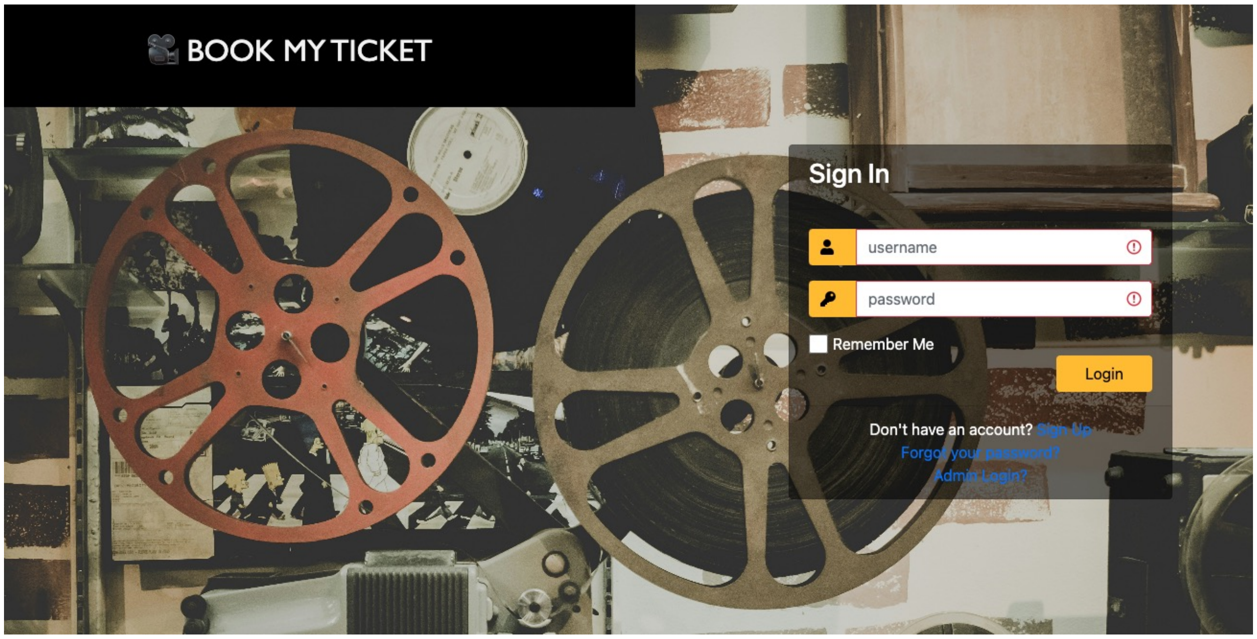
BookMyMovie

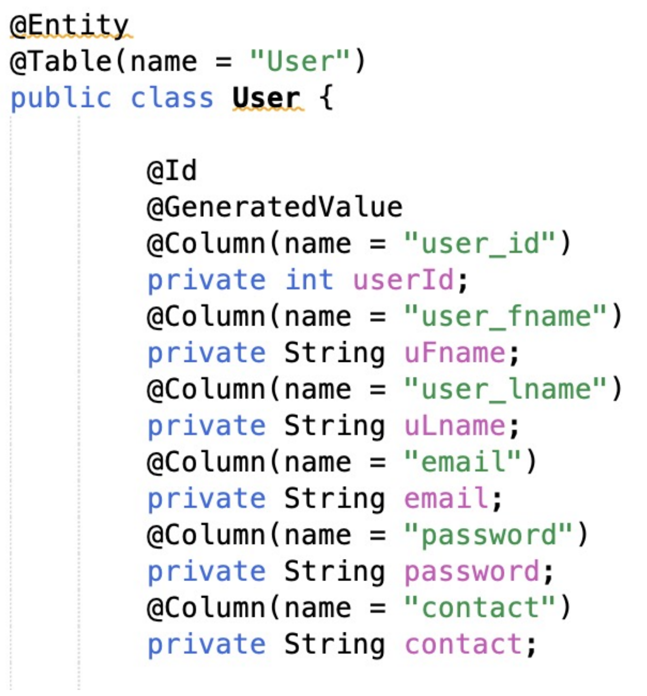
Project Overview



https://www.canva.com/design/DAGODY5ICz4/rMFa59nkykshrypY3×7WHQ/view?utm_content=DAGODY5ICz4&utm_campaign=designshare&utm_medium=link&utm_source=editor

Define Class (using Annotation)

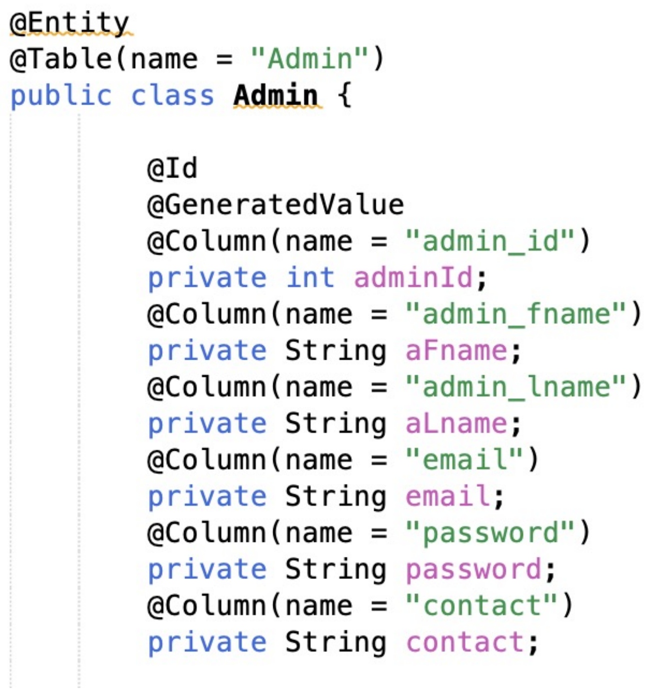
1.User



```
@Entity
@Table(name = "User")
public class User {

        @Id
        @GeneratedValue
        @Column(name = "user_id")
        private int userId;
        @Column(name = "user_fname")
        private String uFname;
        @Column(name = "user_lname")
        private String uLname;
        @Column(name = "email")
        private String email;
        @Column(name = "password")
        private String password;
        @Column(name = "contact")
        private String contact;
```

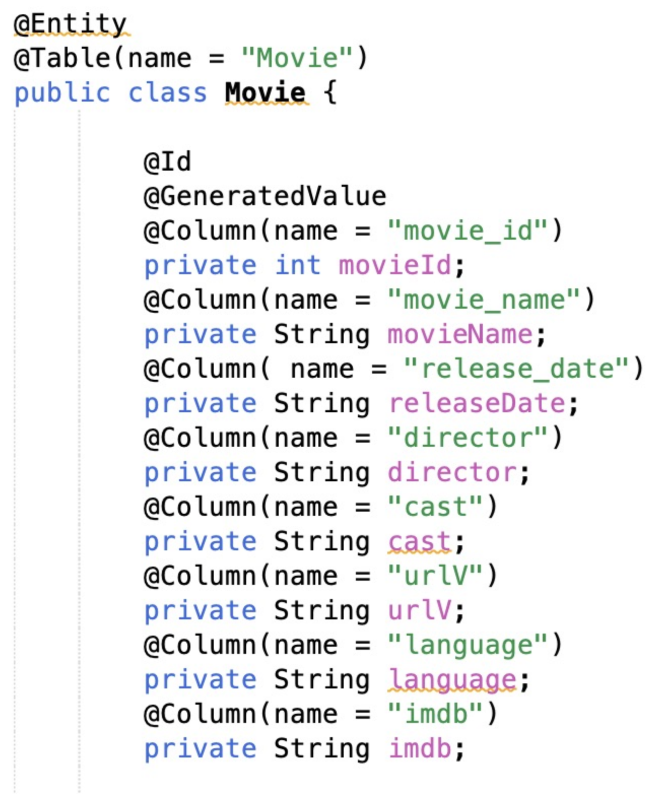
2.Admin



```
@Entity
@Table(name = "Admin")
public class Admin {

        @Id
        @GeneratedValue
        @Column(name = "admin_id")
        private int adminId;
        @Column(name = "admin_fname")
        private String aFname;
        @Column(name = "admin_lname")
        private String aLname;
        @Column(name = "email")
        private String email;
        @Column(name = "password")
        private String password;
        @Column(name = "contact")
        private String contact;
```

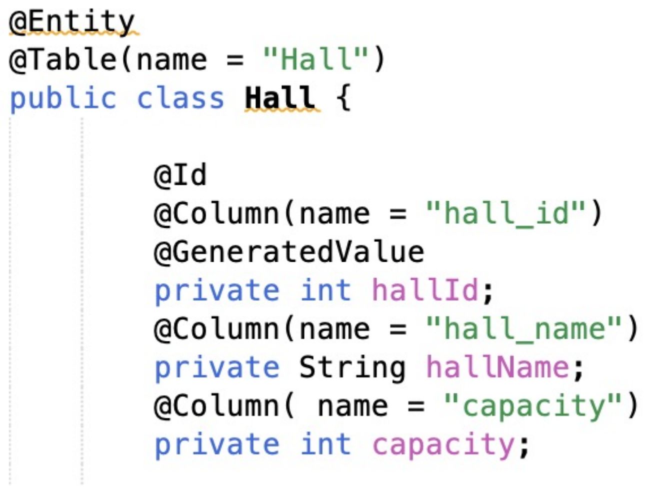
3.Movie



```
@Entity
@Table(name = "Movie")
public class Movie {

        @Id
        @GeneratedValue
        @Column(name = "movie_id")
        private int movieId;
        @Column(name = "movie_name")
        private String movieName;
        @Column( name = "release_date")
        private String releaseDate;
        @Column(name = "director")
        private String director;
        @Column(name = "cast")
        private String cast;
        @Column(name = "urlV")
        private String urlV;
        @Column(name = "language")
        private String language;
        @Column(name = "imdb")
        private String imdb;
```

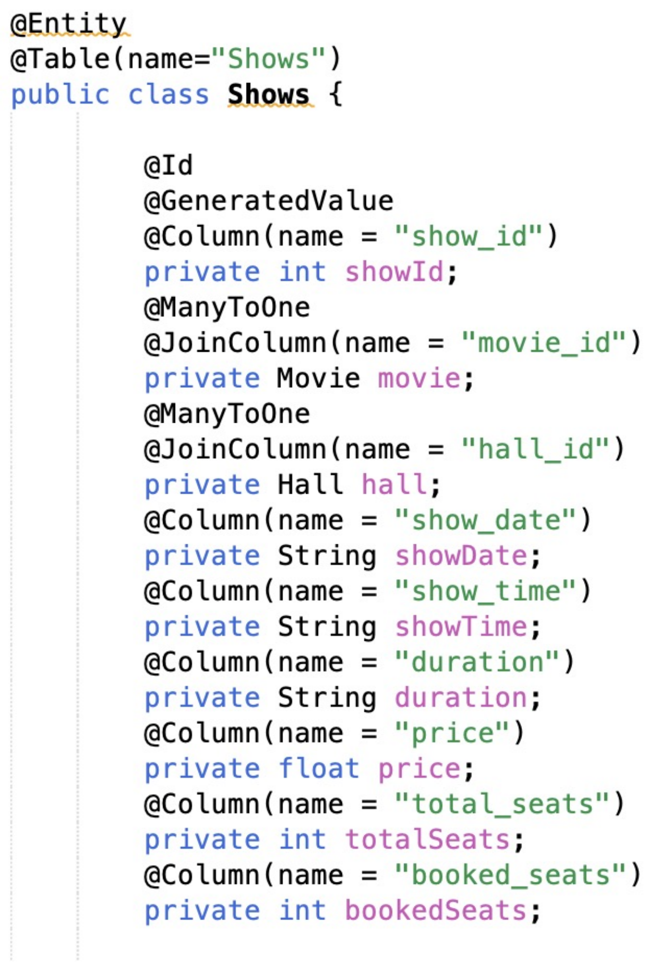
4.Hall



```
@Entity
@Table(name = "Hall")
public class Hall {

        @Id
        @Column(name = "hall_id")
        @GeneratedValue
        private int hallId;
        @Column(name = "hall_name")
        private String hallName;
        @Column( name = "capacity")
        private int capacity;
```

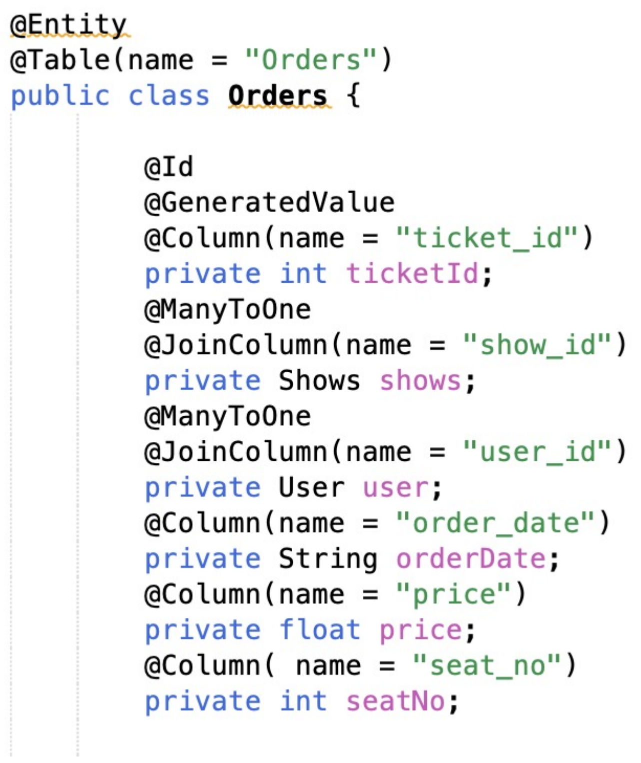
5.Shows



```java
@Entity
@Table(name="Shows")
public class Shows {

        @Id
        @GeneratedValue
        @Column(name = "show_id")
        private int showId;
        @ManyToOne
        @JoinColumn(name = "movie_id")
        private Movie movie;
        @ManyToOne
        @JoinColumn(name = "hall_id")
        private Hall hall;
        @Column(name = "show_date")
        private String showDate;
        @Column(name = "show_time")
        private String showTime;
        @Column(name = "duration")
        private String duration;
        @Column(name = "price")
        private float price;
        @Column(name = "total_seats")
        private int totalSeats;
        @Column(name = "booked_seats")
        private int bookedSeats;
```

6.Orders



```
@Entity
@Table(name = "Orders")
public class Orders {

        @Id
        @GeneratedValue
        @Column(name = "ticket_id")
        private int ticketId;
        @ManyToOne
        @JoinColumn(name = "show_id")
        private Shows shows;
        @ManyToOne
        @JoinColumn(name = "user_id")
        private User user;
        @Column(name = "order_date")
        private String orderDate;
        @Column(name = "price")
        private float price;
        @Column( name = "seat_no")
        private int seatNo;
```

Spring validator class: `UserValidator` is used to validate the user's input when submitting a form. The primary purpose of this class is to ensure that the user data meets certain criteria and formats before being processed further.

e.g. password, email, contact format

```
package com.bookmovie.valid;

import java.util.regex.Pattern;

import org.springframework.stereotype.Service;
import org.springframework.validation.Errors;
import org.springframework.validation.ValidationUtils;
import org.springframework.validation.Validator;

import com.bookmovie.pojo.User;

@Service
public class UserValidator implements Validator {
    @Override
    public boolean supports(Class<?> clazz) {
        return clazz.equals(User.class);
    }

    @Override
    public void validate(Object target, Errors errors) {
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "emai
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "passw

        User user = (User)target;
        if(user!=null) {
            if(user.getPassword().length()<8) {
                errors.rejectValue("password", "passKey", "*pass
            }

            boolean b2=Pattern.compile("^[a-zA-Z0-9._%+-]+@[a-z
            if(!b2) {
                errors.rejectValue("email", "unmKey", "Email mus
            }
            if(user.getuFname()!=null) {
                ValidationUtils.rejectIfEmptyOrWhitespace(errors
                if(user.getuFname().length()<20) {
                    errors.rejectValue("uFname", "ufname1","*Fi
                }
            }
            if(user.getuLname()!=null) {
                ValidationUtils.rejectIfEmptyOrWhitespace(errors
                if(user.getuLname().length()<20) {
                    errors.rejectValue("uLname", "ulname1","*Fi
                }
            }
            if(user.getContact()!=null) {
                ValidationUtils.rejectIfEmptyOrWhitespace(errors
                if(user.getContact().toString().length()<8 || us
                    errors.rejectValue("contact", "contact1","Mo
                }
            }
        }


    }


}
```

DAO pattern

Interact with the database

@Repository

@Autowired

addUser(User user), getUser(User user), getMovies(), getShow(int movieId), forgotPassword(String email), getOrderForUser(int userId)

```
package com.bookmovie.dao;

import java.util.List;

import org.hibernate.HibernateException;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.orm.hibernate4.HibernateCallback;
import org.springframework.orm.hibernate4.HibernateTemplate;
import org.springframework.stereotype.Repository;

import com.bookmovie.pojo.Movie;
import com.bookmovie.pojo.Orders;
import com.bookmovie.pojo.Shows;
import com.bookmovie.pojo.User;

@Repository
public class UserDaoImple implements UserDao {

    @Autowired
    HibernateTemplate hibernateTemplate;


    @Override
    public boolean addUser(User user) {
        return hibernateTemplate.execute(new HibernateCallback<

            @Override
            public Boolean doInHibernate(Session session) throws

                Transaction trans = session.beginTransaction();
                Query q = session.createQuery("from User where 
                q.setString(0, user.getEmail());
                List<User> li = q.list();
                if(!li.isEmpty()) {
                    trans.commit();
                    session.flush();
                    session.close();
                    return false;
                }
                session.save(user);
                trans.commit();
                session.flush();
                session.close();

                return true;

            }
        });

    }

    @Override
    public User getUser(User user) {

        User usr = hibernateTemplate.execute(new HibernateCallba

            @Override
            public User doInHibernate(Session session) throws Hi

                Transaction trans = session.beginTransaction();
                Query q = session.createQuery("from User where e
                q.setString(0, user.getEmail());
                q.setString(1, user.getPassword());
                List<User> li = q.list();
                if(li.isEmpty()) {
                    return null;
                }
                else {
                    return li.get(0);
                }
            }
        });

        return usr;
    }

    @Override
    public List<Movie> getMovies() {

        List<Movie> li = hibernateTemplate.execute(new Hibernate

            @Override
            public List<Movie> doInHibernate(Session session) th
```

```
                Transaction trans = session.beginTransaction();
                Query q = session.createQuery("from Movie");
                List<Movie> l1 = q.list();
                if(!l1.isEmpty()) {
                    return l1;
                }
                return null;
            }} );
        return li;
    }

    @Override
    public List<Shows> getShow(int movieId) {

        return hibernateTemplate.execute(new HibernateCallback<

            @Override
            public List<Shows> doInHibernate(Session session) th
                Transaction trans = session.beginTransaction();
                Query q = session.createQuery("from Shows where
                //System.out.println("hiiiii");
                q.setInteger(0, movieId);
                List<Shows> l1 = q.list();
                //System.out.println("List hereeee");
                if(!l1.isEmpty()) {
                    System.out.println(l1.get(0));
                    return l1;
                }
                return null;
            }} );
    }

    @Override
    public String forgotPassword(String email) {
        String password = hibernateTemplate.execute(new Hibernat
            @Override
            public String doInHibernate(Session session) throws
                Transaction tr = session.beginTransaction();
                Query q = session.createQuery("from User where e
                q.setString(0, email);
                List<User> li = q.list();
                String pass = null;
                if(!li.isEmpty())
                    pass = li.get(0).getPassword();
                tr.commit();
                session.flush();
                session.close();
                return pass;
            }

        });
        return password;
    }

    @Override
    public List<Orders> getOrderForUser(int userId) {

        return  hibernateTemplate.execute(new HibernateCallback<

            @Override
            public List<Orders> doInHibernate(Session session) t
                Transaction tr = session.beginTransaction();
                Query q = session.createQuery("from Orders where
                q.setInteger(0, userId);
                List<Orders> li = q.list();

                if(!li.isEmpty()) {

                tr.commit();
                session.flush();
                session.close();
```

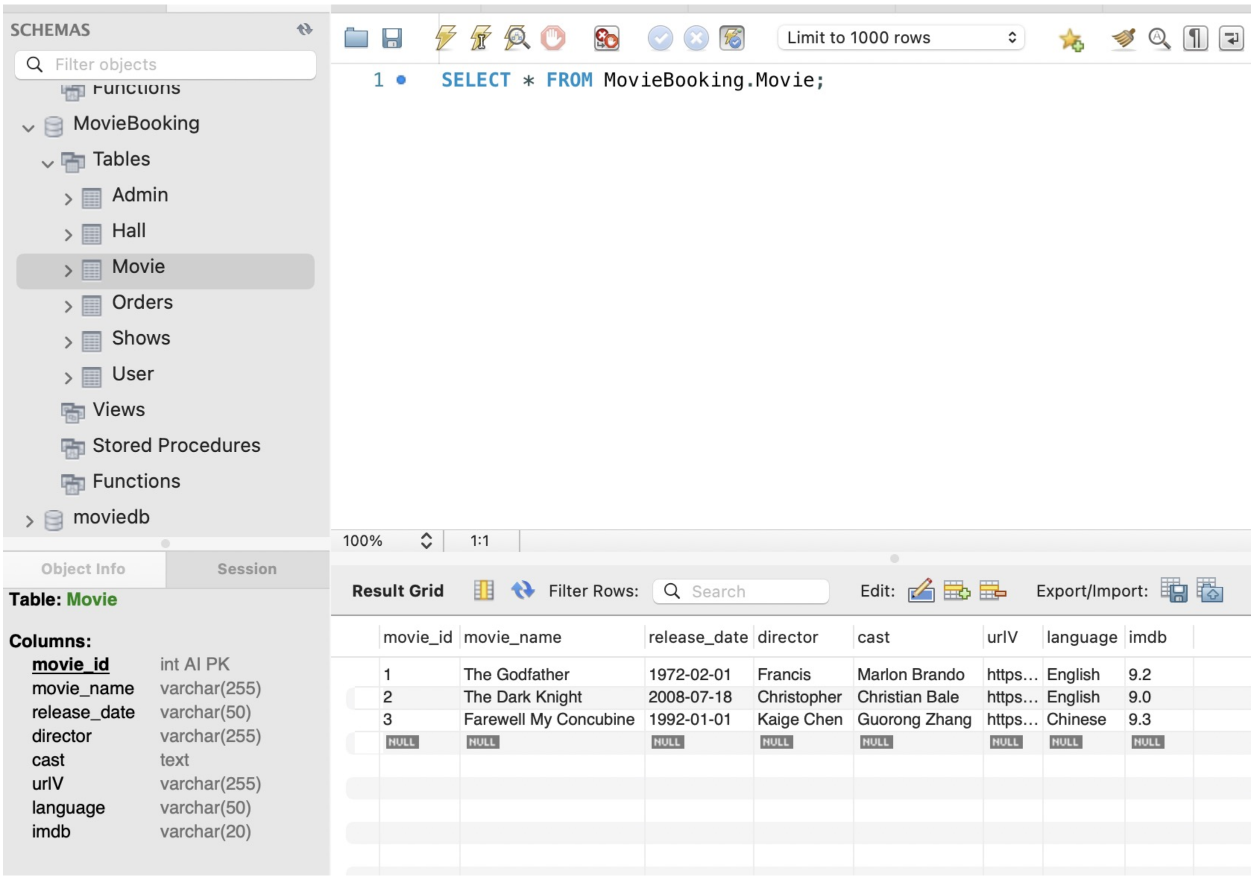
```
                  return li;

                  }
                  return null;


      }});

      }
  }
```

Database (MySQL)



SCHEMAS

- Functions
- MovieBooking
  - Tables
    - Admin
    - Hall
    - Movie
    - Orders
    - Shows
    - User
  - Views
  - Stored Procedures
  - Functions
- moviedb

```
SELECT * FROM MovieBooking.Movie;
```

Table: Movie

Columns:

movie_id	int AI PK
movie_name	varchar(255)
release_date	varchar(50)
director	varchar(255)
cast	text
urlV	varchar(255)
language	varchar(50)
imdb	varchar(20)

movie_id	movie_name	release_date	director	cast	urlV	language	imdb
1	The Godfather	1972-02-01	Francis	Marlon Brando	https…	English	9.2
2	The Dark Knight	2008-07-18	Christopher	Christian Bale	https…	English	9.0
3	Farewell My Concubine	1992-01-01	Kaige Chen	Guorong Zhang	https…	Chinese	9.3
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Controller （Spring MVC）

Handle user requests and call DAO

Handles various HTTP requests related to user and admin actions like logging in, registering, managing movies, halls, and booking shows

```
package com.bookmovie.controller;

import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.mail.MailSender;
import org.springframework.mail.SimpleMailMessage;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;

import com.bookmovie.pojo.Admin;
import com.bookmovie.pojo.Hall;
import com.bookmovie.pojo.Movie;
import com.bookmovie.pojo.Orders;
import com.bookmovie.pojo.Shows;
import com.bookmovie.pojo.User;
import com.bookmovie.service.AdminService;
import com.bookmovie.service.UserService;
import com.bookmovie.valid.UserValidator;

@Controller
public class UserController {
    @Autowired
    private UserService userService;

    @Autowired
    private AdminService adminService;

    @Autowired
    private UserValidator userValidator;

    @Autowired
    private MailSender mailSender;

    @RequestMapping(value = "/forgot_password.htm",method = Requ
    public String forgotPassword(@RequestParam String email,Mode
        String pass = userService.forgotPassword(email);
        String msg = "you are not registered";
        if(pass!=null) {

            SimpleMailMessage message = new SimpleMailMessage()
            message.setFrom("adinathraut3000@gmail.com");
            message.setTo(email);
            message.setSubject("Your password");
            message.setText(pass);
            //sending message
            mailSender.send(message);
            msg = "Check your mail for password";

            session.setAttribute("err_mssg1", "Password Success
            return "index";
        }
        session.setAttribute("err_mssg", "Entered Email is not 
        return "error_page";
    }
    @RequestMapping(value = "/prep_reg_form.htm",method = Reques
    public String prepRegForm(ModelMap map) {
        map.put("user", new User());
        return "signup_page_user";
    }

    @RequestMapping(value = "/prep_log_form.htm",method = Reques
    public String prepLogForm(ModelMap map) {
        map.put("user", new User());
        return "login_page";
    }

    @RequestMapping(value = "/prep_logout.htm",method = RequestM
    public String prepLogout(ModelMap map, HttpSession session)
        System.out.println((User)session.getAttribute("user"));
        session.removeAttribute("user");
        session.invalidate();
        return "index";
    }

    @RequestMapping(value = "/registerValidateUser.htm",method =
    public String registerationValidateUser(User user,  BindingR
        //userValidator.validate(user, result);
        map.put("user", new User());
        /*if(result.hasErrors()) {
            return "signup_page_user";
        }*/
        if(userService.insertUser(user)) {
            map.put("user", new User());
            session.setAttribute("err_mssg1", "Account Successfu
            return "login_page";
        }
        session.setAttribute("err_mssg", "Account Already Exists
        return "signup_page_user";
    }
    @RequestMapping(value = "/loginValidate.htm", method = Reque
    public String loginValidate(User user, BindingResult result,

        session.removeAttribute("err_mssg");
        session.removeAttribute("err_mssg1");

        userValidator.validate(user, result);
        map.put("user", new User());
        if(result.hasErrors()) {
            return "login_page";
        }

        User user1 = userService.getUser(user);
        System.out.println(user1);
        if(user1 == null) {
            session.setAttribute("err_mssg", "Invalid Credentia
            return "login_page";
        }
        else {
            session.setAttribute("user", user1);

            List<Movie> li = userService.getMovies();
            if(li == null) {
                return "error_page";
            }
            map.put("user", user1);
            session.setAttribute("user", user1);
            map.put("mList", li);
            return "home_page";
        }
    }

    @RequestMapping(value = "/homeValidate.htm", method = Reques
    public String homeValidate1(HttpSession session, ModelMap ma

        if(session.getAttribute("user") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";

          }
              List<Movie> li = userService.getMovies();
                if(li == null) {
                    return "error_page";
                }
                map.put("mList", li);
                return "home_page";
 }



    @RequestMapping(value = "/prep_show.htm",method = RequestMet
    public String prepShow(ModelMap map, HttpServletRequest requ
        if(session.getAttribute("user") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        int mId = Integer.parseInt(request.getParameter("movie_i

        List<Shows> li = userService.getShow(mId);
        if(li == null) {
            session.setAttribute("err_mssg", "OOPs No Shows Avai
            return "error_page";
        }
        map.put("sList", li);
        return "shows_page";
    }

    @RequestMapping(value = "/prep_search_page.htm", method = Re
    public String prepSearchPage(User user, ModelMap map, HttpSe

        if(session.getAttribute("user") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";

          }

        List<Movie> li = userService.getMovies();
        if(li == null) {
            return "error_page";
        }

        map.put("mList", li);

            return "search_page";

    }
    //user Part

    @RequestMapping(value = "/prep_user_buy_orders.htm", method 
    public String userBuyOrders(User user, ModelMap map, HttpSes

        if(session.getAttribute("user") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";

          }
                int userId = ((User)session.getAttribute("user")
              List<Orders> li = userService.getOrdersForUser(use
                if(li == null) {
                    return "home_page";
                }
                map.put("mList", li);
                return "user_buy_order_page";
 }
    //Booking Part

        @RequestMapping(value = "/prep_user_orders.htm",method =
        public String prepUserOrders(ModelMap map, User user, Ht
            if(session.getAttribute("user") == null){
                 session.setAttribute("err_mssg", "Please Login
                 return "index";

              }
            if(request.getParameter("show_id") == null) {
                session.setAttribute("err_mssg", "Error Occured
                 return "index";
            }

            int showId = Integer.parseInt(request.getParameter("
            Shows shows =  adminService.getShow(showId);
            int lastSeatNo = shows.getBookedSeats();
            int totalSeats = shows.getTotalSeats();
            Orders orders = new Orders();

            if(lastSeatNo <= totalSeats) {
                User user1 = (User)session.getAttribute("user")

                 int seatNo = lastSeatNo + 1;
                 float price = shows.getPrice() + ((shows.getPr

                 System.out.println(java.time.LocalDate.now());
                 String odrDate = java.time.LocalDate.now().toS
                 shows.setBookedSeats(seatNo);
                 orders.setOrderDate(odrDate);
                 orders.setSeatNo(seatNo);
                 orders.setShows(shows);
                 orders.setPrice(price);
                 orders.setUser(user1);
                 map.put("shows", shows);
                 map.put("orders", orders);
                 return "orders_page_confirm";
            }
            else {
```

```
                map.put("err_msgShow", "No Seats Avaliable for t
                return "home_page";

            }
        }


        @RequestMapping(value = "/confirm_book_page.htm",method 
        public String confirmShowBook(ModelMap map, User user,Ht

            Orders orders1 = (Orders)session.getAttribute("order
            adminService.addOrders(orders1);
            adminService.updateShow(orders1.getShows());
            map.put("orders", orders1);
            return "ticket_page";
        }

    //Admin Part

    @RequestMapping(value = "/prep_admin_log.htm",method = Reque
    public String prepAdminLogForm(ModelMap map) {
        map.put("admin", new Admin());
        return "login_admin_page";
    }
    @RequestMapping(value = "/prep_admin_logout.htm",method = Re
    public String prepAdminLogout(ModelMap map, HttpSession sess

        session.removeAttribute("admin");
        session.invalidate();
        return "index";
    }
    @RequestMapping(value = "/admin_log_validate.htm",method = R
    public String AdminLogValidate(ModelMap map, Admin admin, Ht

        session.removeAttribute("err_mssg");
        session.removeAttribute("err_mssg1");
        map.put("admin", new Admin());

        System.out.println(admin);
        Admin admin1 = adminService.getAdmin(admin);
        map.put("admin", new Admin());
        System.out.println(admin1);
        if(admin1 == null) {
            session.setAttribute("err_mssg", "Invalid Credentia
            return "login_admin_page";
        }

            map.put("admin", admin1);
            session.setAttribute("admin", admin1);
            return "admin_home_page";



    }

    @RequestMapping(value = "/admin_home.htm",method = RequestMe
    public String adminHome(ModelMap map, Admin admin, HttpSessi
        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Aga
             return "index";
          }

        return "admin_home_page";
    }

    @RequestMapping(value = "/prep_admin_shows.htm",method = Rec
    public String adminShows(ModelMap map, Admin admin, HttpSess

        List<Shows> li = adminService.showsAll();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("sList",li);
        //System.out.println("update prepared");
        return "admin_shows";
    }


    @RequestMapping(value = "/prep_show_delete.htm",method = Rec
    public String adminShowsDelete(ModelMap map, Admin admin, Ht

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }

        if((request.getParameter("movie_id") == null) || (reques
            session.setAttribute("err_mssgShows", "Cannot Delete
             return "admin_home_page";
        }

        int movieId = Integer.parseInt(request.getParameter("mov
        int showId = Integer.parseInt(request.getParameter("show
        int hallId = Integer.parseInt(request.getParameter("hall
        //System.out.println(movieId + " "+ " " + showId + " " +

        adminService.deleteShows(showId);
        List<Shows> li = adminService.showsAll();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("sList",li);
        session.setAttribute("err_mssgShows1", "Show Deleted Suc
        return "admin_shows";
    }
    @RequestMapping(value = "/prep_show_update.htm",method = Rec
    public String adminShowsUpdate(ModelMap map, Admin admin, Ht
        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        if((request.getParameter("show_id")==null)) {
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
        }
        int movieId = Integer.parseInt(request.getParameter("mov
        int showId = Integer.parseInt(request.getParameter("show
        int hallId = Integer.parseInt(request.getParameter("hall

        Shows shows = adminService.getShow(showId);
        map.put("shows", shows);
        map.put("movieId", movieId);
        map.put("hallId",hallId);
        return "admin_shows_update_form";
    }

    @RequestMapping(value = "/admin_show_update_Done.htm",method
    public String adminShowsUpdate(ModelMap map, Shows shows, Ht

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        //System.out.println(shows.getHall());

        boolean flag = adminService.updateShow(shows);
        List<Shows> li = adminService.showsAll();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("sList",li);

        if(!flag) {
            session.setAttribute("err_mssgShows", "Entered Movie
            return "admin_shows";
```

```
        }

        session.setAttribute("err_mssgShows1", "Show Updated Suc
        return "admin_shows";
    }

    @RequestMapping(value = "/admin_shows_add_form.htm",method =
    public String adminShowsAddForm(ModelMap map, Shows shows, H

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        map.put("shows",new Shows());

        return "admin_shows_add_form";
    }

    @RequestMapping(value = "/admin_show_add_done.htm",method =
    public String adminShowsAddFormDone(ModelMap map, Shows show

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }

        boolean flag = adminService.addShow(shows);
        List<Shows> li = adminService.showsAll();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("sList",li);
        if(!flag) {
            session.setAttribute("err_mssgShows", "Movie ID or H
            return "admin_shows";
        }
        session.setAttribute("err_mssgShows1", "Show Added Succe
        return "admin_shows";
    }

    @RequestMapping(value = "/prep_admin_orders.htm",method = Re
    public String prepAdminOrders(ModelMap map, Admin admin, Htt

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }

        List<Orders> li = adminService.showOrders();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("oList",li);
        //System.out.println("update prepared");
        return "admin_orders";
    }

    @RequestMapping(value = "/prep_order_delete.htm",method = Re
    public String prepAdminOrdersDelete(ModelMap map, Admin admi

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        if(request.getParameter("ticket_id")==null) {
             session.setAttribute("err_mssgOrder", "Error while 
             return "admin_home_page";
        }
        int ticketId = Integer.parseInt(request.getParameter("ti

        adminService.deleteOrders(ticketId);
        session.setAttribute("err_mssgOrder1", "Order Deleted Su
        List<Orders> li = adminService.showOrders();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("oList",li);
        return "admin_orders";
    }

    @RequestMapping(value = "/prep_order_update.htm",method = Re
    public String prepAdminOrdersUpdate(ModelMap map, Admin admi

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        if(request.getParameter("ticket_id")==null) {
             session.setAttribute("err_mssgOrder", "Error while 
             return "admin_home_page";
        }
        int ticketId = Integer.parseInt(request.getParameter("ti
        Orders orders = adminService.getOrder(ticketId);
        map.put("orders",orders);
        return "admin_orders_update_form";
    }

    @RequestMapping(value = "/admin_orders_update_done.htm",meth
    public String adminOrdersUpdateDone(ModelMap map, Orders ord

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }

        boolean flag = adminService.updateOrders(orders);
        List<Orders> li = adminService.showOrders();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("oList",li);
        System.out.println(flag);
        if(!flag) {
            session.setAttribute("err_mssgOrder", "Entered Show
            return "admin_orders";
        }
        System.out.println("update done");
        session.setAttribute("err_mssgOrder1", "Order Updated Su
        return "admin_orders";
    }

    @RequestMapping(value = "/prep_admin_movie.htm",method = Req
    public String prepAdminMovie(ModelMap map, Admin admin, Http

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }

        List<Movie> li = adminService.showMovies();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("mList",li);
        //System.out.println("update prepared");
        return "admin_movie";
    }

    @RequestMapping(value = "/prep_admin_movie_update.htm",metho
    public String prepAdminMovieUpdate(ModelMap map, Admin admin

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        if(request.getParameter("movie_id")==null) {
             session.setAttribute("err_mssgMovie", "Error while
             return "admin_home_page";
        }
        int movieId = Integer.parseInt(request.getParameter("mov
        Movie movie = adminService.getMovie(movieId);
        map.put("movie", movie);
        //System.out.println("update prepared");
        return "admin_movie_update_form";
    }

    @RequestMapping(value = "/admin_movie_update_done.htm",metho
    public String prepAdminMovieUpdateDone(ModelMap map, Movie m

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        boolean flag = adminService.updateMovie(movie);
        List<Movie> li = adminService.showMovies();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("mList",li);
        if(!flag) {
            session.setAttribute("err_mssgMovie", "Error while U
            return "admin_movie";
        }
        session.setAttribute("err_mssgMovie1", "Movie Updated Su
        return "admin_movie";
    }

    @RequestMapping(value = "/prep_admin_movie_add.htm",method =
    public String prepAdminMovieAdd(ModelMap map, Admin admin, H
        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }


        map.put("movie", new Movie());

        return "admin_movie_add_form";
    }
    @RequestMapping(value = "/admin_movie_add_done.htm",method =
    public String adminMovieAddDone(ModelMap map, Movie movie, H

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        boolean flag = adminService.addMovie(movie);
        List<Movie> li = adminService.showMovies();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("mList",li);
        if(!flag) {
            session.setAttribute("err_mssgMovie", "Error while A
            return "admin_movie";
        }
        session.setAttribute("err_mssgMovie1", "Movie Added Succ
        return "admin_movie";
    }

    //Hall Connection and Data Entry

    @RequestMapping(value = "/prep_admin_hall.htm",method = Requ
    public String prepAdminHall(ModelMap map, Admin admin, HttpS
        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }

        List<Hall> li = adminService.showHalls();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("hList",li);
        //System.out.println("update prepared");
        return "admin_hall";
    }

    @RequestMapping(value = "/prep_admin_hall_update.htm",methoc
    public String prepAdminHallUpdate(ModelMap map, Admin admin,

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        if(request.getParameter("hall_id")==null) {
             session.setAttribute("err_mssgMovie", "Error while
             return "admin_home_page";
        }
        int hallId = Integer.parseInt(request.getParameter("hall
        Hall hall = adminService.getHall(hallId);
        map.put("hall", hall);
        //System.out.println("update prepared");
        return "admin_hall_update_form";
    }

    @RequestMapping(value = "/admin_hall_update_done.htm",methoc
    public String prepAdminHallUpdateDone(ModelMap map, Hall hal

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        boolean flag = adminService.updateHall(hall);
        List<Hall> li = adminService.showHalls();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("hList",li);
        if(!flag) {
            session.setAttribute("err_mssgHall", "Error while Up
            return "admin_movie";
        }
        session.setAttribute("err_mssgHall1", "Hall Updated Succ
        return "admin_hall";
    }

    @RequestMapping(value = "/prep_admin_hall_add.htm",method =
    public String prepAdminHallAdd(ModelMap map, Admin admin, Ht

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }

        map.put("hall", new Hall());

        return "admin_hall_add_form";
    }
    @RequestMapping(value = "/admin_hall_add_done.htm",method =
    public String adminHallAddDone(ModelMap map, Hall hall, Http

        if(session.getAttribute("admin") == null){
             session.setAttribute("err_mssg", "Please Login Agai
             return "index";
          }
        System.out.println(hall);
        boolean flag = adminService.addHall(hall);
        List<Hall> li = adminService.showHalls();
        if(li == null) {
            return "admin_home_page";
        }
        map.put("hList",li);
        if(!flag) {
            session.setAttribute("err_mssgHall", "Error while Ad
            return "admin_hall";
        }
        session.setAttribute("err_mssgHall1", "Hall Added Succes
        return "admin_hall";
    }
}
```