

# 6250 Final Project

Jiamin Shang 001267391 section4

## 1. Project name:

Blood Supply and management website

## 2. Domain of the project:

This is a project converted from AED project about blood donation and distribution. In this project, I try to build an ecosystem about blood donation and management work flow occurred in clinic, blood banks and blood distribution management centers.

## 3. Scope of the project:

- (1) Implement the system of blood supply and management.
- (2) Build a model and implement the detail work flow of blood donation, use and management.
- (3) Define the layers of blood management system and create an ecosystem. For example, in a specific city there is a blood bank connected with many blood clinics, and this blood bank is one part of the superior blood center at higher level.
- (4) Donor and blood receiver's personal health info track.
- (5) The management function includes the test and the distribution of blood, and records track, etc.
- (6) Provide information for staff from the database of specific role in this ecosystem, and they can learn the conditions and send requirement to contact with others.
- (7) Display the report concerning some vital data on each level of this ecosystem.

## 4. Technologies and tools:

- (1) **Back end:** SpringMVC and hibernate framework.
- (2) **Database:** MySQL.
- (3) **Front end:** bootstrap, JQuery, Ajax, html/css, velocity and jsp(jstl).

## 5. Functions:

- (1) sign up with email message.
- (2) reset password function through email.
- (3) set up avatar by upload image, update user information in profile page.
- (4) ajax used in create organization and user.
- (5) work flow of blood donation, use and management with 6 roles.
- (6) export excel reports by using apache.poi.
- (7) servlet filter to prevent XSS and SQL injection
- (8) 11 hibernate entities with onetoone, manytoone, onetomany, manytomany and Inheritance mapping
- (9) distinct and detail UI with JS, bootstrap, AJAX, JSTL/velocity
- (10) provide work flow/blood/personal info track, query data with HQL and Criteria.
- (11) verify user identity programmatically. And verify admin by interceptor
- (12) handle most exception and have an error message area in every page.
- (13) some of pages use SimpleSpringForm and validator to validate the input.

Login page can auto verify all kind of roles and redirect to different work area. And with the forget password function reset password by email.

## Log in

Username:

Password:

[Forget password?](#)

Log in

Cancel



2:07

reset password

收件人: xdweasdfzxc@gmail.com

Here is the link to reset your password: [http://localhost:8080/jiaminfinalp/resetpassword.htm?GsiL6FQi-JXwPOS5xa\\*Gkscm5AUvMk53HgQNU=2&5sER5t\\*nC8f=5QA/pmQSOtD](http://localhost:8080/jiaminfinalp/resetpassword.htm?GsiL6FQi-JXwPOS5xa*Gkscm5AUvMk53HgQNU=2&5sER5t*nC8f=5QA/pmQSOtD)



2:12

Hi, jerax

收件人: xdweasdfzxc@gmail.com

Sign up successfully

Admin dashboard all data can be tracked the admin have access to disable user and delete organization

## Info

User

Organization

Work requests

User List

	UID	Username	First Name	Last Name	Role	
▼	1	admin	Jiamin	Admin	admin	<div>Disable</div>
▼	2	jiamin	Jiamin	Shang	user	<div>Disable</div>
		gender: Male	Date Of Birth: 1993-09-09	Phone: 8572505488	Email: xdweasdfzxc@gmail.com	
▼	3	jimmy	jimmy	Chen	user	<div>Active</div>
		gender: Female	Date Of Birth: 1993-09-09	Phone: 8572505488	Email: jimmy@service.com	
▼	4	nurse1	n1	n1	nurse	<div>Disable</div>

Info

UserOrganizationWork requests

User List

	UID	Username	First Name	Last Name	Role	
▼	1	admin	Jiamin	Admin	admin	Disable
	gender: Male    Date Of Birth: 1993-09-09    Phone: 8572505488    Email: admin@service.com					
▼	2	jiamin	Jiamin	Shang	user	Disable
	gender: Male    Date Of Birth: 1993-09-09    Phone: 8572505488    Email: xdweasdfzxc@gmail.com					
▼	3	jimmy	jimmy	Chen	user	Disable
▼	4	nurse1	n1	n1	nurse	Disable
▼	5	nurse2	n2	n2	nurse	Disable

Info

UserOrganizationWork requests

Organization List

	OID	Organization name	Organization type	
▼	1	EcoSystem	system	Delete
	User belong to it: jessy , jimmy , jerax , jiamin , deli1 , junxi , admin ,			
▼	2	New England BMC	bmc	Delete
▼	3	Boston BloodBank	bb	Delete
▼	4	Cambridge BloodBank	bb	Delete
▼	5	clinic1	clinic	Delete
	User belong to it: nurse1 , lab1 , nurse2 ,			
▼	6	clinic2	clinic	Delete
▼	7	clinic3	clinic	Delete

# Info

User   Organization   Work requests

## Work requests queue

	Request date	Request type	Status	Quantities
▼	04-23-2017 03:34:37	Donate	Stored	400
blood used by Jiamin request is responsible by Name:L1 Role:labassistant, Name:n1 Role:nurse, Name:d1 Role:deliver, Name:Jiamin Role:user,				
▼	04-23-2017 03:34:50	Donate	Stored	500
▼	04-23-2017 03:35:16	Donate	denied	300
blood used by no one request is responsible by Name:n1 Role:nurse, Name:Jiamin Role:user,				
▼	04-23-2017 03:38:32	Donate	Stored	600

Dashboard

Home

Create staff

Create Organization

Profile

admin

Logout

# Info

User   Organization   Work requests

## Work requests queue

	Request date	Request type	Status	Quantities
▼	04-23-2017 03:34:37	Donate	Stored	400
blood used by Jiamin request is responsible by Name:n1 Role:nurse, Name:L1 Role:labassistant, Name:Jiamin Role:user, Name:d1 Role:deliver,				
▼	04-23-2017 03:34:50	Donate	Stored	500
▼	04-23-2017 03:35:16	Donate	denied	300
▼	04-23-2017 03:38:32	Donate	Stored	600

## Create User

**Username:**

**Password:**

**First Name:**

**Last Name:**

**Gender:**

**Date Of Birth:**

**Email:**

**Phone:**

**Select Role:**

**Select role first:**

create user and organization with AJAX

## Create new Organization

**Organization Name:**


**Select Organization Type:**

**Select type first:**

User profile page can update password user information and avatar

## Profile

**Info** **Security**



**Username:**

**First name:**

**Last name:**

**Gender:**

**Date of birth:**

**Email:**

**Phone:**

## Profile

**Info** **Security**

**Current Password:**

**New Password:**

In user's area user can export excel report

History records

blood recordvital sign

Vital sign records

Date	Healthy?
04-23-2017 21:05:34	Yes
Blood type: AB Diabetes: No	Hemoglobin: Normal Other temporary condition: No Infection: No Other permanent condition: No
04-24-2017 00:27:00	Yes
04-24-2017 00:27:09	Yes

Export report

B6

	A	B	C	D	E	F	G	H	I
1		Jiamin	Shang	1993-09-09	Male				
2	Date	Healthy?	Bloodtype	Hemoglobin	Infection	Diabetes	TempCondit	PermCondition	
3	04-23-2017	Yes	AB	Normal	No	No	No	No	
4	04-24-2017	Yes	AB	Normal	No	No	No	No	
5	04-24-2017	Yes	AB	Normal	No	No	No	No	

Work area

History records

area1area2area3area4

Request queue

Request Date	Request Type	Status	Donor/User	Blood Type	Quantities	
04-24-2017 02:13:51	Use	Pending(nurse)	jimmy	AB	400	<div>detailrequiredeny</div>

Blood inventory data panel

Blood inventory

The whole area

A:	2000ml	29%
B:	2000ml	29%
AB:	1100ml	16%
O:	1600ml	23%

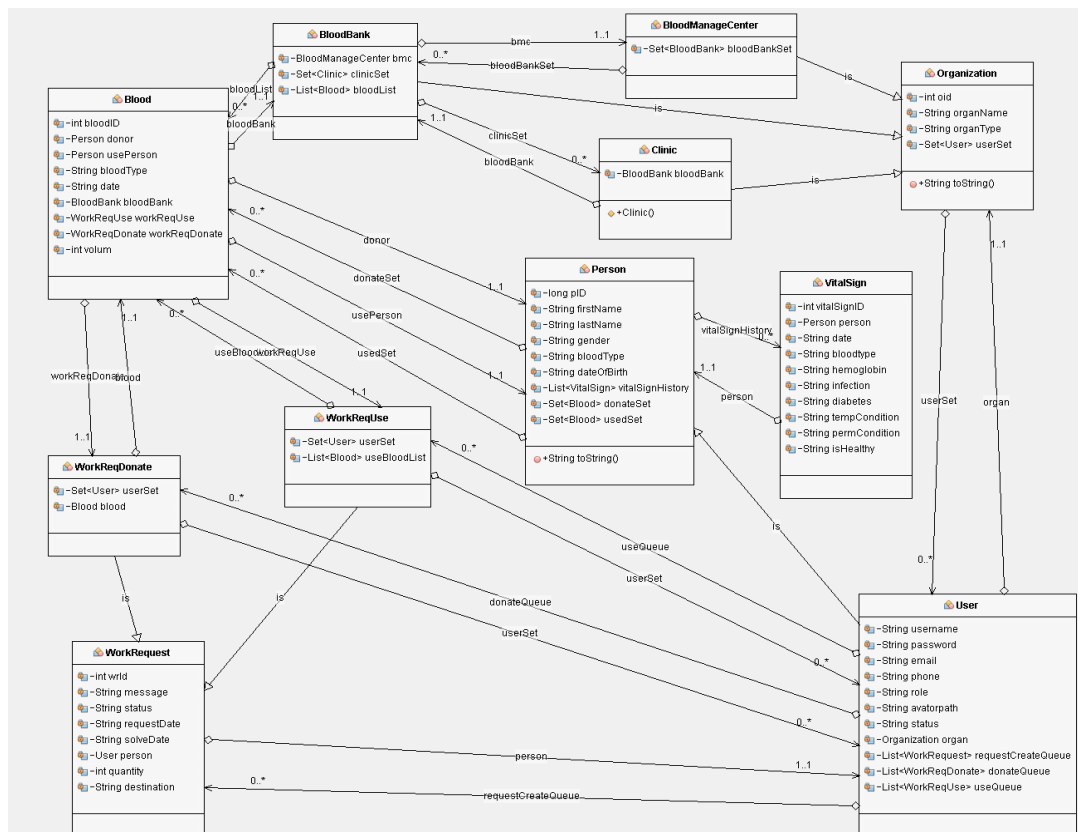
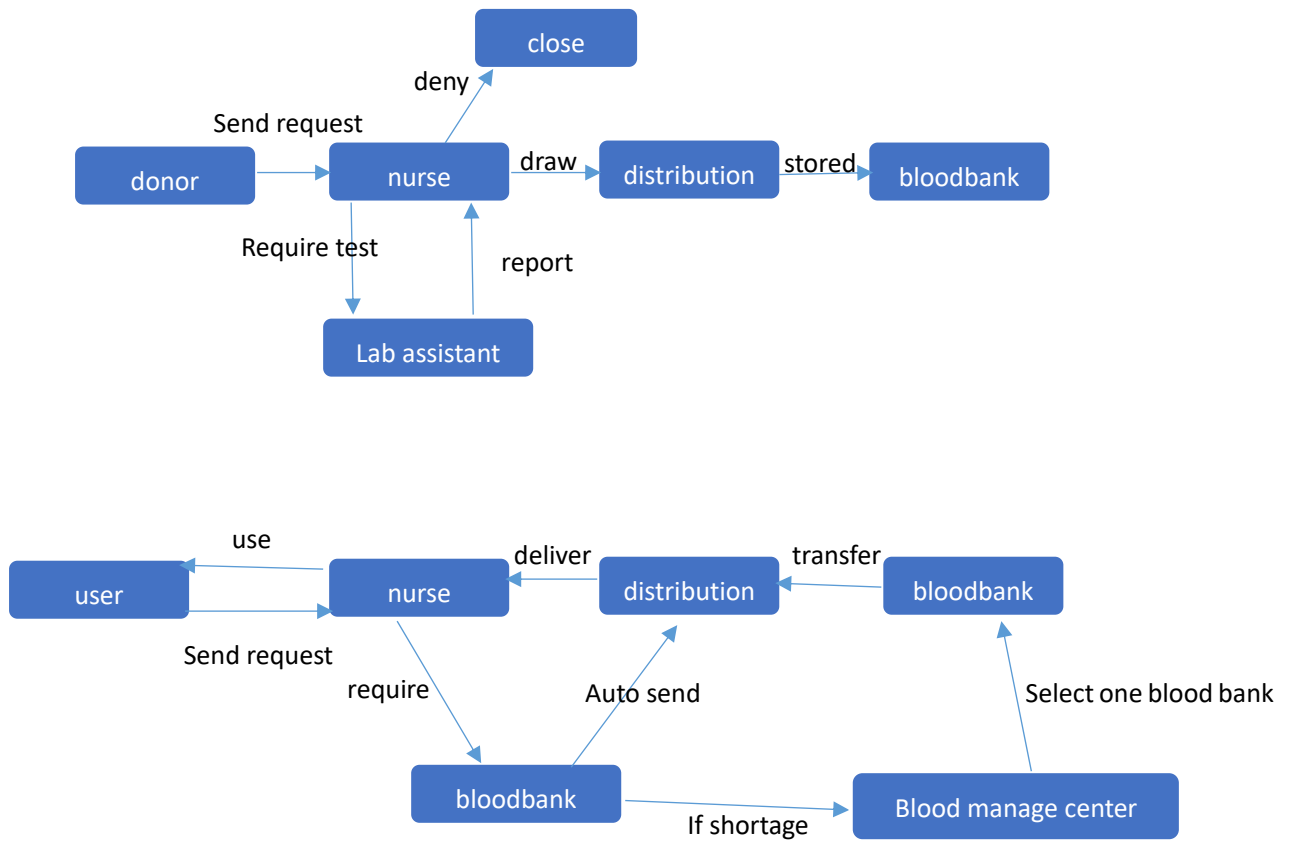
Cambridge BloodBank

A:	0ml	0%
B:	2000ml	100%
AB:	0ml	0%
O:	0ml	0%

Boston BloodBank

A:	2000ml	42%
B:	0ml	0%
AB:	1100ml	23%
O:	1600ml	34%

Export report



## 11 Controllers

### AdminController

```
package com.jiamin.jiaminfinalp;

import java.util.List;

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

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

import com.jiamin.dao.DAO;
import com.jiamin.dao.OrganizationDAO;
import com.jiamin.dao.UserDAO;
import com.jiamin.dao.WorkRequestDAO;
import com.jiamin.exception.OperateException;
import com.jiamin.pojo.BloodBank;
import com.jiamin.pojo.BloodManageCenter;
import com.jiamin.pojo.Clinic;
import com.jiamin.pojo.Organization;
import com.jiamin.pojo.User;
import com.jiamin.pojo.VitalSign;
import com.jiamin.pojo.WorkReqDonate;
import com.jiamin.pojo.WorkReqUse;

/**
 * Handles requests for the application home page.
 */
@Controller
public class AdminController {

    @Autowired
    @Qualifier("userDao")
    UserDAO userDao;

    @Autowired
    @Qualifier("organDao")
    OrganizationDAO organDao;
```



```
@Autowired
@Qualifier("wrDao")
WorkRequestDAO wrDao;
```

```
@RequestMapping(value = "/admin/home.htm", method = RequestMethod.GET)
public ModelAndView adminhome(HttpServletRequest request) {
    ModelAndView mv = new ModelAndView();
    DAO.close();
    try {
        List<Organization> organList = organDao.organList();
        List<User> userList = userDao.getList();
        List<WorkReqDonate> wrdList = wrDao.wrdList();
        List<WorkReqUse> wruList = wrDao.wruList();
        mv.addObject("organList", organList);
        mv.addObject("userList", userList);
        mv.addObject("wrdList", wrdList);
        mv.addObject("wruList", wruList);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("userdonate", "errorMessage", "error while
get organization list");
    }
    mv.setViewName("dashboard");
    return mv;
}
```

```
@RequestMapping(value = "/admin/adduser.htm", method =
RequestMethod.GET)
public ModelAndView adduserpage(HttpServletRequest request) {
    ModelAndView mv = new ModelAndView();

    mv.setViewName("admincreateuser");
    return mv;
}
```

```
@RequestMapping(value = "/admin/addorgan.htm", method =
RequestMethod.GET)
public ModelAndView addOrganpage(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();

    mv.setViewName("createorgan");
    return mv;
}
```

```

    @RequestMapping(value = "/admin/adduser.htm", method =
RequestMethod.POST)
    public ModelAndView addUser(HttpServletRequest request) {
        ModelAndView mv = new ModelAndView();
        try {
            User users = userDao.get(request.getParameter("username"));
            if (users != null) {
                System.out.println("username already exists try another one");
                return new ModelAndView("admincreateuser", "errorMessage",
"username already exists try another one");
            }
            if
(request.getParameter("organ").equals("0") || request.getParameter("urole").equals(
"none")){
                return new ModelAndView("admincreateuser", "errorMessage",
"Select role and organization");
            }
            User user = new User();
            Organization organ =
organDao.getOrgan(Integer.parseInt(request.getParameter("organ")));

            user.setFirstName(request.getParameter("firstName"));
            user.setLastName(request.getParameter("lastName"));
            user.setEmail(request.getParameter("email"));
            user.setGender(request.getParameter("gender"));
            user.setDateOfBirth(request.getParameter("dateOfBirth"));
            user.setPhone(request.getParameter("phone"));
            user.setOrgan(organ);
            user.setUsername(request.getParameter("username"));
            user.setPassword(request.getParameter("password"));
            user.setStatus("active");
            user.setRole(request.getParameter("urole"));

            userDao.register(user);
            mv.addObject("successMessage", "create successfully");
        } catch (OperateException e) {
            System.out.println("Exception: " + e.getMessage());
            return new ModelAndView("admincreateuser", "errorMessage", "error
while create account");
        }

        mv.setViewName("admincreateuser");
    }

```

```

        return mv;
    }

    @RequestMapping(value = "/admin/addorgan.htm", method =
RequestMethod.POST)
    public ModelAndView createOrgan(HttpServletRequest request) throws
Exception {
        String otype = request.getParameter("otype");
        String oname = request.getParameter("oname");
        int pareneto = Integer.parseInt(request.getParameter("pareneto"));
        try {
            Organization o = organDao.getOrgan(oname);
            if (o != null) {
                System.out.println("Organization name already exists try another
one");
                return new ModelAndView("createorgan", "errorMessage",
                    "Organization name already exists try another one");
            }
        } catch (OperateException e) {
            System.out.println("Exception: " + e.getMessage());
            return new ModelAndView("createorgan", "errorMessage", "error while
create new organization");
        }
        if (otype.equals("clinic")) {
            Clinic organ = new Clinic();
            organ.setOrganName(oname);
            organ.setOrganType(otype);
            try {
                BloodBank bb = organDao.getBloodBank(pareneto);
                organ.setBloodBank(bb);
                organDao.createClinic(organ);
            } catch (OperateException e) {
                System.out.println("Exception: " + e.getMessage());
                return new ModelAndView("createorgan", "errorMessage", "error
while create new organization");
            }
        } else if (otype.equals("bb")) {
            BloodBank organ = new BloodBank();
            organ.setOrganName(oname);
            organ.setOrganType(otype);
            try {
                BloodManageCenter bmc = organDao.getBMC(pareneto);
                organ.setBmc(bmc);
                organDao.createBloodBank(organ);
            }

```

```

        } catch (OperateException e) {
            System.out.println("Exception: " + e.getMessage());
            return new ModelAndView("createorgan", "errorMessage", "error
while create new organization");
        }
    } else if (otype.equals("bmc")) {
        BloodManageCenter organ = new BloodManageCenter();
        organ.setOrganName(oname);
        organ.setOrganType(otype);
        try {
            organDao.createBMC(organ);
        } catch (OperateException e) {
            System.out.println("Exception: " + e.getMessage());
            return new ModelAndView("createorgan", "errorMessage", "error
while create new organization");
        }
    } else {
        System.out.println("no type of organization is selected, please select
one");
        return new ModelAndView("createorgan", "errorMessage",
            "no type of organization is selected, please select one");
    }
    return new ModelAndView("createorgan", "successMessage", "create
successfully");
}

```

```

@RequestMapping(value = "/admin/disableuser.htm", method =
RequestMethod.GET)
public ModelAndView disableuser(HttpServletRequest request) {
    try {
        int pid = Integer.parseInt(request.getParameter("pid"));
        User u = userDao.get(pid);
        u.setStatus("disable");
        userDao.update(u);
        return new ModelAndView("redirect:/admin/home.htm");
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("redirect:/admin/home.htm",
"errorMessage", "error while disable user");
    }
}

```

```

@RequestMapping(value = "/admin/activeuser.htm", method =
RequestMethod.GET)

```

```

public ModelAndView activeuser(HttpServletRequest request) {
    try {
        int pid = Integer.parseInt(request.getParameter("pid"));
        User u = userDao.get(pid);
        u.setStatus("active");
        System.out.println(u.getStatus());
        userDao.update(u);
        return new ModelAndView("redirect:/admin/home.htm");
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("redirect:/admin/home.htm",
"errorMessage", "error while active user");
    }
}

@RequestMapping(value = "/admin/deleteorgan.htm", method =
RequestMethod.GET)
public ModelAndView deleteOrgan(HttpServletRequest request) {

    try {
        int oid = Integer.parseInt(request.getParameter("oid"));
        Organization o = organDao.getOrgan(oid);
        organDao.delete(o);
        return new ModelAndView("redirect:/admin/home.htm");
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("redirect:/admin/home.htm",
"errorMessage", "error while delete organization");
    }
}

@RequestMapping(value = "/adminerror.htm", method = RequestMethod.GET)
public ModelAndView adminerror(HttpServletRequest request) {
    ModelAndView mv = new ModelAndView();
    mv.addObject("errorMessage", "Sensitive resources, please don't access");
    mv.setViewName("error");
    return mv;
}
}

```

## AjaxController

```
package com.jiamin.jiaminfinalp;

import java.io.PrintWriter;
import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import com.jiamin.dao.OrganizationDAO;
import com.jiamin.pojo.Organization;

/**
 * Handles requests for the application home page.
 */
@Controller
public class AjaxController {

    @Autowired
    @Qualifier("organDao")
    OrganizationDAO organDao;

    @RequestMapping(value = "/ajax/organ", method = RequestMethod.POST)
    public String showOrgan(HttpServletRequest request, HttpServletResponse
response)throws Exception{

        StringBuilder str = new StringBuilder();
        String otype = request.getParameter("otype");
        String parenttype = "none";
        String defaultMsg = "--Select--";
        System.out.println(otype);
        if(otype.equals("bb")){
            parenttype = "bmc";
        }else if(otype.equals("clinic")){
            parenttype = "bb";
        }

        if(parenttype.equals("none")){
```

```

        defaultMsg = "no parent organizations";
    }
    List<Organization> list = organDao.organList(parenttype);
    str.append("<option value=\"");
    str.append("0");
    str.append("\>");
    str.append(defaultMsg);
    str.append("</option>");

    for (Organization organ : list)
    {
        str.append("<option value=\"");
        str.append(organ.getOid());
        str.append("\>");
        str.append(organ.getOrganName());
        System.out.println(organ.getOrganName());
        str.append("</option>");
    }
    //response.setCharacterEncoding("UTF-8");
    response.setContentType("text/html");

    PrintWriter out=response.getWriter();
    out.print(str.toString());
    System.out.println(str.toString());
    out.close();

    return null;
}

```

```

@RequestMapping(value = "/ajax/user", method = RequestMethod.POST)
public String showOrganization(HttpServletRequest request,
    HttpServletResponse response)throws Exception{

```

```

    StringBuilder str = new StringBuilder();
    String urole = request.getParameter("urole");
    String organtype = "none";
    String defaultMsg = "--Select--";
    System.out.println(urole);
    if(urole.equals("bmcm")){
        organtype = "bmc";
    }else if(urole.equals("labassistant") || urole.equals("nurse")){
        organtype = "clinic";
    }else if(urole.equals("bbm")){
        organtype = "bb";
    }

```

```

    }else if(urole.equals("deliver") || urole.equals("user")){
        organtype = "system";
    }

    if(organtype.equals("none")){
        defaultMsg = "-----";
    }
    List<Organization> list = organDao.organList(organtype);
    str.append("<option value=\"");
    str.append("0");
    str.append("\>");
    str.append(defaultMsg);
    str.append("</option>");

    for (Organization organ : list)
    {
        str.append("<option value=\"");
        str.append(organ.getOid());
        str.append("\>");
        str.append(organ.getOrganName());
        System.out.println(organ.getOrganName());
        str.append("</option>");
    }
    //response.setCharacterEncoding("UTF-8");
    response.setContentType("text/html");

    PrintWriter out=response.getWriter();
    out.print(str.toString());
    System.out.println(str.toString());
    out.close();

    return null;
}

}

```



BMCController

```
package com.jiamin.jiaminfinalp;
```

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Set;
```

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

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;
```

```
import com.jiamin.dao.DAO;
import com.jiamin.dao.InfoDAO;
import com.jiamin.dao.OrganizationDAO;
import com.jiamin.dao.UserDAO;
import com.jiamin.dao.WorkRequestDAO;
import com.jiamin.exception.OperateException;
import com.jiamin.pojo.Blood;
import com.jiamin.pojo.BloodBank;
import com.jiamin.pojo.User;
import com.jiamin.pojo.WorkReqDonate;
import com.jiamin.pojo.WorkReqUse;
import com.jiamin.pojo.WorkRequest;
import com.jiamin.tools.BloodBankInventory;
import com.jiamin.tools.BloodBankStatus;
import com.jiamin.tools.WruBbs;
```

```
/**
 * Handles requests for the application home page.
 */
```

```
@Controller
```

```
public class BMCController {
```

```
    @Autowired
```

```
    @Qualifier("userDao")
```

```
UserDAO userDao;
```

```
@Autowired
```

```
@Qualifier("organDao")
```

```
OrganizationDAO organDao;
```

```
@Autowired
```

```
@Qualifier("wrDao")
```

```
WorkRequestDAO wrDao;
```

```
@Autowired
```

```
@Qualifier("infoDao")
```

```
InfoDAO infoDao;
```

```
@RequestMapping(value = "/bmc/home.htm", method = RequestMethod.GET)
```

```
public ModelAndView bmc(HttpServletRequest request) {
```

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

```
    ModelAndView mv = new ModelAndView();
```

```
    DAO.close();
```

```
    User sessionuser = (User) session.getAttribute("user");
```

```
    User user = null;
```

```
    try {
```

```
        if(sessionuser != null)
```

```
            user = userDao.get((int)sessionuser.getId());
```

```
    } catch (OperateException e) {
```

```
        System.out.println("Exception: " + e.getMessage());
```

```
        return new ModelAndView("nurseworkarea", "errorMessage", "error  
while get user");
```

```
    }
```

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

```
        mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
```

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

```
        mv.setViewName("login");
```

```
        return mv;
```

```
    }
```

```
    if (!user.getRole().equals("bmcm")) {
```

```
        mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
```

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

```
        mv.setViewName("login");
```

```
        return mv;
```

```
    }
```

```
    try {
```

```
        List<WorkReqUse> wruList = wrDao.wruListBySta("Blood shortage");
```

```
        List<WruBbs> wrubbsList = new ArrayList<WruBbs>();
```

```

        Set<BloodBank> bankList =
organDao.getBMC(user.getOrgan().getOrganName()).getBloodBankSet();
        for (WorkReqUse wru : wruList){
            WruBbs wrubbs = new WruBbs();
            wrubbs.setWru(wru);
            wrubbs.setBbsList(organDao.bbsList(bankList,
wru.getPerson().getBloodType(), wru.getQuantity()));
            wrubbsList.add(wrubbs);
        }
        mv.addObject("wrubbsList", wrubbsList);
        //mv.addObject("wruList", wruList);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("bmcworkarea", "errorMessage", "error
while get WorkReqUse list");
    }
    mv.setViewName("bmcworkarea");
    return mv;
}

```

```

@RequestMapping(value = "/bmc/info.htm", method = RequestMethod.GET)
public ModelAndView bmcdatapanel(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    DAO.close();
    User sessionuser = (User) session.getAttribute("user");
    User user = null;
    try {
        if(sessionuser != null)
            user = userDao.get((int)sessionuser.getId());
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while get user");
    }
    if (user == null) {
        mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("bmcm")) {
        mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
        mv.addObject("user", new User());
    }
}

```

```

        mv.setViewName("login");
        return mv;
    }
    try {
        Set<BloodBank> bankList =
organDao.getBMC(user.getOrgan().getOrganName()).getBloodBankSet();
        List<BloodBankInventory> bbiList = organDao.bbiList(bankList);
        mv.addObject("bbiList", bbiList);
        mv.addObject("user", user);
        //mv.addObject("wruList", wruList);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("bmcdatapanel", "errorMessage", "error
while get bloodbank list");
    }
    mv.setViewName("bmcdatapanel");
    return mv;
}

```

```

@RequestMapping(value = "/bmc/transfer.htm", method = RequestMethod.POST)
public ModelAndView test(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    int id = Integer.parseInt(request.getParameter("bb"));
    DAO.close();
    User sessionuser = (User) session.getAttribute("user");
    User user = null;
    try {
        if(sessionuser != null)
            user = userDao.get((int)sessionuser.getpID());
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while get user");
    }
    if (user == null) {
        mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("bmcm")) {
        mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
        mv.addObject("user", new User());
    }
}

```

```

        mv.setViewName("login");
        return mv;
    }

    if (id == 0){
        mv.addObject("errorMessage", "Select BloodBank");
        mv.setViewName("error");
        return mv;
    }

    try {
        WorkReqUse                wru                =
wrDao.getWru(Integer.parseInt(request.getParameter("wruid")));
        BloodBank                bb                =
organDao.getBloodBank(Integer.parseInt(request.getParameter("bb")));
        wru.setDestination(bb.getOrganName());
        wru.getUserSet().add(user);
        wru.setStatus("Waiting for blood");
        wrDao.updateWru(wru);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while setting transfer");
    }

    mv.setViewName("redirect:/bmc/home.htm");
    return mv;
}

@RequestMapping(value = "/bmc/report.xls", method = RequestMethod.GET)
public ModelAndView excelReport(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    DAO.close();
    User sessionuser = (User) session.getAttribute("user");
    User user = null;
    try {
        if(sessionuser != null)
            user = userDao.get((int)sessionuser.getpID());
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while get user");
    }
}

```

```

        if (user == null) {
            mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
            mv.addObject("user", new User());
            mv.setViewName("login");
            return mv;
        }
        if (!user.getRole().equals("bmcm")) {
            mv.addObject("errorMessage", "Login as bloodManageCenter staff first");
            mv.addObject("user", new User());
            mv.setViewName("login");
            return mv;
        }
        try {
            Set<BloodBank> bankList =
organDao.getBMC(user.getOrgan().getOrganName()).getBloodBankSet();
            List<BloodBankInventory> bbiList = organDao.bbiList(bankList);
            BMCEExcelView view = new BMCEExcelView();
            view.setUser(user);
            view.setBbiList(bbiList);
            return new ModelAndView(view);
        } catch (OperateException e) {
            System.out.println("Exception: " + e.getMessage());
            return new ModelAndView("bmcdatapanel", "errorMessage", "error
while get bloodbank list");
        }
    }
}

```

CommonActionController

```
package com.jiamin.jiaminfinalp;
```

```
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;
```

```
import javax.servlet.ServletContext;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
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.multipart.commons.CommonsMultipartFile;
import org.springframework.web.servlet.ModelAndView;
```

```
import com.jiamin.dao.OrganizationDAO;
import com.jiamin.dao.UserDAO;
import com.jiamin.dao.WorkRequestDAO;
import com.jiamin.exception.OperateException;
import com.jiamin.pojo.BloodBank;
import com.jiamin.pojo.BloodManageCenter;
import com.jiamin.pojo.Clinic;
import com.jiamin.pojo.Fileupload;
import com.jiamin.pojo.Organization;
import com.jiamin.pojo.User;
import com.jiamin.pojo.VitalSign;
import com.jiamin.pojo.WorkReqDonate;
import com.jiamin.pojo.WorkReqUse;
import com.jiamin.pojo.WorkRequest;
```

```
/**
 * Handles requests for the application home page.
 */
```

@Controller

```
public class CommonActionController {
```

```
    @Autowired
```

```
    @Qualifier("userDao")
```

```
    UserDAO userDao;
```

```
    @Autowired
```

```
    @Qualifier("organDao")
```

```
    OrganizationDAO organDao;
```

```
    @Autowired
```

```
    @Qualifier("wrDao")
```

```
    WorkRequestDAO wrDao;
```

```
    @Autowired
```

```
    ServletContext servletContext;
```

```
    @RequestMapping(value = "/*/deny.htm", method = RequestMethod.GET)
```

```
    public ModelAndView denyAction(HttpServletRequest request) {
```

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

```
        ModelAndView mv = new ModelAndView();
```

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

```
        int id = Integer.parseInt(request.getParameter("wrid"));
```

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

```
            mv.addObject("errorMessage", "Login as staff first");
```

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

```
            mv.setViewName("login");
```

```
            return mv;
```

```
        }
```

```
        if (user.getRole().equals("user")) {
```

```
            mv.addObject("errorMessage", "user have no authority on this action");
```

```
            mv.setViewName("userhome");
```

```
            return mv;
```

```
        }
```

```
        try {
```

```
            WorkReqDonate wrd = wrDao.getWrd(id);
```

```
            if (wrd != null) {
```

```
                boolean flag = false;
```

```
                for (User u : wrd.getUserSet()) {
```

```
                    if (u.getpID() == user.getpID())
```



```

        flag = true;
    }
    boolean authority = false;
    if (user.getRole().equals("nurse")
        && (wrd.getStatus().equals("Tested") ||
wrd.getStatus().equals("Pending(nurse)"))) {
        authority = true;
    } else if (user.getRole().equals("labassistant")
        && (wrd.getStatus().equals("Tested") ||
wrd.getStatus().equals("pending(nurse)"))) {

    }

    if (!(flag && authority)) {
        mv.addObject("errorMessage", "user have no authority on this
action");

        mv.setViewName("error");
        return mv;
    }
} else {
    WorkReqUse wru = wrDao.getWru(id);
    if (wru != null) {
        boolean flag = false;
        for (User u : wru.getUserSet()) {
            if (u.getpID() == user.getpID())
                flag = true;
        }

        boolean authority = false;
        if (user.getRole().equals("nurse")
            && (wru.getStatus().equals("Tested") ||
wru.getStatus().equals("pending(nurse)"))) {
            authority = true;
        } else if (user.getRole().equals("labassistant")
            && (wru.getStatus().equals("Tested") ||
wru.getStatus().equals("pending(nurse)"))) {

        }

        if (!(flag && authority)) {
            mv.addObject("errorMessage", "user have no authority on
this action");

            mv.setViewName("error");
            return mv;

```

```

        }
    } else {
        mv.addObject("errorMessage", "no such request");
        mv.setViewName("error");
        return mv;
    }
}
WorkRequest wr = wrDao.getWr(id);
wr.setStatus("denied");
SimpleDateFormat df = new SimpleDateFormat("MM-dd-yyyy
HH:mm:ss");
wr.setSolveDate(df.format(new Date()));
wrDao.updateWr(wr);
if (user.getRole().equals("nurse")) {
    mv.setViewName("redirect:/nurse/home.htm");
} else if (user.getRole().equals("labassistant")) {
    mv.setViewName("redirect:/labassistant/home.htm");
}

} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("error", "errorMessage", "error while get
workquest list");
}
return mv;
}

```

```

@RequestMapping(value = "/*/assigntome.htm", method = RequestMethod.GET)
public ModelAndView assigntomeAction(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    User user = (User) session.getAttribute("user");
    int id = Integer.parseInt(request.getParameter("wrid"));
    if (user == null) {
        mv.addObject("errorMessage", "Login as staff first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (user.getRole().equals("user")) {
        mv.addObject("errorMessage", "user have no authority on this action");
        mv.setViewName("userhome");
        return mv;
    }
}

```

```

try {
    WorkRequest wr = wrDao.getWr(id);
    if (user.getRole().equals("nurse") && wr.getStatus().equals("Request
sent")) {
        String type = wr.getMessage();
        if (type.equals("Donate")) {
            WorkReqDonate wrd = wrDao.getWrd(id);
            wrd.setStatus("Pending(nurse)");
            wrd.getUserSet().add(user);
            wrDao.updateWrd(wrd);
        } else {
            WorkReqUse wru = wrDao.getWru(id);
            wru.setStatus("Pending(nurse)");
            wru.getUserSet().add(user);
            wrDao.updateWru(wru);
        }
        mv.setViewName("redirect:/nurse/home.htm");
    } else if (user.getRole().equals("labassistant") &&
wr.getStatus().equals("Waiting for test")) {
        WorkReqDonate wrd = wrDao.getWrd(id);
        wrd.setStatus("Pending(labassistant)");
        wrd.getUserSet().add(user);
        wrDao.updateWrd(wrd);
        mv.setViewName("redirect:/lab/home.htm");
    } else if (user.getRole().equals("deliver")
&& (wr.getStatus().equals("Waiting for transport") ||
wr.getStatus().equals("Waiting for blood"))) {
        String type = wr.getMessage();
        if (type.equals("Donate")) {
            WorkReqDonate wrd = wrDao.getWrd(id);
            wrd.setStatus("Pending(deliver)");
            wrd.getUserSet().add(user);
            wrDao.updateWrd(wrd);
        } else {
            WorkReqUse wru = wrDao.getWru(id);
            wru.setStatus("Pending(deliver)");
            wru.getUserSet().add(user);
            wrDao.updateWru(wru);
        }
        mv.setViewName("redirect:/deliver/home.htm");
    }
} catch (OperateException e) {

```

```

        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("error", "errorMessage", "error while get
workquest list");
    }
    return mv;
}

```

```

@RequestMapping(value = "/*/viewuser.htm", method = RequestMethod.GET)
public ModelAndView viewUser(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    User user = (User) session.getAttribute("user");
    if (user == null) {
        mv.addObject("errorMessage", "Login as staff first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (user.getRole().equals("user")) {
        mv.addObject("errorMessage", "user have no authority on this action");
        mv.setViewName("userhome");
        return mv;
    }

    try {
        int id = Integer.parseInt(request.getParameter("pid"));
        User u = (User) userDao.get(id);

        List<WorkReqDonate> wrdList = u.getDonateQueue();
        List<WorkReqUse> wruList = u.getUseQueue();
        List<VitalSign> vsList = u.getVitalSignHistory();

        int i = 0;
        if (user.getRole().equals("admin")) {
            i = 1;
            mv.addObject("usertype", i);
        } else if (user.getRole().equals("user")) {
            i = 2;
            mv.addObject("usertype", i);
        } else if (user.getRole().equals("nurse")) {
            i = 3;
            mv.addObject("usertype", i);
        } else {
            mv.addObject("usertype", i);
        }
    }
}

```

```

    }
    mv.addObject("idforexcel",id);
    mv.addObject("uwrdList", wrdList);
    mv.addObject("uwruList", wruList);
    mv.addObject("vsList", vsList);
} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("userhistory", "errorMessage", "error while
user data");
}

```

```

    mv.setViewName("userhistory");
    return mv;
}

```

```

@RequestMapping(value = "/uploadavator.htm", method = RequestMethod.POST)
public ModelAndView userhome(HttpServletRequest request,
@ModelAttribute("fileupload") Fileupload fileupload) {
    HttpSession session = (HttpSession) request.getSession();
    User user = (User) session.getAttribute("user");
    ModelAndView mv = new ModelAndView();
    if (user == null) {
        mv.addObject("errorMessage", "Login as user first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
}

```

```

try {
    File directory;
    String check = File.separator;
    String path = null;
    if (check.equalsIgnoreCase("\\")) {
        path = servletContext.getRealPath("").replace("build\\", "");
        System.out.println(servletContext.getRealPath(""));
        System.out.println(servletContext.getContextPath());
    }

    if (check.equalsIgnoreCase("/")) {
        path = servletContext.getRealPath("").replace("build/", "");
        path += "/";
    }
    directory = new File(path + "\\resources\\" + user.getID());
    boolean temp = directory.exists();
}

```

```

        if (!temp) {
            temp = directory.mkdir();
        }
        if (temp) {
            CommonsMultipartFile photoInMemory = fileupload.getPhoto();

            String fileName = photoInMemory.getOriginalFilename();
            // could generate file names as well

            File localFile = new File(directory.getPath(), fileName);

            // move the file from memory to the file

            photoInMemory.transferTo(localFile);
            user.setAvatorpath("resources/" + user.getpID()+"/"+fileName);
            System.out.println("File is stored at " + localFile.getPath());
            userDao.update(user);
        } else {
            System.out.println("Failed to create directory!");
        }

    } catch (IllegalStateException e) {
        System.out.println("*** IllegalStateException: " + e.getMessage());
    } catch (IOException e) {
        // TODO Auto-generated catch block
        System.out.println("*** IOException: " + e.getMessage());
    } catch (OperateException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    mv.setViewName("redirect:/user/profile.htm");
    return mv;
}

}

```

DeliverController

```
package com.jiamin.jiaminfinalp;
```

```
import java.text.SimpleDateFormat;
```

```
import java.util.ArrayList;
```

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

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

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

```
import javax.servlet.http.HttpSession;
```

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

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

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

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

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

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

```
import com.jiamin.dao.DAO;
```

```
import com.jiamin.dao.InfoDAO;
```

```
import com.jiamin.dao.OrganizationDAO;
```

```
import com.jiamin.dao.UserDAO;
```

```
import com.jiamin.dao.WorkRequestDAO;
```

```
import com.jiamin.exception.OperateException;
```

```
import com.jiamin.pojo.Blood;
```

```
import com.jiamin.pojo.BloodBank;
```

```
import com.jiamin.pojo.User;
```

```
import com.jiamin.pojo.WorkReqDonate;
```

```
import com.jiamin.pojo.WorkReqUse;
```

```
import com.jiamin.pojo.WorkRequest;
```

```
/**
```

```
 * Handles requests for the application home page.
```

```
 */
```

```
@Controller
```

```
public class DeliverController {
```

```
    @Autowired
```

```
    @Qualifier("userDao")
```

```
    UserDAO userDao;
```

```
    @Autowired
```

```
    @Qualifier("organDao")
```

```
OrganizationDAO organDao;
```

```
@Autowired
```

```
@Qualifier("wrDao")
```

```
WorkRequestDAO wrDao;
```

```
@Autowired
```

```
@Qualifier("infoDao")
```

```
InfoDAO infoDao;
```

```
@RequestMapping(value = "/deliver/home.htm", method = RequestMethod.GET)
```

```
public ModelAndView labStation(HttpServletRequest request) {
```

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

```
    ModelAndView mv = new ModelAndView();
```

```
    DAO.close();
```

```
    User sessionuser = (User) session.getAttribute("user");
```

```
    User user = null;
```

```
    try {
```

```
        if(sessionuser != null)
```

```
            user = userDao.get((int)sessionuser.getId());
```

```
    } catch (OperateException e) {
```

```
        System.out.println("Exception: " + e.getMessage());
```

```
        return new ModelAndView("nurseworkarea", "errorMessage", "error
```

```
while get organization list");
```

```
    }
```

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

```
        mv.addObject("errorMessage", "Login as distribution staff first");
```

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

```
        mv.setViewName("login");
```

```
        return mv;
```

```
    }
```

```
    if (!user.getRole().equals("deliver")) {
```

```
        mv.addObject("errorMessage", "Login as distribution staff first");
```

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

```
        mv.setViewName("login");
```

```
        return mv;
```

```
    }
```

```
    try {
```

```
        List<WorkReqDonate> wrdList = wrDao.wrdListByStaSi("Pending(deliver)", user);
```

```
        List<WorkReqUse> wruList = wrDao.wruListByStaSi("Pending(deliver)",
```

```
user);
```

```
        List<WorkRequest> wrList = wrDao.wrListByBiSta("Waiting for transport",
```

```
"Waiting for blood");
```



```

        mv.addObject("wrList", wrList);
        mv.addObject("wrdList", wrdList);
        mv.addObject("wruList", wruList);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("labworkarea", "errorMessage", "error while
get WorkReqDonate list");
    }
    mv.setViewName("deliverworkarea");
    return mv;
}

```

```

@RequestMapping(value = "/deliver/deliver.htm", method =
RequestMethod.GET)

```

```

    public ModelAndView test(HttpServletRequest request) {
        HttpSession session = (HttpSession) request.getSession();
        ModelAndView mv = new ModelAndView();
        int id = Integer.parseInt(request.getParameter("wrid"));
        DAO.close();
        User sessionuser = (User) session.getAttribute("user");
        User user;
        try {
            user = userDao.get((int)sessionuser.getpID());
        } catch (OperateException e) {
            System.out.println("Exception: " + e.getMessage());
            return new ModelAndView("nurseworkarea", "errorMessage", "error
while get organization list");
        }
        if (user == null) {
            mv.addObject("errorMessage", "Login as distribution staff first");
            mv.addObject("user", new User());
            mv.setViewName("login");
            return mv;
        }
        if (!user.getRole().equals("deliver")) {
            mv.addObject("errorMessage", "Login as distribution staff first");
            mv.addObject("user", new User());
            mv.setViewName("login");
            return mv;
        }
        try {
            WorkRequest wr = wrDao.getWr(id);
            if (wr != null){
                String type = wr.getMessage();
            }
        }
    }
}

```

```

        if(type.equals("Donate")){
            System.out.println("donate deliver");
            WorkReqDonate wrd = wrDao.getWrd(id);
            boolean flag = false;
            for (User u : wrd.getUserSet()){
                if (u.getpID() == user.getpID())
                    flag = true;
            }
            boolean authority = false;
            if      (user.getRole().equals("deliver")      &&
wrd.getStatus().equals("Pending(deliver)")){
                authority = true;
            }

            if (!(flag && authority)){
                mv.addObject("errorMessage", "user have no authority on
this action");

                mv.setViewName("error");
                return mv;
            }

            Blood blood = wrd.getBlood();

            blood.setBloodBank(organDao.getBloodBank(wrd.getDestination()));
            SimpleDateFormat df = new SimpleDateFormat("MM-dd-yyyy
HH:mm:ss");

            wrd.setSolveDate(df.format(new Date()));
            wrd.setStatus("Stored");
            infoDao.update(blood);
            wrDao.updateWrd(wrd);
        }else{
            System.out.println("use deliver");
            WorkReqUse wru = wrDao.getWru(id);
            boolean flag = false;
            for (User u : wru.getUserSet()){
                if (u.getpID() == user.getpID())
                    flag = true;
            }
            boolean authority = false;
            if      (user.getRole().equals("deliver")      &&
wru.getStatus().equals("Pending(deliver)")){
                authority = true;
            }
        }
    }
}

```

```

        if (!(flag && authority)){
            mv.addObject("errorMessage", "user have no authority on
this action");

            mv.setViewName("error");
            return mv;
        }
        BloodBank bb = organDao.getBloodBank(wru.getDestination());

        List<Blood> bloodlist = new ArrayList<Blood>();
        wru.setUseBloodList(bloodlist);
        wru.setStatus("Delivered");
        int amount = 0;
        for (Blood blood : bb.getBloodList()){

            if(wru.getPerson().getBloodType().equals(blood.getBloodType()))
                if(amount < wru.getQuantity()){
                    blood.setBloodBank(null);
                    wru.getUseBloodList().add(blood);
                    blood.setWorkReqUse(wru);
                    infoDao.update(blood);
                    amount += blood.getVolum();
                }
            }
            System.out.println(amount);
            wrDao.updateWru(wru);
        }

    }else {
        mv.addObject("errorMessage", "no such workrequest");
        mv.setViewName("error");
        return mv;
    }
} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("redirect:/deliver/home.htm",
"errorMessage", "error while get organization list");
}
mv.setViewName("redirect:/deliver/home.htm");
return mv;
}

}

```

LabController

```
package com.jiamin.jiaminfinalp;
```

```
import java.text.SimpleDateFormat;
```

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

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

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

```
import javax.servlet.http.HttpSession;
```

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

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

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

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

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

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

```
import com.jiamin.dao.DAO;
```

```
import com.jiamin.dao.InfoDAO;
```

```
import com.jiamin.dao.OrganizationDAO;
```

```
import com.jiamin.dao.UserDAO;
```

```
import com.jiamin.dao.WorkRequestDAO;
```

```
import com.jiamin.exception.OperateException;
```

```
import com.jiamin.pojo.User;
```

```
import com.jiamin.pojo.VitalSign;
```

```
import com.jiamin.pojo.WorkReqDonate;
```

```
import com.jiamin.tools.RandomGenerateTool;
```

```
/**
```

```
 * Handles requests for the application home page.
```

```
 */
```

```
@Controller
```

```
public class LabController {
```

```
    @Autowired
```

```
    @Qualifier("userDao")
```

```
    UserDAO userDao;
```

```
    @Autowired
```

```
    @Qualifier("organDao")
```

```
    OrganizationDAO organDao;
```

```
    @Autowired
```

```

@Qualifier("wrDao")
WorkRequestDAO wrDao;

@Autowired
@Qualifier("infoDao")
InfoDAO infoDao;

@RequestMapping(value = "/lab/home.htm", method = RequestMethod.GET)
public ModelAndView labStation(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    DAO.close();
    User sessionuser = (User) session.getAttribute("user");
    User user = null;
    try {
        if(sessionuser != null)
            user = userDao.get((int)sessionuser.getId());
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while get organization list");
    }
    if (user == null) {
        mv.addObject("errorMessage", "Login as labassistant first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("labassistant")) {
        mv.addObject("errorMessage", "Login as labassistant first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        List<WorkReqDonate> wrsList =
wrDao.wrdListByBiStaDesUser("Pending(labassistant)","Test",
user.getOrgan().getOrganName(), user);
        List<WorkReqDonate> wrdList = wrDao.wrdListByStaDes("Waiting for
test", user.getOrgan().getOrganName());
        mv.addObject("lwrList", wrdList);
        mv.addObject("lwrsList", wrsList);
        //System.out.println(wrsList.get(0).getStatus());
    } catch (OperateException e) {

```

```

        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("labworkarea", "errorMessage", "error while
get WorkReqDonate list");
    }
    mv.setViewName("labworkarea");
    return mv;
}

```

```

@RequestMapping(value = "/lab/test.htm", method = RequestMethod.GET)
public ModelAndView test(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    int id = Integer.parseInt(request.getParameter("wrid"));
    User user = (User) session.getAttribute("user");
    if (user == null) {
        mv.addObject("errorMessage", "Login as labassistant first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("labassistant")) {
        mv.addObject("errorMessage", "Login as labassistant first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        WorkReqDonate wrd = wrDao.getWrd(id);
        if (wrd != null){
            boolean flag = false;
            for (User u : wrd.getUserSet()){
                if (u.getpID() == user.getpID())
                    flag = true;
            }
            boolean authority = false;
            if (user.getRole().equals("labassistant") &&
wrd.getStatus().equals("Pending(labassistant)")){
                authority = true;
            }

            if (!(flag && authority)){
                mv.addObject("errorMessage", "user have no authority on this
action");
                mv.setViewName("error");
            }
        }
    }
}

```

```

        return mv;
    }
} else {
    mv.addObject("errorMessage", "no such workrequest");
    mv.setViewName("redirect:/lab/home.htm");
    return mv;
}
VitalSign vs = new VitalSign();
SimpleDateFormat df = new SimpleDateFormat("MM-dd-yyyy
HH:mm:ss");
vs.setDate(df.format(new Date()));
vs.setBloodtype(wrd.getPerson().getBloodType());
RandomGenerateTool rgt = new RandomGenerateTool();
vs.setInfection(rgt.randill());
vs.setTempCondition(rgt.randill());
if(wrd.getPerson().getVitalSignHistory().size() == 0){
    vs.setHemoglobin(rgt.randHemo());
    vs.setDiabetes(rgt.randDiabetes());
    vs.setPermCondition(rgt.randill());
} else {
    VitalSign pvs = wrd.getPerson().getVitalSignHistory().get(0);
    vs.setHemoglobin(pvs.getHemoglobin());
    vs.setDiabetes(pvs.getDiabetes());
    vs.setPermCondition(pvs.getPermCondition());
}
if ((vs.getDiabetes().equals("No") &&
vs.getHemoglobin().equals("Normal") && vs.getInfection().equals("No")
&&vs.getPermCondition().equals("No") && vs.getTempCondition().equals("No"))
    vs.setIsHealthy("Yes");
else
    vs.setIsHealthy("No");
vs.setPerson(wrd.getPerson());
wrd.setStatus("Test");
wrDao.updateWrd(wrd);
infoDao.createVitalSign(vs);
} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("redirect:/lab/home.htm", "errorMessage",
"error while get organization list");
}
mv.setViewName("redirect:/lab/home.htm");
return mv;
}

```

```

@RequestMapping(value = "/lab/sendback.htm", method = RequestMethod.GET)
public ModelAndView sendReport(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    int id = Integer.parseInt(request.getParameter("wrid"));
    User user = (User) session.getAttribute("user");
    if (user == null) {
        mv.addObject("errorMessage", "Login as labassistant first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("labassistant")) {
        mv.addObject("errorMessage", "Login as labassistant first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        WorkReqDonate wrd = wrDao.getWrd(id);
        if (wrd != null){
            boolean flag = false;
            for (User u : wrd.getUserSet()){
                if (u.getpID() == user.getpID())
                    flag = true;
            }
            boolean authority = false;
            if (user.getRole().equals("labassistant") &&
wrd.getStatus().equals("Test")){
                authority = true;
            }

            if (!(flag && authority)){
                mv.addObject("errorMessage", "user have no authority on this
action");

                mv.setViewName("error");
                return mv;
            }
        }
    } else {
        mv.addObject("errorMessage", "no such workrequest");
        mv.setViewName("redirect:/lab/home.htm");
        return mv;
    }
    wrd.setStatus("Tested");
}

```



```
        wrDao.updateWrd(wrd);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("redirect:/lab/home.htm", "errorMessage",
"error while get organization list");
    }
    mv.setViewName("redirect:/lab/home.htm");
    return mv;
}

}
```

LoginController

```
package com.jiamin.jiaminfinalp;
```

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

```
import javax.servlet.http.HttpSession;
```

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

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

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

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

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

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

```
import org.springframework.validation.BindingResult;
```

```
import org.springframework.web.bind.WebDataBinder;
```

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

```
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 com.jiamin.dao.UserDAO;
```

```
import com.jiamin.exception.OperateException;
```

```
import com.jiamin.validator.*;
```

```
import com.jiamin.pojo.User;
```

```
/**
```

```
 * Handles requests for the application home page.
```

```
 */
```

```
@Controller
```

```
public class LoginController {
```

```
    @Autowired
```

```
    @Qualifier("loginValidator")
```

```
    LoginValidator validator;
```

```
    @Autowired
```

```
    @Qualifier("userDao")
```

```
    UserDAO userDao = new UserDAO();
```

```
    @InitBinder
```

```
    private void initBinder(WebDataBinder binder) {
```

```
        binder.setValidator(validator);
```

```
    }
```

```

@RequestMapping(value = "/login.htm", method = RequestMethod.GET)
public ModelAndView loginPage() {
    return new ModelAndView("login", "user", new User());
}

@RequestMapping(value = "/login.htm", method = RequestMethod.POST)
public ModelAndView loginCheck(HttpServletRequest request,
@ModelAttribute("user") User user, BindingResult result)
    throws Exception {
    ModelAndView mv = new ModelAndView();
    validator.validate(user, result);

    if (result.hasErrors()) {
        return new ModelAndView("login", "user", user);
    }
    HttpSession session = (HttpSession) request.getSession();

    try {

        System.out.print("loginUser");

        User u = userDao.get(request.getParameter("username"),
request.getParameter("password"));

        if (u == null) {
            System.out.println("UserName/Password does not exist");
            return new ModelAndView("login", "errorMessage",
"UserName/Password does not match, try again");
        }

        session.setAttribute("user", u);
        if (u.getRole().equals("user")) {
            mv.setViewName("userhome");
            System.out.println("login as user");
        } else if (u.getRole().equals("admin")) {
            return new ModelAndView("redirect:/admin/home.htm");
        } else if (u.getRole().equals("nurse")) {
            return new ModelAndView("redirect:/nurse/home.htm");
        } else if (u.getRole().equals("labassistant")) {
            return new ModelAndView("redirect:/lab/home.htm");
        } else if (u.getRole().equals("deliver")) {
            return new ModelAndView("redirect:/deliver/home.htm");
        } else if (u.getRole().equals("bmcm")) {
            return new ModelAndView("redirect:/bmc/home.htm");
        }
    }
}

```

```

    }

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

@RequestMapping(value = "/forget.htm", method = RequestMethod.GET)
public ModelAndView fogetPage() {
    return new ModelAndView("forgetpassword");
}

@RequestMapping(value = "/forget.htm", method = RequestMethod.POST)
public ModelAndView fogetCheck(HttpServletRequest request) throws Exception
{
    ModelAndView mv = new ModelAndView();
    try {

        User u = userDao.getreset(request.getParameter("username"),
request.getParameter("email"));

        if (u == null) {
            System.out.println("UserName/Email does not match");
            return new ModelAndView("forgetpassword", "errorMessage",
"UserName/Email does not match, try again");
        }
        Email email = new SimpleEmail();
        email.setHostName("smtp.googlemail.com");
        email.setSmtpport(465);
        email.setAuthenticator(new
DefaultAuthenticator("xdweasdfzxc@gmail.com", "sjm19930909001245"));
        email.setSSLonConnect(true);
        email.setFrom(request.getParameter("email"));
        email.setSubject("reset password");
        email.setMsg("Here is the link to reset your password:
http://localhost:8080/jiaminfinalp/resetpassword.htm?GsiL6FQi-
JXwPOS5xa*Gkscm5AUvMk53HgQNU="
+ u.getID() + "&5sER5t*nC8f=5QA/pmQSOtD");
        email.addTo(request.getParameter("email"));
        email.send();
        mv.addObject("successMessage", "Send reset email successfully");
        mv.setViewName("success");
    }
}

```

```

    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("forgetpassword", "errorMessage", "error
while sending reset email");
    }
    return mv;
}

```

```

@RequestMapping(value = "/resetpassword.htm", method =
RequestMethod.GET)
public ModelAndView resetCheck(HttpServletRequest request) throws Exception
{
    ModelAndView mv = new ModelAndView();
    String id = request.getParameter("GsiL6FQj-
JXwPOS5xa*Gkscm5AUvMk53HgQNU");
    String security = request.getParameter("5sER5t*nC8f");
    if (id == null || !security.equals("5QA/pmQSOtD")) {
        return new ModelAndView("forgetpassword", "errorMessage", "error
while verify security");
    }
    HttpSession session = (HttpSession) request.getSession();
    session.setAttribute("resetpid", id);
    mv.setViewName("resetpassword");
    return mv;
}

```

```

@RequestMapping(value = "/resetpassword.htm", method =
RequestMethod.POST)
public ModelAndView resetpassword(HttpServletRequest request) throws
Exception {
    ModelAndView mv = new ModelAndView();
    HttpSession session = (HttpSession) request.getSession();
    try {
        int id = Integer.parseInt((String) session.getAttribute("resetpid"));
        String password = request.getParameter("password");
        String repassword = request.getParameter("repassword");
        if (password == null || !password.equals(repassword)) {
            return new ModelAndView("resetpassword", "errorMessage", "two
password not match");
        }
        User u = userDao.get(id);
        u.setPassword(password);
        userDao.update(u);
        mv.addObject("successMessage", "reset password successfully");
    }
}

```

```
        mv.setViewName("success");
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("resetpassword", "errorMessage", "error
while verify security");
    }
    return mv;
}

}
```

NurseController

```
package com.jiamin.jiaminfinalp;
```

```
import java.text.SimpleDateFormat;
```

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

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

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

```
import javax.servlet.http.HttpSession;
```

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

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

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

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

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

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

```
import com.jiamin.dao.DAO;
```

```
import com.jiamin.dao.InfoDAO;
```

```
import com.jiamin.dao.OrganizationDAO;
```

```
import com.jiamin.dao.UserDAO;
```

```
import com.jiamin.dao.WorkRequestDAO;
```

```
import com.jiamin.exception.OperateException;
```

```
import com.jiamin.pojo.Blood;
```

```
import com.jiamin.pojo.BloodBank;
```

```
import com.jiamin.pojo.User;
```

```
import com.jiamin.pojo.WorkReqDonate;
```

```
import com.jiamin.pojo.WorkReqUse;
```

```
import com.jiamin.pojo.WorkRequest;
```

```
/**
```

```
 * Handles requests for the application home page.
```

```
 */
```

```
@Controller
```

```
public class NurseController {
```

```
    @Autowired
```

```
    @Qualifier("userDao")
```

```
    UserDAO userDao;
```

```
    @Autowired
```

```
    @Qualifier("organDao")
```

```
    OrganizationDAO organDao;
```

```

@Autowired
@Qualifier("wrDao")
WorkRequestDAO wrDao;

```

```

@Autowired
@Qualifier("infoDao")
InfoDAO infoDao;

```

```

@RequestMapping(value = "/nurse/home.htm", method = RequestMethod.GET)
public ModelAndView nurseStation(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    DAO.close();
    User sessionuser = (User) session.getAttribute("user");
    User user = null;
    try {
        if(sessionuser != null)
            user = userDao.get((int)sessionuser.getId());
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while get organization list");
    }
    if (user == null) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("nurse")) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        //List<WorkReqDonate> wrdList =
wrDao.wrdListByStaDes("pending(nurse)", user.getOrgan().getOrganName());
        List<WorkReqUse> wruList = wrDao.wruListByStaTri("Blood shortage",
"Delivered", "Waiting for blood", user);
        //List<WorkReqDonate> wrdtList = wrDao.wrdListByStaDes("Tested",
user.getOrgan().getOrganName());
        List<WorkReqDonate> wrdtList = wrDao.wrdListByStaDesUser("Tested",

```



```

user.getOrgan().getOrganName(), user);
        List<WorkRequest> wrList = wrDao.wrListByStaDes("Requset sent",
user.getOrgan().getOrganName());
        //List<WorkRequest> wrsList = wrDao.wrListByStaDes("Pending(nurse)",
user.getOrgan().getOrganName());
        List<WorkReqDonate> wrsList =
wrDao.wrdListByStaDesUser("Pending(nurse)", user.getOrgan().getOrganName(),
user);
        List<WorkReqUse> wrsuList =
wrDao.wruListByStaDesUser("Pending(nurse)", user.getOrgan().getOrganName(),
user);

        mv.addObject("nwruList", wruList);
        mv.addObject("nwrsList", wrsList);
        mv.addObject("nwrsuList", wrsuList);
        mv.addObject("nwrdrList", wrdrList);
        mv.addObject("nwrlList", wrList);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while get organization list");
    }
    mv.setViewName("nurseworkarea");
    return mv;
}

```

```

@RequestMapping(value = "/nurse/sendtest.htm", method =
RequestMethod.GET)
public ModelAndView sendTest(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    int id = Integer.parseInt(request.getParameter("wrid"));
    User user = (User) session.getAttribute("user");
    if (user == null) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("nurse")) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
}

```

```

try {
    WorkReqDonate wrd = wrDao.getWrd(id);
    if (wrd != null){
        boolean flag = false;
        for (User u : wrd.getUserSet()){
            if (u.getpID() == user.getpID())
                flag = true;
        }
        boolean authority = false;
        if (user.getRole().equals("nurse") &&
wrd.getStatus().equals("Pending(nurse)")){
            authority = true;
        }

        if (!(flag && authority)){
            mv.addObject("errorMessage", "user have no authority on this
action");

            mv.setViewName("error");
            return mv;
        }
    }else {
        mv.addObject("errorMessage", "no such workrequest");
        mv.setViewName("redirect:/nurse/home.htm");
        return mv;
    }
    wrd.setStatus("Waiting for test");
    wrDao.updateWrd(wrd);
} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("redirect:/nurse/home.htm",
"errorMessage", "error while get organization list");
}
mv.setViewName("redirect:/nurse/home.htm");
return mv;
}

```

```

@RequestMapping(value = "/nurse/drawblood.htm", method =
RequestMethod.GET)

```

```

public ModelAndView drawBlood(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    int id = Integer.parseInt(request.getParameter("wrid"));
    User user = (User) session.getAttribute("user");
    if (user == null) {

```

```

        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("nurse")) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        WorkReqDonate wrd = wrDao.getWrd(id);
        if (wrd != null){
            boolean flag = false;
            for (User u : wrd.getUserSet()){
                if (u.getpID() == user.getpID())
                    flag = true;
            }
            boolean authority = false;
            if (user.getRole().equals("nurse") &&
wrd.getStatus().equals("Tested")){
                authority = true;
            }

            if (!(flag && authority)){
                mv.addObject("errorMessage", "user have no authority on this
action");

                mv.setViewName("error");
                return mv;
            }
        }else {
            mv.addObject("errorMessage", "no such workrequest");
            mv.setViewName("redirect:/nurse/home.htm");
            return mv;
        }
        Blood blood = new Blood();
        blood.setBloodType(wrd.getPerson().getBloodType());
        blood.setDonor(wrd.getPerson());
        SimpleDateFormat df = new SimpleDateFormat("MM-dd-yyyy
HH:mm:ss");
        blood.setDate(df.format(new Date()));
        blood.setVolum(wrd.getQuantity());
        blood.setWorkReqDonate(wrd);
    }

```

```

        //wrd.setBlood(blood);
        infoDao.createBlood(blood);

        int oid = user.getOrgan().getOid();
        BloodBank bb = organDao.getClinic(oid).getBloodBank();
        wrd.setDestination(bb.getOrganName());
        wrd.setStatus("Waiting for transport");
        wrDao.updateWrd(wrd);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("redirect:/nurse/home.htm",
"errorMessage", "error while get organization list");
    }
    mv.setViewName("redirect:/nurse/home.htm");
    return mv;
}

```

```

@RequestMapping(value = "/nurse/useblood.htm", method =
RequestMethod.GET)
public ModelAndView useBlood(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    int id = Integer.parseInt(request.getParameter("wrid"));
    User user = (User) session.getAttribute("user");
    if (user == null) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("nurse")) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        WorkReqUse wrd = wrDao.getWru(id);
        if (wrd != null){
            boolean flag = false;
            for (User u : wrd.getUserSet()){
                if (u.getpID() == user.getpID())
                    flag = true;
            }

```

```

        boolean authority = false;
        if (user.getRole().equals("nurse") &&
wrd.getStatus().equals("Delivered")){
            authority = true;
        }

        if (!(flag && authority)){
            mv.addObject("errorMessage", "user have no authority on this
action");

            mv.setViewName("error");
            return mv;
        }
    }else {
        mv.addObject("errorMessage", "no such workrequest");
        mv.setViewName("redirect:/nurse/home.htm");
        return mv;
    }
    wrd.setStatus("Used");
    SimpleDateFormat df = new SimpleDateFormat("MM-dd-yyyy
HH:mm:ss");
    wrd.setSolveDate(df.format(new Date()));
    for (Blood blood: wrd.getUseBloodList()){
        blood.setUsePerson(wrd.getPerson());
        infoDao.update(blood);
    }
    wrDao.updateWru(wrd);
} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("redirect:/nurse/home.htm",
"errorMessage", "error while update worrequeuse list");
}
mv.setViewName("redirect:/nurse/home.htm");
return mv;
}

```

```

@RequestMapping(value = "/nurse/sendrequire.htm", method =
RequestMethod.GET)

```

```

public ModelAndView sendRequire(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    int id = Integer.parseInt(request.getParameter("wrid"));
    User user = (User) session.getAttribute("user");
    if (user == null) {
        mv.addObject("errorMessage", "Login as nurse first");
    }
}

```

```

        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    if (!user.getRole().equals("nurse")) {
        mv.addObject("errorMessage", "Login as nurse first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        WorkReqUse wrd = wrDao.getWru(id);
        if (wrd != null){
            boolean flag = false;
            for (User u : wrd.getUserSet()){
                if (u.getpID() == user.getpID())
                    flag = true;
            }
            boolean authority = false;
            if (user.getRole().equals("nurse") &&
wrd.getStatus().equals("Pending(nurse)")){
                authority = true;
            }

            if (!(flag && authority)){
                mv.addObject("errorMessage", "user have no authority on this
action");

                mv.setViewName("error");
                return mv;
            }
        }else {
            mv.addObject("errorMessage", "no such workrequest");
            mv.setViewName("redirect:/nurse/home.htm");
            return mv;
        }
        int oid = user.getOrgan().getOid();
        BloodBank bb = organDao.getClinic(oid).getBloodBank();
        int i = 0;
        for (Blood blood : bb.getBloodList()){
            if (blood.getBloodType().equals(wrd.getPerson().getBloodType()))
                i += blood.getVolum();
        }
        if (i >= wrd.getQuantity()){
            wrd.setStatus("Waiting for blood");

```

```
        wrd.setDestination(bb.getOrganName());
    }else {
        wrd.setStatus("Blood shortage");
        wrd.setDestination(bb.getOrganName());
    }
    wrDao.updateWru(wrd);
} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("redirect:/nurse/home.htm",
"errorMessage", "error while get organization list");
}
mv.setViewName("redirect:/nurse/home.htm");
return mv;
}

}
```

RedirectController

package com.jiamin.jiaminfinalp;

import javax.servlet.http.HttpServletRequest;

import org.springframework.stereotype.Controller;

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

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

import org.springframework.web.servlet.ModelAndView;

import com.jiamin.dao.DAO;

/\*\*

\* Handles requests for the application home page.

\*/

@Controller

public class RedirectController {

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

public String home() {

return "index";

}

@RequestMapping(value = "/index.htm", method = RequestMethod.GET)

public ModelAndView returntoindex() throws Exception {

ModelAndView mv = new ModelAndView();

mv.setViewName("index");

return mv;

}

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

public String logout(HttpServletRequest request) {

DAO.close();

request.getSession().invalidate();

return "index";

}

}



```

RegisterController
package com.jiamin.jiaminfinalp;

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

import org.apache.commons.mail.DefaultAuthenticator;
import org.apache.commons.mail.Email;
import org.apache.commons.mail.SimpleEmail;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;
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 com.jiamin.dao.OrganizationDAO;
import com.jiamin.dao.UserDAO;
import com.jiamin.exception.OperateException;
import com.jiamin.validator.*;
import com.jiamin.pojo.User;

/**
 * Handles requests for the application home page.
 */
@Controller
public class RegisterController {

    @Autowired
    @Qualifier("registerValidator")
    RegisterValidator validator;

    @Autowired
    @Qualifier("userDao")
    UserDAO userDao;

    @Autowired
    @Qualifier("organDao")
    OrganizationDAO organDao;

```

```

@InitBinder
private void initBinder(WebDataBinder binder) {
    binder.setValidator(validator);
}

@RequestMapping(value = "/signup.htm", method = RequestMethod.GET)
public ModelAndView registerPage() {
    ModelAndView mv = new ModelAndView();
    String[] genderlist = new String[]{"Male", "Female"};
    mv.addObject("genderlist", genderlist);
    mv.addObject("user", new User());
    mv.setViewName("register");
    return mv;
}

@RequestMapping(value = "/signup.htm", method = RequestMethod.POST)
public ModelAndView loginCheck(HttpServletRequest request,
@ModelAttribute("user") User user, BindingResult result) throws Exception {
    ModelAndView mv = new ModelAndView();
    validator.validate(user, result);

    if (result.hasErrors()) {
        return new ModelAndView("register", "user", user);
    }

    try {

        System.out.print("registerUser");

        User u = userDao.get(request.getParameter("username"));

        if(u != null){
            System.out.println("username already exists try another one");
            return new ModelAndView("register", "errorMessage", "username
already exists try another one");
        }

        user.setRole("user");
        user.setOrgan(organDao.getOrgan(1));
        userDao.register(user);
        Email email = new SimpleEmail();
        email.setHostName("smtp.googlemail.com");
        email.setSmtPort(465);
    }
}

```

```

        email.setAuthenticator(new
DefaultAuthenticator("xdweasdfzxc@gmail.com", "sjm19930909001245"));
        email.setSSLOnConnect(true);
        email.setFrom(user.getEmail());
        email.setSubject("Hi,"+user.getUsername());
        email.setMsg("Sign up successfully");
        email.addTo(user.getEmail());
        email.send();
        return new ModelAndView("success", "successMessage", "Register
successfully");

    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("register", "errorMessage", "while sign up");
    }
}
}

```

```

UserController
package com.jiamin.jiaminfinalp;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

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

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
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.View;

import com.jiamin.dao.DAO;
import com.jiamin.dao.OrganizationDAO;
import com.jiamin.dao.UserDAO;
import com.jiamin.dao.WorkRequestDAO;
import com.jiamin.exception.OperateException;
import com.jiamin.pojo.Clinic;
import com.jiamin.pojo.Organization;
import com.jiamin.pojo.User;
import com.jiamin.pojo.VitalSign;
import com.jiamin.pojo.WorkReqDonate;
import com.jiamin.pojo.WorkReqUse;

/**
 * Handles requests for the application home page.
 */
@Controller
public class UserController {

    @Autowired
    @Qualifier("userDao")
    UserDAO userDao;

    @Autowired
    @Qualifier("organDao")

```

```
OrganizationDAO organDao;
```

```
@Autowired
```

```
@Qualifier("wrDao")
```

```
WorkRequestDAO wrDao;
```

```
@RequestMapping(value = "/user/home.htm", method = RequestMethod.GET)
```

```
public ModelAndView userhome(HttpServletRequest request) {
```

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

```
    ModelAndView mv = new ModelAndView();
```

```
    if (session.getAttribute("user") == null) {
```

```
        mv.addObject("errorMessage", "Login as user first");
```

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

```
        mv.setViewName("login");
```

```
        return mv;
```

```
    }
```

```
    mv.setViewName("userhome");
```

```
    return mv;
```

```
}
```

```
@RequestMapping(value = "/user/donate.htm", method = RequestMethod.GET)
```

```
public ModelAndView donatePage(HttpServletRequest request) {
```

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

```
    ModelAndView mv = new ModelAndView();
```

```
    if (session.getAttribute("user") == null) {
```

```
        mv.addObject("errorMessage", "Login as user first");
```

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

```
        mv.setViewName("login");
```

```
        return mv;
```

```
    }
```

```
    try {
```

```
        List<Organization> clinicList = organDao.organList("clinic");
```

```
        mv.addObject("clinicList", clinicList);
```

```
    } catch (OperateException e) {
```

```
        System.out.println("Exception: " + e.getMessage());
```

```
        return new ModelAndView("userdonate", "errorMessage", "error while  
get clinic list");
```

```
    }
```

```
    mv.setViewName("userdonate");
```

```
    return mv;
```

```
}
```

```
@RequestMapping(value = "/user/use.htm", method = RequestMethod.GET)
```

```

public ModelAndView usePage(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    if (session.getAttribute("user") == null) {
        mv.addObject("errorMessage", "Login as user first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    try {
        List<Organization> clinicList = organDao.organList("clinic");
        mv.addObject("clinicList", clinicList);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("userdonate", "errorMessage", "error while
get clinic list");
    }
    mv.setViewName("useruse");
    return mv;
}

```

```

@RequestMapping(value = "/user/history.htm", method = RequestMethod.GET)

```

```

public ModelAndView historyPage(HttpServletRequest request) {

```

```

    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    DAO.close();
    User user = (User) session.getAttribute("user");
    if (session.getAttribute("user") == null) {
        mv.addObject("errorMessage", "Login as user first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }

```

```

};

```

```

try{

```

```

    User u = (User)userDao.get((int)user.getpID());

```

```

    List<WorkReqDonate> wrdList = u.getDonateQueue();

```

```

    List<WorkReqUse> wruList = u.getUseQueue();

```

```

    List<VitalSign> vsList = u.getVitalSignHistory();

```

```

    int i = 2;

```

```

    mv.addObject("usertype", i);

```

```

    mv.addObject("idforexcel",(int)user.getpID());

```

```

        mv.addObject("uwrdList", wrdList);
        mv.addObject("uwruList", wruList);
        mv.addObject("vsList", vsList);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("userhistory", "errorMessage", "error while
user data");
    }

```

```

        mv.setViewName("userhistory");
        return mv;
    }

```

```

@RequestMapping(value = "/user/profile.htm", method = RequestMethod.GET)
public ModelAndView profilePage(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    DAO.close();
    User sessionuser = (User) session.getAttribute("user");
    User user = null;
    try {
        if(sessionuser != null)
            user = userDao.get((int)sessionuser.getId());
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("nurseworkarea", "errorMessage", "error
while get organization list");
    }
    if (user == null) {
        mv.addObject("errorMessage", "Login as user first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    int i = 0;
    if(user.getRole().equals("admin")){
        i = 1;
        mv.addObject("usertype", i);
    }else if(user.getRole().equals("user")){
        i = 2;
        mv.addObject("usertype", i);
    }else if(user.getRole().equals("nurse")){
        i = 3;
        mv.addObject("usertype", i);
    }
}

```

```

    }else if(user.getRole().equals("labassistant")){
        i = 4;
        mv.addObject("usertype", i);
    }else if(user.getRole().equals("deliver")){
        i = 5;
        mv.addObject("usertype", i);
    }else if(user.getRole().equals("bmcm")){
        i = 6;
        mv.addObject("usertype", i);
    }else{
        mv.addObject("usertype", i);
    }
    String avatorpath = "resources/default.jpg";
    if(user.getAvatorpath() != null)
        avatorpath = user.getAvatorpath();
    mv.addObject("avator", avatorpath);
    mv.setViewName("userprofile");
    return mv;
}

```

```

@RequestMapping(value = "/user/profile.htm", method = RequestMethod.POST)
public ModelAndView profileUpdate(HttpServletRequest request) {
    HttpSession session = (HttpSession) request.getSession();
    ModelAndView mv = new ModelAndView();
    User user = (User)session.getAttribute("user");
    if (user == null) {
        mv.addObject("errorMessage", "Login as user first");
        mv.addObject("user", new User());
        mv.setViewName("login");
        return mv;
    }
    user.setEmail(request.getParameter("email"));
    user.setPhone(request.getParameter("phone"));

    try {
        System.out.println(user.getEmail());
        userDao.update(user);
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("redirect:/user/profile.htm",
"errorMessage", "error while update profile");
    }
    mv.setViewName("redirect:/user/profile.htm");
    return mv;
}

```



```

    }

    @RequestMapping(value = "/user/updatepassword.htm", method =
RequestMethod.POST)
    public ModelAndView passwordChange(HttpServletRequest request) {
        HttpSession session = (HttpSession) request.getSession();
        ModelAndView mv = new ModelAndView();
        if (session.getAttribute("user") == null) {
            mv.addObject("errorMessage", "Login as user first");
            mv.addObject("user", new User());
            mv.setViewName("login");
            return mv;
        }
        User user = (User) session.getAttribute("user");
        if (!user.getPassword().equals(request.getParameter("cpassword"))){
            return new ModelAndView("redirect:/user/profile.htm",
"errorMessage", "current password is wrong, try again");
        }
        user.setPassword(request.getParameter("npassword"));
        try {
            userDao.update(user);
        } catch (OperateException e) {
            System.out.println("Exception: " + e.getMessage());
            return new ModelAndView("redirect:/user/profile.htm",
"errorMessage", "error while update password");
        }
        mv.setViewName("redirect:/user/profile.htm");
        return mv;
    }

```

```

@RequestMapping(value = "/user/donate.htm", method = RequestMethod.POST)
public ModelAndView donate(HttpServletRequest request) throws Exception {
    ModelAndView mv = new ModelAndView();
    HttpSession session = (HttpSession) request.getSession();

    try {

        System.out.print("user donate");

        Clinic clinic =
organDao.getClinic(Integer.parseInt(request.getParameter("oid")));

        if (clinic == null) {
            System.out.println("Selcet clinic first");

```

```

        return new ModelAndView("userdonate", "errorMessage", "Selcet
clinic first");
    }

```

```

        User user = (User) session.getAttribute("user");

        WorkReqDonate wrd = new WorkReqDonate();
        SimpleDateFormat df = new SimpleDateFormat("MM-dd-yyyy
HH:mm:ss");
        wrd.setRequestDate(df.format(new Date()));
        wrd.setDestination(clinic.getOrganName());
        wrd.setMessage("Donate");
        wrd.setQuantity(Integer.parseInt(request.getParameter("donations")));
        wrd.setStatus("Requset sent");
        wrd.setPerson(user);
        Set<User> uset= new HashSet<User>();
        uset.add(user);
        wrd.setUserSet(uset);
        wrDao.createWrd(wrd);
        mv.setViewName("redirect:/user/history.htm");

```

```

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

```

```

@RequestMapping(value = "/user/use.htm", method = RequestMethod.POST)
public ModelAndView use(HttpServletRequest request) throws Exception {
    ModelAndView mv = new ModelAndView();
    HttpSession session = (HttpSession) request.getSession();

    try {

        System.out.print("user require blood");

        Clinic clinic =
organDao.getClinic(Integer.parseInt(request.getParameter("oid")));

        if (clinic == null) {
            System.out.println("Selcet clinic first");
            return new ModelAndView("userdonate", "errorMessage", "Selcet

```

```

clinic first");
    }

    User user = (User) session.getAttribute("user");

    WorkReqUse wru = new WorkReqUse();
    SimpleDateFormat df = new SimpleDateFormat("MM-dd-yyyy
HH:mm:ss");
    wru.setRequestDate(df.format(new Date()));
    wru.setDestination(clinic.getOrganName());
    wru.setMessage("Use");
    wru.setPerson(user);
    wru.setQuantity(Integer.parseInt(request.getParameter("quantities")));
    wru.setStatus("Requeset sent");
    Set<User> uset= new HashSet<User>();
    uset.add(user);
    wru.setUserSet(uset);
    wrDao.createWru(wru);
    mv.setViewName("redirect:/user/history.htm");
} catch (OperateException e) {
    System.out.println("Exception: " + e.getMessage());
    return new ModelAndView("login", "errorMessage", "error while submit
request");
}
return mv;
}

@RequestMapping(value = "/user/report.xls", method = RequestMethod.GET)
public ModelAndView excelReport(HttpServletRequest request) {
    DAO.close();
    User user = null;
    try {
        user = userDao.get(Integer.parseInt(request.getParameter("pid")));
    } catch (OperateException e) {
        System.out.println("Exception: " + e.getMessage());
        return new ModelAndView("redirect:/user/history.htm",
"errorMessage", "error while get user info");
    }
    ExcelView view = new ExcelView();
    view.setUser(user);
    return new ModelAndView(view);
}
}

```