
227 }
```

Controller 구성도

작가 마이페이지 작품 삭제 및 관리자 페이지 전반적인 홈페이지 관리

```
>LoginController.java *DemoController.java X
226
227     //작가 마이페이지 - 작품 삭제
228     @GetMapping("/artistPage/Delete")
229     public String deleteArtistArtwork(@RequestParam("artworkNo") int theId) {
230         artshopservice.deleteArtwork(theId);
231
232         return "redirect:/artistPage/artist";
233     }
234     //=====================================================================
235     //어드민 마이페이지 =====
236     @GetMapping("/adminPage/admin")
237     public String showAdmin() {
238         return "admin-mypage";
239     }
240     //=====================================================================
241     //어드민 마이페이지 - 유저 관리 =====
242     @GetMapping("/adminPage/UserList")
243     public String showsystemsUserList(Model theModel) {
244         theModel.addAttribute("thisPage", "UserList");
245         theModel.addAttribute("Users", artshopservice.getUsers());
246         theModel.addAttribute("Authorities", artshopservice.getAuthorities());
247
248     }
249     //어드민 마이페이지 - 유저 관리 - 유저 정보를 수정하기위한 페이지 출력
250     @GetMapping("/adminPage/UpdateUser")
251     public String UserUpdatePage(@RequestParam("userNo") int theId, Model theModel) {
252         theModel.addAttribute("thisPage", "UserUpdate");
253         theModel.addAttribute("User", artshopservice.getUsers(theId));
254
255     }
256     //어드민 마이페이지 - 유저 관리 - 유저정보 수정하는 메서드
257     @GetMapping("/adminPage/UpdateUser/Update")
258     public String saveUser(@ModelAttribute("User") Users theUser,@RequestParam("beforeName") String beforeName, Model theModel) {
259         List<String> theUsers = artshopservice.getUsersName();
260         if(!(theUsers.contains(theUser.getUserName())) || beforeName.equals(theUser.getUserName())) {
261             theUser.setPassword(theUser.getPassword().replace("{noop}", ""));
262             artshopservice.UpdateUser(theUser);
263             return "redirect:/adminPage/UserList";
264         }
265         else
266             return UserUpdatePage(theUser.getUserNo(),theModel);
267     }
```

Controller 구성도

관리자 페이지 유저 관리 작품 관리 등 전반적인 홈페이지 컨트롤

```
>LoginController.java | *DemoController.java ×
268 //어드민 마이페이지 - 유저 관리 - 유저를 삭제하는 메서드
269 @GetMapping("/adminPage/DeleteUser")
270 public String deleteUser(@RequestParam("userNo") int theId) {
271     artshopservice.deleteUser(theId);
272     return "redirect:/adminPage/UserList";
273 }
274
275 //어드민 마이페이지 - 유저 관리 - 유저 검색 메서드
276 @GetMapping("/adminPage/SearchUser")
277 public String searchUser(@RequestParam("searchText") String searchText, Model theModel) {
278     String trimname = searchText.replace(" ", "");
279     List<Users> Users = null;
280     System.out.println(trimname);
281     if(trimname != null)
282         Users = artshopservice.searchUser(trimname);
283     else
284         Users = artshopservice.getUsers();
285     theModel.addAttribute("Users", Users);
286     theModel.addAttribute("thisPage", "UserList");
287     theModel.addAttribute("Authorities", artshopservice.getAuthorities());
288     return "admin-mypage";
289 }
290
291 //어드민 마이페이지 - 아트워크 관리 =====
292 //어드민 마이페이지 - 아트워크 관리 - 아트워크 리스트 출력
293 @GetMapping("/adminPage/ArtworkList")
294 public String ArtworkListUpdatePage(Model theModel) {
295     theModel.addAttribute("thisPage", "ArtworkList");
296     theModel.addAttribute("Artworks", artshopservice.getArtworks());
297     theModel.addAttribute("Users", artshopservice.getUsers());
298     return "admin-mypage";
299 }
300
301 //어드민 마이페이지 - 아트워크 관리 - 아트워크 업데이트 페이지 출력
302 @GetMapping("/adminPage/ArtworkUpdate")
303 public String showsystemsArtworkList(@RequestParam("ArtworkNo") int theId, Model theModel) {
304
305     theModel.addAttribute("thisPage", "ArtworkUpdate");
306     theModel.addAttribute("Artwork", artshopservice.getArtwork(theId));
307     theModel.addAttribute("Users", artshopservice.getUsers());
308     return "admin-mypage";
309 }
```

Controller 구성도

관리자 페이지 작품 판매 내역 관리 및 찜 목록 추가

```
(LoginController.java) *DemoController.java ×

353
354     //어드민 마이페이지 - 판매 내역 관리 - 판매 내역 검색
355     @GetMapping("/adminPage/SearchSales")
356     public String searchSales(@RequestParam("searchText") String searchText, Model theModel) {
357
358         String trimname = searchText.replace(" ", "");
359         List<Sales> Sales = null;
360         System.out.println(trimname);
361         if(trimname != null)
362             Sales = artshopservice.searchSales(trimname);
363         else
364             Sales = artshopservice.getSales();
365         theModel.addAttribute("Sales", Sales);
366         theModel.addAttribute("Artworks", artshopservice.getArtworks());
367         theModel.addAttribute("Users", artshopservice.getUsers());
368         theModel.addAttribute("thisPage", "SalesList");
369         return "admin-mypage";
370     }
371     //찜 목록 =====
372     //찜 목록 - 찜 목록 추가
373     @GetMapping("/savewish")
374     public String saveWish(@ModelAttribute("artworkNum") int ArtworkNum, Authentication auth, Model theModel, HttpServletRequest request) {
375         if (auth==null)
376             return "redirect:/showMyLoginPage";
377
378         String username = auth.getName();
379         int userNo = artshopservice.getIntUsers(username);
380
381         List<Integer> myWish = artshopservice.myWish(userNo);
382
383         if(myWish.contains(ArtworkNum)) {
384             return "redirect:" + request.getHeader("referer");
385         }else {
386             Wishlist theWish = new Wishlist();
387             theWish.setWishlistUserNo(userNo);
388             theWish.setWishlistArtworkNo(ArtworkNum);
389
390             artshopservice.addWish(theWish);
391
392             return "redirect:" + request.getHeader("referer");
393         }
394     }
```

Controller 구성도

찜 목록 삭제 및 장바구니 추가

```
(LoginController.java) DemoController.java X  
395 //찜 목록 - 찜 목록 삭제  
397 @GetMapping("/deleteWish")  
398 public String deleteWish(@RequestParam("WishNo") int WishId, Model theModel) {  
399     artshopservice.deleteWish(WishId);  
400  
401     return "redirect:/showWish";  
402 }  
403 //찜 목록 전체삭제  
404 @GetMapping("/deleteAllWish")  
405 public String deleteallwish(@RequestParam("userId") int userId, Authentication auth, Model theModel) {  
406  
407     String userName = auth.getName();  
408  
409     userId = artshopservice.getIntUsers(userName);  
410  
411     artshopservice.deleteAllWish(userId);  
412  
413     return "redirect:/showWish";  
414 }  
415 //장바구니 ======  
416 //장바구니 - 장바구니 추가하기  
417 @GetMapping("/savecart")  
418 public String saveCart(@ModelAttribute("artworkNO") int ArtworkNo, Authentication auth, Model theModel, HttpServletRequest request) {  
419     if (auth==null)  
420         return "redirect:/showMyLoginPage";  
421     String userName = auth.getName();  
422  
423     int theId = artshopservice.getIntUsers(userName);  
424  
425     List<Integer> myCart = artshopservice.myCart(theId);  
426  
427     if(myCart.contains(ArtworkNo)) {  
428         return "redirect:"+request.getHeader("Referer");  
429     }else {  
430         Cart theCart = new Cart();  
431         theCart.setCartUserNo(theId);  
432         theCart.setCartArtworkNo(ArtworkNo);  
433         artshopservice.addCart(theCart);  
434         return "redirect:"+request.getHeader("Referer");  
435     }  
436 }
```

Controller 구성도

장바구니 삭제, 작품 상세페이지 및 구매하기 기능

```
>LoginController.java DemoController.java ×
438 //장바구니 - 장바구니 삭제
439 @GetMapping("/deleteCart")
440 public String deleteCart(@RequestParam("CartNO") int CartId, Model theModel) {
441     artshopservice.deleteCart(CartId);
442
443     return "redirect:/showcart";
444 }
445
446 //장바구니 전체삭제
447 @GetMapping("/deleteAllCart")
448 public String deleteAllCart(@RequestParam("userId") int userId, Authentication auth, Model theModel) {
449     String userName = auth.getName();
450
451     userId = artshopservice.getIntUsers(userName);
452
453     artshopservice.deleteAllCart(userId);
454
455     return "redirect:/showcart";
456 }
457
458 //작품 상세 페이지
459 @GetMapping("/detail")
460 public String detailView(@ModelAttribute("ArtworkNumber") int theArtworks, Model theModel) {
461     Artwork artworks = artshopservice.getArtwork(theArtworks);
462
463     theModel.addAttribute("Users", artshopservice.getUsers());
464     theModel.addAttribute("Cargorys", artshopservice.getCategory());
465     theModel.addAttribute("Artwork", artworks);
466
467     return "detail-artwork";
468 }
469
470 //구매하기 =====
471 //구매하기 - 장바구니에서 결제창으로 정보 넘기기
472 @GetMapping("/BuyArtwork")
473 public String getBuyArtworkPage(@RequestParam("artworkNo") int artworkNo, Model theModel) {
474     theModel.addAttribute("Artwork", artshopservice.getArtwork(artworkNo));
475     theModel.addAttribute("Users", artshopservice.getUsers());
476
477     return "buypage";
478 }
```

Controller 구성도

최종 구매내역 메서드

```
//구매하기 - 최종결제내역
@GetMapping("/BuyArtwork/Buy")
public String buyArtwork(@RequestParam("artworkNo") int theArtworkNo, @RequestParam("artworkPrice") int theArtworkPrice, Authentication auth, Model theModel) {
    String username = auth.getName();
    int theId = artshopservice.getIntUsers(username);
    Users users = artshopservice.getUsers(theId);

    Sales theSales = new Sales();
    theSales.setSalesUserNo(users.getUserNo());
    theSales.setSalesArtworkNo(theArtworkNo);
    theSales.setSalesArtworkPrice(theArtworkPrice);
    theSales.setSalesPurchaseTime(LocalDateTime.now());

    artshopservice.buyArtwork(theSales, users);

    return "redirect:/";
}
```

DAO 구성도

Java Resources

- src/main/java
 - com.code.springdemo.config
 - DemoAppConfig.java
 - DemoSecurityConfig.java
 - MySpringMvcDispatcherServletInitializer.java
 - SecurityWebApplicationInitializer.java
 - com.code.springdemo.controller
 - DemoController.java
 - LoginController.java
 - com.code.springdemo.dao
 - ArtshopDAO.java
 - ArtshopDAOImpl.java
 - com.code.springdemo.entity
 - Artwork.java
 - Authorities.java
 - Cart.java
 - Category.java
 - Sales.java
 - Users.java
 - Wishlist.java
 - com.code.springdemo.service
 - ArtshopService.java
 - ArtshopServiceImpl.java
 - com.code.testdb
- src/main/resources

persistence-mysql.properties

```
ArtshopServiceImpl.java DemoController.java ArtshopDAO.java X
1 package com.code.springdemo.dao;
2
3 import java.sql.SQLException;
4
5 public interface ArtshopDAO {
6
7     //회원 가입-----
8     public void register(Users theuser, Authorities theauthorities);
9     //회원 탈퇴-----
10    public void unregister(int theId);
11
12    //시제페이지에서 가져온 아이디 정보-----
13    public int getIntUsers(String userName);
14    //회원으로 로그인해서 로그인한 userNo를 가져와서 입력한 정보로 업데이트 하기-----
15    public void updateUsers(int theId, Users theusers);
16
17    public String duplicationConfirm(String loginId);
18
19    public List<Artwork> getArtworks(int userNo, boolean artworkSold);
20    //카테고리별 페이지-----
21    public List<Category> getCategory();
22
23    //작가별 페이지-----
24    public List<Users> getArtists();
25    //유저 관리-----
26    public List<Users> getUsers();
27
28    public Users getUsers(int theId);
29
30    public void UpdateUser(Users theUser);
31
32    public void deleteUser(int theId);
33    //아트워크 관리-----
34    public List<Artwork> getArtworks();
35
36    public Artwork getArtwork(int theUserId);
37
38    public void updateArtwork(Artwork theArtwork);
39
40    public void deleteArtwork(int theArtworkId);
41
42    public List<Artwork> searchArtworks(String searchText);
43
44    public List<Sales> getSales();
45
46    public List<Sales> searchSales(String searchText);
47
48    public List<Authorities> getAuthorities();
49
50    public List<Users> searchUser(String searchText);
51
52    public List<Artwork> getArtworksSold(int userNo, boolean artworkSold);
53
54    public List<Artwork> getArtworksSoldNormal(int userNo, boolean artworkSold, String normal);
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103 }
```

DAO Impl

회원가입 되어있는 유저 권한 추출

```
ArtshopDAOImpl.java X ArtshopDAO.java
1 package com.code.springdemo.dao;
2
3 import java.sql.Connection;
4
5 @Repository
6 public class ArtshopDAOImpl implements ArtshopDAO {
7
8     @Autowired
9     private SessionFactory sessionFactory;
10
11    @Autowired
12    private DataSource dataSource;
13
14 //LoginController =====
15 //권한 가져 오기
16 @Override
17
18 public String getAuth(String username) {
19     Session currentSession = sessionFactory.getCurrentSession();
20     Query<String> theQuery = currentSession
21         .createQuery("Select authority from Authorities where userName = '" + username + "'");
22     List<String> authorities = theQuery.getResultList();
23
24     if (authorities.contains("ROLE_ADMIN")) {
25         return "관리자";
26     } else if (authorities.contains("ROLE_VIP")) {
27         return "VIP";
28     } else if (authorities.contains("ROLE_VIP")) {
29         return "VIP";
30     } else {
31         return "일반";
32     }
33 }
```

회원가입, 탈퇴, 시큐리티에서 가져온 아이디 정보 추출

```
ArtshopDAOImpl.java X ArtshopDAO.java
54 // 회원 가입-----
55 @Override
56 public void register(Users theuser, Authorities theauthorities) {
57     Session currentSession = sessionFactory.getCurrentSession();
58     currentSession.saveOrUpdate(theuser);
59     currentSession.saveOrUpdate(theauthorities);
60 }
61 // 회원 탈퇴-----
62 @Override
63 public void unregister(int theId) {
64     Session currentSession = sessionFactory.getCurrentSession();
65     Users theQuery = currentSession.get(Users.class, theId);
66     currentSession.delete(theQuery);
67 }
68 // 시큐리티에서 가져온 아이디 정보-----
69 @Override
70 public int getIntUsers(String userName) {
71     System.out.println(userName);
72     Session currentSession = sessionFactory.getCurrentSession();
73     String hql = "from Users where userName = '" + userName + "'";
74     Query<Users> theQuery = currentSession.createQuery(hql, Users.class);
75
76     List<Users> theusers = theQuery.getResultList();
77
78     System.out.println(theusers);
79     System.out.println(theusers.get(0).getUserNo());
80
81     return theusers.get(0).getUserNo();
82 }
```

DAO impl

카테고리, 작가, 유저 정보 불러오기 및 유저 정보 수정

```
ArtshopDAOImpl.java X ArtshopDAO.java
83     // 회원으로 로그인해서 로그인한 userNo를 가져와서 입력한 정보로 업데이트 하기-----
84@override
85 public void updateUsers(int theId, Users theusers) {
86     Session currentSession = sessionFactory.getCurrentSession();
87     Query theQuery = currentSession.createQuery(
88         "Update Users Set userRealName = :UserRealName, password = :password Where userNo = :UserNo");
89     theQuery.setParameter("UserRealName", theusers.getUserRealName());
90     theQuery.setParameter("password", theusers.getPassword());
91     theQuery.setParameter("UserNo", theId);
92
93     theQuery.executeUpdate();
94 }
95 //아이디 중복확인
96@override
97 public String duplicationConfirm(String loginId) {
98     Session currentSession = sessionFactory.getCurrentSession();
99     Query<Users> theQuery = currentSession.createQuery("from users where username = '" + loginId + "'",
100             Users.class);
101    List<Users> users = theQuery.getResultList();
102    Users username = users.get(0);
103    String username2 = username.getUserName();
104
105    return username2;
106 }
107 }
```

DAOimpl

유저정보를 전체, 각각 불러오기 및 정보 수정

```
ArtshopDAOImpl.java × ArtshopDAO.java
110     //유저 정보 전부 불러오기
111     @Override
112     public List<Users> getUsers() {
113         Session currentSession = sessionFactory.getCurrentSession();
114         Query<Users> theQuery = currentSession.createQuery("from Users", Users.class);
115         List<Users> users = theQuery.getResultList();
116
117         return users;
118     }
119     //유저no를 통해 선택한 유저 정보만 가져오기
120     @Override
121     public Users getUsers(int theId) {
122
123         Session currentSession = sessionFactory.getCurrentSession();
124         Users theQuery = currentSession.get(Users.class, theId);
125
126         return theQuery;
127     }
128     //유저정보를 가져와서 유저 수정하기
129     @Override
130     public void UpdateUser(Users theUser) {
131         Session currentSession = sessionFactory.getCurrentSession();
132         currentSession.update(theUser);
133         String userBeforeName = getUsers(theUser.getUserNo()).getUserName();
134
135         Query theQuery =
136             currentSession.createQuery("Update Authorities "
137                 + "Set userName = :UserAfterName "
138                 + "Where userName = :UserBeforeName");
139         theQuery.setParameter("UserAfterName", theUser.getUserName());
140         theQuery.setParameter("UserBeforeName", userBeforeName);
141         theQuery.executeUpdate();
142     }
```

DAOimpl

권한내역 보기 및 아트워트 상세정보 관리 및 유저 받아오기

```
+ArtshopDAOImpl.java < ArtshopDAO.java
143     //유저no를 가져와서 유저에 관한 모든 테이블 정보 삭제
144     @Override
145     public void deleteUser(int theUserId) {
146         Session currentSession = sessionFactory.getCurrentSession();
147
148         String userName = getUsers(theUserId).getUserName();
149
150         Query theQuery = currentSession
151             .createQuery("Update Sales " + "Set salesUserNo = 1" + "Where salesUserNo = :SalesUserNo");
152         theQuery.setParameter("SalesUserNo", theUserId);
153         theQuery.executeUpdate();
154
155         theQuery = currentSession.createQuery("Delete from Cart " + "Where cartUserNo = :UserId");
156         theQuery.setParameter("UserId", theUserId);
157         theQuery.executeUpdate();
158
159         theQuery = currentSession.createQuery("Delete from Wishlist " + "Where wishlistUserNo = :UserId");
160         theQuery.setParameter("UserId", theUserId);
161         theQuery.executeUpdate();
162
163         theQuery = currentSession.createQuery("from Artwork " + "Where artworkArtistNo = :UserId");
164         theQuery.setParameter("UserId", theUserId);
165
166         theQuery = currentSession
167             .createQuery("Select artworkNo from Artwork " + "Where artworkArtistNo = " + theUserId);
168
169         List<Integer> theArtworkIds = theQuery.getResultList();
170         for (Integer theArtworkId : theArtworkIds)
171             deleteArtwork(theArtworkId);
172         theQuery = currentSession.createQuery("Delete from Authorities " + "Where userName = :UserName");
173         theQuery.setParameter("UserName", userName);
174         theQuery.executeUpdate();
175
176         Users User = currentSession.get(Users.class, theUserId);
177         currentSession.delete(User);
178     }
```

DAOimpl

유저정보와 아트워크 정보를 통해 판매 내역 추출

```
ArtshopDAOImpl.java < ArtshopDAO.java
429 //유저no랑 그림no를 통해서 판매 내역 검색하기
430 @Override
431 public List<Sales> searchSales(String searchText) {
432     Session currentSession = sessionFactory.getCurrentSession();
433     List<Users> theUsersQuery = searchUser(searchText);
434     List<Artwork> theArtworkQuery = searchArtworks(searchText);
435
436     StringBuilder SelectQuery = new StringBuilder("from Sales ");
437     boolean i = false;
438     for (Users UserQuery : theUsersQuery) {
439         if (i == false) {
440             SelectQuery.append("where salesUserNo = " + UserQuery.getUserNo());
441             i = true;
442         } else
443             SelectQuery.append(" or salesUserNo = " + UserQuery.getUserNo());
444     }
445
446     for (Artwork ArtworkQuery : theArtworkQuery) {
447         if (i == false) {
448             SelectQuery.append("where salesArtworkNo = " + ArtworkQuery.getArtworkNo());
449             i = true;
450         } else
451             SelectQuery.append(" or salesArtworkNo = " + ArtworkQuery.getArtworkNo());
452     }
453
454     Query<Sales> theQuery = currentSession.createQuery(String.valueOf(SelectQuery), Sales.class);
455     List<Sales> Sales = theQuery.getResultList();
456
457     return Sales;
458 }
459
460 }
```

DAOimpl

장바구니 추가,리스트 출력 및 중복 방지

```
ArtshopDAOImpl.java X ArtshopDAO.java
461      // 장바구니 -----
462      // 장바구니 중복 방지
463@override
△464  public List<Integer> myCart(int theId) {
465      Session currentSession = sessionFactory.getCurrentSession();
466
467      Query<Integer> SQL = currentSession.createQuery("Select cartArtworkNo from Cart Where cartUserNo = " + theId);
468
469      List<Integer> myCart = SQL.getResultList();
470
471      return myCart;
472  }
473  //장바구니 추가
474@override
△475  public void addCart(Cart theCart) {
476      Session currentSession = sessionFactory.getCurrentSession();
477      currentSession.saveOrUpdate(theCart);
478  }
479  //장바구니 리스트 출력
480@override
△481  public List<Cart> Cartslists() {
482      Session currentSession = sessionFactory.getCurrentSession();
483      Query<Cart> theQuery = currentSession.createQuery("from Cart", Cart.class);
484      List<Cart> Carts = theQuery.getResultList();
485
486      return Carts;
487  }
```

DAOimpl

유저 기반 장바구니 삭제 및 전체 삭제

```
ArtshopDAOImpl.java X ArtshopDAO.java
488     //장바구니 삭제
489     @Override
△490     public void deleteCart(int CartId) {
491         Session currentSession = sessionFactory.getCurrentSession();
492         Query<Cart> theQuery = currentSession.createQuery("delete from Cart where cartNo = " + CartId);
493
494         theQuery.executeUpdate();
495     }
496     // 장바구니 전체삭제
497     @Override
△498     public void deleteAllcart(int userId) {
499         Session currentSession = sessionFactory.getCurrentSession();
500         Query<Wishlist> theQuery = currentSession.createQuery("delete from Cart where cartUserNo = " + userId);
501
502         theQuery.executeUpdate();
503     }
504 }
```

DAOimpl

작가 기준 짐 목록 등록 순위 BestWish 상품 불러오기

```
*ArtshopDAOImpl.java × ArtshopDAO.java
549     // 짐 목록에 제일 많이 등록된 작가번호 순서 정렬해서 불러오기 - 베스트 위시
550     public void close(Connection conn, Statement myst, ResultSet myrs) {
551         try {
552             if (myrs != null) {
553                 myrs.close();
554             }
555             if (myst != null) {
556                 myst.close();
557             }
558             if (conn != null) {
559                 conn.close();
560             }
561         } catch (Exception e) {
562             e.printStackTrace();
563         }
564     }
565     @Override
566     public List<Wishlist> getBestWish() throws SQLException {
567         List<Wishlist> Bestwish = new ArrayList<Wishlist>();
568         Connection conn = null;
569         Statement myst = null;
570         ResultSet myrs = null;
571         try {
572             conn = dataSource.getConnection();
573             String sql = "SELECT wishlist_artwork_no FROM wishlist GROUP BY wishlist_artwork_no order by count(*) DESC limit 3";
574             myst = conn.createStatement();
575             myrs = myst.executeQuery(sql);
576             while (myrs.next()) {
577                 int wishlist_artwork_no = myrs.getInt("wishlist_artwork_no");
578                 Wishlist beshwish = new Wishlist(wishlist_artwork_no);
579                 Bestwish.add(beshwish);
580             }
581             return Bestwish;
582         } finally {
583             close(conn, myst, myrs);
584         }
585     }
586 }
```

아쉬웠던 점



- 사진 첨부 파일 업로드 기능을 구현하지 못한 점
- 권한 계층 Role Hierarchy 적용에 어려움을 느낀 점
- 지도 API를 활용한 미술관 위치가 구현이 안된 점
- JS를 활용하여 다양한 팝업 및 동작이 구현되지 않은 점



감사합니다.

K-COLLECTIONS 사이트 영상 시연



영상 시연 이 순서

1. 전반적인 artworks, artists 메뉴 부분 먼저 보여주기 (비회원상태로 보여주고 자연스럽게 로그인으로 넘어가기)
2. 유저 및 작가 회원가입 및 로그인 (작가 로그인 및 유저 로그인)
3. 작가 로그인 후 전반적인 로직 설명 (작가 마이페이지 구매내역 및 판매리스트 등)

작가페이지 작품업로드 작품레벨 none으로 넘어가서 관리자 페이지에 permission 뜨는 거 보여주기

4. 유저 로그인 후 전반적인 로직 설명 (유저 마이페이지 구매내역 및 장바구니 등)
5. 관리자 로그인 후 전반적인 로직 설명(유저 삭제, 수정, 아트워크 관리등)
6. 아트워크 리스트 장바구니 추가 및 찜 목록 추가 OR 삭제 및 전체 삭제
7. 아트워크 리스트 상세페이지 이동 시 구매하기 및 장바구니 버튼등 기능 확인
8. 아트워크 리스트 구매하기 시 구매 내역 업데이트 확인
9. 구매된 아트워크 리스트 업데이트 확인
10. 권한 없는 유저가 vip존 이동시 access-denied 페이지 뜨는 거
11. vip가 vvip존 들어가면 뜨는거
12. vvip로 로그인해서 둘 다 잘 뜨는 거 보여주기
13. 월 더널지