**CSYE 6220 Enterprise Software Design**

**Spring 2021**

**Project Report**

**Project title: Tech World Hustle**

Contents

[1. Project Summary: 2](#_Toc70809961)

[2. Project Functionalities: 3](#_Toc70809962)

[2.1 Object model: 3](#_Toc70809963)

[2.2 Key Functionalities: 3](#_Toc70809964)

[3. Technologies used: 4](#_Toc70809965)

[4. User roles and their respective tasks: 4](#_Toc70809966)

[4.1 For both roles: 4](#_Toc70809967)

[4.2 For Candidates: 4](#_Toc70809968)

[4.3 For Recruiters: 5](#_Toc70809969)

[5. Screenshots: 6](#_Toc70809970)

[5.1 Login page: 6](#_Toc70809971)

[5.2 Register Candidate 6](#_Toc70809972)

[5.3 Register Recruiter 6](#_Toc70809973)

[5.4 View Open jobs 7](#_Toc70809974)

[5.5 Advanced search for jobs: 7](#_Toc70809975)

[5.6 View application status 8](#_Toc70809976)

[5.7 Add Company review: 8](#_Toc70809977)

[5.8 Add salary details 9](#_Toc70809978)

[5.9 View company Reviews 10](#_Toc70809979)

[5.10 View Ranking Board 10](#_Toc70809980)

[5.11 View average salary and salary reports 10](#_Toc70809981)

[5.12 View walk-in drives 11](#_Toc70809982)

[5.13 Add job posting 11](#_Toc70809983)

[5.14 View Jobs posted 12](#_Toc70809984)

[5.15 View Applications 12](#_Toc70809985)

[5.16 View Candidates 13](#_Toc70809986)

[5.17 Advanced search candidates 13](#_Toc70809987)

[5.18 Email Alert 13](#_Toc70809988)

[6. Controller Source codes: 14](#_Toc70809989)

[6.1 LoginController: 14](#_Toc70809990)

[6.2 RegisterCandidateController 17](#_Toc70809991)

[6.3 RegisterRecruiterController 20](#_Toc70809992)

[6.4 CandidateActionController 21](#_Toc70809993)

[6.5 RecruiterActionController: 28](#_Toc70809994)

[6.6 WalkInDriveController 35](#_Toc70809995)

[6.7 ReviewController 37](#_Toc70809996)

[6.8 MarketValueSalaryController: 41](#_Toc70809997)

[6.9 JobAlertsSchedular 43](#_Toc70809998)

[7. DAO source codes: 45](#_Toc70809999)

[7.1 DAO: 45](#_Toc70810000)

[7.2 ApplicationDAO 47](#_Toc70810001)

[7.3 CandidateDAO 48](#_Toc70810002)

[7.4 CompanyDAO 51](#_Toc70810003)

[7.5 CompanyReviewDAO 51](#_Toc70810004)

[7.6 IndividualRatingDAO 53](#_Toc70810005)

[7.7 JobAlertsDAO 55](#_Toc70810006)

[7.8 JobDAO 57](#_Toc70810007)

[7.9 MarketValueSalaryDAO 60](#_Toc70810008)

[7.10 RecruiterDAO 62](#_Toc70810009)

[7.11 SkillDAO 63](#_Toc70810010)

[7.12 WalkInDriveDAO 65](#_Toc70810011)

[8. LoginInterceptor Source code: 67](#_Toc70810012)

# Project Summary:

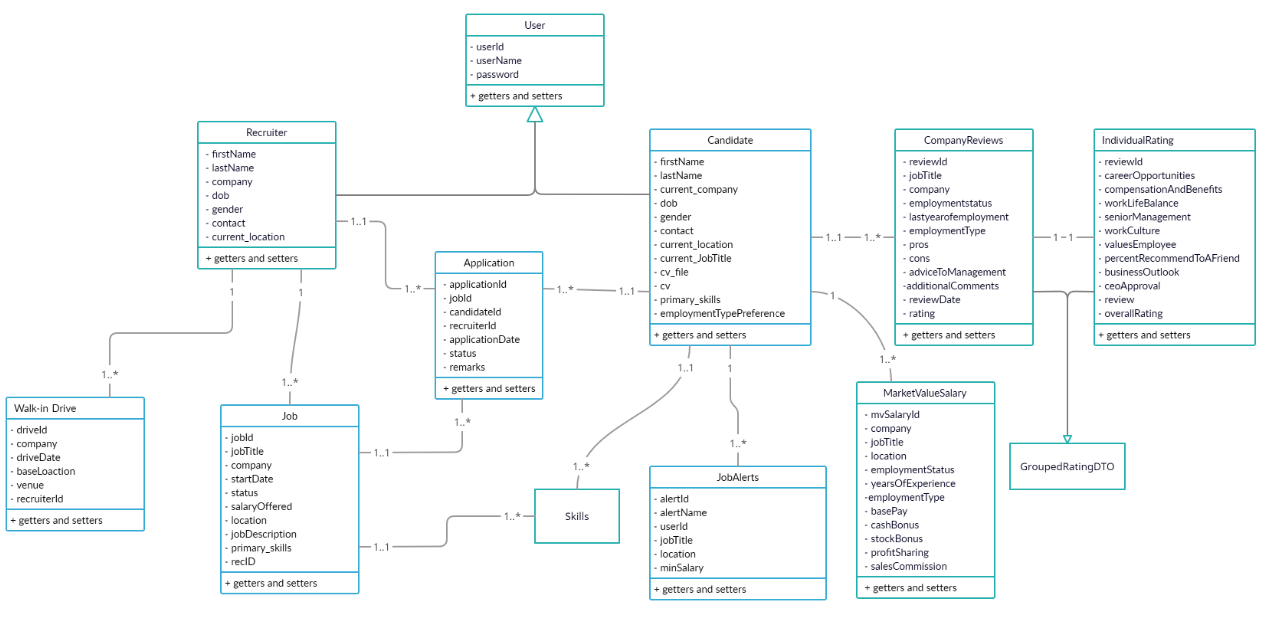
In today’s fast evolving tech world, recruiters and applicants hustle from one web application to another for hiring and for getting hired. This web application provides a common platform to recruiters and various applicants for all the activities related to the job searches - adding company reviews, analysing their market worth, providing fair job offers, posting and reviewing job opportunities, etc. This application is inspired and an intersection of Glassdoor and other job search portals like LinkedIn and Indeed.

# Project Functionalities:

## 2.1 Object model:

The attached is high-resolution image of the class model used for this project.

****

****

## Key Functionalities:

1. Registration for roles (Recruiters and Candidates) – using form tags of spring MVC and HQL to save data in MySQL. Also, used validator of spring MVC to check values of various fields.
2. Login-Logout for respective roles – validates the username and password from DB and starts the user- specific session.
3. Use of interceptors to prevent accessing any view pages/functionalities without logging-in.
4. Searching and applying for suitable jobs by candidates – using Restrictions and HQL.
5. Adding company reviews, ratings and salary information – using HQL
6. Providing candidates with information regarding company reviews, ratings and salaries – using Projections, aggregations and Order
7. Providing candidates with information regarding company ranking – using Data Transformer, Projections, Order and aggregations
8. Alerting candidates with their preferred job postings – using Query By Example (QBE) of hibernate and Email and SimpleEmail class of apache Commons. Also, used Google’s Gmail API for SMTP server
9. Searching suitable candidates by recruiters – using Restrictions and HQL
10. Recruiters can post various job opportunities and walk-in drives– using HQL

# Technologies used:

* Frameworks: Spring MVC 5x, Hibernate 4x
* Language: Java
* IDE: NetBeans
* Other supporting servers/tools: Tomcat 9

# User roles and their respective tasks:

The two key roles in this project are:

1) Candidates

2) Recruiters

## 4.1 For both roles:

Below mentioned are the activities/tasks **for both the roles** –

1. Registration with basic details (that will pass over interceptor and hence the registration page is accessible to all. Unique key constraint has been applied on username-email to prevent re-registration). Also, all the fields are being validated using spring MVC validator before successful registration.
2. Login using the username and password (it will be checked from DB and on successful validation, session id will be stored which is checked by interceptor before accessing any view-page.)
3. Logout to end the session.

---------------------------------------------------------------------------------------------------------------

## 4.2 For Candidates:

The following are the key activities/tasks that can be performed by **the candidates**:

1. Candidates can update their profile (post registration) - key skills, current company, current job title, etc.
2. Candidates can search and view exhaustive list of jobs posted by all the recruiters (sorted over salary being offered and start date of the jobs – used Order class.)
3. Candidates can perform advanced(preference) search based on various parameters like – location, job title, company, minimum salary expectation, employment type (full time, part time, freelancer, contract or intern) and keyskills required by the job (used Restrictions over each parameter).
4. Candidates can save their preferences of job parameters as alerts. They will receive weekly email alerts (scheduled a cron job using @Scheduled and @EnableScheduling) that will state the number of jobs that match their preferences (used QBE- Query by Example to implement job matching).
5. Candidates can apply directly for the jobs from the results obtained via search.
6. Candidates can withdraw their applications in case they change their mind.
7. Candidates can view the status of their application (under processing, accepted or rejected) anytime they want.
8. Candidates can view Walk-in drives posted in ascending order of dates and can plan their schedule.
9. Candidates can anonymously add reviews (pros, cons, comments, etc.) for the former and/or current company.
10. Candidates can anonymously provide ratings to the former and/or current company based on various parameters like work-life balance, opportunities, senior management, etc.
11. Candidates can search and view all the reviews and ratings based on company name and job title.
12. Candidates can search(check) average salary being offered for a particular job title over various locations and companies for better understanding of his/her market value.
13. Candidates can view ranking of the companies based on the ratings provide by all the candidates (used Data Transformer Object of hibernate).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

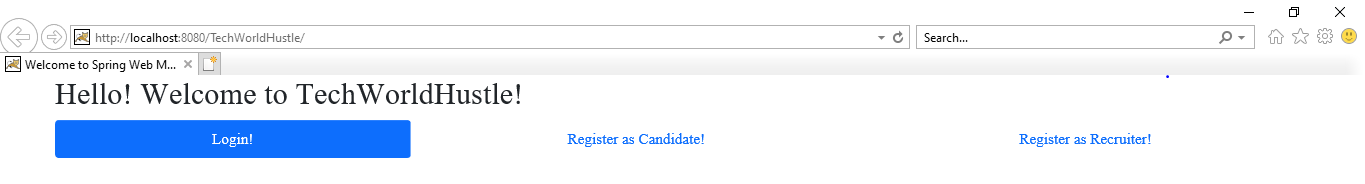
## 4.3 For Recruiters:

The following are the key activities/tasks that can be performed by **the recruiters**:

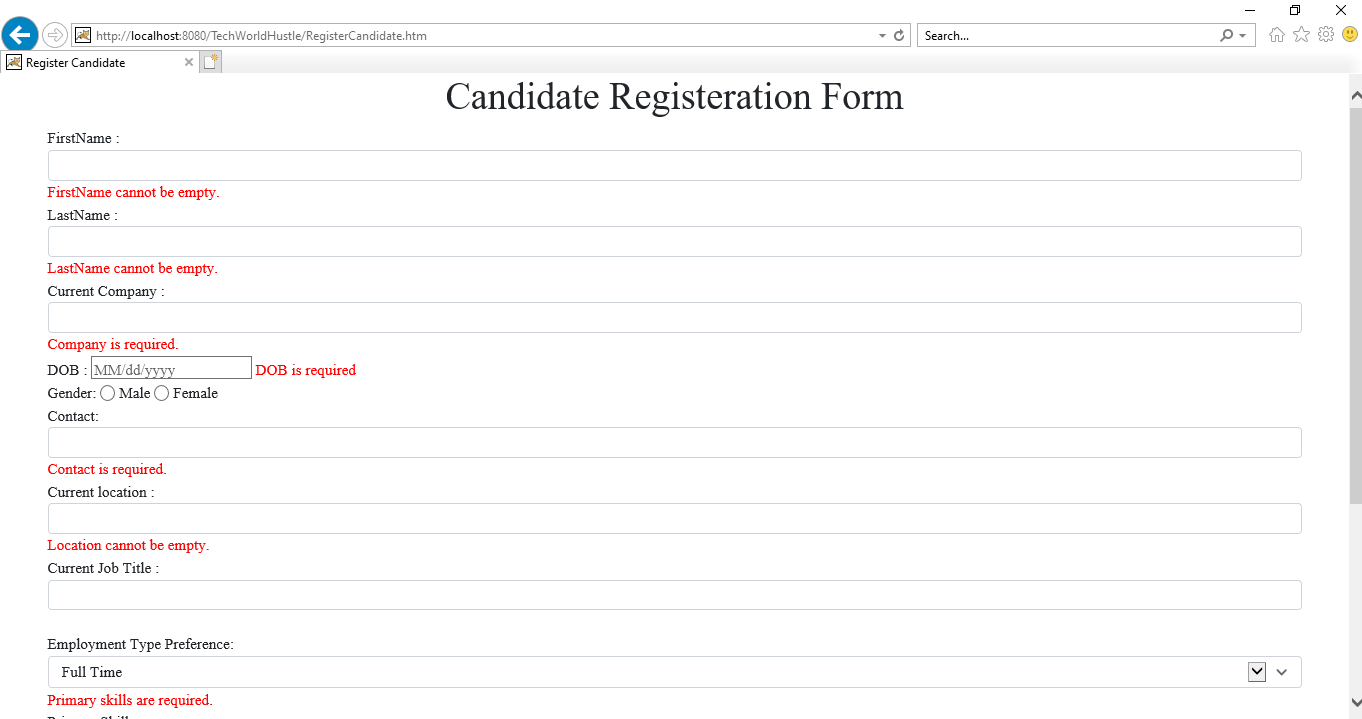
1. Recruiters can update their profile (post registration) - current company, location, etc.
2. Recruiters can post various job openings with the detailed parameters – key skills required, start date, salary being offered, job title, company, etc.
3. Recruiters can update job details posted by them and can even close the job openings (which will then no longer be visible to candidates on searching).
4. Recruiters can search and view exhaustive list of candidates that are actively looking for jobs.
5. Recruiters can perform advanced search of candidates based on various parameters like – location, job title, employment type (full time, part time, freelancer, contract or intern) and key skills required by the job (used Restrictions over each parameter).
6. Recruiters can view applications received for the jobs that they have posted(own) – except the ones that had been withdrawn by the applicant (candidate) itself.
7. Recruiters can update the application status (under processing, accepted or rejected) and add remarks on the applications received.
8. Recruiters can post Walk-in drives for various companies.
9. Recruiters can update Walk-in drives posted by them.

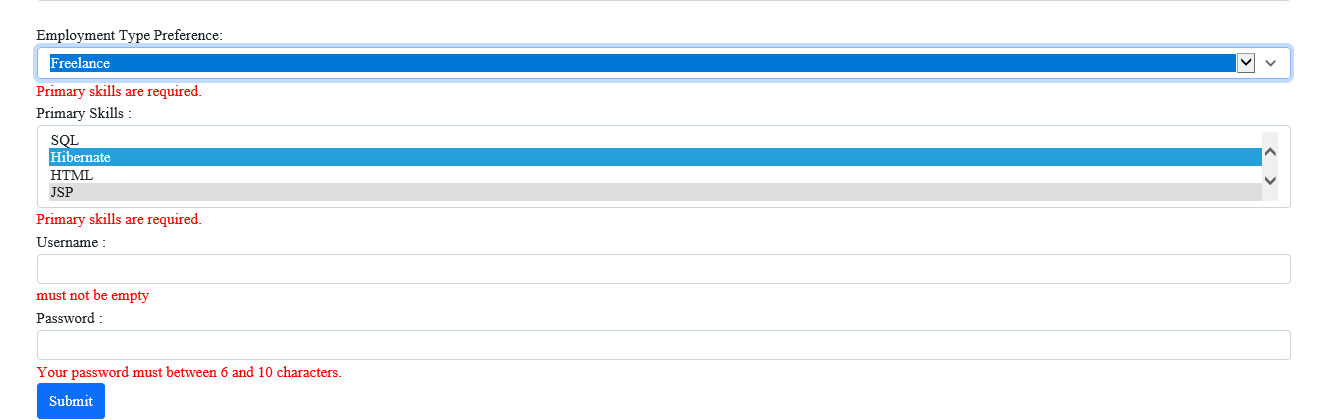
# Screenshots:

## 5.1 Login page:

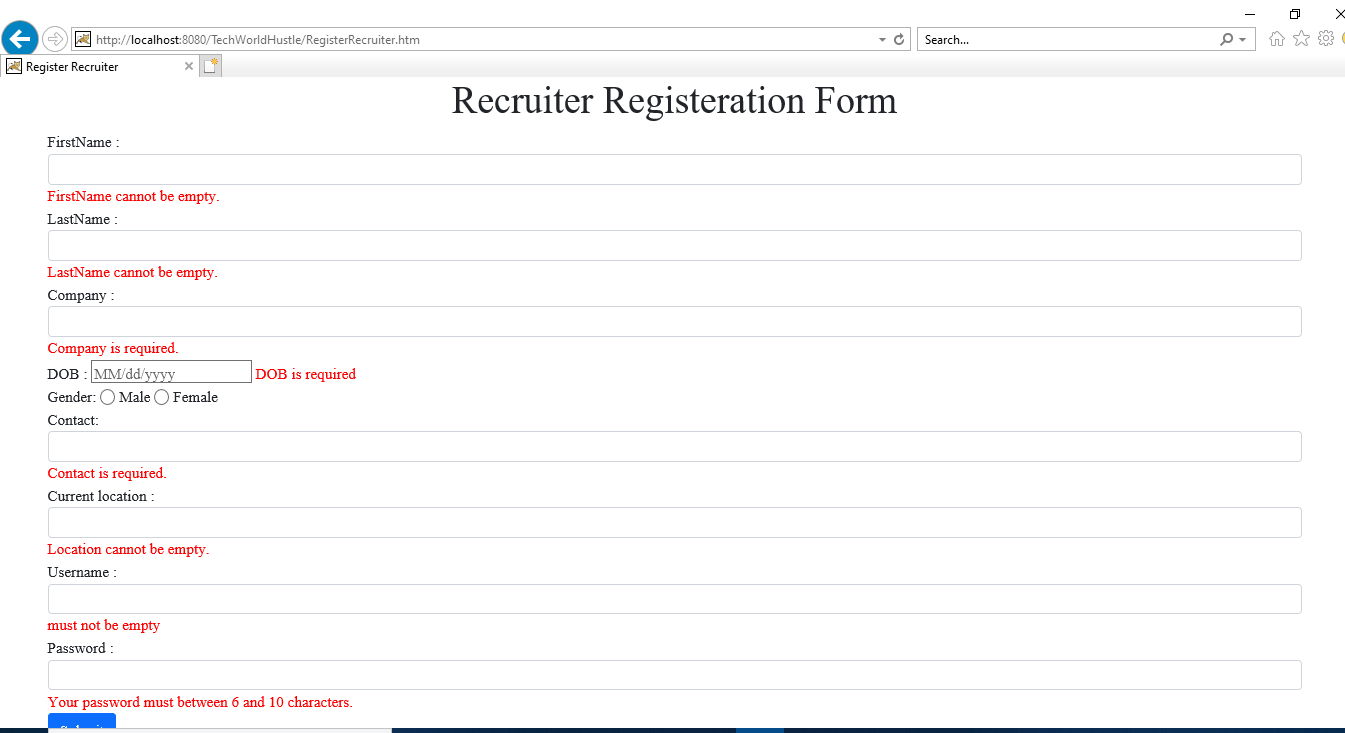


## Register Candidate

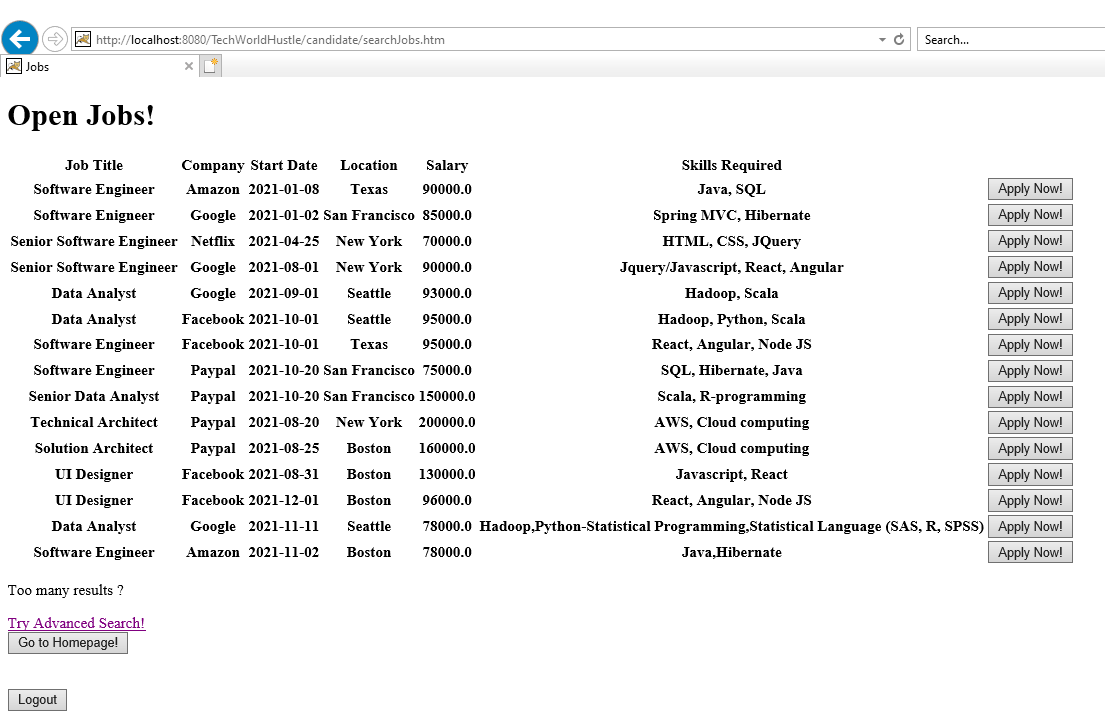




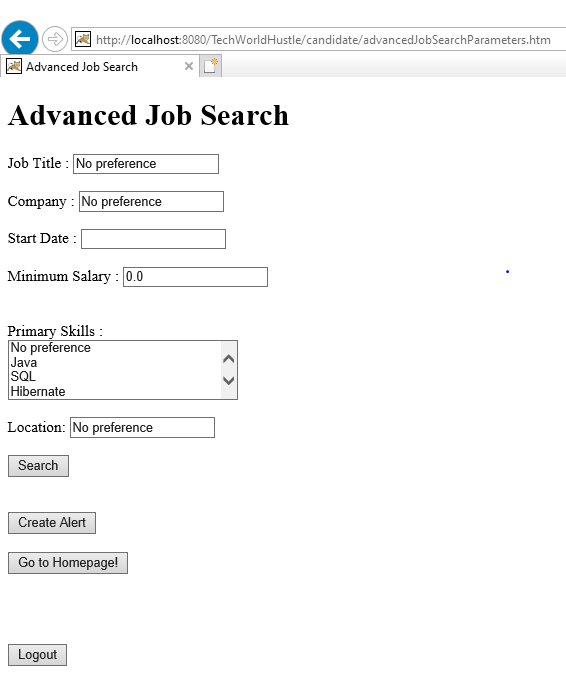
## Register Recruiter



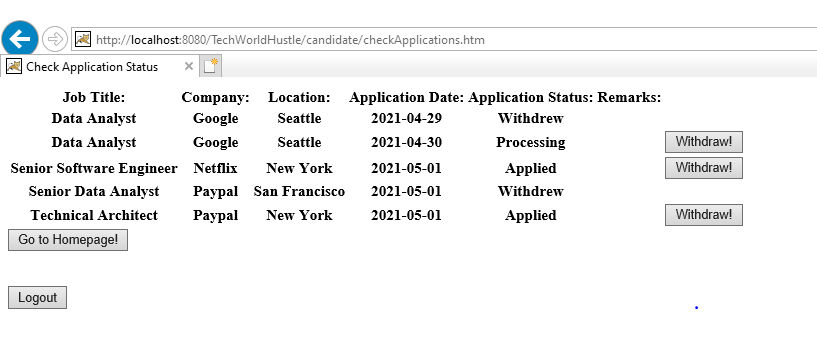
## View Open jobs



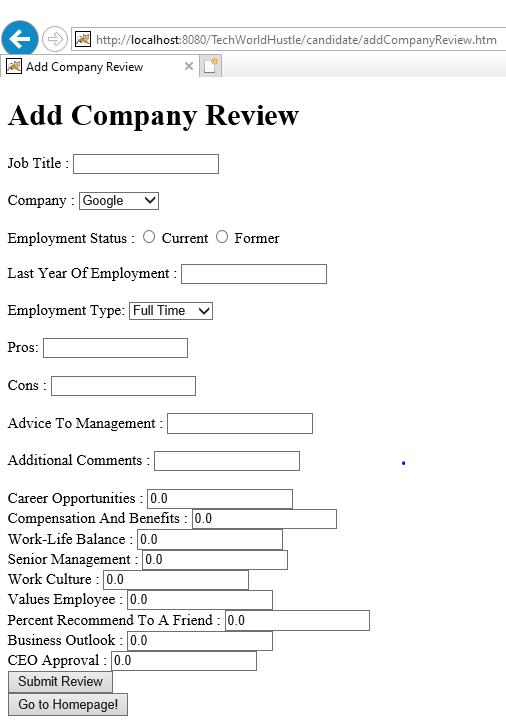
## Advanced search for jobs:



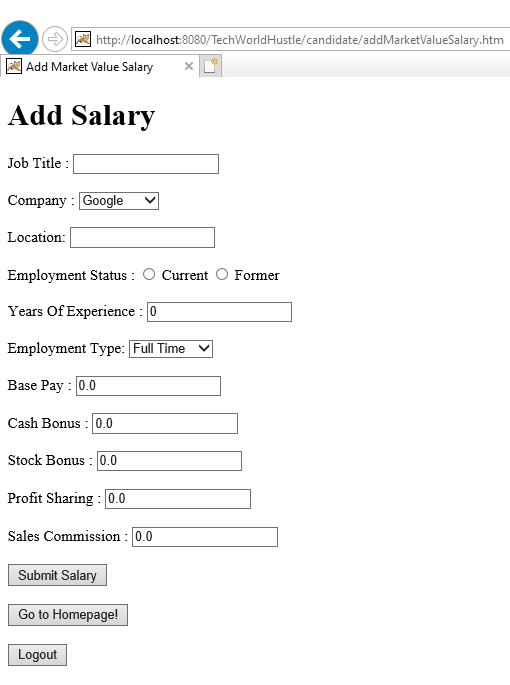
## View application status



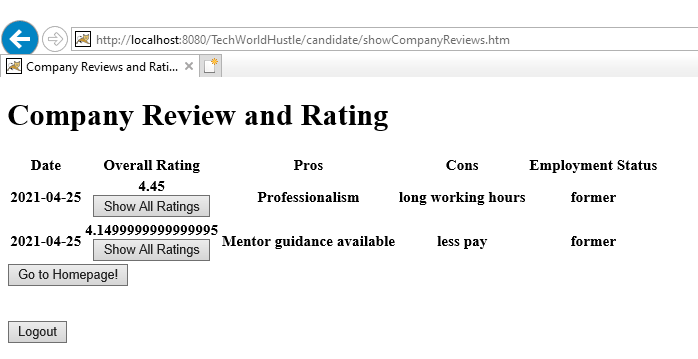
## Add Company review:

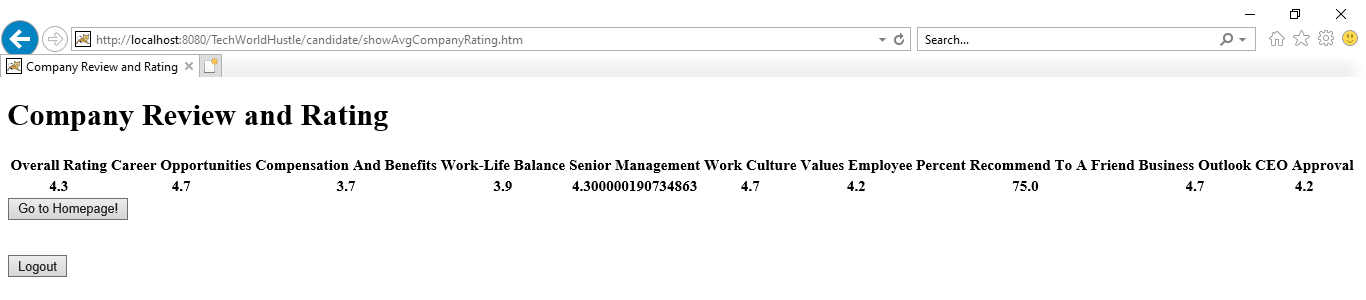


## Add salary details

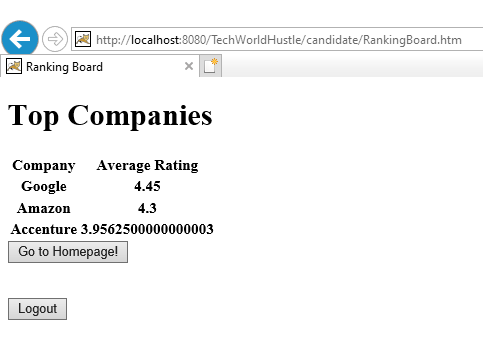


## View company Reviews

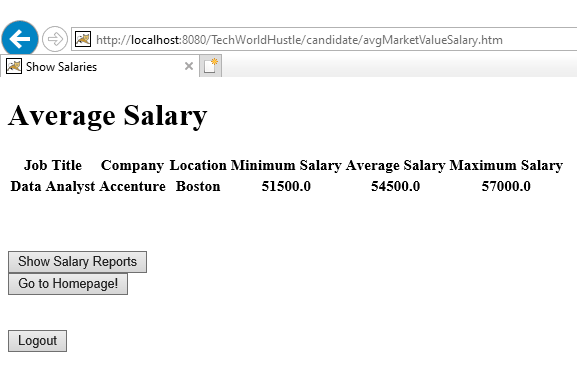


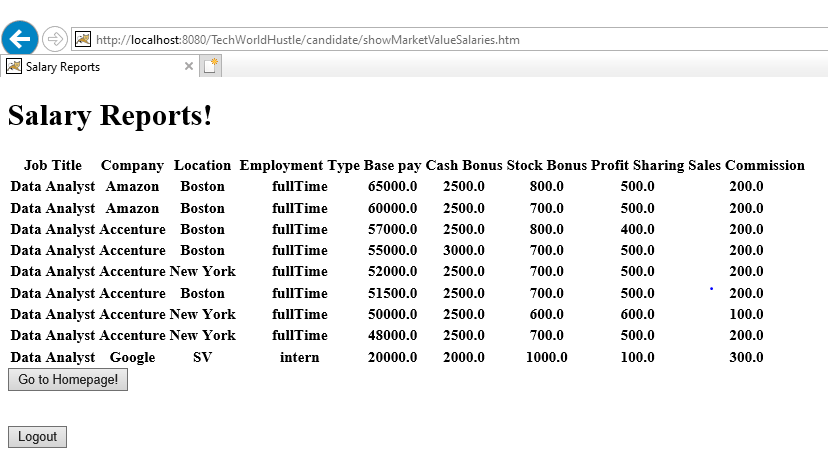


## View Ranking Board

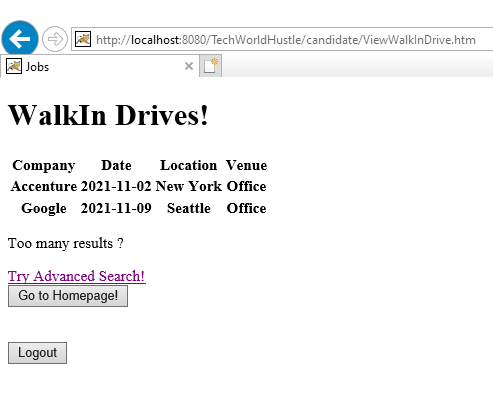


## View average salary and salary reports

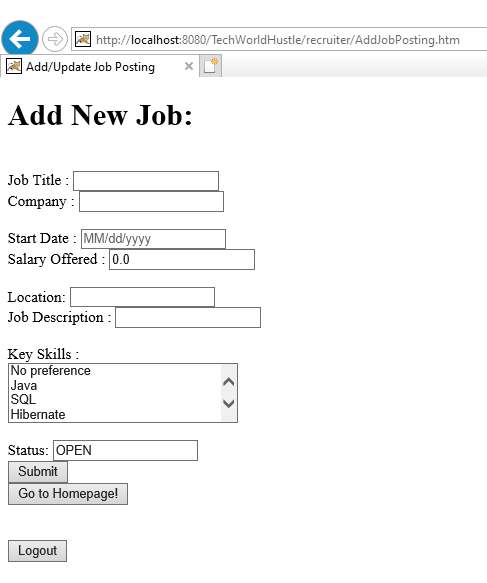




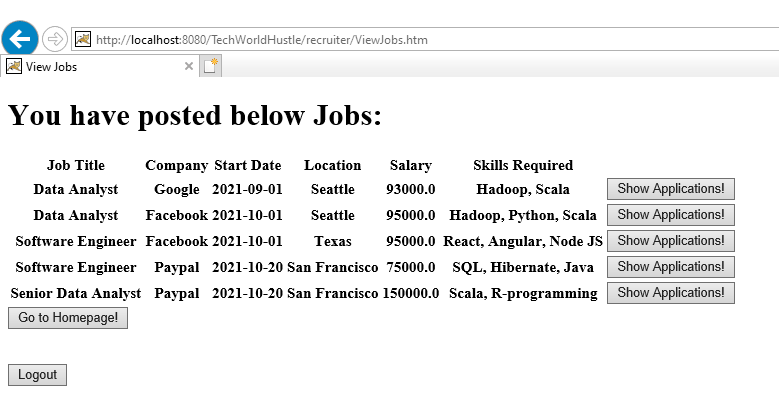
## View walk-in drives



## Add job posting



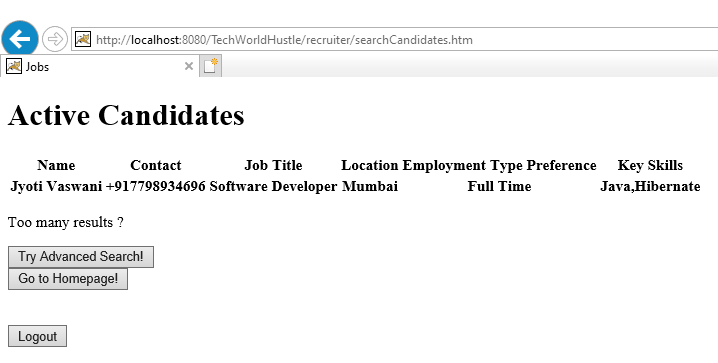
## View Jobs posted



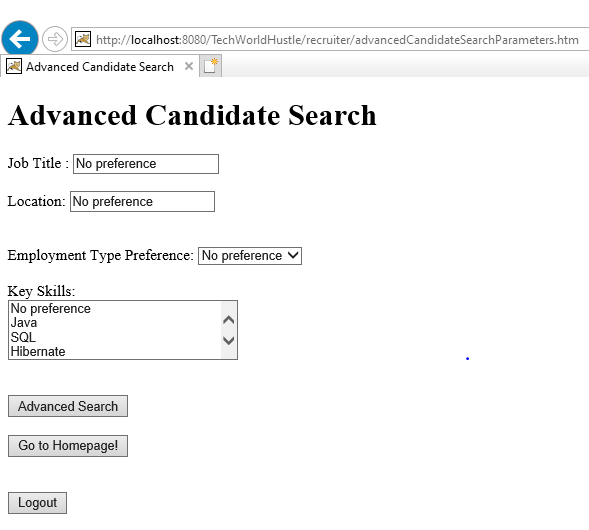
## View Applications



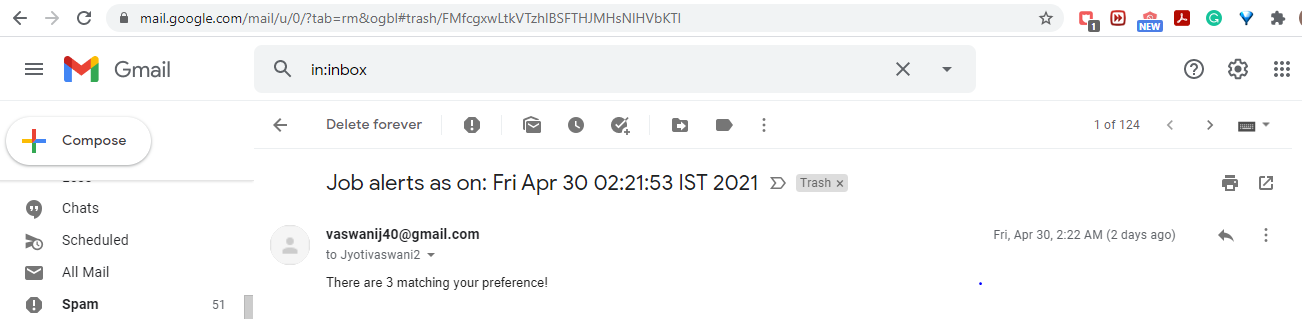
## View Candidates



## Advanced search candidates



## Email Alert



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Controller Source codes:

## 6.1 LoginController:

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.CandidateDAO;

import com.CSyE6220.TechWorldHustle.DAO.RecruiterDAO;

import com.CSyE6220.TechWorldHustle.pojo.Candidate;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.Recruiter;

import java.io.IOException;

import java.util.List;

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.stereotype.Controller;

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

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

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class LoginController {

@Autowired

CandidateDAO CDao;

@Autowired

RecruiterDAO RDao;

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

public String showLoginPage(){

return "Login";

}

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

public ModelAndView login(HttpServletRequest request, HttpServletResponse response) {

HttpSession session = request.getSession(true);

// System.out.print(session);

String username = request.getParameter("userName");

String password = request.getParameter("password"); // Sanitize

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

String pass\_check;

if (role==null)

{

session.setAttribute("login\_session", null);

return new ModelAndView("ErrorPage", "ErrorMsg","Invalid Login Role!");

}

else{

if (role.equals("Recruiter")) {

Recruiter rec = RDao.searchRecruiter(username);

if (rec!=null)

{pass\_check = rec.getPassword();

if(password.equals(pass\_check))

{

session.setAttribute("login\_session",session);

session.setAttribute("login\_user",rec.getUserId());

return new ModelAndView("recruiter/RecruiterHome");

}

else {session.setAttribute("login\_session", null);

return new ModelAndView("ErrorPage", "ErrorMsg","Invalid Username and/or Password!");}

}

else {session.setAttribute("login\_session", null);

return new ModelAndView("ErrorPage", "ErrorMsg","You have not registered!");}

} else if (role.equals("Candidate")) {

Candidate can = CDao.searchCandidate(username);

if (can!=null)

{pass\_check = can.getPassword();

if(password.equals(pass\_check))

{

session.setAttribute("login\_session", session);

session.setAttribute("login\_user",can.getUserId());

return new ModelAndView("candidate/CandidateHome");

}

else {session.setAttribute("login\_session", null);

return new ModelAndView("ErrorPage", "ErrorMsg","Invalid Username and/or Password!");}

}

else {session.setAttribute("login\_session", null);

return new ModelAndView("ErrorPage", "ErrorMsg","You have not registered!");}

} else {

session.setAttribute("login\_session", null);

return new ModelAndView("ErrorPage", "ErrorMsg","Invalid Login Role!");

}

}

}

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

public void logout(HttpServletRequest request, HttpServletResponse response) throws IOException{

HttpSession session = request.getSession(false);

session.setAttribute("login\_session", null);

session.setAttribute("login\_user",null);

response.sendRedirect("/TechWorldHustle/Welcome.jsp");

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## RegisterCandidateController

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.CandidateDAO;

import com.CSyE6220.TechWorldHustle.DAO.SkillDAO;

import com.CSyE6220.TechWorldHustle.pojo.Candidate;

import com.CSyE6220.TechWorldHustle.pojo.Skill;

import java.io.File;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import javax.validation.Valid;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

import org.springframework.validation.BindingResult;

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

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

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

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

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

import org.springframework.web.bind.support.SessionStatus;

import org.springframework.web.multipart.MultipartFile;

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

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class RegisterCandidateController {

@Autowired

SkillDAO SDao;

@Autowired

CandidateDAO CDao;

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

public ModelAndView showForm(ModelMap model, Candidate candidate){

List<Skill> skillset = SDao.getSkillList();

List<String> universal\_skillset = new ArrayList<String>();

for (Skill s : skillset){universal\_skillset.add(s.getSkillName());}

return new ModelAndView("RegisterCandidate", "UniversalSkillSet", universal\_skillset);

}

@RequestMapping(value="/RegisterCandidate.htm",method = RequestMethod.POST)

public ModelAndView save(@Valid @ModelAttribute("candidate") Candidate candidate, BindingResult result,

HttpServletRequest request, CandidateDAO CDao) throws IOException{

List<Skill> skillset = SDao.getSkillList();

List<String> universal\_skillset = new ArrayList<>();

for (Skill s : skillset){universal\_skillset.add(s.getSkillName());}

if(result.hasErrors())

{ return new ModelAndView ("RegisterCandidate", "UniversalSkillSet",universal\_skillset); }

else {

HttpSession session = request.getSession();

// System.out.println("File "+candidate.getCv\_file());

// File file = new File("C:\\CSyE6220ResumeUploads\\"+ session.getAttribute("login\_user")+"\_"+System.currentTimeMillis());

// candidate.getCv\_file().transferTo(file);

// candidate.setCv(file.getAbsolutePath());

CDao.addCandidate(candidate);

return new ModelAndView("RegistrationSuccess", "successMessage", "You have been successfully Registered!");

}

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6.3 RegisterRecruiterController

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.RecruiterDAO;

import com.CSyE6220.TechWorldHustle.pojo.Recruiter;

import javax.validation.Valid;

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

import org.springframework.validation.BindingResult;

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

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

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

import org.springframework.web.bind.support.SessionStatus;

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class RegisterRecruiterController {

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

public String showForm(ModelMap model, Recruiter recruiter){

return "RegisterRecruiter";

}

@RequestMapping(value="/RegisterRecruiter.htm",method = RequestMethod.POST)

public ModelAndView save(@Valid @ModelAttribute("recruiter") Recruiter recruiter, BindingResult result, SessionStatus status, RecruiterDAO RDao){

if(result.hasErrors())

{ return new ModelAndView ("RegisterRecruiter"); }

else {

RDao.addRecruiter(recruiter);

return new ModelAndView("RegistrationSuccess", "successMessage", "You have been successfully Registered!");

}

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6.4 CandidateActionController

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.ApplicationDAO;

import com.CSyE6220.TechWorldHustle.DAO.CandidateDAO;

import com.CSyE6220.TechWorldHustle.DAO.IndividualRatingDAO;

import com.CSyE6220.TechWorldHustle.DAO.JobDAO;

import com.CSyE6220.TechWorldHustle.DAO.RecruiterDAO;

import com.CSyE6220.TechWorldHustle.DAO.SkillDAO;

import com.CSyE6220.TechWorldHustle.DAO.JobAlertsDAO;

import com.CSyE6220.TechWorldHustle.pojo.Application;

import com.CSyE6220.TechWorldHustle.pojo.Candidate;

import com.CSyE6220.TechWorldHustle.pojo.GroupedRatingDTO;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.JobAlerts;

import com.CSyE6220.TechWorldHustle.pojo.Skill;

import com.CSyE6220.TechWorldHustle.pojo.UploadedFile;

import java.io.File;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Date;

import java.util.GregorianCalendar;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.ui.ModelMap;

import org.springframework.validation.BindingResult;

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

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

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

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

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

import org.springframework.web.multipart.MultipartFile;

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class CandidateActionController {

@Autowired

CandidateDAO CDao;

@Autowired

RecruiterDAO RDao;

@Autowired

JobDAO JDao;

@Autowired

ApplicationDAO AppDao;

@Autowired

SkillDAO SDao;

@Autowired

JobAlertsDAO JADao;

@Autowired

IndividualRatingDAO IRDao;

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

public ModelAndView fetchJobs(){

List <Job> job\_list = JDao.getOpenJobList();

return new ModelAndView("candidate/displayJobs", "JobList", job\_list);

}

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

public String goHomepage()

{return "candidate/CandidateHome";}

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

public ModelAndView advancedSearchParameters(ModelMap model, Job job){

List<Skill> skillset = SDao.getSkillList();

List<String> universal\_skillset = new ArrayList<>();

for (Skill s : skillset){universal\_skillset.add(s.getSkillName());}

job.setCompany("No preference");

job.setJobTitle("No preference");

job.setLocation("No preference");

job.setSalaryOffered(0.0);

return new ModelAndView("candidate/advancedJobSearchParameters","UniversalSkillSet",universal\_skillset);

}

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

public String uploadPage() {

System.out.println("Enter Get of Upload");

return "candidate/uploadcv";

}

@RequestMapping(value="/candidate/uploadcv.htm", method = RequestMethod.POST)

public void saveFile(HttpServletRequest request, Object o) {

UploadedFile uploadedFile = new UploadedFile();

uploadedFile.setMultipartFile((MultipartFile)o);

System.out.println("Enter Post Controller");

HttpSession session = request.getSession();

MultipartFile multipartFile = uploadedFile.getMultipartFile();

String fileName = multipartFile.getOriginalFilename();

System.out.println("Absolute path: "+fileName);

try {

File file = new File("C:\\CSyE6220ResumeUploads\\"+ session.getAttribute("login\_user")+"\_"+System.currentTimeMillis());

System.out.println("Absolute path: "+file.getAbsolutePath());

multipartFile.transferTo(file);

} catch (IOException e) {

e.printStackTrace();

}

}

@RequestMapping(value="/candidate/advancedJobSearchParameters.htm", method = RequestMethod.POST)

public ModelAndView advancedJobSearch(@ModelAttribute("job") Job job, BindingResult result,HttpServletRequest request){

if (request.getParameter("jobaction").equals("addForAlerts"))

{

HttpSession session = request.getSession();

JobAlerts job\_alert = new JobAlerts();

job\_alert.setUserId(CDao.getCandidate((int)session.getAttribute("login\_user")));

job\_alert.setJobTitle(job.getJobTitle());

job\_alert.setLocation(job.getLocation());

job\_alert.setMinSalary(job.getSalaryOffered());

job\_alert.setAlertName(session.getAttribute("login\_user")+"\_"+System.currentTimeMillis());

JADao.addJobAlerts(job\_alert);

return new ModelAndView("candidate/ActvitySuccess", "successMessage", "Job Alert has been activated!");}

else{

String location = job.getLocation();

String jobTitle = job.getJobTitle();

String company = job.getCompany();

double salaryOffered = job.getSalaryOffered();

Date startDate = job.getStartDate();

String primary\_skills = job.getPrimary\_skills();

List <Job> job\_list = JDao.filterAdvancedSearch(location, jobTitle, company, salaryOffered,startDate, primary\_skills);

return new ModelAndView("candidate/displayJobs", "JobList", job\_list);}

}

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

public ModelAndView ApplyJobs(HttpServletRequest request){

HttpSession session = request.getSession();

Application app = new Application();

app.setJobId(JDao.getJob(Integer.parseInt(request.getParameter("jid"))));

app.setRecruiterId(RDao.getRecruiter(Integer.parseInt(request.getParameter("rid"))));

app.setCandidateId(CDao.getCandidate((int)session.getAttribute("login\_user")));

app.setApplicationDate(new Date());

app.setStatus("Applied");

AppDao.addApplication(app);

return new ModelAndView("candidate/ActvitySuccess", "successMessage", "Application Submitted!");

}

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

public ModelAndView checkApplications(HttpServletRequest request){

HttpSession session = request.getSession();

List <Application> app\_list = AppDao.getCandidateApplications((int)session.getAttribute("login\_user"));

return new ModelAndView("candidate/checkApplications", "ApplicationList", app\_list);

}

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

public ModelAndView withdrawApplications(HttpServletRequest request){

Application app\_update = AppDao.getApplication(Integer.parseInt(request.getParameter("appid")));

app\_update.setStatus("Withdrew");

AppDao.updateApplication(app\_update);

return new ModelAndView("candidate/ActvitySuccess", "successMessage", "Application has been withdrawn!");

}

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

public ModelAndView showRankingBoard(HttpServletRequest request){

List<GroupedRatingDTO> results = IRDao.getTopCompanyIndividualRating();

return new ModelAndView("candidate/RankingBoard", "RatingRanks", results);

}

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

public ModelAndView updateCandidateProfile(ModelMap model, Candidate candidate, HttpServletRequest request){

HttpSession session = request.getSession();

List<Skill> skillset = SDao.getSkillList();

List<String> universal\_skillset = new ArrayList<>();

for (Skill s : skillset){universal\_skillset.add(s.getSkillName());}

Candidate c = CDao.getCandidate((int)session.getAttribute("login\_user"));

candidate.setFirstName(c.getFirstName());

candidate.setLastName(c.getLastName());

candidate.setGender(c.getGender());

candidate.setUserName(c.getUserName());

candidate.setDob(c.getDob());

candidate.setContact(c.getContact());

candidate.setEmploymentTypePreference(c.getEmploymentTypePreference());

candidate.setCurrent\_Company(c.getCurrent\_Company());

candidate.setCurrent\_JobTitle(c.getCurrent\_JobTitle());

candidate.setCurrent\_location(c.getCurrent\_location());

candidate.setPrimary\_skills(c.getPrimary\_skills());

candidate.setPassword(c.getPassword());

return new ModelAndView("RegisterCandidate","UniversalSkillSet",universal\_skillset);

}

@RequestMapping(value="/candidate/updateProfile.htm",method = RequestMethod.POST)

public ModelAndView updateCandidateProfile(@ModelAttribute("candidate") Candidate candidate, BindingResult result, HttpServletRequest request, CandidateDAO CDao){

Candidate updateCandidate = new Candidate();

updateCandidate.setPrimary\_skills(candidate.getPrimary\_skills());

updateCandidate.setContact(candidate.getContact());

updateCandidate.setCurrent\_Company(candidate.getCurrent\_Company());

updateCandidate.setCurrent\_JobTitle(candidate.getCurrent\_JobTitle());

updateCandidate.setEmploymentTypePreference(candidate.getEmploymentTypePreference());

updateCandidate.setCurrent\_location(candidate.getCurrent\_location());

updateCandidate.setCv(request.getParameter("cv"));

updateCandidate.setPassword(candidate.getPassword());

CDao.updateCandidate(updateCandidate);

return new ModelAndView("RegistrationSuccess", "successMessage", "Your profile has been updated");

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6.5 RecruiterActionController:

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.ApplicationDAO;

import com.CSyE6220.TechWorldHustle.DAO.CandidateDAO;

import com.CSyE6220.TechWorldHustle.DAO.JobDAO;

import com.CSyE6220.TechWorldHustle.DAO.RecruiterDAO;

import com.CSyE6220.TechWorldHustle.DAO.SkillDAO;

import com.CSyE6220.TechWorldHustle.pojo.Application;

import com.CSyE6220.TechWorldHustle.pojo.Candidate;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.JobAlerts;

import com.CSyE6220.TechWorldHustle.pojo.Recruiter;

import com.CSyE6220.TechWorldHustle.pojo.Skill;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.HashMap;

import java.util.HashSet;

import java.util.List;

import java.util.ListIterator;

import java.util.Map;

import java.util.Set;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import javax.validation.Valid;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

import org.springframework.validation.BindingResult;

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

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

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

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class RecruiterActionController {

@Autowired

CandidateDAO CDao;

@Autowired

RecruiterDAO RDao;

@Autowired

JobDAO JDao;

@Autowired

ApplicationDAO AppDao;

@Autowired

SkillDAO SDao;

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

public ModelAndView showForm(ModelMap model, Job job, HttpServletRequest request){

List<Skill> skillset = SDao.getSkillList();

request.setAttribute("UniversalSkillSet",skillset);

job.setStatus("OPEN");

return new ModelAndView("recruiter/AddUpdateJobPosting");

}

@RequestMapping(value="/recruiter/AddJobPosting.htm", method = RequestMethod.POST)

public ModelAndView save(@Valid @ModelAttribute("job") Job job, BindingResult result,HttpServletRequest request){

List<Skill> skillset = SDao.getSkillList();

request.setAttribute("UniversalSkillSet",skillset);

if(result.hasErrors())

{ return new ModelAndView ("/recruiter/AddUpdateJobPosting"); }

else

{

HttpSession session = request.getSession();

job.setRecId(RDao.getRecruiter((int)session.getAttribute("login\_user")));

JDao.addJob(job);

return new ModelAndView("recruiter/JobActivitySuccess", "successMessage", "Job has been posted successfully!");

}

}

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

public ModelAndView fetchJobs(HttpServletRequest request){

HttpSession session = request.getSession();

List <Job> job\_list = JDao.filterRecruiter((int)session.getAttribute("login\_user"));

return new ModelAndView("recruiter/ViewJobs", "JobPosted", job\_list);

}

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

public ModelAndView fetchApplications(HttpServletRequest request){

List <Application> app\_list = AppDao.filterJob(Integer.parseInt(request.getParameter("jid")));

return new ModelAndView("recruiter/ViewApplications", "ApplicationsReceived", app\_list);

}

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

public ModelAndView updateApplication(HttpServletRequest request){

Application app\_update = AppDao.getApplication(Integer.parseInt(request.getParameter("appid")));

app\_update.setStatus(request.getParameter("sts"));

app\_update.setRemarks(request.getParameter("remarks"));

AppDao.updateApplication(app\_update);

return new ModelAndView("recruiter/JobActivitySuccess", "successMessage", "Application has been updated!");

}

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

public ModelAndView fetchJobs(){

List <Candidate> can\_list = CDao.getCandidateList();

return new ModelAndView("recruiter/displayCandidates", "CandidateList", can\_list);

}

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

public String goHomepage()

{return "recruiter/RecruiterHome";}

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

public ModelAndView advancedCandidateSearchParameters(ModelMap model, Candidate candidate){

List<Skill> skillset = SDao.getSkillList();

List<String> universal\_skillset = new ArrayList<>();

for (Skill s : skillset){universal\_skillset.add(s.getSkillName());}

candidate.setCurrent\_location("No preference");

candidate.setEmploymentTypePreference("No preference");

candidate.setCurrent\_JobTitle("No preference");

return new ModelAndView("recruiter/advancedCandidateSearchParameters","UniversalSkillSet",universal\_skillset);

}

@RequestMapping(value="/recruiter/advancedCandidateSearchParameters.htm", method = RequestMethod.POST)

public ModelAndView advancedCandidateSearch(@ModelAttribute("candidate") Candidate candidate, BindingResult result,HttpServletRequest request){

HttpSession session = request.getSession();

String location = candidate.getCurrent\_location();

String jobTitle = candidate.getCurrent\_JobTitle();

String employmentTypePreference = candidate.getEmploymentTypePreference();

String primary\_skills = candidate.getPrimary\_skills();

List<Candidate> suitableCan = CDao.filterAdvancedSearch(location, jobTitle, primary\_skills, employmentTypePreference);

return new ModelAndView("recruiter/displayCandidates", "CandidateList", suitableCan);

}

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

public ModelAndView updateJobPosting(ModelMap model, Job job, HttpServletRequest request){

List<Skill> skillset = SDao.getSkillList();

List<String> universal\_skillset = new ArrayList<>();

for (Skill s : skillset){universal\_skillset.add(s.getSkillName());}

Job j = JDao.getJob(Integer.parseInt(request.getParameter("jid")));

job.setJobTitle(j.getJobTitle());

job.setCompany(j.getCompany());

job.setLocation(j.getLocation());

job.setJobDescription(j.getJobDescription());

job.setPrimary\_skills(j.getPrimary\_skills());

job.setSalaryOffered(j.getSalaryOffered());

job.setStartDate(j.getStartDate());

job.setStatus("OPEN");

return new ModelAndView("recruiter/AddUpdateJobPosting", "UniversalSkillSet",universal\_skillset );

}

@RequestMapping(value="/recruiter/updateJobPosting.htm", method = RequestMethod.POST)

public ModelAndView updateJobPosting(@ModelAttribute("job") Job job, BindingResult result,HttpServletRequest request){

HttpSession session = request.getSession();

Job updatejob = new Job();

updatejob.setRecId(RDao.getRecruiter((int)session.getAttribute("login\_user")));

updatejob.setJobDescription(job.getJobDescription());

updatejob.setLocation(job.getLocation());

updatejob.setPrimary\_skills(job.getPrimary\_skills());

updatejob.setStartDate(job.getStartDate());

updatejob.setSalaryOffered(job.getSalaryOffered());

JDao.updateJob(updatejob);

return new ModelAndView("recruiter/JobActivitySuccess", "successMessage", "Job has been updated successfully!");

}

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

public ModelAndView updateRecruiterProfile(ModelMap model, Recruiter recruiter, HttpServletRequest request){

HttpSession session = request.getSession();

Recruiter r = RDao.getRecruiter((int)session.getAttribute("login\_user"));

recruiter.setFirstName(r.getFirstName());

recruiter.setLastName(r.getLastName());

recruiter.setGender(r.getGender());

recruiter.setUserName(r.getUserName());

recruiter.setDob(r.getDob());

recruiter.setContact(r.getContact());

recruiter.setCompany(r.getCompany());

recruiter.setCurrent\_location(r.getCurrent\_location());

recruiter.setPassword(r.getPassword());

return new ModelAndView("RegisterRecruiter");

}

@RequestMapping(value="/recruiter/updateProfile.htm",method = RequestMethod.POST)

public ModelAndView updateRecruiterProfile(@ModelAttribute("recruiter") Recruiter recruiter, BindingResult result, HttpServletRequest request){

Recruiter updateRecruiter = new Recruiter();

updateRecruiter.setContact(recruiter.getContact());

updateRecruiter.setCompany(recruiter.getCompany());

updateRecruiter.setCurrent\_location(recruiter.getCurrent\_location());

updateRecruiter.setPassword(recruiter.getPassword());

RDao.updateRecruiter(updateRecruiter);

return new ModelAndView("RegistrationSuccess", "successMessage", "Your profile has been updated");

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6.6 WalkInDriveController

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.RecruiterDAO;

import com.CSyE6220.TechWorldHustle.DAO.SkillDAO;

import com.CSyE6220.TechWorldHustle.DAO.WalkInDriveDAO;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.Skill;

import com.CSyE6220.TechWorldHustle.pojo.WalkInDrive;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import javax.validation.Valid;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.ModelMap;

import org.springframework.validation.BindingResult;

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

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

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

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class WalkInDriveController {

@Autowired

WalkInDriveDAO WDDao;

@Autowired

RecruiterDAO RDao;

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

public String showForm(ModelMap model, WalkInDrive walkInDrive , HttpServletRequest request){

return "recruiter/AddWalkInDrive";

}

@RequestMapping(value="/recruiter/AddWalkInDrive.htm", method = RequestMethod.POST)

public ModelAndView save(@Valid @ModelAttribute("walkInDrive") WalkInDrive walkInDrive, BindingResult result,HttpServletRequest request){

HttpSession session = request.getSession();

walkInDrive.setRecruiterId(RDao.getRecruiter((int)session.getAttribute("login\_user")));

WDDao.addWalkInDrive(walkInDrive);

return new ModelAndView("recruiter/JobActivitySuccess", "successMessage", "WalkIn-Drive been posted successfully!");

}

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

public ModelAndView fetchWalkInDrive(HttpServletRequest request){

HttpSession session = request.getSession();

List <WalkInDrive> walkInDrive\_list = WDDao.getWalkInDriveList();

return new ModelAndView("candidate/displayWalkInDrive", "WalkInDrivePosted", walkInDrive\_list);

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6.7 ReviewController

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.CompanyDAO;

import com.CSyE6220.TechWorldHustle.DAO.CompanyReviewDAO;

import com.CSyE6220.TechWorldHustle.DAO.IndividualRatingDAO;

import com.CSyE6220.TechWorldHustle.pojo.Company;

import com.CSyE6220.TechWorldHustle.pojo.CompanyReview;

import com.CSyE6220.TechWorldHustle.pojo.IndividualRating;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

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.ModelMap;

import org.springframework.validation.BindingResult;

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

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

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

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class ReviewController {

@Autowired

CompanyDAO ComDao;

@Autowired

CompanyReviewDAO CRDao;

@Autowired

IndividualRatingDAO IRDao;

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

public ModelAndView addCompanyReviewForm(ModelMap model, CompanyReview companyReview){

List<Company> companyset = ComDao.getCompanyList();

List<String> universal\_companyset = new ArrayList<>();

for (Company c : companyset){universal\_companyset.add(c.getCompanyName());}

return new ModelAndView("candidate/addCompanyReview", "UniversalCompanySet",universal\_companyset);

}

@RequestMapping(value="/candidate/addCompanyReview.htm", method = RequestMethod.POST)

public ModelAndView addCompanyReview(@ModelAttribute("companyReview") CompanyReview companyReview, BindingResult result,

HttpServletRequest request){

companyReview.setReviewDate(new Date());

IndividualRating ir = new IndividualRating();

ir.setReviewId(companyReview.getReviewId());

double co = companyReview.getRating().getCareerOpportunities();

ir.setCareerOpportunities(co);

double cb = companyReview.getRating().getCompensationAndBenefits();

ir.setCompensationAndBenefits(cb);

double wlb = companyReview.getRating().getWorkLifeBalance();

ir.setWorkLifeBalance(wlb);

double sm = companyReview.getRating().getSeniorManagement();

ir.setSeniorManagement(sm);

double wc = companyReview.getRating().getWorkCulture();

ir.setWorkCulture(wc);

double ve = companyReview.getRating().getValuesEmployee();

ir.setValuesEmployee(ve);

double prf = companyReview.getRating().getPercentRecommendToAFriend();

ir.setPercentRecommendToAFriend(prf);

double bo = companyReview.getRating().getBusinessOutlook();

ir.setBusinessOutlook(bo);

double ca = companyReview.getRating().getCeoApproval();

ir.setCeoApproval(ca);

double overallRating = (co+cb+wlb+sm+wc+ve+bo+ca)/8;

ir.setOverallRating(overallRating);

ir.setReview(companyReview);

companyReview.setRating(ir);

CRDao.addCompanyReview(companyReview);

return new ModelAndView("candidate/ActvitySuccess", "successMessage", "Review has been added successfully!");

}

@RequestMapping(value="/candidate/showAvgCompanyRating.htm", method=RequestMethod.POST)

public ModelAndView showAverageRating(HttpServletRequest request){

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

List<IndividualRating> avg\_ratings = IRDao.getAverageIndividualRating(company);

return new ModelAndView("candidate/showCompanyRating", "Rating",avg\_ratings);

}

@RequestMapping(value="/candidate/showCompanyReviews.htm", method=RequestMethod.POST)

public ModelAndView showCompanyReviews(HttpServletRequest request){

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

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

List<CompanyReview> cmp\_reviews;

cmp\_reviews = CRDao.getCompanyReviewList(company,jobtitle);

return new ModelAndView("candidate/showCompanyReviews","CompanyReviews",cmp\_reviews);

}

@RequestMapping(value="/candidate/showDetailedCompanyRating.htm", method=RequestMethod.POST)

public ModelAndView showDetailedRating(HttpServletRequest request){

String reviewId = request.getParameter("rId");

List<IndividualRating> detailed\_ratings = IRDao.getIndividualRatingReviewId(Integer.parseInt(reviewId));

return new ModelAndView("candidate/showCompanyRating", "Rating",detailed\_ratings);

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6.8 MarketValueSalaryController:

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.CompanyDAO;

import com.CSyE6220.TechWorldHustle.DAO.MarketValueSalaryDAO;

import com.CSyE6220.TechWorldHustle.pojo.Application;

import com.CSyE6220.TechWorldHustle.pojo.Company;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.MarketValueSalary;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

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.ModelMap;

import org.springframework.validation.BindingResult;

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

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

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

import org.springframework.web.servlet.ModelAndView;

/\*\*

\*

\* @author jyoti

\*/

@Controller

public class MarketValueSalaryController {

@Autowired

CompanyDAO ComDao;

@Autowired

MarketValueSalaryDAO MVSDao;

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

public ModelAndView addSalaryForm(ModelMap model, MarketValueSalary marketValueSalary){

List<Company> companyset = ComDao.getCompanyList();

List<String> universal\_companyset = new ArrayList<>();

for (Company c : companyset){universal\_companyset.add(c.getCompanyName());}

return new ModelAndView("candidate/addMarketValueSalary", "UniversalCompanySet", universal\_companyset);

}

@RequestMapping(value="/candidate/addMarketValueSalary.htm", method = RequestMethod.POST)

public ModelAndView addSalary(@ModelAttribute("marketValueSalary") MarketValueSalary marketValueSalary, BindingResult result,HttpServletRequest request){

MVSDao.addMarketValueSalary(marketValueSalary);

return new ModelAndView("candidate/ActvitySuccess", "successMessage", "Thank you for valuable data. Your Salary review has been added!");

}

@RequestMapping(value="/candidate/avgMarketValueSalary.htm", method = RequestMethod.POST)

public ModelAndView avgMarketValueSalarySalary(HttpServletRequest request){

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

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

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

Map<String, String> mvs = MVSDao.getAverageBasePay(jobTitle,company,location);

return new ModelAndView("candidate/showAvgSalaries", "AvgSalary",mvs);

}

@RequestMapping(value="/candidate/showMarketValueSalaries.htm", method = RequestMethod.POST)

public ModelAndView listMarketValueSalary(HttpServletRequest request){

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

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

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

List<MarketValueSalary> mvs = MVSDao.getMarketValueSalaryList(jobTitle,company,location);

return new ModelAndView("candidate/showSalaryReports", "SalaryReports",mvs);

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6.9 JobAlertsSchedular

package com.CSyE6220.TechWorldHustle.Controllers;

import com.CSyE6220.TechWorldHustle.DAO.JobAlertsDAO;

import com.CSyE6220.TechWorldHustle.DAO.JobDAO;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.JobAlerts;

import java.util.Date;

import java.util.List;

import org.springframework.context.annotation.Configuration;

import org.springframework.scheduling.annotation.EnableScheduling;

import org.springframework.scheduling.annotation.Scheduled;

import org.springframework.stereotype.Controller;

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

import org.apache.commons.mail.\*;

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

/\*\*

\*

\* @author jyoti

\*/

@Configuration

@EnableScheduling

public class JobAlertsSchedular {

@Autowired

JobAlertsDAO JADao;

@Autowired

JobDAO JDao;

@Scheduled(cron="0 0 8 \* \* 6") // once a week (0 0 8 \* \* 6)

// for demo -- (cron="\*/5 \* \* \* \* ?")

public void ScheduledJobAlerts() throws EmailException {

long alerts\_count = JADao.numberOfJobAlerts();

for(int i=1;i<=alerts\_count;i++)

{

JobAlerts jobalerts = JADao.getJobAlerts(i);

List<Job> jobs = JDao.JobBasedOnAlerts(jobalerts.getJobTitle(), jobalerts.getLocation(), jobalerts.getMinSalary());

int numberofjobs = jobs.size();

for (Job j : jobs ){System.out.println(j.getJobId()+" "+j.getJobTitle()+" "+j.getLocation()+" "+j.getSalaryOffered());}

Email email = new SimpleEmail();

email.setHostName("smtp.gmail.com");

email.setSmtpPort(465);

email.setAuthenticator(new DefaultAuthenticator("vaswanij40@gmail.com", "Keepcalm2$"));

email.setSSLOnConnect(true);

email.setFrom("jobalerts@techworldhustle.com");

email.setSubject("Job alerts as on: "+ new Date());

email.setMsg("There are "+numberofjobs+" matching your preference!");

email.addTo(jobalerts.getUserId().getUserName());

email.send();

}

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# DAO source codes:

## 7.1 DAO:

package com.CSyE6220.TechWorldHustle.DAO;

/\*\*

\*

\* @author jyoti

\*/

import java.util.logging.Logger;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

public class DAO {

private static final Logger Log = Logger.getAnonymousLogger();

private static final ThreadLocal session = new ThreadLocal();

private static final SessionFactory sessionFactory = new Configuration().configure("hibernate.cfg.xml").buildSessionFactory();

public DAO() {}

public static Session getSession()

{

Session session = (Session) DAO.session.get();

if (session==null)

{

session = sessionFactory.openSession();

DAO.session.set(session);

}

return session;

}

protected void beginTransaction()

{

getSession().beginTransaction();

}

protected void commit()

{

getSession().getTransaction().commit();

}

protected void rollback()

{

getSession().getTransaction().rollback();

getSession().close(); //close

DAO.session.set(null); //remove session from ThreadLocal

}

public static void close()

{

getSession().close();

DAO.session.set(null);

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.2 ApplicationDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Application;

import java.util.List;

import org.hibernate.Criteria;

import org.hibernate.criterion.Criterion;

import org.hibernate.criterion.LogicalExpression;

import org.hibernate.criterion.Restrictions;

import org.hibernate.query.Query;

/\*\*

\*

\* @author jyoti

\*/

public class ApplicationDAO extends DAO{

public void addApplication(Application app)

{

beginTransaction();

getSession().save(app);

commit();

}

public List<Application> filterJob(int jobId)

{

Criteria crit = getSession().createCriteria(Application.class);

Criterion c\_jobId = Restrictions.eq("jobId.jobId",jobId);

Criterion sts\_applied = Restrictions.eq("status","Applied");

Criterion sts\_processing = Restrictions.eq("status","Processing");

LogicalExpression orExp = Restrictions.or(sts\_applied, sts\_processing);

crit.add(orExp);

crit.add(c\_jobId);

List<Application> results=crit.list();

return results;

}

public void updateApplication(Application app)

{

beginTransaction();

getSession().update("status", app);

getSession().update("Remarks", app);

commit();

}

public Application getApplication(int applicationId)

{

beginTransaction();

Application result = getSession().get(Application.class, applicationId);

commit();

return result;

}

public List<Application> getCandidateApplications(int candidateId)

{

beginTransaction();

String hql = "FROM Application where CandidateId=:can\_Id";

Query query = getSession().createQuery(hql);

query.setParameter("can\_Id", candidateId);

List result = query.list();

commit();

return result;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.3 CandidateDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Candidate;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import java.util.Date;

import java.util.List;

import org.hibernate.Criteria;

import org.hibernate.criterion.MatchMode;

import org.hibernate.criterion.Order;

import org.hibernate.criterion.Restrictions;

import org.hibernate.query.Query;

/\*\*

\*

\* @author jyoti

\*/

public class CandidateDAO extends DAO{

public void addCandidate(Candidate c)

{

beginTransaction();

getSession().save(c);

commit();

}

public void deleteCandidate(Candidate c)

{

beginTransaction();

getSession().delete(c);

commit();

}

public Candidate searchCandidate(String Username)

{

beginTransaction();

Criteria cr = getSession().createCriteria(Candidate.class);

cr.add(Restrictions.like("userName",Username));

Candidate c = (Candidate) cr.uniqueResult();

commit();

return c;

}

public Candidate getCandidate(int CandidateId)

{

beginTransaction();

Candidate result = getSession().get(Candidate.class, CandidateId);

commit();

return result;

}

public List<Candidate> getCandidateList()

{

beginTransaction();

String hql = "FROM Candidate";

Query query = getSession().createQuery(hql);

List result = query.list();

commit();

return result;

}

public void updateCandidate(Candidate c)

{

beginTransaction();

getSession().saveOrUpdate(c);

commit();

}

public List<Candidate> getCandidateEmploymentTypeList(String preference)

{

beginTransaction();

Criteria criteria = getSession().createCriteria(Candidate.class);

criteria.add(Restrictions.eq("employmentTypePreference",preference )); // pojo class property name

List result = criteria.list();

commit();

return result;

}

public List<Candidate> filterAdvancedSearch(String location, String jobTitle,String primary\_skills, String employmentTypePreference)

{

beginTransaction();

Criteria criteria = getSession().createCriteria(Candidate.class);

if(!location.equals("No preference"))

{criteria.add(Restrictions.ilike("location",location,MatchMode.ANYWHERE)); }

if(!jobTitle.equals("No preference"))

{criteria.add(Restrictions.ilike("jobTitle",jobTitle,MatchMode.ANYWHERE)); }

if(!primary\_skills.equals("No preference"))

{criteria.add(Restrictions.ilike("primary\_skills",primary\_skills,MatchMode.ANYWHERE)); }

if(!employmentTypePreference.equals("No preference"))

{criteria.add(Restrictions.eq("employmentTypePreference",employmentTypePreference)); }

List result = criteria.list();

commit();

System.out.println("Result: "+result);

return result;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.4 CompanyDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Company;

import com.CSyE6220.TechWorldHustle.pojo.Skill;

import java.util.List;

import org.hibernate.query.Query;

/\*\*

\*

\* @author jyoti

\*/

public class CompanyDAO extends DAO{

public List<Company> getCompanyList()

{

beginTransaction();

String hql = "FROM Company";

Query query = getSession().createQuery(hql);

List result = query.list();

commit();

return result;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.5 CompanyReviewDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Candidate;

import com.CSyE6220.TechWorldHustle.pojo.CompanyReview;

import com.CSyE6220.TechWorldHustle.pojo.IndividualRating;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import org.hibernate.Criteria;

import org.hibernate.criterion.Order;

import org.hibernate.criterion.Projections;

import org.hibernate.criterion.Restrictions;

import org.hibernate.query.Query;

import org.hibernate.sql.JoinType;

/\*\*

\*

\* @author jyoti

\*/

public class CompanyReviewDAO extends DAO {

public void addCompanyReview(CompanyReview cr)

{

beginTransaction();

getSession().save(cr);

commit();

}

public void deleteCompanyReview(CompanyReview cr)

{

beginTransaction();

getSession().delete(cr);

commit();

}

public List<CompanyReview> getCompanyReviewList(String company, String jobTitle)

{

beginTransaction();

Criteria criteria = getSession().createCriteria(CompanyReview.class, "cr");

criteria.add(Restrictions.eq("cr.company", company));

if(jobTitle.length()>0)

{criteria.add(Restrictions.eq("cr.jobTitle", jobTitle));}

criteria.addOrder(Order.asc("reviewDate"));

criteria.createAlias("cr.rating", "ir");

criteria.setProjection(Projections.projectionList()

.add( Projections.property("cr.pros"))

.add( Projections.property("cr.cons"))

.add( Projections.property("cr.employmentStatus"))

.add( Projections.property("cr.reviewDate"))

.add( Projections.property("cr.reviewId"))

.add( Projections.property("ir.overallRating"))).list();

List<Object[]> result = criteria.list();

List<CompanyReview> cr\_result = new ArrayList<>();

for (Object[] o : result){

CompanyReview cmpreview = new CompanyReview();

IndividualRating ir = new IndividualRating();

ir.setReview(cmpreview);

cmpreview.setRating(ir);

cmpreview.setPros((String)o[0]);

cmpreview.setCons((String)o[1]);

cmpreview.setEmploymentStatus((String)o[2]);

cmpreview.setReviewDate((Date)o[3]);

cmpreview.setReviewId((int)o[4]);

cmpreview.getRating().setOverallRating((double)o[5]);

cr\_result.add(cmpreview);

}

return cr\_result;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.6 IndividualRatingDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.CompanyReview;

import com.CSyE6220.TechWorldHustle.pojo.GroupedRatingDTO;

import com.CSyE6220.TechWorldHustle.pojo.IndividualRating;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import java.util.ArrayList;

import java.util.List;

import org.hibernate.Criteria;

import org.hibernate.criterion.Order;

import org.hibernate.criterion.ProjectionList;

import org.hibernate.criterion.Projections;

import org.hibernate.criterion.Restrictions;

import org.hibernate.query.Query;

import org.hibernate.transform.Transformers;

/\*\*

\*

\* @author jyoti

\*/

public class IndividualRatingDAO extends DAO {

public void addIndividualRating(IndividualRating ir)

{

beginTransaction();

getSession().save(ir);

commit();

}

public void deleteIndividualRating(IndividualRating ir)

{

beginTransaction();

getSession().delete(ir);

commit();

}

public List<IndividualRating> getIndividualRatingReviewId(int reviewId)

{

beginTransaction();

String hql = "FROM IndividualRating where ReviewId=:reviewId";

Query query = getSession().createQuery(hql);

query.setParameter("reviewId", reviewId);

List result = query.list();

commit();

return result;

}

public List<IndividualRating> getAverageIndividualRating(String company)

{

beginTransaction();

Criteria criteria = getSession().createCriteria(IndividualRating.class);

criteria.createAlias("review", "r");

criteria.add(Restrictions.eq("r.company",company));

ProjectionList projList = Projections.projectionList();

projList.add(Projections.avg("careerOpportunities"));

projList.add(Projections.avg("compensationAndBenefits"));

projList.add(Projections.avg("workLifeBalance"));

projList.add(Projections.avg("seniorManagement"));

projList.add(Projections.avg("workCulture"));

projList.add(Projections.avg("valuesEmployee"));

projList.add(Projections.avg("percentRecommendToAFriend"));

projList.add(Projections.avg("businessOutlook"));

projList.add(Projections.avg("ceoApproval"));

projList.add(Projections.avg("overallRating"));

criteria.setProjection(projList);

List<Object[]> results = criteria.list();

IndividualRating average\_rating = new IndividualRating();

average\_rating.setCareerOpportunities((double)results.get(0)[0]);

average\_rating.setCompensationAndBenefits((double)results.get(0)[1]);

average\_rating.setWorkLifeBalance((double)results.get(0)[2]);

average\_rating.setSeniorManagement((double)results.get(0)[3]);

average\_rating.setWorkCulture((double)results.get(0)[4]);

average\_rating.setValuesEmployee((double)results.get(0)[5]);

average\_rating.setPercentRecommendToAFriend((double)(results.get(0)[6]));

average\_rating.setBusinessOutlook((double)results.get(0)[7]);

average\_rating.setCeoApproval((double)results.get(0)[8]);

average\_rating.setOverallRating((double)results.get(0)[9]);

List<IndividualRating> ir = new ArrayList<>();

ir.add(average\_rating);

commit();

return ir;

}

public List<GroupedRatingDTO> getTopCompanyIndividualRating()

{

beginTransaction();

List<GroupedRatingDTO> results = getSession().createCriteria(IndividualRating.class, "ir")

.createAlias("ir.review", "r")

.setProjection(Projections.projectionList().add(Projections.groupProperty("r.company"),"company")

.add(Projections.avg("ir.overallRating"), "avgOARating"))

.addOrder(Order.desc("avgOARating"))

.setResultTransformer(Transformers.aliasToBean(GroupedRatingDTO.class))

.list();

for (GroupedRatingDTO r : results ){System.out.println(r.getCompany()+"--"+r.getAvgOARating());}

commit();

return results;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.7 JobAlertsDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.JobAlerts;

import org.hibernate.query.Query;

/\*\*

\*

\* @author jyoti

\*/

public class JobAlertsDAO extends DAO{

public void addJobAlerts(JobAlerts jobalerts)

{

beginTransaction();

getSession().save(jobalerts);

commit();

}

public void deleteJobAlerts(JobAlerts jobalerts)

{

beginTransaction();

getSession().delete(jobalerts);

commit();

}

public JobAlerts getJobAlerts(int alertId)

{

beginTransaction();

JobAlerts result = getSession().get(JobAlerts.class, alertId);

commit();

return result;

}

public Long numberOfJobAlerts()

{

beginTransaction();

String hql = "Select count(\*) FROM JobAlerts";

Query query = getSession().createQuery(hql);

System.out.println("count: "+ query);

commit();

return ((Long)query.uniqueResult()).longValue();

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.8 JobDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.Recruiter;

import com.CSyE6220.TechWorldHustle.pojo.Skill;

import java.util.Date;

import java.util.List;

import java.util.Set;

import org.hibernate.Criteria;

import org.hibernate.FetchMode;

import org.hibernate.criterion.Example;

import org.hibernate.criterion.MatchMode;

import org.hibernate.criterion.Order;

import org.hibernate.criterion.Restrictions;

import org.hibernate.query.Query;

/\*\*

\*

\* @author jyoti

\*/

public class JobDAO extends DAO{

public void addJob(Job j)

{

beginTransaction();

getSession().save(j);

commit();

}

public void deleteJob(Job j)

{

beginTransaction();

getSession().delete(j);

commit();

}

public void updateJob(Job j)

{

beginTransaction();

getSession().saveOrUpdate(j);

commit();

}

public List<Job> getJobList()

{

beginTransaction();

String hql = "FROM Job";

Query query = getSession().createQuery(hql);

List result = query.list();

commit();

return result;

}

public Job getJob(int jobId)

{

beginTransaction();

Job result = getSession().get(Job.class, jobId);

commit();

return result;

}

public List<Job> getOpenJobList()

{

beginTransaction();

Criteria criteria = getSession().createCriteria(Job.class);

criteria.add(Restrictions.eq("status", "OPEN")); // pojo class property name

List result = criteria.list();

commit();

return result;

}

public List<Job> filterRecruiter(int Rec\_Id)

{

beginTransaction();

String hql = "FROM Job where RecruiterId=:RecId";

Query query = getSession().createQuery(hql);

query.setParameter("RecId", Rec\_Id);

List result = query.list();

commit();

return result;

}

public List<Job> filterAdvancedSearch(String location, String jobTitle, String company,

double salaryOffered, Date startDate,String primary\_skills)

{

beginTransaction();

Criteria criteria = getSession().createCriteria(Job.class, "j");

criteria.add(Restrictions.eq("status", "OPEN"));

if(!location.equals("No preference"))

{criteria.add(Restrictions.ilike("location",location,MatchMode.ANYWHERE)); }

if(!jobTitle.equals("No preference"))

{criteria.add(Restrictions.ilike("jobTitle",jobTitle,MatchMode.ANYWHERE)); }

if(!company.equals("No preference"))

{criteria.add(Restrictions.ilike("company",company,MatchMode.ANYWHERE)); }

if(salaryOffered!=0.0)

{

criteria.add(Restrictions.gt("salaryOffered",salaryOffered));

criteria.addOrder(Order.desc("salaryOffered"));

}

if((startDate!=null) && (startDate.compareTo(new Date()))>=0)

{

criteria.add(Restrictions.gt("startDate",startDate));

criteria.addOrder(Order.asc("startDate"));

}

if(!primary\_skills.equals("No preference"))

{criteria.add(Restrictions.ilike("primary\_skills", primary\_skills, MatchMode.ANYWHERE));}

List result = criteria.list();

commit();

System.out.println("Result: "+result);

return result;

}

public List<Job> JobBasedOnAlerts(String jobTitle, String location, double sal)

{

beginTransaction();

Criteria crit = getSession().createCriteria(Job.class);

Job exampleJob = new Job();

exampleJob.setJobTitle(jobTitle);

exampleJob.setLocation(location);

exampleJob.setSalaryOffered(sal);

Example example = Example.create(exampleJob);

if(jobTitle.equals("No preference"))

{example.excludeProperty("jobTitle"); }

if(location.equals("No preference"))

{example.excludeProperty("location"); }

if(sal==0.0)

{example.excludeProperty("salaryOffered");}

example.excludeZeroes();

crit.add(example);

List<Job> results = crit.list();

commit();

return results;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.9 MarketValueSalaryDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.CompanyReview;

import com.CSyE6220.TechWorldHustle.pojo.IndividualRating;

import com.CSyE6220.TechWorldHustle.pojo.MarketValueSalary;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.hibernate.Criteria;

import org.hibernate.criterion.Order;

import org.hibernate.criterion.ProjectionList;

import org.hibernate.criterion.Projections;

import org.hibernate.criterion.Restrictions;

/\*\*

\*

\* @author jyoti

\*/

public class MarketValueSalaryDAO extends DAO{

public void addMarketValueSalary(MarketValueSalary mvs)

{

beginTransaction();

getSession().save(mvs);

commit();

}

public void deleteMarketValueSalary(MarketValueSalary mvs)

{

beginTransaction();

getSession().delete(mvs);

commit();

}

public List<MarketValueSalary> getMarketValueSalaryList(String jobTitle, String company, String location)

{

beginTransaction();

Criteria criteria = getSession().createCriteria(MarketValueSalary.class);

criteria.add(Restrictions.ilike("jobTitle", jobTitle));

if(company.length()>0)

{criteria.add(Restrictions.ilike("company", company));}

if(location.length()>0)

{criteria.add(Restrictions.ilike("location", location));}

criteria.addOrder(Order.desc("basePay"));

List<MarketValueSalary> result = criteria.list();

return result;

}

public Map<String, String> getAverageBasePay(String jobTitle, String company, String location)

{

beginTransaction();

Criteria criteria = getSession().createCriteria(MarketValueSalary.class);

criteria.add(Restrictions.eq("jobTitle", jobTitle));

if(company.length()>0)

{criteria.add(Restrictions.ilike("company", company));}

if(location.length()>0)

{criteria.add(Restrictions.ilike("location", location));}

ProjectionList projList = Projections.projectionList();

projList.add(Projections.min("basePay"));

projList.add(Projections.avg("basePay"));

projList.add(Projections.max("basePay"));

criteria.setProjection(projList);

List<Object[]> results = criteria.list();

Map<String, String> base\_pay\_aggregations= new HashMap<>();

String min\_basePay = results.get(0)[0].toString();

String avg\_basePay = results.get(0)[1].toString();

String max\_basePay = results.get(0)[2].toString();

base\_pay\_aggregations.put("min\_basePay",min\_basePay);

base\_pay\_aggregations.put("avg\_basePay",avg\_basePay);

base\_pay\_aggregations.put("max\_basePay",max\_basePay);

base\_pay\_aggregations.put("jobTitle",jobTitle);

base\_pay\_aggregations.put("company",company);

base\_pay\_aggregations.put("location",location);

return base\_pay\_aggregations;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.10 RecruiterDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Candidate;

import com.CSyE6220.TechWorldHustle.pojo.Recruiter;

import org.hibernate.Criteria;

import org.hibernate.criterion.Restrictions;

/\*\*

\*

\* @author jyoti

\*/

public class RecruiterDAO extends DAO{

public void addRecruiter(Recruiter r)

{

beginTransaction();

getSession().save(r);

commit();

}

public void deleteRecruiter(Recruiter r)

{

beginTransaction();

getSession().delete(r);

commit();

}

public Recruiter getRecruiter(int RecruiterId)

{

beginTransaction();

Recruiter result = getSession().get(Recruiter.class, RecruiterId);

commit();

return result;

}

public Recruiter searchRecruiter(String Username)

{

beginTransaction();

Criteria cr = getSession().createCriteria(Recruiter.class);

cr.add(Restrictions.like("userName",Username));

Recruiter r = (Recruiter) cr.uniqueResult();

commit();

return r;

}

public void updateRecruiter(Recruiter r)

{

beginTransaction();

getSession().saveOrUpdate(r);

commit();

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.11 SkillDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.Skill;

import java.util.HashMap;

import java.util.List;

import java.util.ListIterator;

import java.util.Map;

import org.hibernate.query.Query;

/\*\*

\*

\* @author jyoti

\*/

public class SkillDAO extends DAO{

public List<Skill> getSkillList()

{

beginTransaction();

String hql = "FROM Skill";

Query query = getSession().createQuery(hql);

List result = query.list();

commit();

return result;

}

public Skill getSkillFromName(String name)

{

beginTransaction();

String hql = "FROM Skill where SkillName=:name";

Query query = getSession().createQuery(hql);

query.setParameter("name", name);

Skill s = (Skill) query.uniqueResult();

commit();

return s;

}

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## 7.12 WalkInDriveDAO

package com.CSyE6220.TechWorldHustle.DAO;

import static com.CSyE6220.TechWorldHustle.DAO.DAO.getSession;

import com.CSyE6220.TechWorldHustle.pojo.Job;

import com.CSyE6220.TechWorldHustle.pojo.WalkInDrive;

import java.util.Date;

import java.util.List;

import org.hibernate.Criteria;

import org.hibernate.criterion.Order;

import org.hibernate.criterion.Restrictions;

/\*\*

\*

\* @author jyoti

\*/

public class WalkInDriveDAO extends DAO{

public void addWalkInDrive(WalkInDrive w)

{

beginTransaction();

getSession().save(w);

commit();

}

public void deleteWalkInDrive(WalkInDrive w)

{

beginTransaction();

getSession().delete(w);

commit();

}

public void updateWalkInDrive(WalkInDrive w)

{

beginTransaction();

getSession().saveOrUpdate(w);

commit();

}

public List<WalkInDrive> getWalkInDriveList()

{

beginTransaction();

Criteria criteria = getSession().createCriteria(WalkInDrive.class);

criteria.add(Restrictions.gt("driveDate", new Date()));

criteria.addOrder(Order.asc("driveDate")); // pojo class property name

List result = criteria.list();

commit();

return result;

}

}

# LoginInterceptor Source code:

package com.CSyE6220.TechWorldHustle.Interceptors;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import org.springframework.web.servlet.handler.HandlerInterceptorAdapter;

/\*\*

\*

\* @author jyoti

\*/

public class LoginInterceptor extends HandlerInterceptorAdapter{

@Override

public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler)throws Exception

{

HttpSession session = request.getSession(false);

if (session== null)

{

response.sendRedirect("/TechWorldHustle/Welcome.jsp");

return false;}

else

{

if (session.getAttribute("login\_session").equals(session))

{return true;}

else

{

response.sendRedirect("/TechWorldHustle/Welcome.jsp");

return false;}

}

}

}

Thank You.