

Project Report For Web Tools And Methods

NUID-001873473

Dhruvil Shah



“HolidayMaker – A Website To Enhance Your Travel Experience “

Summary:

HolidayMaker is an online website that aids travelers to get information on travel destination from the available posts on the websites. The posts are nothing but details about a completed travel by a traveler who can also display a detail itinerary of his trip with exclusive information and pictures that makes the website attractive and informative for the masses. The website is useful for users to design their vacation with the help of past travel posted on the websites and based on the likes and comments can draw conclusions. It also helps travel bloggers to get a one stop application who wish to preach their experience to the masses. Travelers can like/dislike and comment on an active post which will be used by Admin for Dashboarding. Admin can delete a Post or even a User which will be notified to them. This is a consumer to consumer-based business model.

Roles:

- Travelers: are the major users of the application that can post a completed Travel/Trip or view active posts from other users.
- Admin: Admin is the entity that has full control of the application and can add or remove modules from the website and also delete Post or Users if irrelevant.

Entities:

- 1) Admin:
 - Admin Login
 - CRUD operations for User
 - CRUD operations for Posts
 - Add new modules
 - View Dashboards
- 2) Travelers:
 - Login
 - CRUD operations for traveler post
 - Like/Dislike/Rate a travel post
 - Answer/Comment a travel post

Technology Used:

Based on_Spring MVC and Hibernate.

Front End: HTML5, CSS3, SCSS, Spring Framework, Bootstrap4, JavaScript, JQuery, AJAX, JSP, EL, JSTL

Back End: Spring MVC Controllers and Hibernate, Advance JAVA, Junit Testing.

Database: MySQL

Key Functionalities:

- A Traveler can Add/Update/Delete a Travel Posts and explore other active posts in the website.
- Spring Annotations are used to develop controllers and routes.
- Hibernate Association used to maintain relationships between Users, Posts, Comments, Image List, Itinerary List, and Likes.
- Hibernate Criteria and HQL used for mapping and updating data in the MySQL database.
- Login/Register Functionality for User and Admin built.
- Interceptor used for preventing SQL injection and other input errors.
- Form Validation using JavaScript and HTML5 and Spring Annotations.
- Contact Us form will send an email to Admin with data asynchronously using AJAX.
- Emails sent to users on new registration using Apache Commons Email.
- Explore Destinations web page dynamically update the post using JSP EL and JSTL.
- Hash Maps used to display Pie Charts and Tables for admin dashboards.
- Admin can get detail information from the dashboards to perform tasks such as maintaining relevant posts, adding modules to meet users/posts expectations and deleting irrelevant users if any.
- Serving Static resources by adding resource handlers in Spring MVC.

Screenshots:

Login page

The screenshot shows a web application titled "HOLIDAY MAKER" in the top left corner. In the top right corner, there are three navigation links: "Home" (highlighted in blue), "Contact", and "About". The main content area features a white login card with a purple-to-blue gradient shadow. The card has the title "LOGIN" in bold. Below the title are two input fields: "Email" and "Password". The "Password" field includes a toggle icon (an eye) to show or hide the password. A large, rounded "LOGIN" button with a blue-to-purple gradient is positioned below the input fields. At the bottom of the card, there is a link that says "Don't have an account? Sign Up".

Registration Page:

Registration

Username

Password

FirstName

Eg: John

LastName

Eg: Snow

Email

xyz@gmail.com

Address

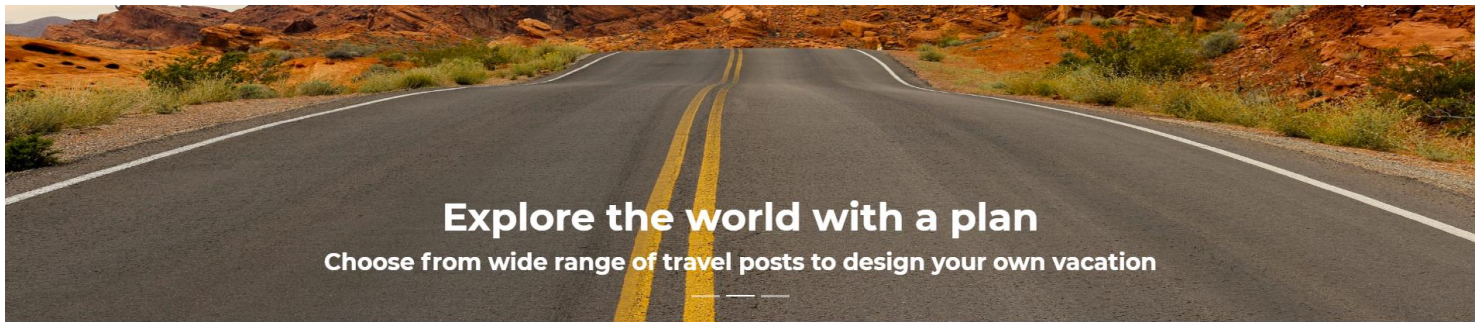
Eg:BOSTON

Home Page:



Travel Around the Globe

Share your Experience with the World



Explore the world with a plan

Choose from wide range of travel posts to design your own vacation

ABOUT



Holiday Maker is a website for travelers who love to share their trip experience and help people to design their own vacation from these experiences

You can add your travel post with details like itinerary and images which can be viewed by users all over the world to enhance their Holidays!

CONTACT US



Name

Email Address

Phone Number

Message

Send

Message

Send

LOCATION

1572 Tremont Street
Boston, MA 02120

AROUND THE WEB



ABOUT HOLIDAY MAKER

A Travel Website Enhancing Travel
Experiences of Users worldwide.

Copyright © HolidayMaker 2020

User Pages:

User homepage with header:



Travel Around the Globe

Share your Experience with the World

Add Post

Title for your Post

MesmerizingAmsterdam

Source Country

INDIA

Destination Country

AMS

Month of Your Visit

OCT

Itinerary

Day1

Enjoyed in AMS

Day2

Awesome View

Upload Images

Choose File pexels-photo-1903702.jpeg

Add Post

HOLIDAY MAKER

HomeExplore DestinationsAdd New Travel PostsView Your PostsDhruvilLOGOUT


Filter

Filter by By CountryAMS

Filter

Remove Filter

AMS




MesmerizingAmsterdam

Month of Visit: OCT

View Details

Likes 0

Israel




AmazingIsrael

Month of Visit: JUN

View Details

Likes 0

ITALY




ItalyReprise

Month of Visit: JUN

View Details

Likes 0

ROME




ROMAN

Month of Visit: SEP

View Details

Likes 0

DOHA




DohaExplore

Month of Visit: MAY

View Details

Likes 0

UAE




DesertSafari

Month of Visit: NOV

View Details

Likes 0

ROME




ROMAN

Month of Visit: SEP

View Details

Likes 0

DOHA




DohaExplore

Month of Visit: MAY

View Details

Likes 0

UAE



DesertSafari

Month of Visit: NOV

View Details

Likes 0

LOCATION

1572 Tremont Street
Boston, MA 02120

AROUND THE WEB


f

in

ABOUT HOLIDAY MAKER

A Travel Website Enhancing Travel
Experiences of Users worldwide.






MesmerizingAmsterdam

🕒 Post by Dhruvil Shah on 04-21-2020

County of Travel: AMS
Traveled From: INDIA
Month of Visit: OCT
some pexels-photo-1903702.jpeg

Itinerary

- ▶ DAY 1
- ▶ DAY 2



Likes 0

Leave a Comment:

Submit



Likes **0**

Leave a Comment:

Submit

1 Comments:

Awesome

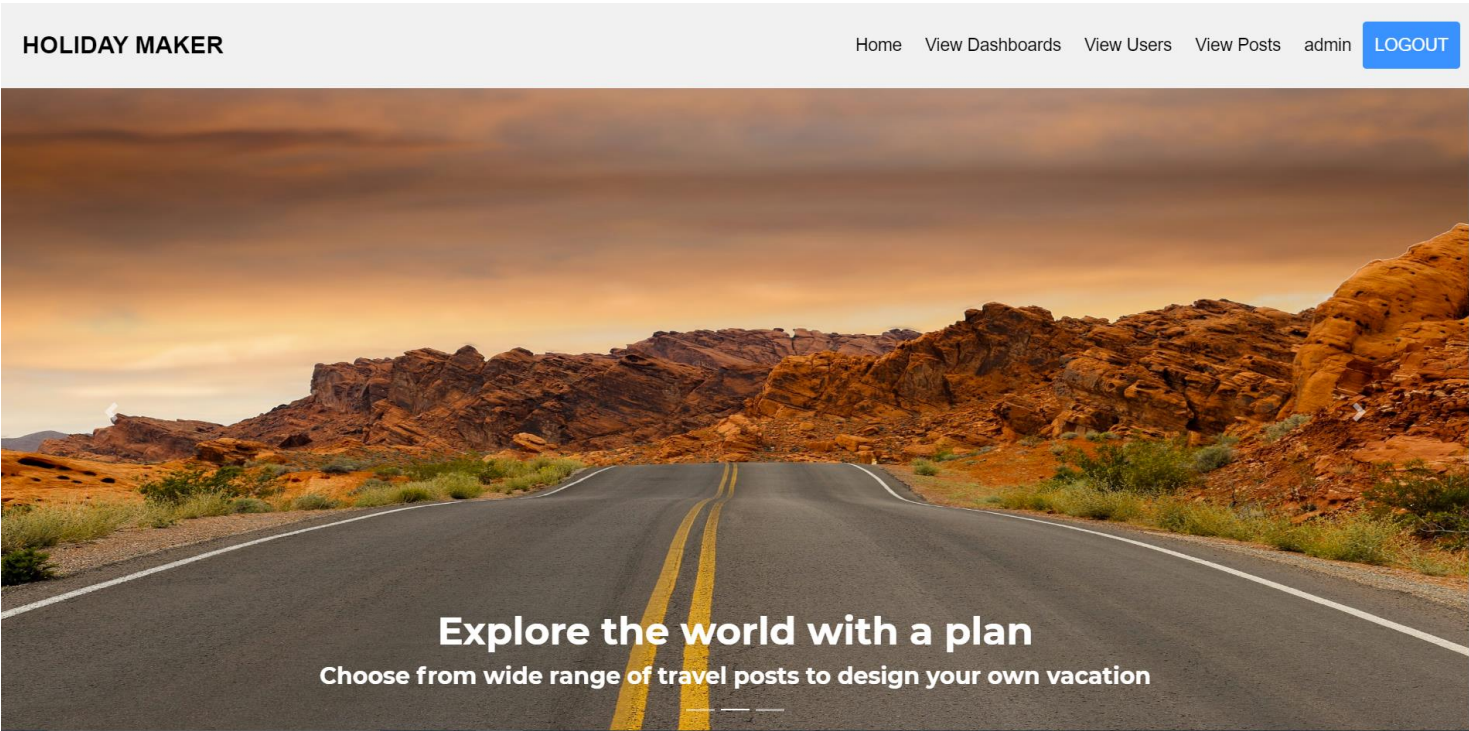
Posted by Dhruvil

View User Posts:

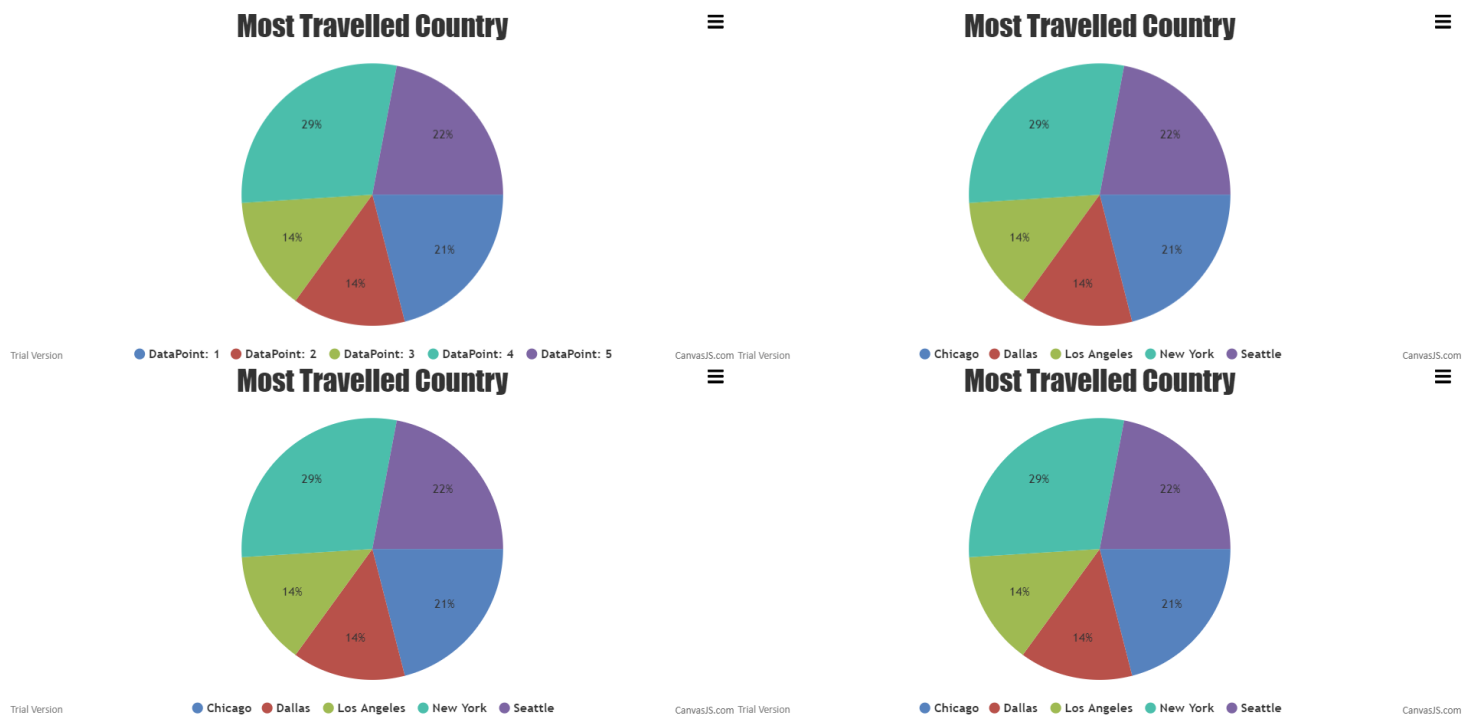
HOLIDAY MAKER									Home	Explore Destinations	Add New Travel Posts	View Your Posts	Dhruvil	LOGOUT
Title	CountryOfTravel	Source Country	MonthOfVisit	Last Updated Date	Likes	Comments	Delete Post	Update Post						
MesmerizingAmsterdam	AMS	INDIA	OCT	04-21-2020	0	1	Delete	Update						
AmazingIsrael	Israel	India	JUN	04-21-2020	0	0	Delete	Update						
ItalyReprise	ITALY	USA	JUN	04-21-2020	0	0	Delete	Update						
ROMAN	ROME	USA	SEP	04-21-2020	0	0	Delete	Update						
DohaExplore	DOHA	Usa	MAY	04-21-2020	0	0	Delete	Update						
DesertSafari	UAE	USA	NOV	04-21-2020	0	0	Delete	Update						

ADMIN PAGES:

Admin home page



View Dashboards page:



View Users page:

HOLIDAY MAKER

Home

Contact

About

User Dashboard

Username	Firstname	Lastname	Email	Phone	Number Of Posts	Comments given	Likes received by Users	Options
kunning	Dimpi	Dedhia	dedhiadimpi@gmail.com	8575882246	0	0	0	<div>Delete</div>
dns7665	Dhruvil	Shah	shah@gmail.com	8575882246	6	1	0	<div>Delete</div>

View All Posts Page:

HOLIDAY MAKER

Home

Contact

About

Posts Dashboard

Title	UserName	FullName	CountryOfTravel	MonthOfVisit	Last Updated Date	Likes	Comments	Options
MesmerizingAmsterdam	dns7665	Dhruvil Dhruvil	AMS	OCT	04-21-2020	0	1	<div>Delete</div>
AmazingIsrael	dns7665	Dhruvil Dhruvil	Israel	JUN	04-21-2020	0	0	<div>Delete</div>
ItalyReprise	dns7665	Dhruvil Dhruvil	ITALY	JUN	04-21-2020	0	0	<div>Delete</div>
ROMAN	dns7665	Dhruvil Dhruvil	ROME	SEP	04-21-2020	0	0	<div>Delete</div>
DohaExplore	dns7665	Dhruvil Dhruvil	DOHA	MAY	04-21-2020	0	0	<div>Delete</div>
DesertSafari	dns7665	Dhruvil Dhruvil	UAE	NOV	04-21-2020	0	0	<div>Delete</div>

Controllers:

ADMIN CONTROLLER

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

```
package com.dns.controller;
```

```
import com.dns.dao.CommentsDao;
import com.dns.dao.PostDao;
import com.dns.dao.UserDao;
import com.dns.pojo.Comments;
import com.dns.pojo.Post;
import com.dns.pojo.User;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.LinkedHashMap;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;
import javax.servlet.RequestDispatcher;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
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;
```

```
/**
 *
 * @author shahd
 */
```

```
@Configuration
@ComponentScan(basePackages="com.dns")
@Controller
```

```

public class AdminController {

    @RequestMapping(value = "/adminViewUsers", method = RequestMethod.GET)
    public ModelAndView viewUsers(HttpServletRequest hsr, HttpServletResponse hsr1) throws Exception {

        hsr1.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from the origin
server
        hsr1.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any circumstance
        hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
        hsr1.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility


        UserDao userDao = new UserDao();
        PostDao postDao = new PostDao();
        CommentsDao commentDao = new CommentsDao();

        List<User> userList = userDao getUsersForAdmin();


        //Calculate number of comments
        Map<User,Integer> commentCnt = new HashMap<User,Integer>();

        for( User user : userList){

            List<Comments> cList = commentDao.getCommentsByUser(user);

            commentCnt.put(user,cList.size() );

        }

        //Calculate number of likes

        Map<User,Integer> likeCnt = new HashMap<User,Integer>();

        for(User user : userList){

            int likeCount=0;

            List<Post> postList = postDao.getPostByUser(user);
            for(Post post : postList){

                likeCount=likeCount+post.getLikes().size();

            }

            likeCnt.put(user, likeCount);

```



```
}
```

```
ModelAndView mav = new ModelAndView("adminViewUsers");
```

```
mav.addObject("likeCnt",likeCnt);  
mav.addObject("commentCnt",commentCnt);  
mav.addObject("userList",userList);  
return mav;
```

```
}
```

```
@RequestMapping(value="/adminViewUsers/deleteUser/{id}", method=RequestMethod.GET )  
@ResponseBody  
public ModelAndView deleteUserPost(HttpServletRequest hsr, HttpServletResponse hsr1,  
@PathVariable("id") long id) throws Exception {  
  
    hsr1.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from the  
origin server  
    hsr1.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any circumstance  
    hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"  
    hsr1.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility
```

```
HttpSession session = hsr.getSession();  
    User userName = (User) session.getAttribute("User");  
if (userName==null) {  
    ModelAndView mav = new ModelAndView("logout");  
    return mav;  
}
```

```
//@RequestParam("id") String id  
System.out.println("ID " + id);  
System.out.println("inside delete");
```

```
UserDao userDao = new UserDao();  
int result = userDao.deleteUserById(id);  
System.out.println("result"+result);
```

```
ModelAndView mav = new ModelAndView("redirect:/adminViewUsers");

//MAPS

PostDao postDao = new PostDao();
CommentsDao commentDao = new CommentsDao();

List<User> userList = userDao getUsersForAdmin();


//Calculate number of comments
Map<User,Integer> commentCnt = new HashMap<User,Integer>();

for( User user : userList){

    List<Comments> cList = commentDao.getCommentsByUser(user);

    commentCnt.put(user,cList.size() );

}

//Calculate number of likes

Map<User,Integer> likeCnt = new HashMap<User,Integer>();

for(User user : userList){

    int likeCount=0;

    List<Post> postList = postDao.getPostByUser(user);
    for(Post post : postList){

        likeCount=likeCount+post.getLikes().size();

    }

    likeCnt.put(user, likeCount);

}

    mav.addObject("likeCnt",likeCnt);
    mav.addObject("commentCnt",commentCnt);
    mav.addObject("userList",userList);
    return mav;

}
```

```
@RequestMapping(value="/adminViewPosts/deletePost/{id}", method=RequestMethod.GET )
@ResponseBody
public ModelAndView deletePost(HttpServletRequest hsr, HttpServletResponse hsr1, @PathVariable("id")
long id) throws Exception {
```

```
    hsr1.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from the
origin server
    hsr1.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any circumstance
    hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
    hsr1.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility
```

```
    HttpSession session = hsr.getSession();
    User userName = (User) session.getAttribute("User");
    if (userName==null) {
        ModelAndView mav = new ModelAndView("logout");
        return mav;
    }
```

```
    //@RequestParam("id") String id
```

```
    ModelAndView mav = new ModelAndView("redirect:/adminPosts");
```

```
    //MAPS
```

```
    PostDao postDao = new PostDao();
    int result = postDao.deletePostById(id);
```

```
    List<Post> adminPostList = postDao.getPosts();
```

```
    mav.addObject("postList",adminPostList);
    return mav;
```

```
}
```

```
@RequestMapping(value = "/adminPosts", method = RequestMethod.GET)
public ModelAndView viewPosts(HttpServletRequest hsr, HttpServletResponse hsr1) throws Exception {
```

```
    hsr1.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from the origin
server
```

```
hsr1.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any circumstance
hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
hsr1.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility
```

```
UserDao userDao = new UserDao();
PostDao postDao = new PostDao();
CommentsDao commentDao = new CommentsDao();
```

```
List<Post> adminPostList = postDao.getPosts();
```

```
List<User> userList = userDao.getUsersForAdmin();
```

```
//Calculate number of comments
Map<User,Integer> commentCnt = new HashMap<User,Integer>();
for( User user : userList){
    List<Comments> cList = commentDao.getCommentsByUser(user);
    commentCnt.put(user,cList.size() );
}
//Calculate number of likes
```

```
Map<User,Integer> likeCnt = new HashMap<User,Integer>();
for(User user : userList){
    int likeCount=0;
    List<Post> postList = postDao.getPostByUser(user);
    for(Post post : postList){
        likeCount=likeCount+post.getLikes().size();
    }
    likeCnt.put(user, likeCount);
}
```

```
ModelAndView mav = new ModelAndView("adminViewPosts");
```

```
mav.addObject("likeCnt",likeCnt);
mav.addObject("commentCnt",commentCnt);
mav.addObject("userList",userList);
mav.addObject("postList",adminPostList);
return mav;
```

```
}
```

```

private static Map<Integer, Integer> sortByComparator(Map<Integer, Integer> unsortMap, final boolean
order)
{
    List<Map.Entry<Integer, Integer>> list = new LinkedList<Map.Entry<Integer,
Integer>>(unsortMap.entrySet());

    // Sorting the list based on values
    Collections.sort(list, new Comparator<Map.Entry<Integer, Integer>>()
{
    public int compare(Map.Entry<Integer, Integer> o1,
        Map.Entry<Integer, Integer> o2)
    {
        if (order)
        {
            return o1.getValue().compareTo(o2.getValue());
        }
        else
        {
            return o2.getValue().compareTo(o1.getValue());
        }
    }
});

    // Maintaining insertion order with the help of LinkedList
    Map<Integer, Integer> sortedMap = new LinkedHashMap<Integer, Integer>();
    for (Map.Entry<Integer, Integer> entry : list)
    {
        sortedMap.put(entry.getKey(), entry.getValue());
    }

    return sortedMap;
}
}

```

EMAIL CONTROLLER:


```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dns.controller;

import com.dns.pojo.User;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.commons.mail.Email;
import org.apache.commons.mail.EmailException;
import org.apache.commons.mail.SimpleEmail;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
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.RequestParam;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author shahd
 */
@Configuration
@ComponentScan(basePackages="com.dns")
@Controller
public class EmailController {

    @RequestMapping(value = "HolidayMaker/sendEmail", method = RequestMethod.POST)
    public ModelAndView contactEmail(HttpServletRequest request, HttpServletResponse response,
    @RequestParam String name,@RequestParam String phone,@RequestParam String email,@RequestParam
String message) throws EmailException {

        response.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the
origin server
        response.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance
        response.setDateHeader("Expires", 0);//Causes the proxy cache to see the page as "stale"
        response.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility

        System.out.println("name"+name);
        System.out.println("phone"+phone);

        Email emailT= new SimpleEmail();
            emailT.setHostName("smtp.googlemail.com");
            emailT.setSmtpPort(465);
            emailT.setAuthentication("holidaymaker2020@gmail.com", "76658145");
            emailT.setSSLOnConnect(true);
            emailT.setFrom(email);

```

```
emailT.setSubject("Holiday Maker Contact From "+name);
emailT.setMsg("Message From "+name+"\n\n "+message);
emailT.addTo("holidaymaker2020@gmail.com");
emailT.send();
```

```
ModelAndView mav = new ModelAndView("error");
```

```
    return mav;
}
}
```

LOGIN CONTROLLER:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dns.controller;
```

```
import com.dns.dao.UserDao;
import com.dns.pojo.Login;
import com.dns.pojo.User;
import com.dns.service.UserService;
import javax.inject.Inject;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import javax.validation.Valid;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.stereotype.Controller;
import org.springframework.stereotype.Service;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.mvc.AbstractController;
```

```
/**
 *
 * @author shahd
 */
```

```

@Configuration
@ComponentScan(basePackages="com.dns")
@Controller
public class LoginController{

    @Autowired
    UserService userService;
    //
    //    @Inject
    //    UserDao userDao;

    @RequestMapping(value = "/login", method = RequestMethod.GET)
    public ModelAndView showLogin(HttpServletRequest request, HttpServletResponse response) throws Exception {

        response.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from the origin
        server
        response.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any circumstance
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
        response.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility

        ModelAndView mav = new ModelAndView("login");
        mav.addObject("login", new Login());
        return mav;
    }
    @RequestMapping(value = "/loginProcess", method = RequestMethod.POST)
    public ModelAndView loginProcess(HttpServletRequest request, HttpServletResponse response,
    @Valid @ModelAttribute("login") Login login, BindingResult br){

    //    response.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from
    the origin server
    //response.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any
    circumstance
    //response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
    //response.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility

    HttpSession session = request.getSession();

    ModelAndView mav = null;
    //    UserDao userDao = new UserDao();

    if(br.hasErrors())
    {

        mav = new ModelAndView("error");
        mav.addObject("errorMessage","Username or password is wrong");
    }
}

```

```

    }

    User user = userService.validateUser(login);
    //    User user = userDao.validateUser(login);
    if (null != user) {

        mav = new ModelAndView("home");
        mav.addObject("firstname", user.getFirstname());
        mav.addObject("Role",user.getRole());

        session.setAttribute("User", user);
    } else {
        mav = new ModelAndView("login");
        mav.addObject("message", "Username or Password is wrong!!");
    }
    return mav;
}

@RequestMapping(value = "/logout", method = RequestMethod.GET)
public ModelAndView logout(HttpServletRequest hsr, HttpServletResponse hsr1) throws Exception {

    hsr1.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from the
    origin server
    hsr1.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any circumstance
    hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
    hsr1.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility

    ModelAndView mav = new ModelAndView("logout");
    hsr.getSession().removeAttribute("User");
    hsr.getSession().invalidate();

    return mav;
}

}

```

REDIRECT CONTROLLER:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dns.controller;

import com.dns.pojo.Login;
import com.dns.pojo.Post;
import com.dns.pojo.User;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

/**
 *
 * @author shahd
 */

@Configuration
@ComponentScan(basePackages="com.dns")
@Controller
public class RedirectController {

    @RequestMapping(value = "/addPosts", method = RequestMethod.GET)
    public ModelAndView addPosts(HttpServletRequest hsr, HttpServletResponse hsr1) throws Exception {

        hsr1.setHeader("Cache-Control","no-cache"); //Forces caches to obtain a new copy of the page from the origin
        server
        hsr1.setHeader("Cache-Control","no-store"); //Directs caches not to store the page under any circumstance
        hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
        hsr1.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility

        ModelAndView mav = new ModelAndView("addNewPost");

        return mav;

    }
}

```



```

    @RequestMapping(value = "/addProcess", method = RequestMethod.POST)
    public ModelAndView addPrcoess(HttpServletRequest request, HttpServletResponse response) {

        response.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from
        the origin server
        response.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance
        response.setDateHeader("Expires", 0);//Causes the proxy cache to see the page as "stale"
        response.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility


        String destCountry = request.getParameter("destCountry");
        int totalDays = Integer.parseInt(request.getParameter("totalDays"));


        System.out.println("dest"+destCountry);
        System.out.println("total"+totalDays);


        HttpSession session = request.getSession();
        session.setAttribute("destCountry",destCountry);
        session.setAttribute("totalDays",totalDays);


        ModelAndView mav = null;
        mav = new ModelAndView("addPosts");
        mav.addObject("post",new Post());


        return mav;
    }

}

```

REGISTRATION CONTROLLER:

```

/*

```

* To change this license header, choose License Headers in Project Properties.

* To change this template file, choose Tools | Templates

* and open the template in the editor.

*/

```
package com.dns.controller;
```

```
import com.dns.dao.UserDao;
```

```
import com.dns.exceptions.UserException;
```

```
import com.dns.pojo.Login;
```

```
import com.dns.pojo.User;
```

```
import com.dns.service.UserService;
```

```
import javax.servlet.http.HttpServletRequest;
```

```
import javax.servlet.http.HttpServletResponse;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.context.annotation.ComponentScan;
```

```
import org.springframework.context.annotation.Configuration;
```

```
import org.springframework.stereotype.Controller;
```

```
import org.apache.commons.mail.Email;
```

```
import org.apache.commons.mail.EmailException;
```

```
import org.apache.commons.mail.SimpleEmail;
```

```
import org.springframework.web.bind.annotation.ModelAttribute;
```

```
import org.springframework.web.bind.annotation.RequestMapping;
```

```
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import org.springframework.web.servlet.ModelAndView;
```

```
/**
```

```
*
```

```
* @author shahd
```

```
*/
```

```
@Configuration
```

```
@ComponentScan(basePackages="com.dns")
```

```
@Controller
```

```
public class RegistrationController {
```

```
    @Autowired
```

```
    UserService userService;
```

```
    @RequestMapping(value = "/register", method = RequestMethod.GET)
```

```
    public ModelAndView showRegister(HttpServletRequest request, HttpServletResponse response) {
```

```
        response.setHeader("Cache-Control", "no-cache"); //Forces caches to obtain a new copy of the page from the origin server
```

```
        response.setHeader("Cache-Control", "no-store"); //Directs caches not to store the page under any circumstance
```

```
        response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
```

```
        response.setHeader("Pragma", "no-cache"); //HTTP 1.0 backward compatibility
```

```
        ModelAndView mav = new ModelAndView("register");
```

```
        mav.addObject("user", new User());
```

```
return mav;
}
```

```
@RequestMapping(value = "/registrationProcess", method = RequestMethod.POST)
public ModelAndView addUser(HttpServletRequest request, HttpServletResponse response,
@ModelAttribute("user") User user) throws EmailException {
```

```
    response.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the
origin server
response.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance
response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
response.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility
```

```
try{
    UserDao userDao = new UserDao();

    int result = userDao.getUserByEmail(user.getEmail());

    if (result==1){
        System.out.println("inside ifffffffffffffffffffff");
        String error="User Already Exists";
        return new ModelAndView("success", "message", error);
    }
```

```
else{

    userDao.addUser(user);
    Email email= new SimpleEmail();
        email.setHostName("smtp.googlemail.com");
        email.setSmtpport(465);
        email.setAuthentication("holidaymaker2020@gmail.com", "76658145");
        email.setSSLonConnect(true);
        email.setFrom("holidaymaker2020@gmail.com");
        email.setSubject("Sign Up Successful");
        email.setMsg("Welcome to the HolidayMaker\n\n Your account has been successfully created.");
        email.addTo(user.getEmail());
        email.send();
        String success="Successfully Registered!!";
        return new ModelAndView("success", "message", success);
    }
```

```
}
catch (UserException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("error", "errorMessage", "error while login");
}
```

```

    }
}

}

```

TRAVEL POSTS CONTROLLER:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dns.controller;

import com.dns.dao.PostDao;
import com.dns.exceptions.UserException;
import com.dns.pojo.Login;
import com.dns.pojo.Post;
import com.dns.pojo.User;
import com.dns.pojo.Itinerary;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Map;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
//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.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.multipart.commons.CommonsMultipartFile;
//import org.springframework.web.bind.annotation.RequestParam;
//import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.servlet.ModelAndView;

```

```

/**
 *
 * @author shahd
 */

@Configuration
@ComponentScan(basePackages="com.dns")
@Controller
public class TravelPostsController {

    @RequestMapping(value = "/viewUserPosts", method = RequestMethod.GET)
    public ModelAndView viewPosts(HttpServletRequest hsr, HttpServletResponse hsr1) {
        hsr1.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the
        origin server
        hsr1.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance
        hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
        hsr1.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility
        ModelAndView mav = null;
        HttpSession session = hsr.getSession();
        User userName = (User) session.getAttribute("User");
        if (userName==null) {
            mav = new ModelAndView("logout");
            return mav;
        }

        else{

            mav = new ModelAndView("viewUserposts");
            User u = (User) session.getAttribute("User");
            List<Post> postList = new ArrayList<Post>();
            PostDao postDao = new PostDao();

            String role = u.getRole();

            postList = postDao.getPostByUser(u);
            mav.addObject("Role",role);
            mav.addObject("postList",postList);
        }
        return mav;
    }

    @RequestMapping(value="/viewUserPosts/deletePost/{id}", method=RequestMethod.POST )
    @ResponseBody
    public ModelAndView deleteUserPost(HttpServletRequest hsr, HttpServletResponse hsr1,

```



```
@PathVariable("id") long id) throws Exception {
```

```
    hsr1.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the  
    origin server  
    hsr1.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance  
    hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"  
    hsr1.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility
```

```
HttpSession session = hsr.getSession();  
    User userName = (User) session.getAttribute("User");  
    if (userName==null) {  
        ModelAndView mav = new ModelAndView("logout");  
        return mav;  
    }  
}
```

```
    //@RequestParam("id") String id  
    System.out.println("ID " + id);  
    System.out.println("inside delete");
```

```
    PostDao postDao = new PostDao();  
    int result = postDao.deletePostById(id);  
    System.out.println("result"+result);
```

```
    ModelAndView mav = new ModelAndView("redirect:/viewUserPosts");  
    User u = (User) session.getAttribute("User");  
    List<Post> postList = new ArrayList<Post>();  
    String role = u.getRole();  
    postList = postDao.getPostByUser(u);  
    System.out.println("post"+postList);  
    mav.addObject("Role",role);  
    mav.addObject("postList",postList);  
  
    return mav;  
}
```

```
    @RequestMapping(value="/viewUserPosts/updatePost/{id}", method=RequestMethod.GET )  
    @ResponseBody  
    public ModelAndView updateUserPost(HttpServletRequest hsr, HttpServletResponse hsr1,  
    @PathVariable("id") long id) throws Exception {
```

```
        hsr1.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the  
        origin server  
        hsr1.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance  
        hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"  
        hsr1.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility
```

```

HttpSession session = hsr.getSession();
    User userName = (User) session.getAttribute("User");
if (userName==null) {
    ModelAndView mav = new ModelAndView("logout");
    return mav;
}

```

```

PostDao postDao = new PostDao();
Post post = postDao.getPostById(id);

```

```

ModelAndView mav = new ModelAndView("updateUserpost");

```

```

    System.out.println("postUser"+post.getUser());

    mav.addObject("post",post);

    return mav;
}

```

```

@RequestMapping(value = "/postProcess", method = RequestMethod.POST)
public ModelAndView newPost(HttpServletRequest request, HttpServletResponse response,
    @ModelAttribute("post") Post post,@RequestParam CommonsMultipartFile file) throws UserException {
    response.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the
origin server
response.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance
response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
response.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility

```

```

HttpSession session = request.getSession();
    User userName = (User) session.getAttribute("User");
if (userName==null) {
    ModelAndView mav = new ModelAndView("logout");
    return mav;
}

```

```

ModelAndView mav = null;
    System.out.println("country"+post.getCountryOfTravel());
    System.out.println("travel"+post.getTravelledFrom());
    System.out.println("title"+post.getTitle());

```

```

int days = (Integer) session.getAttribute("totalDays");
String images = request.getParameter("images");

// post.setImageList(images);
// System.out.println("images"+images);
String path=session.getServletContext().getRealPath("/");
String filename=file.getOriginalFilename();

String storedPath= "C:\\Users\\shahd\\Documents\\Web
Tools\\Netbeans\\HolidayMaker_MileStone3\\HolidayMaker\\src\\main\\webapp\\resources\\images";

System.out.println("path"+path);
System.out.println("filename"+filename);

//later surround in try catch
try{
byte barr[]=file.getBytes();

BufferedOutputStream bout=new BufferedOutputStream(
    new FileOutputStream(storedPath+"/"+filename));
bout.write(barr);
bout.flush();
bout.close();
}catch(Exception e){System.out.println(e);}

post.setImageList(filename);

for(int i =1 ; i<=days;i++){
    Iternary iter = new Iternary();
    iter.setPost(post);
    iter.setDescription(request.getParameter("Day"+i));
    iter.setDayNo(i);
    System.out.println("day"+request.getParameter("Day"+i));

    post.getIterList().add(iter);
}

System.out.println("iterList"+post.getIterList());

PostDao postDao = new PostDao();

post.setUser((User) request.getSession().getAttribute("User"));

String pattern = "MM-dd-yyyy";

```

```
SimpleDateFormat simpleDateFormat = new SimpleDateFormat(pattern);
String date = simpleDateFormat.format(new Date());
post.setDate(date);
int result = postDao.addPost(post);
```

```
if(result==0){
```

```
    mav= new ModelAndView("error");
```

```
}else{
```

```
    mav=new ModelAndView("postAdded");
```

```
}
```

```
// UserDao userDao = new UserDao();
```

```
// User user = userService.validateUser(login);
```

```
// User user = userDao.validateUser(login);
```

```
// if (null != user) {
```

```
// mav = new ModelAndView("welcome");
```

```
// mav.addObject("firstname", user.getFirstname());
```

```
// mav.addObject("Role",user.getRole());
```

```
// } else {
```

```
// mav = new ModelAndView("login");
```

```
// mav.addObject("message", "Username or Password is wrong!!");
```

```
// }
```

```
// return mav;
```

```
return mav;
```

```
}
```

```
@RequestMapping(value = "/updateProcess", method = RequestMethod.POST )
```

```
public ModelAndView updatePost(HttpServletRequest request, HttpServletResponse response,
```

```
@ModelAttribute("post") Post post) throws UserException {
```

```
    response.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the
origin server
```

```
response.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance
```

```
response.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
```

```
response.setHeader("Pragma","no-cache");//HTTP 1.0 backward compatibility
```

```
HttpSession session = request.getSession();
```

```
User userName = (User) session.getAttribute("User");
```

```
if (userName==null) {
```

```
ModelAndView mav = new ModelAndView("logout");
return mav;
}
```

```
ModelAndView mav = null;
System.out.println("country"+post.getCountryOfTravel());
System.out.println("travel"+post.getTravelledFrom());
System.out.println("title"+post.getTitle());
```

```
PostDao postDao = new PostDao();
```

```
Post originalPost = postDao.getPostById(post.getId());
```

```
int days = originalPost.getIterList().size();
```

```
for(int i =1 ; i<=days;i++){
    Iterary iter = new Iterary();
    iter.setPost(post);
    iter.setId(Long.parseLong(request.getParameter("ID"+i)));
    iter.setDescription(request.getParameter("Day"+i));
    iter.setDayNo(i);
    System.out.println("day"+request.getParameter("Day"+i));
```

```
    post.getIterList().add(iter);
}
```

```
System.out.println("post"+post.getUser());
```

```
post.setUser(userName);
String pattern = "MM-dd-yyyy";
SimpleDateFormat simpleDateFormat = new SimpleDateFormat(pattern);
String date = simpleDateFormat.format(new Date());
post.setDate(date);
```

```
int result = postDao.updatePost(post);
```

```
if(result==0){
```

```
    mav= new ModelAndView("error");
```

```
}else{
```

```

        mav=new ModelAndView("postAdded");

    }

// UserDao userDao = new UserDao();
// User user = userService.validateUser(login);

// User user = userDao.validateUser(login);
// if (null != user) {
//     mav = new ModelAndView("welcome");
//     mav.addObject("firstname", user.getFirstname());
//     mav.addObject("Role",user.getRole());
// } else {
//     mav = new ModelAndView("login");
//     mav.addObject("message", "Username or Password is wrong!!");
// }
// return mav;
return mav;
}

}

```

Destination Controller:

```

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package com.dns.controller;

import com.dns.dao.CommentsDao;
import com.dns.dao.PostDao;
import com.dns.dao.UserDao;

import com.dns.exceptions.UserException;
import com.dns.pojo.Comments;
import com.dns.pojo.Itinerary;
import com.dns.pojo.Likes;
import com.dns.pojo.Post;
import com.dns.pojo.User;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;
import java.util.Set;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
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;

/**
 *
 * @author shahd
 */

@Configuration
@ComponentScan(basePackages="com.dns")
@Controller
public class destinationController {

    @RequestMapping(value = "/viewDestinations", method = RequestMethod.GET)
    public ModelAndView viewPosts(HttpServletRequest hsr, HttpServletResponse hsr1) {
        hsr1.setHeader("Cache-Control","no-cache");//Forces caches to obtain a new copy of the page from the
        origin server
        hsr1.setHeader("Cache-Control","no-store");//Directs caches not to store the page under any circumstance
        hsr1.setDateHeader("Expires", 0); //Causes the proxy cache to see the page as "stale"
    }
}

```

```
hsr1.setHeader("Pragma","no-cache"); //HTTP 1.0 backward compatibility
```

```
HttpSession session = hsr.getSession();
    User userName = (User) session.getAttribute("User");
if (userName==null) {
    ModelAndView mav = new ModelAndView("logout");
    return mav;
}
```

```
    ModelAndView mav = new ModelAndView("viewDestinations");
```

```
    List<Post> filterpostList = new ArrayList<Post>();
```

```
    List<Post> postList = new ArrayList<Post>();
```

```
    PostDao postDao = new PostDao();
```

```
    postList = postDao.getPosts();
```

```
    filterpostList=postDao.getPosts();
```

```
    mav.addObject("postList",postList);
```

```
    mav.addObject("filterpostList",filterpostList);
```

```
    return mav;
```

```
}
```

```
@RequestMapping(value = "/viewDestinations", method = RequestMethod.POST)
public ModelAndView filterPosts(HttpServletRequest hsr, HttpServletResponse hsr1) {
```

```
    ModelAndView mav = new ModelAndView("viewDestinations");
```

```
    String country = hsr.getParameter("country");
```

```
    List<Post> postList = new ArrayList<Post>();
```

```
    List<Post> filterpostList = new ArrayList<Post>();
```

```
    PostDao postDao = new PostDao();
```

```
    postList = postDao.getPostByCountry(country);
```

```
    System.out.println("postList"+postList);
```

```
    filterpostList=postDao.getPosts();
```

```
    mav.addObject("postList",postList);
```

```
    mav.addObject("filterpostList",filterpostList);
```

```
    mav.addObject("count",country);
```

```
    return mav;
```

```
}
```



```

@RequestMapping(value = "/viewPostInDetail/{id}", method = RequestMethod.GET)
@ResponseBody
public ModelAndView viewDetails(HttpServletRequest hsr, HttpServletResponse hsr1, @PathVariable("id")
long id) {

    ModelAndView mav = new ModelAndView("viewDetails");

    User u = (User) hsr.getSession().getAttribute("User");

    PostDao postDao = new PostDao();
    Post post = postDao.getPostById(id);
    UserDao userDao = new UserDao();

    Set<Likes> likes = post.getLikes();
    Set<Iternary> iter = post.getIterList();
    List<Iternary> iterList = new ArrayList<Iternary>(iter);

    Comparator<Iternary> compareById = new Comparator<Iternary>() {
@Override
public int compare(Iternary o1, Iternary o2) {
    return Integer.compare(o1.getDayNo(), o2.getDayNo());
}
};

// for Java8 Using Lamba
//     Comparator<Iternary> compareById = (Iternary i1, Iternary i2) ->
//         Integer.compare(i1.getDayNo(), i2.getDayNo());

    Collections.sort(iterList, compareById);

    boolean liked = false;

    for(Likes lik : post.getLikes()){

        if(lik.getUserID()==u.getId()&&lik.isIsLike()){
            liked=true;
        }

    }

    //code for getting number of likes

```

```

        Set<Likes> likeList = post.getLikes();

        int likeCount =0;

        for(Likes like: likeList){

            if(like.isIsLike()){

                likeCount=likeCount+1;

            }

        }

        mav.addObject("likeCount",likeCount);
        List<User> userList = userDao.getAllUsers();
        mav.addObject("post",post);
        mav.addObject("userList",userList);
        mav.addObject("liked",liked);
        mav.addObject("iterList",iterList);

        return mav;
    }

```

```

@RequestMapping(value = "/viewPostInDetail/change/{id}", method = RequestMethod.GET)
@ResponseBody
public ModelAndView likeChange(HttpServletRequest hsr, HttpServletResponse hsr1, @PathVariable("id")
long id) {

```

```

    ModelAndView mav = new ModelAndView("viewDetails");

```

```

    User u = (User) hsr.getSession().getAttribute("User");

```

```

    boolean liked = false;
    String update = hsr.getParameter("like");
    System.out.println("update"+update);
    PostDao postDao = new PostDao();
    Post post = postDao.getPostById(id);

```

```

    if(update.equalsIgnoreCase("like")){
        liked=true;
        boolean present=false;
        for(Likes lik : post.getLikes()){

```

```

            if(u.getId()==lik.getUserID()){
                //System.out.println("inside uuu");

```

```

        present=true;

    }

}

if(present){

    for(Likes lik : post.getLikes()){
        if(u.getId()==lik.getUserID()){
            lik.setIsLike(true);
            postDao.updatePost(post);

        }

    }
}
else{

    Likes l = new Likes();
    l.setUserID(u.getId());
    l.setIsLike(true);
    l.setPost(post);

    post.getLikes().add(l);

    postDao.updatePost(post);

}
}
else{

    liked = false;

    for(Likes lik : post.getLikes()){

        if(u.getId()==lik.getUserID()){
            System.out.println("inside uuu");
            lik.setIsLike(false);
            postDao.updatePost(post);

        }

    }

}

}

```

```

        UserDao userDao = new UserDao();

        Set<Likes> likeList = post.getLikes();

        int likeCount =0;

        for(Likes like: likeList){

            if(like.isIsLike()){

                likeCount=likeCount+1;

            }

        }

        mav.addObject("likeCount",likeCount);

        List<User> userList = userDao.getAllUsers();
        mav.addObject("post",post);
        mav.addObject("userList",userList);
        mav.addObject("liked",liked);

        return mav;
    }

```

```

@RequestMapping(value = "/viewPostInDetail/{id}", method = RequestMethod.POST)
@ResponseBody
public ModelAndView viewDetailsChange(HttpServletRequest hsr, HttpServletResponse hsr1,
@PathVariable("id") long id) throws UserException {

```

```

        ModelAndView mav = new ModelAndView("redirect:/viewPostInDetail/{id}");

```

```

        User u = (User) hsr.getSession().getAttribute("User");

```

```

        long pid = Long.parseLong(hsr.getParameter("postCom"));
        String commentDesc = hsr.getParameter("commentD");

```

```

        System.out.println("user"+u);
        System.out.println("post"+pid);
        PostDao postDao = new PostDao();

```

```

Post post = postDao.getPostById(pid);

Set<Likes> likeList = post.getLikes();

int likeCount =0;

for(Likes like: likeList){

    if(like.isIsLike()){

        likeCount=likeCount+1;

    }

}

//      CommentsDao commentDao = new CommentsDao();

Comments c = new Comments();

c.setPost(post);
c.setUserId(u.getId());
c.setCommentDesc(commentDesc);

    post.getComments().add(c);

postDao.updatePost(post);
UserDao userDao = new UserDao();

List<User> userList = userDao.getAllUsers();

mav.addObject("likeCount",likeCount);
    mav.addObject("userList",userList);
    mav.addObject("post",post);
return mav;
}

}

```