INFO-6250 Web Development Tools & Methods SEC 01

FINAL PROJECT

Name: Sudhanshu Garg

NEU Id: 001267033

**Summary:**

Created a web based Job application portal that lets job seekers look and apply for the jobs they are interested in using spring MVC and hibernate. The application connects the recruiters and job seekers in the most effective and interactive way possible. The application has robust secure backend code with all sorts of data validation and sanitization in place.

**Functionalities Performed:**

1.Recruiters can post jobs on the portal which gets available to the job seekers.

2.Recruiters can view all jobs posted only by them based on their login credentials.

3.Jobseekers can view and search the jobs and can apply for the jobs they are interested in.

4.Jobseekers can view all the jobs applied by them including uploading a resume.

5.Recruiters can also view all the job applications for a particular job.

6.Recruiters can view all the details of the job seeker for the applied job in the job applications and can also **download** job seekers resume.

7.Recruiters can change the status of the job to not available once the position is filled and then that job couldn’t be applied anymore.

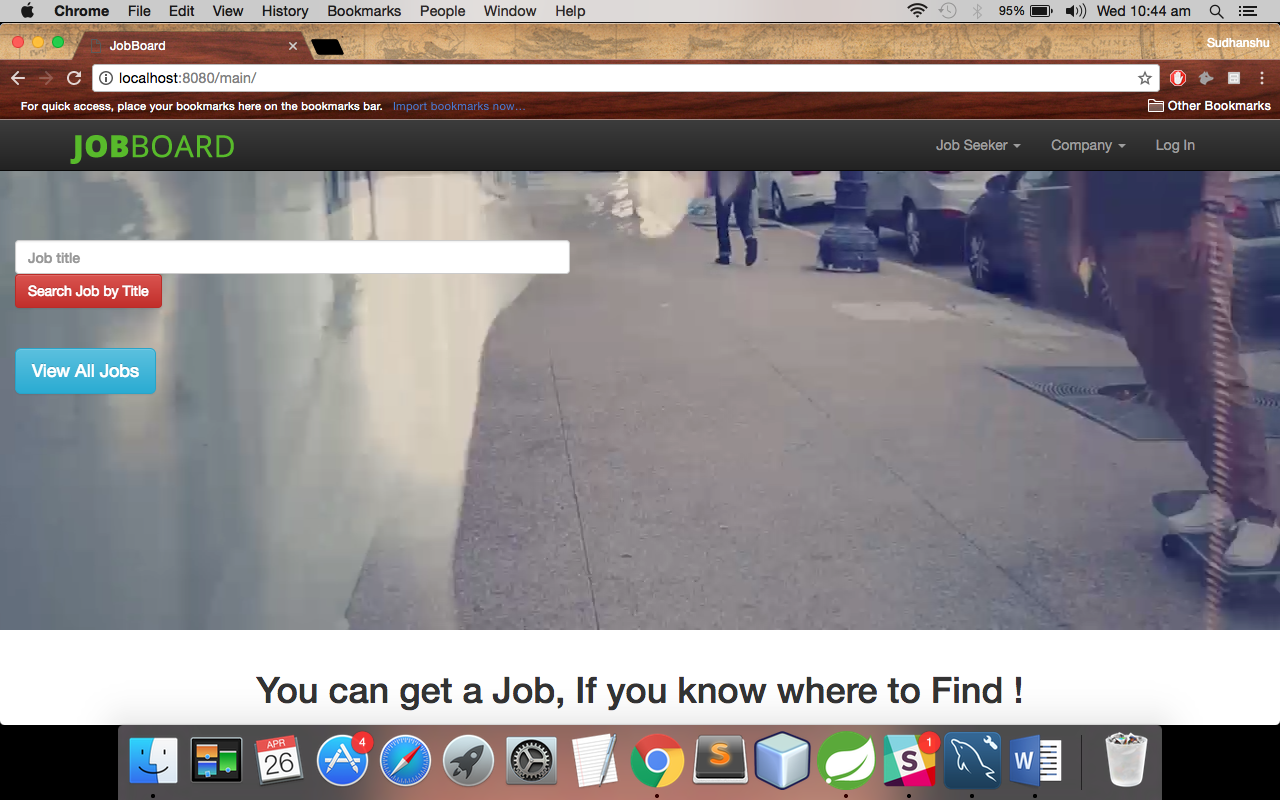
8.The home page is open to anyone. Anyone can visit the website, search and view all jobs. But once a user wants to apply for a job, session is checked. If the user is not logged in, he will be redirected to the login page to first login before applying to any job.

9.When the jobseeker applies to a job, the jobseekers an auto-generated confirmation email on his email Id. The company for that job, also gets an auto-generated email with the details of the job seeker and the job that has been applied.

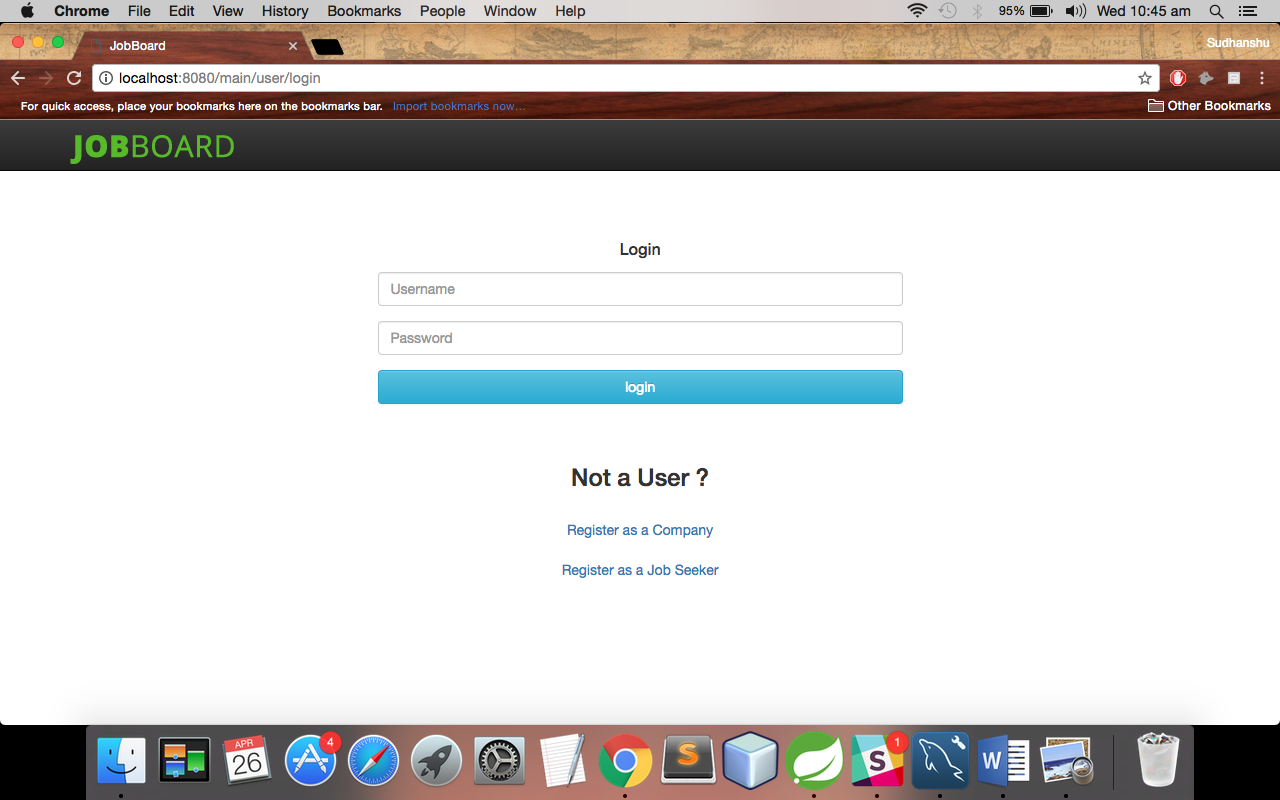
**10. Spring MVC (annotated controller and pojo’s), Validators, Exception Handling in specific classes, Interceptor, Velocity, DAO Pattern, Javax Mail, CommonMultiPart (File upload), Created UI using Bootstrap (although not required, but created for better user experience).**

SCREENSHOTS

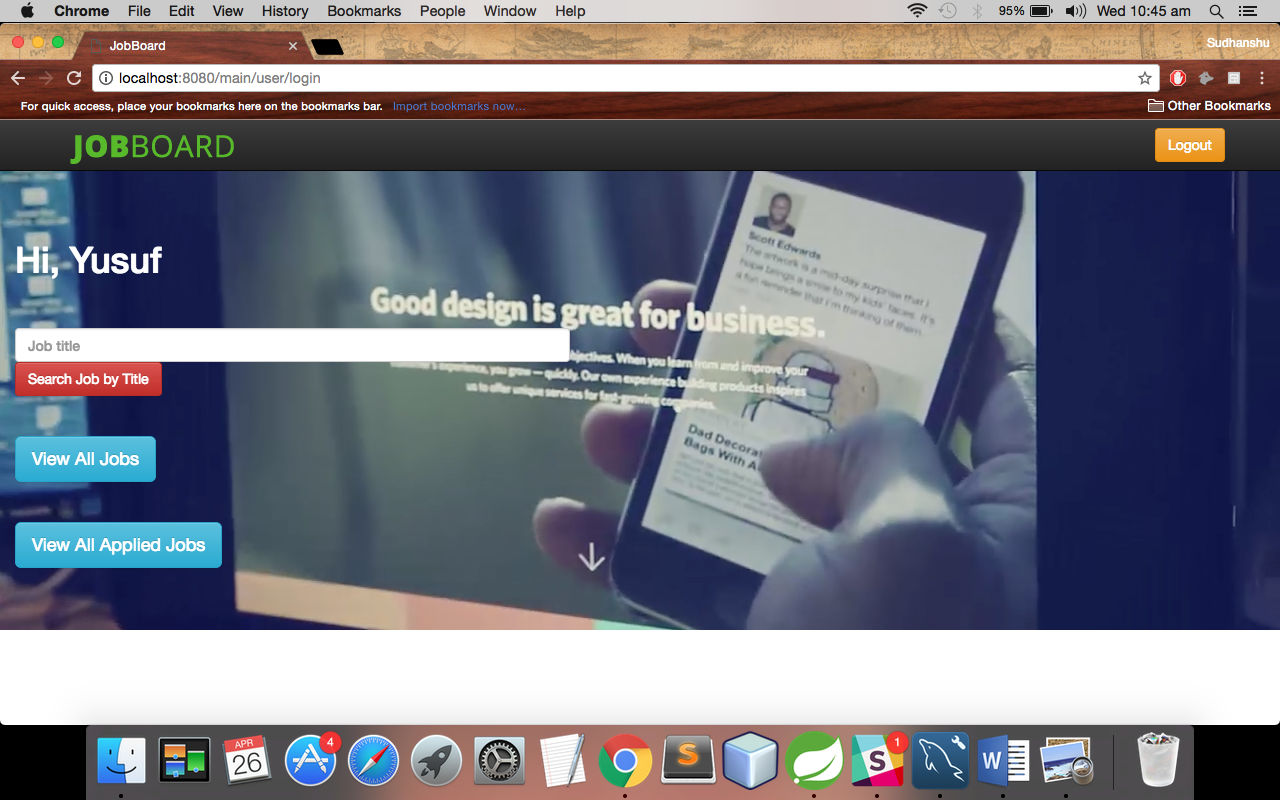
1. Home Page

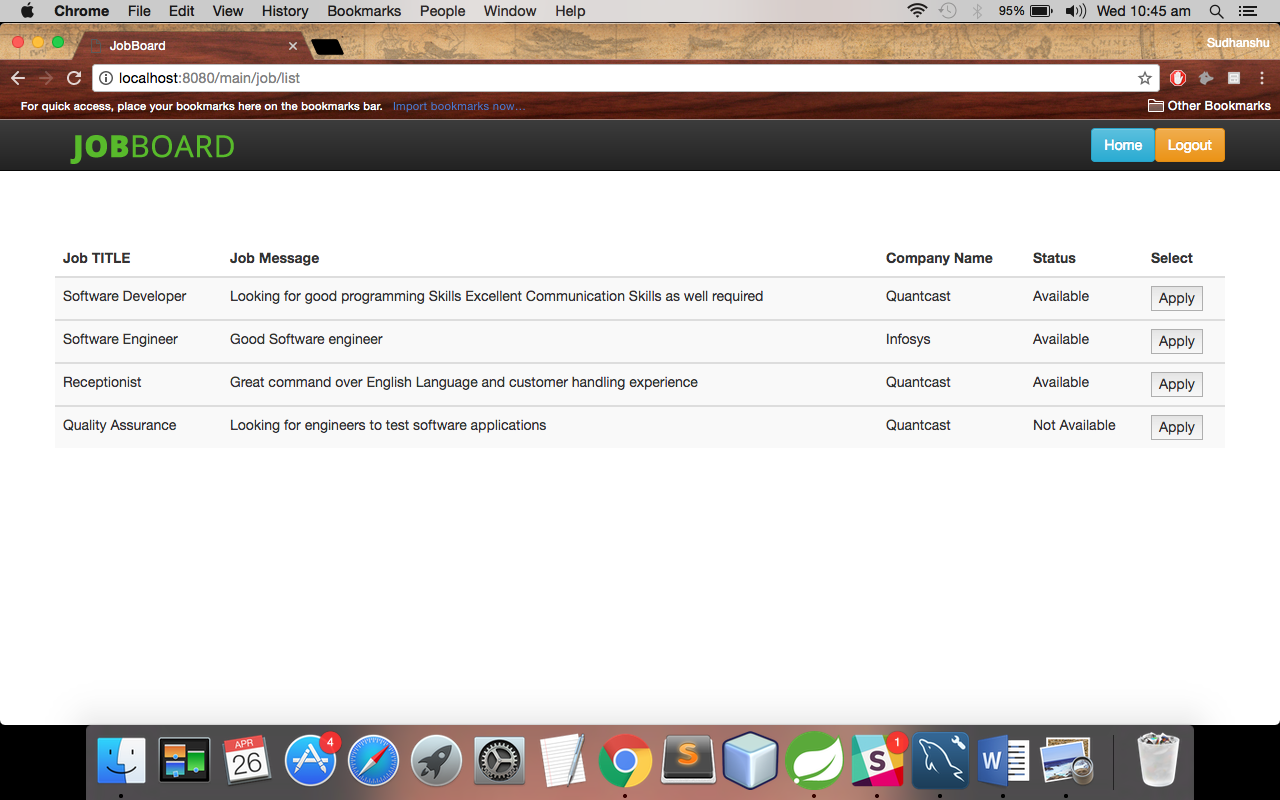


2. User Authentication

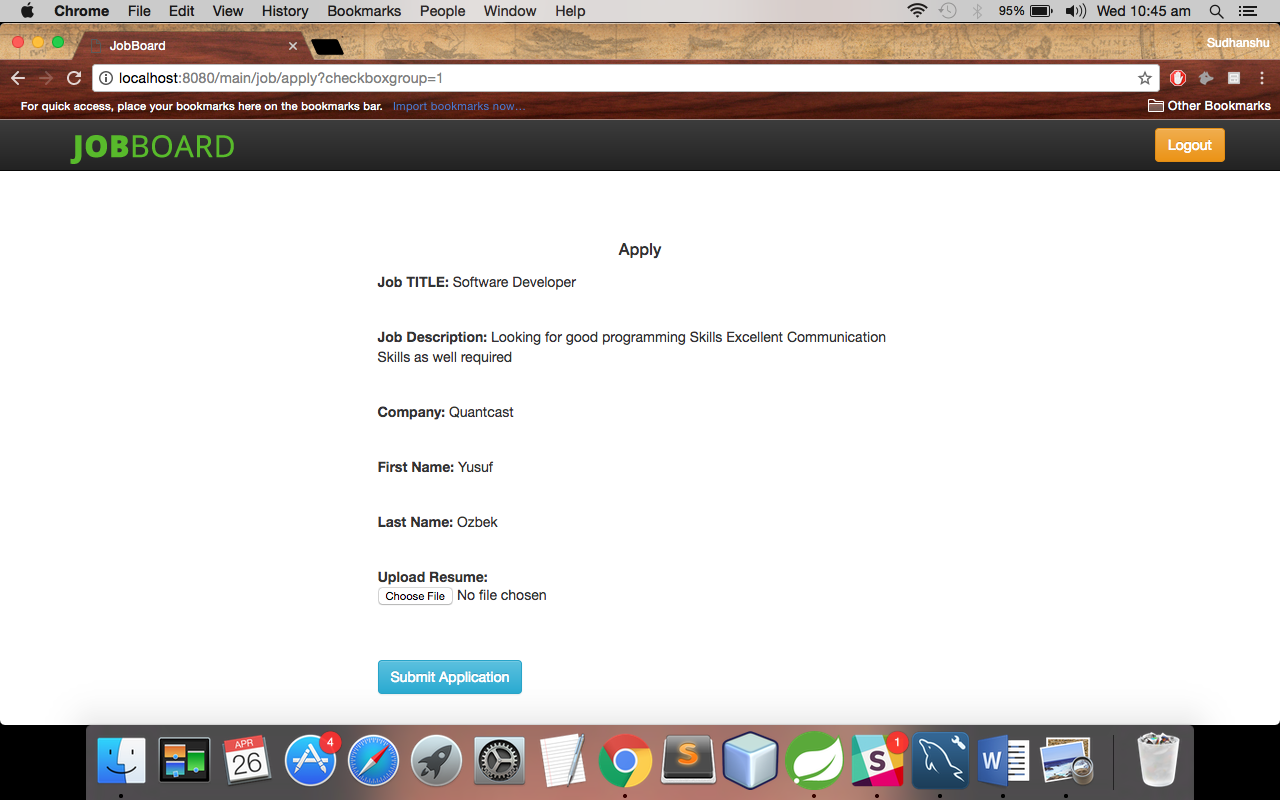


3. User Home Page after verifying login credentials based on his role

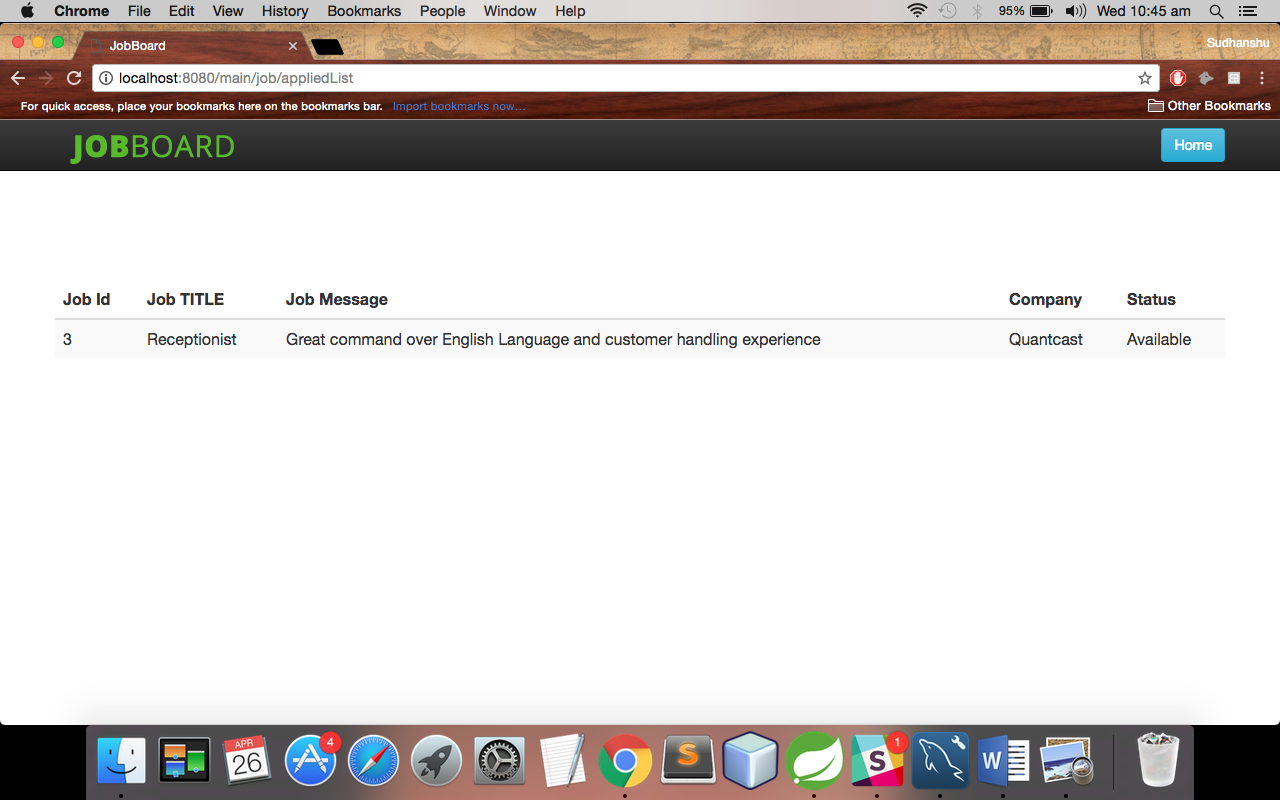


4. View All Jobs or Search for a job

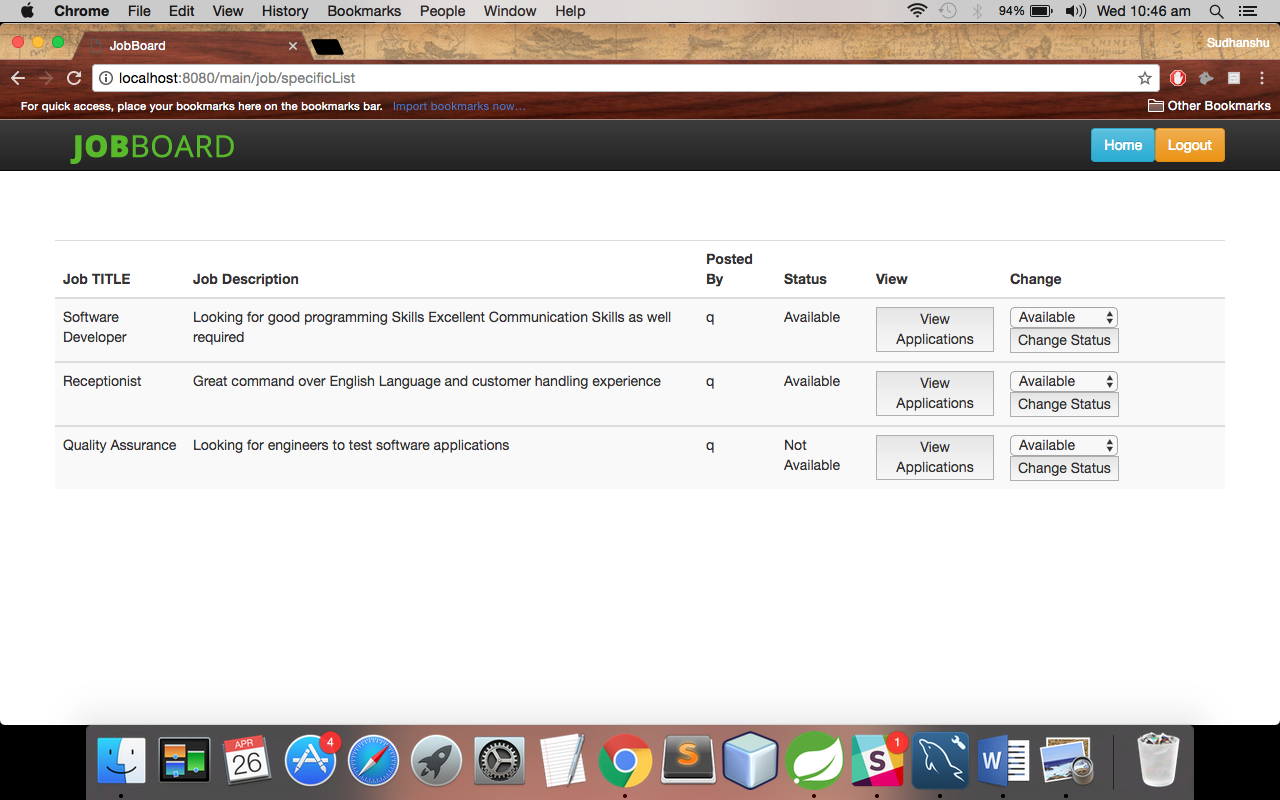
5. Apply to a Selected Job



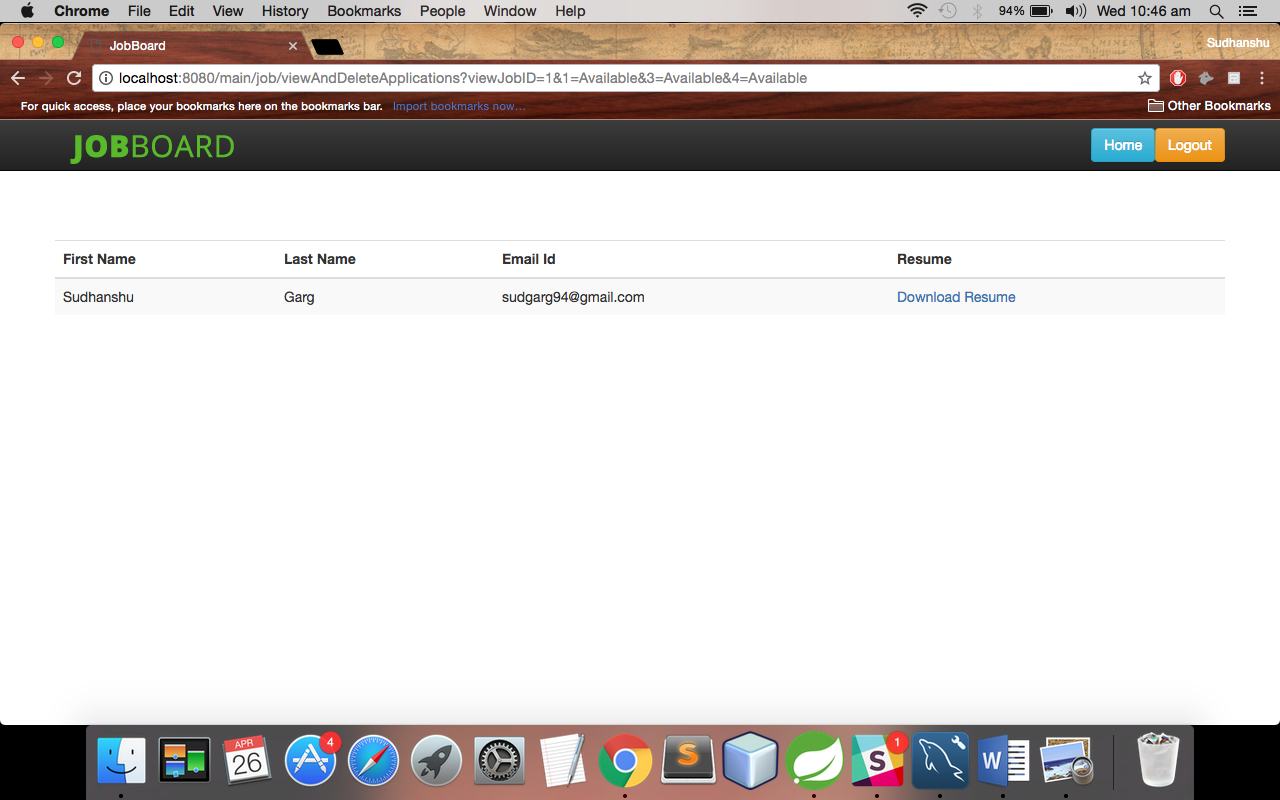
6. View All jobs applied List by Job Seeker



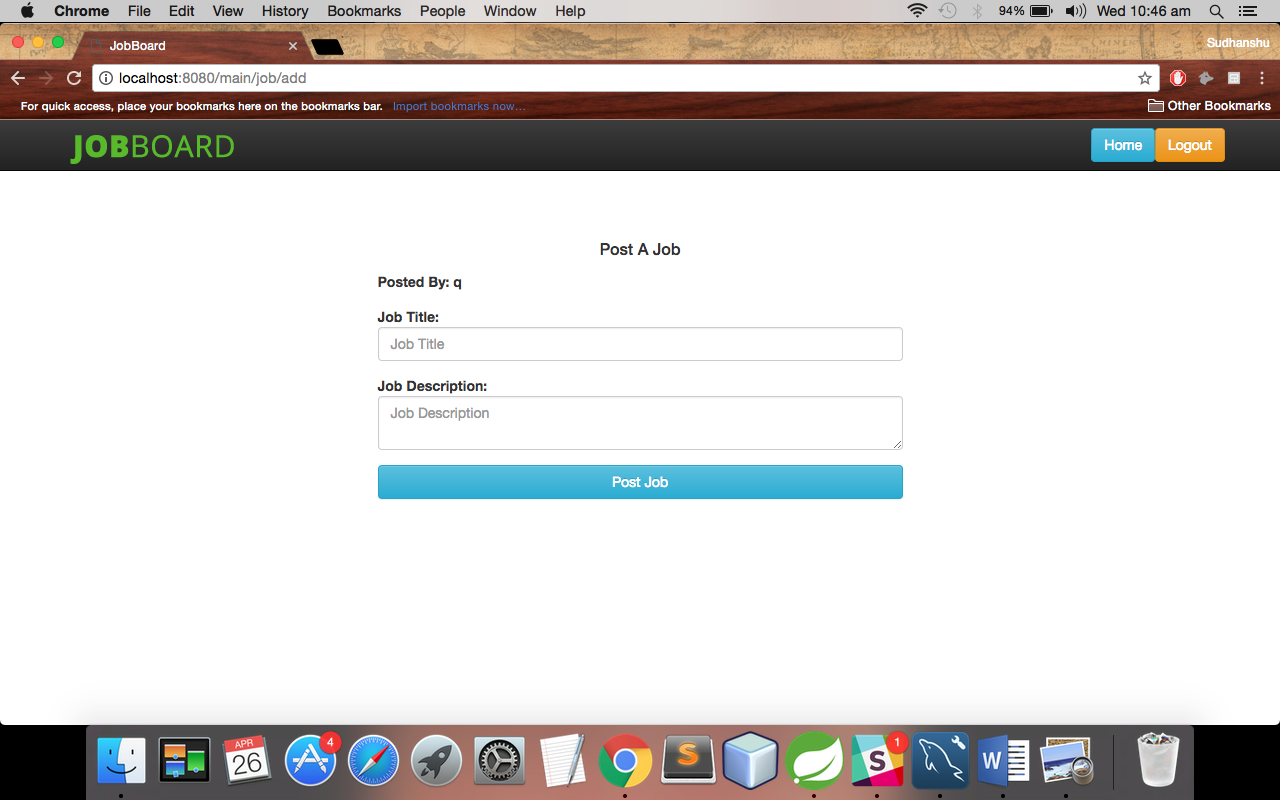
7. View All Jobs Posted by the Company



8. View All Applications to that Job



9. Post A Job by a Company



**CONTROLLERS:**

**USER CONTROLLER**

package com.tryproject.controller;

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.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.tryproject.exception.UserException;

import com.tryproject.pojo.Company;

import com.tryproject.pojo.JobSeeker;

import com.tryproject.pojo.User;

import com.tryproject.dao.UserDAO;

import com.tryproject.validator.CompanyValidator;

import com.tryproject.validator.JobSeekerValidator;

import com.tryproject.validator.UserValidator;

@Controller

@RequestMapping("/user/\*")

public class UserController {

@Autowired

@Qualifier("userDao")

UserDAO userDao;

@Autowired

@Qualifier("userValidator")

UserValidator uvalidator;

@Autowired

@Qualifier("companyValidator")

CompanyValidator cvalidator;

@Autowired

@Qualifier("jobseekerValidator")

JobSeekerValidator jvalidator;

@InitBinder("company")

private void initCompanyBinder(WebDataBinder binder) {

binder.setValidator(cvalidator);

}

@InitBinder("/user/registerJobSeeker")

private void initJobSeekerBinder(WebDataBinder binder) {

binder.setValidator(jvalidator);

}

@InitBinder("user/login")

private void initUserBinder(WebDataBinder binder) {

binder.setValidator(uvalidator);

}

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

protected ModelAndView registerCompany() throws Exception {

return new ModelAndView("register-company", "company", new Company());

}

@RequestMapping(value = "/user/registerCompany", method = RequestMethod.POST)

protected ModelAndView registerNewCompany(HttpServletRequest request, @ModelAttribute("company") Company company, BindingResult result) throws Exception {

cvalidator.validate(company, result);

if (result.hasErrors()) {

return new ModelAndView("register-company", "company", company);

}

try {

Company c = userDao.registerCompany(company);

request.getSession().setAttribute("user", c);

return new ModelAndView("user-home", "user", c);

} catch (UserException e) {

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

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//register Job Seeker

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

protected ModelAndView registerJobSeeker() throws Exception {

System.out.print("registerJobseeker");

return new ModelAndView("register-jobseeker", "jobseeker", new JobSeeker());

}

@RequestMapping(value = "/user/registerJobSeeker", method = RequestMethod.POST)

protected ModelAndView registerJobSeeker(HttpServletRequest request, @ModelAttribute("jobseeker") JobSeeker jobseeker, BindingResult result) throws Exception {

jvalidator.validate(jobseeker, result);

if (result.hasErrors()) {

return new ModelAndView("register-jobseeker", "jobseeker", jobseeker);

}

try {

System.out.print("registerJobseeker here");

JobSeeker j = userDao.registerJobSeeker(jobseeker);

request.getSession().setAttribute("user", j);

return new ModelAndView("user-home", "user", j);

} catch (UserException e) {

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

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//login

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

protected ModelAndView goToUserHome(HttpServletRequest request) throws Exception {

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

return new ModelAndView("login", "user", new User());

}

@RequestMapping(value = "/user/login", method = RequestMethod.POST)

protected ModelAndView loginUser(HttpServletRequest request,@ModelAttribute("user") User user, BindingResult result) throws Exception{

uvalidator.validate(user, result);

if (result.hasErrors()) {

System.out.println("inside result error");

return new ModelAndView("login", "user", user);

//return "login";

}

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

try {

System.out.print("loginUser");

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

Company c = userDao.getCompany(user.getUsername(), user.getPassword());

JobSeeker j = userDao.getJobSeeker(user.getUsername(), user.getPassword());

if(c!=null){

session.setAttribute("user", c);

return new ModelAndView("user-home", "user", c);

//return "user-home";

}

if(j!=null)

{

session.setAttribute("user", j);

return new ModelAndView("user-home", "user", j);

//return "user-home";

}

System.out.println("UserName/Password does not exist");

session.setAttribute("errorMessage", "UserName/Password does not exist");

return new ModelAndView("error", "errorMessage", "error while login");

//return "error";

} catch (UserException e) {

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

session.setAttribute("errorMessage", "error while login");

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//logout

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

public String logout(HttpServletRequest request){

request.getSession().invalidate();

return "redirect: /main";

}

//home

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

public String home(HttpServletRequest request){

return "user-home";

}

}

**JOB CONTROLLER**

package com.tryproject.controller;

import java.io.File;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import javax.mail.MessagingException;

import javax.mail.internet.MimeMessage;

import javax.servlet.ServletContext;

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.mail.MailSender;

import org.springframework.mail.SimpleMailMessage;

import org.springframework.mail.javamail.MimeMessageHelper;

import org.springframework.mail.javamail.MimeMessagePreparator;

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.multipart.MultipartHttpServletRequest;

import org.springframework.web.multipart.commons.CommonsMultipartFile;

import org.springframework.web.servlet.ModelAndView;

import com.tryproject.dao.DAO;

import com.tryproject.dao.JobApplicationDAO;

import com.tryproject.dao.JobDAO;

import com.tryproject.dao.UserDAO;

import com.tryproject.exception.JobException;

import com.tryproject.pojo.Company;

import com.tryproject.pojo.Job;

import com.tryproject.pojo.JobApplication;

import com.tryproject.pojo.JobSeeker;

import com.tryproject.pojo.User;

import com.tryproject.validator.CompanyValidator;

import com.tryproject.validator.JobApplicationValidator;

import com.tryproject.validator.JobValidator;

@Controller

@RequestMapping("/job/\*")

public class JobController {

@Autowired

@Qualifier("jobDao")

JobDAO jobDao;

@Autowired

@Qualifier("jobValidator")

JobValidator jvalidator;

@Autowired

@Qualifier("userDao")

UserDAO userDao;

@Autowired

@Qualifier("jobApplicationDao")

JobApplicationDAO jobApplicationDao;

@Autowired

@Qualifier("jobApplicationValidator")

JobApplicationValidator jApplicationValidator;

@Autowired

@Qualifier("mailSender")

private MailSender mailSender;

@Autowired

ServletContext servletContext;

@InitBinder("job/add")

private void initJobBinder(WebDataBinder binder) {

binder.setValidator(jvalidator);

}

@InitBinder("job/apply")

private void initJobApplicationBinder(WebDataBinder binder) {

binder.setValidator(jApplicationValidator);

}

@RequestMapping(value = "/job/add", method = RequestMethod.POST)

public ModelAndView addJob(@ModelAttribute("job") Job job, BindingResult result) throws Exception {

jvalidator.validate(job, result);

if (result.hasErrors()) {

return new ModelAndView("job-form", "job", job);

}

try {

Company c = userDao.get(job.getPostedBy());

job.setCompany(c);

job.setStatus("Available");

job = jobDao.create(job);

return new ModelAndView("job-success", "job", job);

}

catch(JobException e){

System.out.println(e.getMessage());

return new ModelAndView("error", "errorMessage", "error while login");

}

}

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

public ModelAndView initializeForm(HttpServletRequest request) throws Exception {

System.out.println("inside Job add");

ModelAndView mv = new ModelAndView();

mv.addObject("job", new Job());

mv.setViewName("job-form");

return mv;

}

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

public ModelAndView specificJobListing(HttpServletRequest request) throws Exception {

try {

System.out.println("inside job sepcific list");

//trying company specific list here

Company c = (Company) request.getSession().getAttribute("user");

String userID = String.valueOf(c.getUserID());

System.out.println(userID);

List<Job> jobs = jobDao.specificList(userID);

return new ModelAndView("job-list", "jobs", jobs);

} catch (JobException e) {

System.out.println(e.getMessage());

return new ModelAndView("error", "errorMessage", "error while login");

}

}

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

public ModelAndView addCategory(HttpServletRequest request) throws Exception {

try {

List<Job> jobs= jobDao.list();

return new ModelAndView("job-list", "jobs", jobs);

} catch (JobException e) {

System.out.println(e.getMessage());

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//trying code for applying here

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

public ModelAndView apply(HttpServletRequest request) throws Exception {

System.out.println("inside Job apply");

ModelAndView mv = new ModelAndView();

long j = Long.parseLong(request.getParameter("checkboxgroup"));

Job job = jobDao.get(j);

if(job.getStatus().equals("Not Available")){

mv.setViewName("job-not-available");

return mv;

}

JobSeeker jobseeker = (JobSeeker) request.getSession().getAttribute("user");

List<JobApplication> jobApplicationList = jobApplicationDao.list();

for(JobApplication jobApplication : jobApplicationList){

if(jobApplication.getJobseeker().getUserID() == jobseeker.getUserID() && jobApplication.getJob().getId()==job.getId())

{

mv.setViewName("job-applied-already");

return mv;

}

}

mv.addObject("job", job);

mv.addObject("jobApplication", new JobApplication());

mv.setViewName("apply");

return mv;

//return new ModelAndView("apply", "job", job);

}

//new post method here WORKING NOW. ASK TA HOW TO GIVE 2 COMMAND NAMES WHICH IS NOT POSSIBLE. WHAT IS THE WAY OUT.

@RequestMapping(value = "/job/apply", method = RequestMethod.POST)

public ModelAndView applyJobTry(HttpServletRequest request,@ModelAttribute("jobApplication") JobApplication jobApplication, BindingResult result) throws Exception {

jApplicationValidator.validate(jobApplication, result);

if (result.hasErrors()) {

ModelAndView mv = new ModelAndView();

Long jobId = Long.parseLong(request.getParameter("jobId"));

Job job = jobDao.get(jobId);

mv.addObject("job", job);

mv.addObject("jobApplication", jobApplication);

mv.setViewName("apply");

return mv;

}

try {

//JobApplication jobApplication= new JobApplication();

ModelAndView mv = new ModelAndView();

Long jobId = Long.parseLong(request.getParameter("jobId"));

Job job = jobDao.get(jobId);

jobApplication.setJob(job);

JobSeeker jobseeker = (JobSeeker) request.getSession().getAttribute("user");

jobApplication.setJobseeker(jobseeker);

//SETTING FILENAME AS APPLICATION ID

long count = jobApplicationDao.getCount();

long nextCount = count+1;

String applicationID = String.valueOf(nextCount);

//FILE UPLOAD

// We need to transfer to a file

File localFile = null;

CommonsMultipartFile photoInMemory = jobApplication.getPhoto();

String name = photoInMemory.getOriginalFilename();

if(name.endsWith("doc")){

localFile = new File("/Users/kkgarg/Desktop/allphotos/resumes/", applicationID+".doc");

}

else if(name.endsWith("pdf")){

localFile = new File("/Users/kkgarg/Desktop/allphotos/resumes/", applicationID+".pdf");

}

// move the file from memory to the file

photoInMemory.transferTo(localFile);

String fileName = localFile.getName();

System.out.println("FileName that is uploaded is "+ fileName);

// String lp = localFile.getPath();

jobApplication.setResume(fileName);

System.out.print("registerNewApplication");

jobApplicationDao.create(jobApplication);

//SENDING EMAIL TO JOBSEEKER

SimpleMailMessage email = new SimpleMailMessage();

email.setTo(jobseeker.getEmailID());

email.setSubject("JobBoard Application");

email.setText("Hi "+jobseeker.getFirstName()+". Thanks for applying to "+job.getTitle()+" from "+job.getCompany().getCompanyName());

mailSender.send(email);

//ends

//SENDING EMAIL TO COMPANY

SimpleMailMessage email1 = new SimpleMailMessage();

email1.setTo("garg.su@husky.neu.edu");

email1.setSubject("JobBoard Application");

email1.setText("Hi "+job.getCompany().getCompanyName()+". "+jobseeker.getFirstName()+" applied to your "+job.getTitle()+" job. "

+ "Please Login to our portal to know more details");

mailSender.send(email1);

//ends

mv.addObject("job",job);

mv.addObject("jobApplication",jobApplication);

mv.setViewName("job-applied-success");

return mv;

}

catch(Exception e){

System.out.println(e.getMessage());

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//ends here

// VIEW APPLICATIONS

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

public ModelAndView viewApplications(HttpServletRequest request) throws Exception {

try {

ModelAndView mv = new ModelAndView();

if(request.getParameter("viewJobID")!=null)

{

String jobID = request.getParameter("viewJobID");

List<JobSeeker> jobseeker = jobApplicationDao.getApplicationUsers(jobID);

List<JobApplication> jobApplication = jobApplicationDao.getApplication(jobID);

mv.addObject("jobApplication",jobApplication);

mv.setViewName("application-list");

}

if(request.getParameter("deleteJobID")!=null){

Long jobId = Long.parseLong(request.getParameter("deleteJobID"));

Job job = jobDao.get(jobId);

String jobid = String.valueOf(job.getId());

// jobDao.delete(job);

if(request.getParameter(jobid).equals("Available")){

job.setStatus("Available");

}

else if(request.getParameter(jobid).equals("Not Available")){

job.setStatus("Not Available");

}

Company c = (Company) request.getSession().getAttribute("user");

String userID = String.valueOf(c.getUserID());

List<Job> jobs = jobDao.specificList(userID);

mv.addObject("jobs",jobs);

mv.setViewName("job-list");

}

return mv;

} catch (JobException e) {

System.out.println(e.getMessage());

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//ends here

//View Applied List by jobseeker

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

public ModelAndView viewAppliedList(HttpServletRequest request) throws Exception {

try {

JobSeeker jobseeker = (JobSeeker)request.getSession().getAttribute("user");

String userID = String.valueOf(jobseeker.getUserID());

List<JobApplication> jobApplication = jobApplicationDao.getApplicationsApplied(userID);

ModelAndView mv = new ModelAndView();

mv.addObject("jobApplication",jobApplication);

mv.setViewName("jobseeker-applied-list");

return mv;

} catch (JobException e) {

System.out.println(e.getMessage());

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//ends here

//Search

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

public ModelAndView searchJob(HttpServletRequest request) throws Exception {

try {

String jobTitle = request.getParameter("jobTitle");

List<Job> allJobs = jobDao.list();

List<Job> jobs = new ArrayList<Job>();

for(Job job : allJobs){

if(job.getTitle().contains(jobTitle)){

jobs.add(job);

}

}

return new ModelAndView("job-list", "jobs", jobs);

} catch (JobException e) {

System.out.println(e.getMessage());

return new ModelAndView("error", "errorMessage", "error while login");

}

}

//

}