# Project Report

Manasa Vanga

NUID: 001383682

## Application for Medical Online Services

### Introduction:

The application facilitates users in scheduling appointments with doctors available in different hospitals and specializations.

Application mainly gives access to three kinds of users.

1. Admin
2. Doctor
3. Patient

### Functionality:

Patient can register with their username, password and other required details. They have the access to schedule an appointment from the list of schedules available and can delete the booked slot if there is any change in the plan. At the same time doctor schedules also gets updated and shown as available to other patients.

Doctor can register with their username, password and specialization details. They have access to schedule their available dates and times. They can add multiple dates and times according to their availability. These time slots are visible to the Patient. Doctor home page can view all the appointment slots scheduled by patients.

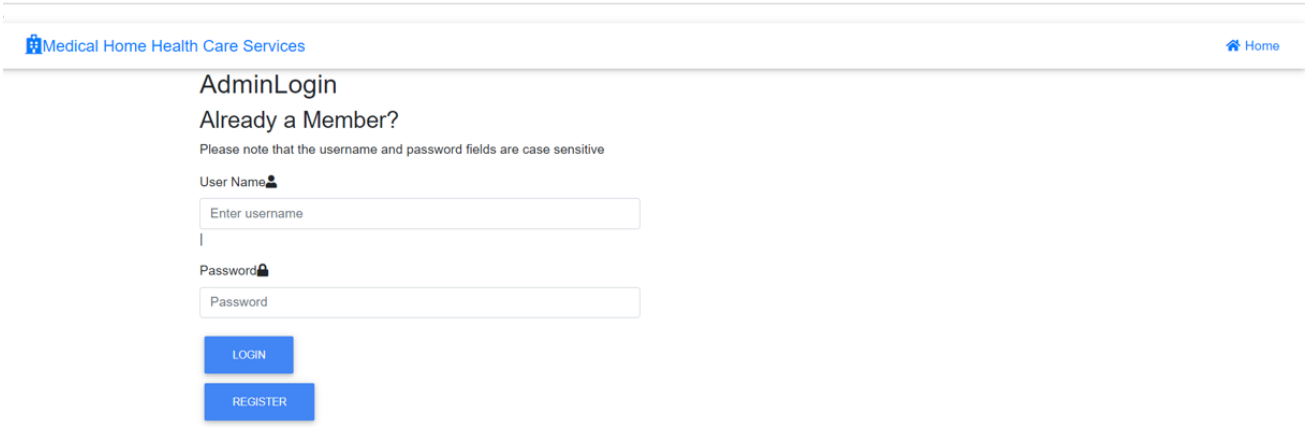
Admin can register, login and has access to view all the Patients, Doctors and Appointment information in his home page. Admin has contact information of doctors and patients and manages the appointments.

### Technologies:

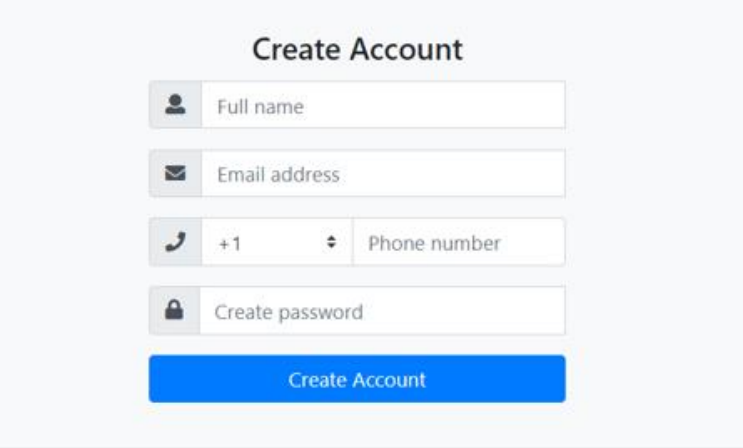
- Frameworks: Spring MVC, Spring Security, Bootstrap, Hibernate
- Server: Tomcat
- Client Side: HTML5, CSS, JavaScript, Ajax
- Database: MySQL
- JavaScript Frameworks: jQuery, Wow, Front-awesome
- IDE: Spring Tool Suite 3

Home page

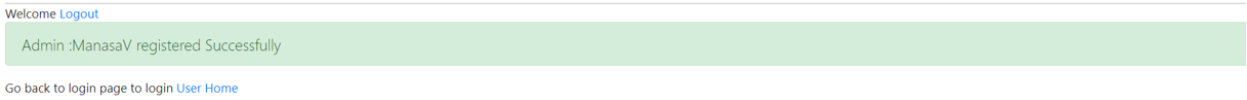

Admin Login or Register Page



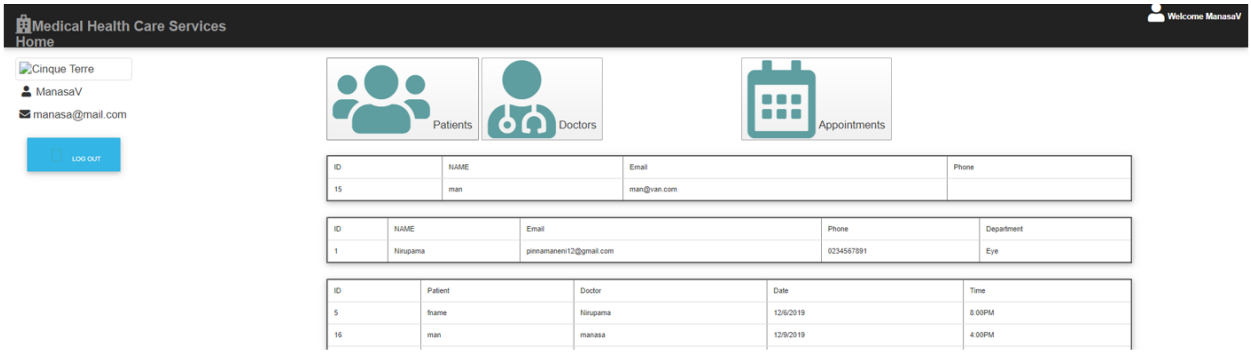

Admin Registration Form



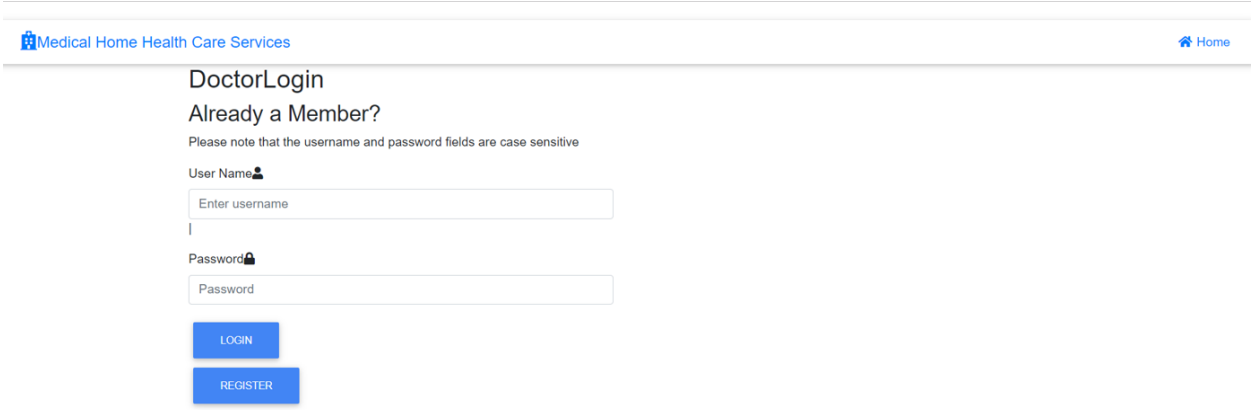
Success Page after Admin Registration



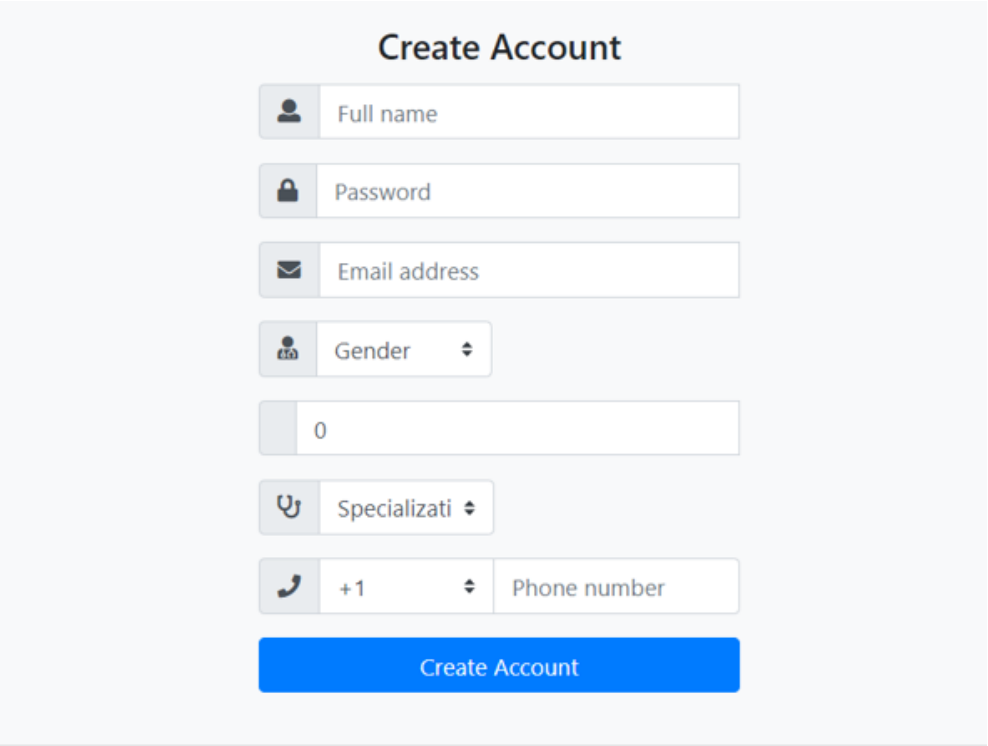

Admin Home page –Patients, Doctors and Appointment data can be viewed.




Doctor Login Form – Register if new user

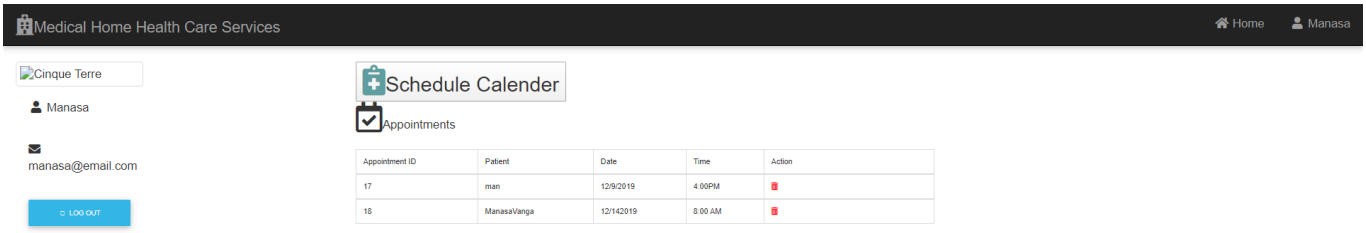


Doctor Registration Form

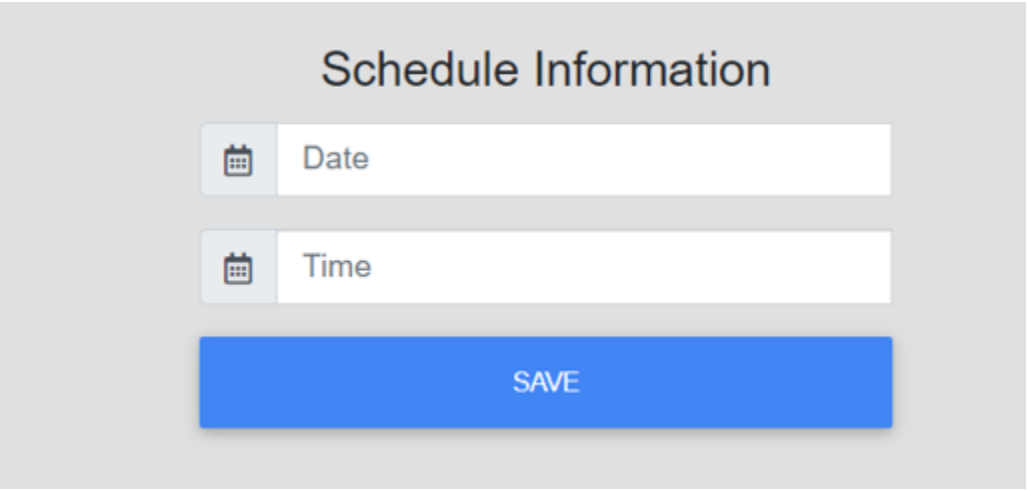



Successful Registration of Doctor – can navigate to home page or logout

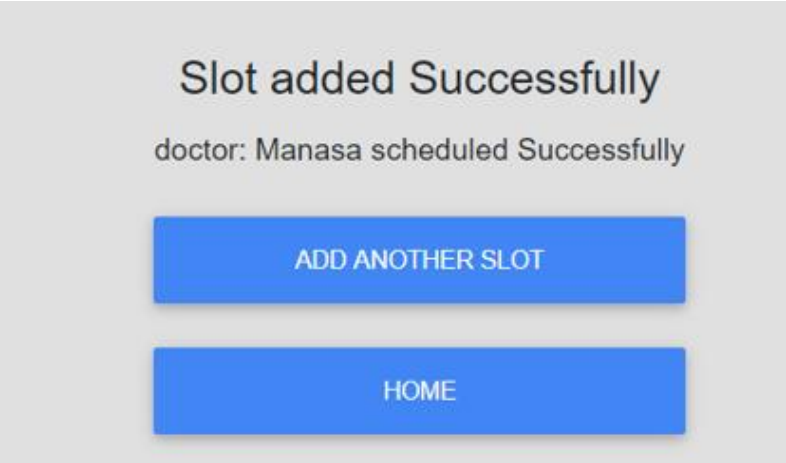

Doctor Home – can view booked appointments or delete if not available in schedules slots.



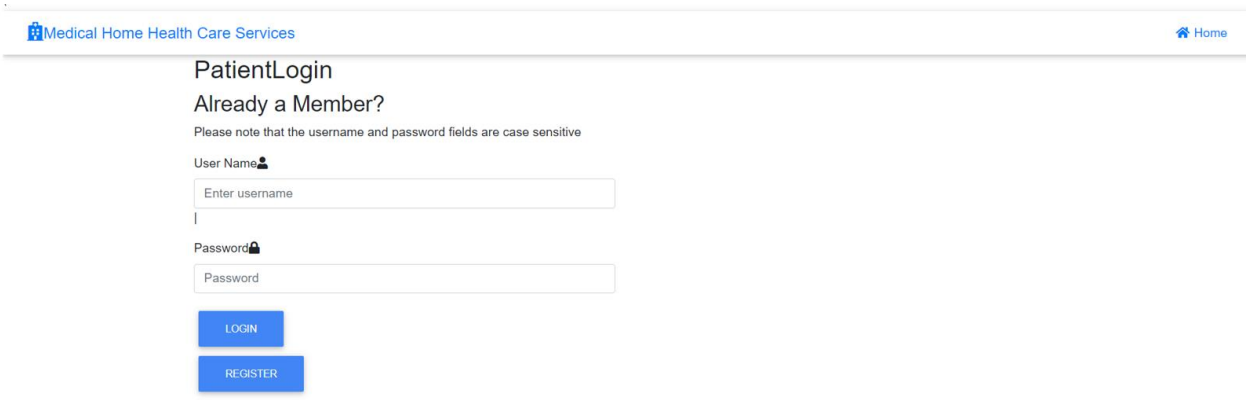
Doctor adding available date and time to available slots.



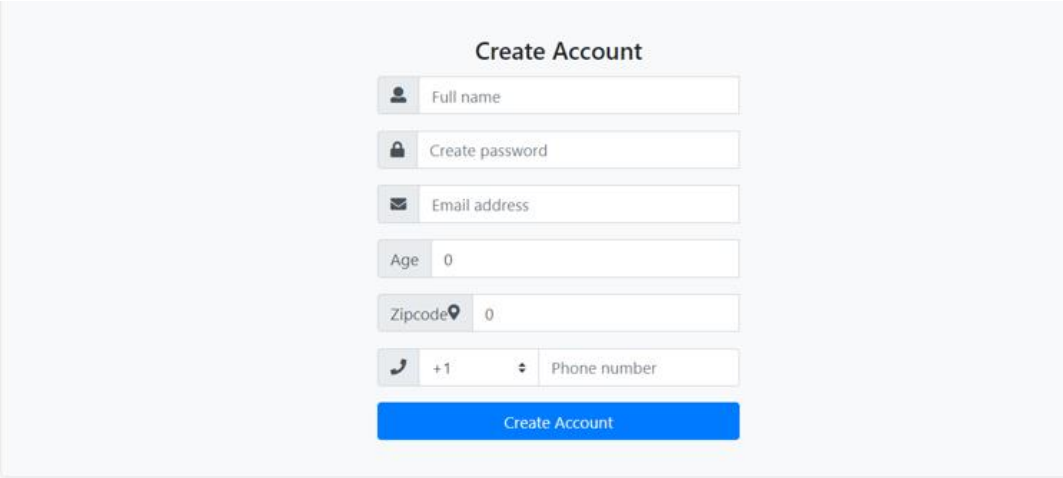

Slot added success page – can add another slot or go to home page




Patient Login Form ( register if new)



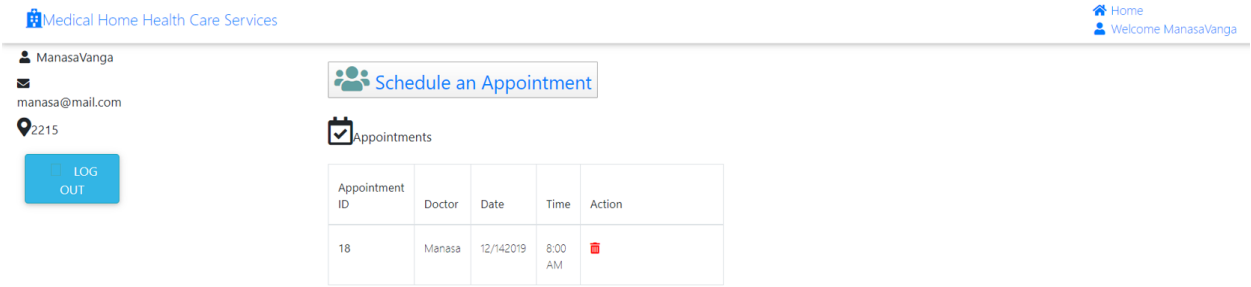
Patient Registration Form



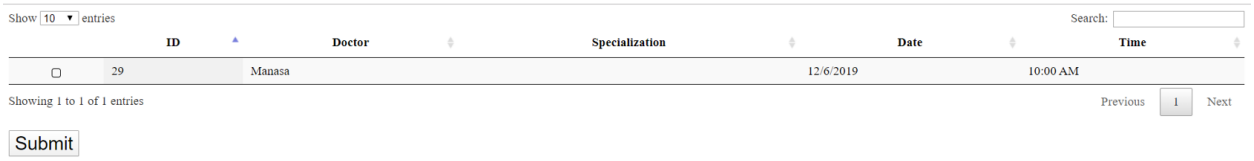

Registration Success page – navigate to home page or logout


Patient Home

- Schedule appointments
- View scheduled appointments



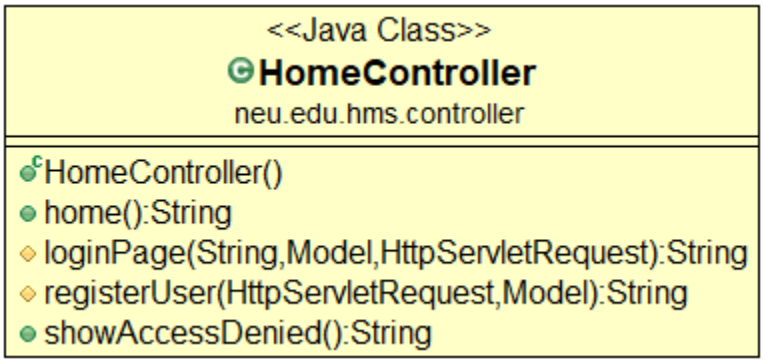
Select date, time and doctor from available slots.




## Controllers

### HomeController:

## Class Diagram:



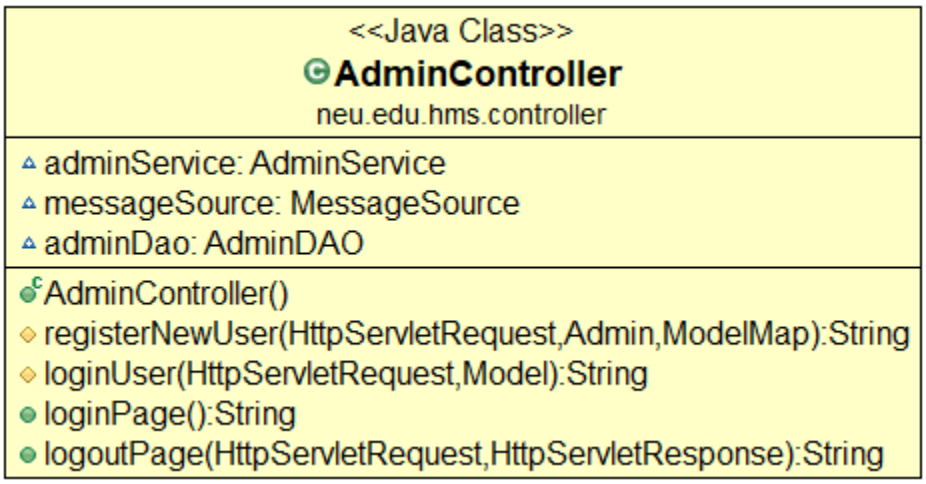

```
package neu.edu.hms.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.springframework.security.authentication.AuthenticationTrustResolver;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import neu.edu.hms.entity.Admin;
import neu.edu.hms.entity.Doctor;
import neu.edu.hms.entity.Patient;


@Controller
```

```
public class HomeController {
	AuthenticationTrustResolver authenticationTrustResolver;

	/**
	 * Method returns the home page of the application
	 */
	@RequestMapping(value = "/", method = RequestMethod.GET)
	public String home() throws Exception {
		return "home";
	}

	/**
	 * This method returns login page depending on role selected in home page
	 */
	@RequestMapping(value = "/login", method = RequestMethod.GET)
	protected String loginPage(@RequestParam("role") String role, Model
model,HttpServletRequest request) throws Exception {

		HttpSession session = request.getSession();
		if(session!=null) {
			session.invalidate();
		}

		session = request.getSession();
		session.setAttribute("role", role);


		return "login";

	}

	/**
	 * This method returns registration of user based on role added in session
	 */
	@RequestMapping(value = "/user/register", method = RequestMethod.GET)
	protected String registerUser(HttpServletRequest request, Model model) throws
Exception {
		HttpSession session = request.getSession();

		String role= (String) session.getAttribute("role");
		String form = null;
		if(role.equalsIgnoreCase("Admin")) {
			form = "adminregistration";
			model.addAttribute("admin", new Admin());
		}else if(role.equalsIgnoreCase("Doctor")) {
			form = "doctorRegistration";
```

```
                model.addAttribute("doctor", new Doctor());
            }else if(role.equalsIgnoreCase("Patient")) {
                form = "patientRegistrationForm";
                model.addAttribute("patient", new Patient());
            }
            return form;

        }

        @RequestMapping(value = "/AccessDenied",method= RequestMethod.GET)
        public String showAccessDenied() {
            return "accessDenied";

        }


}
```

## AdminController:

## Class Diagram:




```
package neu.edu.hms.controller;


import java.util.Locale;

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.context.MessageSource;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.ui.ModelMap;
import org.springframework.validation.FieldError;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import neu.edu.hms.DAO.AdminDAO;
import neu.edu.hms.entity.Admin;
import neu.edu.hms.service.AdminService;

@Controller
public class AdminController {

	@Autowired
	AdminService adminService;

	@Autowired
	MessageSource messageSource;

	@Autowired
	AdminDAO adminDao;

	/**
	 * This method registers new Admin
	 * @return registration success page
```

```
	*/
	@RequestMapping(value = "/admin/register", method = RequestMethod.POST)
	protected String registerNewUser(HttpServletRequest
request,@ModelAttribute("admin") Admin admin,ModelMap model) throws Exception {

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

		if(admin.getAdminName() == null) {
			FieldError registerError = new
FieldError("admin","adminName",messageSource.getMessage("User/Password is not
Valid",new String[] {admin.getAdminName()}, Locale.getDefault()));
			return "login";
		}

		try {

			Admin ad = adminDao.get(admin.getAdminName());
			if(ad == null) {

				adminService.register(admin);
				request.getSession().setAttribute("admin", admin);
				model.addAttribute("success", "Admin :" +
admin.getAdminName()+" registered Successfully");
			}
			else {
				session.setAttribute("errorMessage", "UserName already used");

				return "error";
			}
		} catch (Exception e) {
```

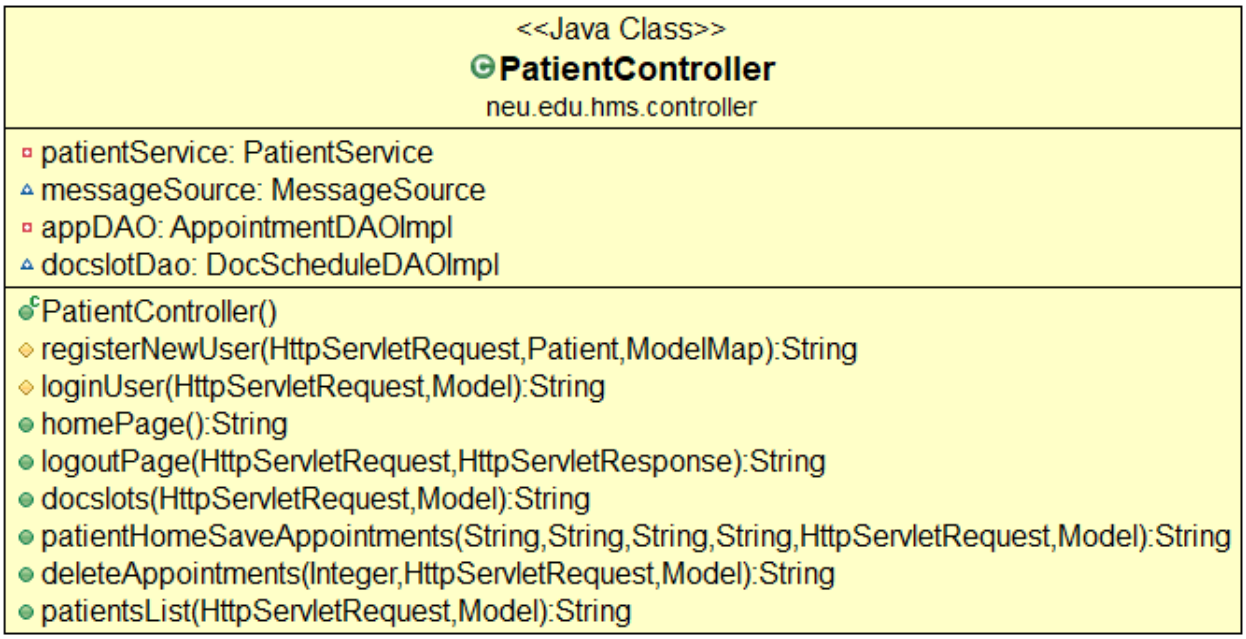
```
                System.out.println("Exception: " + e.getMessage());
                return "error";
            }
            return "registrationSuccess";
        }

        /**
         * This method checks for username and password entered by user and returns error page if
         * username or password entered is incorrect else navigates to home page of user
         */

        @RequestMapping(value = "/admin/login", method = RequestMethod.POST) protected
        String loginUser(HttpServletRequest request,Model model) throws Exception {

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

            try {

                System.out.print("loginUser");

                Admin a = adminService.get(request.getParameter("username"),
                        request.getParameter("password"));

                if(a == null){
                    System.out.println("UserName/Password does not exist");
                    session.setAttribute("errorMessage", "UserName/Password does not exist");
                    model.addAttribute("errorMessage", "Admin :" + a.getAdminName()+" does not exist");
```

```
                    return "error";
                }

                session.setAttribute("admin", a);
                return "admin-home";

            } catch (Exception e) { System.out.println("Exception: " + e.getMessage());
            session.setAttribute("errorMessage", "error while login"); return "error"; }

    }

    /**
     * Navigates to home page of user
     */

    @RequestMapping(value = "/admin/home", method = RequestMethod.GET)
    public String loginPage() {

            return "admin-home";
    }

    /**
     * Returns to login page once user clicks on logout
     */

    @RequestMapping(value="/admin/logout", method = RequestMethod.GET)
    public String logoutPage (HttpServletRequest request, HttpServletResponse response){
            Authentication auth = SecurityContextHolder.getContext().getAuthentication();
```

```
                if (auth != null){
                        SecurityContextHolder.getContext().setAuthentication(null);
                }
                return "redirect:/login?role=Admin";
        }

}
```

## PatientController:

## Class Diagram:




```
package neu.edu.hms.controller;


import java.util.List;

import java.util.Locale;

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.context.MessageSource;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.ui.ModelMap;
import org.springframework.validation.FieldError;
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.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import neu.edu.hms.DAO.AppointmentDAOImpl;
import neu.edu.hms.DAO.DocScheduleDAOImpl;
import neu.edu.hms.entity.Appointment;
import neu.edu.hms.entity.DocSchedule;
import neu.edu.hms.entity.Patient;
import neu.edu.hms.service.PatientService;

@Controller
public class PatientController {

	@Autowired
	private PatientService patientService;

	@Autowired
	MessageSource messageSource;

	@Autowired
```

```
private AppointmentDAOImpl appDAO;

@Autowired
DocScheduleDAOImpl docslotDao;

/**
 * /**
 * This method registers new Patient
 * @return registration success page
 */

@RequestMapping(value = "/patient/register", method = RequestMethod.POST)
protected String registerNewUser(HttpServletRequest
request,@ModelAttribute("patient") Patient patient,ModelMap model) throws Exception {

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

        if(patient.getPatientName() == null) {
                FieldError registerError = new
FieldError("patient","patientName",messageSource.getMessage("User name is not Valid",new
String[] {patient.getPatientName()}, Locale.getDefault()));
                return "home";
        }

        try {

                Patient pat = patientService.get(patient.getPatientName());

                if(pat == null) {
```

```
                    patientService.register(patient);
                    request.getSession().setAttribute("patient", patient);

                    model.addAttribute("success", "Patient: " +
patient.getPatientName() +" registered Successfully");
                }
                else {
                    session.setAttribute("errorMessage", "UserName already used");

                    return "error";
                }
            } catch (Exception e) {
                System.out.println("Exception: " + e.getMessage());
                return "error";
            }
            return "registrationSuccess";
        }

        /**
         * This method checks for username and password entered by user and returns error
page if
         * username or password is incorrect else navigates to home page of user
         */

        @RequestMapping(value = "/patient/login", method = RequestMethod.POST) protected
        String loginUser(HttpServletRequest request,Model model) throws Exception {

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

            try {
```

```
                Patient p = patientService.get(request.getParameter("username"),
                        request.getParameter("password"));

                List<Appointment> appList = appDAO.getAppointments();
                if(p == null){
                    System.out.println("UserName/Password does not exist");
                    session.setAttribute("errorMessage", "UserName/Password does
not exist");
                    return "error";
                }

                session.setAttribute("patient", p);
                model.addAttribute("applist", appList);
                return "patient-home";

            } catch (Exception e) { System.out.println("Exception: " + e.getMessage());
            session.setAttribute("errorMessage", "error while login"); return "error"; }

        }


        @RequestMapping(value = "patient/home", method = RequestMethod.GET)
        public String homePage() {

            return "patient-home";

        }
```

```
@RequestMapping(value="/patient/logout", method = RequestMethod.GET)
public String logoutPage (HttpServletRequest request, HttpServletResponse response){
	Authentication auth = SecurityContextHolder.getContext().getAuthentication();
	if (auth != null){
		SecurityContextHolder.getContext().setAuthentication(null);
	}
	return "redirect:/login?role=Patient";
}

/**
 * This method checks for available slots and returns the list in view page
 */

@RequestMapping(path="/patient/docslots", method = RequestMethod.GET)
public String docslots(HttpServletRequest request,Model model) throws Exception {

	try {
		List<DocSchedule> list = docslotDao.getslots();
		model.addAttribute("slotlist",list);

	}catch(NumberFormatException e) {
		System.out.print(e.getMessage());
	}

	return "show-slotlist";
}

/**
```

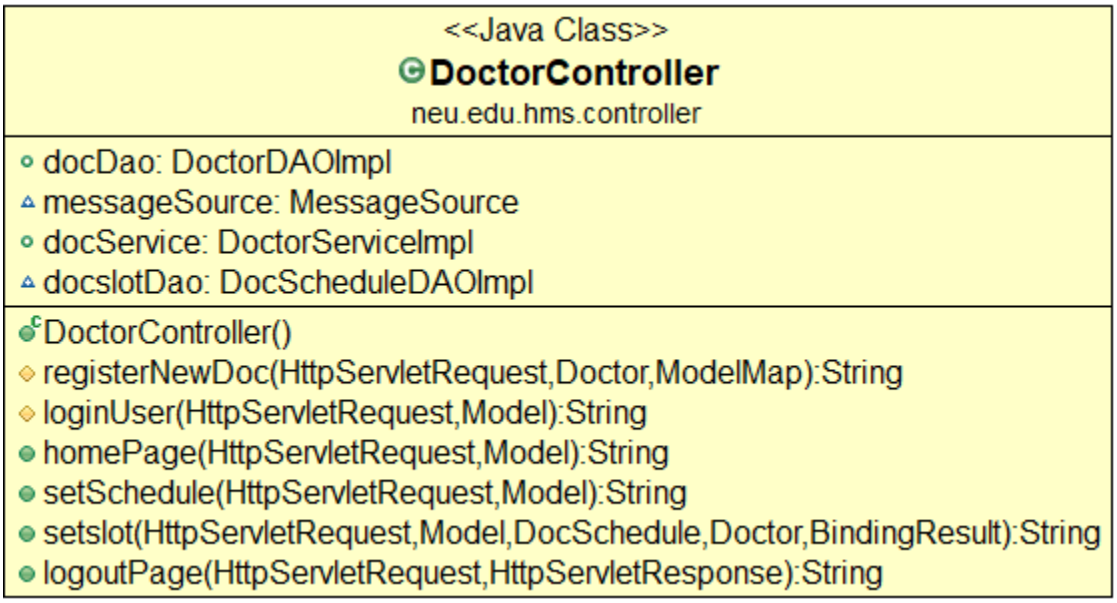
```
	 * This method adds selected date and time slot to appointments list and
	 * updates doctor schedules as booked
	 */
	@RequestMapping(value = "patient/appointments", method=RequestMethod.POST)
	@ResponseBody
	public String patientHomeSaveAppointments(@RequestParam String docslotid,
				@RequestParam String docname,
				@RequestParam String date,
				@RequestParam String time,HttpServletRequest request,Model model)
throws Exception {

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

		int docid = Integer.parseInt(docslotid);
		Patient p = (Patient) session.getAttribute("patient");
		String pname = p.getPatientName();

		appDAO.saveappointment(pname, docname,date, time);
		docslotDao.saveAppointment(docid);

		return "patient-home";
	}

	/**
	 * This method deletes selected booked slot by patient and updates
	 * doctor slots as available
	 */

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

```
@ResponseBody
public String deleteAppointments(@RequestParam Integer appId,HttpServletRequest
request,Model model) throws Exception {
        String doctime =null;
        String docname = null;
        try {

                List<Appointment> app = docslotDao.getAppointment(appId);
                for(Appointment a:app) {
                        doctime = a.getDocTime();
                        docname = a.getDoctorName();
                }
                docslotDao.updateAppointment(doctime,docname);
                appDAO.deleteappointment(appId);



                return "1";
        }catch(Exception e) {
                System.out.println("Cannot delete "+e.getMessage());
                return "-1";
        }
}
/**
 * This method returns list of patients to admin home page
 */

@RequestMapping(path="/patient/List", method = RequestMethod.GET)
public String patientsList(HttpServletRequest request,Model model) throws Exception {
```

```
		try {
			List<Patient> list = patientService.getPatientsList();
			if(list !=null) {
				model.addAttribute("patList",list);
			}else {
				model.addAttribute("listnull",list);
			}

		}catch(NumberFormatException e) {
			System.out.print(e.getMessage());
		}

		return "admin-home";
	}

}
```

## DoctorController:

## Class Diagram:



```java
package neu.edu.hms.controller;

import java.util.List;
import java.util.Locale;
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.context.MessageSource;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.validation.FieldError;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import neu.edu.hms.DAO.DocScheduleDAOImpl;
import neu.edu.hms.DAO.DoctorDAOImpl;
import neu.edu.hms.entity.DocSchedule;
import neu.edu.hms.entity.Doctor;
import neu.edu.hms.entity.Patient;
import neu.edu.hms.service.DoctorServiceImpl;

@Controller
public class DoctorController {

	@Autowired
	public DoctorDAOImpl docDao;

	@Autowired
	MessageSource messageSource;

	@Autowired
	public DoctorServiceImpl docService;

	@Autowired
	DocScheduleDAOImpl docslotDao;


	@RequestMapping(value = "/doctor/register", method = RequestMethod.POST)
	protected String registerNewDoc(HttpServletRequest
request,@ModelAttribute("doctor") Doctor doctor,ModelMap model) throws Exception {
```

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


		if(doctor.getDocName() == null) {
			FieldError registerError = new
FieldError("doctor","docName",messageSource.getMessage("User name is not Valid",new
String[] {doctor.getDocName()}, Locale.getDefault()));
			return "home";
		}

		try {

			Doctor doc = docService.get(doctor.getDocName());

			if(doc == null) {


				docService.register(doctor);
				request.getSession().setAttribute("doctor", doctor);

				model.addAttribute("success", "doctor: " + doctor.getDocName()
+" registered Successfully");
			}else {
				session.setAttribute("errorMessage", "UserName already used");
				return "error";
			}
		} catch (Exception e) {
			System.out.println("Exception: " + e.getMessage());
			return "error";
		}
		return "registrationSuccess";
	}

	@RequestMapping(value = "/doctor/login", method = RequestMethod.POST) protected
	String loginUser(HttpServletRequest request,Model model) throws Exception {

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

		try {

			Doctor p = docService.get(request.getParameter("username"),
				request.getParameter("password"));

			if(p == null){
				System.out.println("UserName/Password does not exist");
```

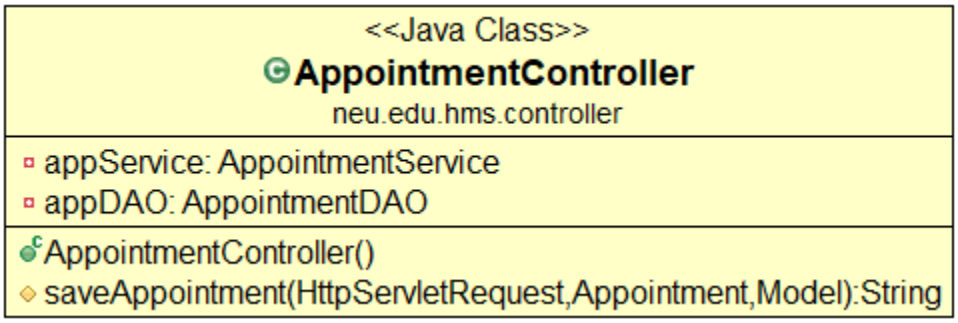
```
                                session.setAttribute("errorMessage", "UserName/Password does
not exist");
                                return "error";
                        }

                        session.setAttribute("doctor", p);
                        return "doctor-home";

                } catch (Exception e) { System.out.println("Exception: " + e.getMessage());
                session.setAttribute("errorMessage", "error while login"); return "error"; }

        }

        @RequestMapping(value = "/doctor/home", method = RequestMethod.GET)
        public String homePage(HttpServletRequest request , Model model) {
                HttpSession session = (HttpSession) request.getSession();
                Doctor doc = (Doctor) session.getAttribute("doctor");
                model.addAttribute("doctor",doc);
                return "doctor-home";
        }

        /**
         * This method returns page to add date and time by doctor.
         */
        @RequestMapping(value="/doctor/setSchedules",method=RequestMethod.GET)
        public String setSchedule(HttpServletRequest request , Model model) {

                HttpSession session = (HttpSession) request.getSession();
                Doctor doc = (Doctor) session.getAttribute("doctor");
                model.addAttribute("doctor", doc);
                model.addAttribute("docslot", new DocSchedule());
                return "doc-slot";
        }

        /**
         * This method adds given date and time by doctor to available schedules list
         * and returns success page.
         */
        @RequestMapping(value="/doctor/saveslot",method=RequestMethod.POST)
        public String setslot(HttpServletRequest request , Model
model,@ModelAttribute("docslot") DocSchedule docslot,
                        @ModelAttribute("doctor") Doctor doc,BindingResult result) {

                HttpSession session = (HttpSession) request.getSession();
                Doctor doctor = (Doctor)session.getAttribute("doctor");
                if (result.hasErrors()) {
```

```
                return "home"; }

            try {

                docslotDao.setSchedule(docslot,doctor);

                request.getSession().setAttribute("docslot", docslot);

                model.addAttribute("success", "doctor: " + doctor.getDocName() +"
scheduled Successfully");
                model.addAttribute("docslot",docslot);
            } catch (Exception e) {
                System.out.println("Exception: " + e.getMessage());
                return "error";
            }
            return "doc-schedules";
        }


        @RequestMapping(value="/doctor/logout", method = RequestMethod.GET)
        public String logoutPage (HttpServletRequest request, HttpServletResponse response){
            Authentication auth = SecurityContextHolder.getContext().getAuthentication();
            if (auth != null){
                SecurityContextHolder.getContext().setAuthentication(null);
            }
            return "redirect:/login?role=Doctor";
        }

        /**
         * This method returns list of doctors to admin home page
         */

        @RequestMapping(path="/doctors/List", method = RequestMethod.GET)
        public String doctorsList(HttpServletRequest request,Model model) throws Exception {

            try {
                List<Doctor> list = docService.getDoctorsList();
                if(list !=null) {
                    model.addAttribute("docList",list);
                }else {
                    model.addAttribute("listnull",list);
                }

            }catch(NumberFormatException e) {
                System.out.print(e.getMessage());
            }
```

```
            return "admin-home";
        }

}
```

## AppointmentController:

## Class Diagram:




```
package neu.edu.hms.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import neu.edu.hms.DAO.AppointmentDAO;
import neu.edu.hms.entity.Appointment;
import neu.edu.hms.entity.Patient;
import neu.edu.hms.service.AppointmentService;

@Controller
```

```
public class AppointmentController {

	@Autowired
	private AppointmentService appService;

	@SuppressWarnings("unused")
	@Autowired
	private AppointmentDAO appDAO;

	/**
	 * This method saves appointment slot selected by patient and adds it in home page
	 */
	@RequestMapping(value = "/patient/saveappointment", method =
RequestMethod.POST)
	protected String saveAppointment(HttpServletRequest
request,@ModelAttribute("appointment") Appointment app,Model model) throws Exception {

		Patient p = (Patient) request.getSession().getAttribute("patient");
		try {

			System.out.print("appointment");

			appService.saveAppointment(app);
			request.getSession().setAttribute("app", app);

			model.addAttribute("success", "Patient: " + p.getPatientName() +"
appointment saved Successfully");

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

```
                return "error";
            }

            return "patient-home";
        }

}
```

## Database Design: