# 회원제 게시판 만들기

## 기능(Service)

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 기능 | 클래스 이름 | 커맨드 URI |
| 회원 관리 | 로그인 | LoginService | login.do |
| 로그아웃 | LogoutService | logout.do |
| 회원가입 | JoinService | join.do |
| 회원탈퇴 | LeaveService | leave.do |
| 게시판 관리 | 목록 보기 | ListService | list.bo |
| 내용 보기 | ContentSerivce | content.bo |
| 글 작성 | WriteService | write.bo |
| 글 수정 | UpdateService | update.bo |
| 글 삭제 | DeleteService | delete.bo |
| 파일 다운로드 | - | downfile.bo |

## 화면(view, JSP)

loginFrm.jsp : 처음 화면 이자 로그인 화면(처음 접속 시 또는 세션에 id 정보가 없을 경우 이 페이지를 보여줌).

joinFrm.jsp : 회원 가입 화면. 로그인 화면에서 ‘회원 가입’ 링크로 연결

boardList.jsp : 게시글 목록 화면.

writeFrm.jsp : 게시글 작성 화면.

contents.jsp : 게시글 내용 보기 화면.

updateFrm.jsp : 게시글 수정 화면.

## 패키지 구성

### com.bean

Member.java : 회원 정보 객체

Board.java : 게시글 객체

### com.controller

MemberController.java : 회원 관리용 컨트롤러

BoardController.java : 게시판 관리용 컨트롤러

### com.service

ServiceForward.java : 페이지 이동 처리용 클래스

Service.java : 각 서비스 처리용 클래스의 인터페이스

### com.service.member

LoginService.java : 로그인 처리용 기능 클래스

LogoutService.java : 로그아웃 처리용 기능 클래스

JoinService.java : 회원가입 처리용 기능 클래스

LeaveService.java : 회원탈퇴 처리용 기능 클래스

### com.service.board

ListService.java : 게시글 목록 보기 처리용 기능 클래스

ContentService.java : 게시글 내용 보기 처리용 기능 클래스

WriteService.java : 게시글 작성 처리용 기능 클래스

UpdateService.java : 게시글 수정 처리용 기능 클래스

DeleteService.java : 게시글 삭제 처리용 기능 클래스

### com.dao

MemberDao.java : 회원 정보 DB 연동 클래스

BoardDao.java : 게시판 DB 연동 클래스

### com.util

DBConnection.java : DB 접속용 클래스

## DB 정의

### 회원 정보(userInfo)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 속성명 | 컬럼명 | 자료형 | 크기 | 비고 |
| 아이디 | **userId** | varchar | 20 | PK |
| 패스워드 | pass | varchar | 20 | Not null |
| 이름 | userName | char | 10 | Not null |
| 주소 | addr | varchar | 50 | Not null |
| 연락처 | phone | char | 12 | Not null |
| 가입날짜 | joinDate | date |  | default now() |

### 게시글(boardTbl)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 속성명 | 컬럼명 | 자료형 | 크기 | 비고 |
| 글번호 | **bNo** | int |  | PK, AI |
| 작성자 아이디 | ***userId*** | varchar | 20 | FK, Not null |
| 글제목 | title | varchar | 50 | Not null |
| 글내용 | content | varchar | 255 | Not null |
| 작성일시 | bDate | date |  | default now() |
| 첨부파일 | fileName | varchar | 100 | Null |

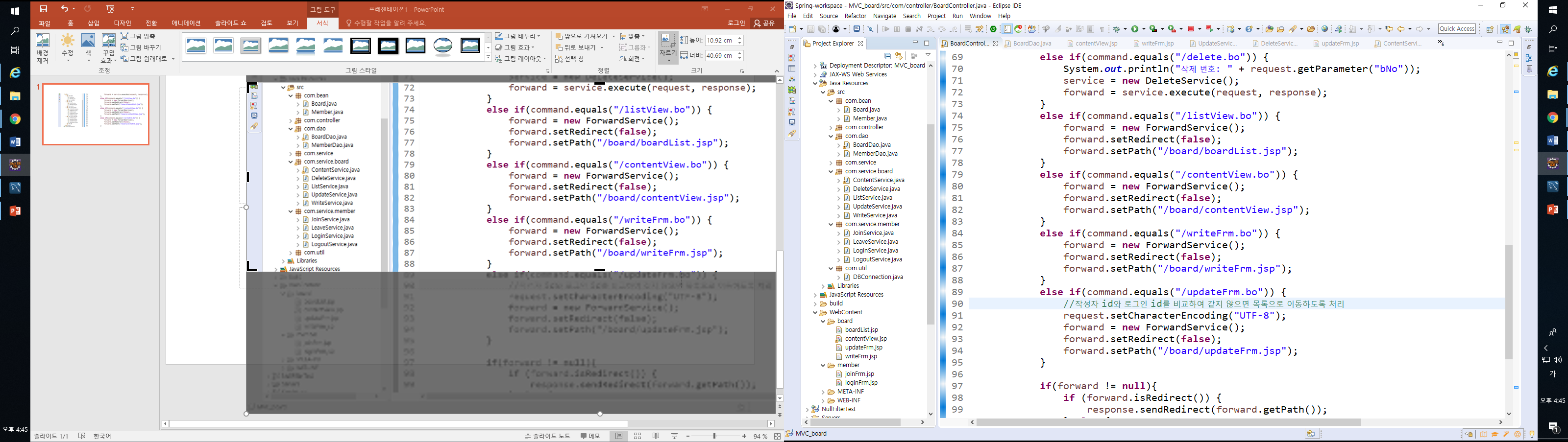
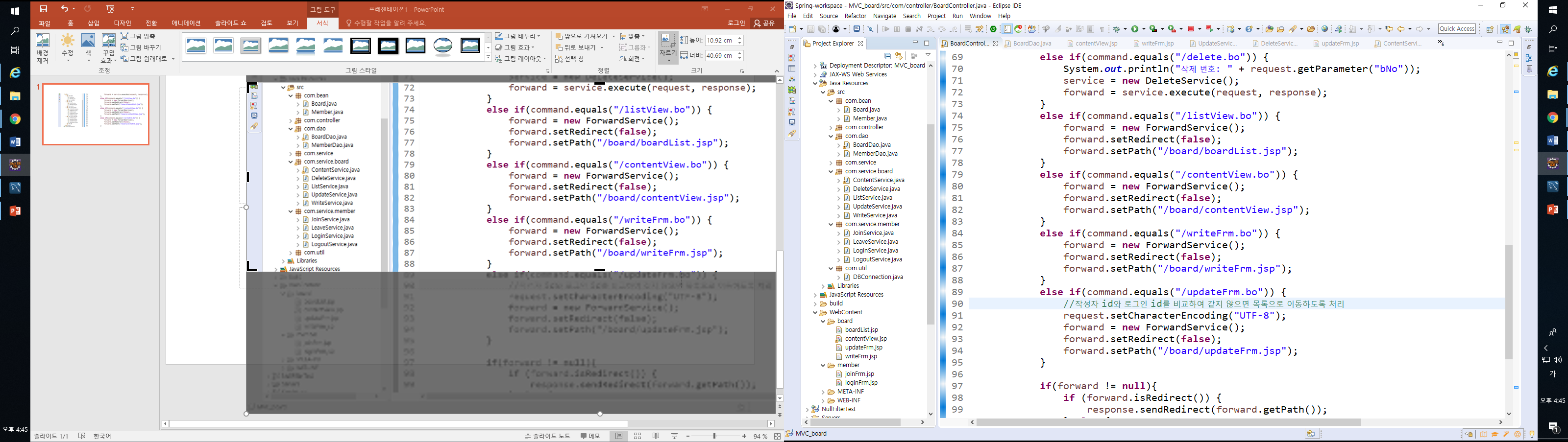
## 쿼리

|  |
| --- |
| -- 스키마 제거  drop database if exists mboarddb;  -- 스키마 생성  create database if not exists mboarddb;  -- 스키마 사용  use mboarddb;  -- 회원 테이블 삭제  drop table if exists userInfo;  -- 회원 테이블 생성  create table if not exists userInfo  (  userId varchar(20) primary key,  pass varchar(20) not null,  userName char(10) not null,  addr varchar(50) not null,  phone char(12) not null,  joinDate datetime default now()  );  -- 테스트 계정 삽입  insert into userInfo values ('test', '1111', '테스터', '인천', '01011111111', default);  select \* from userInfo;  -- 게시판 테이블 삭제  drop table if exists boardTbl;  -- 게시판 테이블 생성  create table if not exists boardTbl  (  bNo int primary key auto\_increment,  userId varchar(20) not null,  title varchar(50) not null,  content varchar(255) not null,  bDate datetime default now(),  fileName varchar(100),  foreign key (userId) references userInfo (userId)  );  -- 테스트글 삽입  insert into boardTbl values (null, 'test', '테스트글', '테스트글입니다.', default, null);  select \* from boardTbl; |

## web.xml 설정

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns=*"http://xmlns.jcp.org/xml/ns/javaee"* xsi:schemaLocation=*"http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd"* version=*"3.1"*>  <display-name>MVC\_board</display-name>    <servlet>  <servlet-name>MemberController</servlet-name>  <servlet-class>com.controller.MemberController</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>MemberController</servlet-name>  <url-pattern>\*.do</url-pattern>  </servlet-mapping>    <servlet>  <servlet-name>BoardController</servlet-name>  <servlet-class>com.controller.BoardController</servlet-class>  </servlet>  <servlet-mapping>  <servlet-name>BoardController</servlet-name>  <url-pattern>\*.bo</url-pattern>  </servlet-mapping>    <resource-ref>  <description>connection</description>  <res-ref-name>jdbc/mysql</res-ref-name>  <res-type>javax.sql.DataSource</res-type>  <res-auth>Container</res-auth>  </resource-ref>  </web-app> |

## 폴더 구조

## 코드 작성 : 회원 관리

### Member.java

|  |
| --- |
| package com.bean;  public class Member {    private String userId;  private String pass;  private String userName;  private String addr;  private String phone;  private String joinDate;    public String getUserId() {  return userId;  }  public void setUserId(String userId) {  this.userId = userId;  }  public String getPass() {  return pass;  }  public void setPass(String pass) {  this.pass = pass;  }  public String getUserName() {  return userName;  }  public void setUserName(String userName) {  this.userName = userName;  }  public String getAddr() {  return addr;  }  public void setAddr(String addr) {  this.addr = addr;  }  public String getPhone() {  return phone;  }  public void setPhone(String phone) {  this.phone = phone;  }  public String getJoinDate() {  return joinDate;  }  public void setJoinDate(String joinDate) {  this.joinDate = joinDate;  }    } |

### loginFrm.jsp

|  |
| --- |
| <%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>  <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>로그인</title>  <c:choose>  <c:when test="${msg == '0'}">  <script>  alert("아이디나 패스워드가 틀렸습니다.");  </script>  <c:remove var="msg" scope="session"/>  </c:when>  <c:when test="${msg == '1'}">  <script>  alert("회원 가입이 완료되었습니다.");  </script>  <c:remove var="msg" scope="session"/>  </c:when>  <c:when test="${msg == '2'}">  <script>  alert("아이디 중복 등의 이유로 회원 가입에 실패하였습니다.");  </script>  <c:remove var="msg" scope="session"/>  </c:when>  <c:when test="${msg == '3'}">  <script>  alert("회원 탈퇴 처리가 완료되었습니다.");  </script>  <c:remove var="msg" scope="session"/>  </c:when>  </c:choose>  </head>  <body>  <h3>로그인</h3>  <form action="./login.do" method="post">  <table>  <tr>  <td width="100">ID</td>  <td width="200"><input type="text" name="id" required="required"></td>  </tr>  <tr>  <td>PASS</td>  <td><input type="password" name="pw" required="required"></td>  </tr>  <tr>  <td colspan="2" align="center">  <input type="submit" value="확인">  <input type="reset" value="취소">  <a href="./joinFrm.do">회원 가입</a>  </td>  </tr>  </table>  </form>  </body>  </html> |

### joinFrm.jsp

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>회원 가입</title>  </head>  <body>  <h3>회원 가입</h3>  <form method=*"post"* action=*"./join.do"* name=*"userInfo"*>  <table>  <tr>  <td width=*"100"*>아이디</td>  <td>  <input type=*"text"* name=*"userId"* maxlength=*"50"*  required=*"required"*>  </td>  </tr>  <tr>  <td>비밀번호</td>  <td><input type=*"password"* name=*"pass"* maxlength=*"50"*  required=*"required"*>  </td>  </tr>  <tr>  <td>이름</td>  <td>  <input type=*"text"* name=*"userName"* maxlength=*"50"*  required=*"required"*>  </td>  </tr>  <tr>  <td>주소</td>  <td><input type=*"text"* name=*"addr"* size=*"50"*  required=*"required"*></td>  </tr>  <tr>  <td>연락처</td>  <td>  <input type=*"text"* name=*"phone"* maxlength=*"11"*  placeholder=*"숫자만 입력하세요."* size=*"20"*  required=*"required"*>  </td>  </tr>  </table>  <br>  <input type=*"submit"* value=*"가입"* />  <input type=*"button"* value=*"취소"* onclick="location.href='./'">  </form>  </body>  </html> |

### DBconnction.java

|  |
| --- |
| package com.util;  import java.sql.Connection;  import java.sql.SQLException;  import javax.naming.Context;  import javax.naming.InitialContext;  import javax.naming.NamingException;  import javax.sql.DataSource;  public class DBConnection {    public static Connection getConnection() throws SQLException, NamingException,  ClassNotFoundException{  Context initCtx = new InitialContext();    DataSource ds = (DataSource)initCtx.lookup("java:comp/env/jdbc/mysql");  Connection conn = ds.getConnection();    return conn;  }  } |

### Service.java(com.service 패키지)

|  |
| --- |
| package com.service;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  public interface Service {  public ForwardService execute(HttpServletRequest req, HttpServletResponse res) throws Exception;  } |

### ForwardService.java(com.service 패키지)

|  |
| --- |
| package com.service;  public class ForwardService {  private boolean isRedirect = false;  private String path = null;    public boolean isRedirect() {  return isRedirect;  }  public void setRedirect(boolean isRedirect) {  this.isRedirect = isRedirect;  }  public String getPath() {  return path;  }  public void setPath(String path) {  this.path = path;  }  } |

### MemberController.java(com.controller)

|  |
| --- |
| package com.controller;  import java.io.IOException;  import java.sql.SQLException;  import javax.naming.NamingException;  import javax.servlet.RequestDispatcher;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import com.service.ForwardService;  import com.service.Service;  import com.service.member.\*;  @WebServlet("/")  public class MemberController extends HttpServlet {  private static final long serialVersionUID = 1L;  LoginService mService;    public MemberController() {  super();  }  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  doProcess(request, response);  }  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  doProcess(request, response);  }    private void doProcess(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  String command = request.getRequestURI();  HttpSession session = request.getSession();    String view = null;    Service service = null;  ForwardService forward = null;    if (command.indexOf(request.getContextPath()) == 0) {  command = command.substring(request.getContextPath().length());  }  System.out.println(command);    try {  if(command.equals("/")) {  forward = new ForwardService();  forward.setRedirect(false);  forward.setPath("/member/loginFrm.jsp");  }  else if(command.equals("/joinFrm.do")) {  forward = new ForwardService();  forward.setRedirect(false);  forward.setPath("/member/joinFrm.jsp");  }  else if(command.equals("/login.do")) {  service = new LoginService();  forward = service.execute(request, response);  }  else if(command.equals("/logout.do")) {  service = new LogoutService();  forward = service.execute(request, response);  }  else if(command.equals("/join.do")) {  service = new JoinService();  forward = service.execute(request, response);  }  else if(command.equals("/leave.do")) {  service = new LeaveService();  forward = service.execute(request, response);  }    if(forward != null){  if (forward.isRedirect()) {  response.sendRedirect(forward.getPath());  } else {  RequestDispatcher dispatcher = request  .getRequestDispatcher(forward.getPath());  dispatcher.forward(request, response);  }  }  } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }  } |

### LoginService.java

|  |
| --- |
| package com.service.member;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import com.dao.MemberDao;  import com.service.ForwardService;  import com.service.Service;  public class LoginService implements Service{    @Override  public ForwardService execute(HttpServletRequest req, HttpServletResponse res)  throws Exception {  MemberDao mDao = MemberDao.getInstance();  ForwardService forward = new ForwardService();    HttpSession session = req.getSession();    String id = req.getParameter("id");  String pw = req.getParameter("pw");    System.out.println("id: " + id + ", pw: " + pw);    String name = mDao.loginProc(id, pw);    if(name != null) {  System.out.println("name: " + name);  session.setAttribute("id", id);  session.setAttribute("name", name);  forward.setRedirect(true);  **forward.setPath("./list.bo?pnum=1");**  }  else {  session.setAttribute("msg", "0");  forward.setRedirect(true);  forward.setPath("./");  }    return forward;  }  } |

### JoinService.java

|  |
| --- |
| package com.service.member;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import com.bean.Member;  import com.dao.MemberDao;  import com.service.ForwardService;  import com.service.Service;  public class JoinService implements Service {  @Override  public ForwardService execute(HttpServletRequest req, HttpServletResponse res)  throws Exception {  req.setCharacterEncoding("UTF-8");  HttpSession session = req.getSession();    ForwardService forward = new ForwardService();  MemberDao mDao = MemberDao.getInstance();  Member member = new Member();    member.setUserId(req.getParameter("userId"));  member.setPass(req.getParameter("pass"));  member.setUserName(req.getParameter("userName"));  member.setAddr(req.getParameter("addr"));  member.setPhone(req.getParameter("phone"));    int result = mDao.insertMember(member);    if(result > 0) { session.setAttribute("msg", "1"); }//가입 성공  else { session.setAttribute("msg", "2"); }//가입 실패    forward.setRedirect(true);  forward.setPath("./");    return forward;  }  } |

### LogoutService.java

|  |
| --- |
| package com.service.member;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import com.service.ForwardService;  import com.service.Service;  public class LogoutService implements Service {  @Override  public ForwardService execute(HttpServletRequest req, HttpServletResponse res)  throws Exception {  ForwardService forward = new ForwardService();    //세션 내용 삭제  req.getSession().invalidate();    forward.setRedirect(true);  forward.setPath("./");    return forward;  }  } |

### LeaveService.java

|  |
| --- |
| package com.service.member;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import com.dao.MemberDao;  import com.service.ForwardService;  import com.service.Service;  public class LeaveService implements Service {  @Override  public ForwardService execute(HttpServletRequest req, HttpServletResponse res)  throws Exception {  MemberDao mDao = MemberDao.getInstance();  ForwardService forward = new ForwardService();    HttpSession session = req.getSession();    String id = (String)session.getAttribute("id");    int result = mDao.deleteMember(id);    if(result == 1) {  //로그인 정보를 세션에서 삭제  session.removeAttribute("id");  session.removeAttribute("name");    //탈퇴처리를 처음 페이지로 전달하는 세션  session.setAttribute("msg", "2");    forward.setRedirect(true);  forward.setPath("./");  }    return forward;  }  } |

### MemberDao.java

|  |
| --- |
| package com.dao;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import javax.naming.NamingException;  import com.bean.Member;  import com.util.DBConnection;  public class MemberDao {  //싱글톤 인스턴스  private static MemberDao instance;  private MemberDao() {  }  public static MemberDao getInstance() {  if(instance == null) {  instance = new MemberDao();  }  return instance;  }  //로그인 처리 메서드  public String loginProc(String id, String pw) {  String name = null;  String query = "select pass, userName from userInfo where userid=?;";  String dbpw = null;  PreparedStatement pstmt = null;//파라미터 있는 SQL 쿼리문 실행용  ResultSet rs = null;  Connection conn = null;  try {  conn = DBConnection.getConnection();  pstmt = conn.prepareStatement(query);  pstmt.setString(1, id);  rs = pstmt.executeQuery();  if(rs.next()) {  dbpw = rs.getString("pass");  if(pw.equals(dbpw)) {  name = rs.getString("userName");  }  }  } catch (ClassNotFoundException | SQLException | NamingException e) {  e.printStackTrace();  throw new RuntimeException(e.getMessage());  } finally {  // Connection, PreparedStatement를 닫는다.  try{  if(rs != null) { rs.close(); }  if(pstmt != null){ pstmt.close(); }  if(conn != null){ conn.close(); }  }catch(Exception e){  throw new RuntimeException(e.getMessage());  }  }  return name;  }  //회원 가입 처리 메서드  public int insertMember(Member member) throws SQLException {  int result = 0;  String query = "insert into userInfo values (?, ?, ?, ?, ?, default);";    PreparedStatement pstmt = null;  Connection conn = null;  try {  conn = DBConnection.getConnection();  pstmt = conn.prepareStatement(query);  pstmt.setString(1, member.getUserId());  pstmt.setString(2, member.getPass());  pstmt.setString(3, member.getUserName());  pstmt.setString(4, member.getAddr());  pstmt.setString(5, member.getPhone());  pstmt.executeUpdate();  result = pstmt.executeUpdate();  } catch (ClassNotFoundException | SQLException | NamingException e) {  // TODO Auto-generated catch block  e.printStackTrace();  throw new RuntimeException(e.getMessage());  } finally {  // Connection, PreparedStatement를 닫는다.  try{  if(pstmt != null){ pstmt.close(); }  if(conn != null){ conn.close(); }  }catch(Exception e){  throw new RuntimeException(e.getMessage());  }  }  return result;  }  //회원 탈퇴 처리 메서드  public int deleteMember(String id) throws SQLException {  int delResult = 0;  String query = "delete from userInfo where userId=?;";    PreparedStatement pstmt = null;  Connection conn = null;  try {  conn = DBConnection.getConnection();  //자동 승인이 되지 않도록 처리  conn.setAutoCommit(false);  pstmt = conn.prepareStatement(query);  pstmt.setString(1, id);  pstmt.executeUpdate();  conn.commit();//트랜잭션 처리 : 입력 완료 시 최종 실행 승인  delResult = 1;  } catch (ClassNotFoundException | SQLException | NamingException e) {  // TODO Auto-generated catch block  e.printStackTrace();  conn.rollback();//트랜잭션 처리 : 입력 실패시 되돌림  delResult = 0;  } finally {  try{  if(pstmt != null){ pstmt.close(); }  if(conn != null){ conn.close(); }  }catch(Exception e){  throw new RuntimeException(e.getMessage());  }  }  return delResult;  }  } |

### Board.java

|  |
| --- |
| package com.bean;  public class Board {    private String bNo;  private String userId;  private String title;  private String content;  private String bDate;  private String fileName;    public String getbNo() {  return bNo;  }  public void setbNo(String bNo) {  this.bNo = bNo;  }  public String getUserId() {  return userId;  }  public void setUserId(String userId) {  this.userId = userId;  }  public String getTitle() {  return title;  }  public void setTitle(String title) {  this.title = title;  }  public String getContent() {  return content;  }  public void setContent(String content) {  this.content = content;  }  public String getbDate() {  return bDate;  }  public void setbDate(String bDate) {  this.bDate = bDate;  }  public String getFileName() {  return fileName;  }  public void setFileName(String fileName) {  this.fileName = fileName;  }  } |

### BoardController.java(목록 처리까지)

|  |
| --- |
| package com.controller;  import java.io.IOException;  import javax.servlet.RequestDispatcher;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import com.bean.Board;  import com.service.ForwardService;  import com.service.Service;  import com.service.board.ListService;  public class BoardController extends HttpServlet {  private static final long serialVersionUID = 1L;    public BoardController() {  super();  }  protected void doGet(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  doProcess(request, response);  }  protected void doPost(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  doProcess(request, response);  }  private void doProcess(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {  String command = request.getRequestURI();  HttpSession session = request.getSession();    String view = null;    Service service = null;  ForwardService forward = null;    if (command.indexOf(request.getContextPath()) == 0) {  command = command.substring(request.getContextPath().length());  }  System.out.println(command);    try {  if(command.equals("/list.bo")) {  service = new ListService();  forward = service.execute(request, response);  }    if(forward != null){  if (forward.isRedirect()) {  response.sendRedirect(forward.getPath());  } else {  RequestDispatcher dispatcher = request  .getRequestDispatcher(forward.getPath());  dispatcher.forward(request, response);  }  }  } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }    }  } |

### ListService.java

|  |
| --- |
| package com.service.board;  import java.util.ArrayList;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import javax.servlet.http.HttpSession;  import com.bean.Board;  import com.dao.BoardDao;  import com.service.ForwardService;  import com.service.Service;  public class ListService implements Service {  @Override  public ForwardService execute(HttpServletRequest req, HttpServletResponse res)  throws Exception {  ForwardService forward = new ForwardService();    HttpSession session = req.getSession();    String userId = (String)session.getAttribute("id");    System.out.println("id: " + userId);    int pageNum = 1;  String pNum = req.getParameter("pnum");    System.out.println("pnum: " + pNum);  if(pNum != null) {  pageNum = Integer.parseInt(req.getParameter("pnum"));  }    BoardDao bDao = BoardDao.getInstance();    int listCnt = bDao.getBoardListCnt();//전체 글 개수 구하기    ArrayList<Board> bList = bDao.getBoardList(pageNum);    // 전체 페이지 수  int maxPage = (int)(listCnt/10.0 + 0.9);  //시작 페이지 번호  int startPage = (int)(pageNum/5.0 + 0.8) \* 5 - 4;  //마지막 페이지 번호  int endPage = startPage + 4;  if(endPage > maxPage) {  endPage = maxPage;  }    // 4개 페이지번호 저장  req.setAttribute("spage", pageNum);  req.setAttribute("maxPage", maxPage);  req.setAttribute("startPage", startPage);  req.setAttribute("endPage", endPage);    req.setAttribute("bList", bList);    forward.setRedirect(false);  forward.setPath("/board/boardList.jsp");    return forward;  }  } |

### BoardDao.java

|  |
| --- |
| package com.dao;  import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.sql.Statement;  import java.util.ArrayList;  import javax.naming.NamingException;  import com.bean.Board;  import com.util.DBConnection;  public class BoardDao {  private static BoardDao *instance*;    private BoardDao() {  }    public static BoardDao getInstance() {  if(*instance* == null) {  *instance* = new BoardDao();  }  return *instance*;  }  public int getBoardListCnt() {  int result = 0;  String query = "select count(\*) as cnt from boardTbl;";    Connection conn = null;  Statement stmt = null;  ResultSet rs = null;    try {  conn = DBConnection.*getConnection*();  stmt = conn.createStatement();  rs = stmt.executeQuery(query);    if(rs.next()) {  result = rs.getInt("cnt");  }  } catch (ClassNotFoundException | SQLException | NamingException e) {  // TODO Auto-generated catch block  e.printStackTrace();  throw new RuntimeException(e.getMessage());  } finally {  try{  if(rs != null) { rs.close(); }  if(stmt != null){ stmt.close(); }  if(conn != null){ conn.close(); }  }catch(Exception e){  e.printStackTrace();  throw new RuntimeException(e.getMessage());  }  }  return result;  }    public ArrayList<Board> getBoardList(){  ArrayList<Board> bList = new ArrayList<>();    String query = "select \* from boardTbl;";    Connection conn = null;  Statement stmt = null;//파라미터 있는 SQL 쿼리문 실행용  ResultSet rs = null;  try {  conn = DBConnection.*getConnection*();    stmt = conn.createStatement();  rs = stmt.executeQuery(query);    while(rs.next()) {  Board board = new Board();    board.setbNo(String.*valueOf*(rs.getInt("bNo")));  board.setUserId(rs.getString("userId"));  board.setTitle(rs.getString("title"));  board.setContent(rs.getString("content"));  board.setbDate(rs.getString("bDate"));    bList.add(board);  }  } catch (ClassNotFoundException | SQLException | NamingException e) {  e.printStackTrace();  throw new RuntimeException(e.getMessage());  }  return bList;  }  } |

### boardList.jsp

|  |
| --- |
| <%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>  <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>게시판</title>  <script>  function leave(){  var res = confirm('회원을 탈퇴하시겠습니까?');  console.log(res);  if(res){  location.href='./leave.do'  }  }  </script>  <c:if test="${empty id}">  <script>  alert("잘못된 접근입니다.");  location.href="./";  </script>  </c:if>  <c:choose>  <c:when test="${bmsg == '1'}">  <script>  alert("글이 저장되었습니다..");  </script>  <c:remove var="bmsg" scope="session"/>  </c:when>  <c:when test=*"*${bmsg == '2'}*"*>  <script>  alert("글이 저장에 실패하였습니다.");  </script>  <c:remove var=*"bmsg"* scope=*"session"*/>  </c:when>  <c:when test="${bmsg == '3'}">  <script>  alert("글이 삭제되었습니다.");  </script>  <c:remove var="bmsg" scope="session"/>  </c:when>  </c:choose>  <style>  th{  background-color: skyblue;  text-align: center  }  </style>  </head>  <body>  <h3>회원제 게시판</h3>  <table>  <tr>  <td width="600" style="text-align: right">${name}님 환영합니다.  <input type="button" value="로그아웃"  onclick="location.href='./logout.do'">  <input type="button" value="회원탈퇴" onclick="leave()">  </td>  </tr>  </table>  <br>  <table>  <tr>  <th width="50">번호</th>  <th width="250">제목</th>  <th width="100">작성자</th>  <th width="180">작성일시</th>  </tr>  <c:forEach var="board" items="${bList}">  <tr>  <td align="center">${board.bNo}</td>  <td align="center">  <a href="./content.bo?bNo=${board.bNo}">${board.title}</a>  </td>  <td align="center">${board.userId}</td>  <td align="center">${board.bDate}</td>  </tr>  </c:forEach>  <tr></tr>  <tr>  <td colspan=*"4"* align=*"center"*>  <c:if test=*"*${startPage != 1}*"*>  <a href=*'./list.bo?pnum=*${startPage-1}*'*>[이전]</a>  </c:if>  <c:forEach var=*"pageNum"* begin=*"*${startPage}*"* end=*"*${endPage}*"*>  <c:if test=*"*${pageNum == spage}*"*>  ${pageNum}&nbsp;  </c:if>  <c:if test=*"*${pageNum != spage}*"*>  <a href=*'./list.bo?pnum=*${pageNum}*'*>${pageNum}&nbsp;</a>  </c:if>  </c:forEach>  <c:if test=*"*${endPage != maxPage }*"*>  <a href=*'./list.bo?pnum=*${endPage+1}*'*>[다음]</a>  </c:if>  </td>  </tr>  </table>  <br>  <input type="button" value="글쓰기" onclick="location.href='writeFrm.bo'">  </body>  </html> |