

Governors State University

## OPUS Open Portal to University Scholarship

---

All Capstone Projects

Student Capstone Projects

---

2022

### R Stack Career

Hari Krishna Patti

Follow this and additional works at: <https://opus.govst.edu/capstones>



Part of the [Computer Engineering Commons](#)

---

For more information about the academic degree, extended learning, and certificate programs of Governors State University, go to [http://www.govst.edu/Academics/Degree\\_Programs\\_and\\_Certifications/](http://www.govst.edu/Academics/Degree_Programs_and_Certifications/)

Visit the [Governors State Computer Science Department](#)

This Capstone Project is brought to you for free and open access by the Student Capstone Projects at OPUS Open Portal to University Scholarship. It has been accepted for inclusion in All Capstone Projects by an authorized administrator of OPUS Open Portal to University Scholarship. For more information, please contact [opus@govst.edu](mailto:opus@govst.edu).

# **R Stack Career**

By

**Hari Krishna Patti**

B.Tech., Swarna Bharathi Institute of Science and Technology, 2019

GRADUATE CAPSTONE SEMINAR PROJECT

Submitted in partial fulfillment of the requirements

For the Degree of Master of Science,

With a Major in Computer Science



Governors State University  
University Park, IL 60484

2022

## **ABSTRACT**

The Career Engine's main purpose is to create a platform for job searchers to find suitable and gratifying work based on their qualifications. It also connects job searchers with the recruiting process. This career engine recognizes when a job seeker applies for a job and when a company publishes a job and chooses a candidate. A career engine's principal objective is to aid job searchers in getting a quick career. In today's society, there are several employment portal platforms. In this case, we must create a web-based application utilizing the Career Engine. In this period of decline, everyone, experienced or unskilled, is seeking for a job. Our career engine might be quite beneficial since it allows people with a variety of profiles to submit their CVs and search for employment depending on their qualifications. Each user may connect with their user id and apply for many jobs at the same time. Our team opted to build the website with the MVC framework in Dot Net Core for the backend and HTML5, CSS3, Bootstrap, JavaScript, and Ajax for the front end, with data saved in MS-SQL. The career engine system is used by Applicants, Recruiters, and Administrators. The administrator oversees the entire application. Recruiters can post job openings as needed. Applicants may submit a CV containing both personal and professional information.

# Table of Content

<b>1</b>	<b><i>Project Description</i></b> .....	1
1.1	Competitive Information .....	1
1.2	Relationship to Other Applications/Projects .....	1
1.3	Assumptions and Dependencies .....	1
1.4	Future Enhancements.....	2
1.5	Definitions and Acronyms.....	2
<b>2</b>	<b><i>Project Technical Description</i></b> .....	2
2.1	Application Architecture .....	3
2.2	Application Information flows.....	3
2.3	Interactions with other Applications .....	5
2.4	Capabilities .....	5
2.5	Risk Assessment and Management .....	6
<b>3</b>	<b><i>Project Requirements</i></b> .....	6
3.1	Identification of Requirements .....	6
3.2	Operations, Administration, Maintenance and Provisioning (OAM&P) .....	6
3.3	Security and Fraud Prevention.....	7
3.4	Release and Transition Plan.....	7
<b>4</b>	<b><i>Project Design Description</i></b> .....	7
<b>5</b>	<b><i>Internal/external Interface Impacts and Specification</i></b> .....	8
<b>6</b>	<b><i>Design Units Impacts</i></b> .....	18
6.1	Functional Area A/Design Unit A .....	18
6.1.1	<b>Functional Overview</b> .....	18
6.1.2	<b>Impacts</b> .....	19
6.1.3	<b>Requirements</b> .....	19
6.2	Functional Area B/Design Unit B.....	19
6.2.1	<b>Functional Overview</b> .....	19
6.2.2	<b>Impacts</b> .....	19
6.2.3	<b>Requirements</b> .....	19
<b>7</b>	<b><i>Open Issues</i></b> .....	19
<b>8</b>	<b><i>Acknowledgements</i></b> .....	20
<b>9</b>	<b><i>References</i></b> .....	20
<b>10</b>	<b><i>Appendices</i></b> .....	20

## ***1 Project Description***

The main objective of a Career Engine is to make it simpler for job searchers to swiftly obtain employment. As a consequence, it makes it easy for both applicants and those who don't apply to search for employment, as well as for recruiters to find the best candidates. Users can take on the roles of Home, Admin, Recruiter, and Seeker in this application. The development of the career engine has as its goal to offer a platform for job seekers to find suitable and fulfilling work based on their qualifications. Additionally, it connects recruiters with job seekers. Employers may select the most qualified candidate from the available CVs by allowing job searchers to submit their resumes and apply for job ads. Recruiters advertise openings and pick the top individuals on this website, which functions effectively as a job board for applicants. All work categories are provided by career engines, which also help people find different jobs.

### ***1.1 Competitive Information***

Though there are many consulting firms, the majority need money before beginning the search. One of the most recognizable competitors in the job site sector was picked to be Career Builder [2]. Applying for jobs and looking at job seekers' profiles are both possible on career builder. In comparison to Career Builder, we provide a free job search application that also allows visitors to hunt for employment. So that recruiters could more easily promote the opportunity, we featured the position on the top page.

### ***1.2 Relationship to Other Applications/Projects***

We establish cooperation with the Indeed application. The presentation of pertinent information, as we did, may arise from entering the job title, keyword, or location. Job seekers must sign in using the same method we did for our career engine application to submit their resumes.

### ***1.3 Assumptions and Dependencies***

- The computer should have an internet connection and the capability to operate as an internet server;
- The user is familiar with using computers.
- Because the user interface is only available in English, the user must be fluent in English.
- The member database may be accessed by the product.
- The administrator has to be familiar with this career management application.

## ***1.4 Future Enhancements***

Integrations with social media help businesses establish a solid online presence. Some of the social media channels that users of our career engine site use to easily share job posts include LinkedIn [5], Twitter [8], Facebook [3], and Instagram [4]. Consequently, we might be able to do so by upgrading our application.

## ***1.5 Definitions and Acronyms***

- HTML5 is Hypertext Markup Language 5
- CSS3 is Cascading Style Sheet
- Bootstrap is the most widely used CSS Framework for creating responsive and mobile-first websites. Bootstrap 5 is the most recent version of the framework.
- JSON is JavaScript Object Notation.
- EF is the Entity Framework Core
- MVC is Model –View – Controller
- LINQ – Language Integrated Query
- .NET Core is the new web framework from Microsoft. .NET Core is the framework you want to use for web development with .NET is Network Enabled Technology

## ***2 Project Technical Description***

The R Stack Career project is a platform for job portals. The user interfaces for this project was built using Dot Net Core MVC [6] features including Entity Framework 6, HTML, CSS, JS, and BOOTSTRAP. We used Identity User to streamline the user administration process, including registration, login, authorizing users, and role assignment. The identity table and other tables may be added to or updated according to data migration, and LINQ [7] makes it simple to convert data into objects. We used an interface and a repository to streamline the project code, giving us more control over the data while maintaining simple, understandable code. We made advantage of partial view login layouts. We employed four consoles in our application. Other names include Admin, Home, Recruiter, and Seeker; these are just a few possibilities. To access the sign-in, sign-up, default roles, and admin credentials, we employed an authentication controller. We used the search, job list, company list, and guest index features of the home controller. We developed the admin controller to manage everything, including creating categories, skill sets, job types, career levels, and other things. Among other things, we collect resume updates, reviews, and job searches

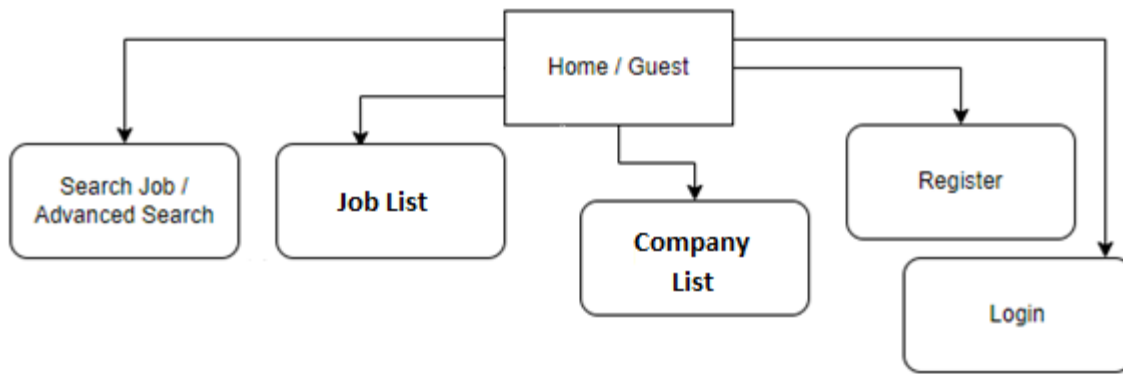
in the seeker controller. In the recruiter controller, we may advertise openings, arrange applicant interviews, and more.

## ***2.1 Application Architecture***

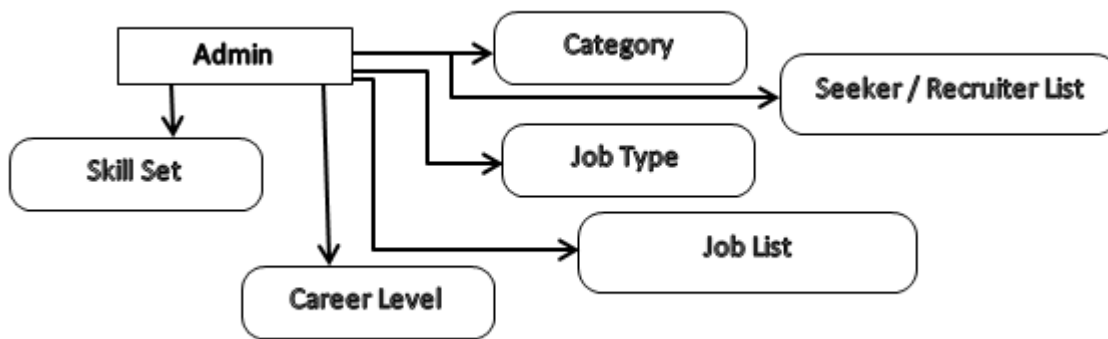
Without having to dump and recreate the database, we were able to update the database schema and alter the data model in this application by using the EF migrations feature. The Home, Admin, Seeker, and Recruiter modules make up this project's architecture shall start by describing the home page before going on to the responsibilities. Anyone can view the verified job posting, look around the business, search for job seekers, register, and log in into the application without having to sign in. Admin is the application's default role. Only admins have the ability to establish categories, skills, job types, and career levels. They can also manage users and job information. The second position in the application is the recruiter. When advertising a job opening, recruiters have the option to modify the picture associated with the position, amend their company profile, locate applicants, go through their resumes, talk with applicants, schedule interviews with applicants, and employ or reject applicants. The third job in the application is the Seeker. The applicant might contact the recruiter and update their resume. The other interface is called home, and its main functions include registering users, finishing the sign-in process, establishing an admin account, assigning roles like "admin," "seeker," and "recruiter," as well as registering users. For both visitors and everyone, we implemented a single layout based on the login function, such as admin, seeker, or recruiter.

## ***2.2 Application Information flows***

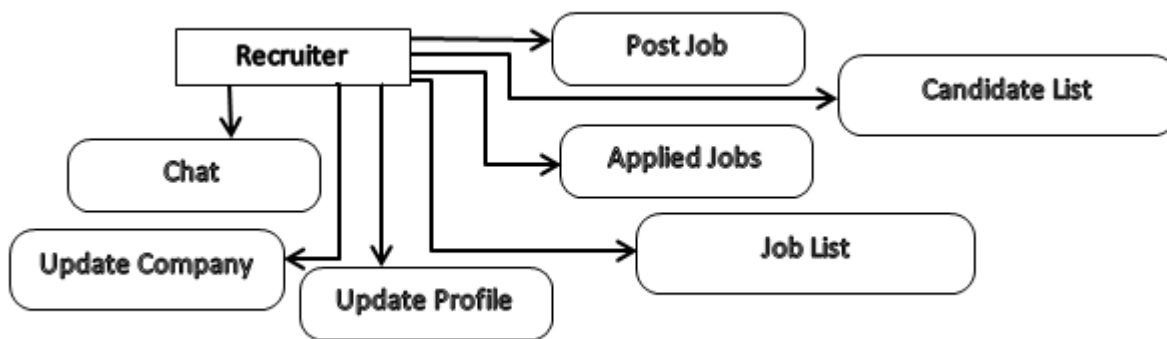
Three distinct processes were implemented to build a website for the career engine. Figure 1 below shows the application procedure. Finding their ideal career is the main phase of the entire program. It can be necessary for the client to use the homepage to search for jobs, look up companies, and explore jobs. The registration procedure is then directed toward you after that. After completing the registration procedure, they will be sent to the login page. Job seekers, recruiters, and administrators can access the main page and search for jobs after logging in.



**Fig 1: Main Page Workflow Diagram**

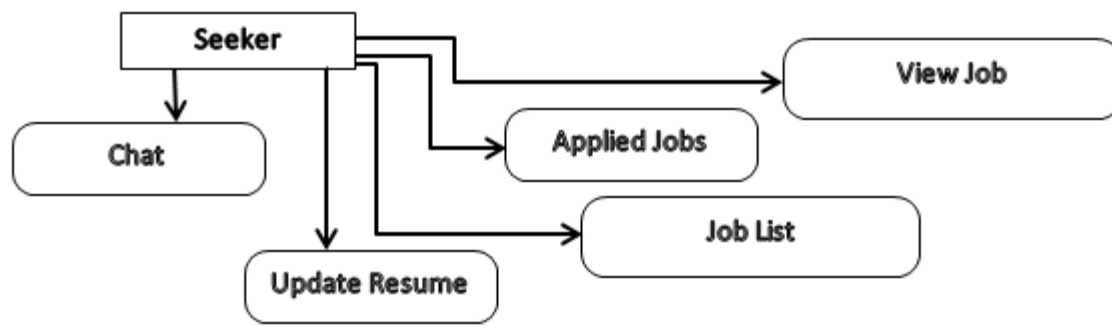


**Fig 2: Admin Workflow Diagram**



**Fig 3: Recruiter Workflow Diagram**





**Fig 4: Seeker Workflow Diagram**

### **2.3 Interactions with other Applications**

Intelligent job search, resume processing, identifying the best match applicant, and hiring process should be basic aspects of all job portals and recruitment software. The majority of our apps are usually designed with broad topographies to encourage user involvement for both recruiters and job seekers, who may benefit from these features to enhance their user experiences in selecting the right candidates at the right moment. The requirements for accepting our application for resolving a typical set of job board business challenges. The three stakeholder kinds that make up the benefit of our career engine fall under the following three categories:

- The Owner of an R Stack Career Application is a Portal Admin.
- Recruiters can post jobs by doing this.
- Seekers as they discover Jobs.

### **2.4 Capabilities**

- ✓ Easy customization; aesthetically pleasing visibility; intuitive user interface, complex features that are also user-friendly
- ✓ Comprehensive job search functionality
- ✓ Depending on the terms, there are several search choices.
- ✓ Location-specific options for searching, Using the category and title filters for jobs Interacting with the recruiter and the job seeker

## ***2.5 Risk Assessment and Management***

Risk management is the hardest part of job portal projects. The risks associated with the application may be anticipated and reduced by using a straightforward and efficient risk management method, even if we can never forecast the future with absolute accuracy. Risk management aids in both crisis scenario prevention and remembering and learning from past errors. This increases the likelihood that the project will be completed successfully while decreasing the risks' negative effects. We have a long way to go before we create a risk management strategy that works for us. We must participate in a continuous process to be able to continuously enhance our procedures to increase the effectiveness of our operations.

### ***3 Project Requirements***

#### ***3.1 Identification of Requirements***

<GSU-RSC\_2022-01Admin-capability-RSC001>

**Handle-** Only the job, the seeker, and the recruiter may be managed by Admin.

< GSU-RSC\_2022-02 Seeker-capability-RSC002>

**Review and Ratings-** Only the seeker can write a review and rating to the company; everyone else is unable to do so.

< GSU-RSC\_2022-03 Seeker-capability-RSC003>

**Resume-** Seeker must upload their resumes before the recruiter selects them; otherwise, the candidate will be initially rejected.

< GSU-RSC \_2022-04 Recruiter-capability- RSC004>

**Schedule Interview-** The recruiter may only arrange interviews for applicants.

< GSU-RSC \_2022-04Communication-capability-RSC005>

**Chat-** To communicate properly between the recruiter and the job seeker.

#### ***3.2 Operations, Administration, Maintenance and Provisioning (OAM&P)***

##### **Operations**

This application is easy to use and understand and is made to help you get a job at the company you want. It should be safe and simple for recruiters. You may visit the website without logging in or registering. Results from the search option should be included in those positions and companies.

##### **Administration**

Only the administrator has access to all users and jobs. The only person who can create, alter, and remove categories, career levels, job types, and other things is the administrator. The user may be

granted access to more information, such as the capability for a recruiter to post a position and for a seeker to update their resume, after being approved by the administrator.

### **Maintenance**

Pages load more quickly when large file sizes from complex web apps are removed. This program ensures that the regular backup includes the entire website and all of the data.

### **Provisioning**

This application guarantees a stable environment and gives users the option to monitor their performance.

## ***3.3 Security and Fraud Prevention***

The security and fraud detection process may be established through security and fraud prevention planning. The following steps make up the planning process for fraud prevention.

- Through the appropriate registration procedure, all user data should be retained, and the program should be regularly updated.
- Management and recruiters have been assigned clear responsibilities and functions.

## ***3.4 Release and Transition Plan***

Website development cannot start until the site is finished and extensively tested. The career portal website has to be released before the issues are fixed. Once the program has been booted, the data storage process will begin, and the maintenance procedure will be used to maintain the application's continuous workflow.

## ***4 Project Design Description***

Dot Net Core was used to construct every aspect of this project. We used the MVC pattern format to do this. We choose the website as a career builder to replicate. In our application, we used two different layouts: one for the main menu and one for registered users. The primary objective of the layout is to display all types of employment information, both with and without a registration process. All user categories are registered using the same process. We made use of a single login page for all users, including administrators, recruiters, and job searchers. We allow each user to take on their own duties, not those of other people. A seeker cannot, for instance, play the part of a recruiter or an administrator.

When a user logs in as an administrator, the Admin Layout Page appears as the first page. The administrator may view all of the data associated with this application on the Dashboard using this Admin Layout. Only the administrator has the authority to create categories, skill sets, job types, and career levels. They may

also control recruiters, job seekers, and job listings. Administrators are unable to change their passwords or make changes to their personal data.

A recruiter who logs in is immediately sent to the Recruiter Layout Page. The recruiter allows users to post jobs, change company information, view application lists, and talk with other users. The position may only be posted by the recruiter, and it will remain open until the closing date has passed. The decision to conduct the interview, employ or reject the candidate, and communicate with them rests solely with the recruiter. The accounts and passwords of recruiters can also be changed.

Upon logging in, a seeker is directed to the seeker layout page. The applicants for the position should just update their resumes; the recruiter will then set up an interview. The primary layout page is accessible to the seeker, recruiter, and administrator. You may perform a job or company search on the homepage. They will all be able to see details about each company, job, etc. A job seeker must first register as a job seeker in order to apply for a position, and a recruiter must first register as a recruiter in order to post a job for their company.

## ***5 Internal/external Interface Impacts and Specification***

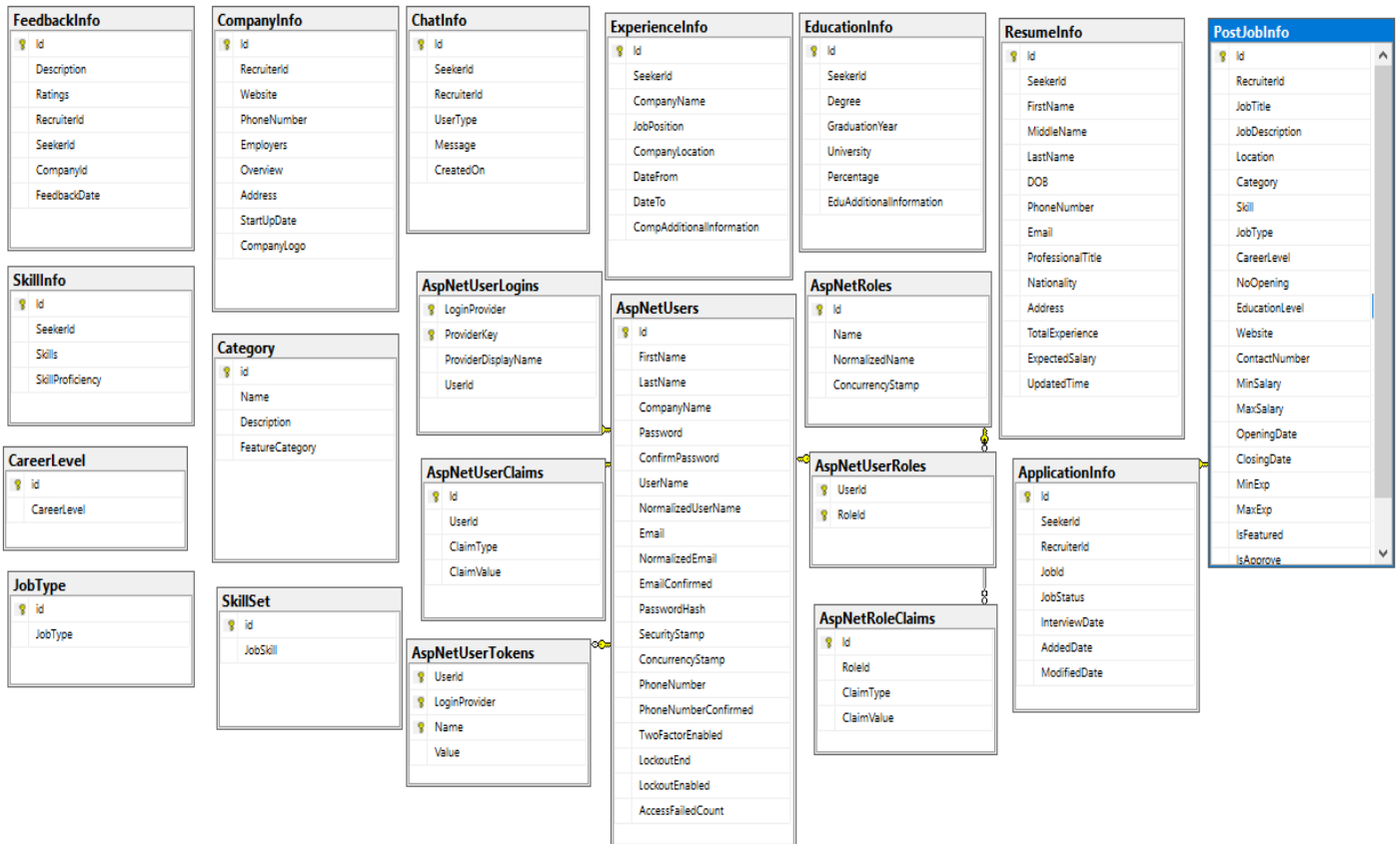
The defining of requirements is a crucial phase in the life cycle of software development. The software and hardware requirements that we utilized to construct this job search website are stated below, even though this application may be built utilizing several software and hardware requirements:

### **Software Requirements:**

<b>Operating System</b>	:	Windows 10
<b>Language</b>	:	C#
<b>Database</b>	:	MSSQL
<b>Front End</b>	:	HTML5, CSS3, Bootstrap5, JavaScript, jQuery
<b>Browser</b>	:	Preferable Google Chrome or Mozilla Firefox or Edge

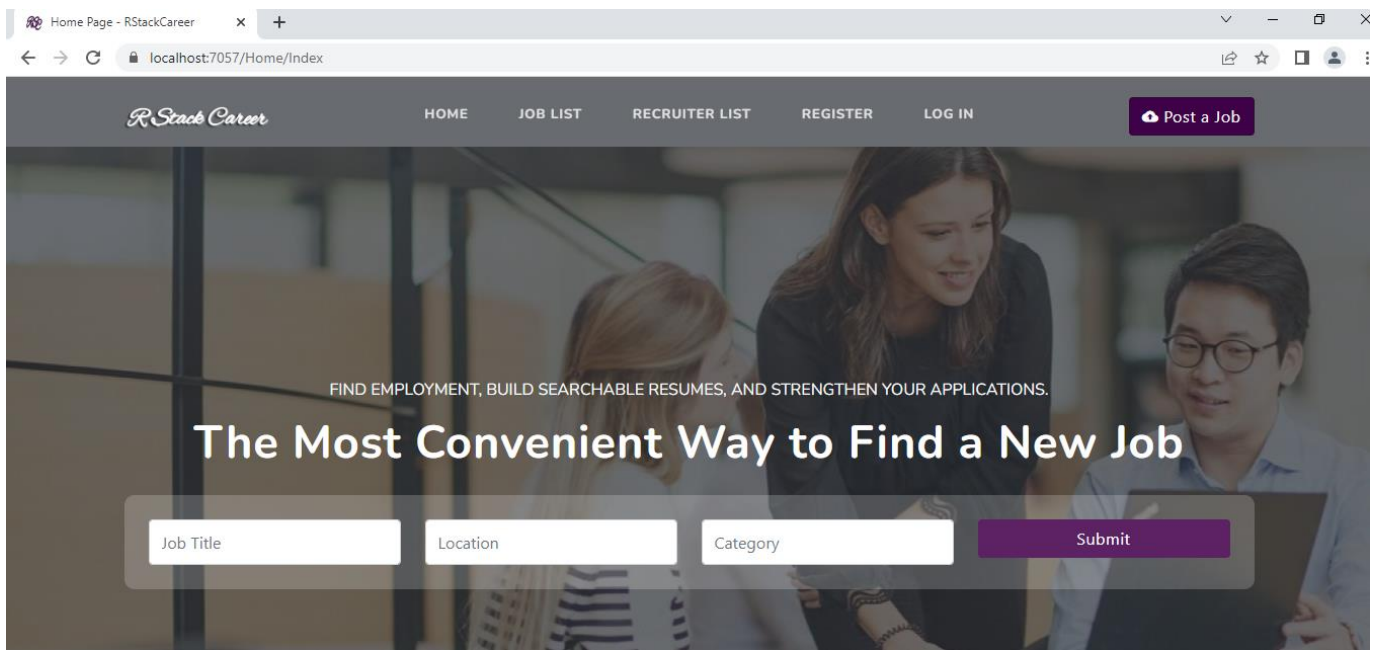


**Fig 5: Software Development Process [1]**

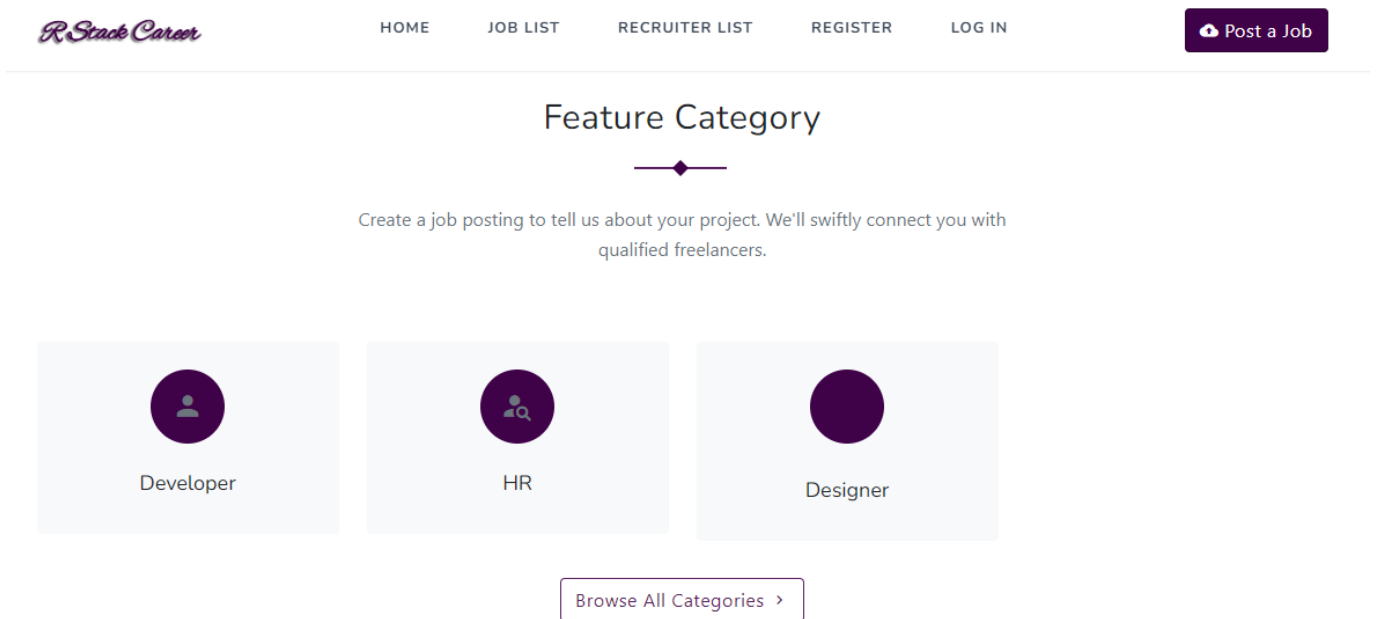


**Fig 6: Database ER Diagram**


The features of the application's design can be seen in the following screenshots:



**Fig 7: Main Page**



**Fig 8: Category List**

	UX Designer Recruit LLC	NY	\$4000.00 - 3000.00	21-11-2022 00:00:00
Experience : 0 - 1 years				Closed <<


	UI Designer Recruit LLC	NY	\$6000.00 - 8000.00	05-12-2022 00:00:00
Experience : 1 - 2 years				Apply Now >>

Fig 9: Feature Job List

R Stack Career    HOME    JOB LIST    RECRUITER LIST    REGISTER    LOG IN    Post a Job

Job Title    Location    Category    Submit

Available job for you

Post a job to tell us about your project. We'll quickly match you with the right freelancers.

Companies

- Developer
- HR
- Designer


	UX Designer Recruit LLC	NY	21-11-2022 00:00:00	4000.00 - 3000.00	Closed
---	----------------------------	----	---------------------	-------------------	--------

Fig 10: All Job List

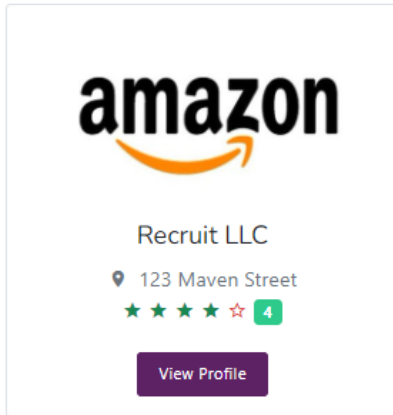
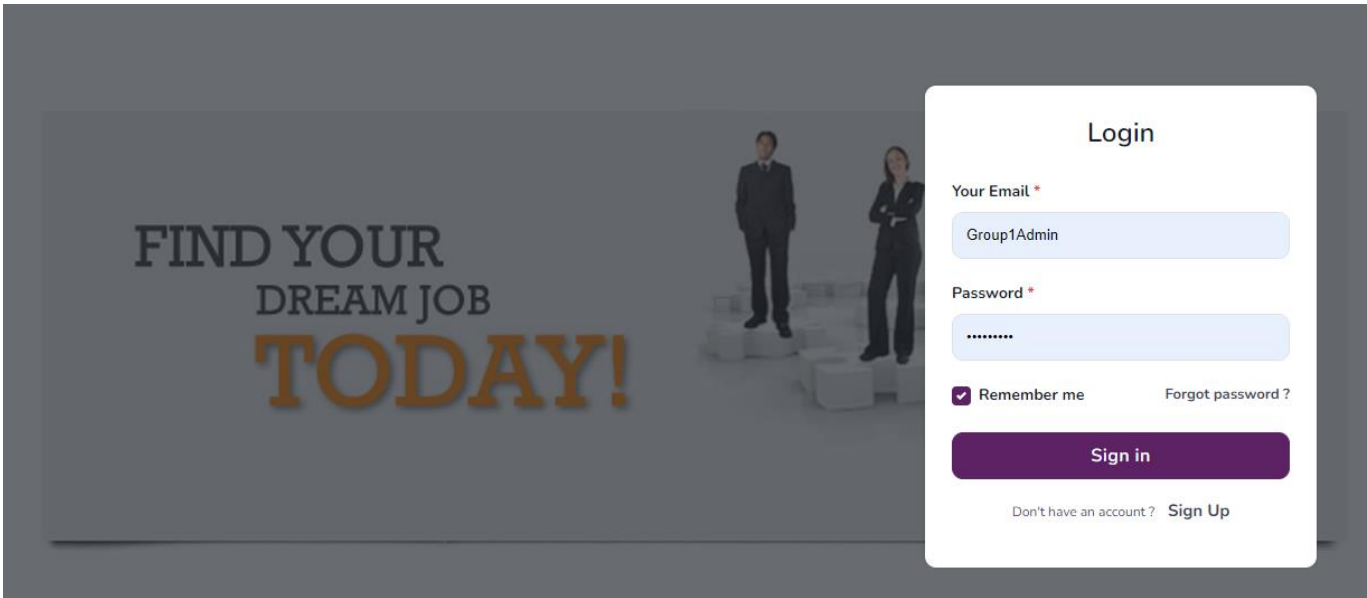


Fig 11: Recruiter List

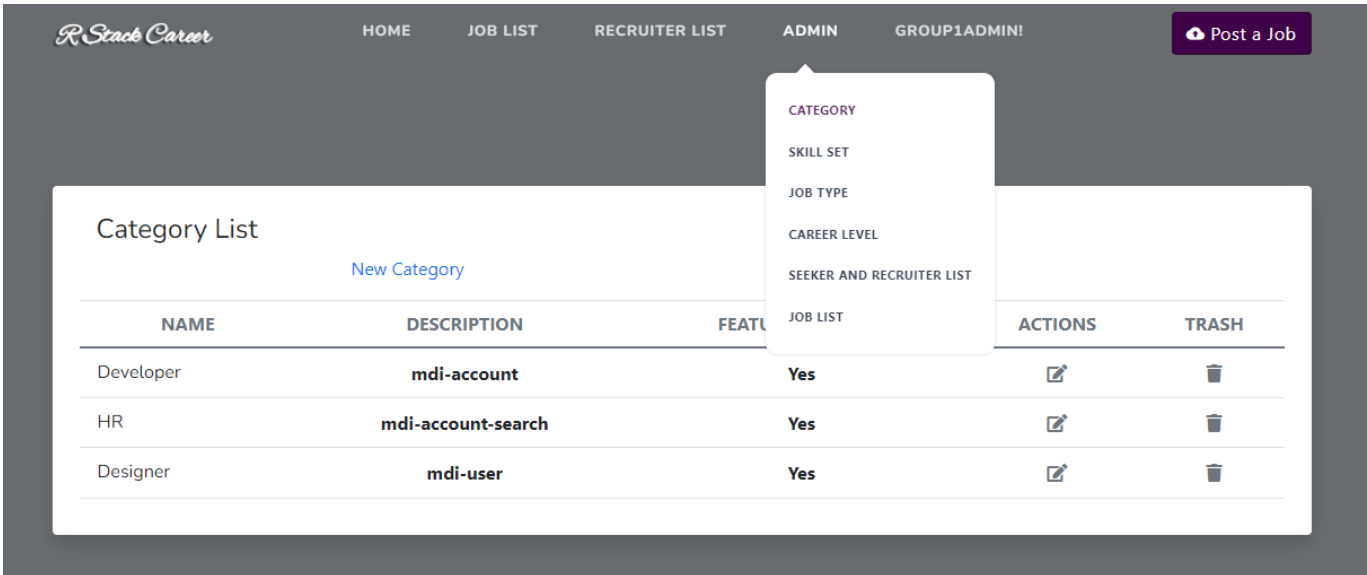


Fig 12: Registration





**Fig 13: Login**



**Fig 14: Admin Layout**

*R Stack Career*    HOME    JOB LIST    RECRUITER LIST    ADMIN    GROUP1ADMIN!    [Post a Job](#)

### Job List

JOB TITLE	JOB DESCRIPTION	LOCATION	JOB OPENING DATE	CLOSING DATE	ACTIONS	DETAILS	APPROVE STATUS
Full Stack Developer	We are looking forward to hire freshers who are interested to work as a Software Developer in Product development. _Job Responsibilities - The job requires an individual who can understand business requirements, assess the fitment with current and emerging technologies _and work within a team to provide effective results.Such individuals will possess great problem solving and analytical skills.	NY	18-11-2022 00:00:00	15-12-2022 00:00:00			Activate
UI Designer	We are looking forward to hire freshers who are interested to work as a Designer in Product development.	NY	21-11-2022 00:00:00	05-12-2022 00:00:00			Activate

**Fig 15: Active / Inactive Jobs**

*R Stack Career*    HOME    JOB LIST    RECRUITER LIST    RECRUITER    RECRUIT!    [Post a Job](#)

### Post a New Job :

**Profile Image**  
 No file chosen

**Job Title**

**Job Type**      **Job Category**  
           

**Career Level**      **Location**      **Website**  
           

**Minimum Salary \$**      **Maximum Salary \$**

- [POST JOB](#)
- [JOB LIST](#)
- [CANDIDATE LIST](#)
- [APPLIED JOBS](#)
- [CHAT TO SEEKER](#)
- [UPDATE COMPANY INFO](#)
- [UPDATE PROFILE](#)

**Fig 16: Recruiter Layout**


*R Stack Career*    HOME    JOB LIST    RECRUITER LIST    RECRUITER    RECRUIT!    [Post a Job](#)

### Applied Job List

JOB TITLE	RECRUITER / SEEKER NAME	APPLIED DATE	APPLICATION STATUS	INTERVIEW PLAN	APPROVE STATUS
Full Stack Developer	FSeeker	20-11-2022 22:12:34	Applied	01-01-0001 00:00:00	<a href="#">View Job</a>
UI Designer	FSeeker	22-11-2022 22:12:34	ShortList	01-01-0001 00:00:00	<a href="#">View Job</a>
UX Designer	FSeeker	22-11-2022 22:12:34	Rejected	01-01-0001 00:00:00	<a href="#">View Job</a>

**Fig 17: Applied Job List**

*R Stack Career*    HOME    JOB LIST    RECRUITER LIST    RECRUITER    RECRUIT!    [Post a Job](#)



**FSeeker** [❤️](#)


Full Stack Developer

Applied

20-11-2022 22:12:34

01-01-0001 00:00:00

[View Profile](#)



**FSeeker** [❤️](#)

UI Designer


ShortList

22-11-2022 22:12:34

01-01-0001 00:00:00

[View Profile](#)    [Arrange Interview](#)

dd-mm-yyyy --:--  [Confirm](#)



**FSeeker** [❤️](#)

UX Designer

Rejected

[View Profile](#)

**Fig 18: Applicant Status**



Update Profile :

**CompanyName**

**First Name**  **Last Name**  **Phone Number**

**Password**  **Confirm Password**

**Fig 19: Profile Update**



- UPDATE RESUME
- APPLIED JOBS
- CHAT TO RECRUITER

### Recruit LLC

Email	Website	Address	Employer	StartUp Date
Recruiter@gmail.com	https://www.abc.com	123 Maven Street 15233	10000 +	29-11-2022 00:00:00

#### Company Overview :

All Amazon teams and businesses, from Prime delivery to AWS, are guided by four key tenets: customer obsession rather than competitor focus, passion for invention, commitment to operational excellence, and long-term thinking.

#### Company Review :

FSeeker 20-11-2022 14:19:01

Seeker@gmail.com

★★★★★

Really smart people, a lot of opportunity for growth, always encouraged to be innovative, think big, and create

**Fig 20: Seeker Layout - Company Details and Review**

Applied Recruiter List

Recruit LLC

Recent 3 Chat List

Hi! Welcome to our Company!  
20-11-2022 19:25:11

Hello!! Im very happy to chat with you!!  
21-11-2022 19:25:11

SFR recruiter

FSeeker  
20-11-2022 19:25:11  
Hi! Welcome to our Company!


Recruit LLC  
21-11-2022 19:25:11  
Hello!! Im very happy to chat with you!!

✍ Enter Your Discussion

Message

**Fig 21: Chat**

## UX Designer

 UX Designer  
<http://www.recr.com/>  
NY

### Contact

- 🏢 : Recruit LLC
- ✉ : Recruiter@gmail.com
- 🌐 : NY
- 📞 : 234567
- 💰 : \$4000.00 - \$3000.00
- 🕒 : 0 To 1 Years.
- 👥 : 5 Opening
- 📅 : 18-11-2022 upto 21-11-2022

Job Description :

We are looking forward to hire freshers who are interested to work as a Designer in Product development.

Qualification :

**Fig 10: Job Details**

## 6 Design Units Impacts

A website's or an application's design has a big impact on the business. Developers can use this design method to identify the needs of the company. The designing process will be accessed in accordance with the requirements of the business, and the designing approach may be altered to satisfy the client's requirements. The process of receiving input ought to be included in the design. The software's design will benefit from client feedback, which will also help to protect the storage of customer data. The design goal should specify the strategy for maintaining security and spotting fraud.

### 6.1 Functional Area A/Design Unit A

#### 6.1.1 Functional Overview

Functional regions' main objective is to ensure that all crucial business tasks are successfully executed. If the company's goals and objectives are to be achieved, this is essential. The operating functions of the system will benefit from this design unit and workflow technique as they alter the system to suit functional and customer demands. The work process will include authentication, authorization, data changes, user information updates, backup and recovery, and administration.

### **6.1.2 Impacts**

The work process includes authentication, data updates, user information updates, backup and recovery procedures, and administrative operations. Thanks to this design unit and workflow process, the computer's operating functions may modify the system to meet operational and customer demands.

### **6.1.3 Requirements**

- ✓ User authentication is handled by the administrator.
- ✓ To see this application, there is no need for a login or registration process.
- ✓ Interfaces between the outside world and the inside world must be properly set up and reliable.
- ✓ Incorporating a data backup strategy is necessary to protect client data.

## **6.2 Functional Area B/Design Unit B**

### **6.2.1 Functional Overview**

Information about the consumer should be securely kept in the database, which should be updated. Only approved people will be able to access the website, and the database will be password-protected.

### **6.2.2 Impacts**

It will make it possible for the software to run. User must authenticate them before they may attempt to log into the system. Included is the end goal, which might be a real product, an online catalog, or an ordering system. Additionally, authorization levels and approval processes can be combined.

### **6.2.3 Requirements**

Functional needs include things like business rules, administrative procedures, authentication, access levels, external interfaces, reporting, and so on. These specifications aid in ensuring that the customer and the development team are pursuing the same goals.

## **7 Open Issues**

Alerts and notifications are required to be sent to the administrator, seeker, and recruiter in accordance with their respective tasks, such as those involving approval or rejection.

## 8 Acknowledgements

We would like to thank Dr. Dae Wook Kim, our lecturer, for giving us outstanding opportunities to create this fantastic project plan. We were able to perform considerable study thanks to this initiative. Our learning from this Career Engine initiative will be extensive. It helped us increase our research and report-writing ability.

## 9 References

- [1] Kim, D (2022). CPSC 8985 Project Guideline, Governors State University.
- [2] CareerBuilder. n.d. <https://www.careerbuilder.com/> (accessed 12 2022).
- [3] Facebook. n.d. <https://www.facebook.com/>.
- [4] Instagram. n.d. <https://www.instagram.com/>.
- [5] LinkedIn. n.d. <https://www.linkedin.com/feed/> (accessed November 2022).
- [6] MVC. n.d. [https://www.tutorialspoint.com/asp.net\\_mvc/index.htm](https://www.tutorialspoint.com/asp.net_mvc/index.htm).
- [7] "LINQ." In Programming Microsoft®, by Marco Russo & Paolo Pialorsi. 2010.
- [8] Twitter. n.d. <https://twitter.com/>.

## 10 Appendices

```
// GET: HomeController/ Recruiter List
0 references
public IActionResult RecruiterList(GuestUserView guestUser)
{
    guestUser.ListRecentJob = _jobRepo.GetList().Where(j => j.IsApprove == true).OrderByDescending(x => x.Id).To
    guestUser.ListRecentJob.ForEach(j => j.CompanyNameNotMapped = _context.Users.FirstOrDefault(e => e.Id == j.Re
    return View(guestUser);
}

// GET: HomeController/ Job Details Info
0 references
public IActionResult JobDetails(int id)
{
    GuestUserView guestUser = new GuestUserView();
    guestUser.SingleJob = _jobRepo.GetPostJobById(id);
    guestUser.ListRecentJob = _jobRepo.GetList().Where(j => j.IsApprove == true).OrderByDescending(x => x.Id).To
    guestUser.ListRecentJob.ForEach(j =>
    {
        j.SkillNotMapped = _skillRepo.GetSkillInfoById(Convert.ToInt32(j.Skill)).JobSkill;
        j.CareerLevelNotMapped = _careerRepo.GetCareerLevelInfoById(j.CareerLevel).CareerLevel;
        j.CategoryNotMapped = _categoryRepo.GetCategoryInfoById(Convert.ToInt32(j.Category)).Name;
        j.JobTypeNotMapped = _jobTypeRepo.GetJobTypeInfoById(j.JobType).JobType;
        j.CompanyNameNotMapped = _userRepo.GetUserInfoByUserId(j.RecruiterId).CompanyName;
        j.CompanyEmailNotMapped = _userRepo.GetUserInfoByUserId(j.RecruiterId).Email;
    });

    guestUser.ListSkill = _skillRepo.GetList().ToList();
    return View(guestUser);
}
```

Fig 23: Home Controller



```

// POST: RecruiterController/Create Company Update
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public ActionResult UpdateCompany(int? id, UserView updateCompany)
{
    updateCompany.GetCompanyInfo.RecruiterId = this.User.FindFirstValue(ClaimTypes.NameIdentifier);
    if (updateCompany.GetCompanyInfo.CompanyImageNotMapped != null)
    {
        string FilePathDefault = Path.Combine(_env.WebRootPath, "CompanyLogo");
        string FilePath = FilePathDefault;
        if (!Directory.Exists(FilePath))
            Directory.CreateDirectory(FilePath);
        var FileName = updateCompany.GetCompanyInfo.CompanyImageNotMapped.FileName; //await _userManager.
        FilePath = Path.Combine(FilePathDefault, FileName);
        using (FileStream fs = System.IO.File.Create(FilePath))
        {
            updateCompany.GetCompanyInfo.CompanyImageNotMapped.CopyTo(fs);
            updateCompany.GetCompanyInfo.CompanyLogo = FileName;
        }
    }
    _jobRepo.NewUpdateCompanyProfile(updateCompany.GetCompanyInfo);
    return RedirectToAction("JobList");
}

```

**Fig 11: Recruiter Controller**

```

// GET: RecruiterController/Applied Job List
0 references
public async Task<IActionResult> AppliedJobs()
{
    GuestUserView guestUser = new GuestUserView();
    var user = this.User.FindFirstValue(ClaimTypes.NameIdentifier);
    guestUser.ListApplication = User.IsInRole("Seeker") ? _jobRepo.GetAllApplicationList(
        User.IsInRole("Recruiter") ? _jobRepo.GetAllApplicationList().Where(x => x.Re

    if (guestUser.ListApplication.Count > 0 && user != null && User?.Identity?.IsAuthenti
    {
        guestUser.ListApplication.ForEach(a =>
        {
            a.RecruiterNameNotMapped = _userRepo.GetUserInfoByUserId(a.RecruiterId).Compa
            a.JobTitleNotMapped = _jobRepo.GetPostJobById(a.JobId).JobTitle;
            a.SeekerNameNotMapped = _userRepo.GetUserInfoByUserId(a.SeekerId).FirstName;
        });
        return View(guestUser.ListApplication);
    }
}
return View(guestUser);
}
// GET: RecruiterController/Seeker Detail|
0 references
public async Task<IActionResult> SeekerDetail(string id = "") {
    UserView guestUser = new UserView();
    guestUser.GetResumeInfo = _resumeRepo.GetResumeById(id);
    guestUser.ExperienceList = _resumeRepo.GetExperienceList().Where(x => x.SeekerId == i
    guestUser.EducationList = _resumeRepo.GetEducationList().Where(x => x.SeekerId == id)
    guestUser.SkillList = _resumeRepo.GetSkillList().Where(x => x.SeekerId == id).ToList(
    guestUser.SkillList.ForEach(s => s.SkillNotMapped = _resumeRepo.GetSkill(Convert.ToIn
    return View(guestUser); }
}

```

**Fig 12: Recruiter Controller**

```

// GET: AdminController/ User and Job Activation
0 references
public IActionResult Activate(string? UserId, int? PostId)
{
    var status = _context.Users.FirstOrDefault(x => x.Id == UserId); //_userR.UserListById(id);
    if (status != null)
    {
        status.EmailConfirmed = !status.EmailConfirmed;
        _context.Entry(status).State = EntityState.Modified;
        _context.SaveChanges();
        return RedirectToAction("SRList");
    }
    var jobstatus = _context.PostJobInfo.FirstOrDefault(p => p.Id == PostId);
    if (jobstatus != null)
    {
        jobstatus.IsApprove = !jobstatus.IsApprove;
        _context.Entry(jobstatus).State = EntityState.Modified;
        _context.SaveChanges();
        return RedirectToAction("JobList", "Recruiter");
    }
    return RedirectToAction("SRList");
}

```

**Fig 13: Admin Controller**

```
List<Application> IJob.GetAllApplicationList()
```

2 references

```
public string NewUpdateCompanyProfile(CompanyUpdate updateCompany)
{
    if (updateCompany.Id > 0)
        _context.CompanyInfo.Update(updateCompany);
    else
        _context.CompanyInfo.Add(updateCompany);

    _context.SaveChanges();
    return updateCompany.CompanyNameNotMapped + " Modified Successfully";
}
```

2 references

```
public List<CompanyUpdate> GetRecruiterList()
{
    return _context.CompanyInfo.ToList();
}
```

5 references

```
public CompanyUpdate GetCompanyById(string recruiterId)
{
    if (recruiterId != null)
    {
        var company = _context.CompanyInfo.FirstOrDefault(c => c.RecruiterId == recruiterId);
        return company != null ? company : new CompanyUpdate();
    }
    return new CompanyUpdate();
}
```

**Fig 14: Implementation**

2 references

```
public string SignIn(string userName, string password)
{
    var user = _userManager.FindByNameAsync(userName);
    if (user.Result != null)
    {
        var signIn = _signInManager.PasswordSignInAsync(user.Result, password, false, false);
        signIn.Wait();
        return signIn.Result.Succeeded ? "Success" : "Invalid Password ";
    }
    return "Invalid Username";
}
```

1 reference

```
public string UpdateUser(UserDetail userDetails)
{
    var result = _userManager.UpdateAsync(userDetails);
    result.Wait();
    return "Updated Successfully";
}
```

**Fig 15: Implementation**

```
using RStackCareer.Models;

namespace RStackCareer.Domain
{
    10 references
    public interface IUserAuthenticate
    {
        2 references
        string NewUser(UserDetail userDetails);
        1 reference
        string UpdateUser(UserDetail userDetails);
        1 reference
        UserDetails GetUserInfoByUserName(string userName);
        21 references
        UserDetails GetUserInfoById(string Id);
        1 reference
        IEnumerable<UserDetail> GetUserInfo();
        2 references
        string SignIn(string userName, string password);
    }
}
```

**Fig 16: Interface**