16 장 완성 코드

controller/ArticleController.java

```
package com.example.firstproject.controller;

import com.example.firstproject.dto.ArticleForm;
import com.example.firstproject.dto.CommentDto;
import com.example.firstproject.entity.Article;
import com.example.firstproject.repository.ArticleRepository;
import com.example.firstproject.service.CommentService;
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;

    @Autowired
    private CommentService commentService; // 서비스 객체 주입

    @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);
        List<CommentDto> commentsDtos = commentService.comments(id);

        // 2. 모델에 데이터 등록하기
        model.addAttribute("article", articleEntity);
        model.addAttribute("commentDtos", commentsDtos); // 댓글 목록 모델에
등록

        // 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(갱
신)
        }

        // 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";
    }

}
```

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>

{{>comments/_comments}}
{{>layouts/footer}}
```

comments/comments.mustache

```
<div>
    <!-- 댓글 목록 보기 -->
    {{>comments/_list}}
    <!-- 새 댓글 작성하기 -->
    {{>comments/_new}}
</div>
```

comments/_list.mustache

```
<div id="comments-list">
    {{#commentDtos}}
        <div class="card m-2" id="comments-{{id}}">
            <div class="card-header">
                {{nickname}}
```

```
            </div>
            <div class="card-body">
                {{body}}
            </div>
        </div>
    {{/commentDtos}}
</div>
```