Final Project - STEMer

STEM fields job seeking website

Xingtong Dong

2020/4/20

INFO6250 Web dev tools

SUMMARY

The STEAMer website Provides employment services for job seekers in STEM fields. Workers and employers could create profiles, searching for companies, jobs and people that they are interested in, and apply for jobs. Employers can view applications and send feedback to applicants.

FUNCTIONALITY

1. User registration, login and logout.
2. Searching for companies, positions and users.
3. Modifying profile, position and company information.
4. Job applying and application feedback.

TECHNOLOGIES

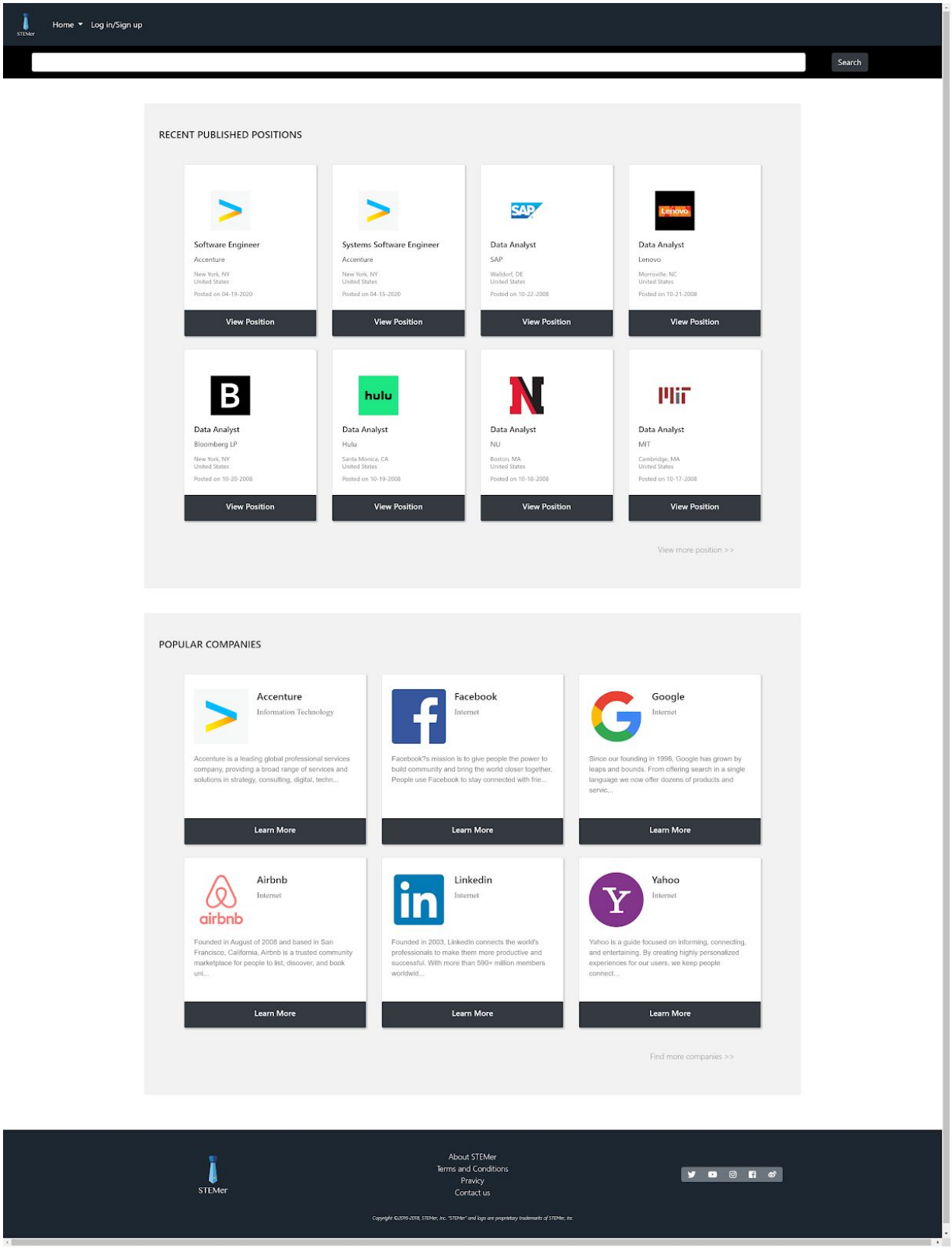
1. Spring MVC framework 4.3.0 (using annotations) and Maven
2. Hibernate 4.3.6 (using annotations)
    a. Criteria
    b. Table per subclass hibernate mapping
    c. One-to-one, one-to-many and many-to-one mapping
3. MySQL57
4. Generic DAO interface and service interface
5. Commons-fileupload 1.4 (file uploading)
6. Commons-email1.5 (email sending)
7. Log4j 1.2.17 (logging)
8. Spring-security-crypto 3.1.0 (password encryption)
9. Plain JSP and JSP Page Directives
10. JSTL, expression language and form tags
11. JQuery (datatables, date pickers, frontend validations and other dynamic panels)
12. AJAX to send asynchronous requests
13. Interceptors to check the authentication and permission of users
14. Validators - backend validations.
15. Session tracking
16. Git version control

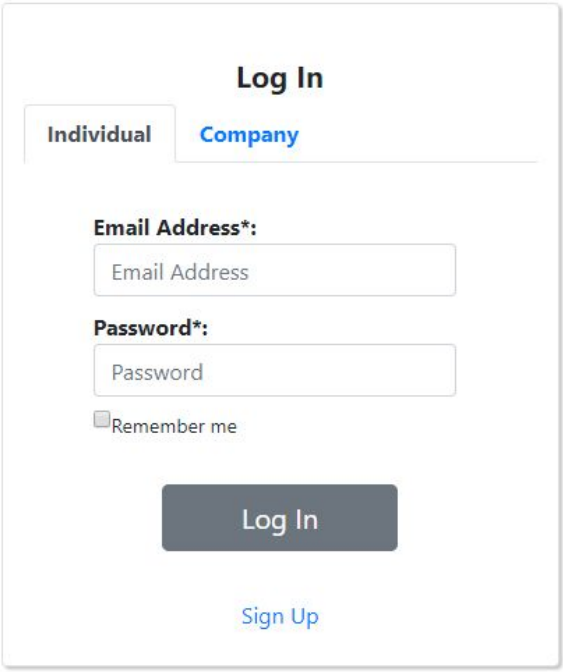
17. Cookies management

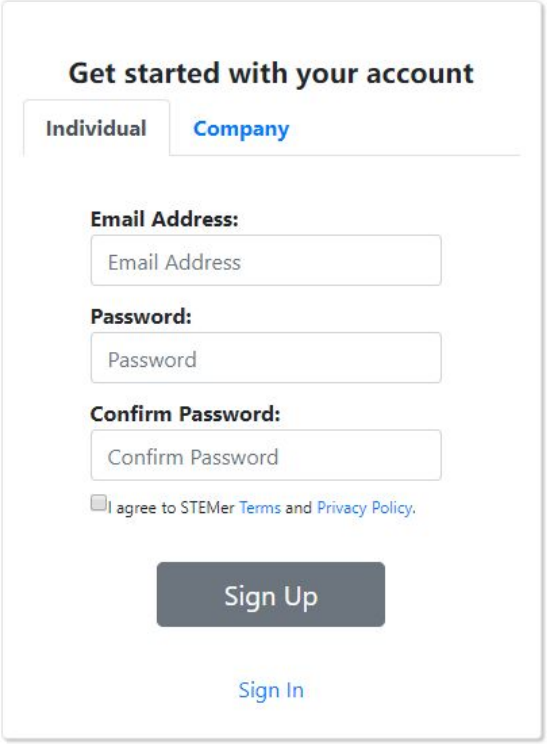
USER ROLES AND TASKS

ROLES	TASKS
Job Seekers	1. Sign up/Login/Logout 2. Auto login with cookies 3. Create and maintain profiles including their basic information, work experience, education background, training skills, and a photo 4. Search and view posted positions and see detailed information 5. Search and view companies and see detailed information 6. Search and view people and see detailed profile 7. Apply for jobs and upload resume 8. Track application status and view feedback 9. Download applications table
Employers	1. Sign up/Login/Logout 2. Auto login with cookies 3. Create and maintain company information, uploading company logo 4. Post positions with basic information and requirements 5. Edit and manage posted positions 6. Download positions table 7. View and download applications table 8. View applicants information and download resumes 9. Feedback applicant with a decision, scheduling an interview or rejecting the application 10. Send emails to applicants 11. Search and view posted positions and see detailed information 12. Search and view companies and see detailed information 13. Search and view people and see detailed profile

SCREENSHOTS OF KEY SCREEN

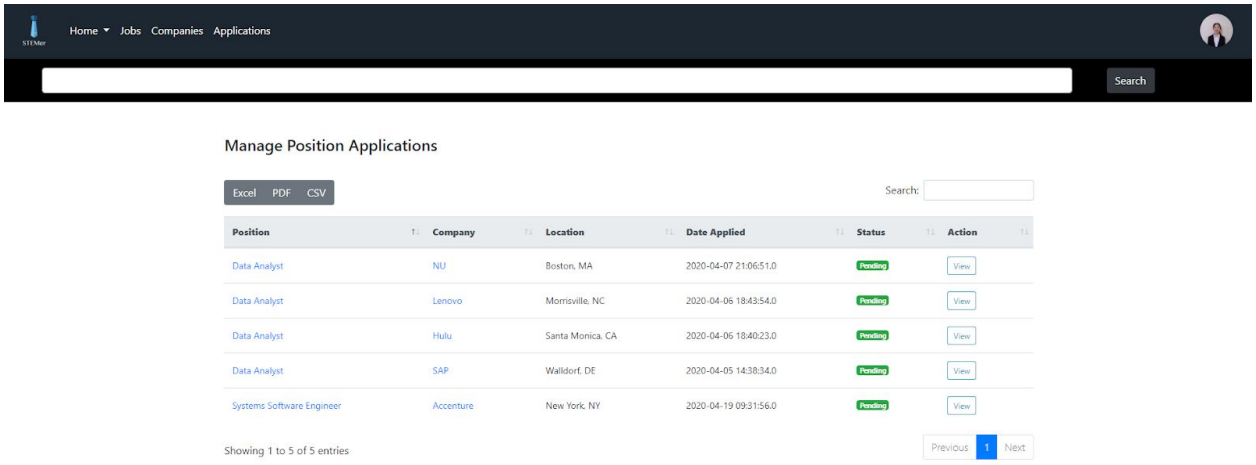

STEMer
Home
Log in/Sign up
Search
RECENT PUBLISHED POSITIONS
Software Engineer
Accenture
New York, NY
United States
Posted on 04-19-2020
View Position
Systems Software Engineer
Accenture
New York, NY
United States
Posted on 04-15-2020
View Position
Data Analyst
SAP
Walldorf, DE
United States
Posted on 10-22-2008
View Position
Data Analyst
Lenovo
Morrisville, NC
United States
Posted on 10-21-2008
View Position
Data Analyst
Bloomberg LP
New York, NY
United States
Posted on 10-20-2008
View Position
Data Analyst
Hulu
Santa Monica, CA
United States
Posted on 10-19-2008
View Position
Data Analyst
NU
Boston, MA
United States
Posted on 10-18-2008
View Position
Data Analyst
MIT
Cambridge, MA
United States
Posted on 10-17-2008
View Position
View more position >>
POPULAR COMPANIES
Accenture
Information Technology
Accenture is a leading global professional services company, providing a broad range of services and solutions in strategy, consulting, digital, techn...
Learn More
Facebook
Internet
Facebook?s mission is to give people the power to build community and bring the world closer together. People use Facebook to stay connected with frie...
Learn More
Google
Internet
Since our founding in 1998, Google has grown by leaps and bounds. From offering search in a single language we now offer dozens of products and servic...
Learn More
Airbnb
Internet
Founded in August of 2008 and based in San Francisco, California, Airbnb is a trusted community marketplace for people to list, discover, and book uni...
Learn More
Linkedin
Internet
Founded in 2003, LinkedIn connects the world's professionals to make them more productive and successful. With more than 590+ million members worldwid...
Learn More
Yahoo
Internet
Yahoo is a guide focused on informing, connecting, and entertaining. By creating highly personalized experiences for our users, we keep people connect...
Learn More
Find more companies >>
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc

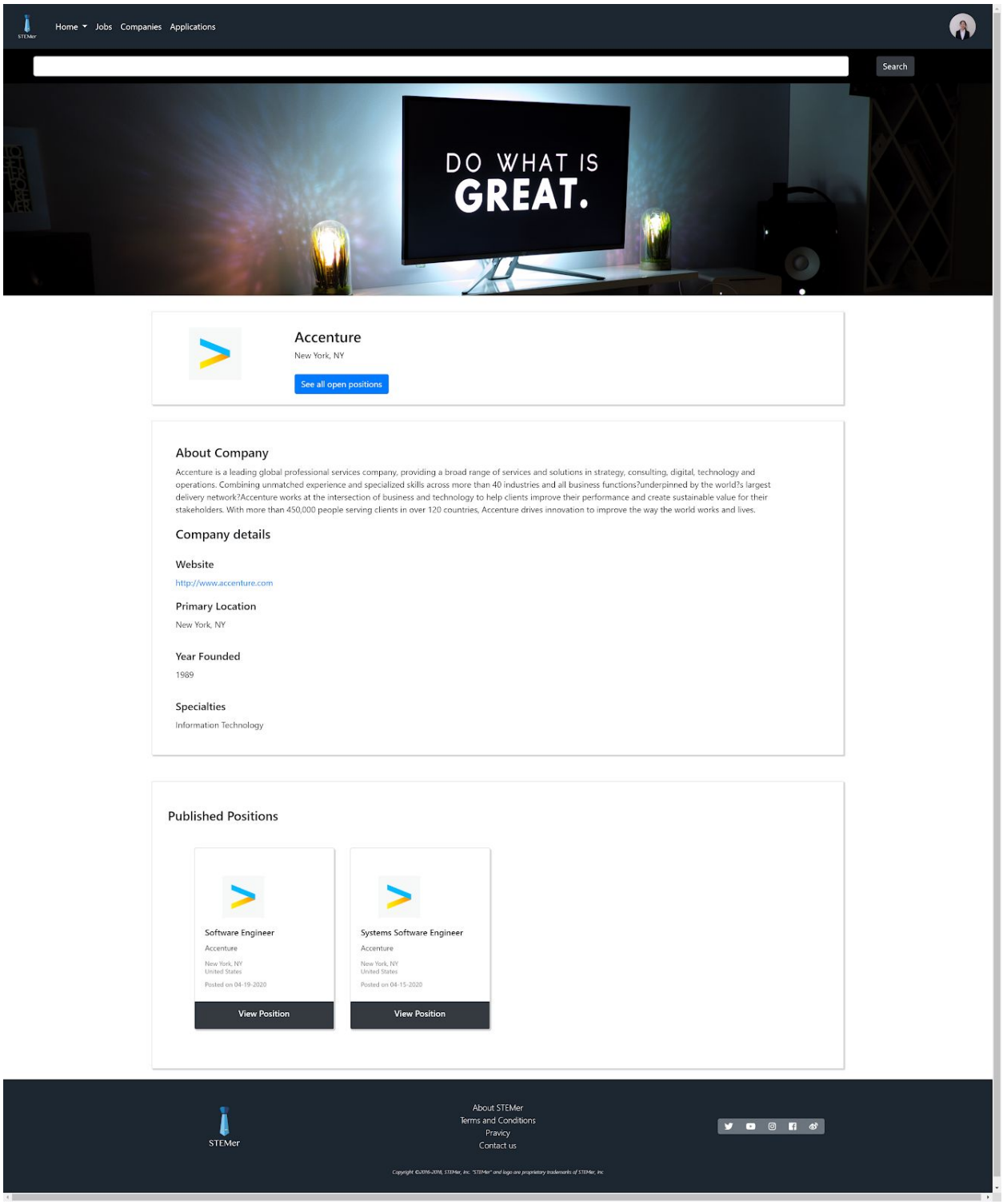


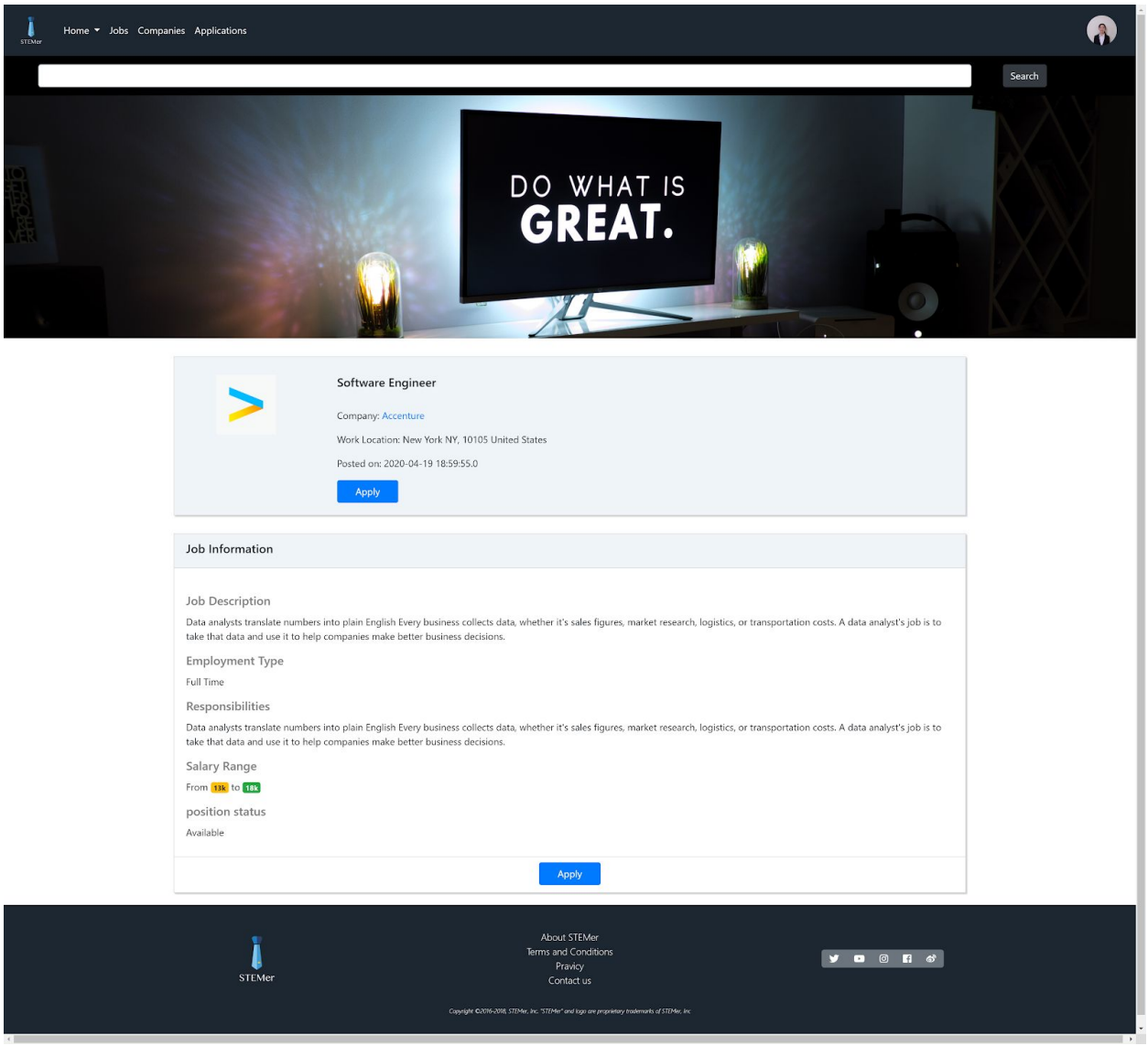


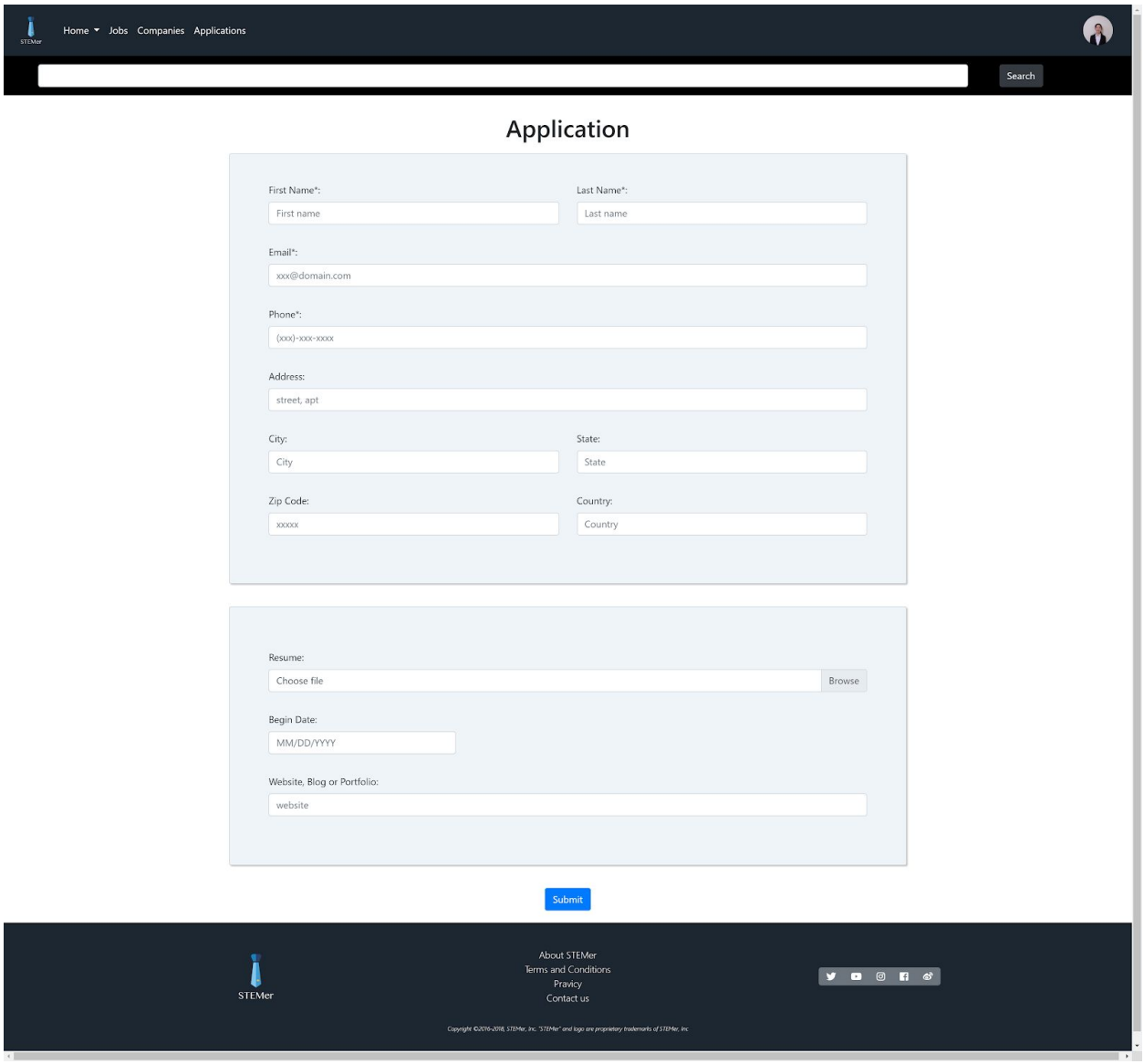
User side:

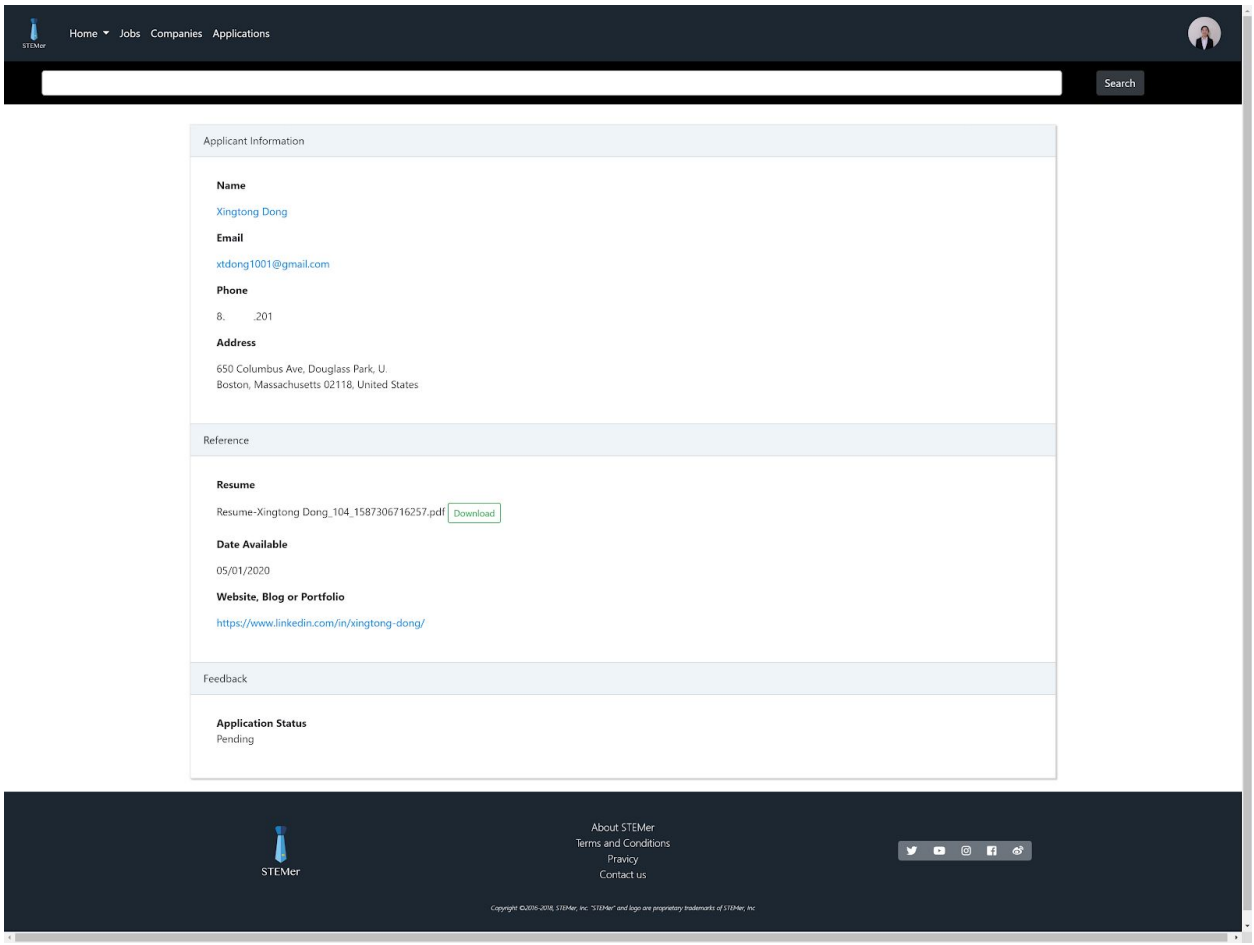
Applications page

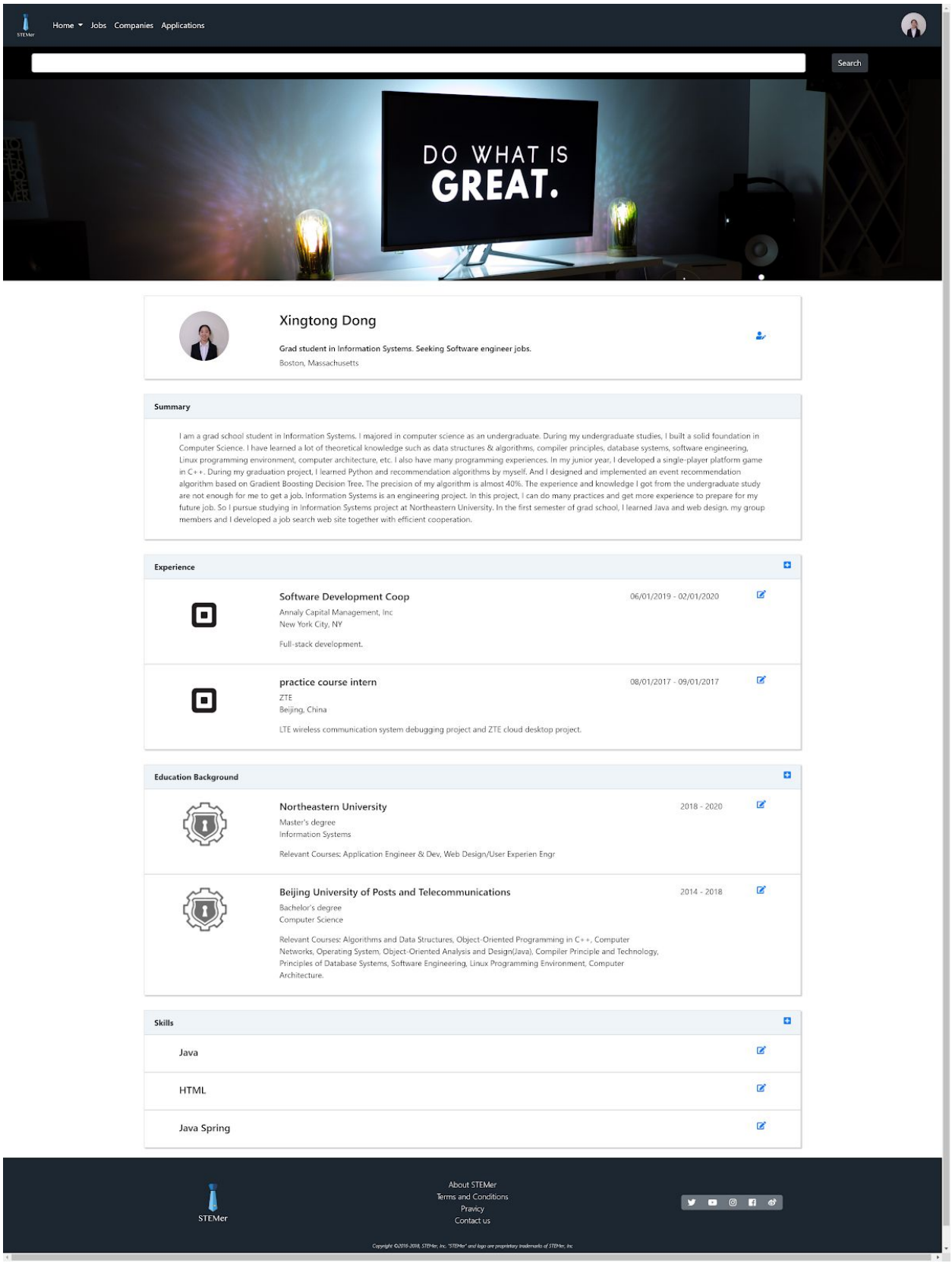


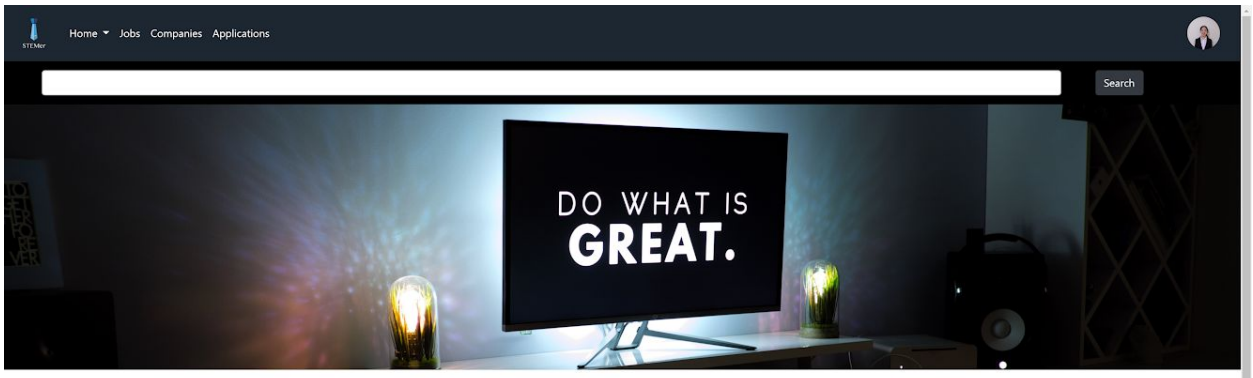

STEMer
Home
Jobs
Companies
Applications
Search
DO WHAT IS
GREAT.
Accenture
New York, NY
See all open positions
About Company
Accenture is a leading global professional services company, providing a broad range of services and solutions in strategy, consulting, digital, technology and operations. Combining unmatched experience and specialized skills across more than 40 industries and all business functions?underpinned by the world?s largest delivery network?Accenture works at the intersection of business and technology to help clients improve their performance and create sustainable value for their stakeholders. With more than 450,000 people serving clients in over 120 countries, Accenture drives innovation to improve the way the world works and lives.
Company details
Website
http://www.accenture.com
Primary Location
New York, NY
Year Founded
1989
Specialties
Information Technology
Published Positions
Software Engineer
Accenture
New York, NY
United States
Posted on 04-19-2020
View Position
Systems Software Engineer
Accenture
New York, NY
United States
Posted on 04-15-2020
View Position
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us


STEMer
Home
Jobs
Companies
Applications
Search
DO WHAT IS
GREAT.
Software Engineer
Company: Accenture
Work Location: New York NY, 10105 United States
Posted on: 2020-04-19 18:59:55.0
Apply
Job Information
Job Description
Data analysts translate numbers into plain English Every business collects data, whether it's sales figures, market research, logistics, or transportation costs. A data analyst's job is to take that data and use it to help companies make better business decisions.
Employment Type
Full Time
Responsibilities
Data analysts translate numbers into plain English Every business collects data, whether it's sales figures, market research, logistics, or transportation costs. A data analyst's job is to take that data and use it to help companies make better business decisions.
Salary Range
From 13k to 18k
position status
Available
Apply
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc.


STEMer
Home
Jobs
Companies
Applications
Search
Application
First Name*:
First name
Last Name*:
Last name
Email*:
xxx@domain.com
Phone*:
(xxx)-xxx-xxxx
Address:
street, apt
City:
City
State:
State
Zip Code:
xxxxx
Country:
Country
Resume:
Choose file
Browse
Begin Date:
MM/DD/YYYY
Website, Blog or Portfolio:
website
Submit
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc


STEMer
Home
Jobs
Companies
Applications
Search
Applicant Information
Name
Xingtong Dong
Email
xtdong1001@gmail.com
Phone
8. .201
Address
650 Columbus Ave, Douglass Park, U.
Boston, Massachusetts 02118, United States
Reference
Resume
Resume-Xingtong Dong_104_1587306716257.pdf
Download
Date Available
05/01/2020
Website, Blog or Portfolio
https://www.linkedin.com/in/xingtong-dong/
Feedback
Application Status
Pending
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc.


STEMer
Home
Jobs
Companies
Applications
Search
DO WHAT IS
GREAT.
Xingtong Dong
Grad student in Information Systems. Seeking Software engineer jobs.
Boston, Massachusetts
Summary
I am a grad school student in Information Systems. I majored in computer science as an undergraduate. During my undergraduate studies, I built a solid foundation in Computer Science. I have learned a lot of theoretical knowledge such as data structures & algorithms, compiler principles, database systems, software engineering, Linux programming environment, computer architecture, etc. I also have many programming experiences. In my junior year, I developed a single-player platform game in C++. During my graduation project, I learned Python and recommendation algorithms by myself. And I designed and implemented an event recommendation algorithm based on Gradient Boosting Decision Tree. The precision of my algorithm is almost 40%. The experience and knowledge I got from the undergraduate study are not enough for me to get a job. Information Systems is an engineering project. In this project, I can do many practices and get more experience to prepare for my future job. So I pursue studying in Information Systems project at Northeastern University. In the first semester of grad school, I learned Java and web design. my group members and I developed a job search web site together with efficient cooperation.
Experience
Software Development Coop
06/01/2019 - 02/01/2020
Annaly Capital Management, Inc
New York City, NY
Full-stack development.
practice course intern
08/01/2017 - 09/01/2017
ZTE
Beijing, China
LTE wireless communication system debugging project and ZTE cloud desktop project.
Education Background
Northeastern University
2018 - 2020
Master's degree
Information Systems
Relevant Courses: Application Engineer & Dev, Web Design/User Experien Engr
Beijing University of Posts and Telecommunications
2014 - 2018
Bachelor's degree
Computer Science
Relevant Courses: Algorithms and Data Structures, Object-Oriented Programming in C++, Computer Networks, Operating System, Object-Oriented Analysis and Design(Java), Compiler Principle and Technology, Principles of Database Systems, Software Engineering, Linux Programming Environment, Computer Architecture.
Skills
Java
HTML
Java Spring
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us



Profile

Portrait:

Choose file | Browse

First Name*: Xingtong

Last Name*: Dong

Headline:

Grad student in Information Systems. Seeking Software engineer jobs.

Email*: dong.xing@husky.neu.edu

Phone: 8574722201

Address: 650 Columbus Ave, Douglass Park, Unit232

City: Boston

State: Massachusetts

Zip Code: 02118

Country: United States

Summary:

I am a grad school student in Information Systems. I majored in computer science as an undergraduate.

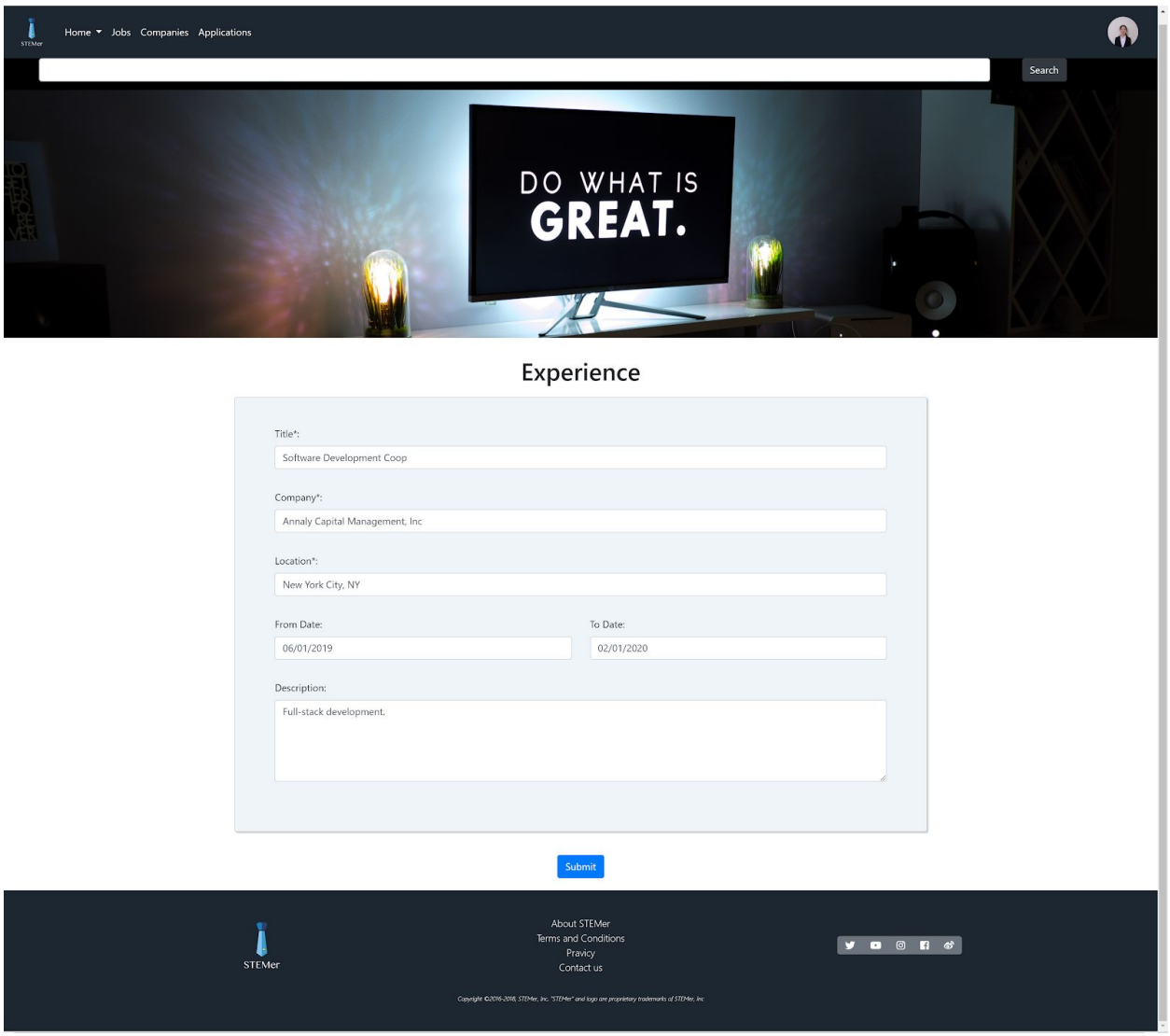
During my undergraduate studies, I built a solid foundation in Computer Science. I have learned a lot of theoretical knowledge such as data structures & algorithms, compiler principles, database systems, software engineering, Linux programming environment, computer architecture, etc.

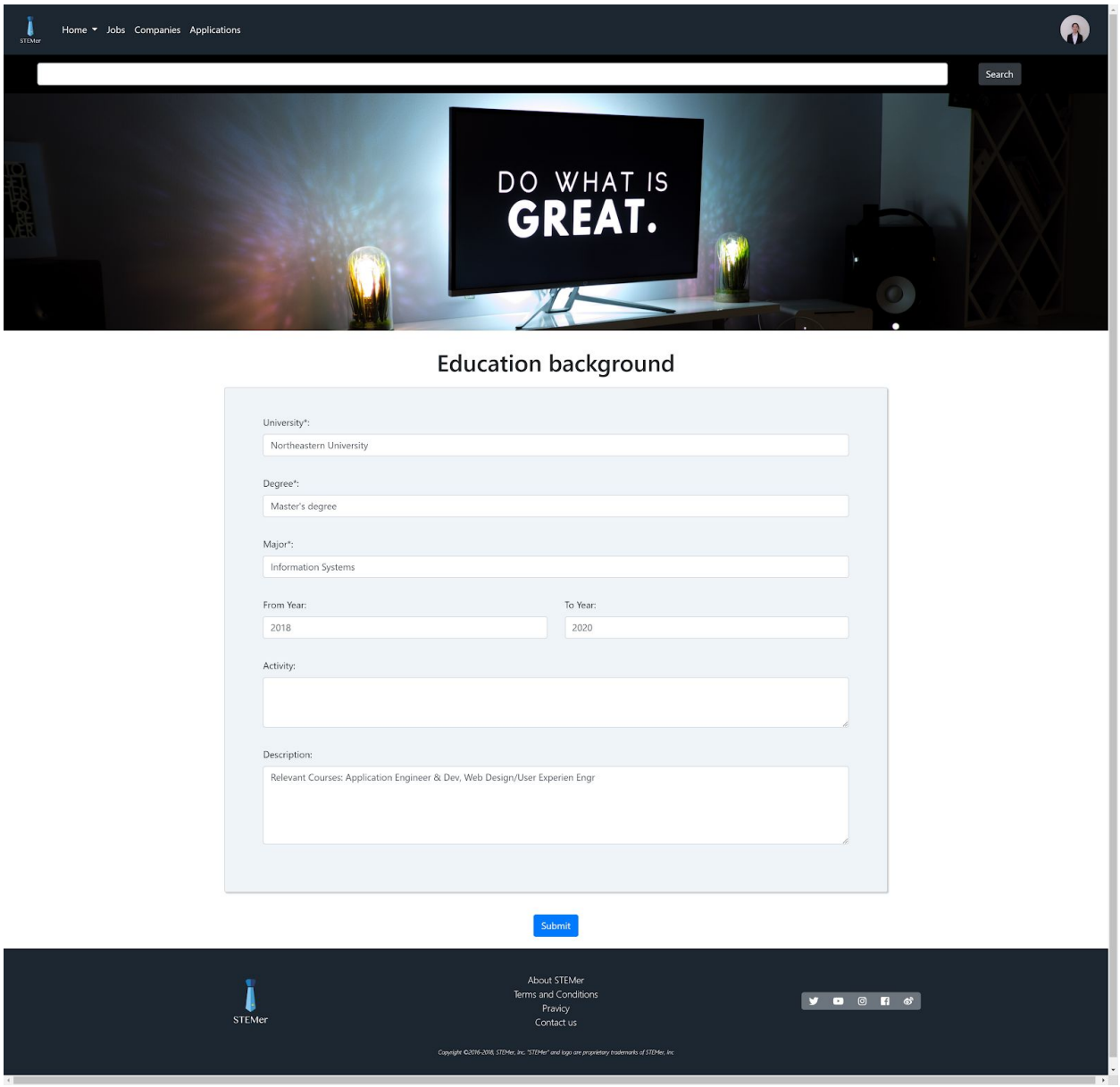
Submit

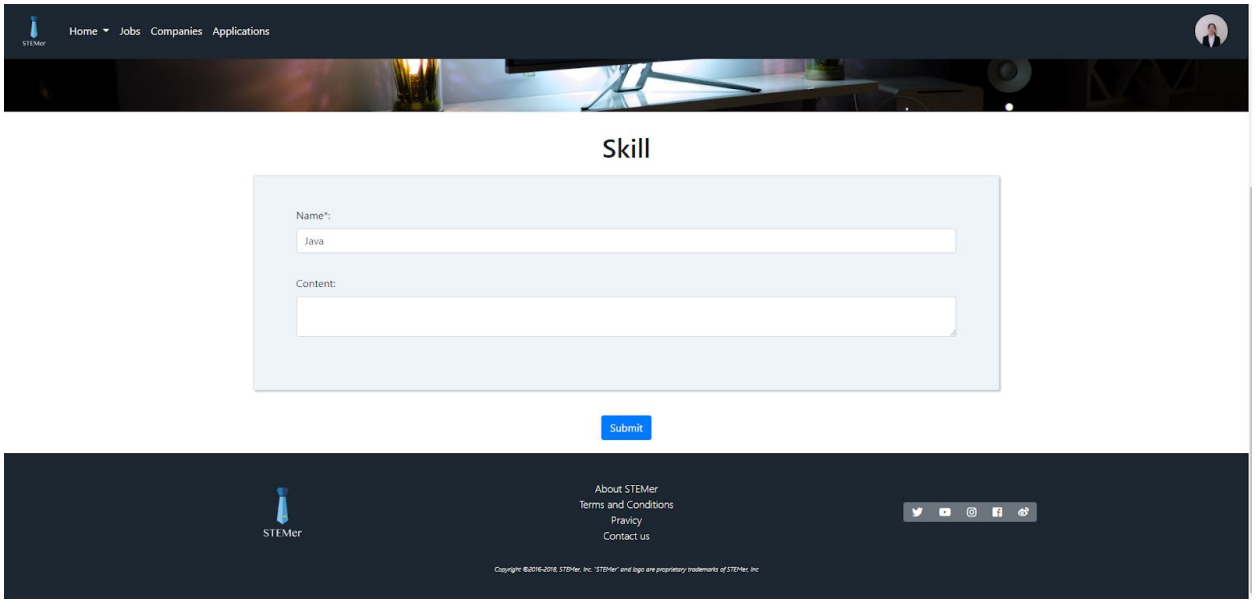
STEMer

About STEMer
Terms and Conditions
Pravicy
Contact us

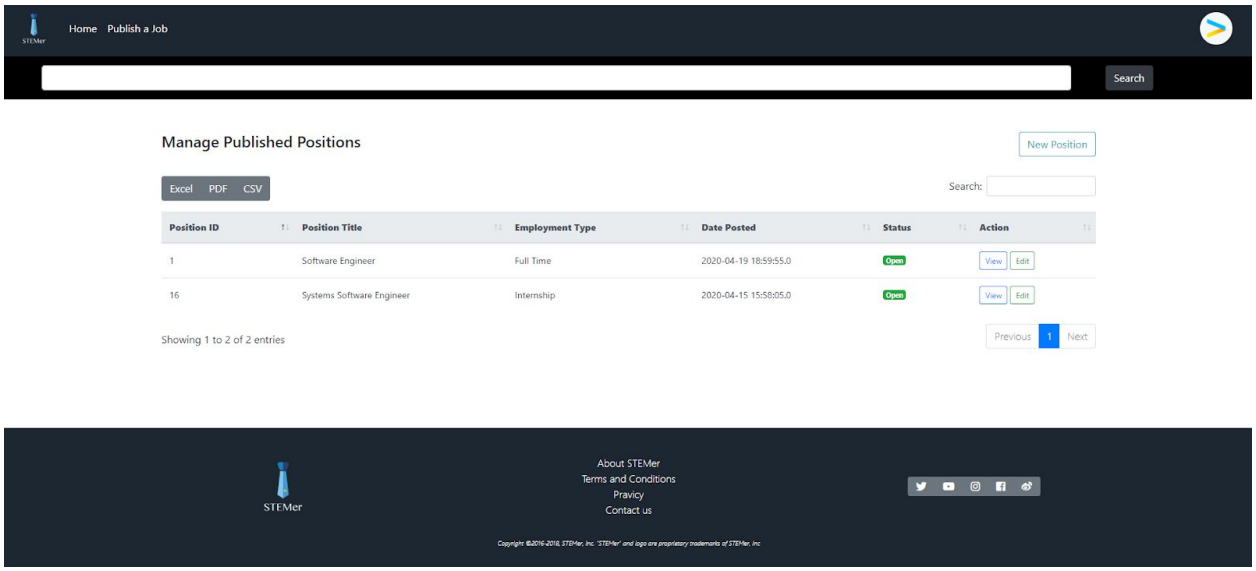
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc

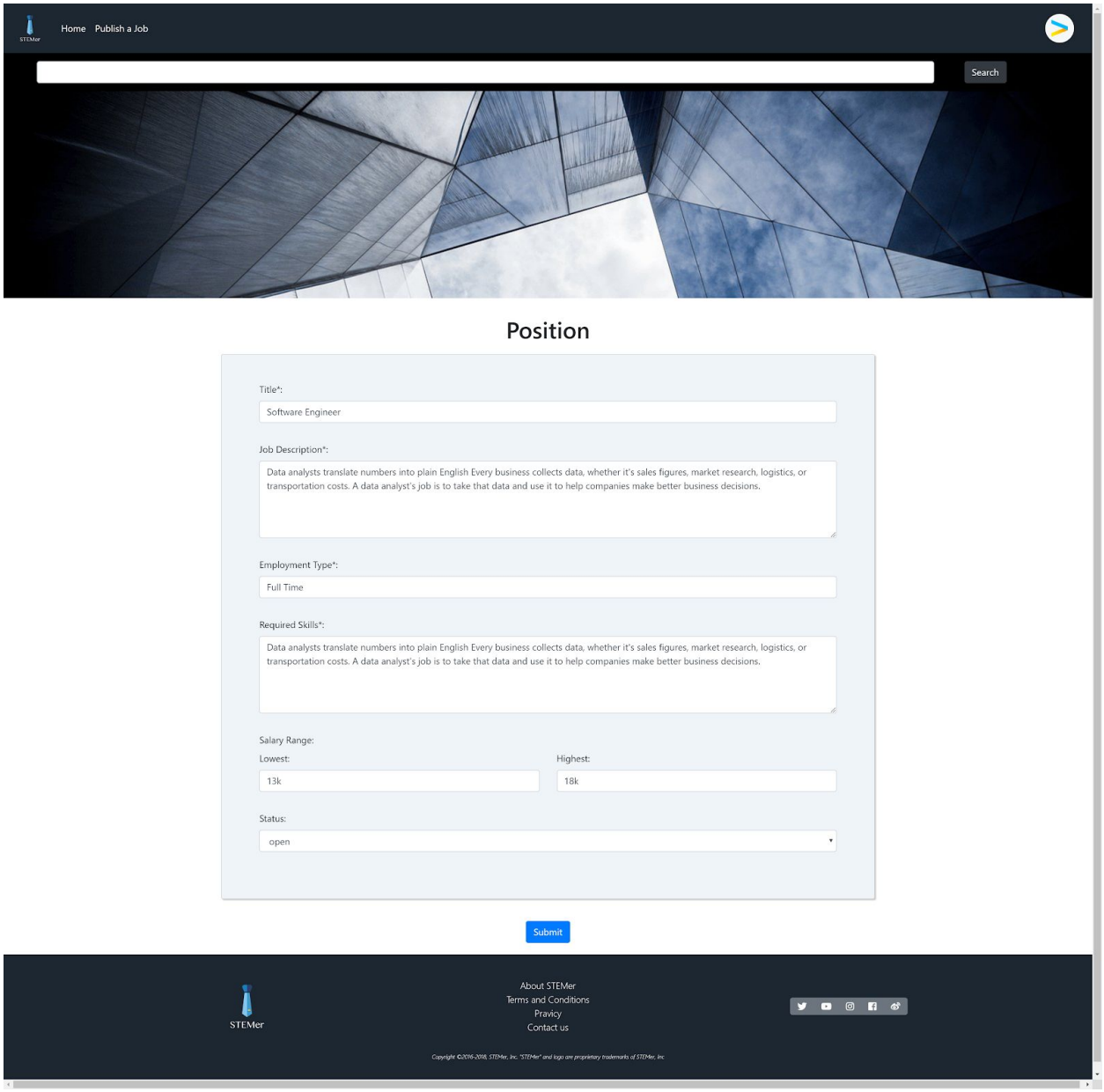

STEMer
Home
Jobs
Companies
Applications
Search
DO WHAT IS
GREAT.
Experience
Title*:
Software Development Coop
Company*:
Annaly Capital Management, Inc
Location*:
New York City, NY
From Date:
06/01/2019
To Date:
02/01/2020
Description:
Full-stack development.
Submit
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc.

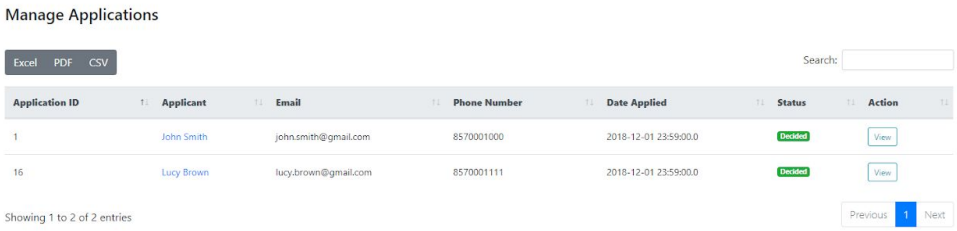

STEMer
Home
Jobs
Companies
Applications
Search
DO WHAT IS
GREAT.
Education background
University*:
Northeastern University
Degree*:
Master's degree
Major*:
Information Systems
From Year:
2018
To Year:
2020
Activity:
Description:
Relevant Courses: Application Engineer & Dev, Web Design/User Experien Engr
Submit
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc.



Company Side:




STEMer
Home
Publish a Job
Search
Position
Title*:
Software Engineer
Job Description*:
Data analysts translate numbers into plain English Every business collects data, whether it's sales figures, market research, logistics, or transportation costs. A data analyst's job is to take that data and use it to help companies make better business decisions.
Employment Type*:
Full Time
Required Skills*:
Data analysts translate numbers into plain English Every business collects data, whether it's sales figures, market research, logistics, or transportation costs. A data analyst's job is to take that data and use it to help companies make better business decisions.
Salary Range:
Lowest:
13k
Highest:
18k
Status:
open
Submit
STEMer
About STEMer
Terms and Conditions
Pravicy
Contact us
Copyright ©2016-2018, STEMer, Inc. "STEMer" and logo are proprietary trademarks of STEMer, Inc


Manage Applications
Excel
PDF
CSV
Search:
Application ID
Applicant
Email
Phone Number
Date Applied
Status
Action
1
John Smith
john.smith@gmail.com
8570001000
2018-12-01 23:59:00.0
Decided
View
16
Lucy Brown
lucy.brown@gmail.com
8570001111
2018-12-01 23:59:00.0
Decided
View
Showing 1 to 2 of 2 entries
Previous
1
Next

Feedback

Application Status
Pending

Schedule an Interview | Reject

Interview Arrangement

Interview Time:

MM/DD/YYYY

Interview Location:

Interview Location

Comments:

Comments

Submit

Feedback

Application Status
Pending

Schedule an Interview | Reject

Rejection

Comments:

Comments

Submit

Feedback

Application Status
Decided

Scheduled Interview

Interview Arrangement

Date

2018-12-10 23:59:00.0

Location

Boston

Comments

Congratulations to have an interview

Company

Company Name*:
Accenture

Address:
1345 6th Ave

City:
New York

State:
NY

Zip Code:
10105

Country:
United States

Found Year:
1989

Industry:
Information Technology

Website
http://www.accenture.com

Description:
Accenture is a leading global professional services company, providing a broad range of services and solutions in strategy, consulting, digital, technology and operations. Combining unmatched experience and specialized skills across more than 40 industries and all business functions?underpinned by the world?s largest delivery network?Accenture works at the intersection of business and technology to help clients improve their performance and create sustainable value for their stakeholders. With more than 450,000 people serving clients in over 120 countries, Accenture drives innovation to improve the way the world works and lives.

Logo:
Choose file Browse

Submit

CONTROLLER SOURCE CODE

ApplicationController.java

```
package com.xdong.controller;

import java.io.File;
import java.io.IOException;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.List;

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

import org.apache.commons.io.FilenameUtils;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.multipart.commons.CommonsMultipartFile;
import org.springframework.web.servlet.ModelAndView;

import com.xdong.business.EmailSend;
import com.xdong.model.Application;
import com.xdong.model.UserAccount;
import com.xdong.service.IGenericService;
import com.xdong.validator.ApplicationValidator;
import com.xdong.service.IApplicationService;

@Controller
public class ApplicationController {

      private static final Logger logger =
Logger.getLogger(ApplicationController.class);
```

```
	@Autowired
	IApplicationService<Application> applicationService;

	public static final String archivePath = "D:/xtdong/grad/6250 Web dev
tools/Archive";

	@RequestMapping(value = "/application/{id}", method =
RequestMethod.GET)
	public ModelAndView get(@PathVariable("id") int id,
HttpServletRequest request) {
		Application application = applicationService.getById(id);
//
//		if(request.getSession(false).getAttribute("userId") !=
application.getUserAccount().getUserId())
//			return new ModelAndView("redirect:/error");

		ModelAndView mav = new
ModelAndView("applicationDetailed_user");
		mav.addObject("application", application);
		return mav;
	}

	@RequestMapping(value = "/apply/{positionId}", method =
RequestMethod.GET)
	public ModelAndView showForm(HttpServletRequest request,
@PathVariable("positionId") int positionId) {
		ModelAndView mav = new ModelAndView("apply");
		Application application = new Application();
		mav.addObject("application", application);
		mav.addObject("positionId",positionId);
		return mav;
	}

	@RequestMapping(value = "/application/submit", method =
RequestMethod.POST)
	public ModelAndView handleForm(@ModelAttribute("application")
Application application, BindingResult result, HttpServletRequest request)
{
		applicationService.validate(application, result);

		if(result.hasErrors()) {
```

```
                ModelAndView mav = new ModelAndView("apply");
                mav.addObject("application", application);
                mav.addObject("positionId",
application.getPosition().getPositionId());
                return mav;
            }

            application.setApplyTime(new Date());
            application.setStatus("Pending");

            CommonsMultipartFile resume = application.getResume();
            String filename = resume.getOriginalFilename();
            filename = FilenameUtils.removeExtension(filename)
                        + "_" +
request.getSession(false).getAttribute("userId")
                        + "_" + Calendar.getInstance().getTimeInMillis()
                        + "." + FilenameUtils.getExtension(filename);
            application.setResumePath(filename);
            File file = new File(archivePath, filename);
            try {
                resume.transferTo(file);
            } catch (IllegalStateException | IOException e) {
                logger.error(e.getStackTrace());
                return new ModelAndView("error");
            }

            applicationService.saveOrUpdate(application);

            return new ModelAndView("redirect:/user/applications");
        }

        @RequestMapping(value = "/company/application/{id}", method =
RequestMethod.GET)
        public ModelAndView getCompanySide(@PathVariable("id") int id,
HttpServletRequest request) {

            Application application = applicationService.getById(id);
            if(request.getSession(false) == null ||
request.getSession(false).getAttribute("userId") == null) {
                ModelAndView mav = new ModelAndView("login");
                mav.addObject("userAccount", new UserAccount());
                return mav;
```

```
            }
//          else if(request.getSession(false).getAttribute("userId") !=
application.getUserAccount().getUserId())
//                return new ModelAndView("error");
            ModelAndView mav = new
ModelAndView("applicationDetailed_company");
            mav.addObject("application", application);
            return mav;
      }

      @RequestMapping(value = "/company/schedule/{id}", method =
RequestMethod.POST)
      public ModelAndView schedule(@PathVariable("id") int id,
HttpServletRequest request) {
            Application application = applicationService.getById(id);
            application.setStatus("Decided");
            application.setResult("Interview Scheduled");

application.setInterviewLocation(request.getParameter("interviewLocation"))
;
            application.setComments(request.getParameter("comments"));
            SimpleDateFormat fmt = new SimpleDateFormat("MM/dd/yyyy");
            try {

application.setInterviewTime(fmt.parse(request.getParameter("interviewTime"
)));
            } catch (ParseException e) {
                  logger.error(e.getStackTrace());
            }

            applicationService.saveOrUpdate(application);

            EmailSend.send("mailNotify@STEMer.com", application.getEmail(),
"Notification from: STEMer - Your application status has been updated!",
"Notification from: STEMer\nYour application status has been updated!");
            logger.info("Sent email to "+ application.getEmail());

            return new
ModelAndView("redirect:/company/application/"+application.getApplicationId(
));
      }
```

```
	@RequestMapping(value = "/company/reject/{id}", method =
RequestMethod.POST)
	public ModelAndView reject(@PathVariable("id") int id,
HttpServletRequest request) {
		Application application = applicationService.getById(id);
		application.setStatus("Decided");
		application.setResult("Rejected");
		application.setComments(request.getParameter("comments"));
		applicationService.saveOrUpdate(application);

		EmailSend.send("mailNotify@STEMer.com", application.getEmail(),
"Notification from: STEMer - Your application status has been updated!",
"Notification from: STEMer\nYour application status has been updated!");
		logger.info("Sent email to "+ application.getEmail());
		return new
ModelAndView("redirect:/company/application/"+application.getApplicationId(
));
	}
```

```
}
```

CompanyAccountController.java

```
package com.xdong.controller;

import java.util.List;

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

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
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;
```

```
import com.xdong.model.CompanyAccount;
import com.xdong.model.IndividualAccount;
import com.xdong.model.RegisterAccount;
import com.xdong.model.UserAccount;
import com.xdong.service.IGenericService;
import com.xdong.service.IUserAccountService;
import com.xdong.validator.RegisterAccountValidator;

@Controller
public class CompanyAccountController {

	private static final Logger logger =
Logger.getLogger(CompanyAccountController.class);

	@Autowired
	IGenericService<CompanyAccount> companyAccountService;

	@Autowired
	IUserAccountService<UserAccount> userAccountService;

	@RequestMapping(value = "/company/loginProcess", method =
RequestMethod.POST)
	public ModelAndView validateUser(@ModelAttribute("userAccount")
UserAccount userAccount, BindingResult result, HttpServletRequest request,
HttpServletResponse response) {
		userAccountService.validate(userAccount, result);
		if(result.hasErrors()) {
			ModelAndView mav = new ModelAndView("login");
			mav.addObject("userAccount", userAccount);
			return mav;
		}

		if(!userAccountService.check(userAccount)) {
			return new ModelAndView("login", "errMsg", "Email,
password or account type is incorrect.");
		}

		int uId =
userAccountService.getByEmail(userAccount.getEmail()).getUserId();
		CompanyAccount companyAccount =
companyAccountService.getById(uId);
```

```
            Integer companyId = companyAccount.getCompany().getCompanyId();

            HttpSession session = request.getSession(true);
            session.setAttribute("userId", uId);
            session.setAttribute("accountType",
userAccount.getAccountType());
            session.setAttribute("companyId", companyId);

            if(request.getParameter("companyCookies") != null &&
request.getParameter("companyCookies").equals("true")) {
                  Cookie uIdCookie = new Cookie("userId", ""+uId);
                  uIdCookie.setDomain("localhost");
                  uIdCookie.setPath("/STEMer");
                  uIdCookie.setMaxAge(60*60*24*7);

                  Cookie typeCookie = new Cookie("accountType",
userAccount.getAccountType());
                  typeCookie.setDomain("localhost");
                  typeCookie.setPath("/STEMer");
                  typeCookie.setMaxAge(60*60*24*7);

                  Cookie idCookie = new Cookie("Id", ""+companyId);
                  idCookie.setDomain("localhost");
                  idCookie.setPath("/STEMer");
                  idCookie.setMaxAge(60*60*24*7);

                  response.addCookie(uIdCookie);
                  response.addCookie(typeCookie);
                  response.addCookie(idCookie);
            }

            return new ModelAndView("redirect:/company/index");

      }

      @RequestMapping(value = "/company/registerProcess", method =
RequestMethod.POST)
      public ModelAndView addUser(@ModelAttribute("registerAccount")
RegisterAccount registerAccount, BindingResult result, HttpServletRequest
request) {

            new RegisterAccountValidator().validate(registerAccount,
```

```
result);

            if(result.hasErrors()) {
                  ModelAndView mav = new ModelAndView("register");
                  mav.addObject("registerAccount", registerAccount);
                  return mav;
            }

            if(userAccountService.getByEmail(registerAccount.getEmail()) !=
null) {
                  return new ModelAndView("register", "errMsg", "Email is
already used.");
            }

            companyAccountService.add(new CompanyAccount(registerAccount));

            int uId =
userAccountService.getByEmail(registerAccount.getEmail()).getUserId();
            CompanyAccount companyAccount =
companyAccountService.getById(uId);
            Integer companyId = companyAccount.getCompany().getCompanyId();

            HttpSession session = request.getSession(true);
            session.setAttribute("userId", uId);
            session.setAttribute("accountType",
registerAccount.getAccountType());
            session.setAttribute("companyId", companyId);

            return new ModelAndView("redirect:/company/index");
      }
}
```

CompanyController.java

```
package com.xdong.controller;

import java.io.File;
import java.io.IOException;
import java.util.Calendar;
import java.util.List;
```

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

import org.apache.commons.io.FilenameUtils;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.multipart.commons.CommonsMultipartFile;
import org.springframework.web.servlet.ModelAndView;

import com.xdong.model.Company;
import com.xdong.service.IGenericService;

@Controller
@RequestMapping(value = "/company")
public class CompanyController {

      private static final Logger logger =
Logger.getLogger(CompanyController.class);

      @Autowired
      IGenericService<Company> companyService;

      private static final int MAXCOMPANY_USER = 12;
      public static final String archivePath = "D:/xtdong/grad/6250 Web dev
tools/Archive";

      @RequestMapping(value = "/list", method = RequestMethod.GET)
      public ModelAndView list() {
            ModelAndView model = new ModelAndView("company/list");
            List list = companyService.getAll();
            model.addObject("list", list);

            return model;
      }
```

```
	@RequestMapping(value = "/list/{page}", method = RequestMethod.GET)
	public ModelAndView listLimit(@PathVariable("page") int page) {
		List companies =
companyService.getAllLimit((page-1)*MAXCOMPANY_USER, MAXCOMPANY_USER);
		int maxpages =
(int)Math.ceil((double)companyService.getCount()/MAXCOMPANY_USER);

		ModelAndView model = new ModelAndView("companies_user");
		model.addObject("companies", companies);
		model.addObject("pages", maxpages);
		model.addObject("currentPage", page);
		return model;
	}

	@RequestMapping(value = "/{id}", method = RequestMethod.GET)
	public ModelAndView get(@PathVariable("id") int id) {
		ModelAndView mav = new ModelAndView("companyDetailed_user");
		Company company = companyService.getById(id);
		mav.addObject("company", company);
		mav.addObject("positions", company.getPositions());
		return mav;
	}

	@RequestMapping(value = "/mine", method = RequestMethod.GET)
	public ModelAndView getMineCompany(HttpServletRequest request) {
		Integer id =
(Integer)request.getSession(false).getAttribute("companyId");
		ModelAndView mav = new ModelAndView("companyDetailed_company");
		Company company = companyService.getById(id);
		mav.addObject("company", company);
		return mav;
	}

	@RequestMapping(value = "/update", method = RequestMethod.GET)
	public ModelAndView update(HttpServletRequest request) {
		Integer id =
(Integer)request.getSession(false).getAttribute("companyId");
		ModelAndView mav = new ModelAndView("companyEdit");
		Company company = companyService.getById(id);
		mav.addObject("company", company);

		return mav;
```

```
	}

	@RequestMapping(value = "/save", method = RequestMethod.POST)
	public ModelAndView save(@ModelAttribute("company") Company company,
BindingResult result, HttpServletRequest request) {
		companyService.validate(company, result);

		if(result.hasErrors()) {
			ModelAndView mav = new ModelAndView("companyEdit");
			mav.addObject("company", company);
			return mav;
		}

		CommonsMultipartFile logo = company.getLogo();
		if(logo.getSize() != 0) {
			String filename = logo.getOriginalFilename();
			filename = FilenameUtils.removeExtension(filename)
				+ "_" + company.getCompanyId()
				+ "_" +
Calendar.getInstance().getTimeInMillis()
				+ "." + FilenameUtils.getExtension(filename);
			company.setLogoPath(filename);
			File file = new File(archivePath, filename);
			try {
				logo.transferTo(file);
			} catch (IllegalStateException | IOException e) {
				logger.error(e.getStackTrace());
				return new ModelAndView("error");
			}
			logger.info(filename + " saved successfully.");
		}

		companyService.saveOrUpdate(company);
		return new ModelAndView("redirect:/company/mine");
	}

	@RequestMapping(value = "/index", method = RequestMethod.GET)
	public ModelAndView getPositions(HttpServletRequest request) {
		int companyId = (int)
request.getSession(false).getAttribute("companyId");
		Company company =  companyService.getById(companyId);
		List positions = company.getPositions();
```

```
            ModelAndView model = new ModelAndView("positions_company");
            model.addObject("positions", positions);
            return model;
      }

      @ResponseBody
      @RequestMapping(value = "/logo", method = RequestMethod.GET)
      public String getLogo(HttpServletRequest request) {
            int companyId = (int)
request.getSession(false).getAttribute("companyId");
            Company company =  companyService.getById(companyId);
            return company.getLogoPath();
      }
}
```

EduBackgroundController.java

```
package com.xdong.controller;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

import com.xdong.model.EduBackground;
import com.xdong.model.EduBackground;
import com.xdong.model.Profile;
import com.xdong.service.IGenericService;

@Controller
@RequestMapping(value = "/eduBackground")
public class EduBackgroundController {
```

```
	private static final Logger logger =
Logger.getLogger(EduBackgroundController.class);

	@Autowired
	IGenericService<EduBackground> eduBackgroundService;

	@RequestMapping(value = "/{id}", method = RequestMethod.GET)
	public ModelAndView get(@PathVariable("id") int id) {
		ModelAndView mav = new
ModelAndView("eduBackgroundDetailed_user");
		EduBackground eduBackground = eduBackgroundService.getById(id);
		mav.addObject("eduBackground", eduBackground);
		return mav;
	}

	@RequestMapping(value = "/add", method = RequestMethod.GET)
	public ModelAndView add() {
		ModelAndView mav = new ModelAndView("eduBackgroundEdit");
		EduBackground eduBackground = new EduBackground();
		mav.addObject("eduBackground", eduBackground);
		return mav;
	}


	@RequestMapping(value = "/update/{experId}", method =
RequestMethod.GET)
	public ModelAndView update(@PathVariable("experId") int experId) {
		ModelAndView mav = new ModelAndView("eduBackgroundEdit");
		EduBackground eduBackground =
eduBackgroundService.getById(experId);
		mav.addObject("eduBackground", eduBackground);

		return mav;
	}

	@RequestMapping(value = "/save", method = RequestMethod.POST)
	public ModelAndView save(@ModelAttribute("eduBackground")
EduBackground eduBackground, BindingResult result, HttpServletRequest
request) {
		eduBackgroundService.validate(eduBackground, result);

		if(result.hasErrors()) {
```

```
                    ModelAndView mav = new ModelAndView("eduBackgroundEdit");
                    mav.addObject("eduBackground", eduBackground);
                    return mav;
            }
            Profile profile = new Profile();

profile.setProfileId((Integer)request.getSession(false).getAttribute("profi
leId"));
            eduBackground.setProfile(profile);
            eduBackgroundService.saveOrUpdate(eduBackground);
            return new ModelAndView("redirect:/profile/" +
eduBackground.getProfile().getProfileId());
      }
}
```

ExperienceController.java

```
package com.xdong.controller;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

import com.xdong.model.Company;
import com.xdong.model.Experience;
import com.xdong.model.Profile;
import com.xdong.service.IGenericService;

@Controller
@RequestMapping(value = "/experience")
public class ExperienceController {
```

```
	private static final Logger logger =
Logger.getLogger(ExperienceController.class);

	@Autowired
	IGenericService<Experience> experienceService;

	@RequestMapping(value = "/{id}", method = RequestMethod.GET)
	public ModelAndView get(@PathVariable("id") int id) {
		ModelAndView mav = new ModelAndView("experienceDetailed_user");
		Experience experience = experienceService.getById(id);
		mav.addObject("experience", experience);
		return mav;
	}

	@RequestMapping(value = "/add", method = RequestMethod.GET)
	public ModelAndView add() {
		ModelAndView mav = new ModelAndView("experienceEdit");
		Experience experience = new Experience();
		mav.addObject("experience", experience);
		return mav;
	}

	@RequestMapping(value = "/update/{experId}", method =
RequestMethod.GET)
	public ModelAndView update(@PathVariable("experId") int experId) {
		ModelAndView mav = new ModelAndView("experienceEdit");
		Experience experience = experienceService.getById(experId);
		mav.addObject("experience", experience);
		return mav;
	}

	@RequestMapping(value = "/save", method = RequestMethod.POST)
	public ModelAndView save(@ModelAttribute("experience") Experience
experience, BindingResult result, HttpServletRequest request) {
		experienceService.validate(experience, result);

		if(result.hasErrors()) {
			ModelAndView mav = new ModelAndView("experienceEdit");
			mav.addObject("experience", experience);
			return mav;
		}
		Profile profile = new Profile();
```

```
profile.setProfileId((Integer)request.getSession(false).getAttribute("profi
leId"));
            experience.setProfile(profile);
            experienceService.saveOrUpdate(experience);
            return new ModelAndView("redirect:/profile/" +
experience.getProfile().getProfileId());
      }
}
```

IndexController.java

```
package com.xdong.controller;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;
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;

import com.xdong.model.Company;
import com.xdong.model.Position;
import com.xdong.model.Profile;
import com.xdong.service.IGenericService;
import com.xdong.service.IPositionService;

@Controller
public class IndexController {

      private static final Logger logger =
Logger.getLogger(IndexController.class);

      @Autowired
      IPositionService<Position> positionService;
      @Autowired
```

```
	IGenericService<Company> companyService;
	@Autowired
	IGenericService<Profile> profileService;

	private static final int POSITION_LIMIT = 8;
	private static final int COMPANY_LIMIT = 6;

	@RequestMapping(value = "/index", method = RequestMethod.GET)
	public ModelAndView handleRequest() {
		List positions = positionService.getAllLimit(0,
POSITION_LIMIT);
		List companies = companyService.getAllLimit(0, COMPANY_LIMIT);
		ModelAndView model = new ModelAndView("index");
		model.addObject("positions", positions);
		model.addObject("companies", companies);
		return model;
	}

	@RequestMapping(value = "/search", method = RequestMethod.POST)
	public ModelAndView search(HttpServletRequest request) {
		String key = request.getParameter("key");
		if(key == null || key.equals(""))
			return new ModelAndView("redirect:/index");
		List positions = positionService.search(key);
		List companies = companyService.search(key);
		List profiles = profileService.search(key);
		ModelAndView model = new ModelAndView("search");
		model.addObject("positions", positions);
		model.addObject("companies", companies);
		model.addObject("profiles", profiles);
		return model;
	}
}
```

IndividualAccountController.java

```
package com.xdong.controller;

import java.util.List;
```

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

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
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;

import com.xdong.model.Application;
import com.xdong.model.CompanyAccount;
import com.xdong.model.IndividualAccount;
import com.xdong.model.RegisterAccount;
import com.xdong.model.UserAccount;
import com.xdong.service.IGenericService;
import com.xdong.service.IUserAccountService;
import com.xdong.service.IndividualAccountService;
import com.xdong.validator.RegisterAccountValidator;

@Controller
public class IndividualAccountController {

      private static final Logger logger =
Logger.getLogger(IndividualAccountController.class);

      @Autowired
      IGenericService<IndividualAccount> individualAccountService;

      @Autowired
      IUserAccountService<UserAccount> userAccountService;

      @RequestMapping(value = "/user/applications", method =
RequestMethod.GET)
      public ModelAndView getUserApplications(HttpServletRequest request) {
            if(request.getSession(false) == null ||
request.getSession(false).getAttribute("userId") == null)
                  return new ModelAndView("login");
```

```
            List applications =
individualAccountService.getById((int)request.getSession(false).getAttribut
e("userId")).getApplications();
            ModelAndView mav = new ModelAndView("applications_user");
            mav.addObject("applications", applications);
            return mav;
      }

      @RequestMapping(value = "/individual/loginProcess", method =
RequestMethod.POST)
      public ModelAndView validateUser(@ModelAttribute("userAccount")
UserAccount userAccount, BindingResult result,  HttpServletRequest request,
HttpServletResponse response) {
            userAccountService.validate(userAccount, result);
            if(result.hasErrors()) {
                  ModelAndView mav = new ModelAndView("login");
                  mav.addObject("userAccount", userAccount);
                  return mav;
            }

            if(!userAccountService.check(userAccount)) {
                  return new ModelAndView("login", "errMsg", "Email,
password or account type is incorrect.");
            }



            int uId =
userAccountService.getByEmail(userAccount.getEmail()).getUserId();
            IndividualAccount individualAccount =
individualAccountService.getById(uId);
            Integer profileId =
individualAccount.getProfile().getProfileId();

            HttpSession session = request.getSession(true);
            session.setAttribute("userId", uId);
            session.setAttribute("accountType",
userAccount.getAccountType());
            session.setAttribute("profileId", profileId);

            if(request.getParameter("individualCookies") != null &&
request.getParameter("individualCookies").equals("true")) {
```

```
                Cookie uIdCookie = new Cookie("userId", ""+uId);
                uIdCookie.setDomain("localhost");
                uIdCookie.setPath("/STEMer");
                uIdCookie.setMaxAge(60*60*24*7);

                Cookie typeCookie = new Cookie("accountType",
userAccount.getAccountType());
                typeCookie.setDomain("localhost");
                typeCookie.setPath("/STEMer");
                typeCookie.setMaxAge(60*60*24*7);

                Cookie idCookie = new Cookie("Id", ""+profileId);
                idCookie.setDomain("localhost");
                idCookie.setPath("/STEMer");
                idCookie.setMaxAge(60*60*24*7);

                response.addCookie(uIdCookie);
                response.addCookie(typeCookie);
                response.addCookie(idCookie);
            }

            return new ModelAndView("redirect:/index");

        }

        @RequestMapping(value = "/individual/registerProcess", method =
RequestMethod.POST)
        public ModelAndView addUser(@ModelAttribute("registerAccount")
RegisterAccount registerAccount, BindingResult result, HttpServletRequest
request) {

            new RegisterAccountValidator().validate(registerAccount,
result);

            if(result.hasErrors()) {
                ModelAndView mav = new ModelAndView("register");
                mav.addObject("registerAccount", registerAccount);
                return mav;
            }

            if(userAccountService.getByEmail(registerAccount.getEmail()) !=
null) {
```

```
                    return new ModelAndView("register", "errMsg", "Email is
already used.");
            }

            individualAccountService.add(new
IndividualAccount(registerAccount));

            int uId =
userAccountService.getByEmail(registerAccount.getEmail()).getUserId();

            IndividualAccount individualAccount =
individualAccountService.getById(uId);
            Integer profileId =
individualAccount.getProfile().getProfileId();

            HttpSession session = request.getSession(true);
            session.setAttribute("userId", uId);
            session.setAttribute("accountType",
registerAccount.getAccountType());
            session.setAttribute("profileId", profileId);

            return new ModelAndView("redirect:/index");
      }

}
```

PositionController.java

```
package com.xdong.controller;

import java.util.Date;
import java.util.List;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
```

```
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

import com.xdong.model.Company;
import com.xdong.model.Position;
import com.xdong.service.IGenericService;
import com.xdong.service.IPositionService;

@Controller
@RequestMapping(value = "/position")
public class PositionController {

      private static final Logger logger =
Logger.getLogger(PositionController.class);

      @Autowired
      IPositionService<Position> positionService;
      private static final int MAXPOSITION_USER = 12;

      @RequestMapping(value = "/list", method = RequestMethod.GET)
      public ModelAndView list() {
            ModelAndView model = new ModelAndView("position/list");
            List list = positionService.getAll();
            model.addObject("list", list);

            return model;
      }

      @RequestMapping(value = "/list/{page}", method = RequestMethod.GET)
      public ModelAndView listLimit(@PathVariable("page") int page) {
            List positions =
positionService.getAllLimit((page-1)*MAXPOSITION_USER, MAXPOSITION_USER);
            int maxpages =
(int)Math.ceil((double)positionService.getCount()/MAXPOSITION_USER);

            ModelAndView model = new ModelAndView("positions_user");
            model.addObject("positions", positions);
            model.addObject("pages", maxpages);
            model.addObject("currentPage", page);
            return model;
```

```
	}

	@RequestMapping(value = "/{id}", method = RequestMethod.GET)
	public ModelAndView get(@PathVariable("id") int id) {
		ModelAndView mav = new ModelAndView("positionDetailed_user");
		Position position = positionService.getById(id);
		mav.addObject("position", position);

		return mav;
	}

	@RequestMapping(value = "/company/{id}", method = RequestMethod.GET)
	public ModelAndView getCompanySide(@PathVariable("id") int id) {
		ModelAndView mav = new
ModelAndView("positionDetailed_company");
		Position position = positionService.getById(id);
		mav.addObject("position", position);

		return mav;
	}

	@RequestMapping(value = "/add", method = RequestMethod.GET)
	public ModelAndView add() {
		ModelAndView mav = new ModelAndView("positionEdit");
		Position position = new Position();
		mav.addObject("position", position);

		return mav;
	}

	@RequestMapping(value = "/update/{id}", method = RequestMethod.GET)
	public ModelAndView update(@PathVariable("id") int id) {
		ModelAndView mav = new ModelAndView("positionEdit");
		Position position = positionService.getById(id);
		mav.addObject("position", position);

		return mav;
	}

	@RequestMapping(value = "/save", method = RequestMethod.POST)
	public ModelAndView save(@ModelAttribute("position") Position
position, BindingResult result, HttpServletRequest request) {
```

```
            positionService.validate(position, result);

            if(result.hasErrors()) {
                  ModelAndView mav = new ModelAndView("positionEdit");
                  mav.addObject("position", position);
                  return mav;
            }

            position.setPublishTime(new Date());
            Company company = new Company();

company.setCompanyId((Integer)request.getSession(false).getAttribute("compa
nyId"));
            position.setCompany(company);
            positionService.saveOrUpdate(position);

            return new ModelAndView("redirect:/company/index");
      }

}
```

ProfileController.java

```
package com.xdong.controller;

import java.io.File;
import java.io.IOException;
import java.util.Calendar;
import java.util.List;

import javax.servlet.http.HttpServletRequest;

import org.apache.commons.io.FilenameUtils;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
```

```
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.multipart.commons.CommonsMultipartFile;
import org.springframework.web.servlet.ModelAndView;

import com.xdong.model.Company;
import com.xdong.model.Profile;
import com.xdong.service.IGenericService;

@Controller
@RequestMapping(value = "/profile")
public class ProfileController {

	private static final Logger logger =
Logger.getLogger(ProfileController.class);

	@Autowired
	IGenericService<Profile> profileService;

	public static final String archivePath = "D:/xtdong/grad/6250 Web dev
tools/Archive";


	@RequestMapping(value = "/{id}", method = RequestMethod.GET)
	public ModelAndView get(@PathVariable("id") int id) {
		ModelAndView mav = new ModelAndView("profileDetailed_user");
		Profile profile = profileService.getById(id);
		mav.addObject("profile", profile);
		return mav;
	}

	@RequestMapping(value = "/company/{id}", method = RequestMethod.GET)
	public ModelAndView getCompanySide(@PathVariable("id") int id) {
		ModelAndView mav = new ModelAndView("profileDetailed_company");
		Profile profile = profileService.getById(id);
		mav.addObject("profile", profile);
		return mav;
	}

	@RequestMapping(value = "/update", method = RequestMethod.GET)
	public ModelAndView update(HttpServletRequest request) {
		Integer id =
```

```
(Integer)request.getSession(false).getAttribute("profileId");
            ModelAndView mav = new ModelAndView("profileEdit");
            Profile profile = profileService.getById(id);
            mav.addObject("profile", profile);

            return mav;
      }

      @RequestMapping(value = "/save", method = RequestMethod.POST)
      public ModelAndView save(@ModelAttribute("profile") Profile profile,
BindingResult result, HttpServletRequest request) {
            profileService.validate(profile, result);

            if(result.hasErrors()) {
                  ModelAndView mav = new ModelAndView("profileEdit");
                  mav.addObject("profile", profile);
                  return mav;
            }

            CommonsMultipartFile logo = profile.getPortrait();
            if(logo.getSize() != 0) {
                  String filename = logo.getOriginalFilename();
                  filename = FilenameUtils.removeExtension(filename)
                              + "_" + profile.getProfileId()
                              + "_" +
Calendar.getInstance().getTimeInMillis()
                              + "." + FilenameUtils.getExtension(filename);
                  profile.setPortraitPath(filename);
                  File file = new File(archivePath, filename);
                  try {
                        logo.transferTo(file);
                  } catch (IllegalStateException | IOException e) {
                        logger.error(e.getStackTrace());
                        return new ModelAndView("error");
                  }
                  logger.info(filename + " saved successfully.");
            }

            profileService.saveOrUpdate(profile);
            return new
ModelAndView("redirect:/profile/"+profile.getProfileId());
      }
```

```
	@ResponseBody
	@RequestMapping(value = "/portrait", method = RequestMethod.GET)
	public String getLogo(HttpServletRequest request) {
		int profileId = (int)
request.getSession(false).getAttribute("profileId");
		Profile profile =  profileService.getById(profileId);
		return profile.getPortraitPath();
	}
}
```

SkillController.java

```
package com.xdong.controller;

import javax.servlet.http.HttpServletRequest;

import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

import com.xdong.model.Skill;
import com.xdong.model.Skill;
import com.xdong.model.Profile;
import com.xdong.service.IGenericService;

@Controller
@RequestMapping(value = "/skill")
public class SkillController {

	private static final Logger logger =
Logger.getLogger(SkillController.class);

	@Autowired
```

```
	IGenericService<Skill> skillService;

	@RequestMapping(value = "/{id}", method = RequestMethod.GET)
	public ModelAndView get(@PathVariable("id") int id) {
		ModelAndView mav = new ModelAndView("skillDetailed_user");
		Skill skill = skillService.getById(id);
		mav.addObject("skill", skill);
		return mav;
	}

	@RequestMapping(value = "/add", method = RequestMethod.GET)
	public ModelAndView add() {
		ModelAndView mav = new ModelAndView("skillEdit");
		Skill skill = new Skill();
		mav.addObject("skill", skill);
		return mav;
	}


	@RequestMapping(value = "/update/{experId}", method =
RequestMethod.GET)
	public ModelAndView update(@PathVariable("experId") int experId) {
		ModelAndView mav = new ModelAndView("skillEdit");
		Skill skill = skillService.getById(experId);
		mav.addObject("skill", skill);

		return mav;
	}


	@RequestMapping(value = "/save", method = RequestMethod.POST)
	public ModelAndView save(@ModelAttribute("skill") Skill skill,
BindingResult result, HttpServletRequest request) {
		skillService.validate(skill, result);

		if(result.hasErrors()) {
			ModelAndView mav = new ModelAndView("skillEdit");
			mav.addObject("skill", skill);
			return mav;
		}
		Profile profile = new Profile();
```

```
profile.setProfileId((Integer)request.getSession(false).getAttribute("profi
leId"));
            skill.setProfile(profile);
            skillService.saveOrUpdate(skill);
            return new ModelAndView("redirect:/profile/" +
skill.getProfile().getProfileId());
      }


}
```

UserAccountController.java

```
package com.xdong.controller;

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

import org.apache.log4j.Logger;
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;

import com.xdong.model.RegisterAccount;
import com.xdong.model.UserAccount;
import com.xdong.service.IUserAccountService;

@Controller
public class UserAccountController {

      private static final Logger logger =
Logger.getLogger(UserAccountController.class);

      @RequestMapping(value = "/register", method = RequestMethod.GET)
      public ModelAndView showRegister() {
            ModelAndView mav = new ModelAndView("register");
```

```
            RegisterAccount registerAccount = new RegisterAccount();
            mav.addObject("registerAccount", registerAccount);
            return mav;
      }

      @RequestMapping(value = "/login", method = RequestMethod.GET)
      public ModelAndView showLogin() {
            ModelAndView mav = new ModelAndView("login");
            UserAccount userAccount = new UserAccount();
            mav.addObject("userAccount", userAccount);
            return mav;
      }

      @RequestMapping(value = "/logout", method = RequestMethod.GET)
      public ModelAndView logout(HttpServletRequest request,
HttpServletResponse response) {
            //delete cookies
            for (Cookie cookie : request.getCookies()) {
                if(cookie.getName().equals("userId")
                        || cookie.getName().equals("accountType") ||
cookie.getName().equals("Id")) {
                  cookie.setMaxAge(0);
                  response.addCookie(cookie);
                }
            }

            if(request.getSession(false) == null ||
request.getSession(false).getAttribute("userId") == null)
                  return new ModelAndView("redirect:/company/index");
            request.getSession(false).invalidate();
            return new ModelAndView("redirect:/index");
      }

      @RequestMapping(value = "/error", method = RequestMethod.GET)
      public ModelAndView permissionError(HttpServletRequest request) {
            return new ModelAndView("error", "errMsg", "Sorry, you don't
have the permission");
      }
}
```