A Glory Day 2

✔️ ***Spring Security 세팅***

✔️ ***Board***

✔️ ***Notice***

✔️ ***Member***

Spring Security 세팅

pom.xml

```
<!-- spring-security-web> -->
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-web</artifactId>
  <version>5.4.6</version>
</dependency>

<!-- spring-security-config> -->
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-config</artifactId>
  <version>5.4.6</version>
</dependency>

<!-- spring-security-taglibs> -->
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-taglibs</artifactId>
  <version>5.4.6</version>
</dependency>
```

web.xml

```
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>
    /WEB-INF/spring/root-context.xml
    /WEB-INF/spring/security-context.xml
  </param-value>
</context-param>
-------------------------------------------------------------------------------
<filter>
  <filter-name>springSecurityFilterChain</filter-name>
  <filter-class>org.springframework.web.filter.DelegatingFilterProxy
  </filter-class>
</filter>

<filter-mapping>
  <filter-name>springSecurityFilterChain</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

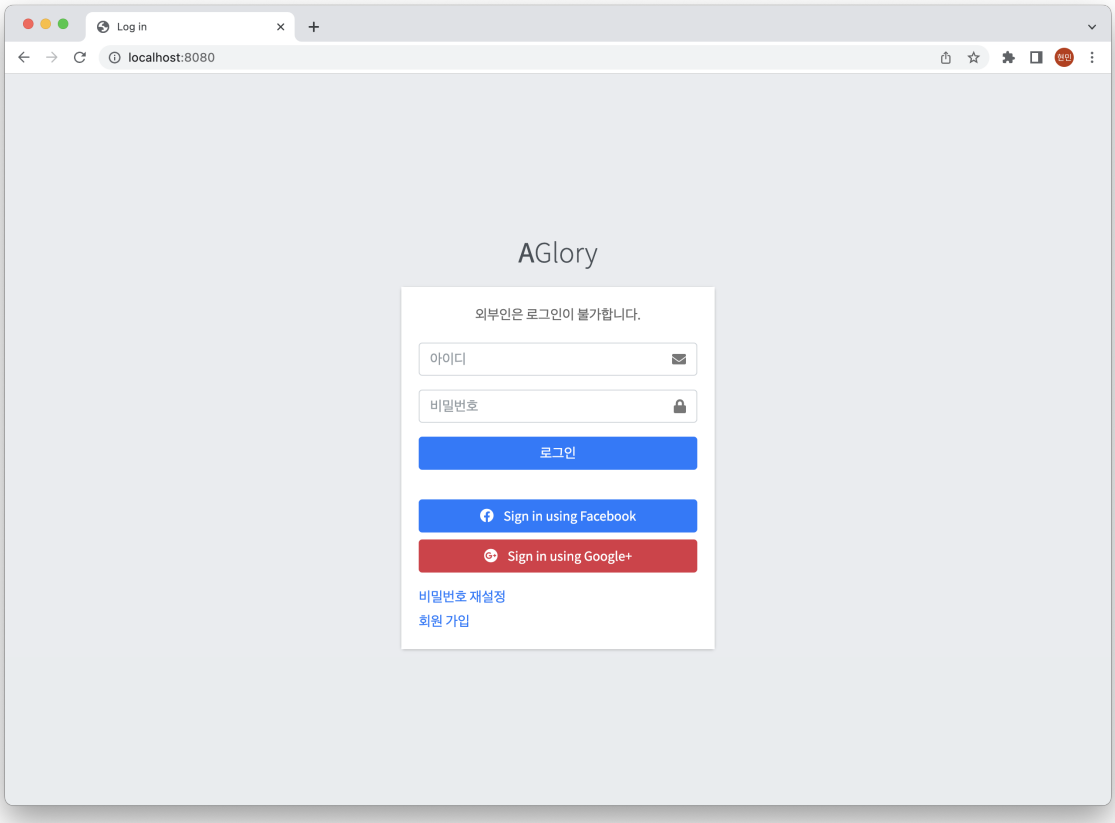
Security-context.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
  xmlns:security="http://www.springframework.org/schema/security"
  xsi:schemaLocation="http://www.springframework.org/schema/security http://www.springframework.org/schema/security/spring-
security.xsd
    http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

  <!--  각각의  interceptr-url, form-login,logout 은  내부적으로 Filter를 만들어 사용한다.  그래서 web.xml에서 이 모든걸 엮어줄 FilterChai
ng을 따로 설정해준다. -->
  <!-- web.xml에서 사용하는 FilterChain의 대한 설정부분이다. -->
  <security:http use-expressions="true">
    <security:intercept-url pattern="/notice/addnotice/**" access="hasAuthority('USER_MANAGER')" />
    <security:intercept-url pattern="/**" access="permitAll" />
    <security:intercept-url pattern="/notice/addnotice" access="hasAuthority('USER_MANAGER')" />
        <security:form-login login-page="/login"
                        default-target-url="/board/list"
                        authentication-failure-url="/loginfailed"
                        username-parameter="username"
                        password-parameter="password" />
        <security:csrf />
        <security:logout logout-success-url="/logout" />
    </security:http>
  <!-- form-login은 기본 로그인 폼 양식을 보여준다.logout은 로그아웃처리를.. -->

  <!-- 암호화를 위한 passwordEncoder -->
  <bean id="bcryptPasswordEncoder" class="org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder"></bean>

  <!-- DB연동은 data-source만 지정해주면 된다, 테이블이름은 정확히.   users 랑 authorities -->
  <security:authentication-manager alias="authenticationManager">
    <security:authentication-provider>
      <security:password-encoder hash="bcrypt"/>
      <security:jdbc-user-service data-source-ref="dataSource"
                  users-by-username-query="SELECT username, password, enabled FROM glorymember WHERE username=?"
                  authorities-by-username-query="SELECT username, authority FROM glorymember WHERE username=?" />
    </security:authentication-provider>
  </security:authentication-manager>

</beans>
```

login.jsp

```
<form action="/login" method="post">
  <input name="username" type="text" class="form-control" placeholder="아이디">

  <input name="password" type="password" class="form-control" placeholder="비밀번호">

  <button type="submit" class="btn btn-primary btn-block">로그인</button>
  <input type="hidden" name="${_csrf.parameterName}" value="${_csrf.token}" />
</form>
```



Board

Repository

```
//C
void setNewBoard(Board board);//게시물 작성

void replynewBoard(Map map);//댓글 작성

//R
List<Board> getAllBoardList();//게시물 전체 목록

Board getBoardById(String bid);//게시물 하나

List<Board> getReplyById(String bid);//댓글 전체 목록

//U
void editStatus(Map<String, Object> status);//게시물 상태 수정

void editBoard(Board board);//게시글 수정

//D
void removeBoard(String bid);//게시글 삭제

void removeReply(String bid);//댓글 삭제
--------------------------------------------------------------------------------
@Autowired
SqlSessionTemplate sqlSessionTemplate;

public void setNewBoard(Board board) {
  this.sqlSessionTemplate.insert("board.insert", board);
}

public List<Board> getAllBoardList() {
  return this.sqlSessionTemplate.selectList("board.select_list");
}
```

```
public Board getBoardById(String bid) {
  return this.sqlSessionTemplate.selectOne("board.select_detail", bid);
}

public void replynewBoard(Map map) {
  this.sqlSessionTemplate.insert("board.insert_reply", map);
}

public List<Board> getReplyById(String bid) {
  return this.sqlSessionTemplate.selectList("board.select_reply", bid);
}

@Override
public void removeBoard(String bid) {
  this.sqlSessionTemplate.delete("board.delete_board", bid);
}

@Override
public void removeReply(String bid) {
  this.sqlSessionTemplate.delete("board.delete_reply", bid);
}

@Override
public void editStatus(Map<String, Object> status) {
  this.sqlSessionTemplate.update("board.update_status", status);
}

@Override
public void editBoard(Board board) {
  this.sqlSessionTemplate.update("board.update_board", board);
}
```

Service

```
void setNewBoard(Board board);

void replynewBoard(Map map);

List<Board> getAllBoardList();

Board getBoardById(String bid);

List<Board> getReplyById(String bid);

void editStatus(Map<String, Object> status);

void editBoard(Board board);

void removeBoard(String bid);

void removeReply(String bid);
--------------------------------------------------------------------------------
@Autowired
BoardRepository boardRepository;

public void setNewBoard(Board board) {
  boardRepository.setNewBoard(board);
}

public List<Board> getAllBoardList() {
  return boardRepository.getAllBoardList();
}

public Board getBoardById(String bid) {
  return boardRepository.getBoardById(bid);
}

public void replynewBoard(Map map) {
  boardRepository.replynewBoard(map);
}

public List<Board> getReplyById(String bid) {
  return boardRepository.getReplyById(bid);
}

@Override
public void removeBoard(String bid) {
```

```
  boardRepository.removeBoard(bid);
}

@Override
public void removeReply(String bid) {
  boardRepository.removeReply(bid);
}

@Override
public void editStatus(Map<String, Object> status) {
  boardRepository.editStatus(status);
}

@Override
public void editBoard(Board board) {
  boardRepository.editBoard(board);
}
```

Controller

```
package com.aglory.board;

import java.util.List;
import java.util.Map;

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.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
@RequestMapping("/board")
public class BoardController {

  @Autowired
  private BoardService boardService;

//  @Autowired
//  private MailService mailService;
//
//  @Autowired
//  private MemberService memberService;

  @GetMapping("/addboard")
  public String requestAddBoardForm(@ModelAttribute("NewBoard") Board board) {
    return "board/addboard";
  }

  @PostMapping("/addboard")
  public String submitAddBoardForm(@ModelAttribute("NewBoard") Board board) {

    boardService.setNewBoard(board);

//    User user = userService.existUsername(board.getBwriter());
//
//    String to = user.getUemail();
//    String subject = board.getBwriter() + " 님이 게시판에 글을 등록하셨습니다.";
//    String body = board.getBcontent();
//
//    mailService.sendMail(to, subject, body);

    return "redirect:/board/list";
  }

  @GetMapping("/list")
  public String BoardList(Model model) {
      List<Board> list = boardService.getAllBoardList();
      model.addAttribute("boardList", list);

      return "board/list";
  }

  @PostMapping("/list")
```

```
  public void editStatus(@RequestParam Map<String, Object> status) {// Map 여러개 바뀜
    boardService.editStatus(status);
  }

  @GetMapping("/detail")
  public String requestBoardById(@RequestParam("bid") String bid, Model model) {
    // 주 게시물
    Board boardById = boardService.getBoardById(bid);
    model.addAttribute("board", boardById);

    // 답변
    List<Board> list = boardService.getReplyById(bid);
    int cnt = list.size();
    model.addAttribute("replyList",list);
    model.addAttribute("cnt", cnt);

    return "board/detail";
  }

  @ResponseBody
  @RequestMapping("/replynew")
  public void replynew(@RequestParam Map<String, Object> map) {

    String rpid = (String)map.get("rpwriter");
    String bpid = (String)map.get("bpwriter");

    boardService.replynewBoard(map);

//    User user = userService.existUsername(bpid);
//
//    String to = user.getUemail();
//    String subject = bpid + " 님의 글에 댓글이 달렸습니다.";
//    String body = (String)map.get("bcontent");
//
//    mailService.sendMail(to, subject, body);

  }

  @ResponseBody
  @RequestMapping("/removeboard")
  public void removeboard(@RequestParam("bid") String bid) {// String 하나 바뀜
    boardService.removeBoard(bid);
  }

  @ResponseBody
  @RequestMapping("/removereply")
  public void removeReply(@RequestParam("bid") String bid) {// String 하나 바뀜
    boardService.removeReply(bid);
  }

  @GetMapping("/edit")
  public String requestEditboard(@RequestParam("bid") String bid, Model model, @ModelAttribute("EditBoard") Board board) {
    Board boardById = boardService.getBoardById(bid);
    model.addAttribute("board", boardById);

    return "board/editboard";
  }

  @PostMapping("/edit")
  public String submitEditboard(@ModelAttribute("EditBoard") Board board) {
    boardService.editBoard(board);

    return "redirect:/board/list";
  }
}
```

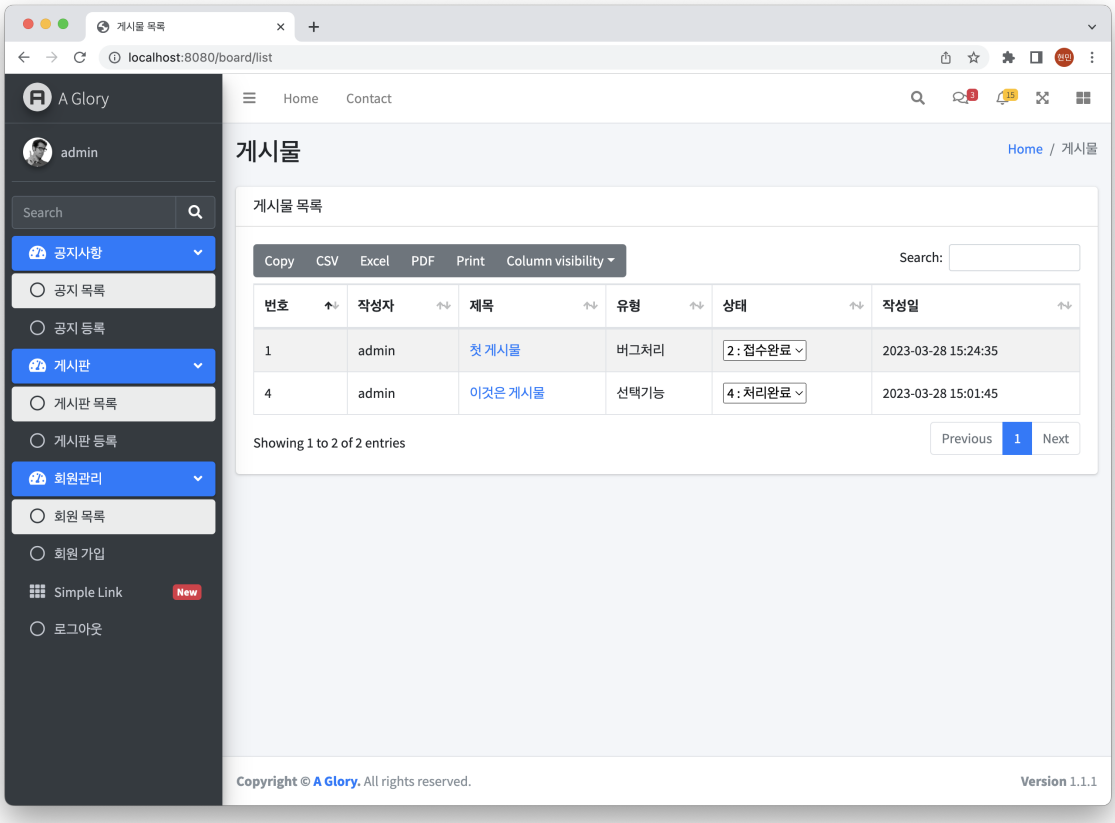
SQL

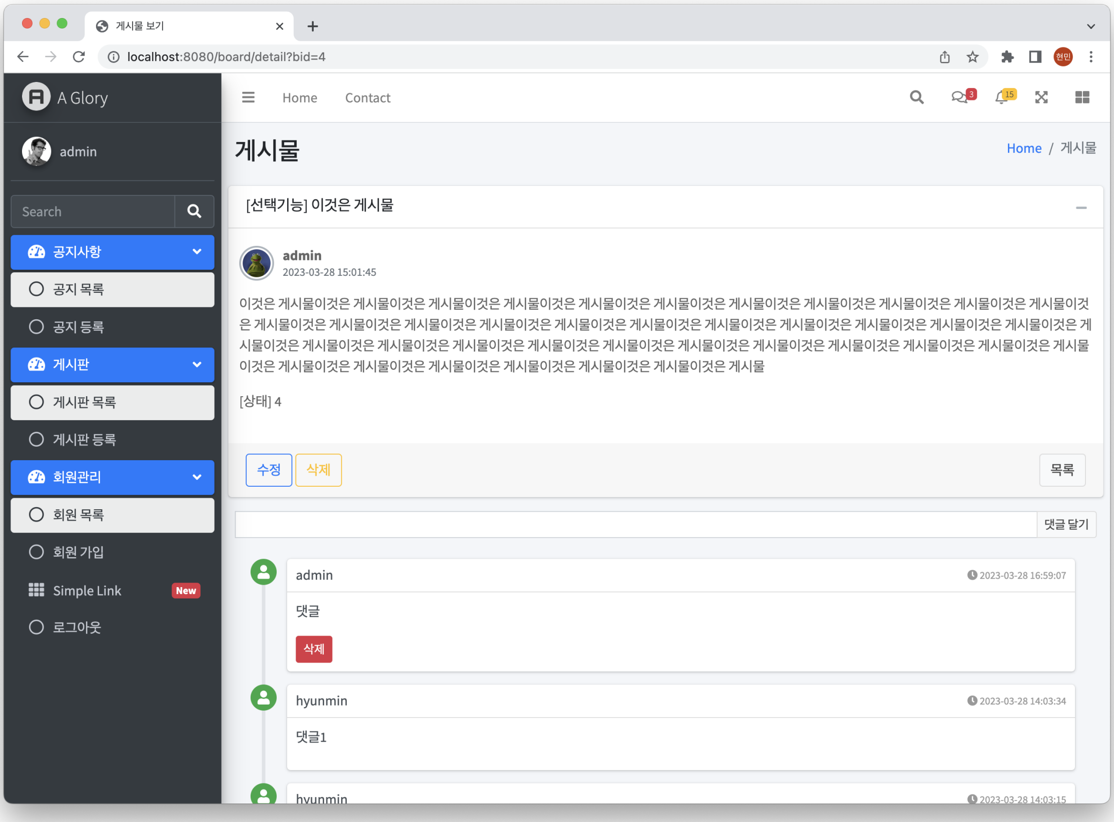
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE  mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="board">

  <insert id="insert"
      parameterType="com.aglory.board.Board"
      useGeneratedKeys="true"
      keyProperty="bid"><!-- 전부 -->
    <![CDATA[
      INSERT INTO gloryboard
```

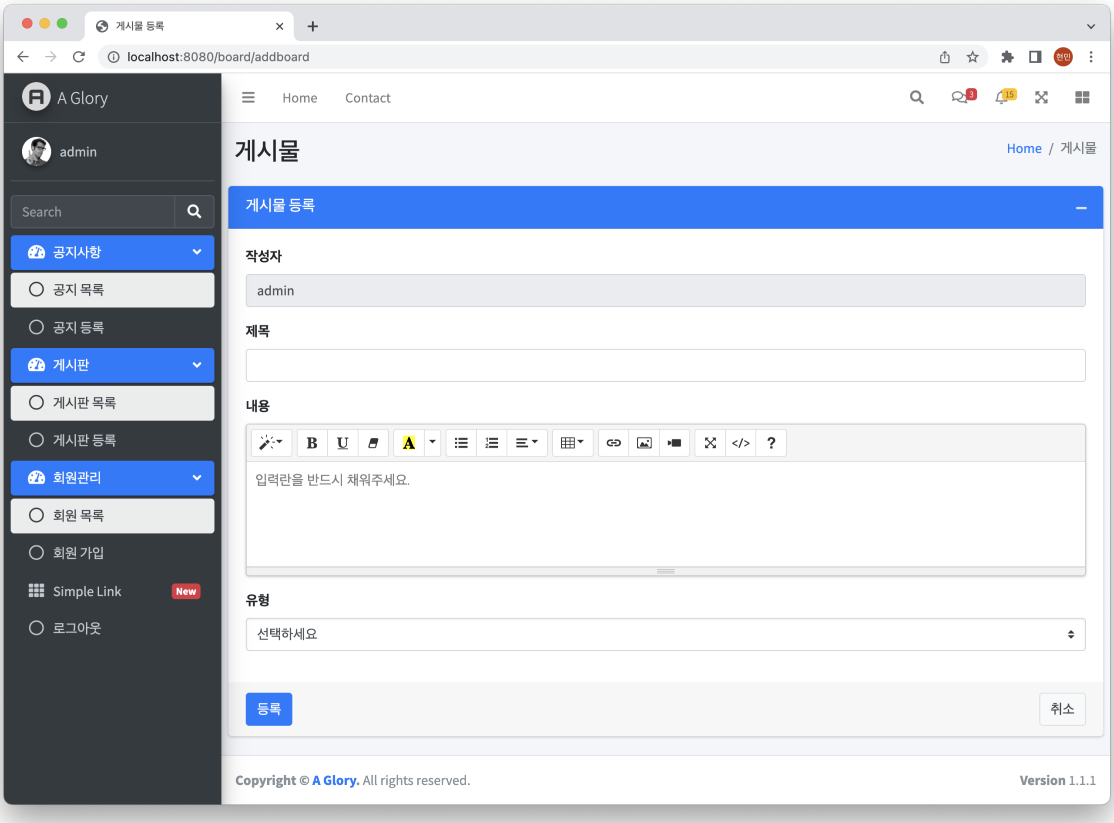
```
    (btitle, bcontent, bcate, bwriter)
    VALUES
    (#{btitle}, #{bcontent}, #{bcate}, #{bwriter})
  ]]>
</insert>

<insert id="insert_reply"
    parameterType="hashMap"
    useGeneratedKeys="true"
    keyProperty="bid">
  <![CDATA[
    INSERT INTO gloryboard
    (bpid, bcontent, bwriter)
    VALUES
    (#{bid}, #{bcontent}, #{bwriter})
  ]]>
</insert>

<select id="select_list"
    resultType="com.aglory.board.Board">
  <![CDATA[
    SELECT *
    FROM gloryboard
    WHERE bpid = '0'
    ORDER BY bid DESC
  ]]>
</select>

<select id="select_detail"
    parameterType="String"
    resultType="com.aglory.board.Board">
  <![CDATA[
    SELECT *
    FROM gloryboard
    WHERE bid = #{bid}
  ]]>
</select>

<select id="select_reply"
    parameterType="String"
    resultType="com.aglory.board.Board">
  <![CDATA[
    SELECT *
    FROM gloryboard
    WHERE bpid = #{bid}
    ORDER BY bdate DESC
  ]]>
</select>

<delete id="delete_board"
    parameterType="String">
  <![CDATA[
    DELETE FROM gloryboard
    WHERE bid = #{bid} OR bpid = #{bid}
  ]]>
</delete>

<delete id="delete_reply"
    parameterType="String">
  <![CDATA[
    DELETE FROM gloryboard
    WHERE bid = #{bid}
  ]]>
</delete>

<update id="update_status"
    parameterType="hashMap">
  <![CDATA[
    UPDATE gloryboard
    SET bstatus = #{bstatus}
    WHERE bid = #{bid}
  ]]>
</update>

<update id="update_board"
    parameterType="com.aglory.board.Board">
  <![CDATA[
    UPDATE gloryboard
    SET
    btitle = #{btitle},
    bcontent = #{bcontent},
    bcate = #{bcate}
```

```
      WHERE bid = #{bid}
    ]]>
  </update>

</mapper>
```

View




게시물 보기
localhost:8080/board/detail?bid=4
A Glory
admin
Search
공지사항
공지 목록
공지 등록
게시판
게시판 목록
게시판 등록
회원관리
회원 목록
회원 가입
Simple Link
New
로그아웃
Home
Contact
게시물
Home / 게시물
[선택기능] 이것은 게시물
admin
2023-03-28 15:01:45
[상태] 4
수정
삭제
목록
댓글 달기
admin
2023-03-28 16:59:07
댓글
삭제
hyunmin
2023-03-28 14:03:34
댓글1
hyunmin


게시물 등록
localhost:8080/board/addboard
A Glory
admin
Search
공지사항
공지 목록
공지 등록
게시판
게시판 목록
게시판 등록
회원관리
회원 목록
회원 가입
Simple Link
New
로그아웃
Home
Contact
게시물
Home / 게시물
게시물 등록
작성자
admin
제목
내용
입력란을 반드시 채워주세요.
유형
선택하세요
등록
취소
Copyright © A Glory. All rights reserved.
Version 1.1.1

Notice

Repository

```
//C
void setNewNotice(Notice notice);//공지 작성

//R
List<Notice> getAllNoticeList();//공지 전체 목록

Notice getNoticeById(String nid);//공지 하나

//U
void editNotice(Notice notice);//공지 수정

void editCategory(Map<String, Object> category);//공지 category 수정

//D
void removeNotice(String nid);//공지 삭제
--------------------------------------------------------------------------------
@Autowired
SqlSessionTemplate sqlSessionTemplate;

@Override
public void setNewNotice(Notice notice) {
  this.sqlSessionTemplate.insert("notice.insert", notice);
}

@Override
public List<Notice> getAllNoticeList() {
  return this.sqlSessionTemplate.selectList("notice.select_list");
}

@Override
public Notice getNoticeById(String nid) {
  return this.sqlSessionTemplate.selectOne("notice.select_detail", nid);
}

@Override
public void editNotice(Notice notice) {
  this.sqlSessionTemplate.update("notice.update_notice", notice);
}

@Override
public void editCategory(Map<String, Object> category) {
  this.sqlSessionTemplate.update("notice.update_category", category);
}

@Override
public void removeNotice(String nid) {
  this.sqlSessionTemplate.delete("notice.delete", nid);
}
```

Service

```
void setNewNotice(Notice notice);

List<Notice> getAllNoticeList();

Notice getNoticeById(String nid);

void editNotice(Notice notice);

void editCategory(Map<String, Object> category);

void removeNotice(String nid);
--------------------------------------------------------------------------------
@Autowired
NoticeRepository noticeRepository;

@Override
public void setNewNotice(Notice notice) {
  noticeRepository.setNewNotice(notice);
}
```

```
@Override
public List<Notice> getAllNoticeList() {
  return noticeRepository.getAllNoticeList();
}

@Override
public Notice getNoticeById(String nid) {
  return noticeRepository.getNoticeById(nid);
}

@Override
public void editNotice(Notice notice) {
  noticeRepository.editNotice(notice);
}

@Override
public void editCategory(Map<String, Object> category) {
  noticeRepository.editCategory(category);
}

@Override
public void removeNotice(String nid) {
  noticeRepository.removeNotice(nid);
}
```

Controller

```
package com.aglory.notice;

import java.util.List;
import java.util.Map;

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.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
@RequestMapping("/notice")
public class NoticeController {

  @Autowired
  private NoticeService noticeService;

  @GetMapping("/addnotice")
  public String requestAddNoticeForm(@ModelAttribute("NewNotice") Notice notice) {
    return "notice/addnotice";
  }

  @PostMapping("/addnotice")
  public String submitAddNoticeForm(@ModelAttribute("NewNotice") Notice notice) {
    noticeService.setNewNotice(notice);

    return "redirect:/notice/list";
  }

  @GetMapping("/list")
  public String NoticeList(Model model) {
      List<Notice> list = noticeService.getAllNoticeList();
      model.addAttribute("noticeList", list);

      return "notice/list";
  }

  @PostMapping("/list")
  public void editCategory(@RequestParam Map<String, Object> category) {// Map 여러개 바뀜
    noticeService.editCategory(category);
  }

  @GetMapping("/detail")
  public String requestNoticeById(@RequestParam("nid") String nid, Model model) {
    Notice noticeById = noticeService.getNoticeById(nid);
```

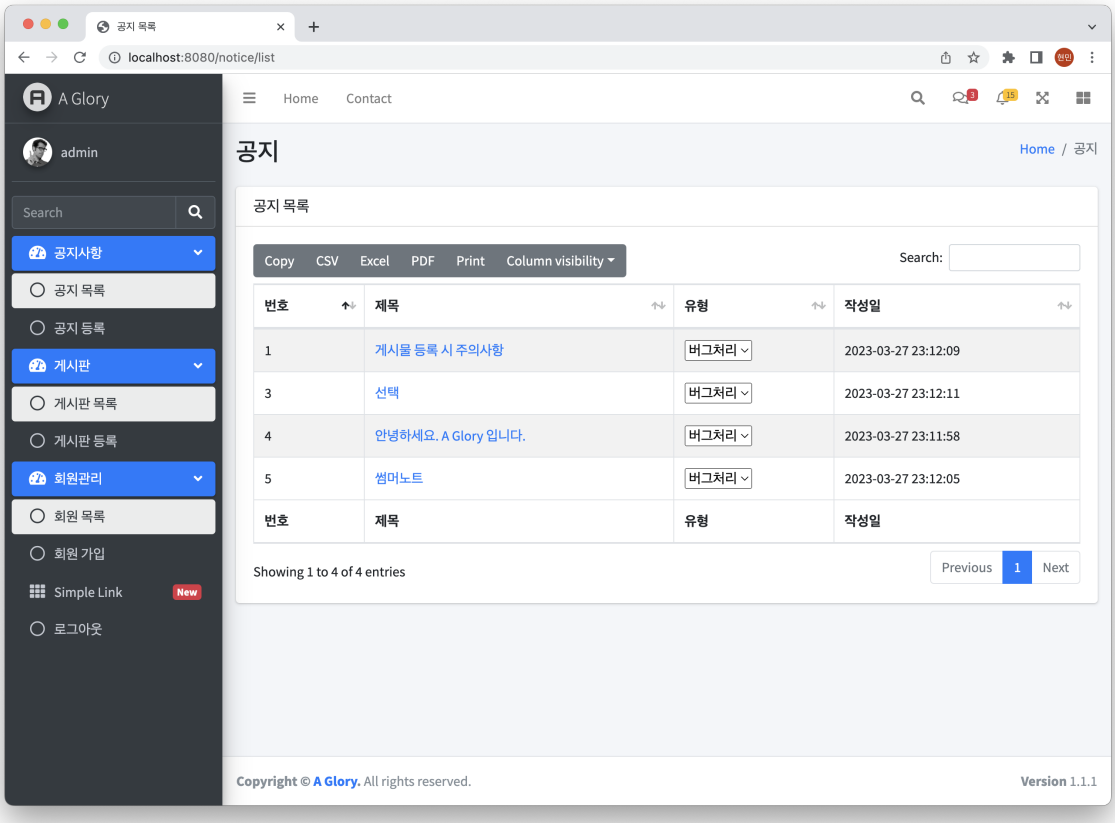
```
    model.addAttribute("notice", noticeById);

    return "notice/detail";
  }

  @ResponseBody
  @RequestMapping("/remove")
  public void removeNotice(@RequestParam("nid") String nid) {// String 하나 바뀜
    noticeService.removeNotice(nid);
  }

  @GetMapping("/edit")
  public String requestEditNotice(@RequestParam("nid") String nid, Model model, @ModelAttribute("EditNotice") Notice notice) {
    Notice noticeById = noticeService.getNoticeById(nid);
    model.addAttribute("notice", noticeById);

    return "notice/editnotice";
  }

  @PostMapping("/edit")
  public String submitEditNotice(@ModelAttribute("EditNotice") Notice notice) {
    noticeService.editNotice(notice);

    return "redirect:/notice/list";
  }
}
```

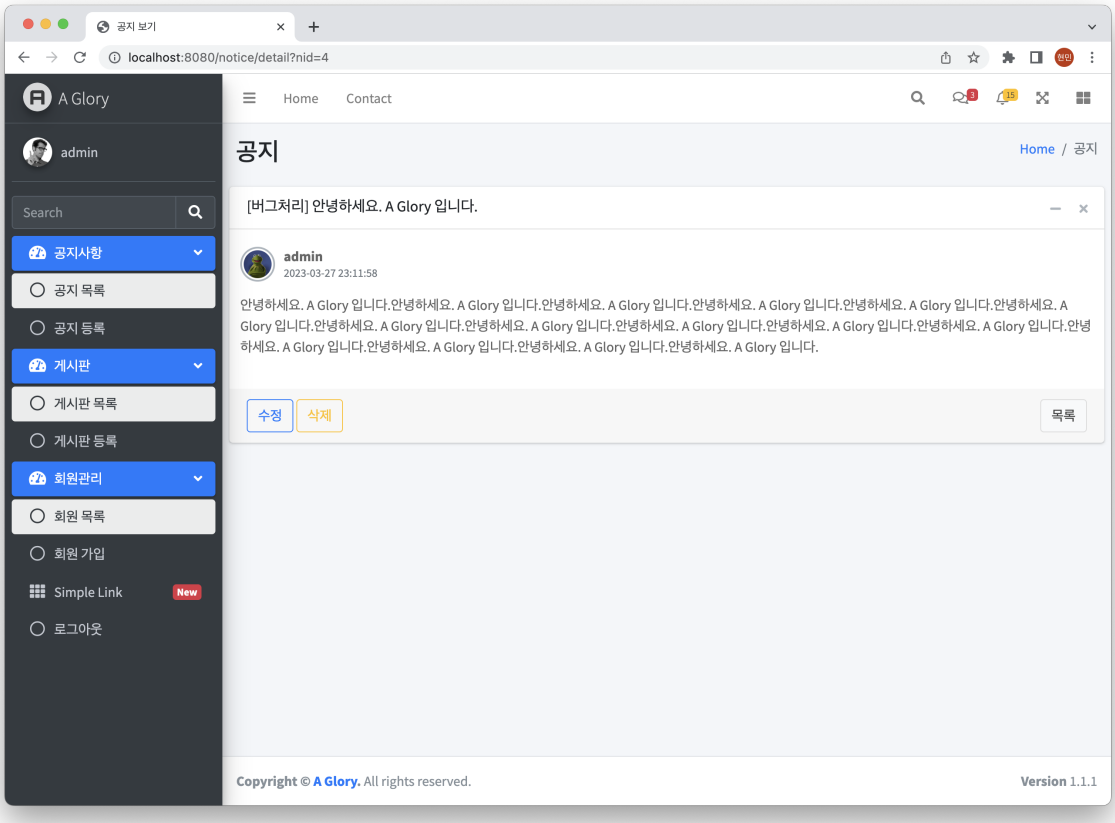
SQL

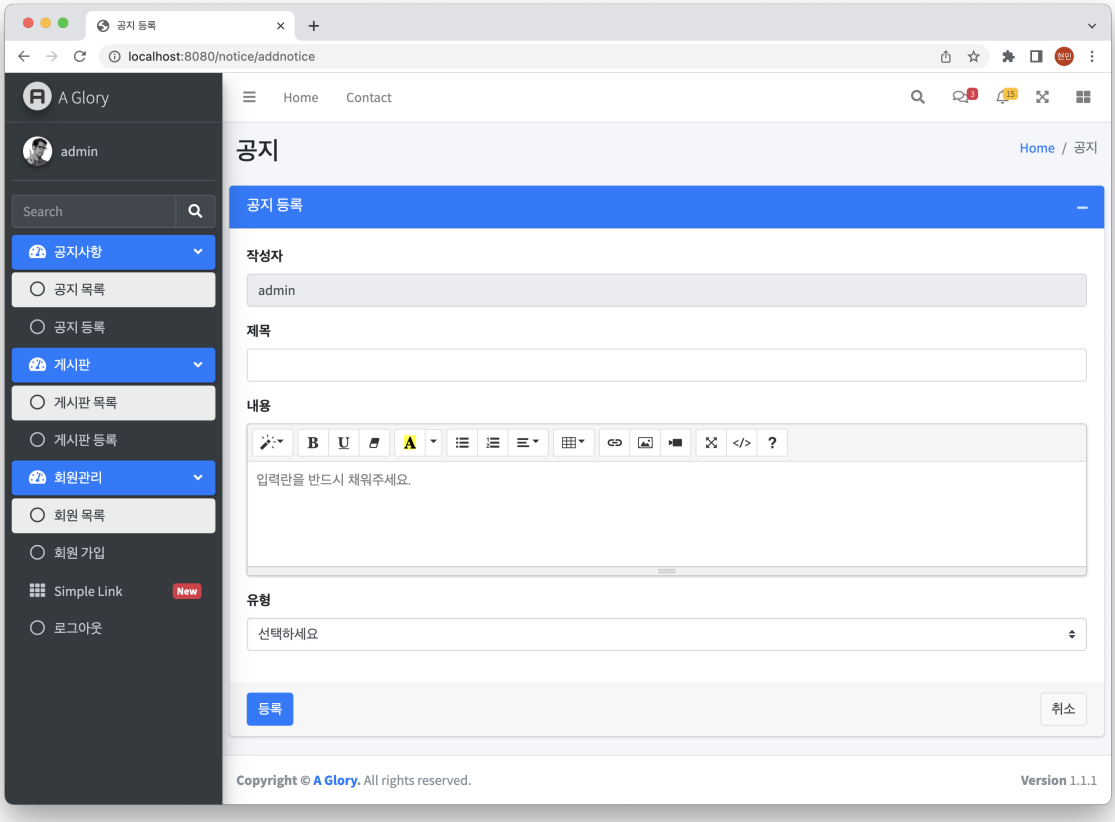
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE  mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="notice">
  <insert id="insert"
      parameterType="com.aglory.notice.Notice"
      useGeneratedKeys="true"
      keyProperty="nid"><!-- 전부 -->
    <![CDATA[
      INSERT INTO glorynotice
      (ntitle, ncontent, ncate, nwriter)
      VALUES
      (#{ntitle}, #{ncontent}, #{ncate}, #{nwriter})
    ]]>
  </insert>

  <select id="select_list"
      resultType="com.aglory.notice.Notice">
    <![CDATA[
      SELECT *
      FROM glorynotice
      ORDER BY nid DESC
    ]]>
  </select>

  <select id="select_detail"
      parameterType="String"
      resultType="com.aglory.notice.Notice">
    <![CDATA[
      SELECT *
      FROM glorynotice
      WHERE nid = #{nid}
    ]]>
  </select>

  <update id="update_notice"
      parameterType="com.aglory.notice.Notice">
    <![CDATA[
      UPDATE glorynotice
      SET
      ntitle = #{ntitle},
      ncontent = #{ncontent},
      ncate = #{ncate}
      WHERE nid = #{nid}
    ]]>
  </update>

  <update id="update_category"
      parameterType="hashMap">
    <![CDATA[
```

```
      UPDATE glorynotice
      SET ncate = #{ncate}
      WHERE nid = #{nid}
    ]]>
  </update>

  <delete id="delete"
      parameterType="String">
    <![CDATA[
      DELETE FROM glorynotice
      WHERE nid = #{nid}
    ]]>
  </delete>
</mapper>
```

View




공지 보기
localhost:8080/notice/detail?nid=4
A Glory
admin
Search
공지사항
공지 목록
공지 등록
게시판
게시판 목록
게시판 등록
회원관리
회원 목록
회원 가입
Simple Link
New
로그아웃
Home
Contact
공지
Home / 공지
[버그처리] 안녕하세요. A Glory 입니다.
admin
2023-03-27 23:11:58
안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.안녕하세요. A Glory 입니다.
수정
삭제
목록
Copyright © A Glory. All rights reserved.
Version 1.1.1


공지 등록
localhost:8080/notice/addnotice
A Glory
admin
Search
공지사항
공지 목록
공지 등록
게시판
게시판 목록
게시판 등록
회원관리
회원 목록
회원 가입
Simple Link
New
로그아웃
Home
Contact
공지
Home / 공지
공지 등록
작성자
admin
제목
내용
입력란을 반드시 채워주세요.
유형
선택하세요
등록
취소
Copyright © A Glory. All rights reserved.
Version 1.1.1

Member

Repository

```
//C
void setNewMember(Member member);//회원 가입

//R
List<Member> getAllMemberList();//회원 전체 목록

Member getMemberById(String mid);//회원 하나

//U
void editAuth(Map<String, Object> auth);//회원 권한 수정

void editEnabled(Map<String, Object> enabled);//회원 권한 수정

void editMember(Member member);//회원 정보 수정

//D
void removeMember(String mid);//회원 삭제
--------------------------------------------------------------------------------
@Autowired
SqlSessionTemplate sqlSessionTemplate;

@Override
public void setNewMember(Member member) {
  this.sqlSessionTemplate.insert("member.insert", member);
}

@Override
public List<Member> getAllMemberList() {
  return this.sqlSessionTemplate.selectList("member.select_list");
}

@Override
public Member getMemberById(String mid) {
  return this.sqlSessionTemplate.selectOne("member.select_detail", mid);
}

@Override
public void editAuth(Map<String, Object> auth) {
  this.sqlSessionTemplate.update("member.update_auth", auth);
}

@Override
public void removeMember(String mid) {
  this.sqlSessionTemplate.delete("member.delete", mid);
}

@Override
public void editMember(Member member) {
  this.sqlSessionTemplate.update("member.update_member", member);
}

@Override
public void editEnabled(Map<String, Object> enabled) {
  this.sqlSessionTemplate.update("member.update_enabled", enabled);
}
```

Service

```
void setNewMember(Member member);

List<Member> getAllMemberList();

Member getMemberById(String mid);

void editAuth(Map<String, Object> auth);

void editEnabled(Map<String, Object> enabled);

void editMember(Member member);
```

```
void removeMember(String mid);
--------------------------------------------------------------------------------
@Autowired
MemberRepository memberRepository;

@Override
public void setNewMember(Member member) {
  memberRepository.setNewMember(member);
}

@Override
public List<Member> getAllMemberList() {
  return memberRepository.getAllMemberList();
}

@Override
public Member getMemberById(String mid) {
  return memberRepository.getMemberById(mid);
}

@Override
public void editAuth(Map<String, Object> auth) {
  memberRepository.editAuth(auth);
}

@Override
public void removeMember(String mid) {
  memberRepository.removeMember(mid);
}

@Override
public void editMember(Member member) {
  memberRepository.editMember(member);
}

@Override
public void editEnabled(Map<String, Object> enabled) {
  memberRepository.editEnabled(enabled);
}
```

Controller

```
package com.aglory.member;

import java.util.List;
import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;

import com.aglory.notice.Notice;

@Controller
@RequestMapping("/member")
public class MemberController {

  @Autowired
  MemberService memberService;

  @Autowired
  BCryptPasswordEncoder bcryptPasswordEncoder;

  @GetMapping("/addmember")
  public String requestAddMemberForm(@ModelAttribute("NewMember") Member member) {
    return "member/addmember";
  }

  @PostMapping("/addmember")
  public String submitAddMemberForm(@ModelAttribute("NewMember") Member member) {
    //스프링은 반드시 password를 암호화 하여 저장해야만 로그인을 할수 있는게 기본이다.
```

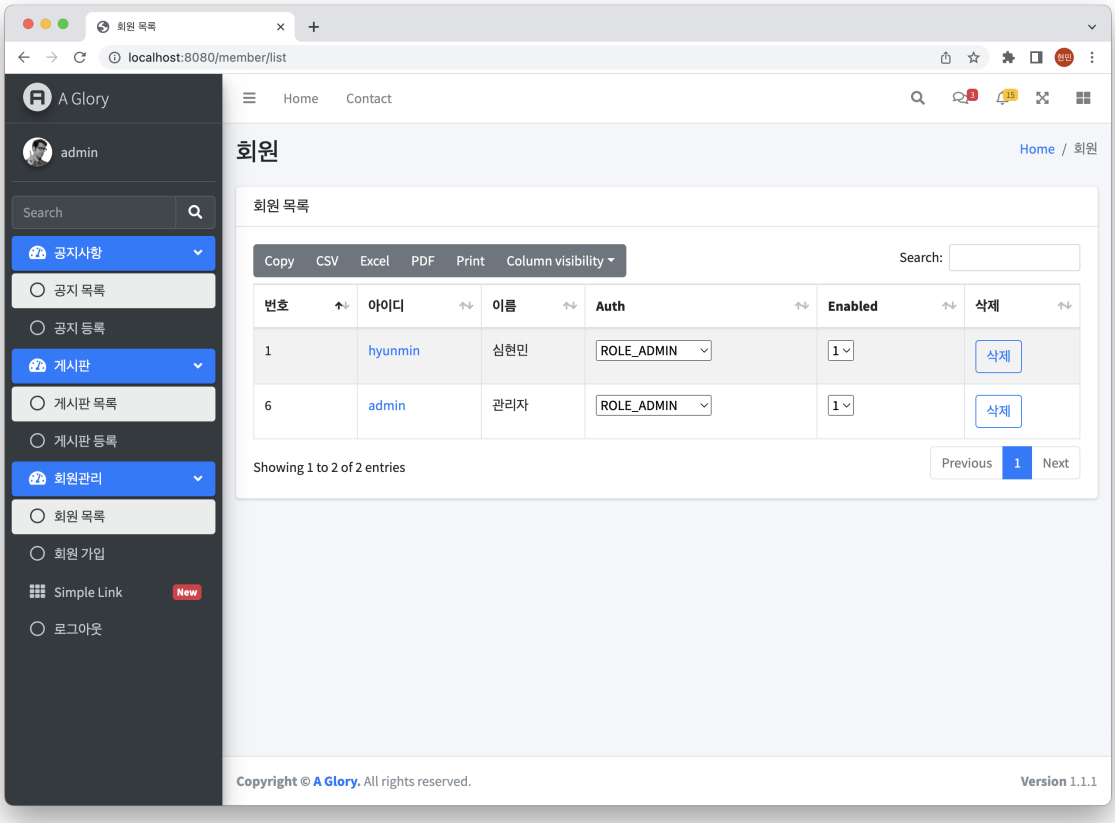
```
    String encodedPassword = bcryptPasswordEncoder.encode(member.getPassword());
    member.setPassword(encodedPassword);

    memberService.setNewMember(member);

    return "redirect:/member/list";
  }

  @GetMapping("/list")
  public String memberList(Model model) {
      List<Member> list = memberService.getAllMemberList();
      model.addAttribute("memberList", list);

      return "member/list";
  }

  @PostMapping("/updateAuth")
  public void editAuth(@RequestParam Map<String, Object> auth) {// Map 여러개 바꿈
    memberService.editAuth(auth);
  }

  @PostMapping("/updateEnabled")
  public void editEnabled(@RequestParam Map<String, Object> enabled) {// Map 여러개 바꿈
    memberService.editEnabled(enabled);
  }

  @ResponseBody
  @RequestMapping("/remove")
  public void removeMember(@RequestParam("mid") String mid) {// String 하나 바꿈
    memberService.removeMember(mid);
  }

  @GetMapping("/detail")
  public String requestMemberById(@RequestParam("mid") String mid, Model model) {
    Member memberById = memberService.getMemberById(mid);
    model.addAttribute("member", memberById);

    return "member/detail";
  }

  @GetMapping("/edit")
  public String requestEditMember(@RequestParam("mid") String mid, Model model, @ModelAttribute("EditMember") Member member) {
    Member memberById = memberService.getMemberById(mid);
    model.addAttribute("member", memberById);

    return "member/editmember";
  }

  @PostMapping("/edit")
  public String submitEditNotice(@ModelAttribute("EditMember") Member member, @RequestParam("mid") String mid) {
    memberService.editMember(member);

    return "redirect:/member/detail?mid="+mid;
  }
}
```

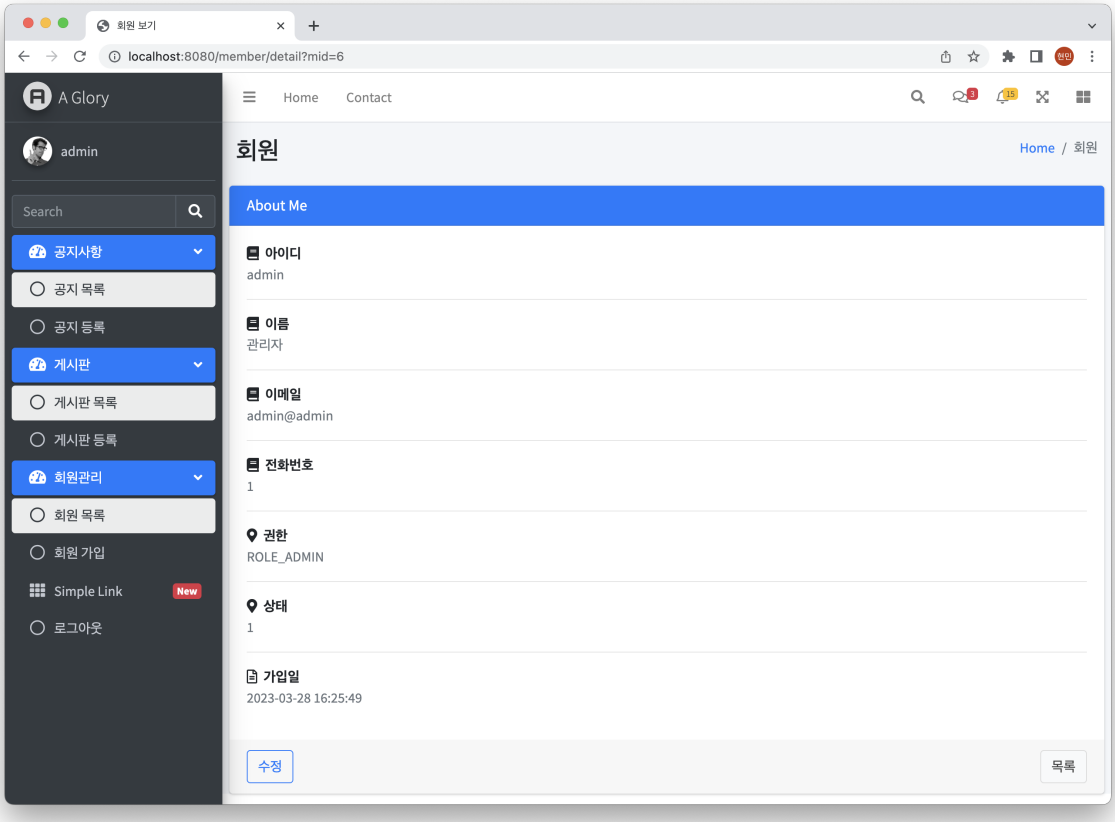
SQL

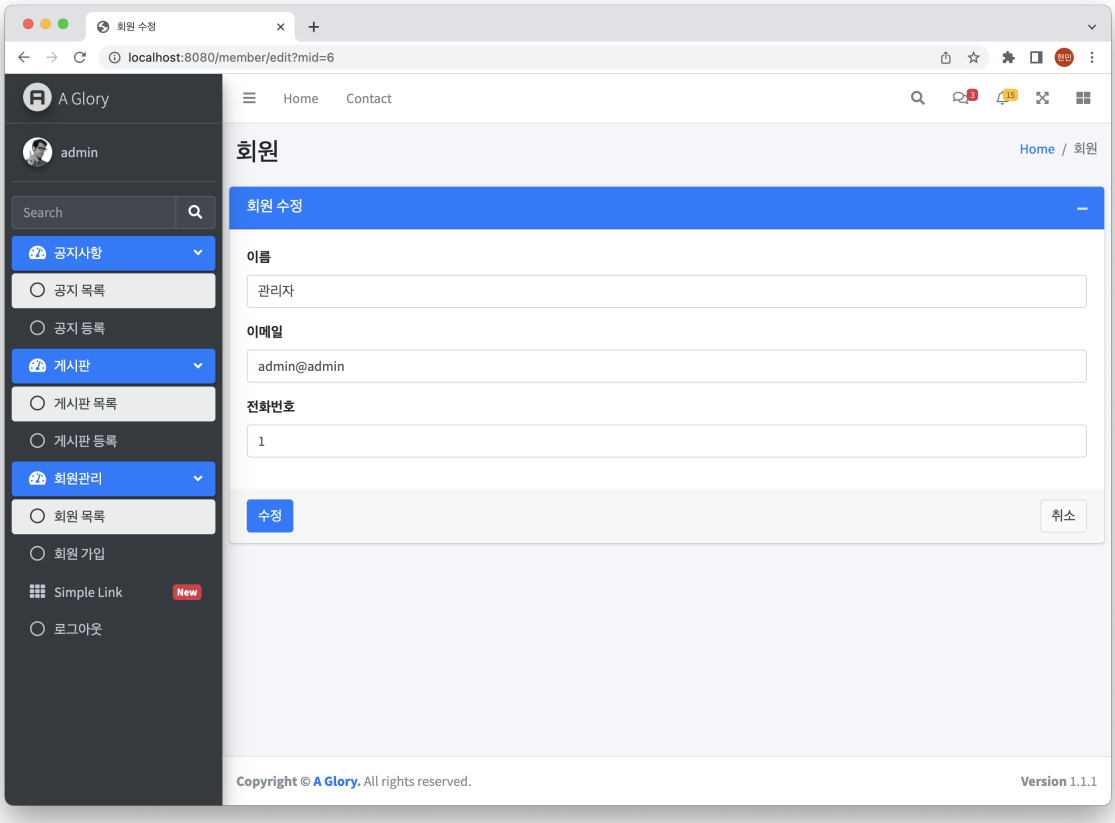
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE  mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="member">
  <insert id="insert"
      parameterType="com.aglory.member.Member"
      useGeneratedKeys="true"
      keyProperty="username"><!-- 전부 -->
    <![CDATA[
      INSERT INTO glorymember
      (username, password, mname, memail, mtel)
      VALUES
      (#{username}, #{password}, #{mname}, #{memail}, #{mtel})
    ]]>
  </insert>

  <select id="select_list"
      resultType="com.aglory.member.Member">
    <![CDATA[
      SELECT *
      FROM glorymember
```

```
      ORDER BY mid DESC
    ]]>
  </select>

  <select id="select_detail"
      parameterType="String"
      resultType="com.aglory.member.Member">
    <![CDATA[
      SELECT *
      FROM glorymember
      WHERE mid = #{mid}
    ]]>
  </select>

  <update id="update_member"
      parameterType="com.aglory.member.Member">
    <![CDATA[
      UPDATE glorymember
      SET
      mname = #{mname},
      memail = #{memail},
      mtel = #{mtel}
      WHERE username = #{username}
    ]]>
  </update>

  <update id="update_auth"
      parameterType="hashMap">
    <![CDATA[
      UPDATE glorymember
      SET authority = #{authority}
      WHERE username = #{username}
    ]]>
  </update>

  <update id="update_enabled"
      parameterType="hashMap">
    <![CDATA[
      UPDATE glorymember
      SET enabled = #{enabled}
      WHERE username = #{username}
    ]]>
  </update>

  <delete id="delete"
      parameterType="String">
    <![CDATA[
      DELETE FROM glorymember
      WHERE mid = #{mid}
    ]]>
  </delete>
</mapper>
```

View


회원 목록
localhost:8080/member/list
A Glory
admin
Search
공지사항
공지 목록
공지 등록
게시판
게시판 목록
게시판 등록
회원관리
회원 목록
회원 가입
Simple Link
New
로그아웃
Home
Contact
회원
Home / 회원
회원 목록
Copy
CSV
Excel
PDF
Print
Column visibility
Search:
번호
아이디
이름
Auth
Enabled
삭제
1
hyunmin
심현민
ROLE_ADMIN
1
삭제
6
admin
관리자
ROLE_ADMIN
1
삭제
Showing 1 to 2 of 2 entries
Previous
1
Next
Copyright © A Glory. All rights reserved.
Version 1.1.1


회원 보기
localhost:8080/member/detail?mid=6
A Glory
admin
Search
공지사항
공지 목록
공지 등록
게시판
게시판 목록
게시판 등록
회원관리
회원 목록
회원 가입
Simple Link
New
로그아웃
Home
Contact
회원
Home / 회원
About Me
아이디
admin
이름
관리자
이메일
admin@admin
전화번호
1
권한
ROLE_ADMIN
상태
1
가입일
2023-03-28 16:25:49
수정
목록


회원 수정
localhost:8080/member/edit?mid=6
A Glory
admin
Search
공지사항
공지 목록
공지 등록
게시판
게시판 목록
게시판 등록
회원관리
회원 목록
회원 가입
Simple Link
New
로그아웃
Home
Contact
회원
Home / 회원
회원 수정
이름
관리자
이메일
admin@admin
전화번호
1
수정
취소
Copyright © A Glory. All rights reserved.
Version 1.1.1