Assurance Insurance Group – A dynamic and responsive web application which allows transaction between doctor,patients and the insurance management!

Assurance Insurance Group

Web Tools Project SECTION 1

Raghavi Kirouchenaradjou - 001826638

Assurance Insurance Group- Project

Summary

Web Application Name: Assurance Insurance Group

Assurance Insurance Group is a Web application that allows patient, doctor and insurance management admin transactions. It makes life easier for booking an appointment with the Doctor, payment management and other transaction. **It sells traditional and consumer directed health care insurance plans and related services, such as medical**.

Summary of the Functionalities

It provides the following feature: (More details about the functionality in future pages)

1.) Registration – Patient, Doctor
2.) Login – Patient, Doctor and Admin
3.) Admin Login – Login via Web Security
4.) Patient Dashboard
    a. Book an appointment with the Doctor
    b. Patient – Check status of the insurance claimed
    c. Patient – Enroll for the listed Insurance Plan
5.) Doctor Dashboard
    a. Submit Patient Case Sheet
    b. See the history - Patients
6.) Admin Dashboard
    a. Sanction Approval
    b. Create Insurance Plan
7.) Log Off – Patient, Doctor

Technologies Used

UI – Bootstrap,HTML,CSS

Script – Jquery,Java Script.

Used **Validators** for Registration and Insurance Plan Creation

PDF Viewer – Generate Report during Sanction Approval

AJAX– Book an Appointment with Doctor, Enroll for the Insurance Plan, Get More Details in Sanction Approval

HQL

Web Security – Admin Login

Exception Handling – User, Plan, Doctor.

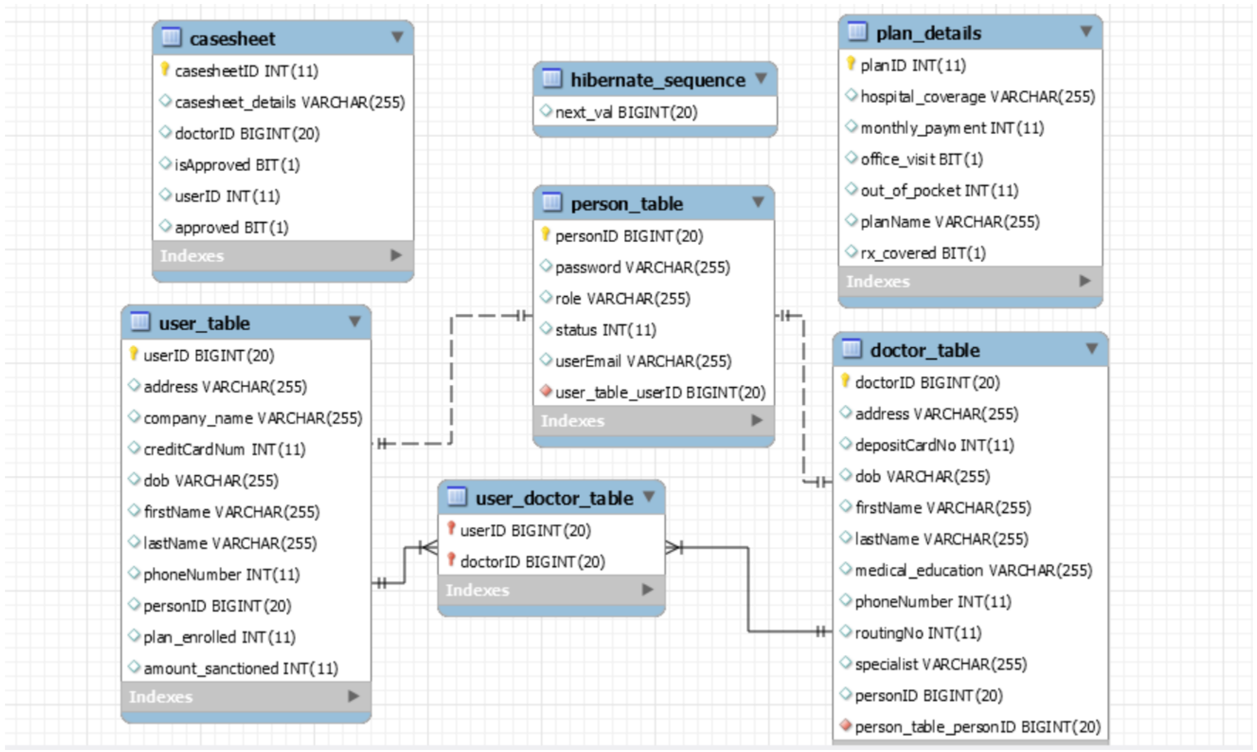
Other Securities – CAPTCHA

Roles

Patient – Has one to one relationship with the Person and has ManyToMany Relationship with the Doctor

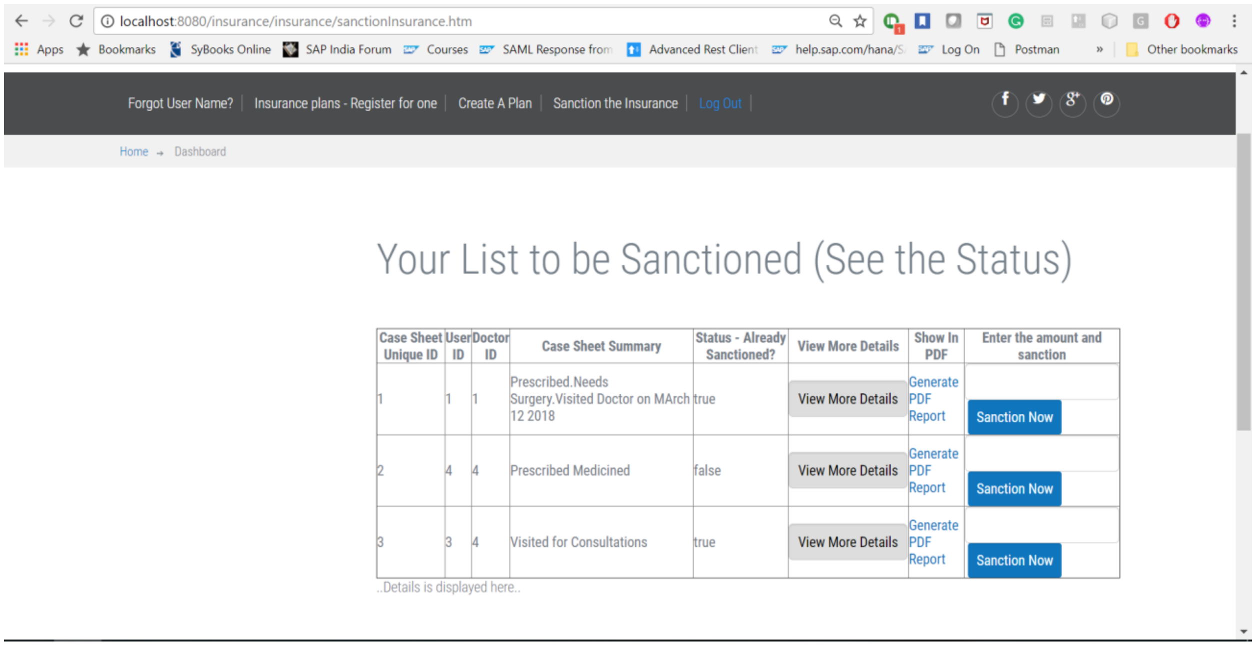
Doctor - Has one to one relationship with the User and has ManyToMany Relationship with the Doctor

Admin – Declarative Web Security – admin is the user role.



Snapshots

Admin Panel – Sanction the Claim – AJAX with JQUERY and PDF Viewer



localhost:8080/insurance/insurance/sanctionInsurance.htm

Forgot User Name? | Insurance plans - Register for one | Create A Plan | Sanction the Insurance | Log Out |

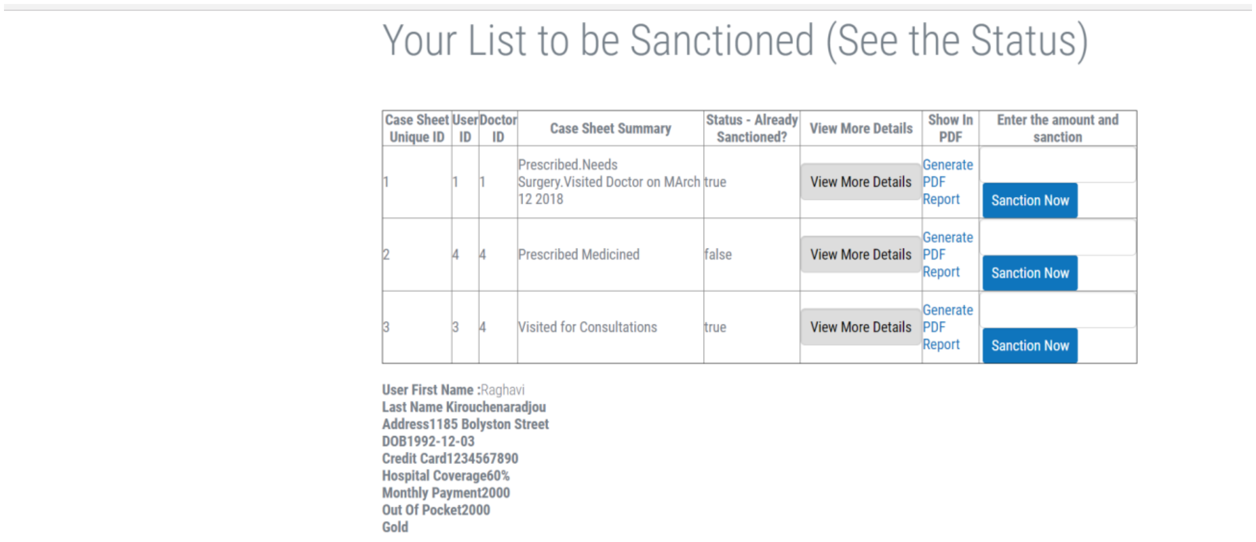
Home → Dashboard

Your List to be Sanctioned (See the Status)

Case Sheet Unique ID	User ID	Doctor ID	Case Sheet Summary	Status - Already Sanctioned?	View More Details	Show In PDF	Enter the amount and sanction
1	1	1	Prescribed.Needs Surgery.Visited Doctor on MArch 12 2018	true	View More Details	Generate PDF Report	Sanction Now
2	4	4	Prescribed Medicined	false	View More Details	Generate PDF Report	Sanction Now
3	3	4	Visited for Consultations	true	View More Details	Generate PDF Report	Sanction Now

..Details is displayed here..

View More Details is an AJAX Call:

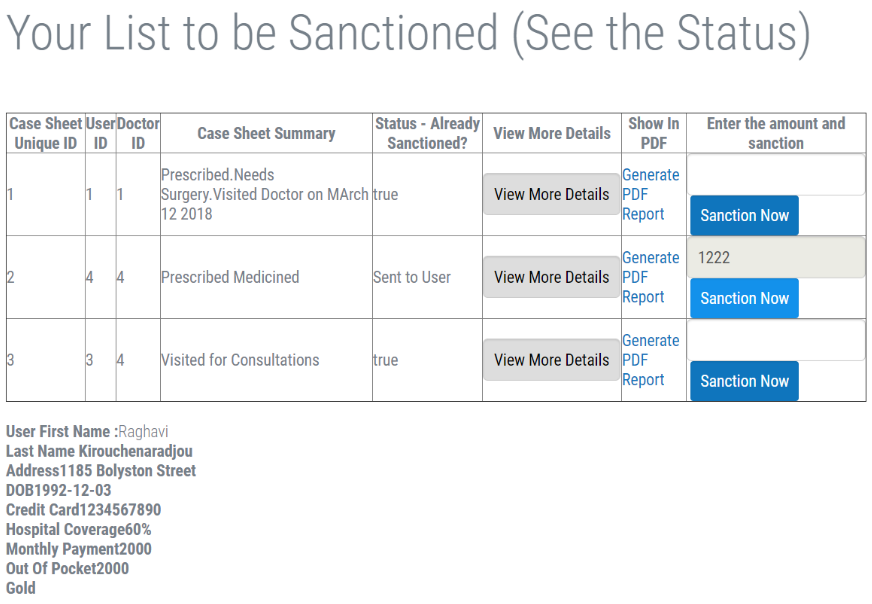


Your List to be Sanctioned (See the Status)

Case Sheet Unique ID	User ID	Doctor ID	Case Sheet Summary	Status - Already Sanctioned?	View More Details	Show In PDF	Enter the amount and sanction
1	1	1	Prescribed.Needs Surgery.Visited Doctor on MArch 12 2018	true	View More Details	Generate PDF Report	Sanction Now
2	4	4	Prescribed Medicined	false	View More Details	Generate PDF Report	Sanction Now
3	3	4	Visited for Consultations	true	View More Details	Generate PDF Report	Sanction Now

User First Name : Raghavi
Last Name Kirouchenaradjou
Address1185 Bolyston Street
DOB1992-12-03
Credit Card1234567890
Hospital Coverage60%
Monthly Payment2000
Out Of Pocket2000
Gold

Sanction Now is also An AJAX CALL which disables the button and updates the status.

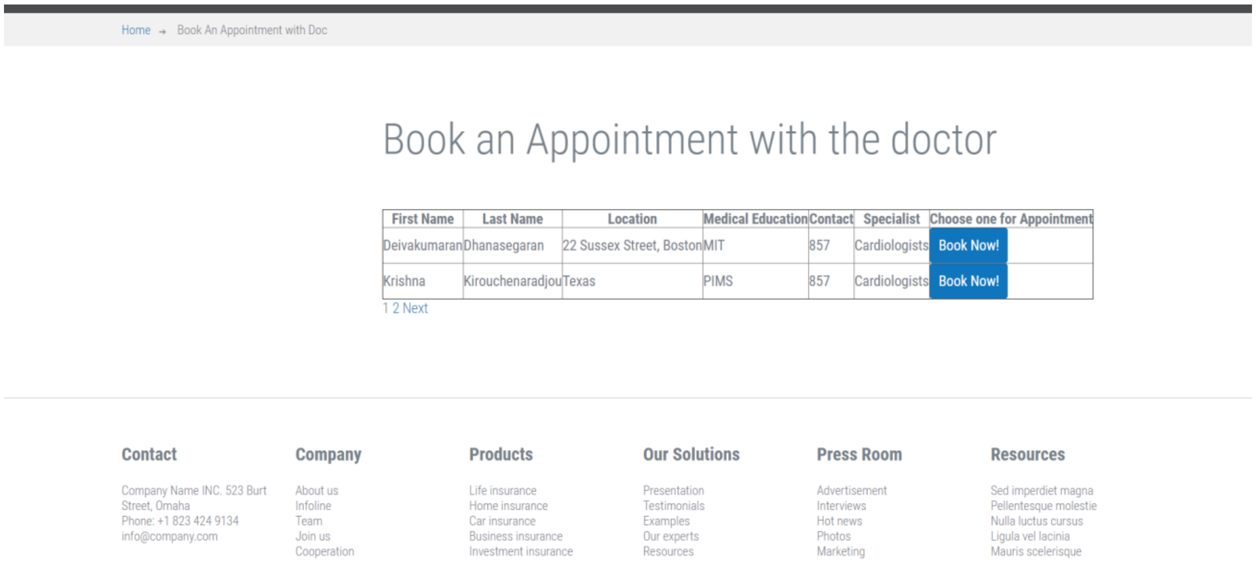


Your List to be Sanctioned (See the Status)

Case Sheet Unique ID	User ID	Doctor ID	Case Sheet Summary	Status - Already Sanctioned?	View More Details	Show In PDF	Enter the amount and sanction
1	1	1	Prescribed.Needs Surgery.Visited Doctor on MArch 12 2018	true	View More Details	Generate PDF Report	Sanction Now
2	4	4	Prescribed Medicined	Sent to User	View More Details	Generate PDF Report	1222 Sanction Now
3	3	4	Visited for Consultations	true	View More Details	Generate PDF Report	Sanction Now

User First Name : Raghavi
Last Name Kirouchenaradjou
Address1185 Bolyston Street
DOB1992-12-03
Credit Card1234567890
Hospital Coverage60%
Monthly Payment2000
Out Of Pocket2000
Gold

Patient – Book an Appointment with Doctor – Pagination



Home → Book An Appointment with Doc

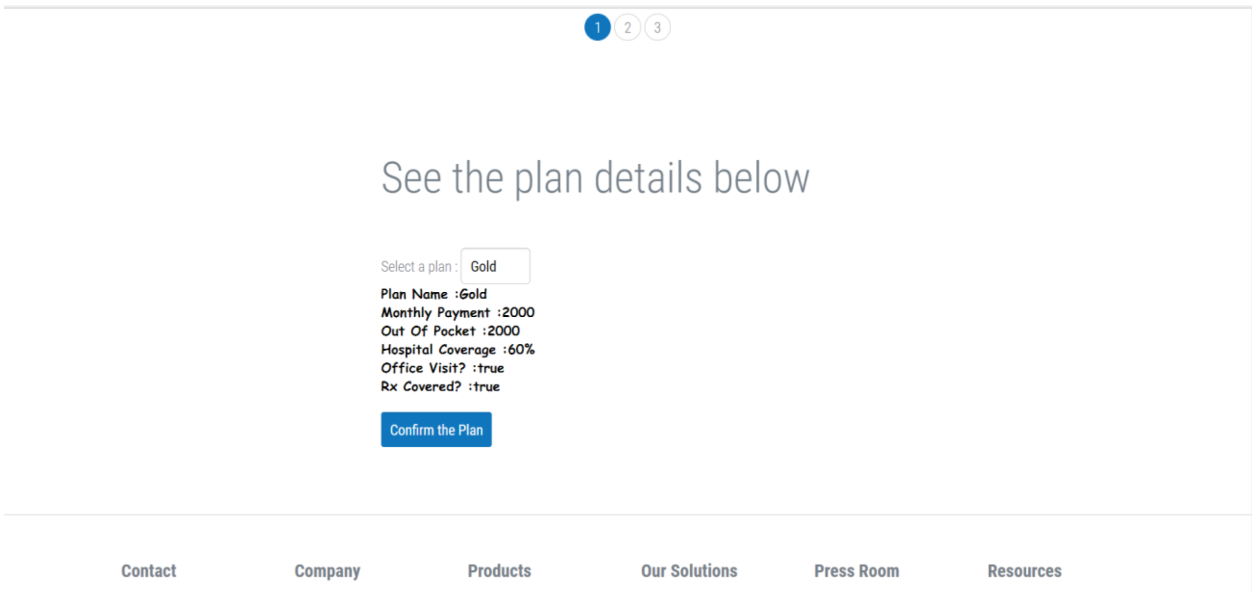
Book an Appointment with the doctor

First Name	Last Name	Location	Medical Education	Contact	Specialist	Choose one for Appointment
Deivakumaran	Dhanasegaran	22 Sussex Street, Boston	MIT	857	Cardiologists	Book Now!
Krishna	Kirouchenaradjou	Texas	PIMS	857	Cardiologists	Book Now!

1 2 Next

Contact	Company	Products	Our Solutions	Press Room	Resources
Company Name INC. 523 Burt Street, Omaha	About us	Life insurance	Presentation	Advertisement	Sed imperdiet magna
Phone: +1 823 424 9134	Infoline	Home insurance	Testimonials	Interviews	Pellentesque molestie
info@company.com	Team	Car insurance	Examples	Hot news	Nulla luctus cursus
	Join us	Business insurance	Our experts	Photos	Ligula vel lacinia
	Cooperation	Investment insurance	Resources	Marketing	Mauris scelerisque

Patient – Confirm/Enroll Plan – AJAX to get the plan details

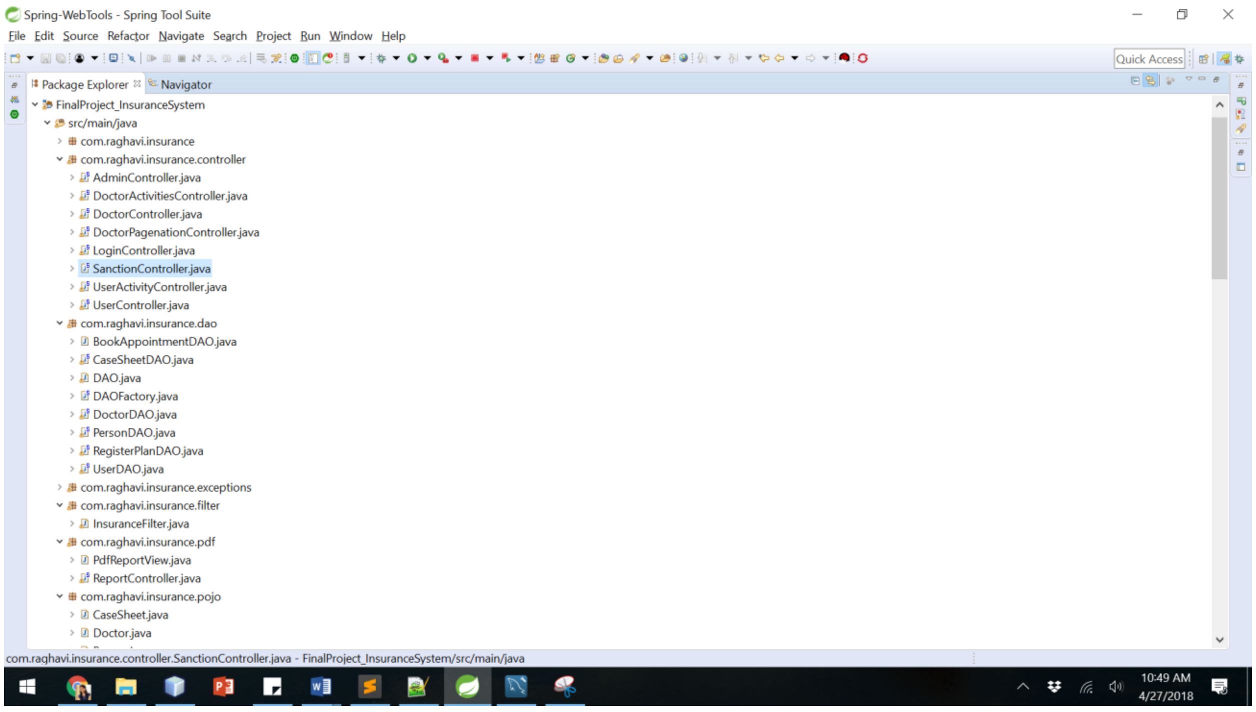


See the plan details below

Select a plan : Gold

Plan Name :Gold
Monthly Payment :2000
Out Of Pocket :2000
Hospital Coverage :60%
Office Visit? :true
Rx Covered? :true

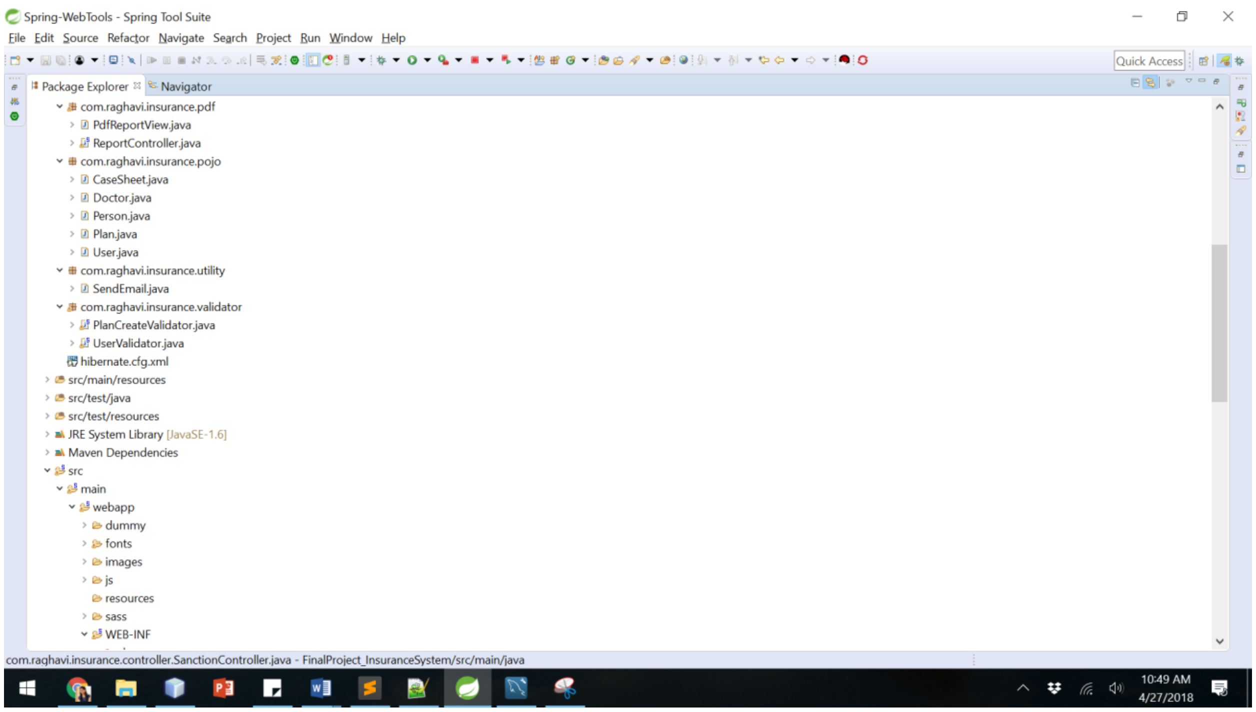
Project Structure


Spring-WebTools - Spring Tool Suite
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Package Explorer
Navigator

FinalProject_InsuranceSystem
src/main/java
com.raghavi.insurance
com.raghavi.insurance.controller
AdminController.java
DoctorActivitiesController.java
DoctorController.java
DoctorPagenationController.java
LoginController.java
SanctionController.java
UserActivityController.java
UserController.java
com.raghavi.insurance.dao
BookAppointmentDAO.java
CaseSheetDAO.java
DAO.java
DAOFactory.java
DoctorDAO.java
PersonDAO.java
RegisterPlanDAO.java
UserDAO.java
com.raghavi.insurance.exceptions
com.raghavi.insurance.filter
InsuranceFilter.java
com.raghavi.insurance.pdf
PdfReportView.java
ReportController.java
com.raghavi.insurance.pojo
CaseSheet.java
Doctor.java

com.raghavi.insurance.controller.SanctionController.java - FinalProject_InsuranceSystem/src/main/java

10:49 AM
4/27/2018

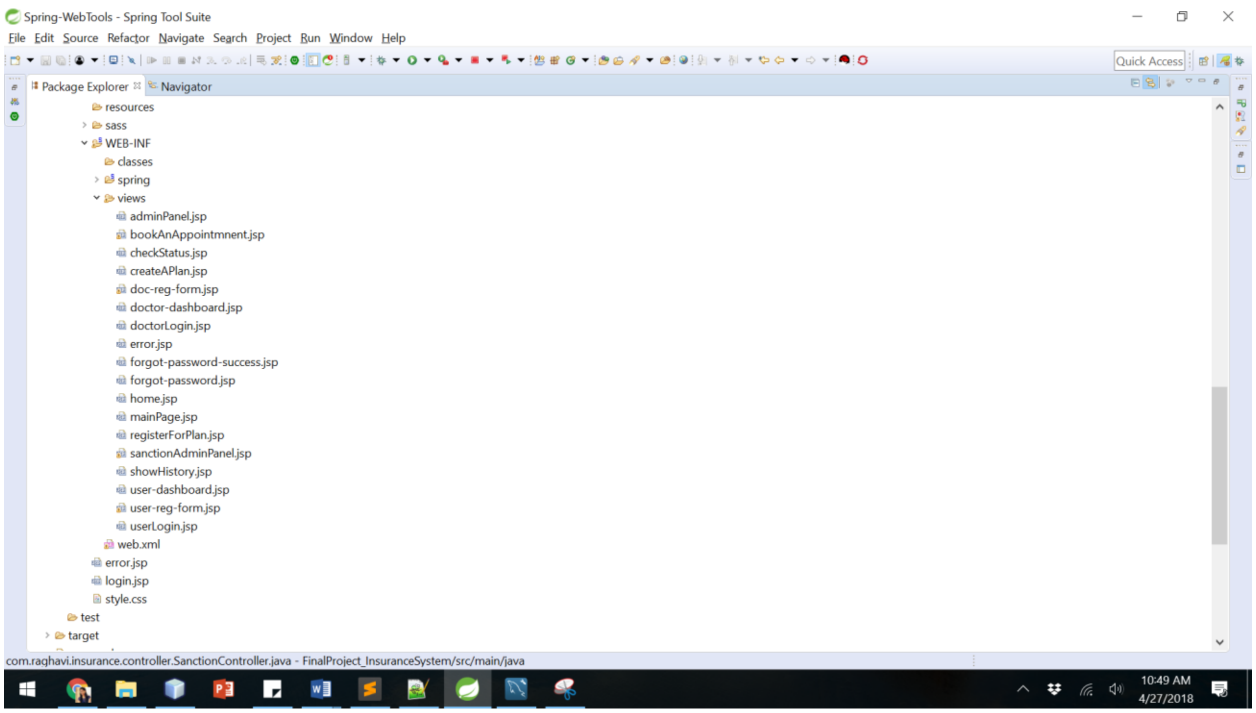

Spring-WebTools - Spring Tool Suite

File Edit Source Refactor Navigate Search Project Run Window Help

```
com.raghavi.insurance.pdf
  PdfReportView.java
  ReportController.java
com.raghavi.insurance.pojo
  CaseSheet.java
  Doctor.java
  Person.java
  Plan.java
  User.java
com.raghavi.insurance.utility
  SendEmail.java
com.raghavi.insurance.validator
  PlanCreateValidator.java
  UserValidator.java
hibernate.cfg.xml
src/main/resources
src/test/java
src/test/resources
JRE System Library [JavaSE-1.6]
Maven Dependencies
src
  main
    webapp
      dummy
      fonts
      images
      js
      resources
      sass
      WEB-INF
```

com.raghavi.insurance.controller.SanctionController.java - FinalProject_InsuranceSystem/src/main/java

10:49 AM 4/27/2018


Spring-WebTools - Spring Tool Suite

File Edit Source Refactor Navigate Search Project Run Window Help

```
      resources
      sass
      WEB-INF
        classes
        spring
        views
          adminPanel.jsp
          bookAnAppointmnent.jsp
          checkStatus.jsp
          createAPlan.jsp
          doc-reg-form.jsp
          doctor-dashboard.jsp
          doctorLogin.jsp
          error.jsp
          forgot-password-success.jsp
          forgot-password.jsp
          home.jsp
          mainPage.jsp
          registerForPlan.jsp
          sanctionAdminPanel.jsp
          showHistory.jsp
          user-dashboard.jsp
          user-reg-form.jsp
          userLogin.jsp
        web.xml
      error.jsp
      login.jsp
      style.css
  test
target
```

com.raghavi.insurance.controller.SanctionController.java - FinalProject_InsuranceSystem/src/main/java

10:49 AM 4/27/2018

APPENDIX

Controller Source Code

AdminController

```
package com.raghavi.insurance.controller;



/**
 * @author Raghavi Kirouchenaradjou
 *
 */
@Controller
@RequestMapping("/insurance/*")
public class AdminController {

	@Autowired
	@Qualifier("planCreateValidator")
	PlanCreateValidator planCreateValidator;

	@InitBinder
	private void initBinder(WebDataBinder binder) {
		binder.setValidator(planCreateValidator);
	}
	@Autowired
	DAOFactory daoFactory;

	@RequestMapping(value = "/insurance/createAPlan.htm", method = 
RequestMethod.GET)
	public ModelAndView createAPlan(HttpServletRequest request) {
		return new ModelAndView("createAPlan", "plan", new Plan());


	}

	@RequestMapping(value = "/insurance/sanctionInsurance.htm", method = 
RequestMethod.GET)
	public ModelAndView sanctionInsurance(HttpServletRequest request) {
		ModelAndView modelAndView = new ModelAndView("sanctionAdminPanel");
		List<CaseSheet> caseSheets = new ArrayList<CaseSheet>();
		try
		{
			CaseSheetDAO caseSheetDAO = daoFactory.createCaseSheetDAO();
			caseSheets = caseSheetDAO.getCaseSheets();
			Map<String, Object> map = new HashMap<String, Object>();
			map.put("type", "caseSheets");
			map.put("cases", caseSheets);
			modelAndView.addObject("map", map);
			return modelAndView;
		}
		catch(Exception ex)
		{
			System.out.println(ex.getMessage()+"Error in Admin Controller - 
Sanction");
		}
		return modelAndView;

	}

	@RequestMapping(value = "/insurance/planCreate.htm", method = 
RequestMethod.POST)
	public String handleCreateForm(HttpServletRequest request, 
@ModelAttribute("plan") Plan plan, ModelMap map,BindingResult result) {

		RegisterPlanDAO registerPlanDAO = daoFactory.createPlanDAO();
		planCreateValidator.validate(plan, result);

		if (result.hasErrors()) {
			return "error-page";
		}
		try
		{
			System.out.print("Create a Plan");

			Plan u = registerPlanDAO.register(plan);
		}
		catch (PlanCreateExceptions e) {
			System.out.println(e.getMessage());
			return "error";
		}
		return "adminPanel";


	}

}
```

DoctorActivitiesController

```java
/**
 * @author Raghavi Kirouchenaradjou
 *
 */
@Controller
@RequestMapping("/insurance/*")
public class DoctorActivitiesController {

	@Autowired
	DAOFactory daoFactory;

	@RequestMapping(value = "/insurance/getUserList.htm", method = 
RequestMethod.POST)
	@ResponseBody
	public String createAPlan(HttpServletRequest request) {
		String userID = request.getParameter("userID");
		HttpSession session = request.getSession();
		String emailID = (String) session.getAttribute("username");
		DoctorDAO doctor = daoFactory.createDoctorDAO();
		CaseSheetDAO caseSheetDAO = daoFactory.createCaseSheetDAO();
		String caseSheetDetails = request.getParameter("caseSheetID");
		try {
			Doctor doctorID = doctor.getDoctorID(emailID);
			CaseSheet caseSheet = new CaseSheet(Integer.parseInt(userID), 
doctorID.getDoctorID(), caseSheetDetails,false);
			boolean success = caseSheetDAO.createCaseSheet(caseSheet);

			if (success)
				return "success";
			else
				return "fail";
		} catch (Exception e) {
			e.printStackTrace();
		}
		return null;

	}
}
```

Doctor Controller

```
@Controller
@RequestMapping("/insurance/*")
public class DoctorController {

	@Autowired
	DAOFactory daoFactory;

	@RequestMapping(value = "/insurance/doclogin.htm", method = 
RequestMethod.POST)
	public ModelAndView handleLoginForm(HttpServletRequest request, PersonDAO 
personDAO, ModelMap map) {

		String username = request.getParameter("username");
		String password = request.getParameter("password");
		try {
			HttpSession session = request.getSession(true);
			session.setAttribute("username", username);
			Person u = personDAO.get(username, password);

			if (u != null && u.getStatus() == 1) {
				ModelAndView modelAndView = new ModelAndView("doctor-
dashboard");
				List<User> userList = new ArrayList<User>();
				try {
					Set<User> userListDetails = 
u.getDoctor().getUsers();
					userList = new ArrayList<User>(userListDetails);
				} catch (Exception e) {
					e.printStackTrace();
				}
				Map<String, Object> map1 = new HashMap<String, Object>();
				map1.put("type", "userList");
				map1.put("users", userList);
				modelAndView.addObject("map1", map1);
				return modelAndView;

			} else if (u != null && u.getStatus() == 0) {
				map.addAttribute("errorMessage", "Please activate your 
account to login!");
				return new ModelAndView("error-page");
			} else {
				map.addAttribute("errorMessage", "Invalid 
username/password!");
				return new ModelAndView("error-page");
			}
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
```

```
            return null;

        }

        @RequestMapping(value = "/insurance/docCreate.htm", method =
RequestMethod.GET)
        public String showCreateForm() {

            return "doc-reg-form";
        }

        @RequestMapping(value = "/insurance/showHistory.htm", method =
RequestMethod.GET)
        public ModelAndView showHistory(HttpServletRequest request) {
            List<CaseSheet> caseSheets = new ArrayList<CaseSheet>();

            ModelAndView modelAndView = new ModelAndView("showHistory");
            HttpSession session = request.getSession();
            String emailID = (String) session.getAttribute("username");
            try {
                DoctorDAO doctorDAO = daoFactory.createDoctorDAO();
                Doctor doc = doctorDAO.getDoctorID(emailID);
                CaseSheetDAO caseSheetDAO = daoFactory.createCaseSheetDAO();
                caseSheets =
caseSheetDAO.getCaseSheetsByDoctor(doc.getDoctorID());
                Map<String, Object> map = new HashMap<String, Object>();
                map.put("type", "caseSheets");
                map.put("cases", caseSheets);
                modelAndView.addObject("map", map);
                return modelAndView;
            } catch (Exception ex) {
                System.out.println(ex);
            }
            return modelAndView;
        }

        @RequestMapping(value = "/insurance/docCreate.htm", method =
RequestMethod.POST)
        public String handleCreateForm(HttpServletRequest request, PersonDAO
personDAO, ModelMap map) {
            Captcha captcha = Captcha.load(request, "CaptchaObject");
            String captchaCode = request.getParameter("captchaCode");
            HttpSession session = request.getSession();
            if (captcha.validate(captchaCode)) {
                String useremail = request.getParameter("username");
                String password = request.getParameter("password");
                String firstName = request.getParameter("firstName");
                String lastName = request.getParameter("lastName");
                Integer phoneNumber =
Integer.parseInt(request.getParameter("phoneNumber"));
```

```
				String specialist = request.getParameter("specialist");
				String address = request.getParameter("address");
				String medical_education =
request.getParameter("medical_education");
				Integer depositCardNo =
Integer.parseInt(request.getParameter("depositCardNo"));
				Integer routingNo =
Integer.parseInt(request.getParameter("routingNo"));
				Person person = new Person();
				Doctor doctor = new Doctor(firstName, lastName, phoneNumber,
specialist, address, medical_education,
						depositCardNo, routingNo,
request.getParameter("dob"));
				person.setUserEmail(useremail);
				person.setPassword(password);
				person.setRole("doctor");
				person.setStatus(0);
				doctor.setPerson(person);
				person.setDoctor(doctor);

				try {
					Person u = personDAO.register(person);
					Random rand = new Random();
					int randomNum1 = rand.nextInt(5000000);
					int randomNum2 = rand.nextInt(5000000);
					try {
						String str =
"http://localhost:8080/insurance/insurance/validateemail.htm?email=" + useremail
								+ "&key1=" + randomNum1 + "&key2=" +
randomNum2;
						session.setAttribute("key1", randomNum1);
						session.setAttribute("key2", randomNum2);
						sendEmail(useremail, "Click on this link to activate
your account : " + str);
					} catch (Exception e) {
						System.out.println("Email cannot be sent");
					}
				} catch (Exception e) {
					// TODO Auto-generated catch block
					e.printStackTrace();
				}
			} else {
				map.addAttribute("errorMessage", "Invalid Captcha!");
				return "doc-reg-form";
			}

			return "doctorLogin";
		}

		/**
		 * @param userEmail
	 * @param string
	 */
	public void sendEmail(String userEmail, String message) {
		try {
			Email email = new SimpleEmail();
			email.setHostName("smtp.googlemail.com");
			email.setSmtpPort(465);
			email.setAuthenticator(new
DefaultAuthenticator("contactapplication2018@gmail.com", "springmvc"));
			email.setSSLOnConnect(true);
			email.setFrom("no-reply@msis.neu.edu"); // This user email does
not

	// exist
			email.setSubject("Web tools lab ");
			email.setMsg(message); // Retrieve email from the DAO and send
this
			email.addTo(userEmail);
			email.send();
		} catch (EmailException e) {
			System.out.println("Email cannot be sent");
		}
	}

	@RequestMapping(value = "/insurance/validateemail.htm", method =
RequestMethod.GET)
	public String validateEmail(HttpServletRequest request, PersonDAO personDao,
ModelMap map) {

		// The user will be sent the following link when the use registers
		// This is the format of the email
		//
http://hostname:8080/lab10/user/validateemail.htm?email=useremail&key1=<random_number
>&key2=<body
		// of the email that when user registers>
		HttpSession session = request.getSession();
		String email = request.getParameter("email");
		int key1 = Integer.parseInt(request.getParameter("key1"));
		int key2 = Integer.parseInt(request.getParameter("key2"));
		System.out.println(session.getAttribute("key1"));
		System.out.println(session.getAttribute("key2"));

		if ((Integer) (session.getAttribute("key1")) == key1 && ((Integer)
session.getAttribute("key2")) == key2) {
			try {
				System.out.println("HI________");
				boolean updateStatus = personDao.updateUser(email);
				if (updateStatus) {
					return "doctorLogin";
				} else {
```

```
                                return "error";
                        }

                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
        } else {
                map.addAttribute("errorMessage", "Link expired , generate new 
link");
                map.addAttribute("resendLink", true);
                return "error";
        }

        return "doctorLogin";

    }

    @RequestMapping(value = "/insurance/forgotpassword.htm", method = 
RequestMethod.GET)
    public String forgotPassword() {

        return "forgot-password";
    }

    @RequestMapping(value = "/insurance/forgotpassword.htm", method = 
RequestMethod.POST)
    public String handleForgotPasswordForm(HttpServletRequest request, PersonDAO 
personDAO) {

        String useremail = request.getParameter("useremail");
        Captcha captcha = Captcha.load(request, "CaptchaObject");
        String captchaCode = request.getParameter("captchaCode");

        if (captcha.validate(captchaCode)) {
                Person user = personDAO.get(useremail);
                sendEmail(useremail, "Your password is : " + user.getPassword());
                return "forgot-password-success";
        } else {
                request.setAttribute("captchamsg", "Captcha not valid");
                return "forgot-password";
        }
    }

}
```

DoctorPagenationController

```
@Controller
@RequestMapping("/insurance/*")
public class DoctorPagenationController {

	@Autowired
	DAOFactory daoFactory;

	@RequestMapping(value = "/insurance/getDoctorList.htm", method =
RequestMethod.GET)
	public ModelAndView listOfUsers(@RequestParam(required = false) Integer page,
			@RequestParam(required = false) String
specilaist,HttpServletRequest request) {
		ModelAndView modelAndView = new ModelAndView("bookAnAppointmnent");
		List<Doctor> docorList = new ArrayList<Doctor>();

		DoctorDAO doctorDAO = daoFactory.createDoctorDAO();
		try {
			if(request.getParameter("specialistSelection")==null ||
request.getParameter("specialistSelection").equals("All"))
			{
				docorList = doctorDAO.getDoclist();
			}
			else
			{
			docorList =
doctorDAO.getDoclist(request.getParameter("specialistSelection"));
			}
		} catch (DoctorException ex) {
			System.out.println("Exception while getting doc details" + ex);
		}

		PagedListHolder<Doctor> pagedListHolder = new
PagedListHolder<Doctor>(docorList);
		pagedListHolder.setPageSize(2);
		modelAndView.addObject("maxPages", pagedListHolder.getPageCount());

		if (page == null || page < 1 || page > pagedListHolder.getPageCount())
			page = 1;

		modelAndView.addObject("page", page);
		if (page == null || page < 1 || page > pagedListHolder.getPageCount()) {
			pagedListHolder.setPage(0);
			modelAndView.addObject("users", pagedListHolder.getPageList());

		} else if (page <= pagedListHolder.getPageCount()) {
			pagedListHolder.setPage(page - 1);
```

```
				modelAndView.addObject("users", pagedListHolder.getPageList());
			}
			Map<String, Object> map = new HashMap<String, Object>();
			map.put("type", "getDoc");
			modelAndView.addObject("map", map);
			return modelAndView;
		}

		@RequestMapping(value = "/insurance/getDoctorList.htm", method =
RequestMethod.POST)
		public ModelAndView bookAnAppointment(HttpServletRequest request) {
			ModelAndView modelAndView = new ModelAndView("bookAnAppointmnent");
			HttpSession session = request.getSession(false);
			String emailID = (String) session.getAttribute("username");
			int doctorID = Integer.parseInt(request.getParameter("docwho"));

			try {
				DoctorDAO doctorDAO = daoFactory.createDoctorDAO();
				doctorDAO.bookAnAppointment(doctorID,emailID);

			} catch (Exception e) {

			}
			Map<String, Object> map = new HashMap<String, Object>();
			map.put("type", "bookNow");
			modelAndView.addObject("map", map);
			return modelAndView;
		}
}
```

SanctionController

```
@Controller
@RequestMapping("/insurance/*")
public class SanctionController {

	@Autowired
	DAOFactory daoFactory;

	@RequestMapping(value = "/insurance/getMoreDetails.htm", method =
RequestMethod.POST)
	@ResponseBody
	public String getPlanDetails(HttpServletRequest request) {
		String result = "";
		UserDAO userDAO = daoFactory.createUserDAO();
		RegisterPlanDAO registerPlanDAO = daoFactory.createPlanDAO();
		int userID = Integer.parseInt(request.getParameter("userID"));
		try {
			User user = userDAO.getUserByUserID(userID);
			Plan planDetails =
registerPlanDAO.getPlanDetails(user.getPlan_enrolled());
			result = "<br><strong>User First Name
:</strong>"+user.getFirstName() + "<br><strong>Last Name </stroong>" +
user.getLastName() + "<br><strong>Address</strong>" + user.getAddress() + "<br>DOB"
					+ user.getDob() + "<br><strong>Credit Card</strong>"
+ user.getCreditCardNum() + "<br><strong>Hospital Coverage</strong>" +
planDetails.getHospital_coverage()
					+ "<br><strong>Monthly Payment</strong>" +
planDetails.getMonthly_payment() + "<br><strong>Out Of Pocket</strong>" +
planDetails.getOut_of_pocket() + "<br>"
					+ planDetails.getPlanName();

			return result;
		} catch (PlanCreateExceptions e) {
			System.out.println(e.getMessage());
		}
		return null;
	}

	@RequestMapping(value = "/insurance/sanctionInsuranceNow.htm", method =
RequestMethod.POST)
	@ResponseBody
	public String santionInsurance(HttpServletRequest request) {
		UserDAO userDAO = daoFactory.createUserDAO();
		CaseSheetDAO caseSheetDAO = daoFactory.createCaseSheetDAO();
		int userID = Integer.parseInt(request.getParameter("userID"));
		try {
			User user = userDAO.getUserByUserID(userID);
	user.setAmount_sanctioned(Integer.parseInt(request.getParameter("amount_sancti
oned")));
			boolean yesApproved = userDAO.update(user);
			boolean valuUpdated = 
caseSheetDAO.update(Integer.parseInt(request.getParameter("caseSheetID")));
			String message ="Dear 
".concat(user.getFirstName()).concat("/n")+"Hurray! You have been sanctioned the 
amount";
			SendEmail.sendEmail(user.getPerson().getUserEmail(),message );
			if (yesApproved && valuUpdated)
				return "success";
			else
				return "failedToSanction";
		} catch (Exception e) {
			System.out.println(e.getMessage());
		}
		return null;
	}
}
```

UserActivityController

```
@Controller
@RequestMapping("/insurance/*")
public class UserActivityController {

	@Autowired
	DAOFactory daoFactory;

	@RequestMapping(value = "/insurance/selection.htm", method = 
RequestMethod.POST)
	public ModelAndView activityDropDown(HttpServletRequest request) {

		String action = request.getParameter("activityDropDown");

		if (action.equals("bookAnAppointmnent")) {
			return new ModelAndView("bookAnAppointmnent");
		} else if (action.equals("registerForPlan")) {

			return new ModelAndView("registerForPlan");
		}
		else if (action.equals("checkStatus")) {
			HttpSession session = request.getSession(false);
			String emailID = (String) session.getAttribute("username");

			ModelAndView modelAndView = new ModelAndView("checkStatus");
			try
			{
				UserDAO userDAO = new UserDAO();
				User userByEmail = userDAO.getUserByEmailID(emailID);
				Map<String, Object> map = new HashMap<String, Object>();
				map.put("type", "users");
				map.put("user", userByEmail);
				modelAndView.addObject("map", map);
				return modelAndView;
			}
			catch(Exception ex)
			{
				System.out.println(ex.getMessage()+"Error in Check Ststus 
Controller - Sanction");
			}
			return modelAndView;
		}

		return new ModelAndView();
	}
	//Ajax to retrieve the plan Details
	@RequestMapping(value = "/insurance/getPlanDetails.htm", method = 
RequestMethod.POST)
	@ResponseBody
	public String getPlanDetails(HttpServletRequest request) {
		String result="";
		RegisterPlanDAO registerPlanDAO = daoFactory.createPlanDAO();
		int planID = Integer.parseInt(request.getParameter("planID"));
		try {
			Plan planDetails = registerPlanDAO.getPlanDetails(planID);
			result = "<div class='planStyle'> Plan Name
:"+planDetails.getPlanName() + "<br>" +
					"<div class='planStyle'> Monthly Payment :"+
planDetails.getMonthly_payment() + "<br>" +
					"<div class='planStyle'> Out Of Pocket :"+
planDetails.getOut_of_pocket() + "<br>" +
					"<div class='planStyle'> Hospital Coverage :"+
planDetails.getHospital_coverage() + "<br>"+
					"<div class='planStyle'> Office Visit? :"+
planDetails.isOffice_visit() + "<br>" +
					"<div class='planStyle'> Rx Covered? :"+
planDetails.isRx_covered() + "<br>";
			return result;
		} catch (PlanCreateExceptions e) {
			System.out.println(e.getMessage());
		}
		return null;
	}

	@RequestMapping(value = "/insurance/confirmPlan.htm", method =
RequestMethod.POST)
	public String confirmThePlan(HttpServletRequest request) {
		HttpSession session = request.getSession(false);
		String emailID = (String) session.getAttribute("username");
		int planSelectionID =
Integer.parseInt(request.getParameter("planSelection"));
		RegisterPlanDAO updatePlan = daoFactory.createPlanDAO();
		try {
			boolean success = updatePlan.updatePlanEnrolled(emailID,
planSelectionID);
			if(success)
			{
				return "user-dashboard";
			}
		} catch (Exception e) {
			System.out.println(e.getMessage());
		}

		return "error-page";

	}
	@RequestMapping(value = "/insurance/confirmPlan.htm", method =
RequestMethod.GET)
	public String getToUserDashboard(HttpServletRequest request) {
		return "user-dashboard";
	}
```

```
}
```

UserController

```
@Controller
@RequestMapping("/insurance/*")
public class UserController {

	/*@Autowired
	@Qualifier("userValidator")
	UserValidator validator;

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

		@RequestMapping(value = "/insurance/userlogin.htm", method =
RequestMethod.POST)
		public String handleLoginForm(HttpServletRequest request, PersonDAO
personDAO, ModelMap map) {

				String username = request.getParameter("username");
				String password = request.getParameter("password");
				try {
					HttpSession session = request.getSession();
					session.setAttribute("username", username);
					Person u = personDAO.get(username, password);

					if (u != null && u.getStatus() == 1) {
						return "user-dashboard";
					} else if (u != null && u.getStatus() == 0) {
						map.addAttribute("errorMessage", "Please activate
your account to login!");
						return "error";
					} else {
						map.addAttribute("errorMessage", "Invalid
username/password!");
						return "error";
					}
				} catch (Exception e) {
					// TODO Auto-generated catch block
					e.printStackTrace();
				}

				return null;

		}
		@RequestMapping(value = "/insurance/userCreate.htm", method =
RequestMethod.GET)
		public ModelAndView showCreateForm() {
			return new ModelAndView("user-reg-form", "user", new User());

		}

		@RequestMapping(value = "/insurance/userCreate.htm", method =
RequestMethod.POST)
		public String handleCreateForm(HttpServletRequest request,
@ModelAttribute("user") User user,PersonDAO personDAO, ModelMap map,BindingResult
result) {

			/*validator.validate(user, result);

			if (result.hasErrors()) {
				return "";
			}*/

			Captcha captcha = Captcha.load(request, "CaptchaObject");
			String captchaCode = request.getParameter("captchaCode");
			HttpSession session = request.getSession();
			if (captcha.validate(captchaCode)) {
				System.out.println(user.getFirstName());
				String firstName = request.getParameter("firstName");
				String lastName = request.getParameter("lastName");
				// phoneNumber =
Integer.parseInt(request.getParameter("phoneNumber"));
				String address =  request.getParameter("address");
				//Integer creditCardNum =
Integer.parseInt(request.getParameter("creditCardNum"));
				String company_name =
request.getParameter("company_name");

				Person person = new Person();
				User userAccount = new User(firstName,lastName,
user.getPhoneNumber(), address, user.getCreditCardNum(), request.getParameter("dob"),
company_name,0,0);
				person.setUserEmail(user.getPerson().getUserEmail());
				person.setPassword(user.getPerson().getPassword());
				person.setRole("user");
				person.setStatus(0);

				userAccount.setPerson(person);
				person.setUser(userAccount);

				try {
					Person u = personDAO.register(person);
					Random rand = new Random();
					int randomNum1 = rand.nextInt(5000000);
					int randomNum2 = rand.nextInt(5000000);
```

```
                                    try {
                                            String str = 
"http://localhost:8080/insurance/insurance/validateuseremail.htm?email=" + 
user.getPerson().getUserEmail() + "&key1="
                                                        + randomNum1 + "&key2=" + 
randomNum2;
                                            session.setAttribute("key1", randomNum1);
                                            session.setAttribute("key2", randomNum2);
                                            String message ="Dear 
".concat(userAccount.getFirstName()).concat("/n")+"Click on this link to activate 
your account : " + str;

        sendEmail(user.getPerson().getUserEmail(),message );
                                    } catch (Exception e) {
                                            System.out.println("Email cannot be sent");
                                    }
                            } catch (Exception e) {
                                    // TODO Auto-generated catch block
                                    e.printStackTrace();
                            }
                    } else {
                            map.addAttribute("errorMessage", "Invalid Captcha!");
                            return "user-reg-form";
                    }

                    return "userLogin";
            }

            /**
             * @param userEmail
             * @param string
             */
            public void sendEmail(String userEmail, String message) {
                    try {
                            Email email = new SimpleEmail();
                            email.setHostName("smtp.googlemail.com");
                            email.setSmtpPort(465);
                            email.setAuthenticator(new 
DefaultAuthenticator("contactapplication2018@gmail.com", "springmvc"));
                            email.setSSLOnConnect(true);
                            email.setFrom("no-reply@msis.neu.edu"); // This user email 
does not

        // exist
                            email.setSubject("Insurance Company ");
                            email.setMsg(message); // Retrieve email from the DAO and 
send this
                            email.addTo(userEmail);
                            email.send();
                    } catch (EmailException e) {
                            System.out.println("Email cannot be sent");
```

```
                        }
                }

                @RequestMapping(value = "/insurance/validateuseremail.htm", method =
RequestMethod.GET)
                public String validateEmail(HttpServletRequest request, PersonDAO
personDAO, ModelMap map) {

                        // The user will be sent the following link when the use
registers
                        // This is the format of the email
                        //
http://hostname:8080/lab10/user/validateemail.htm?email=useremail&key1=<random_number
>&key2=<body
                        // of the email that when user registers>
                        HttpSession session = request.getSession();
                        String email = request.getParameter("email");
                        int key1 = Integer.parseInt(request.getParameter("key1"));
                        int key2 = Integer.parseInt(request.getParameter("key2"));
                        System.out.println(session.getAttribute("key1"));
                        System.out.println(session.getAttribute("key2"));

                        if ((Integer) (session.getAttribute("key1")) == key1 &&
((Integer) session.getAttribute("key2")) == key2) {
                                try {
                                        System.out.println("HI________");
                                        boolean updateStatus = personDAO.updateUser(email);
                                        if (updateStatus) {
                                                return "userLogin";
                                        } else {

                                                return "error";
                                        }

                                } catch (Exception e) {
                                        // TODO Auto-generated catch block
                                        e.printStackTrace();
                                }
                        } else {
                                map.addAttribute("errorMessage", "Link expired , generate
new link");

                                map.addAttribute("resendLink", true);
                                return "error";
                        }

                        return "userLogin";

                }

        }
```

LoginController

```
@Controller
public class LoginController {

	@RequestMapping(value = "/insurance/login.htm", method = RequestMethod.GET)
	public String showLoginForm() {

		return "mainPage";
	}

	@RequestMapping(value = "/insurance/admin.htm", method = RequestMethod.GET)
	public String showAdminPanel() {

		return "adminPanel";
	}

	@RequestMapping(value = "/insurance/userMainPage.htm", method =
RequestMethod.GET)
	public String showUserPanel() {

		return "userLogin";
	}

	@RequestMapping(value = "/insurance/doctorMainPage.htm", method =
RequestMethod.GET)
	public String showDoctorPanel() {

		return "doctorLogin";
	}

	@RequestMapping(value = "/insurance/logOff.htm", method = RequestMethod.GET)
	public String showLoginFormAfterLogOff(HttpServletRequest request) {
		HttpSession session = request.getSession(false);
		session.invalidate();
		return "mainPage";
	}


}
```

POJO CLASSES

CASESHEET

```
@Entity
@Table(name="casesheet")
public class CaseSheet {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name = "casesheetID")
	private int casesheetID;

	@Column(name="userID")
	private int userID;

	@Column(name="doctorID")
	private long doctorID;

	@Column(name="casesheet_details")
	private String casesheet_details;

	@Column(name="approved")
	private boolean approved;

	public int getCasesheetID() {
		return casesheetID;
	}

	public int getUserID() {
		return userID;
	}

	public void setUserID(int userID) {
		this.userID = userID;
	}

	public long getDoctorID() {
		return doctorID;
	}

	public void setDoctorID(int doctorID) {
		this.doctorID = doctorID;
	}

	public String getCasesheet_details() {
		return casesheet_details;
	}

	public void setCasesheet_details(String casesheet_details) {
		this.casesheet_details = casesheet_details;
	}



	public void setDoctorID(long doctorID) {
		this.doctorID = doctorID;
	}



	public boolean isApproved() {
		return approved;
	}

	public void setApproved(boolean approved) {
		this.approved = approved;
	}

	/**
	 * @param userID
	 * @param doctorID
	 * @param casesheet_details
	 */
	public CaseSheet(int userID, long doctorID, String casesheet_details,boolean
approved) {
		this.userID = userID;
		this.doctorID = doctorID;
		this.casesheet_details = casesheet_details;
		this.approved = approved;
	}

	public CaseSheet()
	{

	}
}
```

DOCTOR

```
/**
 * @author Raghavi Kirouchenaradjou
 *
 */
@Entity
@Table(name = "doctor_table")
public class Doctor {


	//@GenericGenerator(name = "generator", strategy = "foreign", parameters = 
@Parameter(name = "property", value = "person"))
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name = "doctorID")//, unique = true, nullable = false)
	private long doctorID;

	@Column(name = "firstName")
	private String firstName;

	@Column(name = "lastName")
	private String lastName;

	@Column(name = "phoneNumber")
	private int phoneNumber;

	@Column(name = "specialist")
	private String specialist;

	@Column(name = "address")
	private String address;

	@Column(name = "medical_education")
	private String medical_education;

	@Column(name = "depositCardNo")
	private int depositCardNo;

	@Column(name = "routingNo")
	private int routingNo;

	@Column(name = "dob")
	private String dob;



	@OneToOne
	@JoinColumn (name = "personID")
	private Person person;
	@ManyToMany(mappedBy="doctors")
	private Set<User> users = new HashSet<User>();


	public Set<User> getUsers() {
		return users;
	}

	public void setUsers(Set<User> users) {
		this.users = users;
	}

	public Person getPerson() {
		return person;
	}

	public void setPerson(Person person) {
		this.person = person;
	}



	/**
	 * @param firstName
	 * @param lastName
	 * @param phoneNumber
	 * @param specialist
	 * @param address
	 * @param medical_education
	 * @param depositCardNo
	 * @param routingNo
	 */
	public Doctor(String firstName, String lastName, int phoneNumber, String
specialist, String address,
			String medical_education, int depositCardNo, int routingNo,String
dob) {
		super();
		this.firstName = firstName;
		this.lastName = lastName;
		this.phoneNumber = phoneNumber;
		this.specialist = specialist;
		this.address = address;
		this.medical_education = medical_education;
		this.depositCardNo = depositCardNo;
		this.routingNo = routingNo;
		this.dob = dob;
	}


	public String getDob() {
		return dob;
	}

	public void setDob(String dob) {
		this.dob = dob;
	}

	public int getPhoneNumber() {
		return phoneNumber;
	}

	public void setPhoneNumber(int phoneNumber) {
		this.phoneNumber = phoneNumber;
	}

	public String getSpecialist() {
		return specialist;
	}

	public void setSpecialist(String specialist) {
		this.specialist = specialist;
	}

	public String getAddress() {
		return address;
	}

	public void setAddress(String address) {
		this.address = address;
	}

	public String getMedical_education() {
		return medical_education;
	}

	public void setMedical_education(String medical_education) {
		this.medical_education = medical_education;
	}

	public int getDepositCardNo() {
		return depositCardNo;
	}

	public void setDepositCardNo(int depositCardNo) {
		this.depositCardNo = depositCardNo;
	}

	public int getRoutingNo() {
		return routingNo;
	}

	public void setRoutingNo(int routingNo) {
		this.routingNo = routingNo;
	}

	public Doctor() {

	}

	public String getFirstName() {
		return firstName;
	}

	public void setFirstName(String firstName) {
		this.firstName = firstName;
	}

	public String getLastName() {
		return lastName;
	}

	public void setLastName(String lastName) {
		this.lastName = lastName;
	}


	public long getDoctorID() {
		return doctorID;
	}


	@Override
	public String toString(){
		return firstName.concat(" ").concat(lastName);
	}

}
```

PERSON

```
/**
 * @author Raghavi Kirouchenaradjou
 *
 */
@Entity
@Table(name = "person_table")
public class Person {

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name = "personID", unique = true, nullable = false)
	private long personID;

	@Column(name = "userEmail")
	private String userEmail;

	@Column(name = "password")
	private String password;

	@Column(name = "status")
	private int status;

	@Column(name = "role")
	private String role;

	@OneToOne(fetch = FetchType.LAZY, mappedBy = "person", cascade =
CascadeType.ALL)
	@PrimaryKeyJoinColumn
	private Doctor doctor;

	@OneToOne(fetch = FetchType.LAZY, mappedBy = "person", cascade =
CascadeType.ALL)
	@PrimaryKeyJoinColumn
	private User user;

	public Person() {

	}


	public Doctor getDoctor() {
		return doctor;
	}


	public void setDoctor(Doctor doctor) {
		this.doctor = doctor;
	}
	public User getUser() {
		return user;
	}


	public void setUser(User user) {
		this.user = user;
	}


	public String getRole() {
		return role;
	}


	public void setRole(String string) {
		this.role = string;
	}


	public String getUserEmail() {
		return userEmail;
	}

	public void setUserEmail(String userEmail) {
		this.userEmail = userEmail;
	}

	public String getPassword() {
		return password;
	}

	public void setPassword(String password) {
		this.password = password;
	}

	public int getStatus() {
		return status;
	}

	public void setStatus(int status) {
		this.status = status;
	}

}
```

PLAN

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

	@Id
	@GeneratedValue(strategy = GenerationType.SEQUENCE)
	@Column(name = "planID")
	private int planID;

	@Column(name="planName")
	private String planName;

	@Column(name="monthly_payment")
	private int monthly_payment;

	@Column(name="hospital_coverage")
	private String hospital_coverage;

	@Column(name="office_visit")
	private boolean office_visit;

	@Column(name="out_of_pocket")
	private int out_of_pocket;

	@Column(name="rx_covered")
	private boolean rx_covered;

	public String getPlanName() {
		return planName;
	}

	public void setPlanName(String planName) {
		this.planName = planName;
	}

	public int getMonthly_payment() {
		return monthly_payment;
	}

	public void setMonthly_payment(int monthly_payment) {
		this.monthly_payment = monthly_payment;
	}

	public String getHospital_coverage() {
		return hospital_coverage;
	}

	public void setHospital_coverage(String hospital_coverage) {
		this.hospital_coverage = hospital_coverage;
	}

	public boolean isOffice_visit() {
		return office_visit;
	}

	public void setOffice_visit(boolean office_visit) {
		this.office_visit = office_visit;
	}

	public int getOut_of_pocket() {
		return out_of_pocket;
	}

	public void setOut_of_pocket(int out_of_pocket) {
		this.out_of_pocket = out_of_pocket;
	}

	public boolean isRx_covered() {
		return rx_covered;
	}

	public void setRx_covered(boolean rx_covered) {
		this.rx_covered = rx_covered;
	}


	/**
	 * @param planName
	 * @param monthly_payment
	 * @param hospital_coverage
	 * @param office_visit
	 * @param out_of_pocket
	 * @param rx_covered
	 */
	public Plan(String planName, int monthly_payment, String hospital_coverage,
boolean office_visit, int out_of_pocket,
			boolean rx_covered) {
		this.planName = planName;
		this.monthly_payment = monthly_payment;
		this.hospital_coverage = hospital_coverage;
		this.office_visit = office_visit;
		this.out_of_pocket = out_of_pocket;
		this.rx_covered = rx_covered;
	}


	public Plan() {
	}
```

```
}
```

USER

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

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name = "userID")
	private long userID;

	@Column(name = "firstName")
	private String firstName;

	@Column(name = "lastName")
	private String lastName;

	@Column(name = "phoneNumber")
	private int phoneNumber;


	@Column(name = "address")
	private String address;

	@Column(name = "creditCardNum")
	private int creditCardNum;

	@Column(name = "dob")
	private String dob;

	@Column(name="company_name")
	private String company_name;

	@Column(name="plan_enrolled")
	private int plan_enrolled;

	@Column(name="amount_sanctioned")
	private int amount_sanctioned;

	@OneToOne
	@JoinColumn (name = "personID")
	private Person person;

	@ManyToMany
    @JoinTable (
        name="user_doctor_table",
        joinColumns = {@JoinColumn(name="userID", nullable = false, updatable =
false)},
         inverseJoinColumns = {@JoinColumn(name="doctorID" )}
    )
      private Set<Doctor> doctors = new HashSet<Doctor>();


      public Set<Doctor> getDoctors() {
             return doctors;
      }

      public void setDoctors(Set<Doctor> doctors) {
             this.doctors = doctors;
      }

      public String getFirstName() {
             return firstName;
      }

      public void setFirstName(String firstName) {
             this.firstName = firstName;
      }

      public String getLastName() {
             return lastName;
      }

      public void setLastName(String lastName) {
             this.lastName = lastName;
      }

      public Person getPerson() {
             return person;
      }

      public void setPerson(Person person) {
             this.person = person;
      }



      public int getPhoneNumber() {
             return phoneNumber;
      }

      public void setPhoneNumber(int phoneNumber) {
             this.phoneNumber = phoneNumber;
      }

      public String getAddress() {
             return address;
      }
```

```
	public void setAddress(String address) {
		this.address = address;
	}

	public int getCreditCardNum() {
		return creditCardNum;
	}

	public void setCreditCardNum(int creditCardNum) {
		this.creditCardNum = creditCardNum;
	}

	public String getDob() {
		return dob;
	}

	public void setDob(String dob) {
		this.dob = dob;
	}

	public String getCompany_name() {
		return company_name;
	}

	public void setCompany_name(String company_name) {
		this.company_name = company_name;
	}

	public int getPlan_enrolled() {
		return plan_enrolled;
	}

	public void setPlan_enrolled(int plan_enrolled) {
		this.plan_enrolled = plan_enrolled;
	}


	public long getUserID() {
		return userID;
	}



	public int getAmount_sanctioned() {
		return amount_sanctioned;
	}

	public void setAmount_sanctioned(int amount_sanctioned) {
		this.amount_sanctioned = amount_sanctioned;
	}
	/**
	 * @param firstName
	 * @param lastName
	 * @param phoneNumber
	 * @param address
	 * @param creditCardNum
	 * @param dob
	 * @param company_name
	 */
	public User(String firstName, String lastName, int phoneNumber, String
address, int creditCardNum, String dob,
			String company_name,int plan_enrolled,int amount_sanctioned) {
		this.firstName = firstName;
		this.lastName = lastName;
		this.phoneNumber = phoneNumber;
		this.address = address;
		this.creditCardNum = creditCardNum;
		this.dob = dob;
		this.company_name = company_name;
		this.plan_enrolled = plan_enrolled;
		this.amount_sanctioned = amount_sanctioned;
	}

	public User()
	{

	}

	@Override
	public String toString(){
		return firstName.concat(" ").concat(lastName);
	}

}
```