

Governors State University

## OPUS Open Portal to University Scholarship

---

All Capstone Projects

Student Capstone Projects

---

Fall 2022

### Career Engine [Jobesy]

Tejaswini Kapa

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

---

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).

# **CAREER ENGINE [JOBESY]**

By

**TEJASWINI KAPA**

Bachelors, N.B.K.R.I.S.T., 2020

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" software is focused with job posting and registration. We may also use this application to look for employment openings in a certain location, and the name of the firm will be presented. Timings and other details are also available online. It will assist you in applying for a better job than the one you now have. Position seekers may select and register for positions that interest them, and the information received from the recruiter providing the job will be shown on the job details site. We chose the MERN stack to develop our "JobEsy" career engine website, which comprises Express.js, React.js, and Node.js, and I utilize MySQL for the database. The administrator may use Career Engine to get into the system, authorize a vacancy, and post it on the web. Job searchers and recruiters are managed by the administration. Job searchers may also log in and search for open opportunities. They can apply for work by presenting their stuff, such as a resume. Our career engine online application will be responsively maintained. This will adjust web pages to fit the size of the device's window. On this website, all job searchers, recruiters, and administrators have their own logins.

# 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 .....	2
2.2	Application Information flows.....	3
2.3	Interactions with other Projects (if Any) .....	<b>Error! Bookmark not defined.</b>
2.4	Interactions with other Applications .....	5
2.5	Capabilities .....	5
2.6	Risk Assessment and Management.....	5
<b>3</b>	<b><i>Project Requirements</i></b> .....	5
3.1	Identification of Requirements .....	5
3.2	Operations, Administration, Maintenance and Provisioning (OAM&P).....	5
3.3	Security and Fraud Prevention.....	6
3.4	Release and Transition Plan.....	6
<b>4</b>	<b><i>Project Design Description</i></b> .....	6
<b>5</b>	<b><i>Internal/external Interface Impacts and Specification</i></b> .....	16
<b>6</b>	<b><i>Design Units Impacts</i></b> .....	19
6.1	Functional Area A/Design Unit A .....	19
6.1.1	<b><i>Functional Overview</i></b> .....	19
6.1.2	<b><i>Impacts</i></b> .....	19
6.1.3	<b><i>Requirements</i></b> .....	19
6.2	Functional Area B/Design Unit B.....	<b>Error! Bookmark not defined.</b>
6.2.1	<b><i>Functional Overview</i></b> .....	<b>Error! Bookmark not defined.</b>
6.2.2	<b><i>Impacts</i></b> .....	<b>Error! Bookmark not defined.</b>
6.2.3	<b><i>Requirements</i></b> .....	<b>Error! Bookmark not defined.</b>
<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***

A portal for job searching is Career Engine named as JobEsy. Connecting job seekers and employers is made simple, quick, easy, and well-organized by this platform. Candidates looking for work may register, browse for vacancies depending on their qualifications, and instantly apply for those jobs using this platform. The personal information and skill sets that job seekers provided upon registration can also be changed. On the other hand, employers may sign up on this site and post job opportunities, enabling them to find competent people for their available positions. The employer has the option to review the job applications and take the necessary action. The admin must approve both business registration and company job ads before they may be used to join the job search service.

Some of the current and traditional methods of recruiting include advertisements in newspapers, posters, TVs, numerous job fairs, college career fairs, and other traditional forms of recruiting. But these processes are both costly and time-consuming. Giving out paper printed resumes, keeping track of them, handling and processing them, and then contacting the preferred candidate to be called for an interview all sound like a lot of work. With the growth of the Internet and the quick advancement of technology, these efforts can be reduced. The internet job search tool that is available at this time will save you both time and money on advertisements. The entire application or job search process is accelerated. Manual processes are replaced by automated ones.

Because of job search portals, the custom of paper resumes is being supplanted by online resumes. Additionally, these resumes are stored for later use in corporate databases. In a couple of seconds, employers and candidates may connect. Another advantage is that after registering and submitting an application, a candidate's information is saved in the company database for use in both available positions now and in the future. We created two layouts for the Angular module, one for the administrator and the other for the user. Non-registered users can also access the Guest Pages, search job, employer list, job list, and feature job list. As an administrator, you have the ability to see and manage recruiters, job seekers, and available opportunities. Additionally, the administrator has the authority to specify a skill known as a Job Application Category. Password changes are available to all logged-in users. A job can be posted, and recruiters can browse a list of applicants, chat with job seekers, and read their reviews. They can also update their company profile. The feature may be used by those who are looking for work.

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

There are already a number of career portal websites available; I decided to link our application with LinkedIn. Our application performs three essential tasks and may be set up in many ways. It reduces the amount of space required while also saving time.

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

The jobs in that category will be shown when you choose one. We utilized the easy search feature on the visitor page to find a job, and then I used advanced search, just like on the replica website, to look for a few different combinations.

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

- A solid understanding of computers is necessary for the user.
- The computer has to be capable of accessing the internet and serving as an Internet server.
- Since the user interface will be in English, they must be able to communicate in English.

## ***1.4 Future Enhancements***

There have been several improvements and new features added to this program. This application may be improved to automatically schedule interviews based on whether or not a resume is accepted. Companies have the option to automatically delete jobs when the employment availability period has passed. A job suggestion system based on users' frequent search results may be included in the software. Additionally, candidates may sign up for email notifications about job vacancies through the platform. The User functionality might be improved to enable users to upload numerous papers, save jobs, and then apply at a later time. To make it more aesthetically pleasing and user-friendly, the User Interface may be enhanced. The UI might be improved with additional AngularJS wizardry. There can be a feedback or review section on the application. Furthermore, unlike the current job description page, the user may click on the job description to view it on another website.

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

- HTML (Hyper Text Markup Language)
- NODES (Network Operation And Design Engineering System)
- REACT (Radio Emergency Associated Communications Teams)
- API (Application Programming Interface)
- NPM (Node Package Manager)
- HTTP (Hyper Text Transfer Protocol)

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

This curriculum's technical components were all created with the goal of assisting students in becoming acquainted with the most recent cutting-edge technology. The JavaScript frameworks Node.js and react.js, as well as interacting with them, were exposed to me greatly through this application. A client-side scripting language called JavaScript expands and improves the functionality of client-side coding in conventional web applications. In contrast, the server-side programming language NodeJS. It is JavaScript with the additional capability of being utilized outside of the browser as a server-side programming language. I used NodeJS and its well-known module Express to offer server-side functionality for my project. Node.js [2] is built on Google's V8 engine. A promise-based object relational model, a NodeJS web server, and React.js [3] are used in this Representational State Transfer API on the front end, together with ES6 JavaScript, HTML5, CSS3, and jQuery on the server. The use of AngularJS is used for all validations. I worked with MySQL as my database language.

### ***2.1 Application Architecture***

The server receives a request from the user. This request is added to the backlog of others. Each request is assigned to a thread in the thread pool by a single thread using the event loop mechanism. The input/output activities that a thread in the thread pool is doing, for example, determines how this delegation mechanism is used. When a request is assigned to a thread in the thread pool asynchronously, the thread processes other requests while it waits for this one to complete. Once this request is complete, it either returns a success or an error. A promise or callback system is another name for this type of asynchronous processing. NodeJS has the single threading feature as a major benefit.

Web applications developed with React Components are referred to as single page apps. We don't have to create any crucial codes or logic, and I don't have to search for them either. React refers to the mechanism as Dependency Injection. React has built-in validators, but I can also make our own [6]. When a user interacts with the interface, the Components form control highlights the field and displays a warning if the input is incorrect. A three-tier architecture for software design is called MVC.

All of the data from the database is handled in the model. The view is the user interface, sometimes referred to as the front end. All of the business logic, or the information that links the model and the view, is kept in the controller. The whole API of a node is event-driven and asynchronous. The JS library is intended to not block. The server contacts an API and does not wait for a response before moving on to the next API. A promise, a callback mechanism that notifies the server of the result or error, is sent to it after the request has been fulfilled. NodeJS is speedy as a result. NodeJS employs a single threaded strategy similar to event looping. It is quite adaptive. An open source JavaScript-based structural web framework called Angular JS adheres to the MVC paradigm. Without having to constantly refresh it, I can use components to load an HTML page and dynamically update it.

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

Our application used Job-Portal and JobsyApi. To retrieve values from databases, all API routes obtain and post the database. Components that are utilized to display UI design Only administrators have the power to make categories. Additionally, the recruiter, seeker, and job posting can only be authorized by the administrator. The essential aspect is that the feature job, which selects which work will display on the top page, is only open to administrators. After the admin has created the category, only the seeker and recruiter may edit their profiles. Anyone may see the website's jobs and recruiter information without registering, but in order to apply for a job or post a job, a user has to register. Positions can be submitted by recruiters, and the admin must approve them before users or job seekers can access them. If a job seeker decides to apply for a position, a toast notice stating that they have done so or that they have already done so and cannot apply again will be displayed to them. Only the candidate and the recruiter will have access to one another's communications. The seeker is just given the ability to evaluate the company. On our home page, you could search for jobs and recruiters. We also had an advanced search tool. We could learn the task from user preferences in this feature.

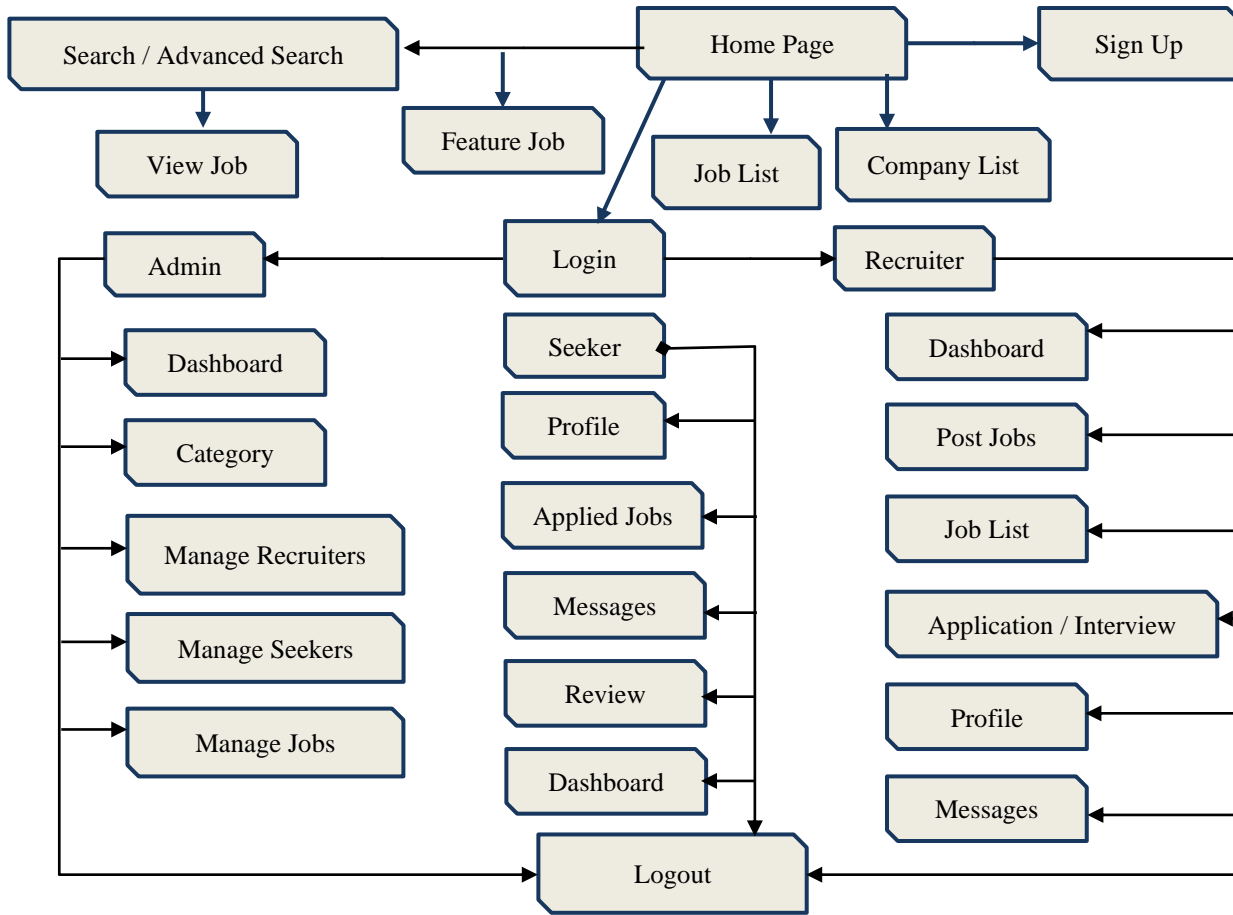


Fig 1: Application Work Flow

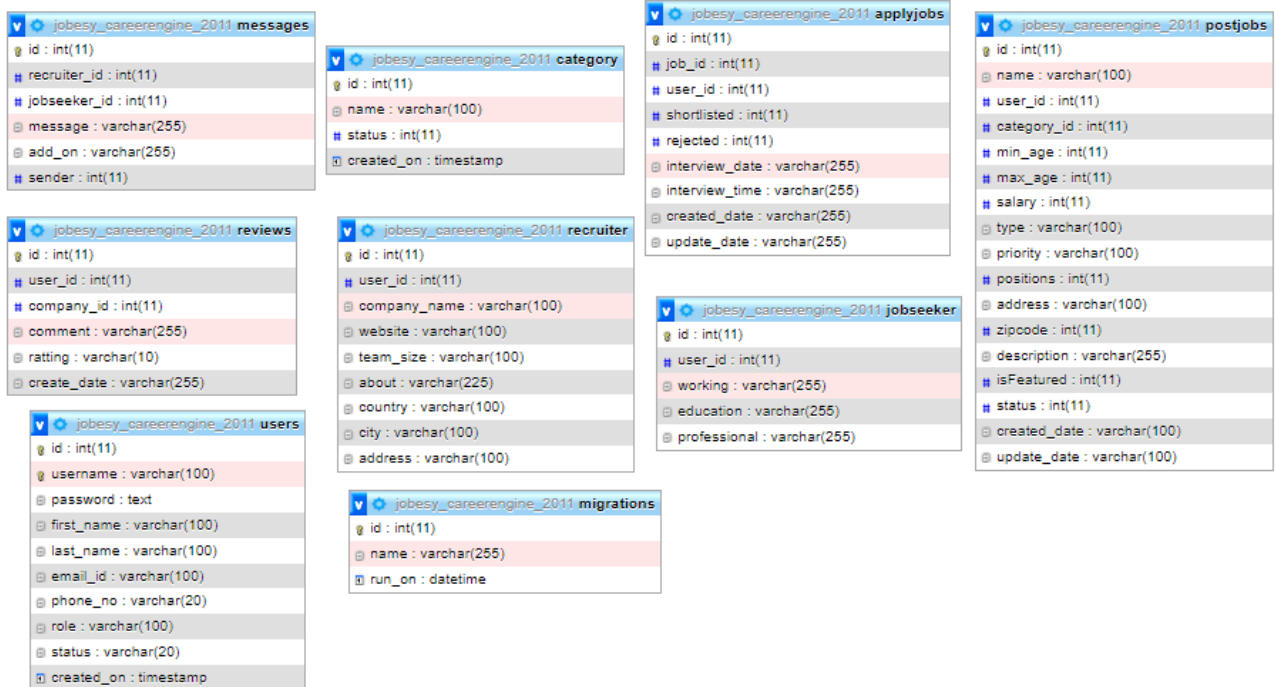


Fig 2: JobEsy Database Diagram



### 2.3 *Interactions with other Applications*

- When a candidate submits an application for a particular post, the recruiter is informed.
- Once the post has been listed, the administrator will be contacted.
- The candidate will be told if their application has been accepted or denied by the recruiter.
- An applicant gets an alert notification whenever a recruiter starts a conversation with them.
- A review from a candidate will notify the recruiter.
- The candidate is informed when the administrator activates or deactivates the post.

### 2.4 *Capabilities*

There are a variety of search options depending on the keywords. There are different search possibilities available depending on where you are. Searching for employment according to category and interacting with employers and job seekers. a user interface that is easy to use and customize and has an appealing visual appearance. The capacity to search for a job using a number of approaches; Modern features that are also user-friendly.

### 2.5 *Risk Assessment and Management*

We provide this application to one Admin, who is in charge of everything in it and has access to it. Every job seeker, recruiter, and approval of a position is kept track of by the admin, which is a hassle. Our long-term strategy will be modified. In addition, I offered a single recruiter login for a single organization, whereas other companies require several logins for interviewers, HR, and so forth. The design had to be altered to account for the fact that this application only required one login to access the business, whereas typical organizations require numerous logins.

## 3 *Project Requirements*

### 3.1 *Identification of Requirements*

<JobEsy-CPSC8985-1 UserInterface-capability-“JE898501”>

**Enhance UI** - A better user interface is required.

<JobEsy-CPSC8985-2 Admin-capability-“JE898502”>

**Auto Approval** - For the addition of a new job, recruiter, or seeker, admin approval is required. We make advantage of auto approval to save the administrative load.

< JobEsy-CPSC8985-3 Employer-capability-“ JE898503”>

**Recruiter Subscription** - Only recruiters will be able to submit unwanted jobs since I will use the subscription to publicise.

< JobEsy-CPSC8985-4 Seeker-capability-“ JE898504”>

**Seeker Resume Information** - To make the candidate selection process easier for the recruiter, we'll use additional information.

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

- The capacity to monitor user progress and impose access restrictions.

- The website's job and seeker search functionality should be straightforward and secure.
- Pages load more quickly for complex web apps like this when large file sizes are reduced.
- You do not need to log in or register as a user in order to access the website. Guests can access the general pages.
- The data acquired via the search option have to be kept in the company's or job seeker's database.
- This software was developed to help job searchers locate vacant vacancies.
- This application is trustworthy and easy to use.
- This software verifies that the settings are accurate.

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

The construction of a security and fraud detection method may be accomplished through security and fraud prevention planning. Real-time alerts and warnings that bring potential fraud cases to your attention are part of the planning process for preventing fraud. Admin has been assigned tasks with the appropriate roles. The application has to be updated often. All user information is saved throughout the necessary registration process.

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

The website development process has to be finished and tested before it can be pushed to the live server. The website shouldn't be used for employment counselling before flaws are fixed. Upon booting, the data storage process will be launched, and the maintenance process will be used to maintain the application's continuing operation.

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

The website's management is under the purview of the administrator. He decides whether to approve or disapprove job postings and whether to allow a business to register on his website. Only admin selects which feature category appears on the front page.

**Companies** - A company can register with the site and log in after receiving administrator approval. New employment positions may also be advertised by them. They could check to see whether any people have applied for the positions they've posted. They can change the information on their corporate profile. After the candidates have been shortlisted, whether the individual has been chosen, rejected, or shortlisted again, the interview for that particular applicant may be scheduled. The message menu can be used by the recruiter to get in touch with the applicant or seeker.

**Job Seekers** - Interested parties can register and logged-in candidates. They may search for opportunities that match their requirements after logging in, apply for them, and submit their resumes. Additionally, whenever they wish, they may update their data. They can read the job listings and company profiles, but job seekers are only able to rate the businesses. Additionally, job seekers begin to contact recruiters. Despite only intending to apply once, the candidate submitted two applications. Then a warning notice such as "already applied" appeared. In their dashboard, job seekers may also see the status of their applications.

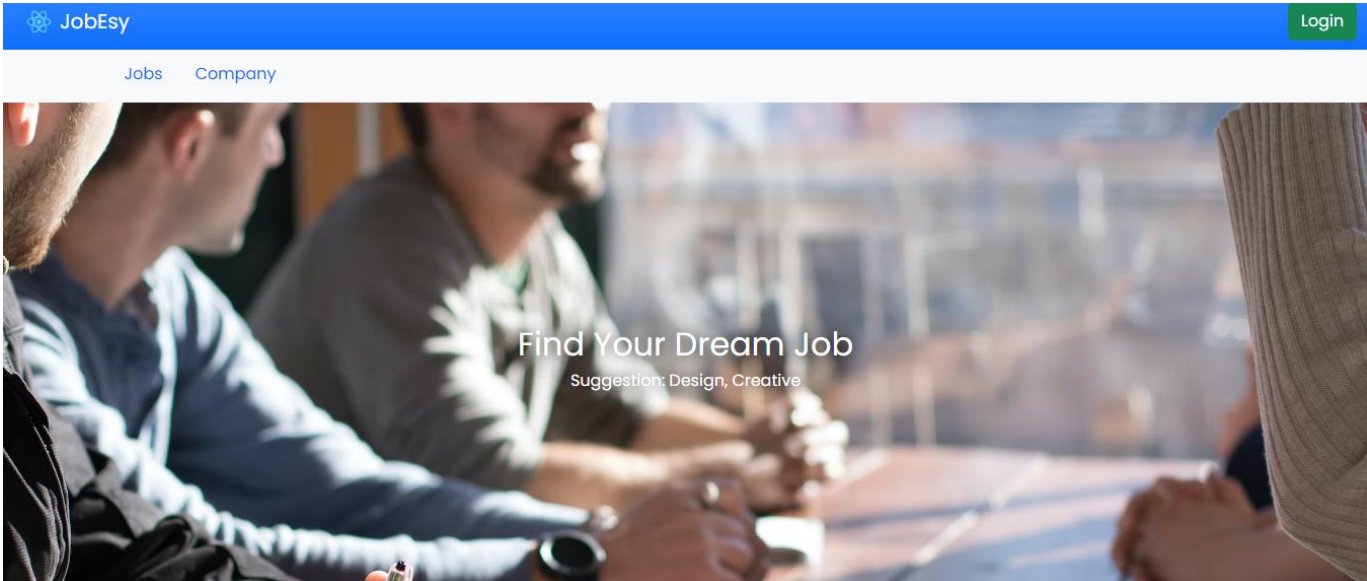


Fig 3: Main Page

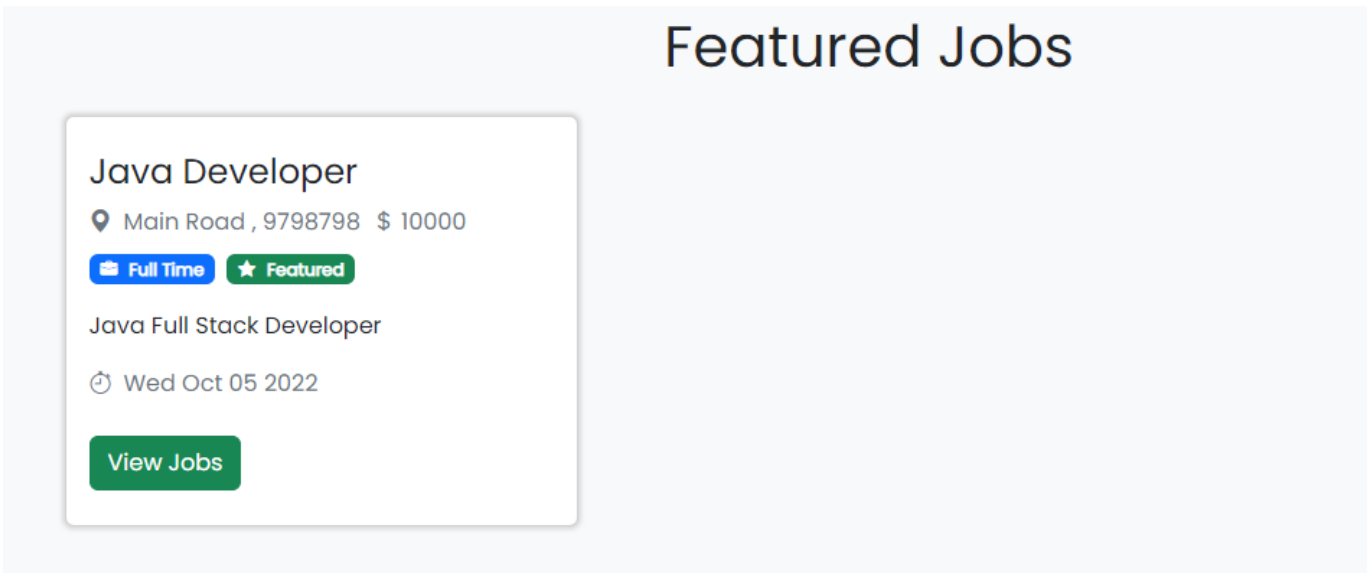
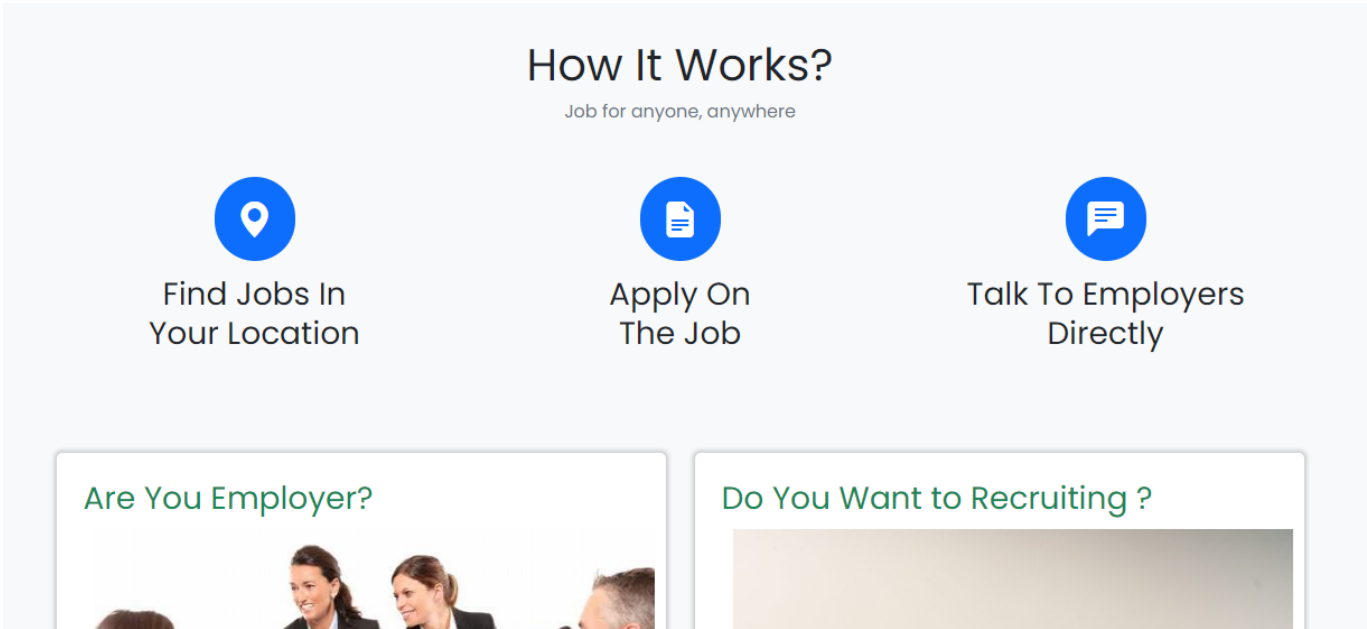
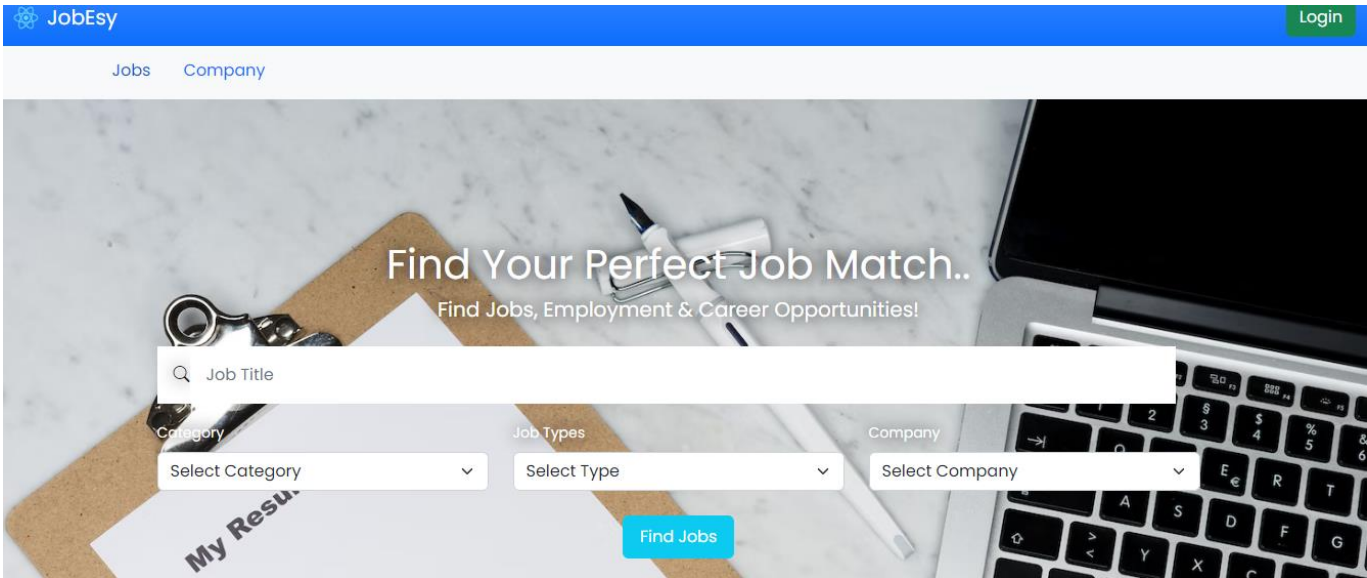


Fig 4: Featured Jobs



**Fig 5 : Main Page**



**Fig 6: Search Page**

## Search result

Java Developer

📍 Main Road , 9798798 \$ 10000

🕒 Full Time ★ Featured

View Jobs

Java Full Stack Developer

🕒 Wed Oct 05 2022

Android Developer

📍 New Road , 2223 \$ 13000

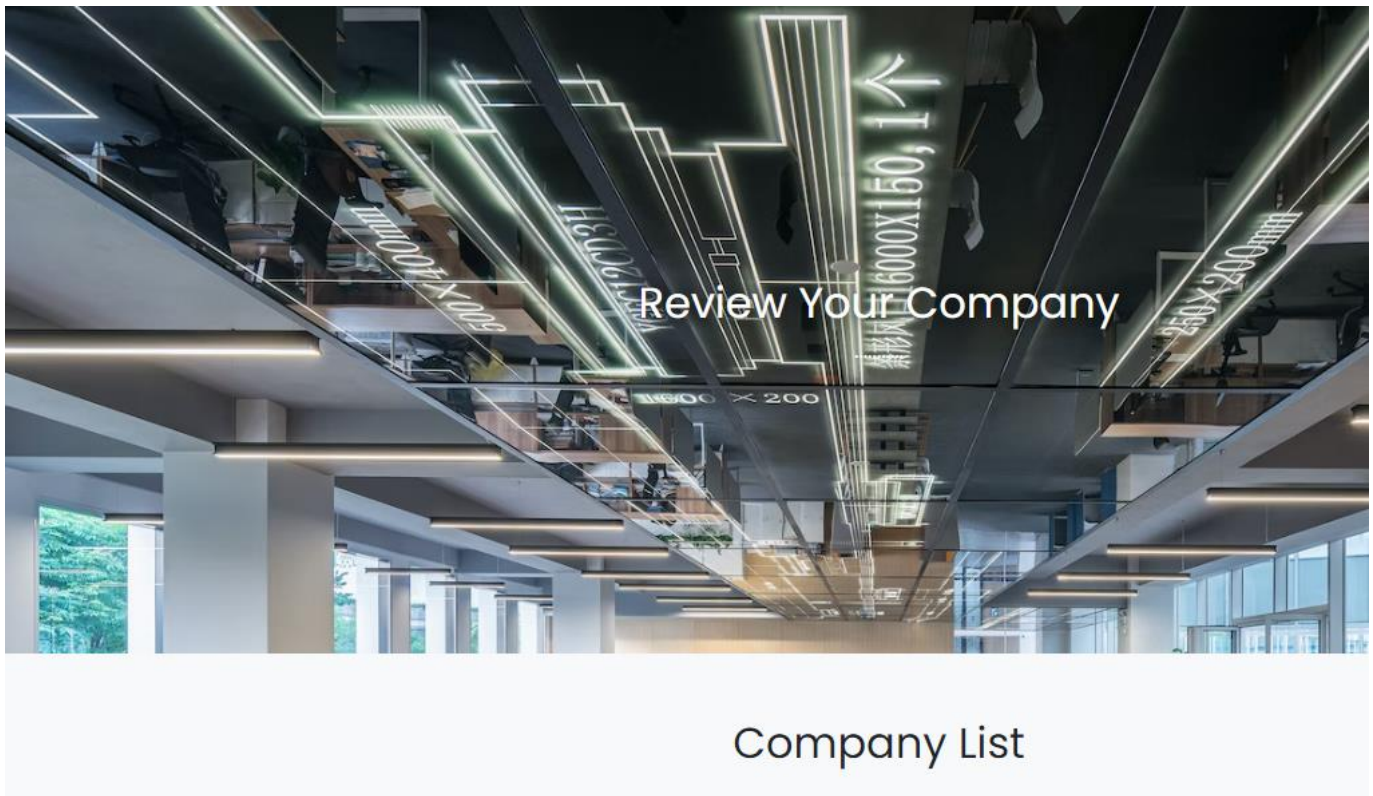
🕒 Full Time

View Jobs

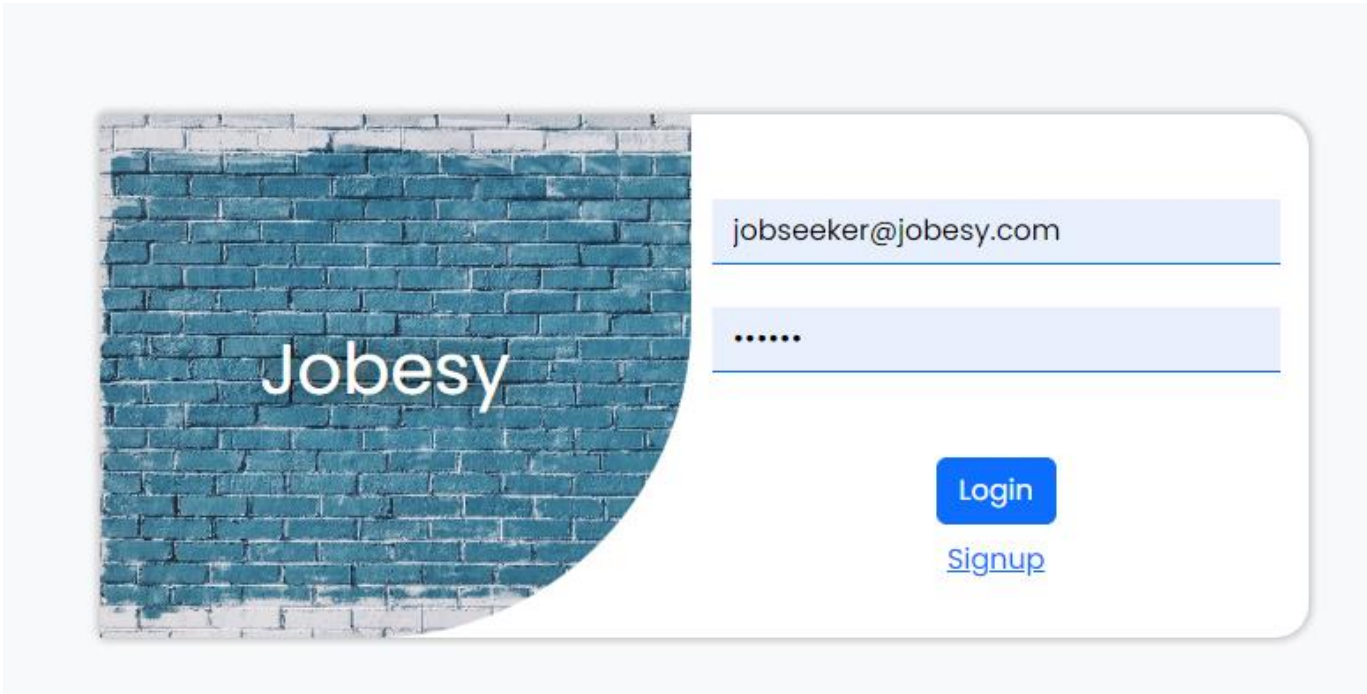
Android Developer

🕒 Wed Oct 05 2022

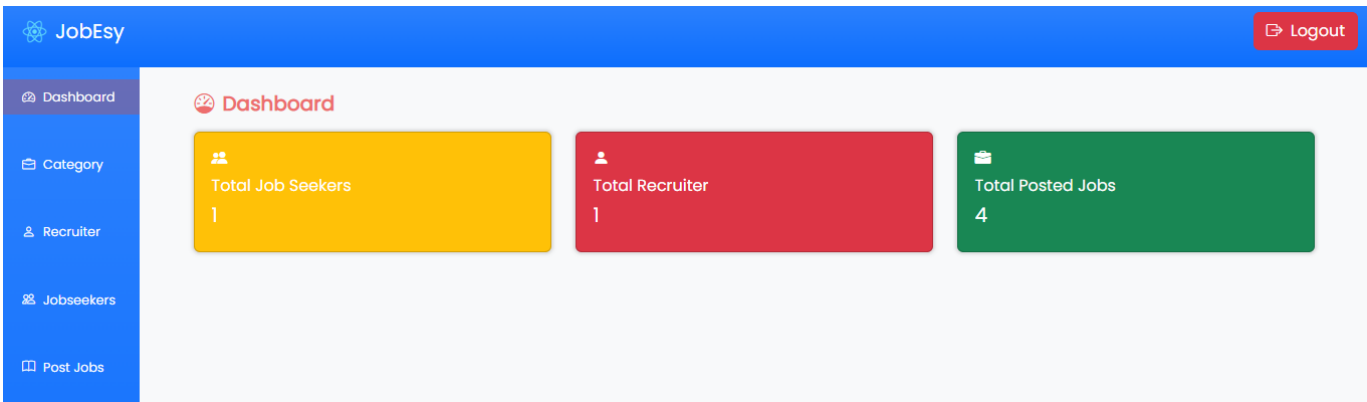
**Fig 7: Search Result**



**Fig 8: Company List**




**Fig 9: Login Page**



**Fig 10: Admin Dashboard**


TITLE	ACTION
Java Full Stack Developer	<input type="button" value="Enabled"/>
Android Developer	<input type="button" value="Enabled"/>
IOS Developer	<input type="button" value="Enabled"/>
UI Designer	<input type="button" value="Enabled"/>

**Fig 11: Category List**

 **Recruiter**

First Name	Last Name	Email	Phone	Status
Google	Inc	recruiter@jobesy.com	976543210	<input type="button" value="Active"/>

**Fig 12: Recruiter List**

 **Jobseekers**

First Name	Last Name	Email	Phone	Status
Vijaya	Lakshmi	jobseeker@jobesy.com	9988776655	<input type="button" value="Active"/>

**Fig 23: Job Seekers List**

**Post Jobs List**

Recruiter Name	Category Name	Job Name	Featured	Status
Google Inc	Java Full Stack Developer	Java Developer	Active	Active
Google Inc	Android Developer	Android Developer	Inactive	Active
Google Inc	IOS Developer	MERN Stack developer	Inactive	Active
Google Inc	UX Designer	VFX Designer	Active	Inactive

**Fig 34: Post Jobs List**

**JobEsy**

Dashboard

Post Jobs

Posted Jobs

Applications

Profile

**Dashboard**

Total Posted Jobs  
4

Total Application Jobs  
4

**Fig 45: Recruiter Dashboard**

**JobEsy** Logout

Dashboard

Post Jobs

Posted Jobs

Applications

Profile

Messages

Job Title  Select Category

Minimum Age of Expeirnce  Maximum Age of Expeirnce  Salary

Type  Priority  Positions

Address  Zip Code

**Fig 56: Post Jobs**



Job Title	Status	Action
<b>Java Developer</b> Main Road , 9798798 \$ 10000	Active	<a href="#">Edit</a>
<b>Android Developer</b> New Road , 2223 \$ 13000	Active	<a href="#">Edit</a>
<b>MERN Stack developer</b> asdasdasd , 123123 \$ 213123	Active	<a href="#">Edit</a>

**Fig 67: Job List**

Job Title				
<b>Java Developer</b> Main Road , 9798798 \$ 10000 <b>Interview Scheduled on 2022-11-23 at 10:11</b>	<a href="#">View Profile</a>	<a href="#">Schedule Interview</a>	Shortlisted	<a href="#">Reject</a>
<b>Android Developer</b> New Road , 2223 \$ 13000	<a href="#">View Profile</a>		Rejected	Rejected
<b>MERN Stack developer</b> asdasdasd , 123123 \$ 213123	<a href="#">View Profile</a>		<a href="#">Approve</a>	<a href="#">Reject</a>

**Fig 78: All Applications**

**Profile**

**My Profile**

First Name:  Last Name:

Email:  Phone:

**Company Details**

Company Name:  Website:  Team Size:

**Fig 89: Recruiter Profile**

**Messages**

**Chats**

Vijaya Lakshmi

Hai There !

Sun Oct 30 2022

**Fig 20: Chat**

**Dashboard**

Total Applied Jobs: 5

Total Shortlisted Jobs: 2

Total Rejected Jobs: 1

**Fig 29: Seeker Dashboard**

Dashboard

**Profile**

Applied Jobs

Messages

### My Profile

First Name	Last Name
<input type="text" value="Vijaya"/>	<input type="text" value="Lakshmi"/>
Email	Phone
<input type="text" value="jobseeker@jobesy.com"/>	<input type="text" value="9988776655"/>

---

### Working History

Working History

**Fig 102: Seeker Profile**

### Applied Jobs

Job Title	Status
<p><b>Java Developer</b></p> <p>📍 Main Road , 9798798 \$ 10000</p> <p><span style="background-color: #007bff; color: white; padding: 2px 5px;">Full Time</span> <span style="background-color: #28a745; color: white; padding: 2px 5px;">★ Featured</span></p> <p><span style="color: red; font-weight: bold;">Interview Scheduled on 2022-11-23 at 10:11</span></p>	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="background-color: #28a745; color: white; padding: 5px 10px; border-radius: 5px;">View Jobs</div> <div style="background-color: #28a745; color: white; padding: 5px 10px; border-radius: 5px;">Send Message</div> </div> <p style="color: #6c757d; font-weight: bold;">Active</p>
<p><b>Android Developer</b></p> <p>📍 New Road , 2223 \$ 13000</p> <p><span style="background-color: #007bff; color: white; padding: 2px 5px;">Full Time</span></p>	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="background-color: #28a745; color: white; padding: 5px 10px; border-radius: 5px;">View Jobs</div> <div style="background-color: #28a745; color: white; padding: 5px 10px; border-radius: 5px;">Send Message</div> </div> <p style="color: red; font-weight: bold;">Rejected</p>

**Fig 113: Applied Jobs**


### Messages

Chats

---

Google Inc

---



No Messages Found

**Fig 124: Chat From Seeker**

## 5 Internal/external Interface Impacts and Specification

The first and most crucial stage of the software development life cycle is requirement analysis (SDLC). The entire goal is to identify the functional and nonfunctional requirements for the product and to assess the viability of those requirements. We promise that by gathering requirements, project goals and objectives will be established much more quickly. A solid understanding of the requirements is necessary for the effective development of software. No matter how hard I work, if I omit this step, I will never get the desired outcome. This is crucial since the intended result cannot be achieved without knowing the precise requirements. The defining of requirements is a crucial phase in the life cycle of software development. Although a number of software and hardware specifications may be used to produce this application, the software and hardware requirements that I used to develop this job search website are stated below:

### Software Requirements:

IDE: Visual Studio Code

Operating System: Windows 10

Frameworks: ReactJS

Components: HTML5, CSS3 [1],

ReactJS [5], JavaScript, and jQuery

Database: MySQL

### System Requirements

Intel Core i5 processor

2.50 GHz Processor Speed

8 GB RAM

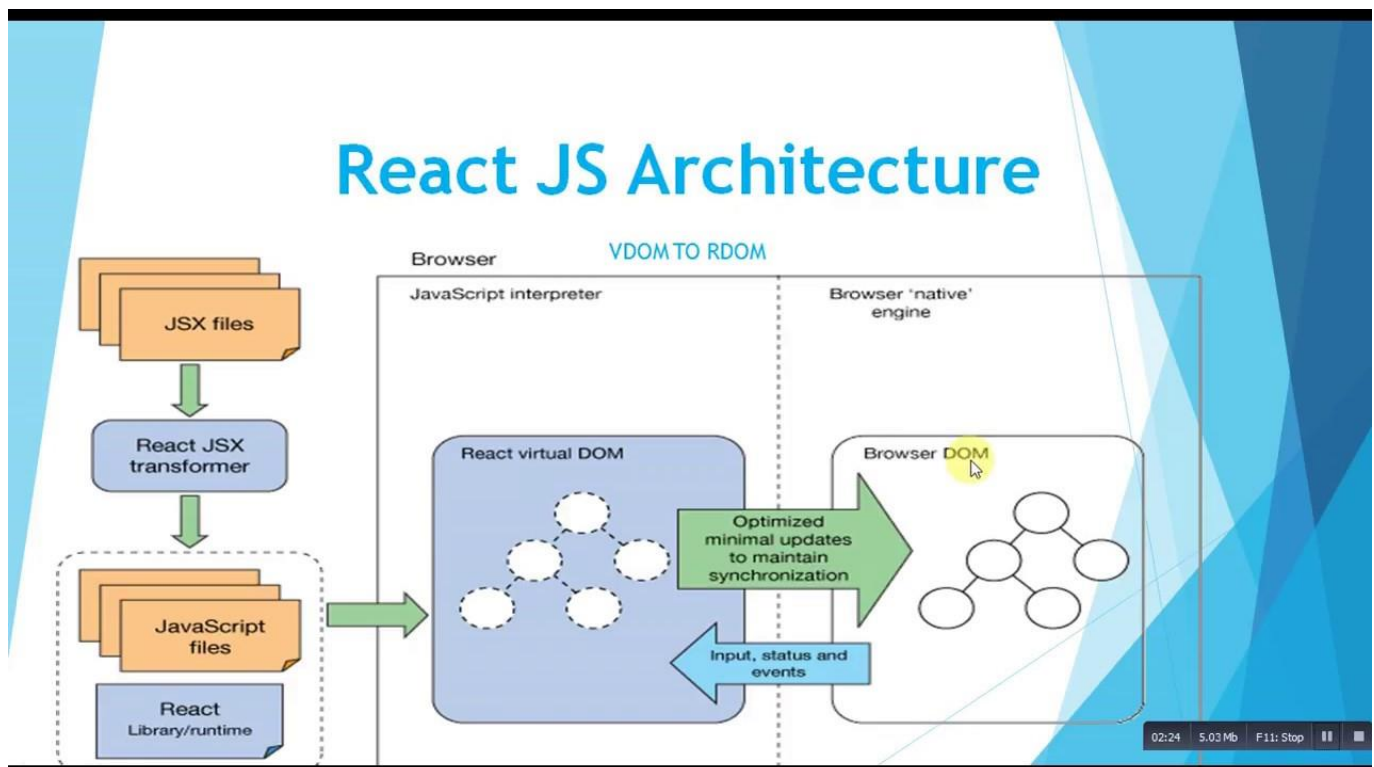


Fig 25: React JS Architecture [4]

## **6 Design Units Impacts**

We view recruiters and job seekers as clients. The admin will function without relying on the client in the event of a problem with the client, such as a server outage or anything similar. The applications are for Employer, Jobseeker, and Admin.

### **6.1 Functional Area A/Design Unit A**

#### **6.1.1 Functional Overview**

The suggested system features the following capabilities:

- Job Seeker Sign In - In order to apply for a job, a Seeker must first log in.
- Job Seeker Sign Up - To access the complete website, a seeker must first register, which is approved by the administrator.
- A job seeker can browse all open positions.
- Job seekers may use advanced search to narrow their results by category.
- A job applicant can change their personal information, including their education, experience, and abilities.
- A job applicant may submit more than one application.
- A job seeker may browse all open positions.
- If a job applicant forgets their password, they can retrieve it.
- Job seekers can evaluate the businesses.
- After shortlisting, the job seeker can speak with the recruiter.
- Businesses can sign up for the portal.
- Companies may advertise positions if the administrator accepts their registration.
- Applications are visible to employers.
- All authorized positions are visible to employers.
- The application profile is accessible to employers.
- Admin has access to every job.
- The company's positions are approved by the administrator.
- Admins can label jobs as features.

#### **6.1.2 Impacts**

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

#### **6.1.3 Requirements**

It is not necessary to log in or register to use this web application. The administrator should monitor the hiring process for both the company and the candidate. The internal and external interfaces need to be properly configured and reliable. A procedure for data backup should be used to secure client data.

## **7 Open Issues**

- Put your resume and a photo online.

- Messages of Warning.

## 8 Acknowledgements

I would want to thank my professor, Dr. Dae Wook Kim, for giving me the outstanding opportunities to create this excellent plan on the topic. This project helped us prepare a significant quantity of research. This project will give us a lot to learn. It helped us increase our research and report-writing ability. I also want to thank my parents and classmates for their assistance in getting the assignment finished successfully.

## 9 References

- [1] Cascading Style Sheets. (2016). Retrieved from W3C: <https://www.npmjs.com/package/w3c-css>
- [2] NodeJS - Server Side JS. (2017, 10 10). Retrieved from <https://www.slideshare.net/ganeshkondal/nodejs-server-side-js>
- [3] React - Ng Book - Murray N. (n.d.). Retrieved from [https://www.academia.edu/49151733/Murray\\_N\\_ng\\_book\\_The\\_Complete\\_Book\\_on\\_Angular](https://www.academia.edu/49151733/Murray_N_ng_book_The_Complete_Book_on_Angular)
- [4] React JS Architecture. (n.d.). Retrieved 11 2022, from <https://developer.ibm.com/tutorials/wa-react-intro/>
- [5] React tutorials. (n.d.). Retrieved 2022, from W3Schools: <https://www.w3schools.com/react/default.aspReactJs>
- [6] React-Componentet. (n.d.). Retrieved 12 2022, from <https://reactjs.org/>

## 10 Appendices

```
46 //category add & update
47 router.post('/category', (req, res, next) => {
48   if (
49     !req.headers.authorization ||
50     !req.headers.authorization.startsWith('Bearer') ||
51     !req.headers.authorization.split(' ')[1]
52   ) {
53     return res.status(401).json({
54       message: "Unauthorized Access",
55     });
56   }
57   const theToken = req.headers.authorization.split(' ')[1];
58   const decoded = jwt.verify(theToken, 'jobesy');
59   if (!req.body.id) {
60     db.query(
61       `INSERT INTO category (name) VALUES ('${req.body.name}')`,
62       (err, result) => {
63         if (err) {
64           throw err;
65           return res.status(400).send({
66             msg: err
67           });
68         }
69         return res.status(201).send({
```

Fig 136: Category Add / Update Backend Code

```

//write comment on company
router.post('/write-comment', (req, res, next) => {
  if (
    !req.headers.authorization ||
    !req.headers.authorization.startsWith('Bearer') ||
    !req.headers.authorization.split(' ')[1]
  ) {
    return res.status(401).json({
      message: "Unauthorized Access",
    });
  }
  const theToken = req.headers.authorization.split(' ')[1];
  const decoded = jwt.verify(theToken, 'jobesy');
  db.query(
    `INSERT INTO reviews (user_id, company_id, comment, rating, create_date) VALUES ('${req.body.user_
    (err, result) => {
      if (err) {
        throw err;
        return res.status(400).send({
          msg: err
        });
      }
      return res.status(201).send({
        msg: 'Review Send'
      });
    }
  });
});

```

Fig 147: Review Code

```

//get posted jobs method
router.get('/posted-jobs', (req, res, next) => {
  if (
    !req.headers.authorization ||
    !req.headers.authorization.startsWith('Bearer') ||
    !req.headers.authorization.split(' ')[1]
  ) {
    return res.status(401).json({
      message: "Unauthorized Access",
    });
  }
  const theToken = req.headers.authorization.split(' ')[1];
  const decoded = jwt.verify(theToken, 'jobesy');
  db.query(`SELECT
  postjobs.*
  FROM recruiter INNER JOIN postjobs ON postjobs.user_id = recruiter.id where recruiter.user_id =? `,
    if (error) throw error;
    return res.send(results);
  });
});

```

Fig 158: Job List Code

```
JS dbConfig.js X JS 20221120081807-insertapplyjobs.js {} database.json
JobesyAPI > JS dbConfig.js > ...
1 var mysql = require('mysql')
2 var db = mysql.createConnection({
3   host: 'localhost',
4   user: 'root',
5   password: '',
6   database: 'jobesy_careerengine'
7 });
8
9 module.exports = db;
```

Fig 169: Db Configuration

```
app.use(cors());

app.use('/api', indexController);
app.use('/api/admin', adminController);
app.use('/api/jobseeker', jobseekerController);
app.use('/api/recruiter', recruiterController);

app.use((err, req, res, next) => {
  err.statusCode = err.statusCode || 500;
  err.message = err.message || "Internal Server Error";
  res.status(err.statusCode).json({
    message: err.message,
  });
});

app.listen(3000, () => console.log('Server is running on port 3000'));
```

Fig 30: Server Port Configuration



```

//get post jobs method
router.get('/get-postjobs', (req, res, next) => {
  db.query(`SELECT
    category.id AS id,
    category.name AS category_name,
    postjobs.*
  FROM postjobs INNER JOIN  category ON postjobs.category_id=category.id where postjobs.status = 1`,
    (error) throw error;
    return res.send(results);
  });
});

//get featured posted jobs method
router.get('/get-postjobs-featured', (req, res, next) => {
  db.query(`SELECT
    category.id AS id,
    category.name AS category_name,
    postjobs.*
  FROM postjobs INNER JOIN  category ON postjobs.category_id=category.id where postjobs.status = 1 AND
    (error) throw error;
    return res.send(results);
  });
});
});

```

**Fig 31: API Controller Code**

<pre> {} package-lock.json {} package.json Job-Portal   &gt; node_modules   &gt; public   &gt; src     &gt; components       &gt; admin       &gt; Homee       &gt; jobseekers       &gt; recruiters         JS applications.js         JS dasborad.js         JS editPostedJob.js         JS postedjobs.js         JS postjobs.js         JS profileRecruiter.js         JS recruiterMessages.js         JS viewCompany.js         JS viewprofile.js       &gt; signup     JS header.js     JS sidebar.js   &gt; images   &gt; route </pre>	<pre> 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 </pre> <pre>     },     const profileFetch = async () =&gt; {       //get recruiter profile method       const { data: res } = await api.get('recruiter/get-profile', { headers });       setUserData(res);       console.log(res);     }      const setProfileData = async () =&gt; {       setUserid(userData.user_id)       setfirstname(userData.first_name)       setlastname(userData.last_name)       setphone(userData.phone_no)       setCompanyName(userData.company_name)       setemail(userData.email_id)       setWebsite(userData.website)       setTeamSize(userData.team_size)       setAboutCompany(userData.about)       setCountry(userData.country)       setCity(userData.city)       setAddress(userData.address)     }      const saveInfo = async (e) =&gt; {       let data;       if(userid === undefined){ </pre>
--	---

**Fig 32: Profile Recruiter Code**

```

103 {} package-lock.json
104 {} package.json
105 > Job-Portal
106 > node_modules
107 > public
108 > src
109 > components
110 > admin
111 > Homee
112 > jobseekers
113 > recruiters
114 JS applications.js
115 JS dasborad.js
116 JS editPostedJobs.js
117 JS postedjobs.js
118 JS postjobs.js
119 JS profileRecruiter.js
120 JS recruiterMessages.js
121 JS viewCompany.js
122 JS viewprofile.js
123 > signup
124 JS header.js
125 JS sidebar.js
126 > images
127 <>
128 <h5 className="my-3 page-title fw-bold"><i className="bi bi-person-video2 me-2"></i>Profile</h5>
129 <div className="row bg-white box-s br-15 pb-3 mt-5 px-3">
130 <h6 className="col-12 text-success my-3 fs-bold">My Profile</h6>
131 <div className="mb-4 col-6">
132 <label className="form-label required">First Name</label>
133 <input type="text" className="form-control" value={firstname} onChange={e => setfirstt
134 </div>
135 <div className="mb-4 col-6">
136 <label className="form-label">Last Name</label>
137 <input type="text" className="form-control" value={lastname} onChange={e => setlastn
138 </div>
139 <div className="mb-4 col-6">
140 <label className="form-label">Email</label>
141 <input type="email" className="form-control" value={email} onChange={e => setemail(e.
142 </div>
143 <div className="mb-4 col-6">
144 <label className="form-label">Phone</label>
145 <input type="number" className="form-control" value={phone} onChange={e => setphone(e
146 </div>
147 </div>
148 <div className="row bg-white box-s br-15 pb-3 mt-5 px-3">
149 <h6 className="col-12 text-success my-3 fs-bold">Company Details</h6>
150 <div className="mb-4 col-4">
151 <label className="form-label required">Company Name</label>
152 <input type="text" className="form-control" value={companyName} onChange={e => setCom

```

**Fig 173: Recruiter Profile View Code**

```

14 exports.setup = function(options, seedLink) {
15   dbm = options.dbmigrate;
16   type = dbm.dataType;
17   seed = seedLink;
18   Promise = options.Promise;
19 };
20
21 exports.up = function(db) {
22   var filePath = path.join(__dirname, 'sqls', '20221120071930-insertjobs-up.sql');
23   return new Promise( function( resolve, reject ) {
24     fs.readFile(filePath, {encoding: 'utf-8'}, function(err,data){
25       if (err) return reject(err);
26       console.log('received data: ' + data);
27
28       resolve(data);
29     });
30   });
31   .then(function(data) {
32     return db.runSql(data);
33   });
34 };
35
36 exports.down = function(db) {
37   var filePath = path.join(__dirname, 'sqls', '20221120071930-insertjobs-down.sql');
38   return new Promise( function( resolve, reject ) {
39     fs.readFile(filePath, {encoding: 'utf-8'}, function(err,data){
40       if (err) return reject(err);

```

**Fig 184: Insert Job Migration Code**