## 8장 완성 코**드**

### templates/articles/show.mustache

{{>**layouts**/**header**}}  
  
<**table class="table"**>  
 <**thead**>  
 <**tr**>  
 <**th scope="col"**>Id</**th**>  
 <**th scope="col"**>Title</**th**>  
 <**th scope="col"**>Content</**th**>  
 </**tr**>  
 </**thead**>  
 <**tbody**>  
 {{#**article**}}  
 <**tr**>  
 <**th**>{{**id**}}</**th**>  
 <**td**>{{**title**}}</**td**>  
 <**td**>{{**content**}}</**td**>  
 </**tr**>  
 {{/**article**}}  
 </**tbody**>  
</**table**>  
  
<**a href="/articles/**{{**article**.**id**}}**/edit" class="btn btn-primary"**>Edit</**a**>  
<**a href="/articles/**{{**article**.**id**}}**/delete" class="btn btn-danger"**>Delete</**a**>  
<**a href="/articles"**>Go to Article List</**a**>  
  
{{>**layouts**/**footer**}}

### controller/ArticleController.java

**package** com.example.firstproject.controller;  
  
**import** com.example.firstproject.dto.ArticleForm;  
**import** com.example.firstproject.entity.Article;  
**import** com.example.firstproject.repository.ArticleRepository;  
**import** lombok.extern.slf4j.Slf4j;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.ui.Model;  
**import** org.springframework.web.bind.annotation.GetMapping;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**import** org.springframework.web.bind.annotation.PostMapping;  
**import** org.springframework.web.servlet.mvc.support.RedirectAttributes;  
  
**import** java.util.List;  
  
@Slf4j  
@Controller  
**public class** ArticleController {  
 @Autowired  
 **private** ArticleRepository **articleRepository**;  
  
 @GetMapping(**"/articles/new"**)  
 **public** String newArticleForm() {  
 **return "articles/new"**;  
 }  
  
 @PostMapping(**"/articles/create"**)  
 **public** String createArticle(ArticleForm form) {  
 ***log***.info(form.toString());  
 *// System.out.println(form.toString());  
  
 // 1. DTO를 엔티티로 변환* Article article = form.toEntity();  
 ***log***.info(article.toString());  
 *// System.out.println(article.toString());  
  
 // 2. 리파지터리로 엔티티를 DB에 저장* Article saved = **articleRepository**.save(article);  
 ***log***.info(saved.toString());  
 *// System.out.println(saved.toString());* **return "redirect:/articles/"** + saved.getId();  
 }  
  
 @GetMapping(**"/articles/{id}"**) *// 데이터 조회 요청 접수* **public** String show(@PathVariable Long id, Model model) { *// 매개변수로 id 받아오기* ***log***.info(**"id = "** + id); *// id를 잘 받았는지 확인하는 로그 찍기  
  
 // 1. id를 조회하여 데이터 가져오기* Article articleEntity = **articleRepository**.findById(id).orElse(**null**);  
  
 *// 2. 모델에 데이터 등록하기* model.addAttribute(**"article"**, articleEntity);  
  
 *// 3. 뷰 페이지 반환하기* **return "articles/show"**;  
 }  
  
 @GetMapping(**"/articles"**)  
 **public** String index(Model model) {  
 *// 1. 모든 데이터 가져오기* List<Article> articleEntityList = **articleRepository**.findAll();  
  
 *// 2. 모델에 데이터 등록하기* model.addAttribute(**"articleList"**, articleEntityList);  
  
 *// 3. 뷰 페이지 설정하기* **return "articles/index"**;  
 }  
  
 @GetMapping(**"/articles/{id}/edit"**)  
 **public** String edit(@PathVariable Long id, Model model) {  
 *// 수정할 데이터 가져오기* Article articleEntity = **articleRepository**.findById(id).orElse(**null**);  
  
 *// 모델에 데이터 등록하기* model.addAttribute(**"article"**, articleEntity);  
  
 *// 뷰 페이지 설정하기* **return "articles/edit"**;  
 }  
  
 @PostMapping(**"/articles/update"**)  
 **public** String update(ArticleForm form) {  
 ***log***.info(form.toString());  
  
 *// 1. DTO를 엔티티로 변환하기* Article articleEntity = form.toEntity();  
 ***log***.info(articleEntity.toString());  
  
 *// 2. 엔티티를 DB로 저장하기  
 // 2-1. DB에서 기존 데이터 가져오기* Article target = **articleRepository**.findById(articleEntity.getId()).orElse(**null**);  
  
 *// 2-2. 기존 데이터 값을 갱신하기* **if** (target != **null**) {  
 **articleRepository**.save(articleEntity); *// 엔티티를 DB에 저장(갱신)* }  
  
 *// 3. 수정 결과 페이지로 리다이렉트 하기* **return "redirect:/articles/"** + articleEntity.getId();  
 }  
  
 @GetMapping(**"/articles/{id}/delete"**)  
 **public** String delete(@PathVariable Long id, RedirectAttributes rttr) {  
 ***log***.info(**"삭제 요청이 들어왔습니다!!"**);  
  
 *// 1. 삭제할 대상 가져오기* Article target = **articleRepository**.findById(id).orElse(**null**);  
 ***log***.info(target.toString());  
  
 *// 2. 대상 엔티티 삭제하기* **if** (target != **null**) {  
 **articleRepository**.delete(target);  
 rttr.addFlashAttribute(**"msg"**, **"삭제됐습니다!"**);  
 }  
  
 *// 3. 결과 페이지로 리다이렉트하기* **return "redirect:/articles"**;  
 }  
}

### templastes/layouts/header.mustache

<!doctype html>  
<html lang="en">  
<head>  
 *<!-- Required meta tags -->* <meta charset="utf-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
  
 *<!-- Bootstrap CSS -->* <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">  
  
 <title>Hello, world!</title>  
</head>  
<body>  
*<!-- navigation -->*<nav class="navbar navbar-expand-lg navbar-light bg-light">  
 <div class="container-fluid">  
 <a class="navbar-brand" href="#">Navbar</a>  
 <button class="navbar-toggler" type="button"   
 data-bs-toggle="collapse"  
 data-bs-target="#navbarSupportedContent"   
 aria-controls="navbarSupportedContent"   
 aria-expanded="false" aria-label="Toggle navigation">  
 <span class="navbar-toggler-icon"></span>  
 </button>  
 <div class="collapse navbar-collapse" id="navbarSupportedContent">  
 <ul class="navbar-nav me-auto mb-2 mb-lg-0">  
 <li class="nav-item">  
 <a class="nav-link active"   
 aria-current="page" href="#">Home</a>  
 </li>  
 <li class="nav-item">  
 <a class="nav-link" href="#">Link</a>  
 </li>  
 <li class="nav-item dropdown">  
 <a class="nav-link dropdown-toggle"   
 href="#" id="navbarDropdown" role="button"   
 data-bs-toggle="dropdown" aria-expanded="false">  
 Dropdown  
 </a>  
 <ul class="dropdown-menu"   
 aria-labelledby="navbarDropdown">  
 <li><a class="dropdown-item" href="#">Action</a></li>  
 <li><a class="dropdown-item" href="#">  
 Another action</a></li>  
 <li><hr class="dropdown-divider"></li>  
 <li><a class="dropdown-item" href="#">  
 Something else here</a></li>  
 </ul>  
 </li>  
 <li class="nav-item">  
 <a class="nav-link disabled" href="#" tabindex="-1"   
 aria-disabled="true">Disabled</a>  
 </li>  
 </ul>  
 <form class="d-flex">  
 <input class="form-control me-2" type="search"  
 placeholder="Search" aria-label="Search">  
 <button class="btn btn-outline-success"  
 type="submit">Search</button>  
 </form>  
 </div>  
 </div>  
</nav>  
  
{{#msg}}  
<div class="alert alert-primary alert-dismissible">  
 {{msg}}  
 <button type="button" class="btn-close" data-bs-dismiss="alert"  
 aria-label="Close"></button>  
</div>  
{{/msg}}

### 셀프체크 정답

#### controller/MemberController.java

**package** com.example.firstproject.controller;  
  
**import** com.example.firstproject.entity.Member;  
**import** com.example.firstproject.dto.MemberForm;  
**import** com.example.firstproject.repository.MemberRepository;  
**import** lombok.extern.slf4j.Slf4j;  
**import** org.springframework.beans.factory.annotation.Autowired;  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.ui.Model;  
**import** org.springframework.web.bind.annotation.GetMapping;  
**import** org.springframework.web.bind.annotation.PathVariable;  
**import** org.springframework.web.bind.annotation.PostMapping;  
**import** org.springframework.web.servlet.mvc.support.RedirectAttributes;  
  
@Slf4j  
@Controller  
**public class** MemberController {  
  
 @Autowired  
 MemberRepository **memberRepository**;  
  
 @GetMapping(**"/members/new"**)  
 **public** String newMemberForm() {  
 **return "members/new"**;  
 }  
  
 @GetMapping(**"/signup"**)  
 **public** String signUpPage() {  
 **return "members/new"**;  
 }  
  
 @PostMapping(**"/join"**)  
 **public** String join(MemberForm memberForm) {  
 ***log***.info(memberForm.toString());  
 Member member = memberForm.toEntity();  
 ***log***.info(member.toString());  
 Member saved = **memberRepository**.save(member);  
 ***log***.info(saved.toString());  
 **return "redirect:/members/"** + saved.getId();  
 }  
  
 @GetMapping(**"/members/{id}"**)  
 **public** String show(@PathVariable Long id, Model model) {  
 Member member = **memberRepository**.findById(id).orElse(**null**);  
 model.addAttribute(**"member"**, member);  
 **return "members/show"**;  
 }  
  
 @GetMapping(**"/members"**)  
 **public** String index(Model model) {  
 Iterable<Member> members = **memberRepository**.findAll();  
 model.addAttribute(**"members"**, members);  
 **return "members/index"**;  
 }  
  
 @GetMapping(**"/members/{id}/edit"**)  
 **public** String edit(@PathVariable Long id, Model model) {  
 Member memberEntity = **memberRepository**.findById(id).orElse(**null**);  
 model.addAttribute(**"member"**, memberEntity);  
 **return "members/edit"**;  
 }  
  
 @PostMapping(**"/members/update"**)  
 **public** String update(MemberForm form) {  
 ***log***.info(form.toString());  
 Member memberEntity = form.toEntity();  
 Member target = **memberRepository**.findById(memberEntity.getId()).orElse(**null**);  
 **if** (target != **null**) {  
 **memberRepository**.save(memberEntity);  
 }  
 **return "redirect:/members/"** + memberEntity.getId();  
 }  
  
 @GetMapping(**"/members/{id}/delete"**)  
 **public** String delete(@PathVariable Long id,  
 RedirectAttributes rttr,  
 Model model) {  
 ***log***.info(**"삭제 요청이 들어왔습니다!!"**);  
 *// 1. 삭제 대상을 가져옴* Member target = **memberRepository**.findById(id).orElse(**null**);  
 ***log***.info(target.toString());  
 *// 2. 대상 삭제* **if** (target != **null**) {  
 **memberRepository**.delete(target);  
 rttr.addFlashAttribute(**"msg"**, **"삭제됐습니다."**);  
 }  
 *// 3. 결과 페이지로 리다이렉트* **return "redirect:/members"**;  
 }  
}

#### templates/members/show.mustache

{{>**layouts**/**header**}}  
<**table class="table"**>  
 <**thead**>  
 <**tr**>  
 <**th scope="col"**>Id</**th**>  
 <**th scope="col"**>Email</**th**>  
 <**th scope="col"**>Password</**th**>  
 </**tr**>  
 </**thead**>  
 <**tbody**>  
 {{#**member**}}  
 <**tr**>  
 <**th**>{{**id**}}</**th**>  
 <**td**>{{**email**}}</**td**>  
 <**td**>{{**password**}}</**td**>  
 </**tr**>  
 {{/**member**}}  
 </**tbody**>  
</**table**>  
<**a href="/members/**{{**member**.**id**}}**/edit" class="btn btn-primary"**>수정하기</**a**>  
<**a href="/members/**{{**member**.**id**}}**/delete" class="btn btn-danger"**>삭제하기</**a**>  
<**a href="/members"**>Go to member list</**a**>  
  
{{>**layouts**/**footer**}}