

## 4장. 회원 및 게시판 관리 시스템



*Member And Board*



Spring Boot

# 회원 및 게시판 관리 시스템

Good Releation - 좋은 사이

---



[로그인](#) | [회원 가입](#) | [회원 목록](#) | [게시판](#)



# 회원 및 게시판 관리 시스템

## 프로젝트 계층 구조도

```
> Members2 [boot] [devtools] [javaweb main]
├── src/main/java
│   ├── com.springboot
│   │   ├── controller
│   │   │   ├── BoardController.java
│   │   │   ├── HomeController.java
│   │   │   └── MemberController.java
│   │   ├── dto
│   │   │   ├── BoardDTO.java
│   │   │   └── MemberDTO.java
│   │   ├── entity
│   │   │   ├── Board.java
│   │   │   └── Member.java
│   │   ├── repository
│   │   │   ├── BoardRepository.java
│   │   │   └── MemberRepository.java
│   │   └── service
│   │       ├── BoardService.java
│   │       └── MemberService.java
```

```
└── src/main/resources
    ├── static
    ├── templates
    │   ├── board
    │   │   ├── detail.html
    │   │   ├── list.html
    │   │   ├── pages.html
    │   │   ├── update.html
    │   │   └── write.html
    │   ├── error
    │   │   └── 404.html
    │   └── member
    │       ├── info.html
    │       ├── join.html
    │       ├── list.html
    │       ├── login.html
    │       └── index.html
    └── application.properties
```



# 회원 및 게시판 관리 시스템

## ■ 프로젝트 생성 및 기본 설정

Service URL	<input type="text" value="https://start.spring.io"/>		
Name	<input type="text" value="Members"/>		
<input checked="" type="checkbox"/> Use default location			
Location	<input type="text" value="D:\큐비트온&amp;스페이스시엘\javaweb\springworks\Me"/> <input type="button" value="Browse"/>		
Type:	<input type="text" value="Maven"/>	Packaging:	<input type="text" value="Jar"/>
Java Version:	<input type="text" value="21"/>	Language:	<input type="text" value="Java"/>
Group	<input type="text" value="com.springboot"/>		
Artifact	<input type="text" value="Members"/>		
Version	<input type="text" value="0.0.1-SNAPSHOT"/>		
Description	<input type="text" value="Demo project for Spring Boot"/>		
Package	<input type="text" value="com.springboot"/>		

[New] -> [Spring Starter Project]

Name – Members

Type - Maven

Packing – Jar

Java Version – 21

Group – com.springboot


Package – com.springboot.jp



# 회원 및 게시판 관리 시스템

## ■ 프로젝트 생성 및 기본 설정

### New Spring Starter Project Dependencies



Spring Boot Version:

Frequently Used:

<input checked="" type="checkbox"/> Lombok	<input checked="" type="checkbox"/> MySQL Driver	<input checked="" type="checkbox"/> Spring Boot DevTools
<input checked="" type="checkbox"/> Spring Data JPA	<input checked="" type="checkbox"/> Spring Web	<input checked="" type="checkbox"/> Thymeleaf

Available:

- ▶ AI
- ▶ Developer Tools
- ▶ Google Cloud
- ▶ I/O

Selected:

- X Spring Boot DevTools
- X Lombok
- X Spring Data JPA
- X MySQL Driver
- X Thymeleaf
- X Spring Web



# 회원 및 게시판 관리 시스템

## ▪ JPA 및 MySQL 설정

```
# DataSource 설정 - mysql
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://127.0.0.1:3306/bootdb?serverTime=Asia/Seoul
spring.datasource.username=bootuser
spring.datasource.hikari.password=pwboot

# JPA 설정
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
```



# 회원 및 게시판 관리 시스템

- HomeController

```
@Controller
public class HomeController {

    @GetMapping("/")
    public String Home() {
        return "index";
    }
}
```



# 회원 및 게시판 관리 시스템

## ▪ index.html

```
<section id="container">
  <div id="main">
    <h2>Good Releation - 좋은 사이</h2>
    <div class="main-pic">
      
    </div>
    <div class="menu">
      <th:block th:if="${session.loginEmail}">
        <span style="color: ■ blue"
          th:text="'[' + ${session.loginName} + '님]'"></span>
        <a th:href="@{/members/logout}" th:text='로그아웃'></a> |
      </th:block>
      <th:block th:unless="${session.loginEmail}">
        <a th:href="@{/members/login}" th:text='로그인'></a> |
      </th:block>

      <a th:href="@{/members/join}">회원 가입</a> |
      <a th:href="@{/members}">회원 목록</a> |
      <a th:href="@{/boards/pages}">게시판</a>
    </div>
  </div>
</section>
```





# 회원 및 게시판 관리 시스템

## ▪ style.css

```
/* 공통 스타일 */
#container{
    width: 80%;
    margin: 0 auto;
}
ul li{list-style: none; margin: 16px;}
table thead{background-color: #eee;}
button{padding: 5px 10px;}
.msg{color: #00f; margin: 10px;}
.error{color: #f00; margin: 10px;}

/* index 스타일 */
#main{text-align: center;}
h2{
    border-bottom: 2px solid #ccc;
    padding: 16px;
    text-align: center;
}
.main-pic img{width: 40%; border-radius: 8px;}
.menu{margin: 10px;}
```



# 회원 및 게시판 관리 시스템

## ▪ style.css

```
/* join 스타일 */
.joinForm, .LoginForm{
    width: 400px;
    margin: 0 auto;
}
.joinForm label, .LoginForm label{
    width: 80px;
    float: left;
}

/* memberList 스타일 */
.memberList, .boardList{
    width: 600px;
    margin: 20px auto;
    border: 1px solid #ccc;
    border-collapse: collapse;
}
.memberList, th, td, .boardList th, td{
    border: 1px solid #ccc;
    padding: 5px 10px;
}
```

```
/* memberInfo 스타일 */
.memberInfo{
    width: 400px;
    margin: 0 auto;
}
.memberInfo label{
    width: 80px;
    float: left;
}
.btnInfo{margin: 16px;}
.btnInfo a{text-decoration: none}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberDTO

```
@ToString  
@Setter  
@Getter  
public class MemberDTO {  
    private Long id;  
    private String email;  
    private String passwd;  
    private String name;  
    private String gender;  
    private Timestamp joinDate;  
}
```



# 회원 및 게시판 관리 시스템

## ▪ Member

```
@Data
@Table(name = "t_member")
@Entity
public class Member {
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private Long id;

    @Column(unique=true)
    private String email;

    @Column(nullable=false)
    private String passwd;

    @Column(length=30, nullable=false)
    private String name;
```



# 회원 및 게시판 관리 시스템

## ▪ Member

```
@Column(length=10)
private String gender;

@CreationTimestamp
private Timestamp joinDate;

//DTO를 Entity로 변환하는 메서드
public static Member toSaveEntity(MemberDTO dto) {
    Member member = new Member();
    member.setEmail(dto.getEmail());
    member.setPasswd(dto.getPasswd());
    member.setName(dto.getName());
    member.setGender(dto.getGender());

    return member;
}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberRepository

```
public interface MemberRepository extends JpaRepository<Member, Long>{  
  
    //이메일로 회원을 검색하는 메서드  
    Optional<Member> findByEmail(String email);  
}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberService

```
@AllArgsConstructor
@Service
public class MemberService {

    MemberRepository repository;

    //회원 추가
    public void save(MemberDTO dto) {
        if (repository.findByEmail(dto.getEmail()).isPresent()) {
            throw new IllegalArgumentException("이미 존재하는 이메일입니다.");
        }
        //DTO를 Entity로 변환 메서드 호출
        Member member = Member.toSaveEntity(dto);
        repository.save(member);
    }

    //회원 목록
    public List<Member> findAll() {
        return repository.findAll();
    }
}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberService

```
//회원 정보
public Member findById(Long id) {
    /*Optional<Member> member = repository.findById(id);
    return member.orElse(null);*/
    //예외 처리
    return repository.findById(id)
        .orElseThrow(
            () -> new IllegalArgumentException("회원이 존재하지 않습니다. ID=" + id));
}

//회원 삭제
public void delete(Long id) {
    repository.deleteById(id);
}
```





# 회원 및 게시판 관리 시스템

## ▪ MemberService

```
//로그인
public MemberDTO login(String email, String passwd) {
    Member member = repository.findByEmail(email)
        .orElseThrow(() -> new IllegalArgumentException("존재하지 않는 이메일입니다.));

    if (!member.getPasswd().equals(passwd)) {
        throw new IllegalArgumentException("비밀번호가 일치하지 않습니다.");
    }

    MemberDTO dto = new MemberDTO();
    dto.setId(member.getId());
    dto.setEmail(member.getEmail());
    dto.setName(member.getName());
    dto.setGender(member.getGender());
    return dto;
}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberController

```
@RequiredArgsConstructor
@RequestMapping("/members")
@Controller
public class MemberController {
    private final MemberService service;

    @GetMapping("/join") //회원 가입 페이지
    public String joinForm() {
        return "member/join";
    }

    @PostMapping("/join") //회원 가입 처리
    public String join(@ModelAttribute MemberDTO dto,
        RedirectAttributes ra) {
        try {
            service.save(dto);
            ra.addFlashAttribute("msg", "회원가입 성공!");
            return "redirect:/members/login";
        } catch (Exception e) {
            ra.addFlashAttribute("error", e.getMessage());
            return "redirect:/members/join";
        }
    }
}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberController

```
@GetMapping //회원 목록
public String getMemberList(Model model) {
    List<Member> memberList = service.findAll();
    model.addAttribute("memberList", memberList);
    return "member/list";
}

@GetMapping("/{id}") //회원 정보(상세)
public String getMember(@PathVariable Long id,
                        Model model) {
    try {
        Member member = service.findById(id);
        model.addAttribute("member", member);
        return "member/info";
    } catch (IllegalArgumentException e) {
        model.addAttribute("errorMsg", e.getMessage());
        return "error/404";
    }
}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberController

```
//회원 삭제
@GetMapping("/delete/{id}")
public String deleteMember(@PathVariable Long id) {
    service.delete(id);
    return "redirect:/members";
}

//로그인 페이지
@GetMapping("/login")
public String loginForm() {
    return "member/login";
}

//로그아웃 처리
@GetMapping("/logout")
public String logout(HttpSession session) {
    session.invalidate();
    return "redirect:/";
}
```



# 회원 및 게시판 관리 시스템

## ▪ MemberController

```
//로그인 처리
@PostMapping("/login")
public String login(@RequestParam String email,
                    @RequestParam String passwd,
                    HttpSession session,
                    Model model,
                    RedirectAttributes ra) {

    try {
        MemberDTO member = service.login(email, passwd);
        //세션 발급 - 이메일, 이름
        session.setAttribute("loginEmail", member.getEmail());
        session.setAttribute("loginName", member.getName());
        return "redirect:/";
    } catch (Exception e) {
        ra.addFlashAttribute("error", e.getMessage());
        return "redirect:/members/login";
    }
}
```



# 회원 및 게시판 관리 시스템

## 회원 가입

이메일	<input type="text" value="coli@robot.com"/>
비밀번호	<input type="password" value="....."/>
이름	<input type="text" value="박선화"/>
성별	<input type="radio"/> 남 <input checked="" type="radio"/> 여
<input type="button" value="가입"/> <input type="button" value="취소"/>	



# 회원 및 게시판 관리 시스템

## ▪ join.html

```
<section id="container">
  <h2>회원 가입</h2>
  <form th:action="@{/members/join}" method="post"
        class="joinForm">
    <div th:if="${error}" class="error">
      [[${error}]]
    </div>
    <fieldset>
      <ul>
        <li>
          <label for="email">이메일</label>
          <input type="text" name="email">
        </li>
        <li>
          <label for="mid">비밀번호</label>
          <input type="password" name="passwd">
        </li>
      </ul>
    </fieldset>
  </form>
</section>
```



# 회원 및 게시판 관리 시스템

## ▪ join.html

```
<li>
    <label for="mid">이름</label>
    <input type="text" name="name">
</li>
<li>
    <label for="mid">성별</label>
    <input type="radio" name="gender" value="남자" checked>남
    <input type="radio" name="gender" value="여자">여
</li>
<li>
    <input type="submit" value="가입">
    <input type="reset" value="취소">
</li>
</ul>
</fieldset>
</form>
```





# 회원 및 게시판 관리 시스템

## 회원 목록

번호	이메일	이름	성별	가입일
1	<a href="mailto:today@space.com">today@space.com</a>	김기용	남자	2025-10-23 19:20
2	<a href="mailto:coli@robot.com">coli@robot.com</a>	박선화	여자	2025-10-23 21:55



# 회원 및 게시판 관리 시스템

## ▪ list.html

```
<h2>회원 목록</h2>
<table class="memberList">
  <thead>
    <tr>
      <th>번호</th>
      <th>이메일</th>
      <th>이름</th>
      <th>성별</th>
      <th>가입일</th>
    </tr>
  </thead>
  <tbody>
    <tr th:each="m: ${memberList}">
      <td th:text="${m.id}"></td>
      <td>
        <a th:href="@{/members/{id}(id=${m.id})}" th:text="${m.email}"></a>
      </td>
      <td th:text="${m.name}"></td>
      <td th:text="${m.gender}"></td>
      <td th:text="${#dates.format(m.joinDate, 'yyyy-MM-dd HH:mm')}"></td>
    </tr>
  </tbody>
</table>
```



# 회원 및 게시판 관리 시스템

- 세션이 있는 경우

## 회원 정보

이메일	<input type="text" value="today@space.com"/>
비밀번호	<input type="password" value="....."/>
이름	<input type="text" value="김기용"/>
성별	<input checked="" type="radio"/> 남 <input type="radio"/> 여

수정

삭제

목록



# 회원 및 게시판 관리 시스템

- 세션이 없는 경우

## 회원 정보

이메일	<input type="text" value="coli@robot.com"/>
비밀번호	<input type="password" value="....."/>
이름	<input type="text" value="우영우"/>
성별	<input type="radio"/> 남 <input checked="" type="radio"/> 여

목록



# 회원 및 게시판 관리 시스템

## ▪ info.html

```
<h2>회원 정보</h2>
<div class="memberInfo">
  <fieldset>
    <ul>
      <li>
        <label for="email">이메일</label>
        <input type="text" name="email"
          th:value="${member.email}" readonly>
      </li>
      <li>
        <label for="passwd">비밀번호</label>
        <input type="password" name="passwd"
          th:value="${member.passwd}" readonly>
      </li>
      <li>
        <label for="name">이름</label>
        <input type="text" name="name"
          th:value="${member.name}" readonly>
      </li>
    </ul>
  </fieldset>
</div>
```



# 회원 및 게시판 관리 시스템

## ▪ info.html

```
<li>
  <label for="gender">성별</label>
  <input type="radio" name="gender"
    value="남자" th:checked="${member.gender == '남자'}">남
  <input type="radio" name="gender"
    value="여자" th:checked="${member.gender == '여자'}">여
</li>
</ul>
</fieldset>
<div class="btnInfo">
  <th:block th:if="${member.email == session.loginEmail}">
    <a th:href="@{/members/edit/{id}(id=${member.id})}">
      <button>수정</button>
    </a>
    <a th:href="@{/members/delete/{id}(id=${member.id})}"
      onclick="return confirm('정말 삭제하시겠습니까?')">
      <button>삭제</button>
    </a>
  </th:block>
  <a th:href="@{/members}"><button>목록</button></a>
</div>
</div>
```



# 회원 및 게시판 관리 시스템

## 로그인

### 로그인

회원가입 성공!

이메일	<input type="text"/>
비밀번호	<input type="password"/>
<input type="button" value="로그인"/>	

회원 가입 후 이동

### 로그인

이메일	<input type="text" value="coli@robot.com"/>
비밀번호	<input type="password" value="....."/>
<input type="button" value="로그인"/>	



# 회원 및 게시판 관리 시스템

## 로그인

존재하지 않는 이메일입니다.

이메일

비밀번호

로그인

비밀번호가 일치하지 않습니다.

이메일

비밀번호

로그인





# 회원 및 게시판 관리 시스템

## ▪ login.html

```
<h2>로그인</h2>
<form th:action="@{/members/login}" method="post" class="loginForm">
  <div th:if="${msg}" class="msg">
    [[${msg}]]
  </div>
  <div th:if="${error}" class="error">
    [[${error}]]
  </div>
  <fieldset>
    <ul>
      <li>
        <label for="email">이메일</label>
        <input type="text" name="email">
      </li>
      <li>
        <label for="pwd">비밀번호</label>
        <input type="password" name="passwd">
      </li>
      <li>
        <input type="submit" value="로그인">
      </li>
    </ul>
  </fieldset>
</form>
```



# 회원 및 게시판 관리 시스템

- 로그아웃

Good Releation - 좋은 사이

---



[김기용님] [로그아웃](#) | [회원 가입](#) | [회원 목록](#) | [게시판](#)



# 회원 및 게시판 관리 시스템

## ■ 404.html

회원이 존재하지 않습니다. ID=6

[회원 목록](#) | [게시글 목록](#)

```
<section id="container">
  <h2 th:text="${errorMsg}">요청하신 정보를 찾을 수 없습니다.</h2>
  <a th:href="@{/members}">회원 목록</a> |
  <a th:href="@{/boards}">게시글 목록</a>
</section>
```

