INFO 6250 Final Project Report

Xuhang Liu
001434973

Project Statement:

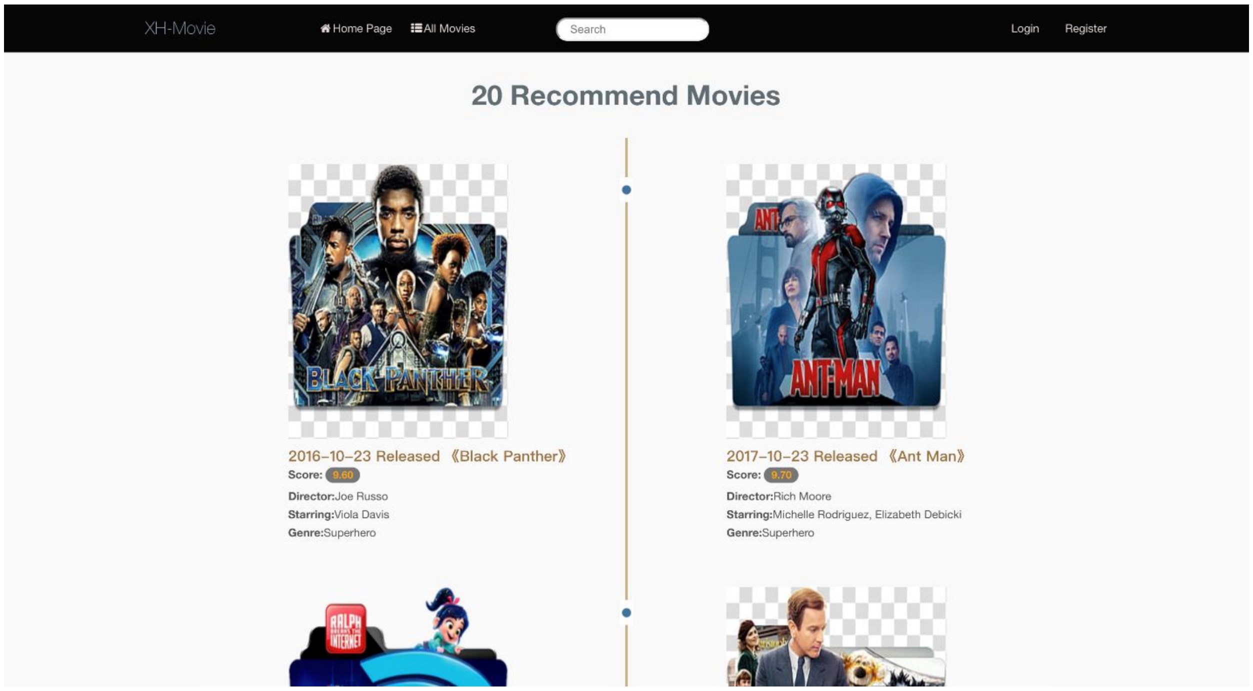
Based on the INFO6150 Web design class's final project. I build a web application for movies view and collection using Spring MVC and Hibernate with MYSQL database and Bootstrap.
Since now days, more and more movies come out on the market which makes us hard to select the right movie to watch without wasting time in the theater for more than 2 hours.
Also, people like to discuss all the movies they watched and liked, but sometimes they need something to remind them of the movies they liked or watched before.

Functionalities:

1. Homepage with recommend movies.
2. User can browser all the movies with implement sort by hot, sort by data, sort by score, list with year, area, and genre using criteria with restrictions.
3. Users can click on the movie thumbnail to view the individual movie details.
4. Users can search for movies by keyword.
5. User will have to register and login to review and collect the movies with interceptor pre-handle method.
6. After user register and provide their interested movie type, it will automatically find related movies and recommend the user on the home page.
7. All the movies stored with related comments for user to see.
8. Users can login to manage their own user information and their movie collections.
9. Admin user can login and only admin user can manage all the users and all the movie information with interceptor separating user types.
10. Used MD5 API to encrypt the user id and password.
11. Prevent SQL statement injection attacks using filters with XSS-Filter and named parameters.

12. Used Bootstrap with paginator method to implement pagination.

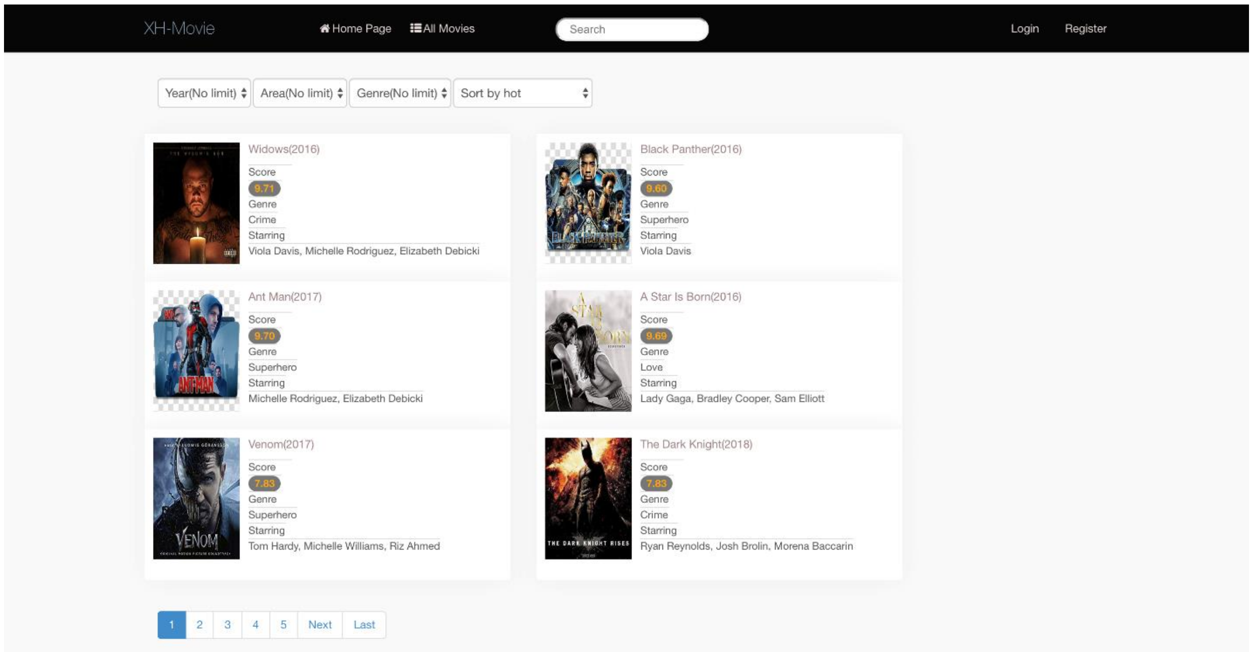
Screen Shots:



20 Recommend Movies

2016–10–23 Released 《Black Panther》
Score: 9.60
Director:Joe Russo
Starring:Viola Davis
Genre:Superhero

2017–10–23 Released 《Ant Man》
Score: 9.70
Director:Rich Moore
Starring:Michelle Rodriguez, Elizabeth Debicki
Genre:Superhero



Widows(2016)
Score
9.71
Genre
Crime
Starring
Viola Davis, Michelle Rodriguez, Elizabeth Debicki

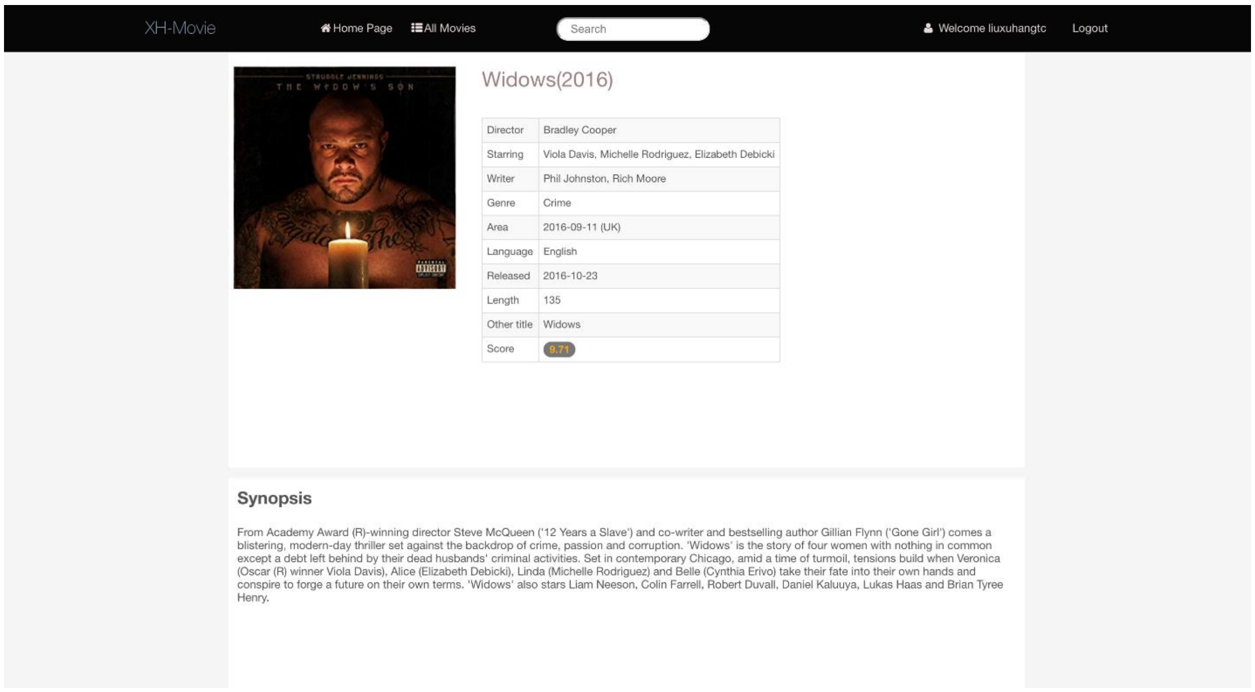
Black Panther(2016)
Score
9.60
Genre
Superhero
Starring
Viola Davis

Ant Man(2017)
Score
9.70
Genre
Superhero
Starring
Michelle Rodriguez, Elizabeth Debicki

A Star Is Born(2016)
Score
9.69
Genre
Love
Starring
Lady Gaga, Bradley Cooper, Sam Elliott

Venom(2017)
Score
7.83
Genre
Superhero
Starring
Tom Hardy, Michelle Williams, Riz Ahmed

The Dark Knight(2018)
Score
7.83
Genre
Crime
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin

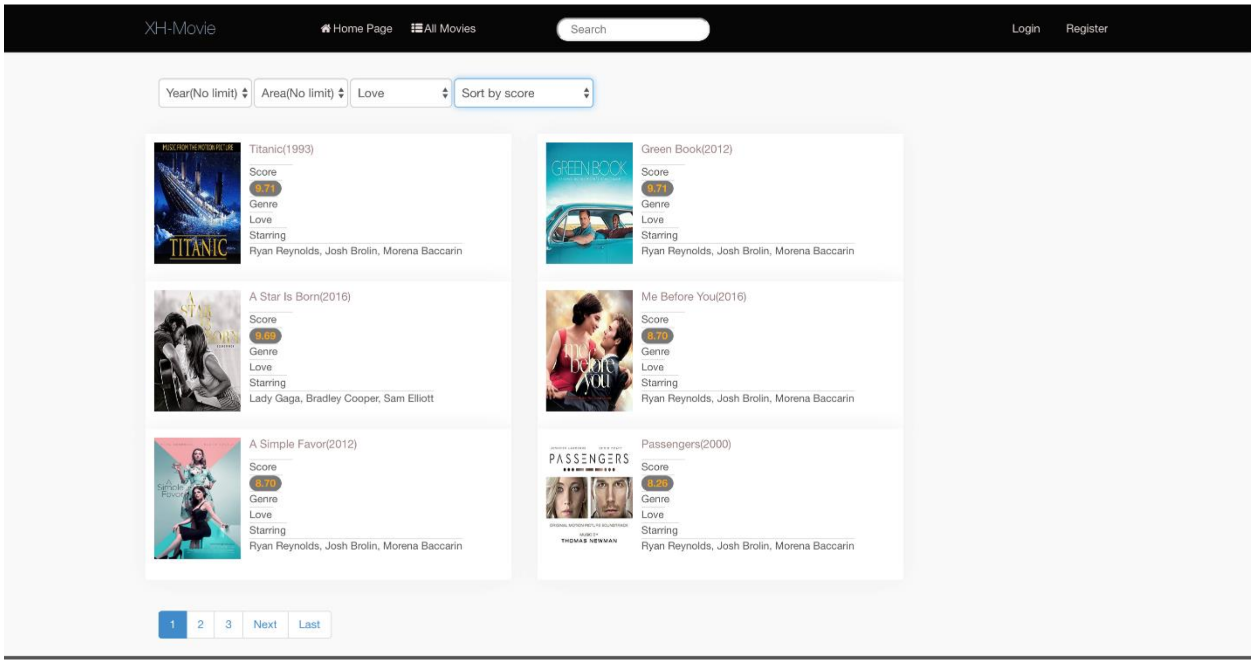

XH-Movie Home Page All Movies Search Welcome liuxuhangtc Logout

Widows(2016)

Director	Bradley Cooper
Starring	Viola Davis, Michelle Rodriguez, Elizabeth Debicki
Writer	Phil Johnston, Rich Moore
Genre	Crime
Area	2016-09-11 (UK)
Language	English
Released	2016-10-23
Length	135
Other title	Widows
Score	9.71

Synopsis

From Academy Award (R)-winning director Steve McQueen ('12 Years a Slave') and co-writer and bestselling author Gillian Flynn ('Gone Girl') comes a blistering, modern-day thriller set against the backdrop of crime, passion and corruption. 'Widows' is the story of four women with nothing in common except a debt left behind by their dead husbands' criminal activities. Set in contemporary Chicago, amid a time of turmoil, tensions build when Veronica (Oscar (R) winner Viola Davis), Alice (Elizabeth Debicki), Linda (Michelle Rodriguez) and Belle (Cynthia Erivo) take their fate into their own hands and conspire to forge a future on their own terms. 'Widows' also stars Liam Neeson, Colin Farrell, Robert Duvall, Daniel Kaluuya, Lukas Haas and Brian Tyree Henry.


XH-Movie Home Page All Movies Search Login Register

Year(No limit) Area(No limit) Love Sort by score

Titanic(1993)
Score
9.71
Genre
Love
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin

Green Book(2012)
Score
9.71
Genre
Love
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin

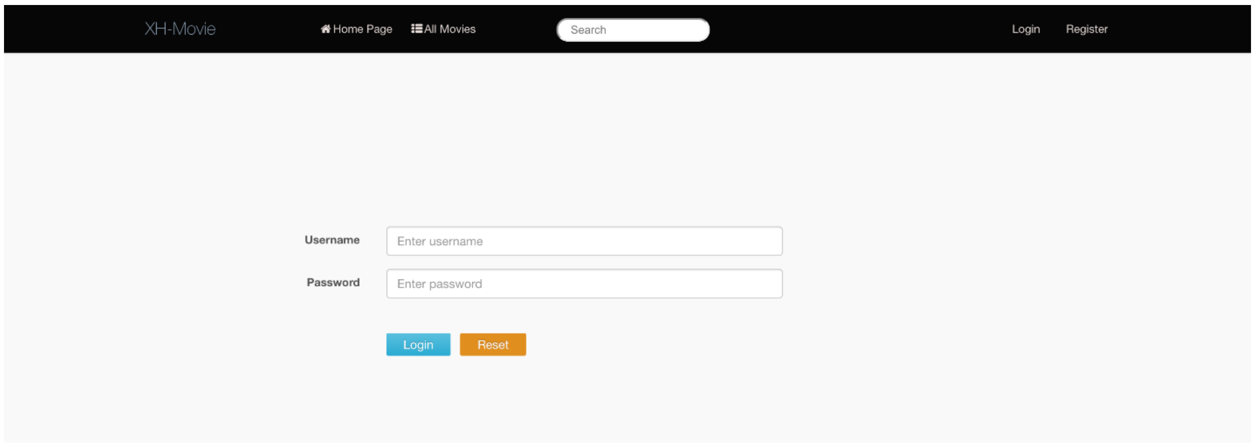
A Star Is Born(2016)
Score
9.69
Genre
Love
Starring
Lady Gaga, Bradley Cooper, Sam Elliott

Me Before You(2016)
Score
8.70
Genre
Love
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin

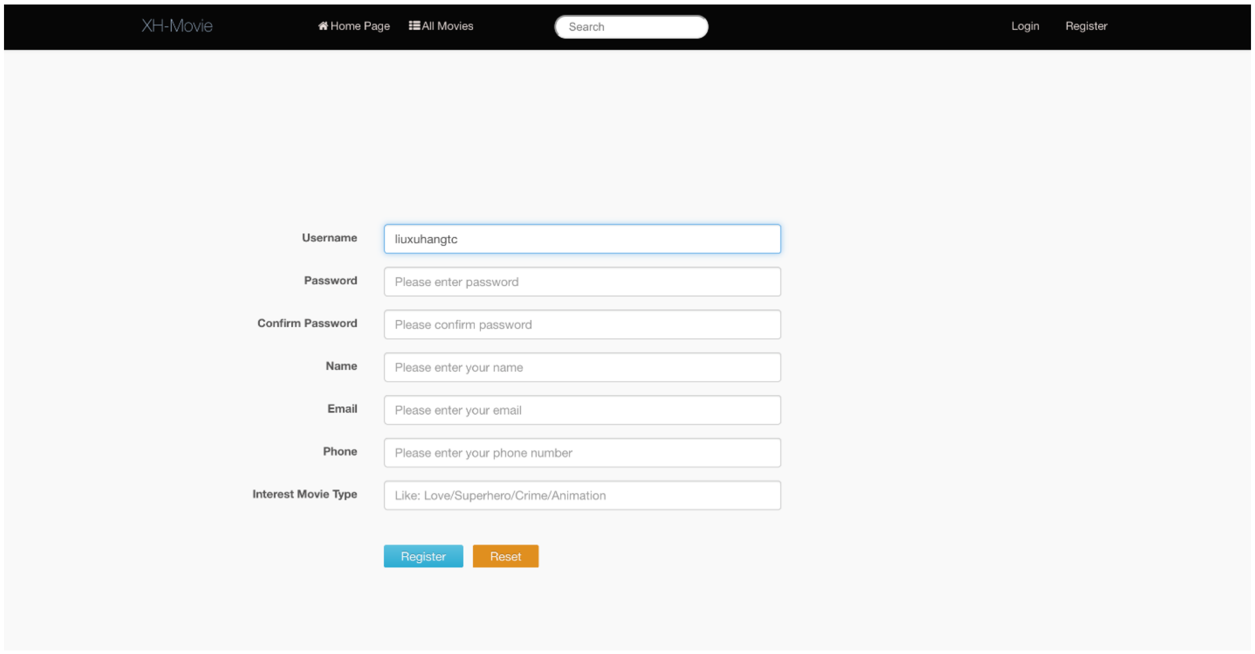
A Simple Favor(2012)
Score
8.70
Genre
Love
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin

Passengers(2000)
Score
8.26
Genre
Love
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin

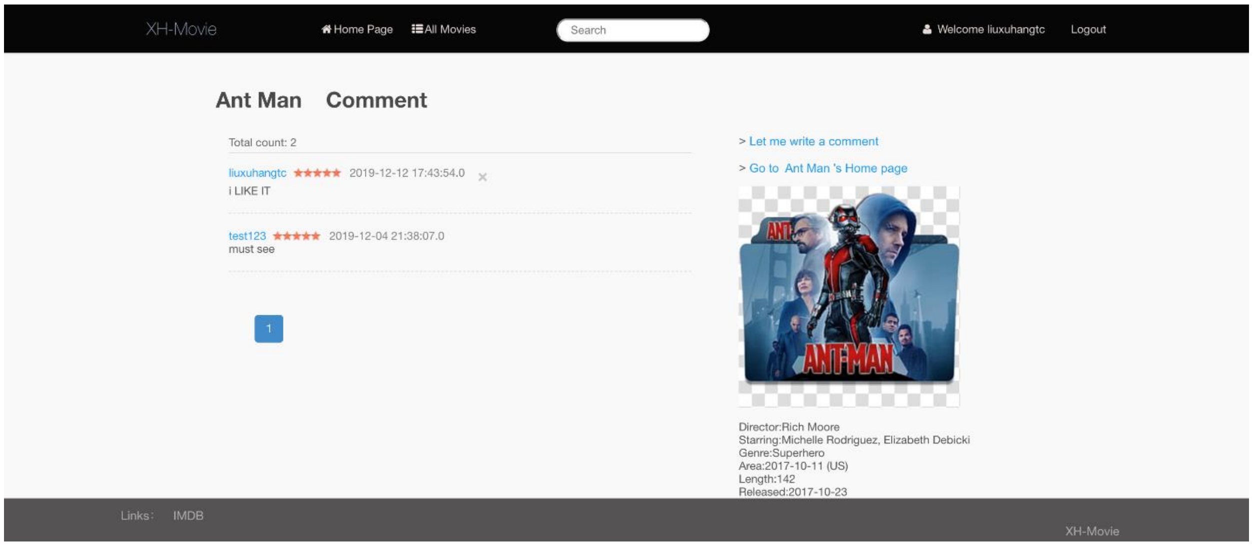
1 2 3 Next Last



Username	Enter username
Password	Enter password



Username	liuxuhangtc
Password	Please enter password
Confirm Password	Please confirm password
Name	Please enter your name
Email	Please enter your email
Phone	Please enter your phone number
Interest Movie Type	Like: Love/Superhero/Crime/Animation



Ant Man Comment

Total count: 2

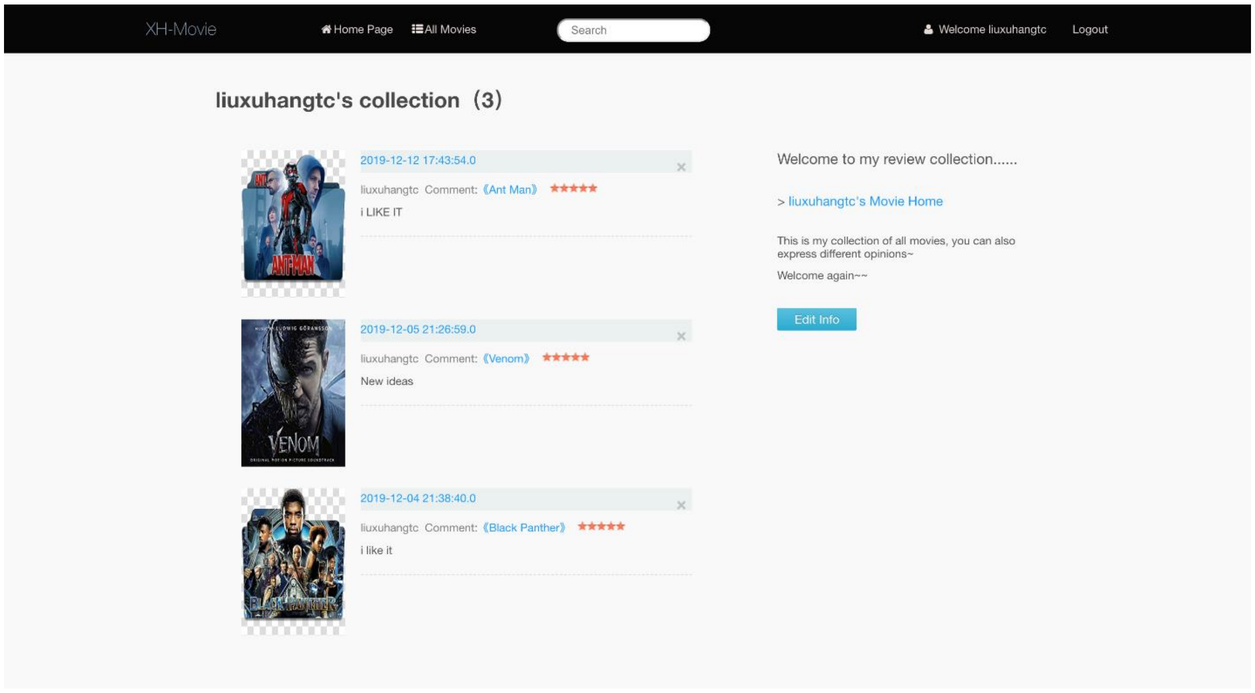
liuxuhangtc ★★★★★ 2019-12-12 17:43:54.0 ×
i LIKE IT

test123 ★★★★★ 2019-12-04 21:38:07.0
must see

> Let me write a comment

> Go to Ant Man 's Home page

Director:Rich Moore
Starring:Michelle Rodriguez, Elizabeth Debicki
Genre:Superhero
Area:2017-10-11 (US)
Length:142
Released:2017-10-23



XH-Movie Home Page All Movies Search Welcome liuxuhangtc Logout

liuxuhangtc's collection（3）

2019-12-12 17:43:54.0
liuxuhangtc Comment: 《Ant Man》 ★★★★★
i LIKE IT

2019-12-05 21:26:59.0
liuxuhangtc Comment: 《Venom》 ★★★★★
New ideas

2019-12-04 21:38:40.0
liuxuhangtc Comment: 《Black Panther》 ★★★★★
i like it

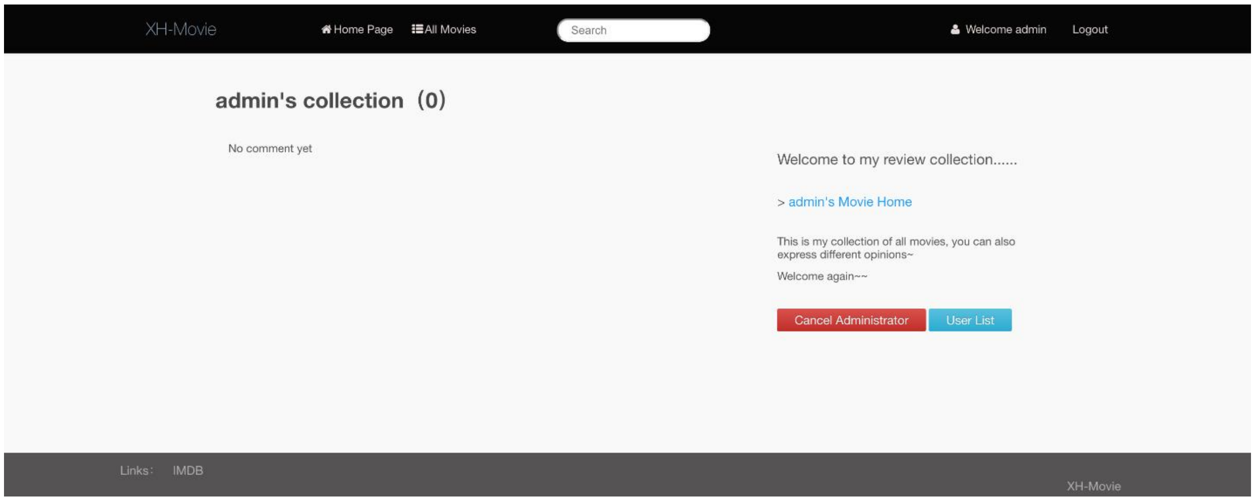
Welcome to my review collection......

> liuxuhangtc's Movie Home

This is my collection of all movies, you can also express different opinions~

Welcome again~~

Edit Info



XH-Movie Home Page All Movies Search Welcome admin Logout

admin's collection（0）

No comment yet

Welcome to my review collection......

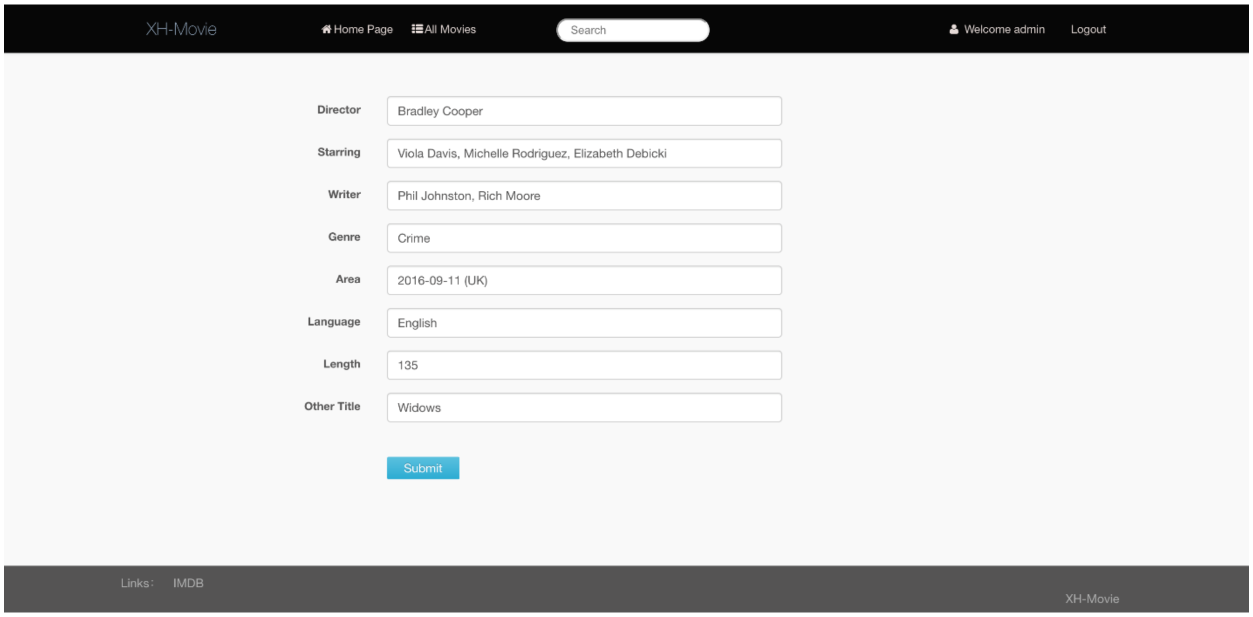
> admin's Movie Home

This is my collection of all movies, you can also express different opinions~

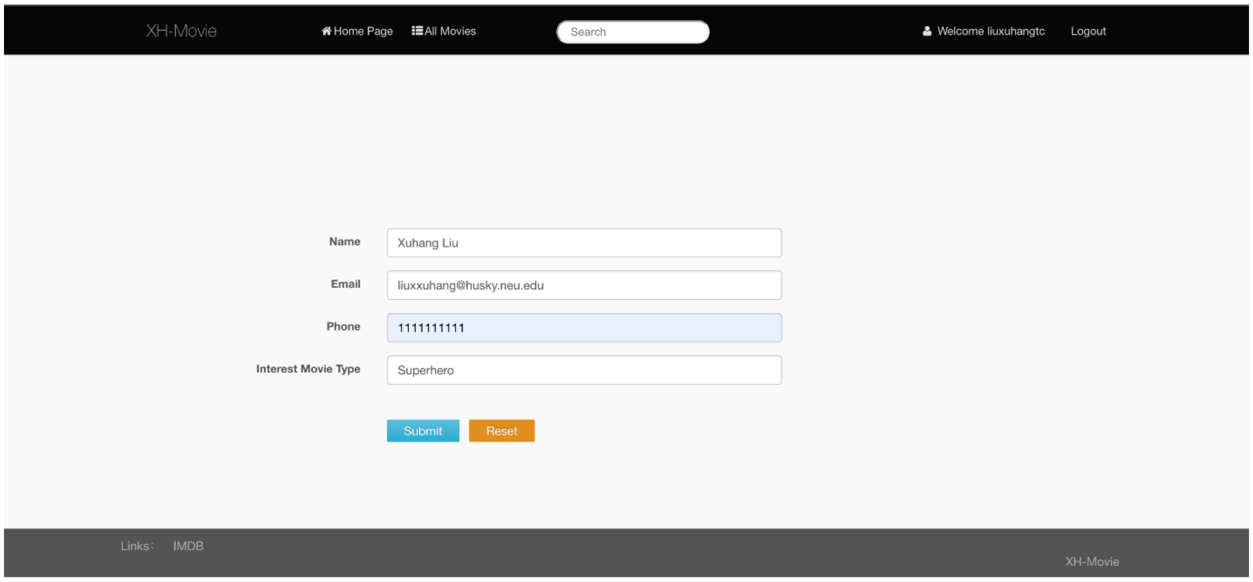
Welcome again~~

Cancel Administrator User List

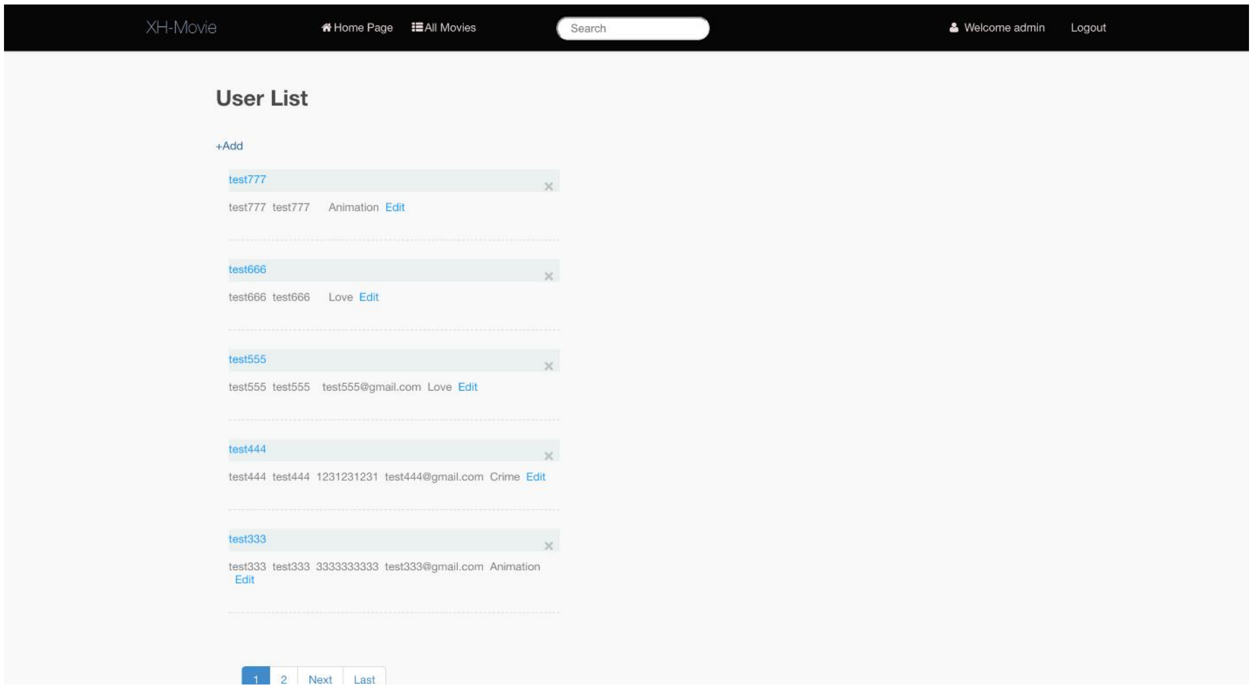
Links： IMDB XH-Movie



Director	Bradley Cooper
Starring	Viola Davis, Michelle Rodriguez, Elizabeth Debicki
Writer	Phil Johnston, Rich Moore
Genre	Crime
Area	2016-09-11 (UK)
Language	English
Length	135
Other Title	Widows



Name	Xuhang Liu
Email	liuxxuhang@husky.neu.edu
Phone	1111111111
Interest Movie Type	Superhero



User List

+Add

test777

test777 test777 Animation Edit

test666

test666 test666 Love Edit

test555

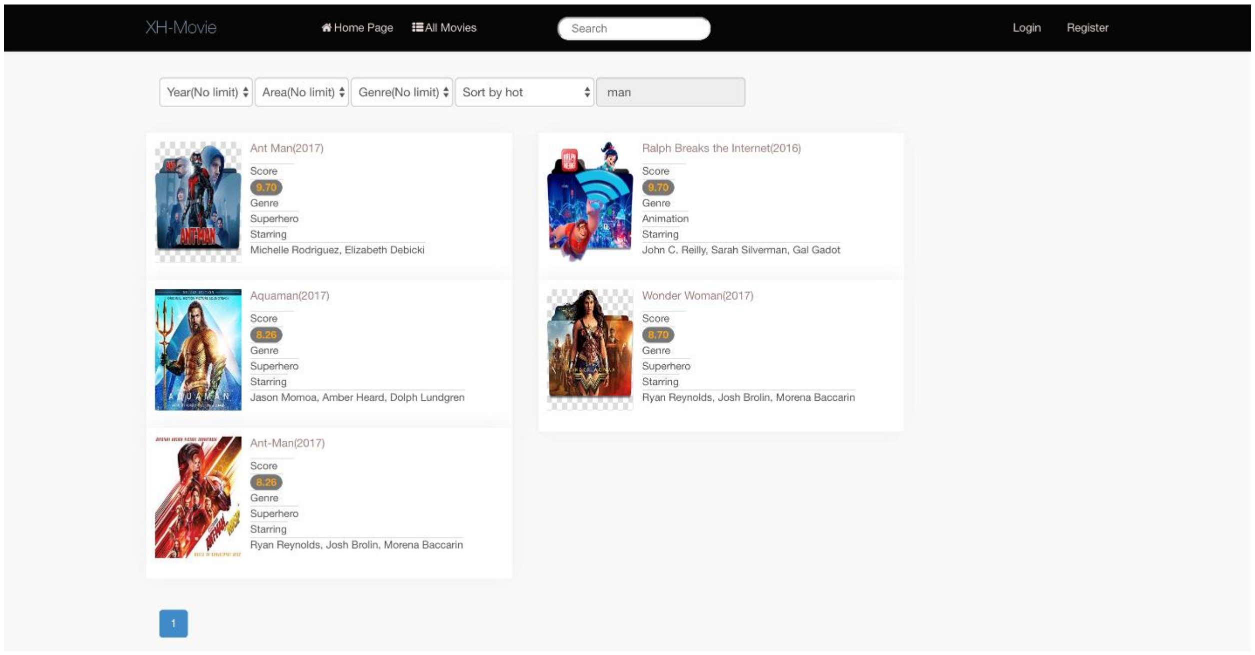
test555 test555 test555@gmail.com Love Edit

test444

test444 test444 1231231231 test444@gmail.com Crime Edit

test333

test333 test333 3333333333 test333@gmail.com Animation Edit



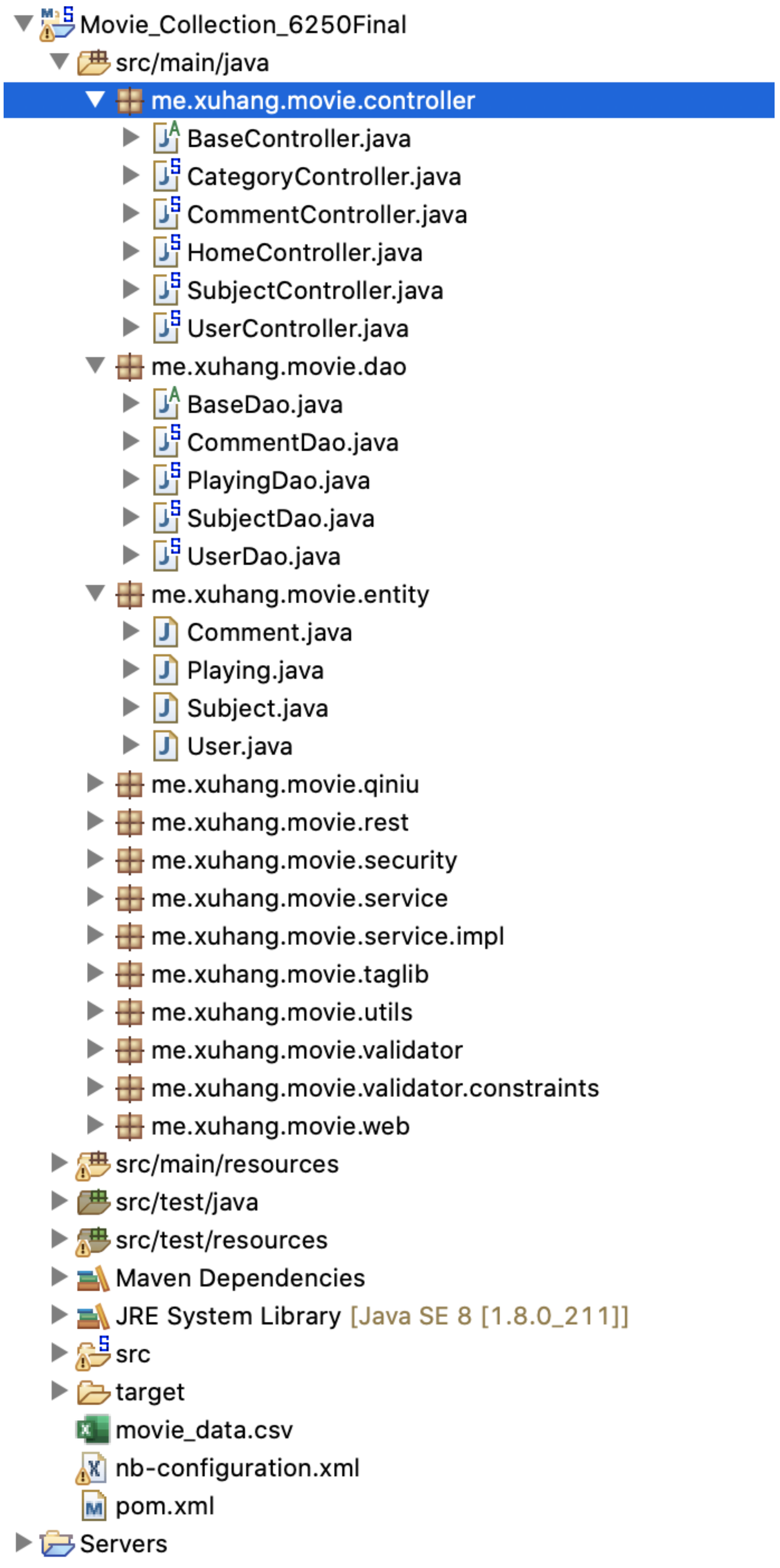
Ant Man(2017)
Score
9.70
Genre
Superhero
Starring
Michelle Rodriguez, Elizabeth Debicki

Ralph Breaks the Internet(2016)
Score
9.70
Genre
Animation
Starring
John C. Reilly, Sarah Silverman, Gal Gadot

Aquaman(2017)
Score
8.26
Genre
Superhero
Starring
Jason Momoa, Amber Heard, Dolph Lundgren

Wonder Woman(2017)
Score
8.70
Genre
Superhero
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin

Ant-Man(2017)
Score
8.26
Genre
Superhero
Starring
Ryan Reynolds, Josh Brolin, Morena Baccarin



```
▼ Movie_Collection_6250Final
  ▼ src/main/java
    ▼ me.xuhang.movie.controller
      ▶ BaseController.java
      ▶ CategoryController.java
      ▶ CommentController.java
      ▶ HomeController.java
      ▶ SubjectController.java
      ▶ UserController.java
    ▼ me.xuhang.movie.dao
      ▶ BaseDao.java
      ▶ CommentDao.java
      ▶ PlayingDao.java
      ▶ SubjectDao.java
      ▶ UserDao.java
    ▼ me.xuhang.movie.entity
      ▶ Comment.java
      ▶ Playing.java
      ▶ Subject.java
      ▶ User.java
    ▶ me.xuhang.movie.qiniu
    ▶ me.xuhang.movie.rest
    ▶ me.xuhang.movie.security
    ▶ me.xuhang.movie.service
    ▶ me.xuhang.movie.service.impl
    ▶ me.xuhang.movie.taglib
    ▶ me.xuhang.movie.utils
    ▶ me.xuhang.movie.validator
    ▶ me.xuhang.movie.validator.constraints
    ▶ me.xuhang.movie.web
  ▶ src/main/resources
  ▶ src/test/java
  ▶ src/test/resources
  ▶ Maven Dependencies
  ▶ JRE System Library [Java SE 8 [1.8.0_211]]
  ▶ src
  ▶ target
    movie_data.csv
    nb-configuration.xml
    pom.xml
▶ Servers
```

Controller source codes:
Base Controller

```java
package me.xuhang.movie.controller;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.propertyeditors.CustomDateEditor;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;

import me.xuhang.movie.validator.ValidatorWrapper;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.validation.Validation;
import javax.validation.Validator;
import javax.validation.ValidatorFactory;
import java.text.SimpleDateFormat;
import java.util.Date;

/**
 * Created by xuhang on 2019/11/20.
 */
public abstract class BaseController {

    @Autowired
    protected HttpServletRequest request;

    protected HttpServletResponse response;

    protected ValidatorFactory factory = 
Validation.buildDefaultValidatorFactory();
    protected Validator validator = factory.getValidator();
    private ValidatorWrapper validatorWrapper;

    protected Logger logger = null;

    public BaseController() {
        logger = LoggerFactory.getLogger(getClass().getName());
    }

    protected ValidatorWrapper getValidatorWrapper() {
        if (validatorWrapper == null) {
            validatorWrapper = new ValidatorWrapper(validator);
        }
        return validatorWrapper;
    }

    @InitBinder
    public void initBinder(WebDataBinder binder) {
        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
        dateFormat.setLenient(false);
        binder.registerCustomEditor(Date.class, new CustomDateEditor(dateFormat, 
true));
    }
}
```

Category Controller

```
package me.xuhang.movie.controller;

import me.xuhang.movie.entity.Subject;
import me.xuhang.movie.rest.PageInfo;
import me.xuhang.movie.service.DoubanService;
import me.xuhang.movie.service.SubjectService;

import org.apache.commons.lang3.StringUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.ModelAndView;

/**
 * Created by xuhang on 2019/11/20.
 */
@Controller
@RequestMapping("/category")
public class CategoryController extends BaseController {

    @Autowired
    private SubjectService subjectService;

    @Autowired
    private DoubanService doubanService;

    @RequestMapping({ "/index", "" })
    public ModelAndView index(String key) {
        ModelAndView modelAndView = new ModelAndView("/category/list");
        if (StringUtils.isNotEmpty(key)) {
            key = key.trim();
            modelAndView.addObject("key", key);
            doubanService.saveBySearch(key);
        }
        return modelAndView;
    }

    @RequestMapping("/list")
    @ResponseBody
    public PageInfo<Subject> list(int pageNo, String year, String place, String
type, String sort, String key) {
        return subjectService.listBySearch(pageNo, 6, year, place, type, sort,
key);
    }

}
```

Comment Controller

```
package me.xuhang.movie.controller;

import me.xuhang.movie.dao.CommentDao;
import me.xuhang.movie.dao.SubjectDao;
import me.xuhang.movie.entity.Comment;
import me.xuhang.movie.entity.Subject;
import me.xuhang.movie.entity.User;
import me.xuhang.movie.rest.RestData;
import me.xuhang.movie.utils.ContextUtils;

import org.apache.commons.lang3.StringUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.util.HtmlUtils;

import java.util.Date;

/**
 * Created by xuhang on 2019/11/20.
 */
@Controller
```

```java
@RequestMapping("/comment")
public class CommentController extends BaseController {

    @Autowired
    private CommentDao commentDao;

    @Autowired
    private SubjectDao subjectDao;

    @RequestMapping(value = "/post", method = RequestMethod.POST)
    public ModelAndView getComments(Comment comment) {
        ModelAndView modelAndView = new ModelAndView("redirect:/subject/" +
comment.getSubjectId());
        if (StringUtils.isEmpty(ContextUtils.getUserId(request))) {
            modelAndView.addObject("error", "Login to Review and Collect");
            return modelAndView;
        }
        comment.setUserId(ContextUtils.getUserId(request));
        comment.setSubmitDate(new Date());
        comment.setContent(HtmlUtils.htmlEscape(comment.getContent()));
        commentDao.create(comment);
        Subject subject = subjectDao.find(comment.getSubjectId());
        subject.setTotalRating(subject.getTotalRating() + comment.getRating());
        subject.setRatingCount(subject.getRatingCount() + 1);
        subject.setCommentCount(subject.getCommentCount() + 1);
        subjectDao.update(subject);
        return modelAndView;
    }

    @RequestMapping(value = "/delete", method = RequestMethod.POST)
    @ResponseBody
    public RestData deleteComment(String id) {
        Comment comment = commentDao.find(id);
        RestData restData = new RestData();
        if (null == comment) {
            restData.setComment("This collection not exist! ");
            return restData;
        }
        User currentUser = ContextUtils.getUser(request);
        if (null == currentUser) {
            restData.setComment("Please login first! ");
            return restData;
        }
        if (currentUser.isAdmin() ||
currentUser.getId().equals(comment.getUserId())) {
```

```
            Subject subject = subjectDao.find(comment.getSubjectId());
            subject.setCommentCount(subject.getCommentCount() - 1);
            subject.setTotalRating(subject.getTotalRating() -
comment.getRating());
            subject.setRatingCount(subject.getRatingCount() - 1);
            subjectDao.update(subject);
            commentDao.delete(comment);
            restData.setSuccess(1);
            return restData;

        } else {
            restData.setComment(
                    "You are not the author or moderator of this comment
collection, you do not have permission to delete this comment collection.");
            return restData;
        }
    }
}
```

Home Controller

```
package me.xuhang.movie.controller;

import me.xuhang.movie.dao.UserDao;
import me.xuhang.movie.entity.Subject;
import me.xuhang.movie.entity.User;
import me.xuhang.movie.rest.RestData;
import me.xuhang.movie.security.LoginRequired;
import me.xuhang.movie.service.DoubanService;
import me.xuhang.movie.service.UserService;
import me.xuhang.movie.utils.ContextUtils;
import me.xuhang.movie.utils.WebUtils;
import me.xuhang.movie.validator.InvalidException;

import org.apache.commons.codec.digest.DigestUtils;
import org.apache.commons.lang3.StringUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.ModelAndView;

import java.util.ArrayList;
```

```java
import java.util.Date;
import java.util.List;

/**
 * Created by xuhang on 2019/11/20.
 */
@SuppressWarnings("unused")
@Controller
@RequestMapping("/")
public class HomeController extends BaseController {

    @Autowired
    private UserService userService;

    @Autowired
    private DoubanService doubanService;

    @Autowired
    private UserDao userDao;

    @RequestMapping({ "/index", "/" })
    public ModelAndView index() {
        ModelAndView modelAndView = new ModelAndView("/index");
        List<Subject> list = doubanService.findPlaying();
        List<Subject> resule = new ArrayList<Subject>();
        User user = ContextUtils.getUser(request);
        if (user != null) {
            String aihao = user.getAihao();
            for (Subject subject : list) {
                if (subject.getGenres().contains(aihao)) {
                    resule.add(subject);
                }
            }
        } else {
            resule.addAll(list);
        }
        modelAndView.addObject("subjects", resule);
        return modelAndView;
    }

    @RequestMapping(value = "/register", method = RequestMethod.GET)
    public String getRegister() {
        return "/register";
    }
    public static void main(String[] ages) {
        System.out.println(DigestUtils.md5Hex("123"));
    }

    @RequestMapping(value = "/register", method = RequestMethod.POST)
    public ModelAndView register(User user, String captcha) {
        ModelAndView modelAndView = new ModelAndView("/register");

        logger.debug("user:{}", user);
        user.setPassword(DigestUtils.md5Hex(user.getPassword()));
        user.setCreateTime(new Date());
        try {
            getValidatorWrapper().tryValidate(user);
            userService.create(user);
        } catch (InvalidException ex) {
            logger.error("Invalid User Object: {}", user.toString(), ex);
            modelAndView.addObject("error", ex.getMessage());
            return modelAndView;
        }

        if (ContextUtils.getSessionUtils(request).getUser() == null) {
            return new ModelAndView("redirect:/index");
        } else if (ContextUtils.getSessionUtils(request).getUser().isAdmin()) {
            return new ModelAndView("redirect:/user/list");
        } else {
            return new ModelAndView("redirect:/index");
        }
    }

    @RequestMapping(value = "/checkUserName", method = RequestMethod.GET)
    @ResponseBody
    public RestData checkUserName(String userName) {
        RestData restData = new RestData();
        if (userDao.checkUserName(userName)) {
            restData.setSuccess(1);
        } else {
            restData.setComment("Username already in use");
        }
        return restData;
    }

    @RequestMapping(value = "/login", method = RequestMethod.GET)
    public ModelAndView getLogin(String redirect) {
        ModelAndView modelAndView = new ModelAndView("/login");
        if (StringUtils.isNotEmpty(redirect)) {
            modelAndView.addObject("error", "Please login first!");
        }
        return modelAndView;
    }

    @RequestMapping(value = "/login", method = RequestMethod.POST)
    public ModelAndView login(String userName, String password, String captcha) {
        ModelAndView modelAndView = new ModelAndView("/login");

        User user = userDao.findByUserName(userName);

        if (null != user &&
user.getPassword().equals(DigestUtils.md5Hex(password))) {
            ContextUtils.getSessionUtils(request).setUser(user);
            return new ModelAndView("redirect:/");
        } else {
            modelAndView.addObject("error", "Username or password incorrect");
            return modelAndView;
        }
    }

    @RequestMapping(value = "/password", method = RequestMethod.GET)
    public String getPassword() {
        return "/password";
    }

    @RequestMapping(value = "/password", method = RequestMethod.POST)
    @LoginRequired
    public ModelAndView postPassword(String oldPassword, String newPassword,
String captcha) {
        ModelAndView modelAndView = new ModelAndView("/password");

        User user = ContextUtils.getUser(request);
        if (user.getPassword().equals(DigestUtils.md5Hex(oldPassword))) {
            user.setPassword(DigestUtils.md5Hex(newPassword));
            userDao.update(user);
            return new ModelAndView("redirect:/");
        } else {
            modelAndView.addObject("error", "Old password incorrect");
        }
        return modelAndView;

    }

    @RequestMapping(value = "/editUser", method = RequestMethod.GET)
    public ModelAndView editUser(String id) {
        ModelAndView modelAndView = new ModelAndView("/editUser");
        User user = userDao.findByUserId(id);
        modelAndView.addObject("user", user);
        return modelAndView;
    }

    @RequestMapping(value = "/editUser", method = RequestMethod.POST)
    @LoginRequired
    public ModelAndView postPassword(String id, String aihao, String email,
String name, String phone) {
        ModelAndView modelAndView = new ModelAndView("/editUser");

        User user = userDao.findByUserId(id);
        if (user != null) {
            user.setAihao(aihao);
            user.setEmail(email);
            user.setName(name);
            user.setPhone(phone);
            userDao.update(user);
            if (ContextUtils.getSessionUtils(request).getUser().isAdmin()) {
                return new ModelAndView("redirect:/user/list");
            } else {
                return new ModelAndView("redirect:/user/" +
ContextUtils.getSessionUtils(request).getUser().getId());
            }
        } else {
            modelAndView.addObject("error", "User data error");
        }
        return modelAndView;

    }

    @RequestMapping("logout")
    public String logout() {
        request.getSession().invalidate();
        return "redirect:/";
    }

    @RequestMapping("403")
    public String get403() {
        return "/misc/403";
    }

}
```

Subject Controller

```java
package me.xuhang.movie.controller;

import me.xuhang.movie.dao.CommentDao;
import me.xuhang.movie.dao.SubjectDao;
import me.xuhang.movie.entity.Comment;
import me.xuhang.movie.entity.Subject;
import me.xuhang.movie.qiniu.QiniuUtils;
import me.xuhang.movie.rest.PageInfo;
import me.xuhang.movie.security.AdminRequired;
import me.xuhang.movie.service.DoubanService;

import org.apache.commons.lang3.StringUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;

/**
 * Created by xuhang on 2019/11/20.
 */
@Controller
@RequestMapping("/subject")
public class SubjectController extends BaseController {

    @Autowired
    private DoubanService doubanService;

    @Autowired
    private CommentDao commentDao;

    @Autowired
    private SubjectDao subjectDao;

    @RequestMapping("/{id}")
    public ModelAndView getSubject(@PathVariable(value = "id") String id, String
error) {
        ModelAndView modelAndView = new ModelAndView("/subject/details");
        Subject subject = doubanService.find(id);
        if (null == subject) {
            return new ModelAndView("/misc/404");
        }
        modelAndView.addObject("subject", subject);
        if (subject.getCommentCount() > 0) {
            modelAndView.addObject("comments", commentDao.listBySubjectId(id, 0, 
5));
        }
        if (StringUtils.isNotEmpty(error)) {
            modelAndView.addObject("error", error);
        }
        return modelAndView;
    }

    @RequestMapping("/{id}/comments")
    public ModelAndView getComments(@PathVariable(value = "id") String id,
            @RequestParam(value = "pageNo", defaultValue = "1") int pageNo) {
        ModelAndView modelAndView = new ModelAndView("/subject/comments");
        Subject subject = doubanService.find(id);
        modelAndView.addObject("subject", subject);
        if (null == subject) {
            return new ModelAndView("/misc/404");
        }
        PageInfo<Comment> pageInfo = new PageInfo<Comment>(pageNo, 5);
        pageInfo.setTotalRows(subject.getCommentCount());
        if (subject.getCommentCount() > 0) {
            pageInfo.setResultList(commentDao.listBySubjectId(id, 
pageInfo.getStartRow(), 5));
        }
        modelAndView.addObject("pageInfo", pageInfo);
        return modelAndView;
    }

    @RequestMapping("/{id}/edit")
    @AdminRequired
    public ModelAndView getEdit(@PathVariable(value = "id") String id) {
        ModelAndView modelAndView = new ModelAndView("/subject/edit");
        Subject subject = doubanService.find(id);
        modelAndView.addObject("subject", subject);
        if (null == subject) {
            return new ModelAndView("/misc/404");
        }
        return modelAndView;
    }

    @RequestMapping("/{id}/update")
    @AdminRequired
```

```java
    public String getUpdate(@PathVariable(value = "id") String id, Subject
subject) {
        Subject storedSubject = doubanService.find(id);
        if (null == storedSubject) {
            return "/misc/404";
        }
        storedSubject.setDirectors(subject.getDirectors());
        storedSubject.setCasts(subject.getCasts());
        storedSubject.setWriters(subject.getWriters());
        storedSubject.setGenres(subject.getGenres());
        storedSubject.setCountries(subject.getCountries());
        storedSubject.setLanguages(subject.getLanguages());
        storedSubject.setDurations(subject.getDurations());
        storedSubject.setOriginalTitle(subject.getOriginalTitle());
        subjectDao.update(storedSubject);
        return "redirect:/subject/" + id;
    }

    @RequestMapping("/{id}/delete")
    @AdminRequired
    public String getDelete(@PathVariable(value = "id") String id) {
        Subject subject = doubanService.find(id);
        if (null == subject) {
            return "/misc/404";
        } else {
            subjectDao.delete(subject);
            QiniuUtils.deleteFromQiniu(id);
        }
        return "redirect:/";
    }

}
```

User Controller

```java
package me.xuhang.movie.controller;

import me.xuhang.movie.dao.CommentDao;
import me.xuhang.movie.dao.UserDao;
import me.xuhang.movie.entity.Comment;
import me.xuhang.movie.entity.Subject;
import me.xuhang.movie.entity.User;
import me.xuhang.movie.rest.PageInfo;
import me.xuhang.movie.rest.RestData;
import me.xuhang.movie.security.AdminRequired;
import me.xuhang.movie.utils.ContextUtils;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.ModelAndView;

/**
 * Created by xuhang on 2019/11/20.
 */
@SuppressWarnings("unused")
@Controller
@RequestMapping("/user")
public class UserController extends BaseController {

    @Autowired
    private CommentDao commentDao;

    @Autowired
    private UserDao userDao;

    @RequestMapping("/{id}")
    public ModelAndView getAccount(@PathVariable(value = "id") String id,
            @RequestParam(value = "pageNo", defaultValue = "1") int pageNo) {
        ModelAndView modelAndView = new ModelAndView("/user");
        User user = userDao.find(id);
        PageInfo<Comment> pageInfo = new PageInfo<Comment>(pageNo, 5);
        modelAndView.addObject("user", user);
        int commentCount = commentDao.countByUserId(id);
        modelAndView.addObject("commentCount", commentCount);
        if (commentCount > 0) {
            pageInfo.setResultList(commentDao.listByUserId(id,
pageInfo.getStartRow(), 5));
            pageInfo.setTotalRows(commentDao.countByUserId(id));
        }
        modelAndView.addObject("pageInfo", pageInfo);

        return modelAndView;
    }

    @RequestMapping("/{id}/setAdmin")
    @AdminRequired
    public String setAdmin(@PathVariable(value = "id") String id) {
        User user = userDao.find(id);
        if (null == user) {
            return "/misc/404";
        }
        if (user.isAdmin()) {
            user.setAdmin(false);
        } else {
            user.setAdmin(true);
        }
        userDao.update(user);
        return "redirect:/user/" + id;
    }

    @RequestMapping(value = "/delete", method = RequestMethod.POST)
    @ResponseBody
    public RestData deleteComment(String id) {
        User user = userDao.find(id);
        RestData restData = new RestData();
        if (null == user) {
            restData.setComment("user not exist! ");
            return restData;
        }
        User currentUser = ContextUtils.getUser(request);
        if (null == currentUser) {
            restData.setComment("Please login first! ");
            return restData;
        }
        if (currentUser != null && currentUser.isAdmin()) {
            userDao.delete(user);
            restData.setSuccess(1);
            return restData;

        } else {
            restData.setComment("Only admin can delete");
            return restData;
        }
    }

    @RequestMapping("/list")
    public ModelAndView getList(@RequestParam(value = "pageNo", defaultValue =
"1") int pageNo) {
        ModelAndView modelAndView = new ModelAndView("/userlist");
        PageInfo<User> pageInfo = new PageInfo<User>(pageNo, 5);
        pageInfo.setResultList(userDao.listByUserId(pageInfo.getStartRow(), 5));
        pageInfo.setTotalRows(userDao.countByUserId());
        modelAndView.addObject("pageInfo", pageInfo);
        return modelAndView;
    }
}
```