CSE 471 : System Analysis and Design
Project Documentation
Project Title : Job Seeking Website

Section:07 **Group: 02** **Session: Fall 23**	
Name	**ID**
Syed Faysel Ahammad Rajo	21101078
Naomi Afrin Jalil	21201325
Anika Islam	21101298

Table of Contents

Introduction

In this competitive world, there is a constant search for a suitable job which can provide both financial and mental bliss. To make this procedure more simple and fast, our job seeking website provides a chance for the applicant to search for their desired job under well-renowned workplaces and the company to receive talented and expertised employees under the same platform.

Looking forward to our website, we have two user types : company and user. The companies can sign up as the company, update their profile, post about job vacancies and requirements, and accept/reject an applicant's application for a job. On the other hand, the applicants can sign up as the user, view their profile, update their profile, view all the jobs they applied for, and search for the job posts by the companies and apply for a job. All companies view and job listing are also included with sorting, searching and filtering features for faster access of jobs by job name and company.

Some of the key features are listed below :

- Posting about job vacancies by companies
- Searching for jobs by applicants
- Applying for jobs by applicants
- Companies view list
- Job Posts list
- Sort by newest-oldest, A-Z,Z-A or experience or job types
- Job details view

With the help of our website, there comes an easier scope to look for work under many options provided by many companies under the same platform. Also, companies can choose the best among all the applicants for their betterment and profit.

Functional Requirements

1. **Login & Sign in** (JWT based)
(a) User login
(b) User sign in
(c) Company login
(d) Company sign in

2. **Company Profile :**
(a) Company profile view
(b) Update company profile
(c) Upload job
(d) View all jobs posted
(e) View applicants' resume applied for a job

3. **User Profile :**
(a) User profile view
(b) Update user profile
(c) Update CV
(d) View all applied jobs

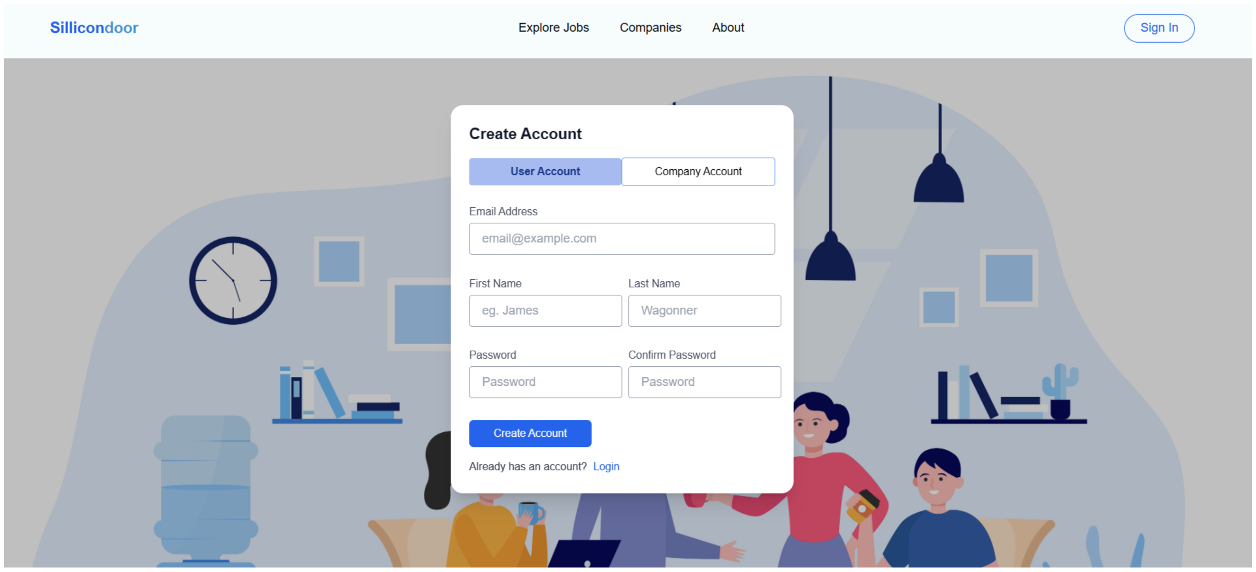
4. **Categorize Jobs:**
(a) Show all jobs
(b) Searching & filtering jobs
(c) Sorting jobs
(d) Show individual job page & apply job option
(e) Show similar jobs in individual job page

5. **Categorize Companies:**
(a) Show all companies
(b) Searching companies
(c) Sorting
(d) Show individual company page

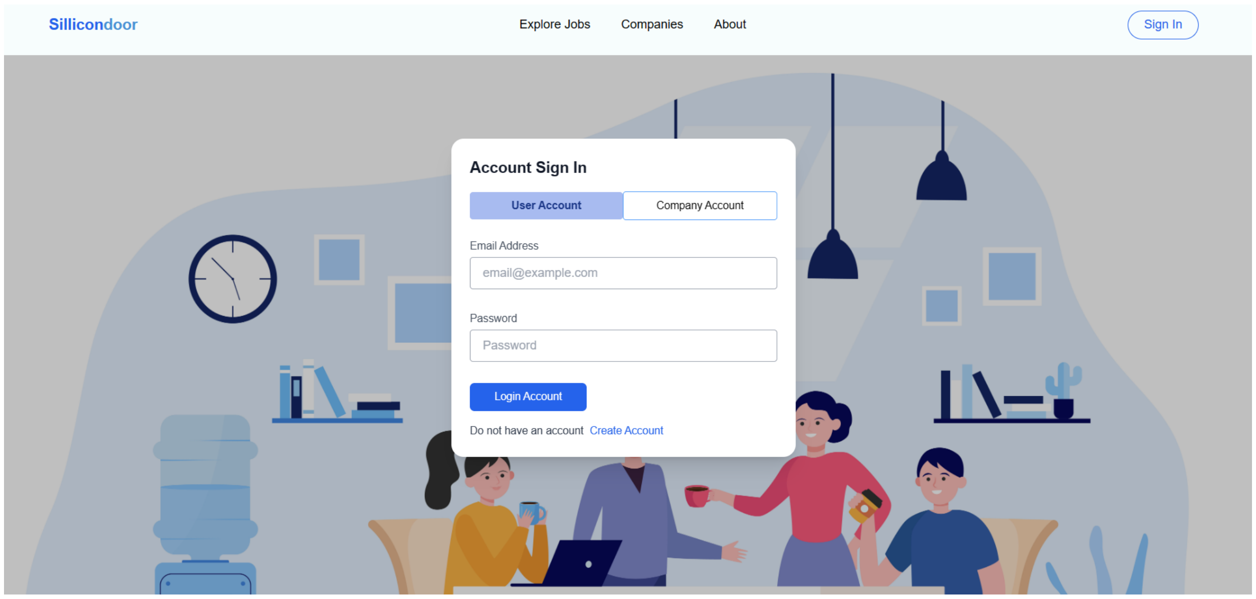
<u>User Manual</u> (For User Account)

User Account is used by the job applicants who will be using the website for viewing job posts posted by companies, searching for their desired job from the listed job availability and applying to the one which suits their desire.

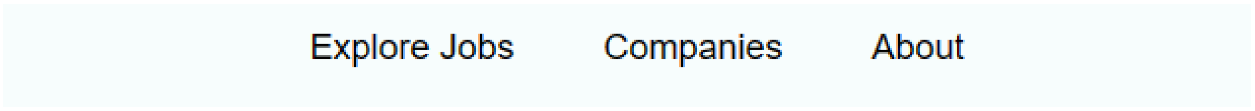
(1) If the job applicant does not have an account on the website, they can create one by clicking on the "User Account" and fill in all the required information asked in the form. Clicking on the "Create Account" button will create an account for them.



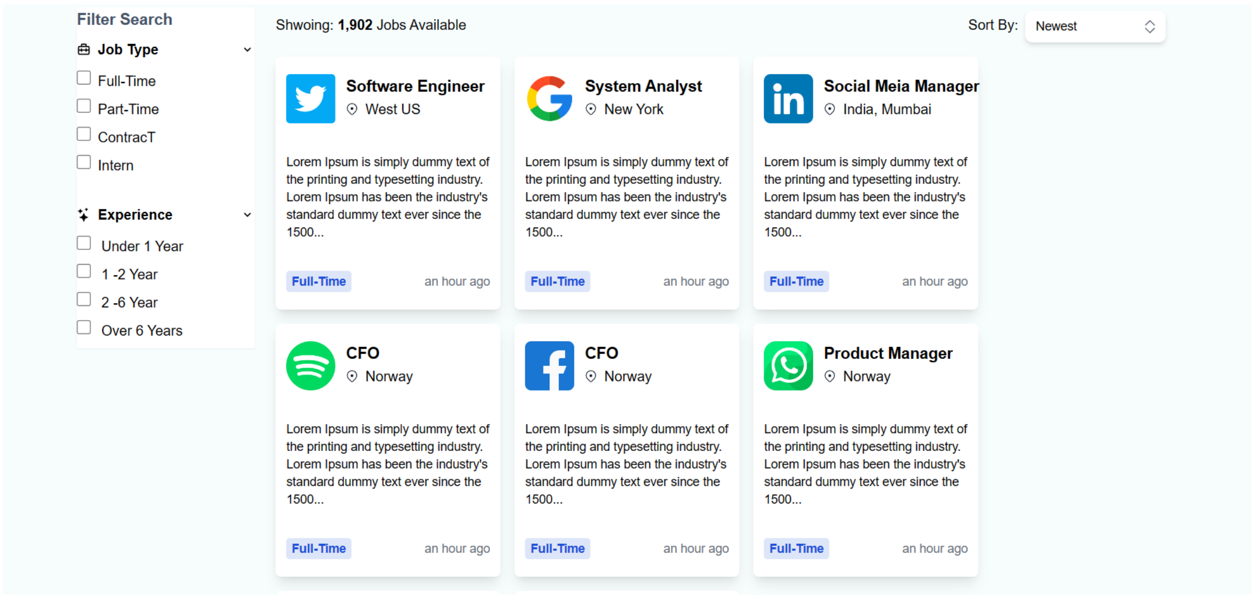
(2) If the job applicant already has an account, they need to just click on the "User Account" on the Sign In view and provide their correct email address and password for successful login.



(3) For viewing all the jobs available, the user needs to click on the "Explore jobs" on the navigation bar. All the job posts created by the companies will be displayed on the page.



Navigation bar



Software Engineer
West US

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time an hour ago

System Analyst
New York

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time an hour ago

Social Meia Manager
India, Mumbai

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time an hour ago

CFO
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time an hour ago

CFO
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

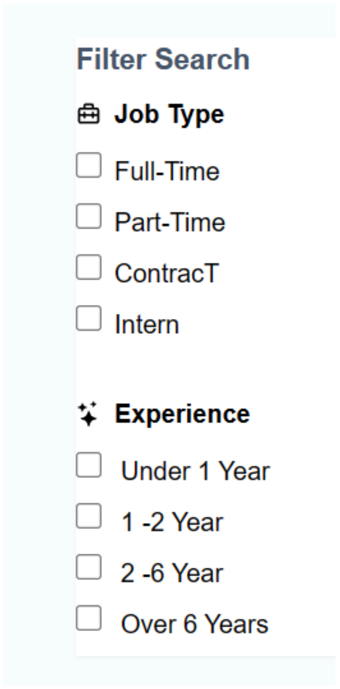
Full-Time an hour ago

Product Manager
Norway

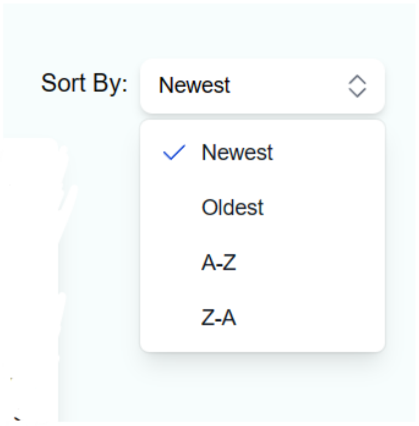
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time an hour ago

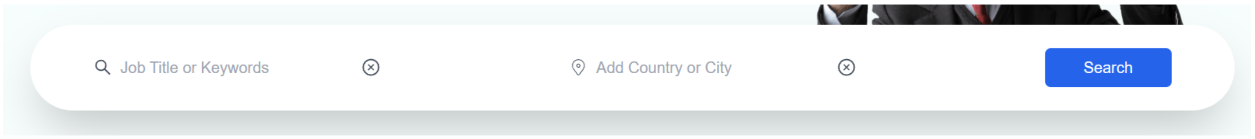
(4) Filter Search options are given. By clicking on the "Job Type" options will arrange all the job posts according to their job type. Also, by clicking on the "Experience" will arrange all the job posts according to their experience required.



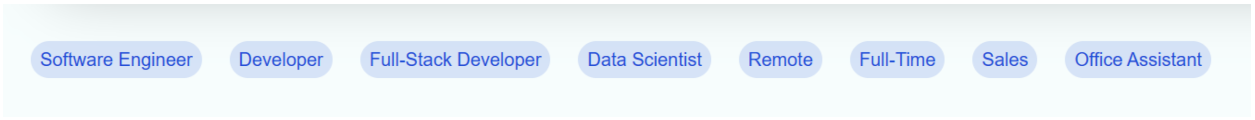
(5) A Sort Dropdown list is also provided. Clicking on any of the options below, the job posts will be arranged according to that option.



(6) Search bar is provided where one can search by name either job or company.

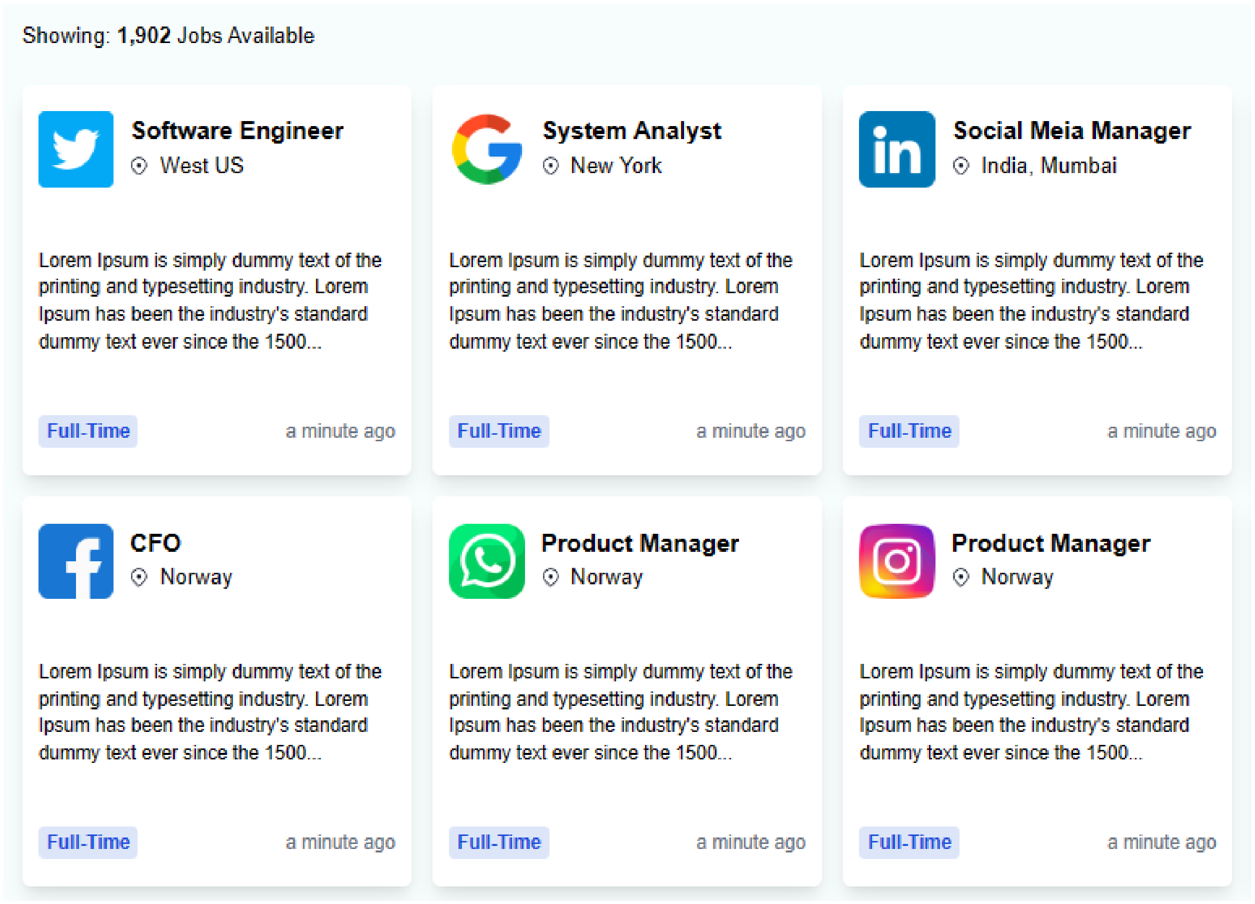


(7) Job Types are provided here by clicking on a specific job type and will take the user to jobs available under that job type.



Software Engineer | Developer | Full-Stack Developer | Data Scientist | Remote | Full-Time | Sales | Office Assistant

(8) By clicking on any of the job posts, will take one to the job details page of that specific job provided by a company.



Showing: **1,902** Jobs Available

Software Engineer
West US

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

System Analyst
New York

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

Social Meia Manager
India, Mumbai

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

CFO
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

Product Manager
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

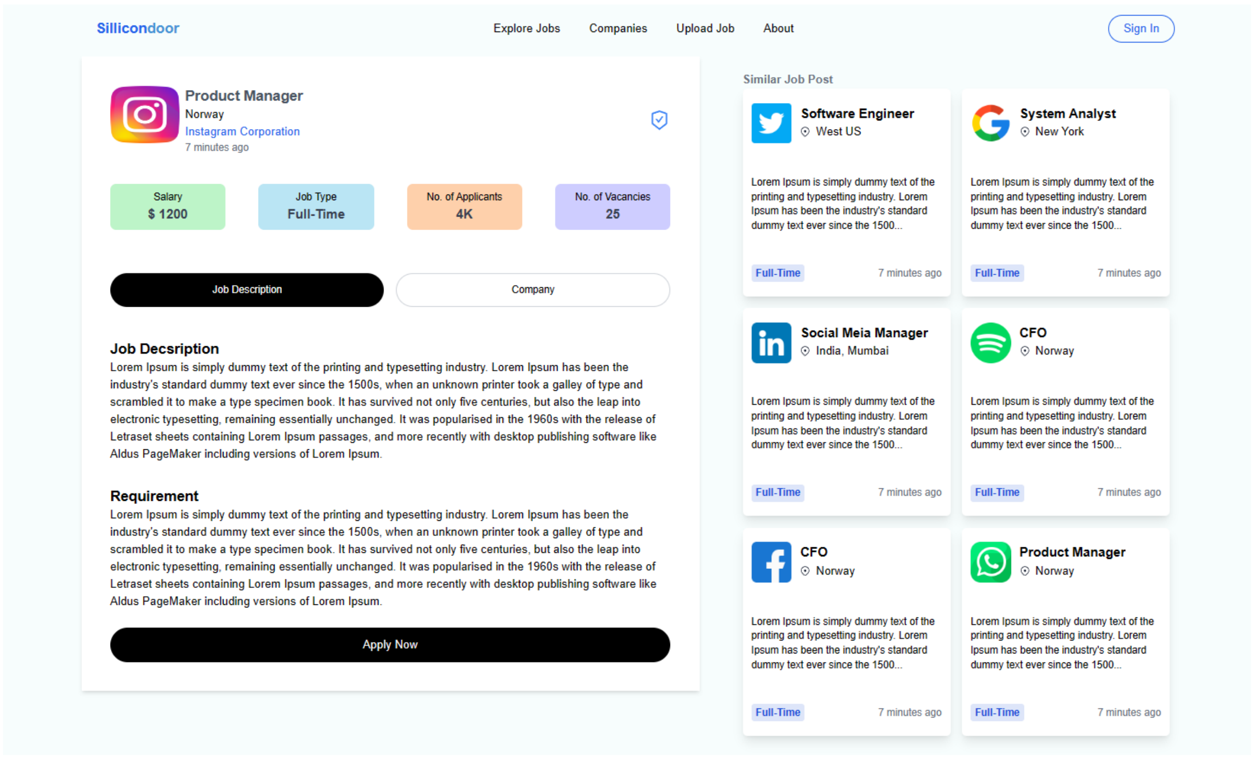
Full-Time a minute ago

Product Manager
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

(9) For example, clicking on the product manager job post from the jobs available posts, will take one to the job details of the product manager. The right hand side of the page has all the similar job posts present. While the left hand side has the job details.



Product Manager
Norway
Instagram Corporation
7 minutes ago

Salary	Job Type	No. of Applicants	No. of Vacancies
$ 1200	Full-Time	4K	25

Job Decsription
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Requirement
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Software Engineer
West US
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

System Analyst
New York
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

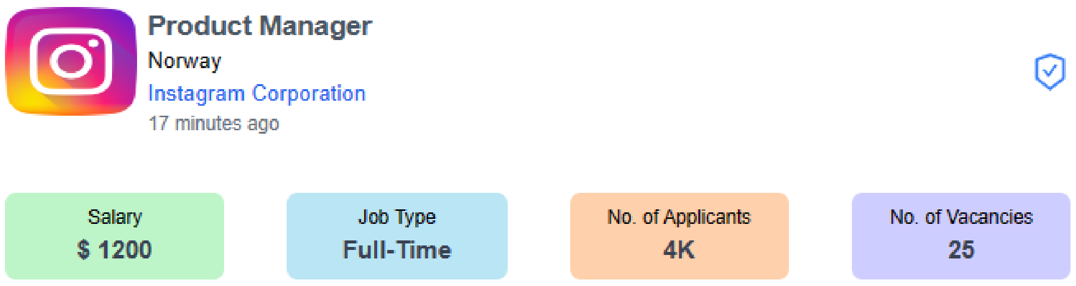
Social Meia Manager
India, Mumbai
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

Product Manager
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

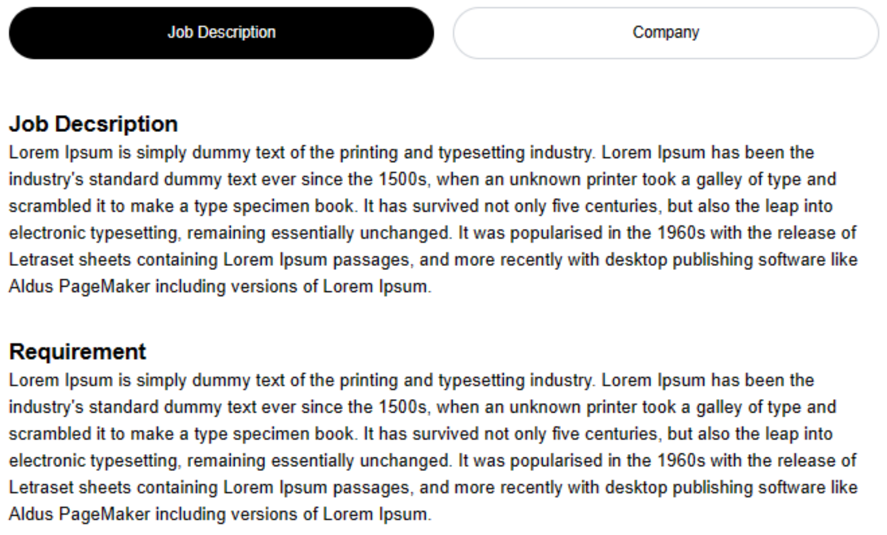
(10) On this side, the salary and working duration for the job, the number of applicants and vacancies are shown.



Product Manager
Norway
Instagram Corporation
17 minutes ago

Salary	Job Type	No. of Applicants	No. of Vacancies
$ 1200	Full-Time	4K	25

(11) On clicking on the Job description will provide all the job information provided by the company.



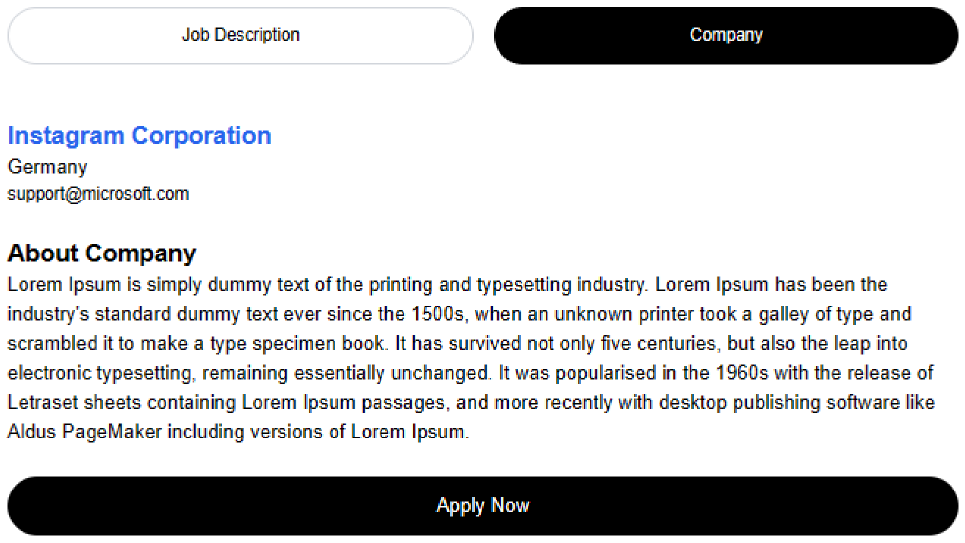
Job Decsription

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Requirement

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

(12) On clicking on the company, will provide the company description.



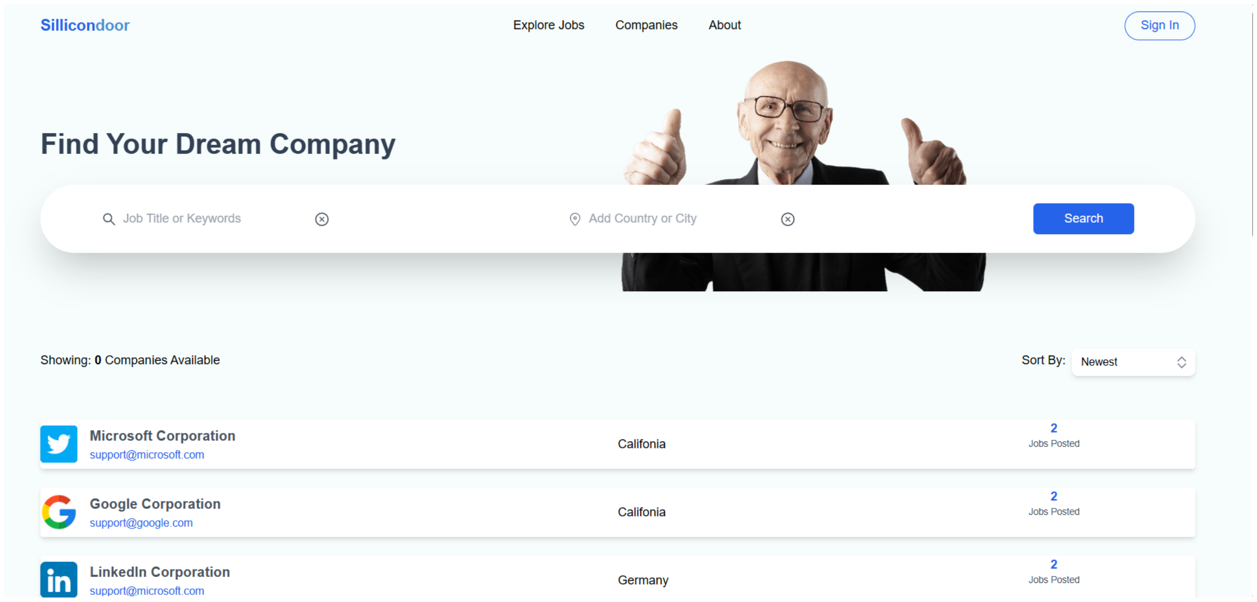
Instagram Corporation
Germany
support@microsoft.com

About Company

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

(13) On clicking on the "Apply Now" button, applicant information will be redirected to the company.

(14) For viewing all the companies seeking applicants, the user needs to click on the "Companies" on the navigation bar. All the companies present will be shown.

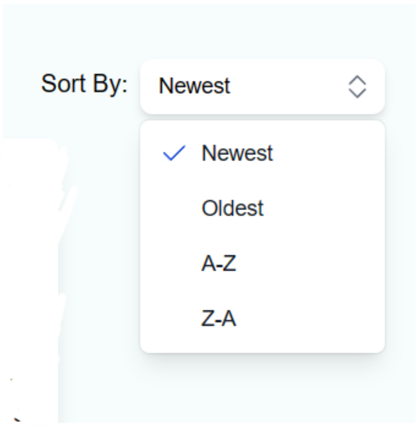


Showing: **0** Companies Available

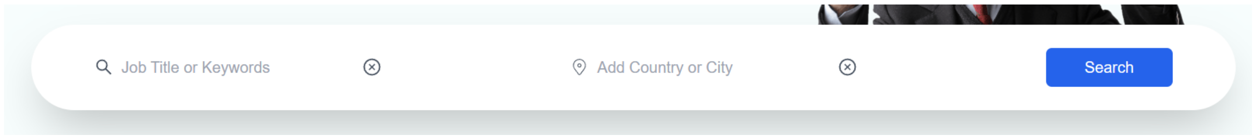
Sort By: Newest

Microsoft Corporation support@microsoft.com	Califonia	2 Jobs Posted
Google Corporation support@google.com	Califonia	2 Jobs Posted
LinkedIn Corporation support@microsoft.com	Germany	2 Jobs Posted

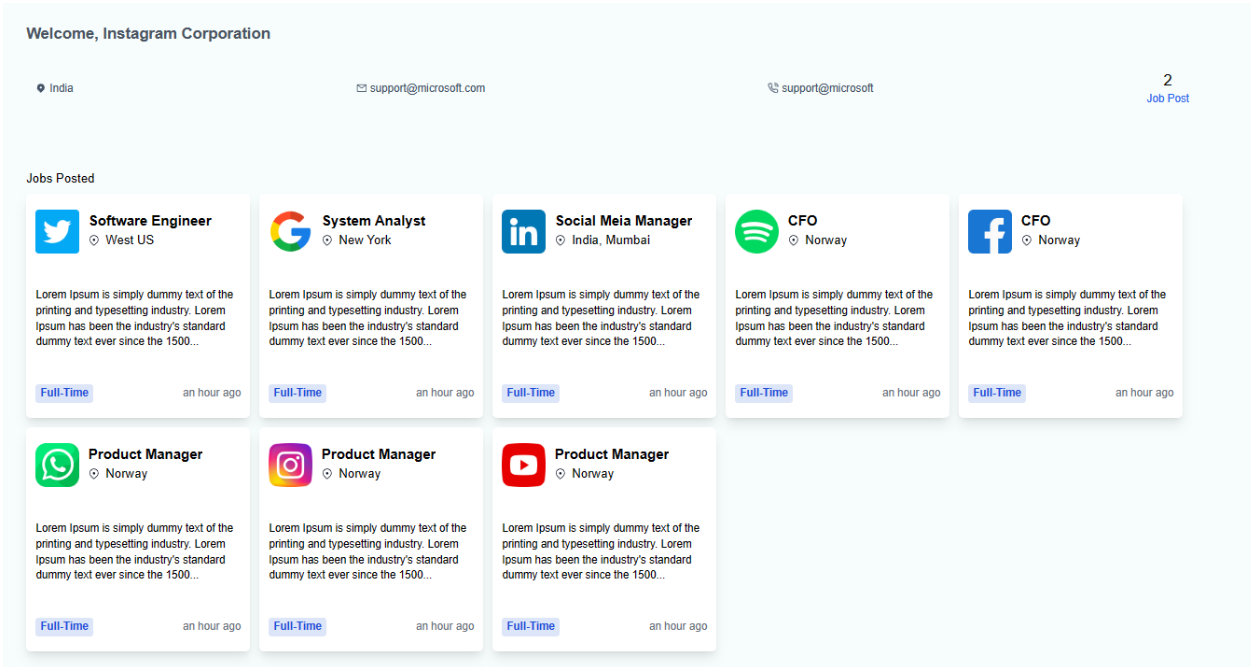
(15) A Sort Dropdown list is also provided like the "Explore Jobs". Clicking on any of the options below, the job posts will be arranged according to that option.



(16) Search bar is provided where one can search by name either job or company.



(17) On clicking on any of the companies,will take one to the company profile. The company profile has its name, email address and location, together with all the job posts it has posted and the total job posts created.

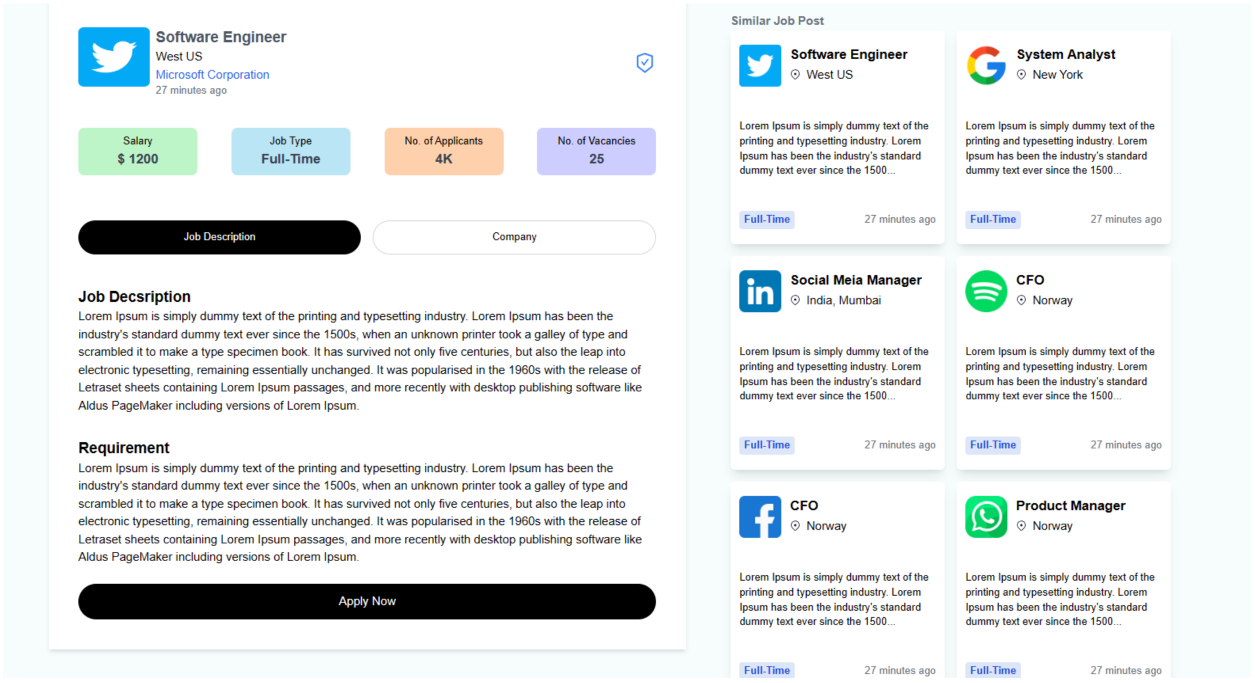

Welcome, Instagram Corporation

India | support@microsoft.com | support@microsoft | 2 Job Post

Jobs Posted

Title	Location	Description	Type	Posted
Software Engineer	West US	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago
System Analyst	New York	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago
Social Meia Manager	India, Mumbai	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago
CFO	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago
CFO	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago
Product Manager	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago
Product Manager	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago
Product Manager	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	an hour ago

(18) On clicking on any of the job posts, will take one to the job details of that specific job offered by the company.


Software Engineer
West US
Microsoft Corporation
27 minutes ago

Salary	Job Type	No. of Applicants	No. of Vacancies
$ 1200	Full-Time	4K	25

Job Decsription

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

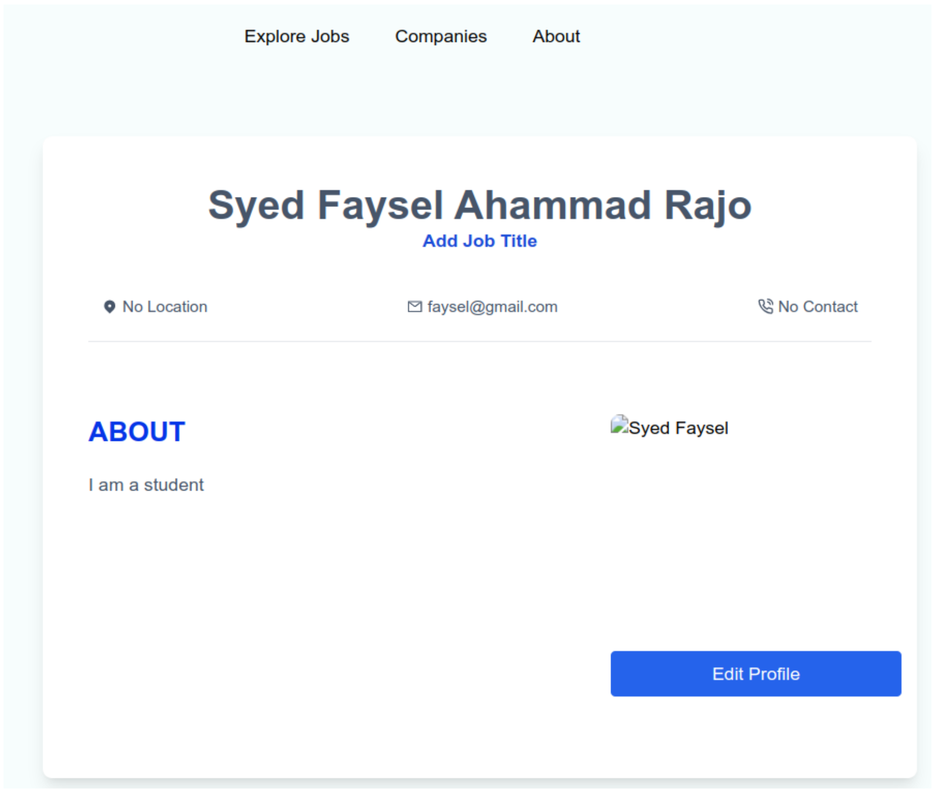
Requirement

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Similar Job Post

Title	Location	Description	Type	Posted
Software Engineer	West US	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	27 minutes ago
System Analyst	New York	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	27 minutes ago
Social Meia Manager	India, Mumbai	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	27 minutes ago
CFO	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	27 minutes ago
CFO	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	27 minutes ago
Product Manager	Norway	Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...	Full-Time	27 minutes ago

(19) Users can view their own profile. Name of user, about and profile image will be displayed on that view.



Syed Faysel Ahammad Rajo

Add Job Title

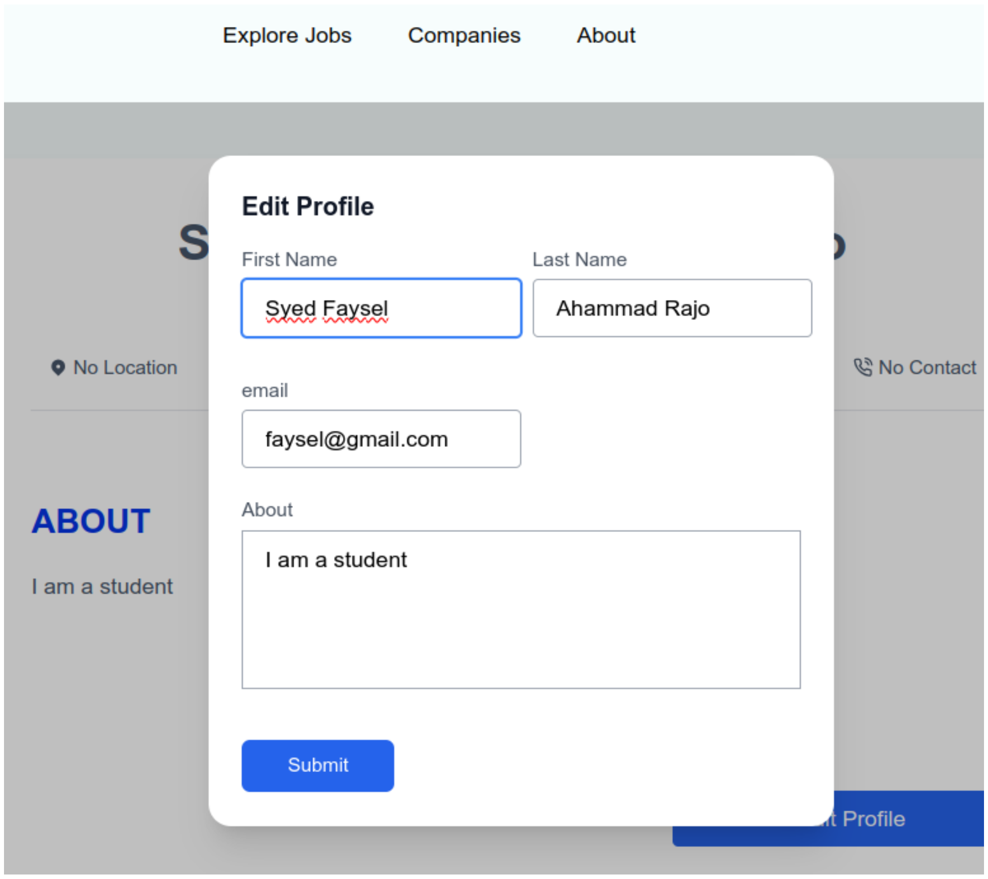
No Location faysel@gmail.com No Contact

ABOUT

I am a student

Syed Faysel

(20) On clicking on the “Edit Profile” button, the user will be directed to a page where they can edit their information. On clicking the submit button, the edited information will be displayed on the user view.



Edit Profile

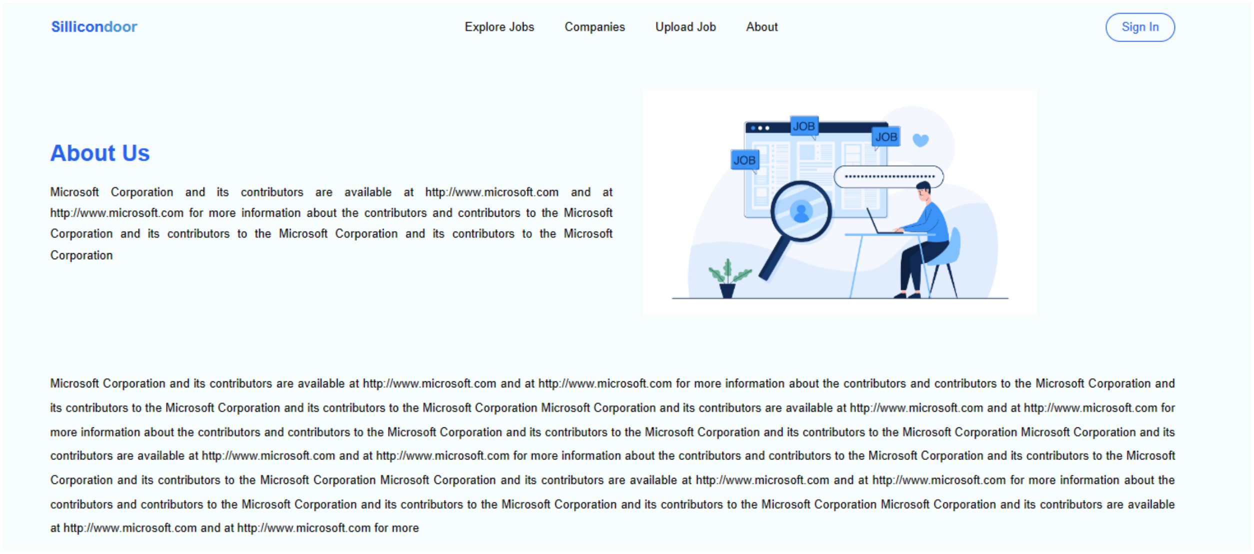
First Name	Last Name
Syed Faysel	Ahammad Rajo
email	
faysel@gmail.com	
About	
I am a student	

Submit

ABOUT

I am a student

(21) On clicking on "About" on the navigation bar, will take one to the information of the platform.

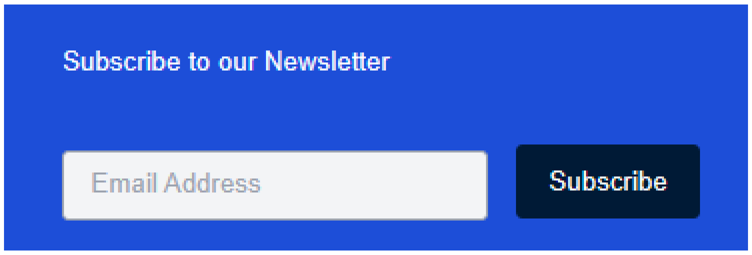


About Us

Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation

Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more

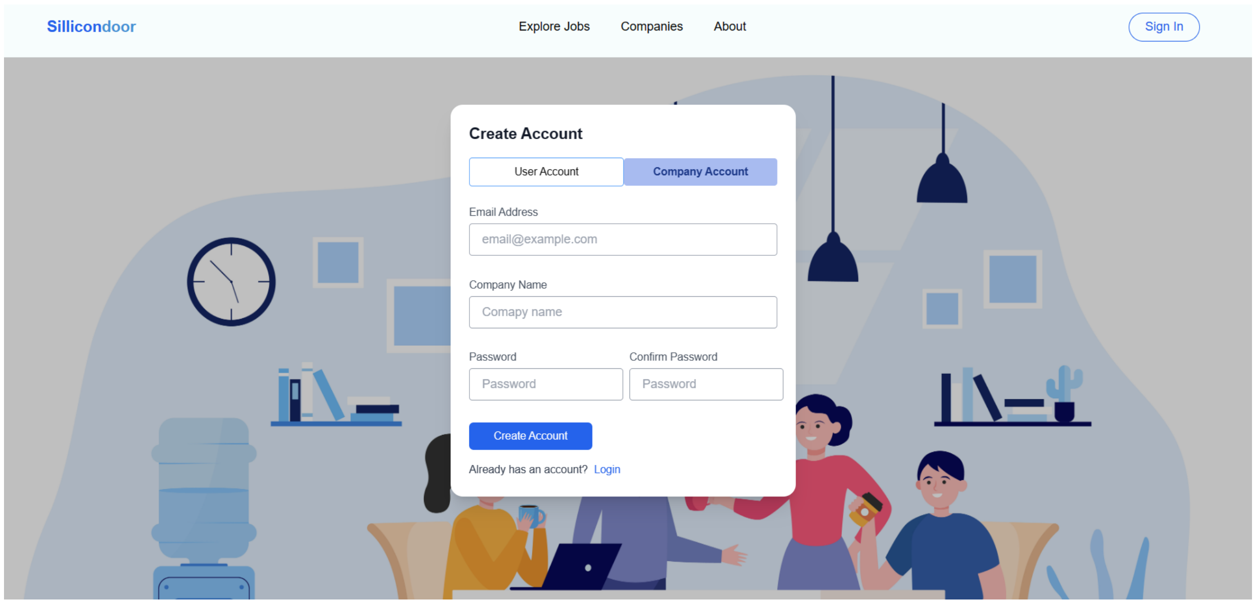
(22) In the footer, there is a newsletter subscription form. To get new updates about the platform and companies seeking applicants, one can receive direct notification if he/she is subscribed to the newsletter.


Subscribe to our Newsletter
Email Address
Subscribe

<u>User Manual</u> (For Company Account)

Company Account is used by the companies who want to use the website to post for job vacancies with proper requirements provided .

(1) If the company does not have an account on the website, they can create one by clicking on the "Company Account" and fill in all the required information asked in the form. Clicking on the "Create Account" button will create an account for them.



Create Account

Email Address

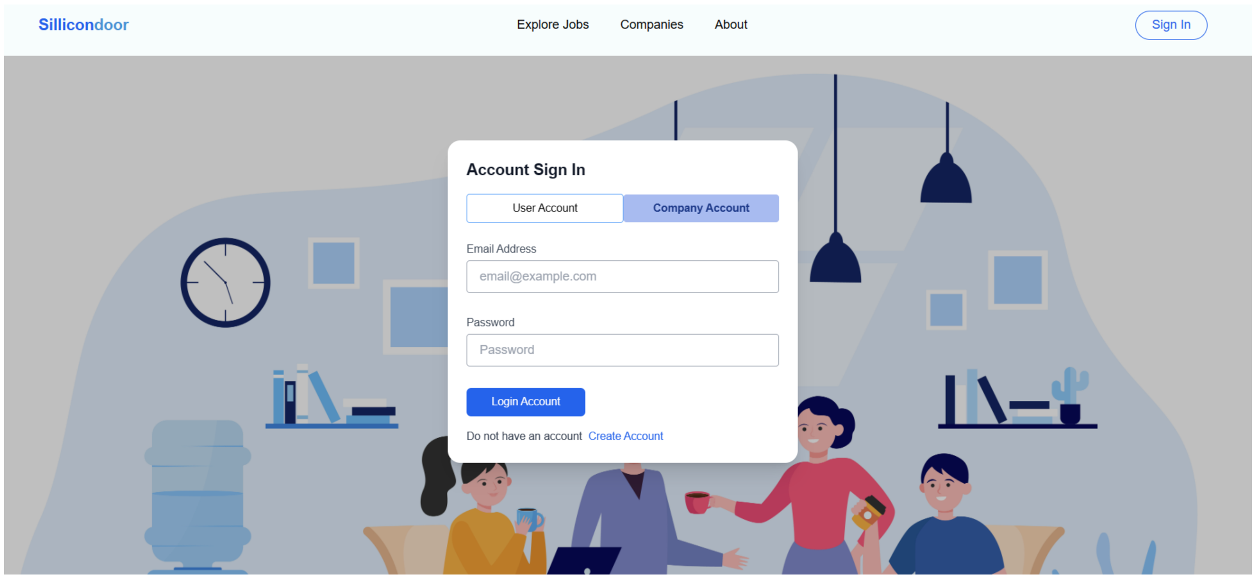
Company Name

Password

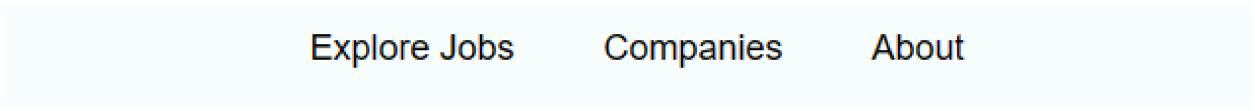
Confirm Password

Already has an account?

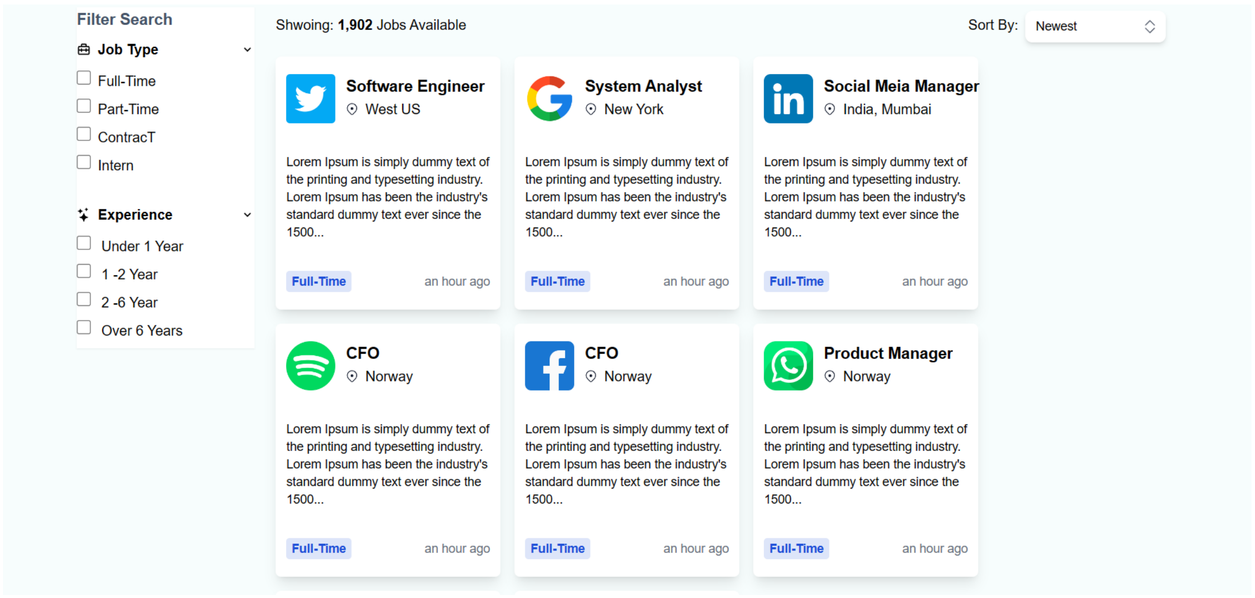
(2) If the company already has an account, they need to just click on the "Company Account" on the Sign In view and provide their correct email address and password for successful login.



(3) For viewing all the job posts, the company needs to click on the "Explore jobs" on the navigation bar. All the job posts created by the companies will be displayed on the page.



Navigation bar



Software Engineer
West US
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time an hour ago

System Analyst
New York
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time an hour ago

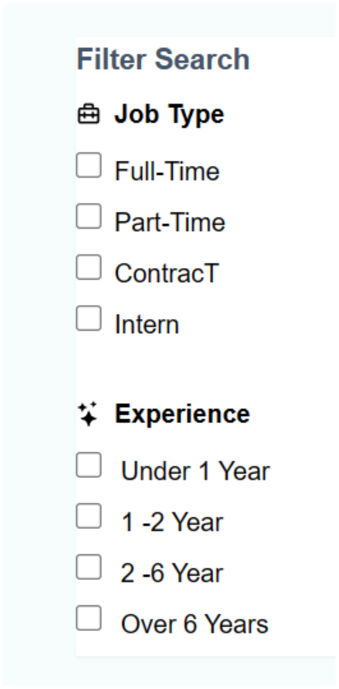
Social Meia Manager
India, Mumbai
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time an hour ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time an hour ago

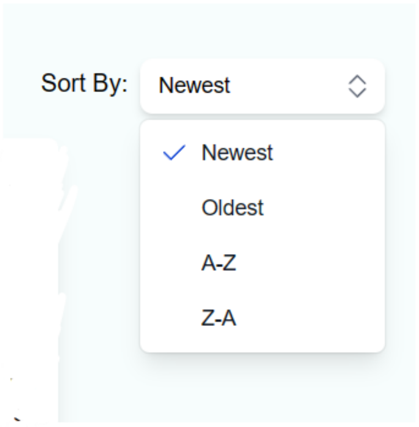
CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time an hour ago

Product Manager
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time an hour ago

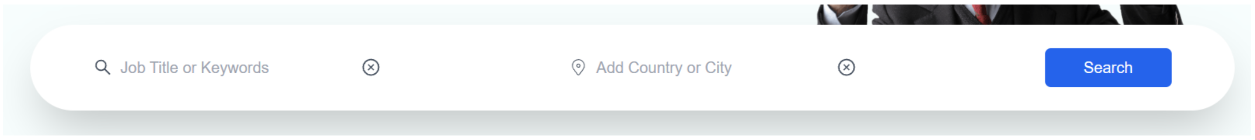
(4) Filter Search options are given. By clicking on the "Job Type" options will arrange all the job posts according to their job type. Also, by clicking on the "Experience" will arrange all the job posts according to their experience required.



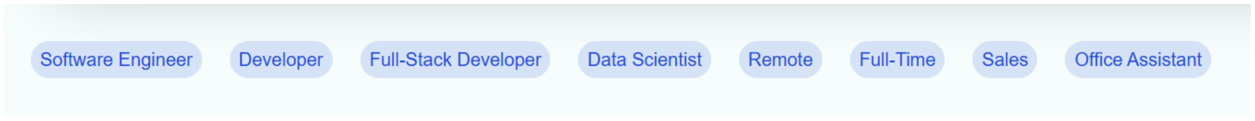
(5) A Sort Dropdown list is also provided. Clicking on any of the options below, the job posts will be arranged according to that option.



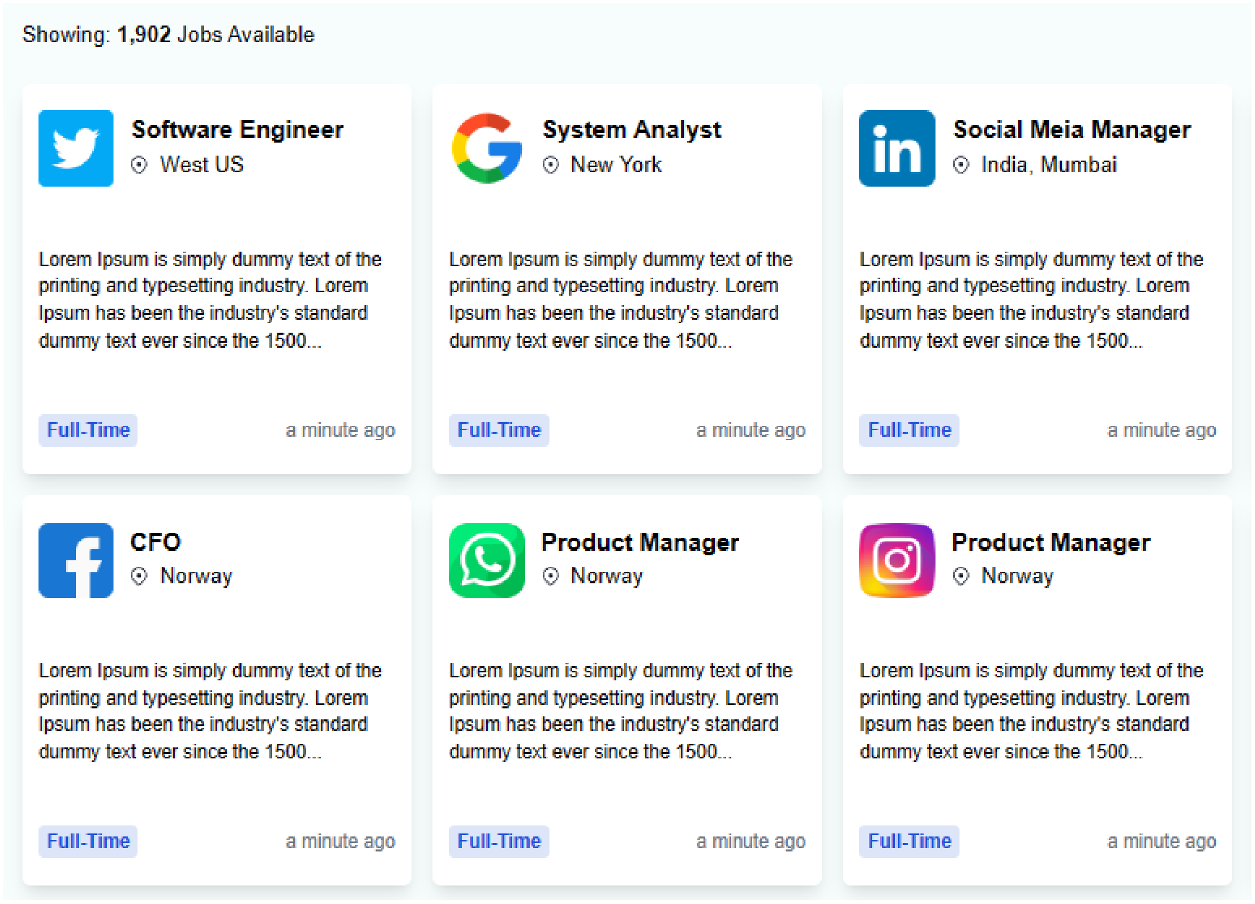
(6) Search bar is provided where one can search by name either job or company.



(7) Job Types are provided here by clicking on a specific job type and will take the company to jobs available under that job type.



(8) By clicking on any of the job posts, will take one to the job details page of that specific job provided by a company.



Showing: **1,902** Jobs Available

Software Engineer
West US

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

System Analyst
New York

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

Social Meia Manager
India, Mumbai

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

CFO
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

Product Manager
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

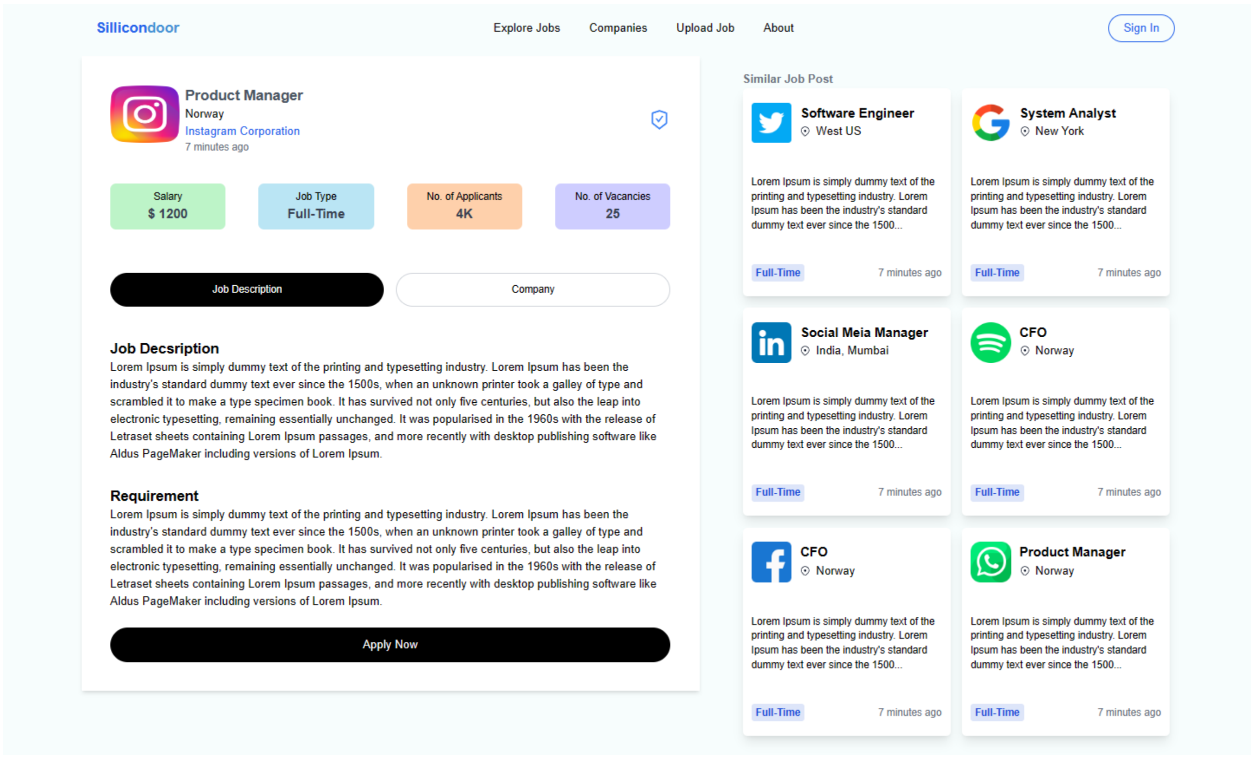
Full-Time a minute ago

Product Manager
Norway

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time a minute ago

(9) For example, clicking on the product manager job post from the jobs available posts, will take one to the job details of the product manager. The right hand side of the page has all the similar job posts present. While the left hand side has the job description.



Product Manager
Norway
Instagram Corporation
7 minutes ago

Salary	Job Type	No. of Applicants	No. of Vacancies
$ 1200	Full-Time	4K	25

Job Decsription
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Requirement
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Similar Job Post

Software Engineer
West US
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

System Analyst
New York
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

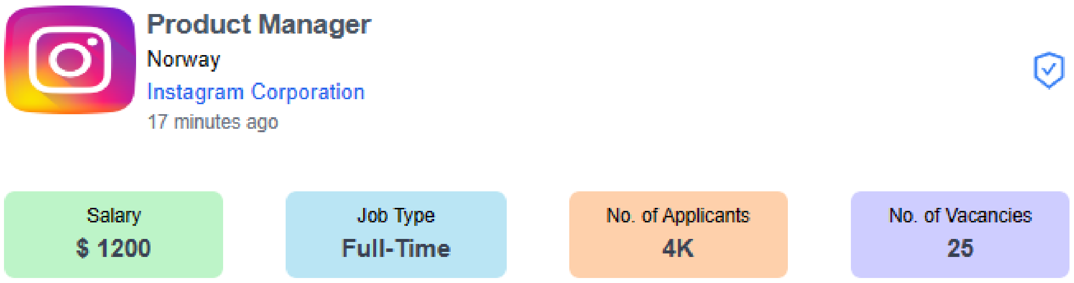
Social Meia Manager
India, Mumbai
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

Product Manager
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time 7 minutes ago

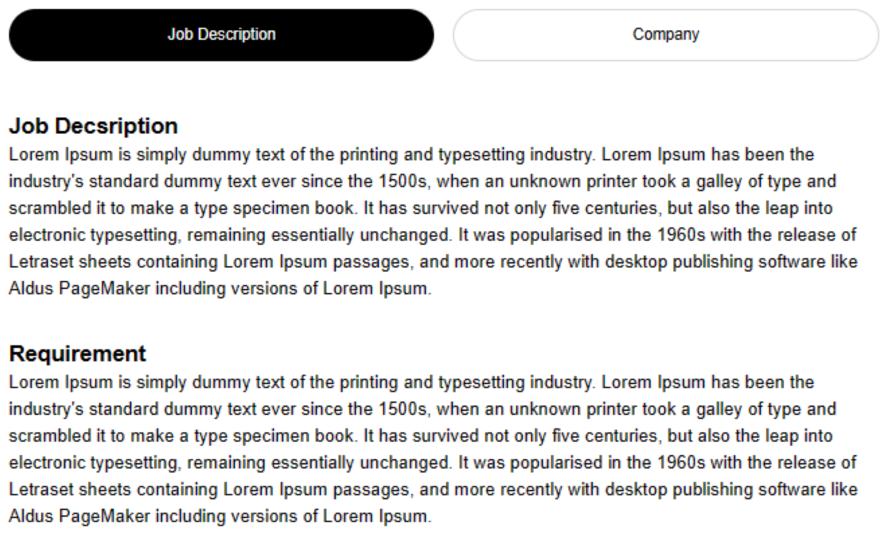
(10) On this side, the salary and working duration for the job, the number of applicants and vacancies are shown.



Product Manager
Norway
Instagram Corporation
17 minutes ago

Salary	Job Type	No. of Applicants	No. of Vacancies
$ 1200	Full-Time	4K	25

(11) On clicking on the Job description will provide all the job information provided by the company.



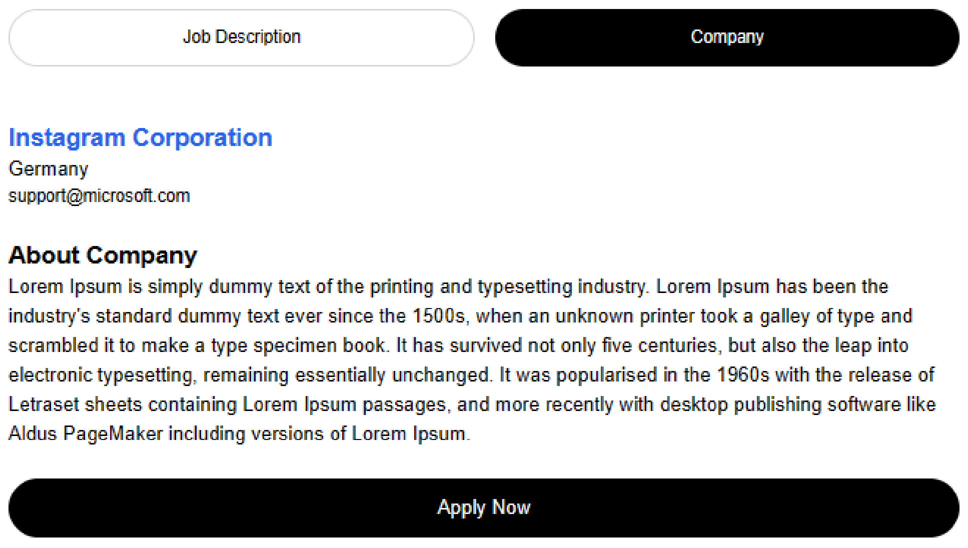
Job Decsription

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Requirement

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

(12) On clicking on the company, will provide the company description.



Instagram Corporation

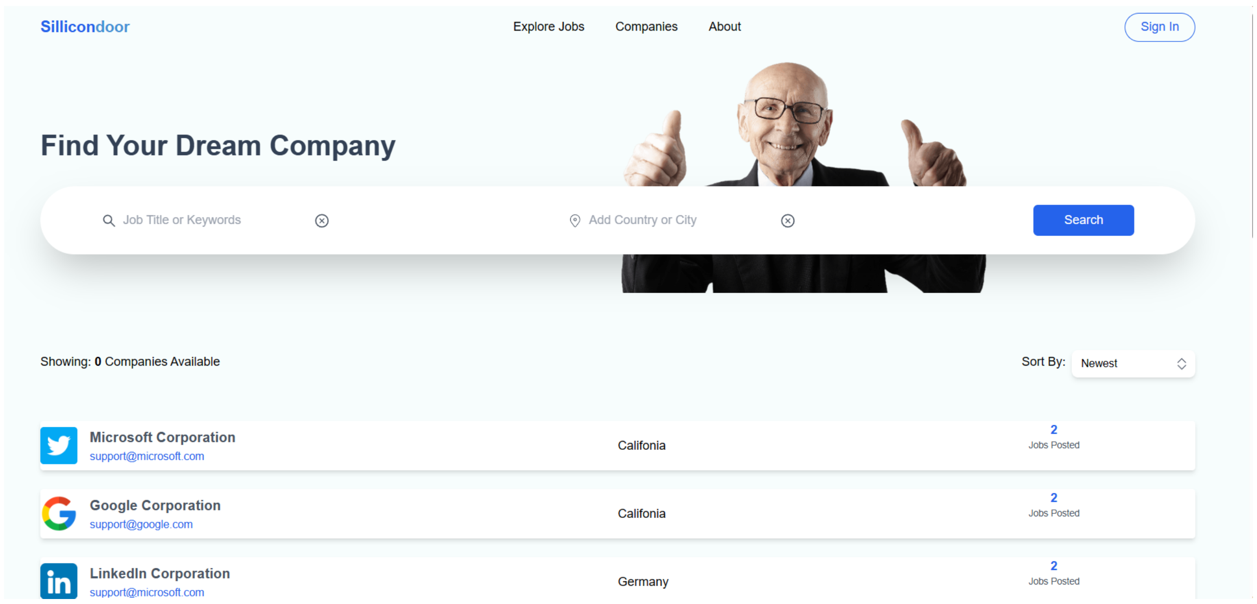
Germany

support@microsoft.com

About Company

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

(13) For viewing all the companies seeking applicants, the company needs to click on the "Companies" on the navigation bar. All the companies present will be shown.

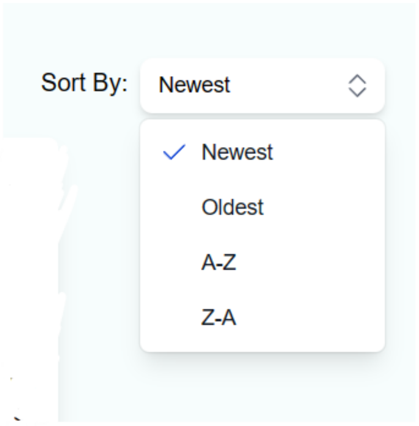


Showing: **0** Companies Available

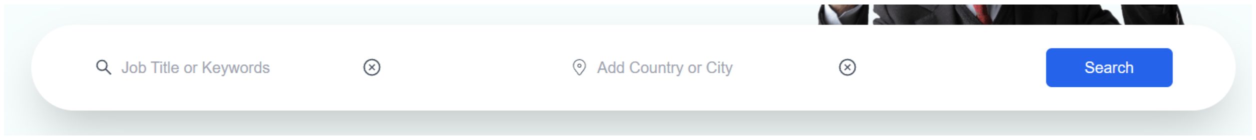
Sort By: Newest

Microsoft Corporation support@microsoft.com	Califonia	2 Jobs Posted
Google Corporation support@google.com	Califonia	2 Jobs Posted
LinkedIn Corporation support@microsoft.com	Germany	2 Jobs Posted

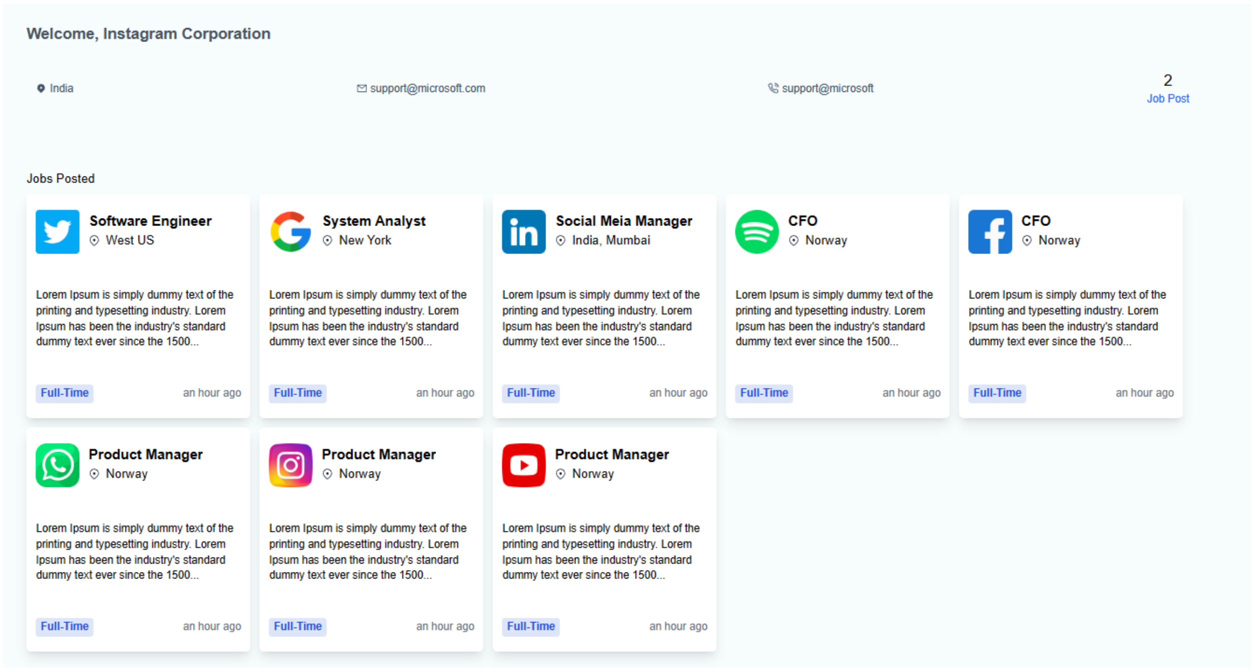
(14) A Sort Dropdown list is also provided like the "Explore Jobs". Clicking on any of the options below, the job posts will be arranged according to that option.



(15) Search bar is provided where one can search by name either job or company.



(16) On clicking on any of the companies,will take one to the company profile. The company profile has its name, email address and location, together with all the job posts it has posted and the total job posts created.



Welcome, Instagram Corporation

India
support@microsoft.com
support@microsoft

2
Job Post

Jobs Posted

Software Engineer
West US
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

System Analyst
New York
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

Social Meia Manager
India, Mumbai
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

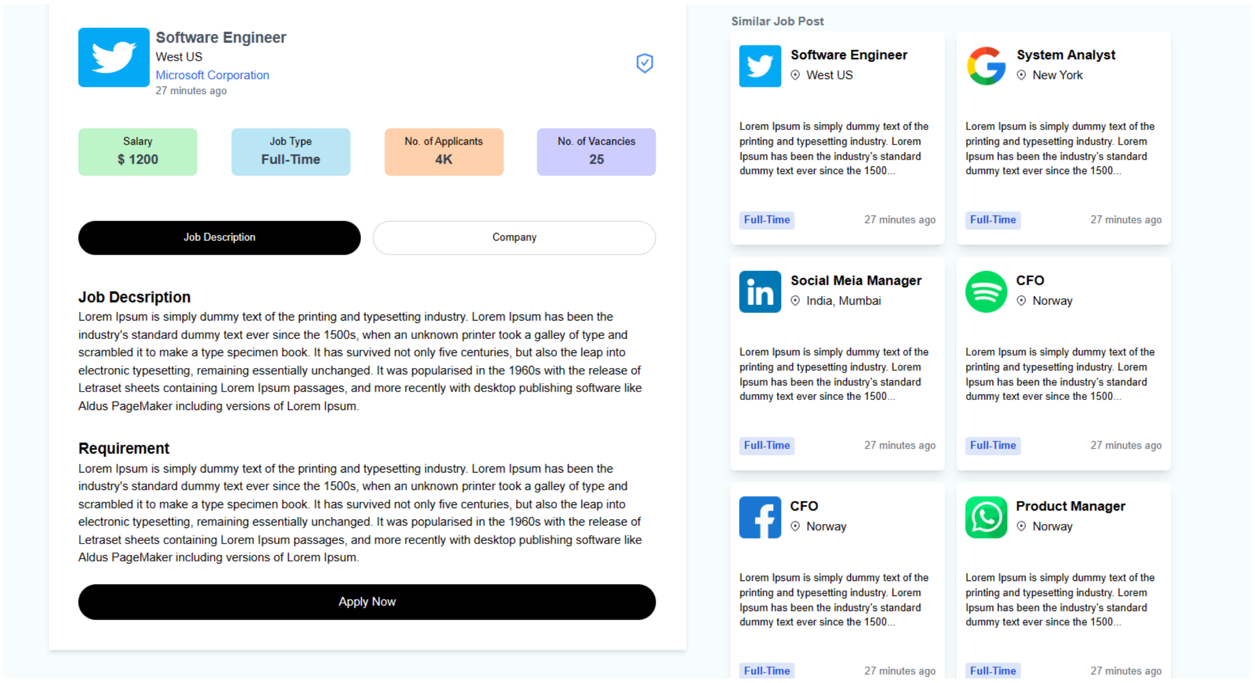
CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

Product Manager
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

Product Manager
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

Product Manager
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
an hour ago

(17) On clicking on any of the job posts, will take one to the job details of that specific job offered by the company.



Software Engineer
West US
Microsoft Corporation
27 minutes ago

Salary	Job Type	No. of Applicants	No. of Vacancies
$ 1200	Full-Time	4K	25

Job Decsription
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Requirement
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Similar Job Post

Software Engineer
West US
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
27 minutes ago

System Analyst
New York
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
27 minutes ago

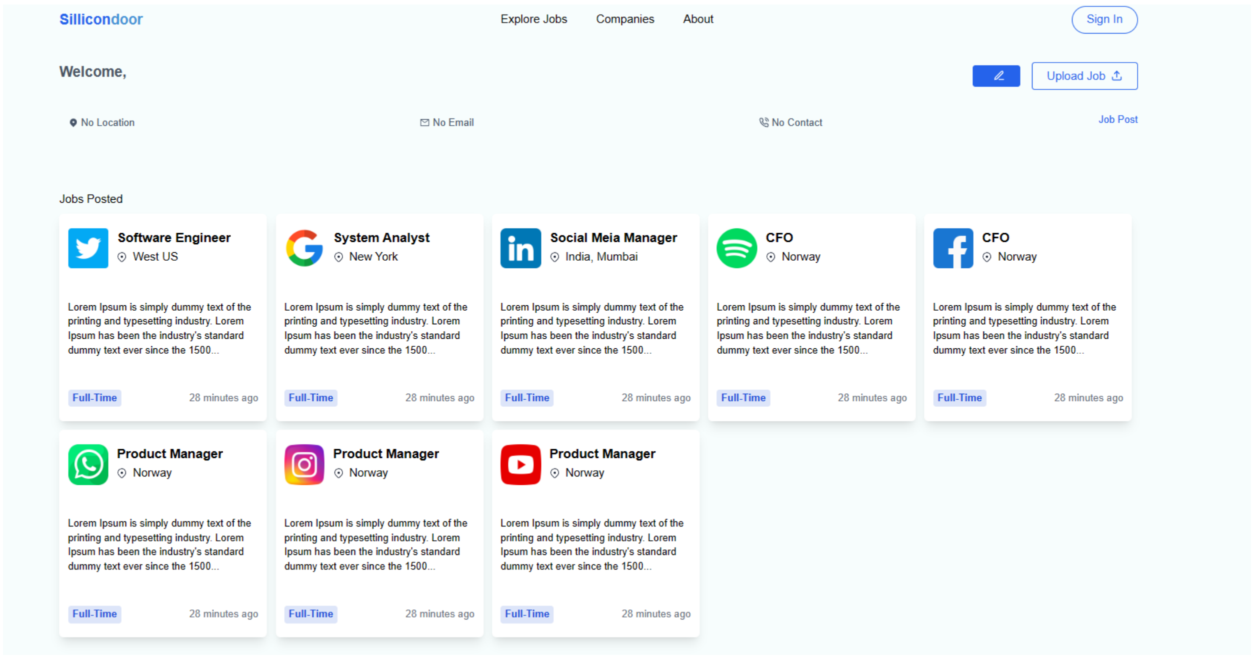
Social Meia Manager
India, Mumbai
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
27 minutes ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
27 minutes ago

CFO
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
27 minutes ago

Product Manager
Norway
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...
Full-Time
27 minutes ago

(18) The company can view their own company profile and all the job posts made will be shown below.


Sillicondoor Explore Jobs Companies About Sign In

Welcome,

No Location No Email No Contact Job Post

Upload Job

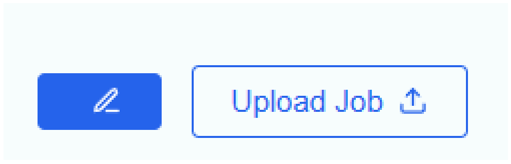
Jobs Posted

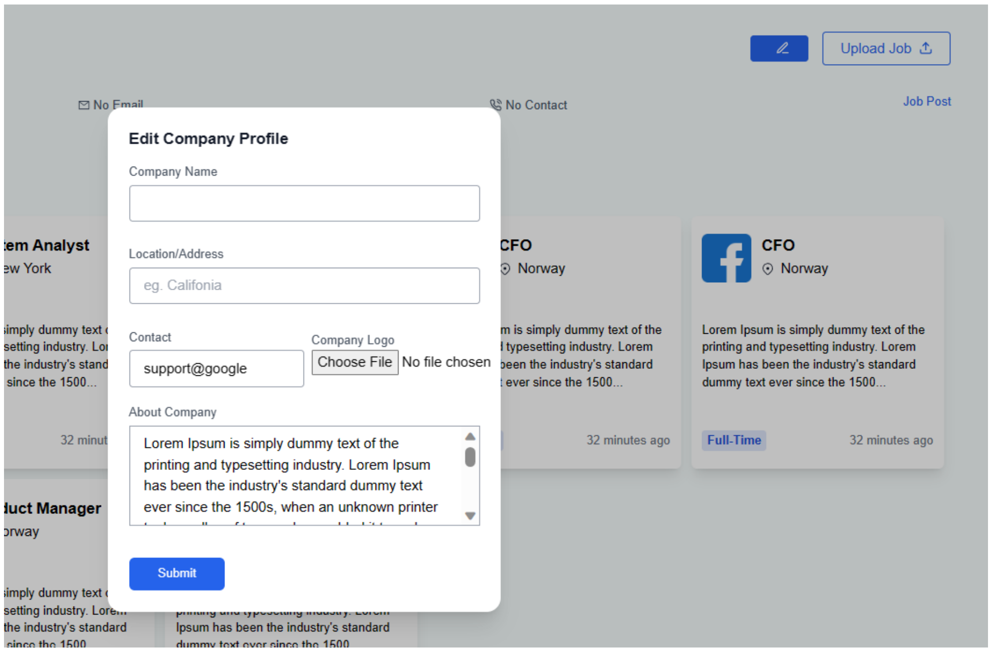
Software Engineer, West US	System Analyst, New York	Social Meia Manager, India, Mumbai	CFO, Norway	CFO, Norway
Product Manager, Norway	Product Manager, Norway	Product Manager, Norway		

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500...

Full-Time 28 minutes ago

(19) On clicking on the edit sign , the company will land into a form where they can edit their profile information, and clicking on the submit button will update their information on the company profile.




Edit Company Profile

Company Name

Location/Address: eg. Califonia

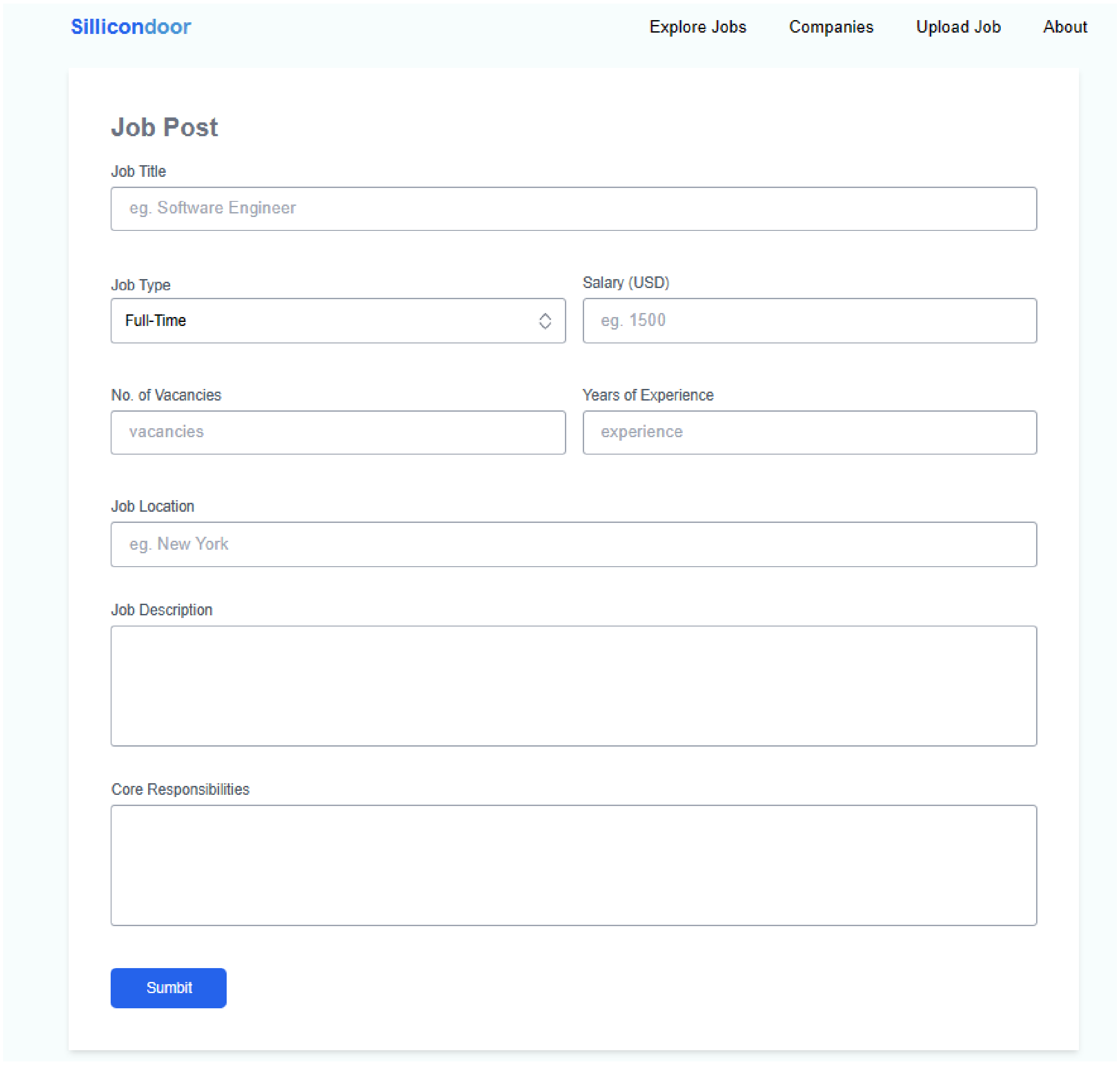
Contact: support@google

Company Logo: Choose File No file chosen

About Company: Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer

Submit

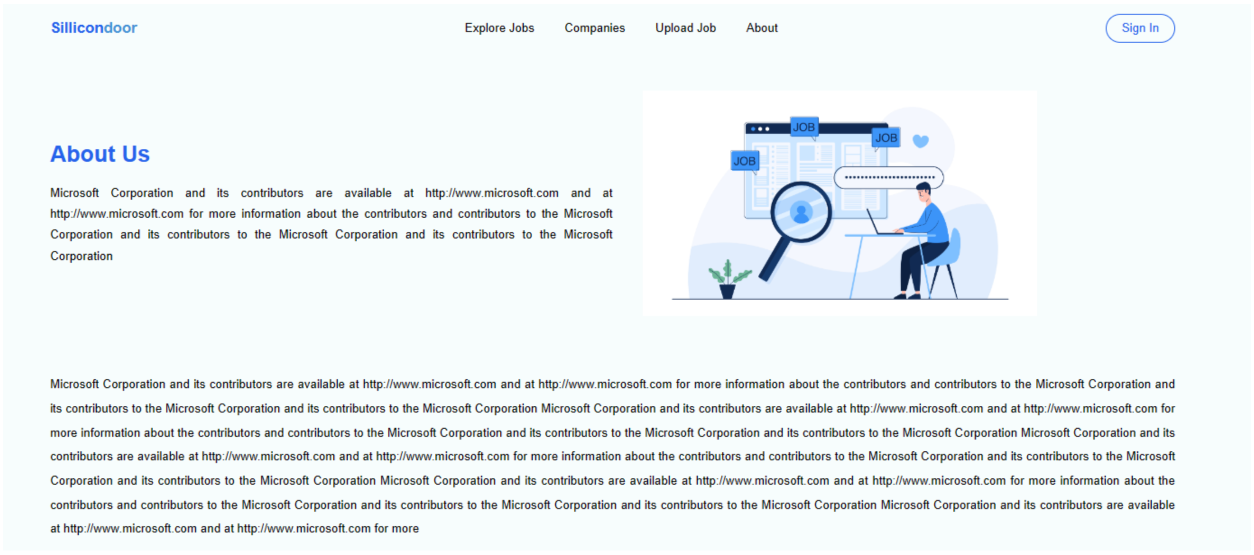
(20) On clicking on the “Upload Job”, the company will be taken to a form where it can fill in the required information and all the job requirements. On clicking “Submit”, the job post will be posted in the company profile.



Job Post

Job Title	eg. Software Engineer
Job Type	Full-Time
Salary (USD)	eg. 1500
No. of Vacancies	vacancies
Years of Experience	experience
Job Location	eg. New York
Job Description	
Core Responsibilities	

(21) On clicking on “About” on the navigation bar, will take one to the information of the platform.

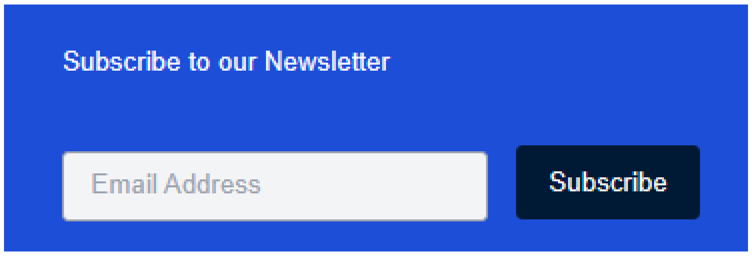


About Us

Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation

Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more information about the contributors and contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation and its contributors to the Microsoft Corporation Microsoft Corporation and its contributors are available at http://www.microsoft.com and at http://www.microsoft.com for more

(22) In the footer, there is a newsletter subscription form. To get new updates about the platform and companies seeking applicants, one can receive direct notification if he/she is subscribed to the newsletter.



Frontend Development

Pages => About (Frontend for the About page)

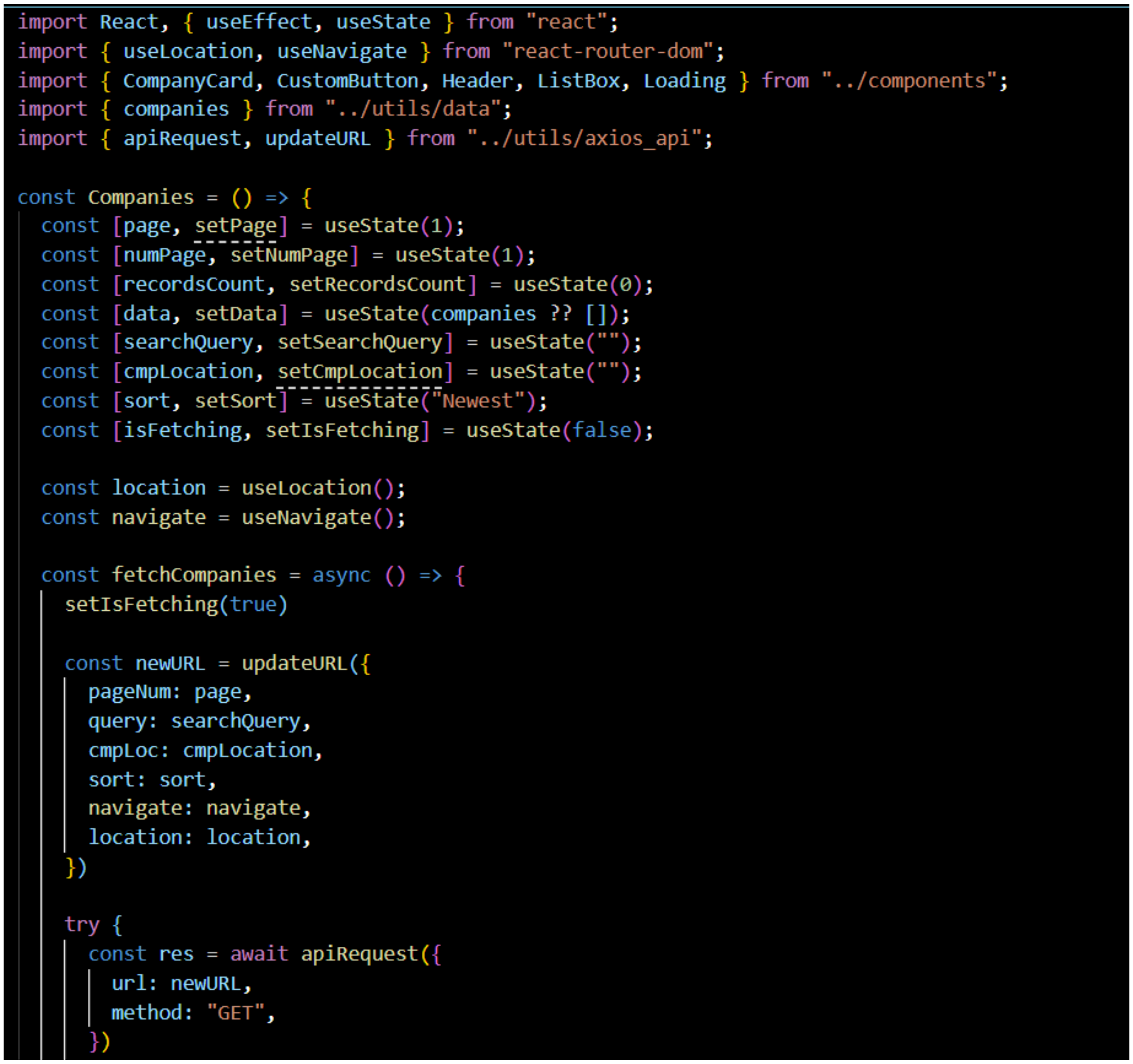
```
import React from "react";
import { JobImg } from "../assets";

const About = () => {
  return (
    <div className='container mx-auto flex flex-col gap-8 2xl:gap-14 py-6
'>
      <div className='w-full flex flex-col-reverse md:flex-row gap-10
items-center p-5'>
        <div className='w-full md:2/3 2xl:w-2/4'>
          <h1 className='text-3xl text-blue-600 font-bold mb-5'>About
Us</h1>
          <p className='text-justify leading-7'>
            Microsoft Corporation and its contributors are available at
            http://www.microsoft.com and at http://www.microsoft.com for
more
            information about the contributors and contributors to the
Microsoft
            Corporation and its contributors to the Microsoft Corporation
and
            its contributors to the Microsoft Corporation
          </p>
        </div>
        <img src={JobImg} alt='About' className='w-auto h-[300px]' />
      </div>

      <div className='leading-8 px-5 text-justify'>
        <p>
          Microsoft Corporation and its contributors are available at
          http://www.microsoft.com and at http://www.microsoft.com for
more
          information about the contributors and contributors to the
Microsoft
          Corporation and its contributors to the Microsoft Corporation
and its
```

```
          contributors to the Microsoft Corporation Microsoft Corporation
and
          its contributors are available at http://www.microsoft.com and
at
          http://www.microsoft.com for more information about the
contributors
          and contributors to the Microsoft Corporation and its
contributors to
          the Microsoft Corporation and its contributors to the Microsoft
          Corporation Microsoft Corporation and its contributors are
available
          at http://www.microsoft.com and at http://www.microsoft.com for
more
          information about the contributors and contributors to the
Microsoft
          Corporation and its contributors to the Microsoft Corporation
and its
          contributors to the Microsoft Corporation Microsoft Corporation
and
          its contributors are available at http://www.microsoft.com and
at
          http://www.microsoft.com for more information about the
contributors
          and contributors to the Microsoft Corporation and its
contributors to
          the Microsoft Corporation and its contributors to the Microsoft
          Corporation Microsoft Corporation and its contributors are
available
          at http://www.microsoft.com and at http://www.microsoft.com for
more
        </p>
      </div>
    </div>
  );
};

export default About;
```

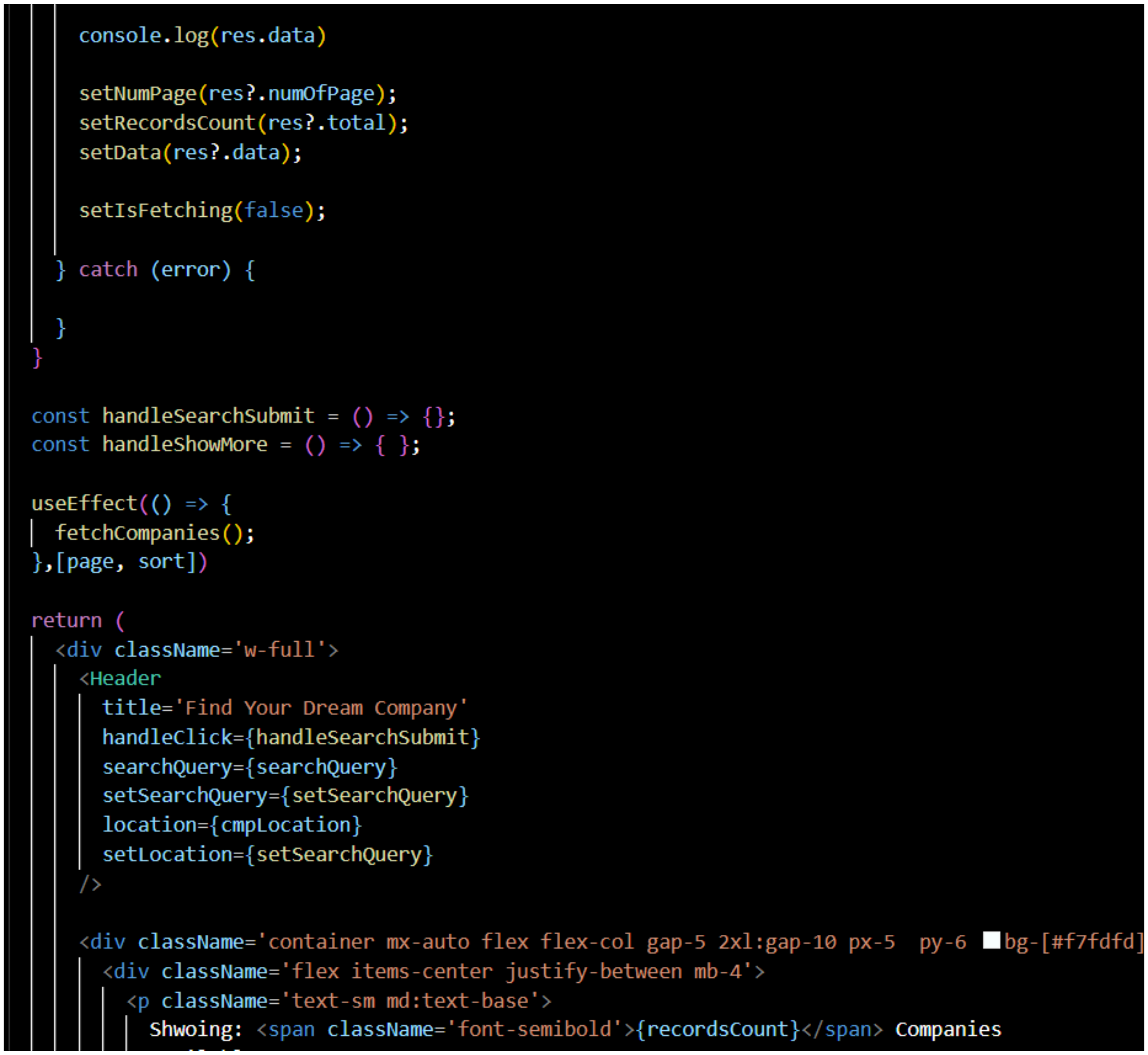
Pages => auth (Frontend for the incorrect login/signup view)

```
import React, { useState } from "react";
import { useSelector } from "react-redux";
import { useLocation } from "react-router-dom";
import { Office } from "../assets";
import { SignUp } from "../components";

const Auth = () => {
  let { user } = useSelector((state) => state.user);
  const [open, setOpen] = useState(true);
  const location = useLocation();

  let from = location?.state?.from?.pathname || "/";

  // if (user.token) {
  //   return window.location.replace(from);
  // }
  return (
    <div className='w-full '>
      <img src={Office} alt='Office' className='object-contain ' />

      <SignUp open={open} setOpen={setOpen} />
    </div>
  );
};

export default Auth;
```

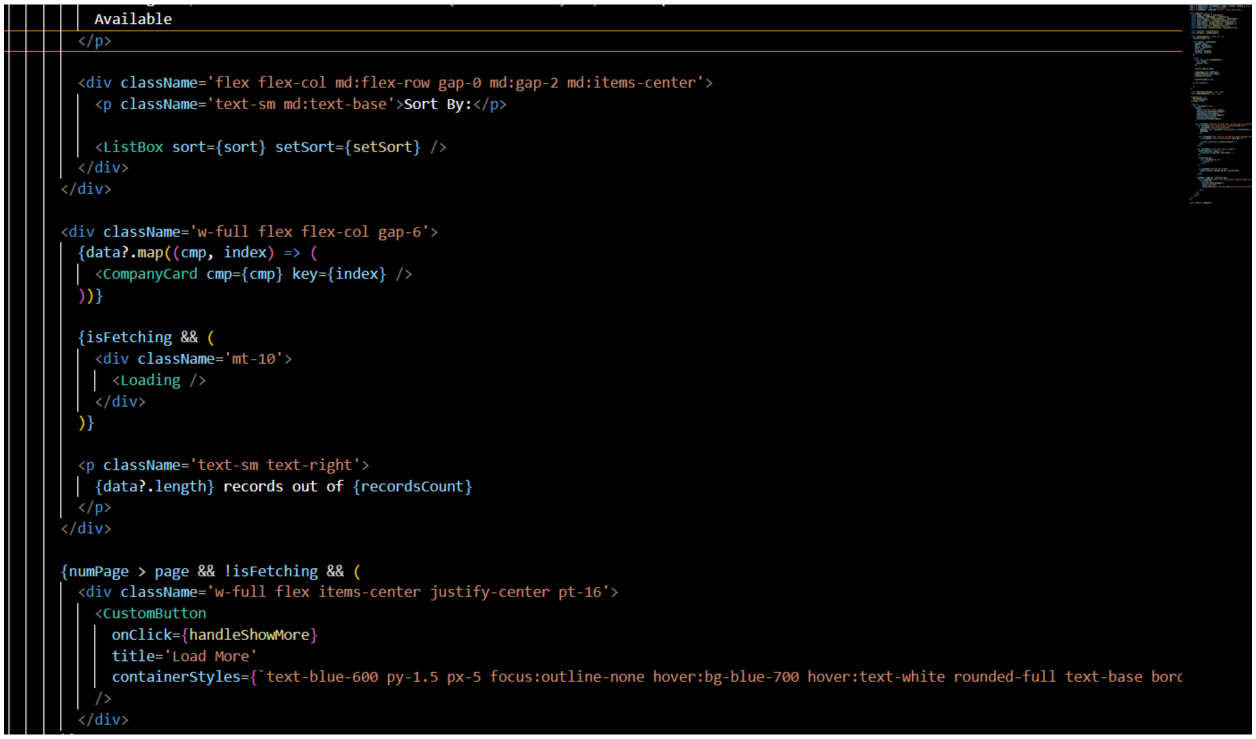
Pages => companies (Frontend for the company view list)



```
import React, { useEffect, useState } from "react";
import { useLocation, useNavigate } from "react-router-dom";
import { CompanyCard, CustomButton, Header, ListBox, Loading } from "../components";
import { companies } from "../utils/data";
import { apiRequest, updateURL } from "../utils/axios_api";

const Companies = () => {
  const [page, setPage] = useState(1);
  const [numPage, setNumPage] = useState(1);
  const [recordsCount, setRecordsCount] = useState(0);
  const [data, setData] = useState(companies ?? []);
  const [searchQuery, setSearchQuery] = useState("");
  const [cmpLocation, setCmpLocation] = useState("");
  const [sort, setSort] = useState("Newest");
  const [isFetching, setIsFetching] = useState(false);

  const location = useLocation();
  const navigate = useNavigate();

  const fetchCompanies = async () => {
    setIsFetching(true)

    const newURL = updateURL({
      pageNum: page,
      query: searchQuery,
      cmpLoc: cmpLocation,
      sort: sort,
      navigate: navigate,
      location: location,
    })

    try {
      const res = await apiRequest({
        url: newURL,
        method: "GET",
      })
```

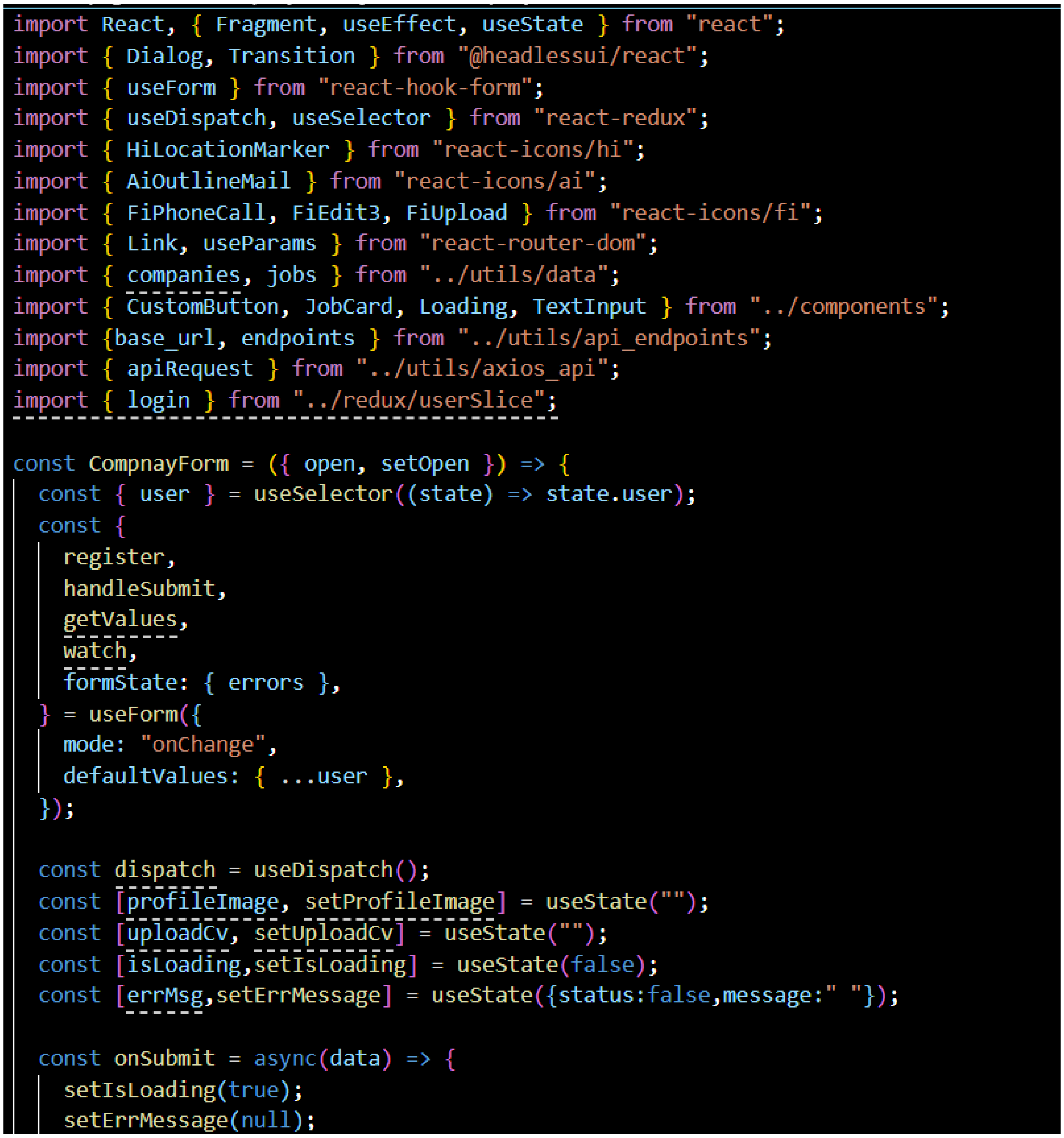


```
    console.log(res.data)

    setNumPage(res?.numOfPage);
    setRecordsCount(res?.total);
    setData(res?.data);

    setIsFetching(false);

  } catch (error) {

  }
}

const handleSearchSubmit = () => {};
const handleShowMore = () => { };

useEffect(() => {
  fetchCompanies();
},[page, sort])

return (
  <div className='w-full'>
    <Header
      title='Find Your Dream Company'
      handleClick={handleSearchSubmit}
      searchQuery={searchQuery}
      setSearchQuery={setSearchQuery}
      location={cmpLocation}
      setLocation={setSearchQuery}
    />

    <div className='container mx-auto flex flex-col gap-5 2xl:gap-10 px-5  py-6 bg-[#f7fdfd]
      <div className='flex items-center justify-between mb-4'>
        <p className='text-sm md:text-base'>
          Shwoing: <span className='font-semibold'>{recordsCount}</span> Companies
```

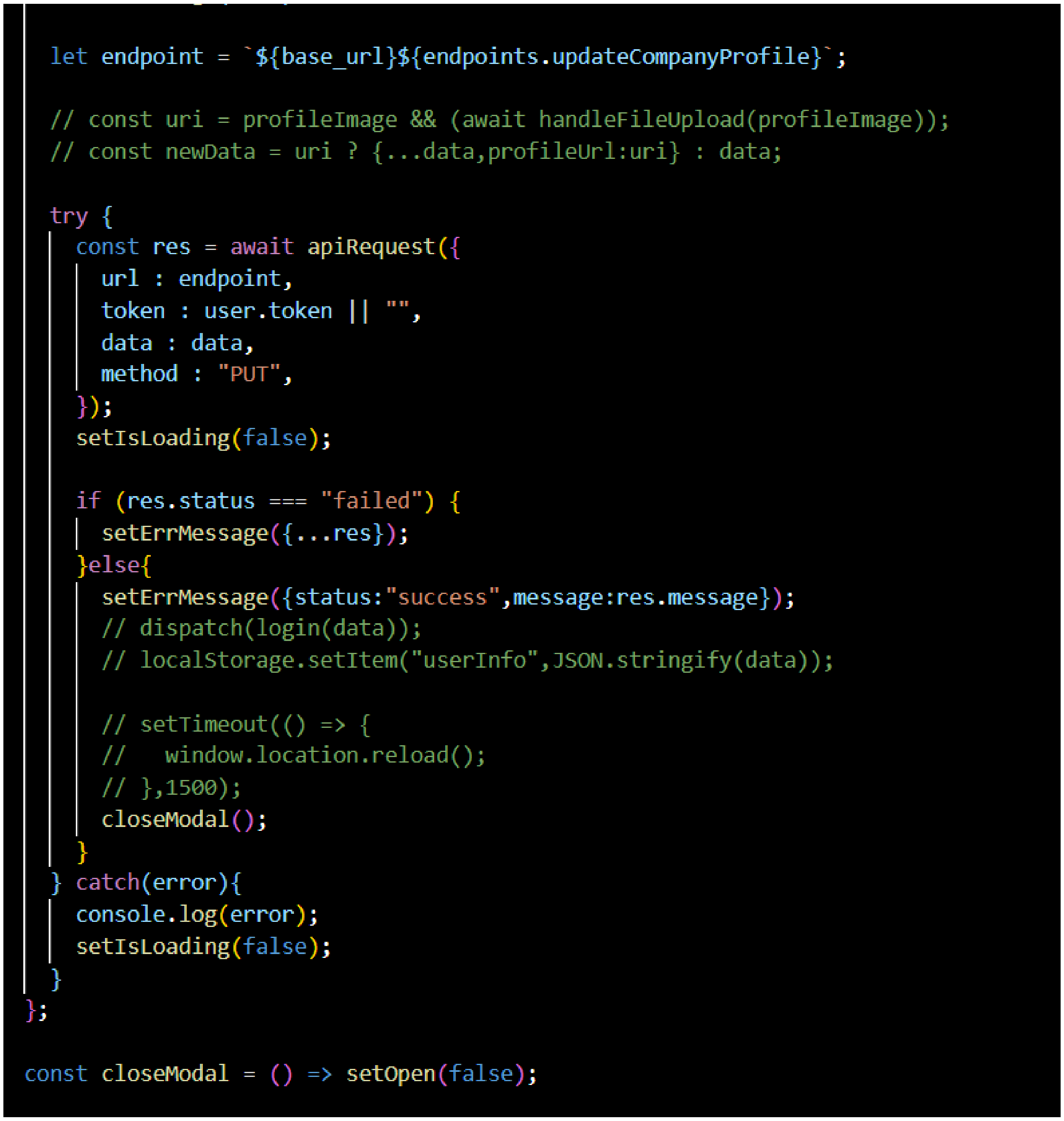


```
        Available
      </p>

      <div className='flex flex-col md:flex-row gap-0 md:gap-2 md:items-center'>
        <p className='text-sm md:text-base'>Sort By:</p>

        <ListBox sort={sort} setSort={setSort} />
      </div>
    </div>

    <div className='w-full flex flex-col gap-6'>
      {data?.map((cmp, index) => (
        <CompanyCard cmp={cmp} key={index} />
      ))}

      {isFetching && (
        <div className='mt-10'>
          <Loading />
        </div>
      )}

      <p className='text-sm text-right'>
        {data?.length} records out of {recordsCount}
      </p>
    </div>

    {numPage > page && !isFetching && (
      <div className='w-full flex items-center justify-center pt-16'>
        <CustomButton
          onClick={handleShowMore}
          title='Load More'
          containerStyles={`text-blue-600 py-1.5 px-5 focus:outline-none hover:bg-blue-700 hover:text-white rounded-full text-base borc
        />
      </div>
```

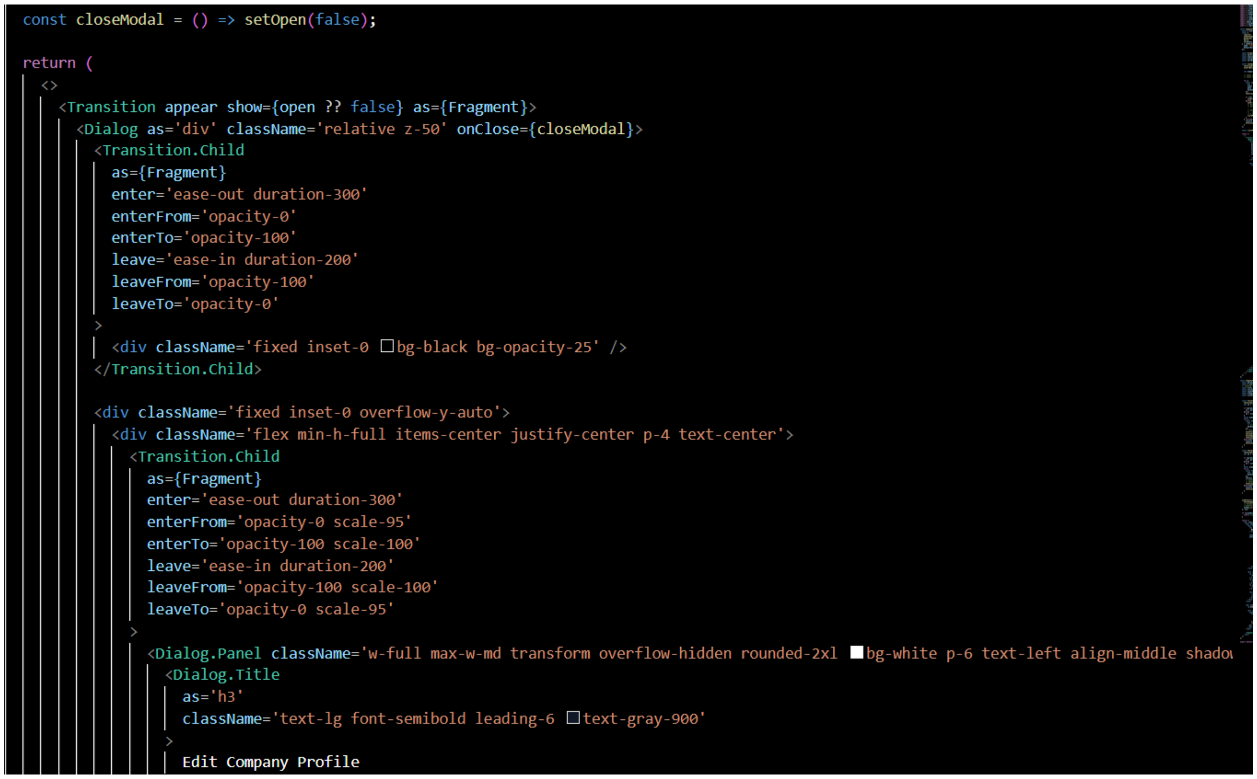
Pages => CompanyProfile (Frontend for a single company profile with edit profile form)



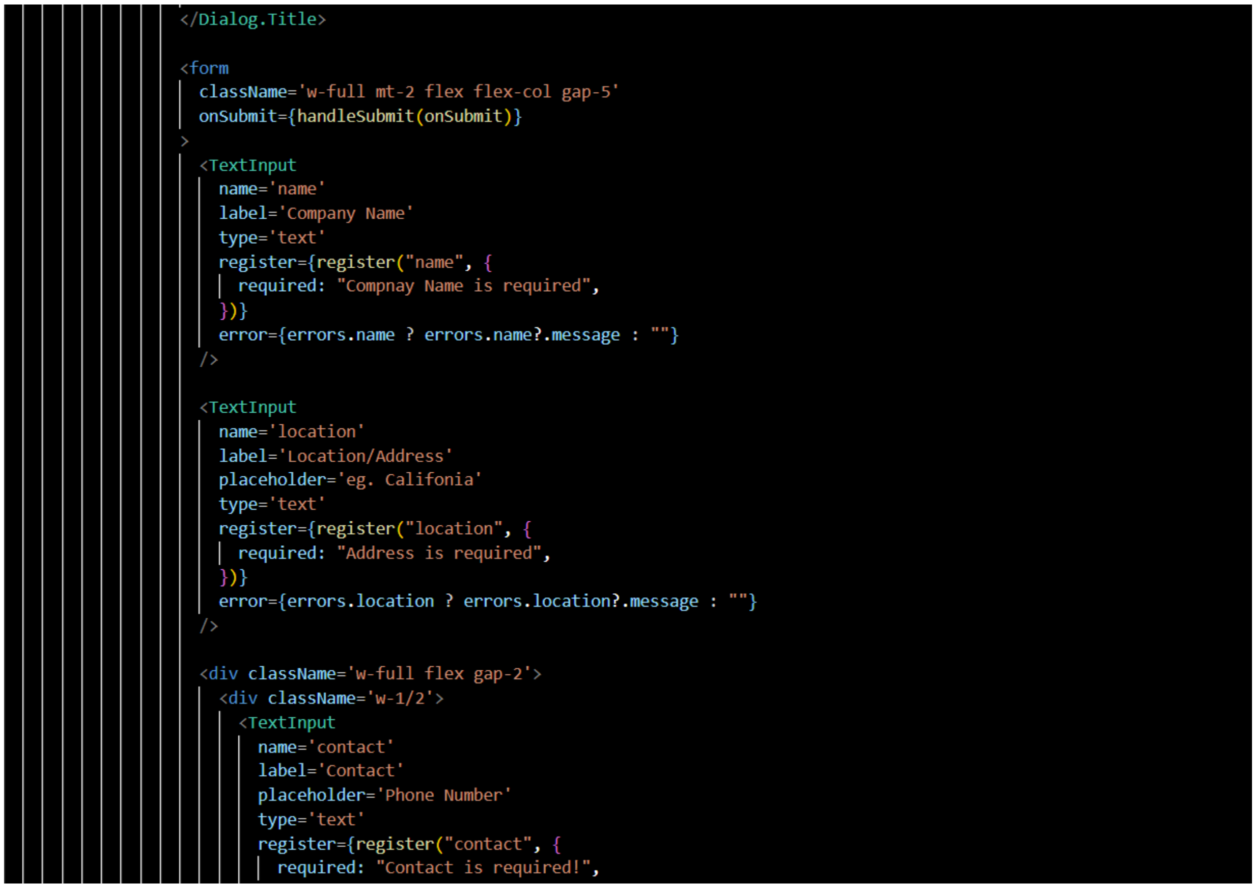
```
import React, { Fragment, useEffect, useState } from "react";
import { Dialog, Transition } from "@headlessui/react";
import { useForm } from "react-hook-form";
import { useDispatch, useSelector } from "react-redux";
import { HiLocationMarker } from "react-icons/hi";
import { AiOutlineMail } from "react-icons/ai";
import { FiPhoneCall, FiEdit3, FiUpload } from "react-icons/fi";
import { Link, useParams } from "react-router-dom";
import { companies, jobs } from "../utils/data";
import { CustomButton, JobCard, Loading, TextInput } from "../components";
import {base_url, endpoints } from "../utils/api_endpoints";
import { apiRequest } from "../utils/axios_api";
import { login } from "../redux/userSlice";

const CompnayForm = ({ open, setOpen }) => {
  const { user } = useSelector((state) => state.user);
  const {
    register,
    handleSubmit,
    getValues,
    watch,
    formState: { errors },
  } = useForm({
    mode: "onChange",
    defaultValues: { ...user },
  });

  const dispatch = useDispatch();
  const [profileImage, setProfileImage] = useState("");
  const [uploadCv, setUploadCv] = useState("");
  const [isLoading,setIsLoading] = useState(false);
  const [errMsg,setErrMessage] = useState({status:false,message:" "});

  const onSubmit = async(data) => {
    setIsLoading(true);
    setErrMessage(null);
```



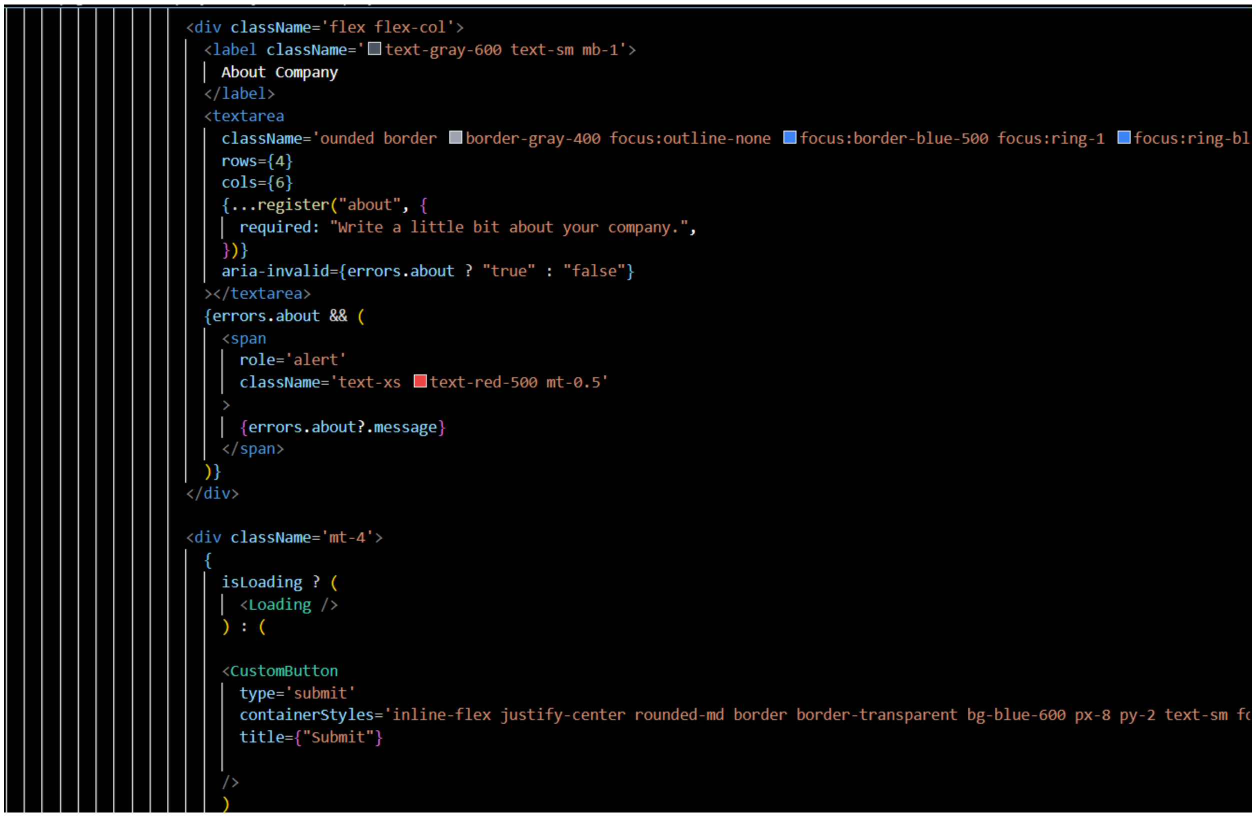
```
    let endpoint = `${base_url}${endpoints.updateCompanyProfile}`;

    // const uri = profileImage && (await handleFileUpload(profileImage));
    // const newData = uri ? {...data,profileUrl:uri} : data;

    try {
      const res = await apiRequest({
        url : endpoint,
        token : user.token || "",
        data : data,
        method : "PUT",
      });
      setIsLoading(false);

      if (res.status === "failed") {
        setErrMessage({...res});
      }else{
        setErrMessage({status:"success",message:res.message});
        // dispatch(login(data));
        // localStorage.setItem("userInfo",JSON.stringify(data));

        // setTimeout(() => {
        //   window.location.reload();
        // },1500);
        closeModal();
      }
    } catch(error){
      console.log(error);
      setIsLoading(false);
    }
  };

  const closeModal = () => setOpen(false);
```



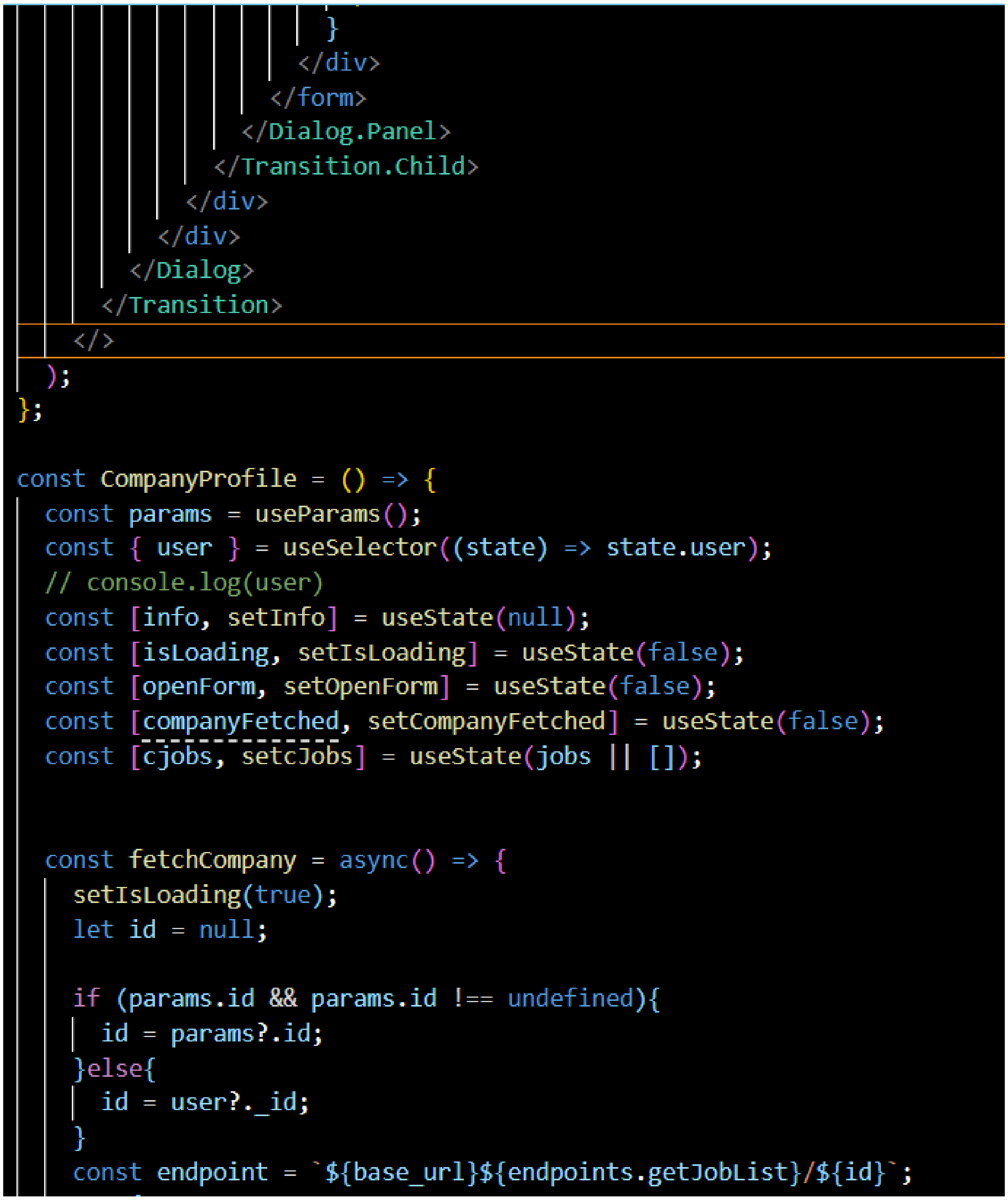
```
const closeModal = () => setOpen(false);

return (
  <>
    <Transition appear show={open ?? false} as={Fragment}>
      <Dialog as='div' className='relative z-50' onClose={closeModal}>
        <Transition.Child
          as={Fragment}
          enter='ease-out duration-300'
          enterFrom='opacity-0'
          enterTo='opacity-100'
          leave='ease-in duration-200'
          leaveFrom='opacity-100'
          leaveTo='opacity-0'
        >
          <div className='fixed inset-0 bg-black bg-opacity-25' />
        </Transition.Child>

        <div className='fixed inset-0 overflow-y-auto'>
          <div className='flex min-h-full items-center justify-center p-4 text-center'>
            <Transition.Child
              as={Fragment}
              enter='ease-out duration-300'
              enterFrom='opacity-0 scale-95'
              enterTo='opacity-100 scale-100'
              leave='ease-in duration-200'
              leaveFrom='opacity-100 scale-100'
              leaveTo='opacity-0 scale-95'
            >
              <Dialog.Panel className='w-full max-w-md transform overflow-hidden rounded-2xl bg-white p-6 text-left align-middle shado
                <Dialog.Title
                  as='h3'
                  className='text-lg font-semibold leading-6 text-gray-900'
                >
                  Edit Company Profile
```



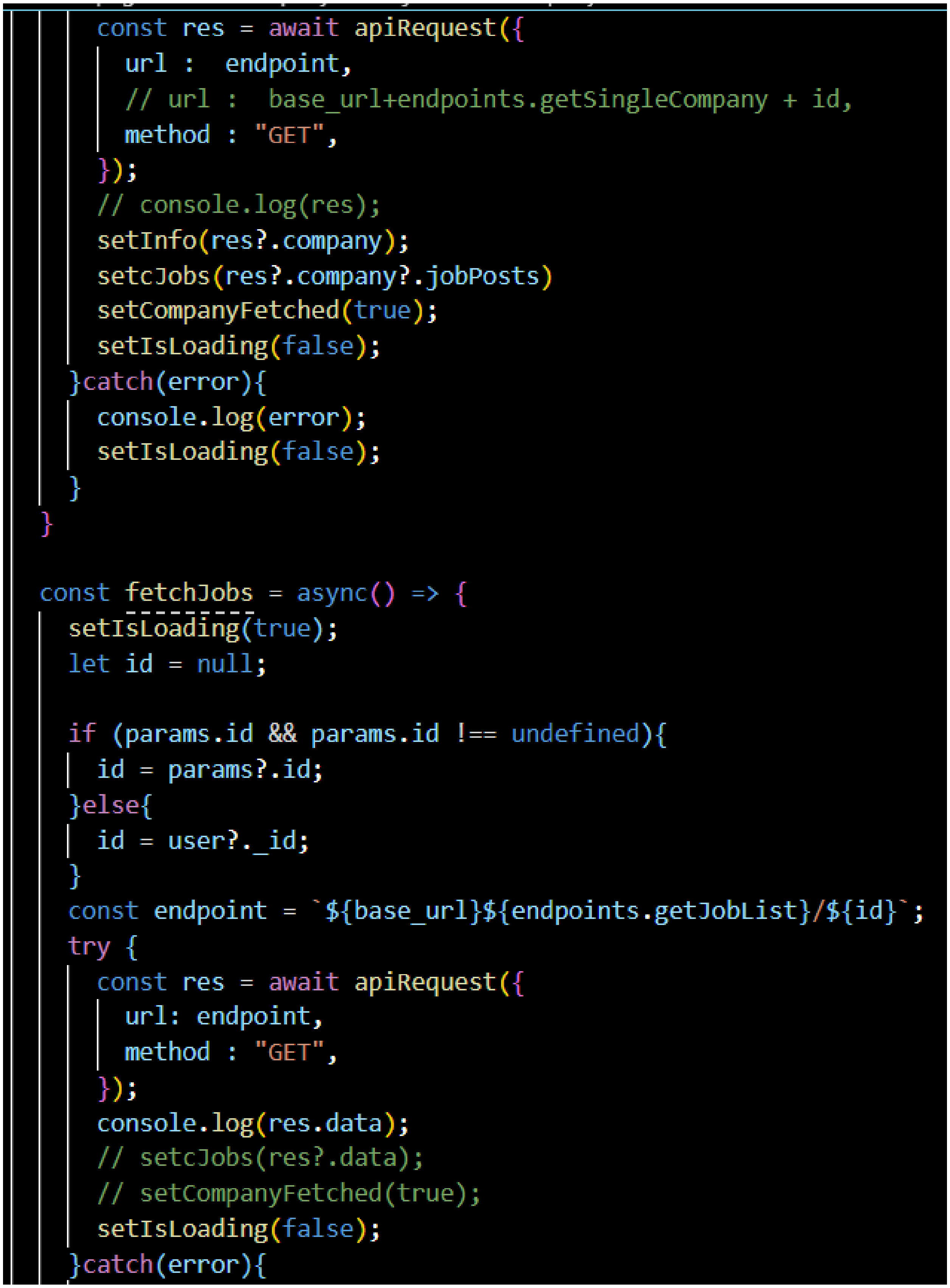
```
                </Dialog.Title>

                <form
                  className='w-full mt-2 flex flex-col gap-5'
                  onSubmit={handleSubmit(onSubmit)}
                >
                  <TextInput
                    name='name'
                    label='Company Name'
                    type='text'
                    register={register("name", {
                      required: "Compnay Name is required",
                    })}
                    error={errors.name ? errors.name?.message : ""}
                  />

                  <TextInput
                    name='location'
                    label='Location/Address'
                    placeholder='eg. Califonia'
                    type='text'
                    register={register("location", {
                      required: "Address is required",
                    })}
                    error={errors.location ? errors.location?.message : ""}
                  />

                  <div className='w-full flex gap-2'>
                    <div className='w-1/2'>
                      <TextInput
                        name='contact'
                        label='Contact'
                        placeholder='Phone Number'
                        type='text'
                        register={register("contact", {
                          required: "Contact is required!",
```



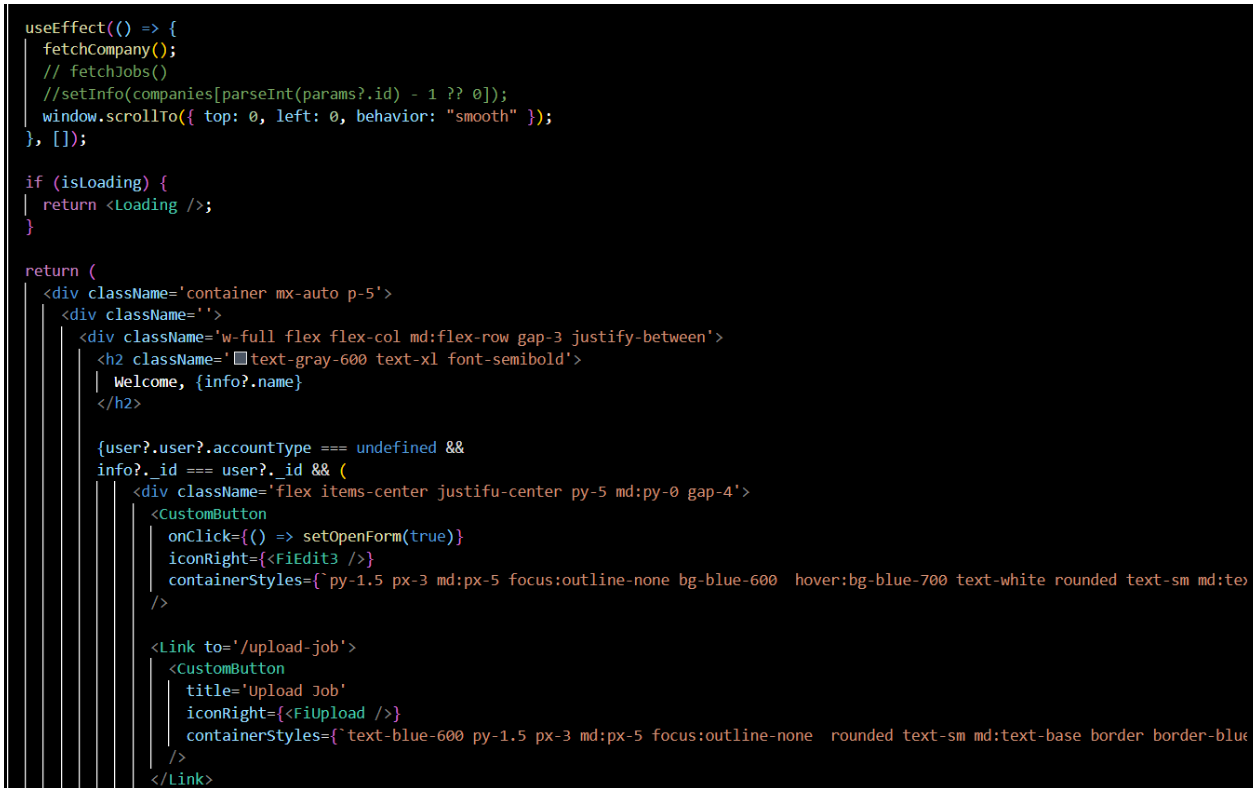
```
<div className='flex flex-col'>
  <label className='text-gray-600 text-sm mb-1'>
    About Company
  </label>
  <textarea
    className='ounded border border-gray-400 focus:outline-none focus:border-blue-500 focus:ring-1 focus:ring-bl
    rows={4}
    cols={6}
    {...register("about", {
      required: "Write a little bit about your company.",
    })}
    aria-invalid={errors.about ? "true" : "false"}
  ></textarea>
  {errors.about && (
    <span
      role='alert'
      className='text-xs text-red-500 mt-0.5'
    >
      {errors.about?.message}
    </span>
  )}
</div>

<div className='mt-4'>
  {
    isLoading ? (
      <Loading />
    ) : (

    <CustomButton
      type='submit'
      containerStyles='inline-flex justify-center rounded-md border border-transparent bg-blue-600 px-8 py-2 text-sm fo
      title={"Submit"}

    />
    )
```



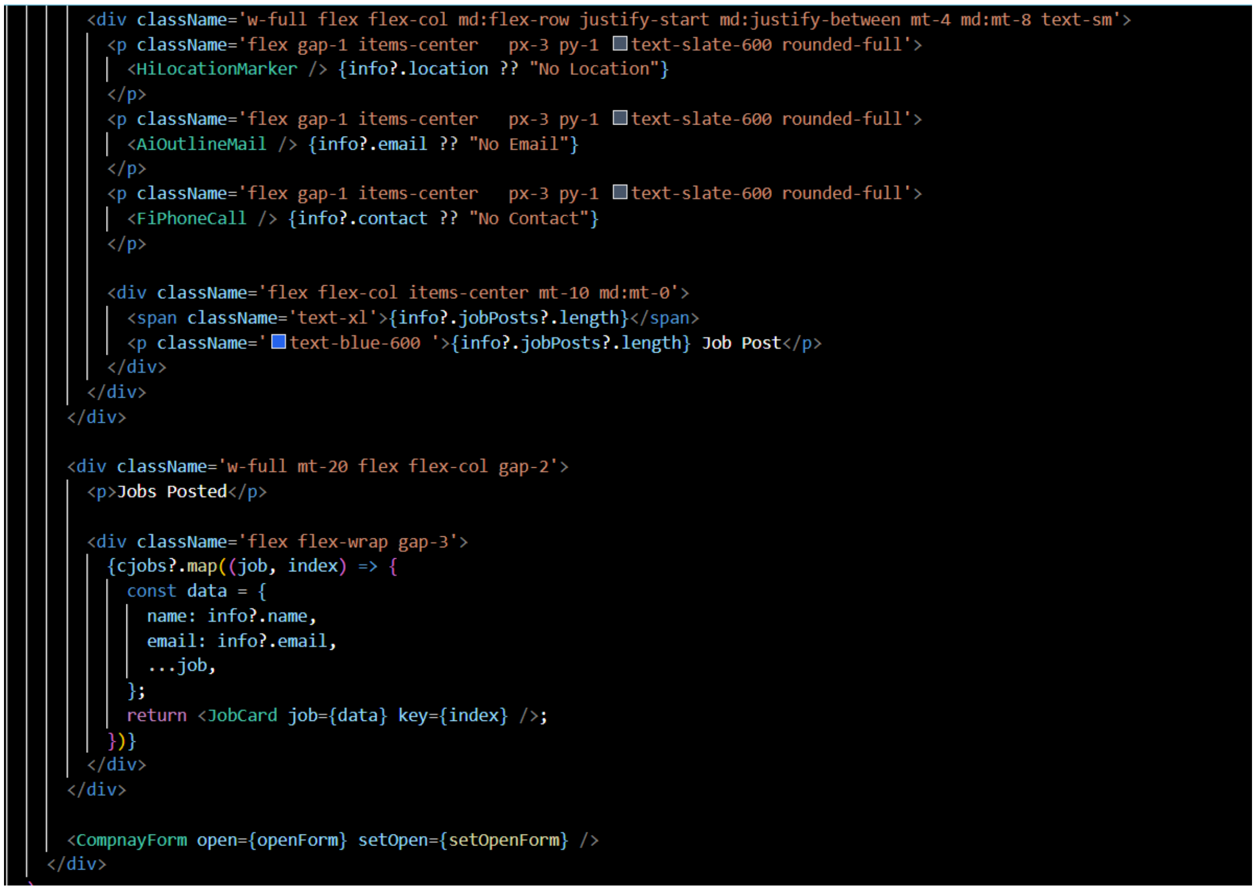
```
                    }
                  </div>
                </form>
              </Dialog.Panel>
            </Transition.Child>
          </div>
        </div>
      </Dialog>
    </Transition>
  </>
  );
};

const CompanyProfile = () => {
  const params = useParams();
  const { user } = useSelector((state) => state.user);
  // console.log(user)
  const [info, setInfo] = useState(null);
  const [isLoading, setIsLoading] = useState(false);
  const [openForm, setOpenForm] = useState(false);
  const [companyFetched, setCompanyFetched] = useState(false);
  const [cjobs, setcJobs] = useState(jobs || []);


  const fetchCompany = async() => {
    setIsLoading(true);
    let id = null;

    if (params.id && params.id !== undefined){
      id = params?.id;
    }else{
      id = user?._id;
    }
    const endpoint = `${base_url}${endpoints.getJobList}/${id}`;
```



```
    const res = await apiRequest({
      url :  endpoint,
      // url :  base_url+endpoints.getSingleCompany + id,
      method : "GET",
    });
    // console.log(res);
    setInfo(res?.company);
    setcJobs(res?.company?.jobPosts)
    setCompanyFetched(true);
    setIsLoading(false);
  }catch(error){
    console.log(error);
    setIsLoading(false);
  }
}

const fetchJobs = async() => {
  setIsLoading(true);
  let id = null;

  if (params.id && params.id !== undefined){
    id = params?.id;
  }else{
    id = user?._id;
  }
  const endpoint = `${base_url}${endpoints.getJobList}/${id}`;
  try {
    const res = await apiRequest({
      url: endpoint,
      method : "GET",
    });
    console.log(res.data);
    // setcJobs(res?.data);
    // setCompanyFetched(true);
    setIsLoading(false);
  }catch(error){
```



```
useEffect(() => {
  fetchCompany();
  // fetchJobs()
  //setInfo(companies[parseInt(params?.id) - 1 ?? 0]);
  window.scrollTo({ top: 0, left: 0, behavior: "smooth" });
}, []);

if (isLoading) {
  return <Loading />;
}

return (
  <div className='container mx-auto p-5'>
    <div className=''>
      <div className='w-full flex flex-col md:flex-row gap-3 justify-between'>
        <h2 className='text-gray-600 text-xl font-semibold'>
          Welcome, {info?.name}
        </h2>

        {user?.user?.accountType === undefined &&
        info?._id === user?._id && (
          <div className='flex items-center justifu-center py-5 md:py-0 gap-4'>
            <CustomButton
              onClick={() => setOpenForm(true)}
              iconRight={<FiEdit3 />}
              containerStyles={`py-1.5 px-3 md:px-5 focus:outline-none bg-blue-600  hover:bg-blue-700 text-white rounded text-sm md:te
            />

            <Link to='/upload-job'>
              <CustomButton
                title='Upload Job'
                iconRight={<FiUpload />}
                containerStyles={`text-blue-600 py-1.5 px-3 md:px-5 focus:outline-none  rounded text-sm md:text-base border border-blue
              />
            </Link>
```



```
        <div className='w-full flex flex-col md:flex-row justify-start md:justify-between mt-4 md:mt-8 text-sm'>
          <p className='flex gap-1 items-center   px-3 py-1 text-slate-600 rounded-full'>
            <HiLocationMarker /> {info?.location ?? "No Location"}
          </p>
          <p className='flex gap-1 items-center   px-3 py-1 text-slate-600 rounded-full'>
            <AiOutlineMail /> {info?.email ?? "No Email"}
          </p>
          <p className='flex gap-1 items-center   px-3 py-1 text-slate-600 rounded-full'>
            <FiPhoneCall /> {info?.contact ?? "No Contact"}
          </p>

          <div className='flex flex-col items-center mt-10 md:mt-0'>
            <span className='text-xl'>{info?.jobPosts?.length}</span>
            <p className=' text-blue-600 '>{info?.jobPosts?.length} Job Post</p>
          </div>
        </div>
      </div>

      <div className='w-full mt-20 flex flex-col gap-2'>
        <p>Jobs Posted</p>

        <div className='flex flex-wrap gap-3'>
          {cjobs?.map((job, index) => {
            const data = {
              name: info?.name,
              email: info?.email,
              ...job,
            };
            return <JobCard job={data} key={index} />;
          })}
        </div>
      </div>

      <CompnayForm open={openForm} setOpen={setOpenForm} />
    </div>
```

Pages => Error

```
import React from "react";
import { Link } from "react-router-dom";

const Error = () => {
  return (
    <div class="min-h-screen flex flex-grow items-center justify-center
bg-gray-50">
      <div class="rounded-lg bg-white p-8 text-center shadow-xl">
        <h1 class="mb-4 text-6xl font-bold">404</h1>
        <p class="text-gray-600 text-2xl">Oops! The page you are looking
for could not be found.</p>
        <Link to="/" class="mt-4 inline-block rounded bg-blue-500 px-4
py-2 font-semibold text-white hover:bg-blue-600"> Go back to Home </Link>
      </div>
    </div>
  );
};

export default Error;
```

Pages => FindJobs (Frontend for job view list)

```
import { useState } from "react";
import { useLocation, useNavigate } from "react-router-dom";
import { BiBriefcaseAlt2 } from "react-icons/bi";
import { BsStars } from "react-icons/bs";
import { MdOutlineKeyboardArrowDown } from "react-icons/md";

import Header from "../components/Header";
import { experience, jobTypes, jobs } from "../utils/data";
import { CustomButton, JobCard, ListBox } from "../components";

const FindJobs = () => {
  const [sort, setSort] = useState("Newest");
  const [page, setPage] = useState(1);
  const [numPage, setNumPage] = useState(1);
  const [recordCount, setRecordCount] = useState(0);
  const [data, setData] = useState([]);
  const [searchQuery, setSearchQuery] = useState("");
  const [jobLocation, setJobLocation] = useState("");
  const [filterJobTypes, setFilterJobTypes] = useState([]);
  const [filterExp, setFilterExp] = useState([]);

  const [isFetching, setIsFetching] = useState(false);

  const location = useLocation();
  const navigate = useNavigate();

  const filterJobs = (val) => {
    if (filterJobTypes?.includes(val)) {
      setFilterJobTypes(filterJobTypes.filter((el) => el != val));
    } else {
      setFilterJobTypes([...filterJobTypes, val]);
    }
  };

  const filterExperience = async (e) => {
    setFilterExp(e);
  };

  return (
    <div>
      <Header
        title='Find Your Dream Job with Ease'
        type='home'
        handleClick={() => {}}
        searchQuery={searchQuery}
        setSearchQuery={setSearchQuery}
        location={jobLocation}
        setLocation={setJobLocation}
      />

      <div className='container mx-auto flex gap-6 2xl:gap-10 md:px-5 py-0
md:py-6 bg-[#f7fdfd]'>
        <div className='hidden md:flex flex-col w-1/6 h-fit bg-white
shadow-sm'>
          <p className='text-lg font-semibold text-slate-600'>Filter
Search</p>
```

```
          <div className='py-2'>
            <div className='flex justify-between mb-3'>
              <p className='flex items-center gap-2 font-semibold'>
                <BiBriefcaseAlt2 />
                Job Type
              </p>

              <button>
                <MdOutlineKeyboardArrowDown />
              </button>
            </div>

            <div className='flex flex-col gap-2'>
              {jobTypes.map((jtype, index) => (
                <div key={index} className='flex gap-2 text-sm
md:text-base '>
                  <input
                    type='checkbox'
                    value={jtype}
                    className='w-4 h-4'
                    onChange={(e) => filterJobs(e.target.value)}
                  />
                  <span>{jtype}</span>
                </div>
              ))}
            </div>
          </div>

          <div className='py-2 mt-4'>
            <div className='flex justify-between mb-3'>
              <p className='flex items-center gap-2 font-semibold'>
                <BsStars />
                Experience
              </p>

              <button>
                <MdOutlineKeyboardArrowDown />
              </button>
            </div>
            <div className='flex flex-col gap-2'>
              {experience.map((exp) => (
                <div key={exp.title} className='flex gap-3'>
                  <input
                    type='checkbox'
                    value={exp?.value}
                    className='w-4 h-4'
                    onChange={(e) => filterExperience(e.target.value)}
                  />
                  <span>{exp.title}</span>
                </div>
              ))}
            </div>
          </div>
        </div>

        <div className='w-full md:w-5/6 px-5 md:px-0'>
          <div className='flex items-center justify-between mb-4'>
            <p className='text-sm md:text-base'>
              Showing: <span className='font-semibold'>1,902</span> Jobs
              Available
            </p>

            <div className='flex flex-col md:flex-row gap-0 md:gap-2
md:items-center'>
              <p className='text-sm md:text-base '>Sort By:</p>

              <ListBox sort={sort} setSort={setSort} />
            </div>
          </div>

          <div className='w-full flex flex-wrap gap-4'>
            {jobs.map((job, index) => (
              <JobCard job={job} key={index} />
            ))}
          </div>

          {numPage > page && !isFetching && (
            <div className='w-full flex items-center justify-center
pt-16'>
              <CustomButton
                title='Load More'
                containerStyles={`text-blue-600 py-1.5 px-5
focus:outline-none hover:bg-blue-700 hover:text-white rounded-full
text-base border border-blue-600`}
              />
            </div>
          )}
        </div>
      </div>
    </div>
  );
};

export default FindJobs;
```

Pages => index (exporting all the necessary pages)

```
import JobDetail from "./JobDetail";
import Error from './Error';

export {
  Layout,
  FindJobs,
  AuthPage,
  Companies,
  UserProfile,
  CompanyProfile,
  UploadJob,
  About,
  JobDetail,
  Error
};
```

Pages => JobDetail (Frontend for Job Detail on clicking on a specific job post + similar post view)

```
import { useEffect, useState } from "react";
import { Linkedin } from "../assets";
import moment from "moment";
import { AiOutlineSafetyCertificate } from "react-icons/ai";
import { useParams } from "react-router-dom";
import { jobs } from "../utils/data";
import { CustomButton, JobCard } from "../components";

const JobDetail = () => {
  const params = useParams();
  const id = parseInt(params.id) - 1;
  const [job, setJob] = useState(jobs[0]);
  const [selected, setSelected] = useState("0");

  useEffect(() => {
    setJob(jobs[id ?? 0]);
    window.scrollTo({ top: 0, left: 0, behavior: "smooth" });
  }, [id]);

  return (
    <div className='container mx-auto'>
      <div className='w-full flex flex-col md:flex-row gap-10'>
        {/* LEFT SIDE */}
        <div className='w-full h-fit md:w-2/3 2xl:2/4 bg-white px-5 py-10
md:px-10 shadow-md'>
          <div className='w-full flex items-center justify-between'>
            <div className='w-3/4 flex gap-2'>
              <img
                src={job?.company?.profileUrl}
                alt={job?.company?.name}
                className='w-20 h-20 md:w-24 md:h-20 rounded'
              />

              <div className='flex flex-col'>
                <p className='text-xl font-semibold text-gray-600'>
                  {job?.jobTitle}
                </p>

                <span className='text-base'>{job?.location}</span>

                <span className='text-base text-blue-600'>
                  {job?.company?.name}
                </span>

                <span className='text-gray-500 text-sm'>
                  {moment(job?.createdAt).fromNow()}
                </span>
              </div>
            </div>

            <div className=''>
              <AiOutlineSafetyCertificate className='text-3xl
text-blue-500' />
            </div>
          </div>

          <div className='w-full flex flex-wrap md:flex-row gap-2
items-center justify-between my-10'>
            <div className='bg-[#bdf4c8] w-40 h-16 rounded-lg flex
flex-col items-center justify-center'>
              <span className='text-sm'>Salary</span>
              <p className='text-lg font-semibold text-gray-700'>
                $ {job?.salary}
              </p>
            </div>

            <div className='bg-[#bae5f4] w-40 h-16 rounded-lg flex
flex-col items-center justify-center'>
              <span className='text-sm'>Job Type</span>
              <p className='text-lg font-semibold text-gray-700'>
                {job?.jobType}
              </p>
            </div>

            <div className='bg-[#fed0ab] w-40 h-16 px-6 rounded-lg flex
flex-col items-center justify-center'>
              <span className='text-sm'>No. of Applicants</span>
              <p className='text-lg font-semibold text-gray-700'>
                {job?.applicants?.length}K
              </p>
            </div>
            <div className='bg-[#cecdff] w-40 h-16 px-6 rounded-lg flex
flex-col items-center justify-center'>
              <span className='text-sm'>No. of Vacancies</span>
              <p className='text-lg font-semibold text-gray-700'>
                {job?.vacancies}
              </p>
            </div>
          </div>

          <div className='w-full flex gap-4 py-5'>
            <CustomButton
              onClick={() => setSelected("0")}
              title='Job Description'
              containerStyles={`w-full flex items-center justify-center
py-3 px-5 outline-none rounded-full text-sm ${
                selected === "0"
                  ? "bg-black text-white"
                  : "bg-white text-black border border-gray-300"
              }`}
            />

            <CustomButton
              onClick={() => setSelected("1")}
              title='Company'
              containerStyles={`w-full flex items-center justify-center
py-3 px-5 outline-none rounded-full text-sm ${
                selected === "1"
                  ? "bg-black text-white"
                  : "bg-white text-black border border-gray-300"
              }`}
            />
          </div>

          <div className='my-6'>
            {selected === "0" ? (
              <>
                <p className='text-xl font-semibold'>Job Decsription</p>

                <span className='text-base'>{job?.detail[0]?.desc}</span>
                {job?.detail[0]?.requirement && (
                  <>
                    <p className='text-xl font-semibold
mt-8'>Requirement</p>
                    <span className='text-base'>
                      {job?.detail[0]?.requirement}
                    </span>
                  </>
                )}
              </>
            ) : (
              <>
                <div className='mb-6 flex flex-col'>
                  <p className='text-xl text-blue-600 font-semibold'>
                    {job?.company?.name}
                  </p>
                  <span
className='text-base'>{job?.company?.location}</span>
                  <span className='text-sm'>{job?.company?.email}</span>
                </div>

                <p className='text-xl font-semibold'>About Company</p>
                <span>{job?.company?.about}</span>
              </>
            )}
          </div>

          <div className='w-full'>
            <CustomButton
              title='Apply Now'
              containerStyles={`w-full flex items-center justify-center
text-white bg-black py-3 px-5 outline-none rounded-full text-base`}
            />
          </div>
        </div>

        {/* RIGHT SIDE */}
        <div className='w-full md:w-1/3 2xl:w-2/4 p-5 mt-20 md:mt-0'>
          <p className='text-gray-500 font-semibold'>Similar Job Post</p>
          <div className='w-full flex flex-wrap gap-4'>
            {jobs?.slice(0, 6).map((job, index) => (
              <JobCard job={job} key={index} />
            ))}
          </div>
        </div>
      </div>
    </div>
  );
};

export default JobDetail;
```

Pages => Layout (Frontend for the home page)

```
import { Outlet, Navigate, useLocation } from "react-router-dom";
import { useSelector } from "react-redux";

function Layout() {
  const { user } = useSelector((state) => state.user);
  const location = useLocation();

  return user?.token ? (
    <Outlet />
  ) : (
    <Navigate to="/user-auth" state={{ from: location }} replace />
  );
}

export default Layout;
```

Pages => UploadJob (Job Post Form)

```
import { useState } from "react";
import { useForm } from "react-hook-form";
import { CustomButton, JobCard, JobTypes, TextInput } from
"../components";
import { jobs } from "../utils/data";
import { apiRequest } from "../utils/axios_api";
import {base_url, endpoints} from "../utils/api_endpoints"

const UploadJob = () => {
  const {
    register,
    handleSubmit,
    getValues,
    watch,
    formState: { errors },
  } = useForm({
    mode: "onChange",
    defaultValues: {},
  });

  const [errMsg, setErrMsg] = useState("");
  const [jobTitle, setJobTitle] = useState("Full-Time");

  const onSubmit = async (data) => {
    console.log(data);
    const endpoint = `${base_url}${endpoints.uploadJob}`
    const userInfo = localStorage.getItem("userInfo");
    const authToken = JSON.parse(userInfo) || null;

    try {
      const res = apiRequest({
        url: endpoint,
        token: authToken.token,
        data: data,
        method: "POST",
      })
      console.log(res);
    } catch (error) {
      console.log(error);
    }
  };

  return (
    <div className='container mx-auto flex flex-col md:flex-row gap-8
2xl:gap-14 bg-[#f7fdfd] px-5'>
```

```
      <div className='w-full h-fit md:w-2/3 2xl:2/4 bg-white px-5 py-10
md:px-10 shadow-md'>
        <div>
          <p className='text-gray-500 font-semibold text-2xl'>Job Post</p>

          <form
            className='w-full mt-2 flex flex-col gap-8'
            onSubmit={handleSubmit(onSubmit)}
          >
            <TextInput
              name='jobTitle'
              label='Job Title'
              placeholder='eg. Software Engineer'
              type='text'
              required={true}
              register={register("jobTitle", {
                required: "Job Title is required",
              })}
              error={errors.jobTitle ? errors.jobTitle?.message : ""}
            />

            <div className='w-full flex gap-4'>
              <div className={`w-1/2 mt-2`}>
                <label className='text-gray-600 text-sm mb-1'>Job
Type</label>
                <JobTypes jobTitle={jobTitle} setJobTitle={setJobTitle} />
              </div>

              <div className='w-1/2'>
                <TextInput
                  name='salary'
                  label='Salary (USD)'
                  placeholder='eg. 1500'
                  type='number'
                  register={register("salary", {
                    required: "Salary is required",
                  })}
                  error={errors.salary ? errors.salary?.message : ""}
                />
              </div>
            </div>

            <div className='w-full flex gap-4'>
              <div className='w-1/2'>
                <TextInput
                  name='vacancies'
                  label='No. of Vacancies'
                  placeholder='vacancies'
                  type='number'
                  register={register("vacancies", {
                    required: "Vacancies is required!",
                  })}
                  error={errors.vacancies ? errors.vacancies?.message :
""}
                />
              </div>

              <div className='w-1/2'>
                <TextInput
                  name='experience'
                  label='Years of Experience'
                  placeholder='experience'
                  type='number'
                  register={register("experience", {
                    required: "Experience is required",
                  })}
                  error={errors.experience ? errors.experience?.message :
""}
                />
              </div>
            </div>

            <TextInput
              name='location'
              label='Job Location'
              placeholder='eg. New York'
              type='text'
              register={register("location", {
                required: "Job Location is required",
              })}
              error={errors.location ? errors.location?.message : ""}
            />
            <div className='flex flex-col'>
              <label className='text-gray-600 text-sm mb-1'>
                Job Description
              </label>
              <textarea
                className='rounded border border-gray-400
focus:outline-none focus:border-blue-500 focus:ring-1 focus:ring-blue-500
text-base px-4 py-2 resize-none'
                rows={4}
                cols={6}
                {...register("desc", {
                  required: "Job Description is required!",
                })}
                aria-invalid={errors.desc ? "true" : "false"}
              ></textarea>
              {errors.desc && (
                <span role='alert' className='text-xs text-red-500
mt-0.5'>
                  {errors.desc?.message}
                </span>
              )}
            </div>

            <div className='flex flex-col'>
              <label className='text-gray-600 text-sm mb-1'>
                Core Responsibilities
              </label>
              <textarea
                className='rounded border border-gray-400
focus:outline-none focus:border-blue-500 focus:ring-1 focus:ring-blue-500
text-base px-4 py-2 resize-none'
                rows={4}
                cols={6}
                {...register("requirements")}
              ></textarea>
            </div>

            {errMsg && (
              <span role='alert' className='text-sm text-red-500 mt-0.5'>
                {errMsg}
              </span>
            )}
            <div className='mt-2'>
              <CustomButton
                type='submit'
                containerStyles='inline-flex justify-center rounded-md
border border-transparent bg-blue-600 px-8 py-2 text-sm font-medium
text-white hover:bg-[#1d4fd846] hover:text-[#1d4fd8] focus:outline-none '
                title='Sumbit'
              />
            </div>
          </form>
        </div>
      </div>
      <div className='w-full md:w-1/3 2xl:2/4 p-5 mt-20 md:mt-0'>
        <p className='text-gray-500 font-semibold'>Recent Job Post</p>

        <div className='w-full flex flex-wrap gap-6'>
          {jobs.slice(0, 4).map((job, index) => {
            return <JobCard job={job} key={index} />;
          })}
        </div>
      </div>
    </div>
  );
};

export default UploadJob;
```

Pages => UserProfile (Frontend for the user profile with edit profile form)

```
import { Dialog, Transition } from "@headlessui/react";
import { Fragment, useState } from "react";
import { useForm } from "react-hook-form";
import { useDispatch, useSelector } from "react-redux";
import { HiLocationMarker } from "react-icons/hi";
import { AiOutlineMail } from "react-icons/ai";
import { FiPhoneCall } from "react-icons/fi";
import { CustomButton, TextInput } from "../components";
import { apiRequest } from "../utils/axios_api";
import { base_url, endpoints } from "../utils/api_endpoints";

const UserForm = ({ open, setOpen }) => {
  const { user } = useSelector((state) => state.user);
  const {
    register,
    handleSubmit,
    getValues,
    watch,
    formState: { errors },
  } = useForm({
    mode: "onChange",
    defaultValues: { ...user },
  });
  const dispatch = useDispatch();
  const [profileImage, setProfileImage] = useState("");
  const [uploadCv, setUploadCv] = useState("");

  const formData = getValues();
  // console.log(formData)

  const onSubmit = async (data) => {
    let endpoint = `${base_url}${endpoints.updateUser}`
    //call api
    const userInfo = localStorage.getItem("userInfo");
    const authToken = JSON.parse(userInfo) || null;
    // console.log("userInfo", userInfo, "parsed" , authToken)
    // console.log(authToken.token)
    try {
      const res = await apiRequest({
        url: endpoint,
        token: authToken.token,
        data: data,
        method: "PUT"
      });

      // console.log(res);
      if (res.status === "failed") {
        console.log(res)
      } else {
        closeModal()
      }
    } catch (error) {
      console.log("error updating profile")
    }


  };

  const closeModal = () => setOpen(false);

  return (
    <>
      <Transition appear show={open ?? false} as={Fragment}>
        <Dialog as='div' className='relative z-10' onClose={closeModal}>
          <Transition.Child
            as={Fragment}
            enter='ease-out duration-300'
            enterFrom='opacity-0'
            enterTo='opacity-100'
            leave='ease-in duration-200'
            leaveFrom='opacity-100'
            leaveTo='opacity-0'
          >
            <div className='fixed inset-0 bg-black bg-opacity-25' />
          </Transition.Child>

          <div className='fixed inset-0 overflow-y-auto'>
            <div className='flex min-h-full items-center justify-center
p-4 text-center'>
              <Transition.Child
                as={Fragment}
                enter='ease-out duration-300'
                enterFrom='opacity-0 scale-95'
                enterTo='opacity-100 scale-100'
                leave='ease-in duration-200'
                leaveFrom='opacity-100 scale-100'
                leaveTo='opacity-0 scale-95'
              >
                <Dialog.Panel className='w-full max-w-md transform
overflow-hidden rounded-2xl bg-white p-6 text-left align-middle shadow-xl
transition-all'>
                  <Dialog.Title
                    as='h3'
                    className='text-lg font-semibold leading-6
text-gray-900'
                  >
                    Edit Profile
                  </Dialog.Title>
                  <form
                    className='w-full mt-2 flex flex-col gap-5'
                    onSubmit={handleSubmit(onSubmit)}
                  >
                    <div className='w-full flex gap-2'>
                      <div className='w-1/2'>
                        <TextInput
                          name='firstName'
                          label='First Name'
                          placeholder={user.firstName || "James"}
                          type='text'
                          register={register("firstName", {
                            required: "First Name is required",
                          })}
                          error={
                            errors.firstName ? errors.firstName?.message :
""

                          }
                        />
                      </div>
                      <div className='w-1/2'>
                        <TextInput
                          name='lastName'
                          label='Last Name'
                          placeholder='Wagonner'
                          type='text'
                          register={register("lastName", {
                            required: "Last Name is required",
                          })}
                          error={
                            errors.lastName ? errors.lastName?.message :
""

                          }
                        />
                      </div>
                    </div>

                    <div className='w-full flex gap-2'>
                      <div className='w-1/2'>
                        <TextInput
                          name='email'
                          label='email'
                          placeholder='email'
                          type='text'
                          register={register("email", {
                            required: "Coontact is required!",
                          })}
                          error={errors.contact ? errors.contact?.message
: ""}

                        />
                      </div>

                      {/* <div className='w-1/2'>
                        <TextInput
                          name='location'
                          label='Location'
                          placeholder='Location'
                          type='text'
                          register={register("location", {
                            required: "Location is required",
                          })}
                          error={
                            errors.location ? errors.location?.message :
""

                          }
                        />
                      </div> */}
                    </div>
                    {/* <TextInput
                      name='jobTitle'
                      label='Job Title'
                      placeholder='Software Engineer'
                      type='text'
                      register={register("jobTitle", {
                        required: "Job Title is required",
                      })}
                      error={errors.jobTitle ? errors.jobTitle?.message :
""}
                    /> */}

                    {/* <div className='w-full flex gap-2 text-sm'>
                      <div className='w-1/2'>
                        <label className='text-gray-600 text-sm mb-1'>
                          Profile Picture
                        </label>
                        <input
                          type='file'
                          onChange={(e) =>
setProfileImage(e.target.files[0])}
                        />
                      </div>

                      <div className='w-1/2'>
                        <label className='text-gray-600 text-sm mb-1'>
                          Resume
                        </label>
                        <input
                          type='file'
                          onChange={(e) => setUploadCv(e.target.files[0])}
                        />
                      </div>
                    </div> */}

                    <div className='flex flex-col'>
                      <label className='text-gray-600 text-sm mb-1'>
                        About
                      </label>
                      <textarea
                        className='ounded border border-gray-400
focus:outline-none focus:border-blue-500 focus:ring-1 focus:ring-blue-500
text-base px-4 py-2 resize-none'
                        rows={4}
                        cols={6}
                        {...register("bio", {
                          required:
                            "Write a little bit about yourself and your
projects",
                        })}
                        aria-invalid={errors.about ? "true" : "false"}
                      ></textarea>
                      {errors.about && (
                        <span
                          role='alert'
                          className='text-xs text-red-500 mt-0.5'
                        >
                          {errors.about?.message}
                        </span>
                      )}
                    </div>

                    <div className='mt-4'>
                      <CustomButton
                        type='submit'
                        containerStyles='inline-flex justify-center
rounded-md border border-transparent bg-blue-600 px-8 py-2 text-sm
font-medium text-white hover:bg-[#1d4fd846] hover:text-[#1d4fd8]
focus:outline-none '
                        title={"Submit"}
                      />
                    </div>
                  </form>
                </Dialog.Panel>
              </Transition.Child>
            </div>
          </div>
        </Dialog>
      </Transition>
```

```
    </>
  );
};

const UserProfile = () => {
  const { user } = useSelector((state) => state.user);
  const [open, setOpen] = useState(false);
  const userInfo = user;
  // console.log(userInfo)

  return (
    <div className='container mx-auto flex items-center justify-center
py-10'>
      <div className='w-full md:w-2/3 2xl:w-2/4 bg-white shadow-lg p-10
pb-20 rounded-lg'>
        <div className='flex flex-col items-center justify-center mb-4'>
          <h1 className='text-4xl font-semibold text-slate-600'>
            {userInfo?.firstName + " " + userInfo?.lastName}
          </h1>

          <h5 className='text-blue-700 text-base font-bold'>
            {userInfo?.jobTitle || "Add Job Title"}
          </h5>

          <div className='w-full flex flex-wrap lg:flex-row
justify-between mt-8 text-sm'>
            <p className='flex gap-1 items-center justify-center  px-3
py-1 text-slate-600 rounded-full'>
              <HiLocationMarker /> {userInfo?.location ?? "No Location"}
            </p>
            <p className='flex gap-1 items-center justify-center  px-3
py-1 text-slate-600 rounded-full'>
              <AiOutlineMail /> {userInfo?.email ?? "No Email"}
            </p>
            <p className='flex gap-1 items-center justify-center  px-3
py-1 text-slate-600 rounded-full'>
              <FiPhoneCall /> {userInfo?.contact ?? "No Contact"}
            </p>
          </div>
        </div>
        <hr />

        <div className='w-full py-10'>
          <div className='w-full flex flex-col-reverse md:flex-row gap-8
py-6'>
            <div className='w-full md:w-2/3 flex flex-col gap-4 text-lg
text-slate-600 mt-20 md:mt-0'>
              <p className='text-[#0536e7]  font-semibold
text-2xl'>ABOUT</p>
              <span className='text-base text-justify leading-7'>
                {userInfo?.bio ?? "No About Found"}
              </span>
            </div>

            <div className='w-full md:w-1/3 h-44'>
              <img
                src={userInfo?.profileUrl}
                alt={userInfo?.firstName}
                className='w-full h-48 object-contain rounded-lg'
              />
              <button
                className='w-full md:w-64 bg-blue-600 text-white mt-4 py-2
rounded'

                onClick={() => setOpen(true)}
              >
                Edit Profile
              </button>
            </div>
          </div>
        </div>
      </div>

      <UserForm open={open} setOpen={setOpen} />
    </div>
  );
};

export default UserProfile;
```

Components => CompanyCard (Frontend for the single company profile information)

```
import React from "react";
import { Link } from "react-router-dom";

const CompanyCard = ({ cmp }) => {
  return (
    <div className="w-full h-16 flex gap-4 items-center justify-between
bg-white shadow-md rounded">
      <div className="w-3/4 md:w-2/4 flex gap-4 items-center">
        <Link to={`/company-profile/${cmp?._id}`}>
          <img
            src={cmp?.profileUrl}
            alt={cmp?.name}
            className="w-8 md:w-12 h-8 md:h-12 rounded"
          />
        </Link>
        <div className="h-full flex flex-col">
          <Link
            to={`/company-profile/${cmp?._id}`}
            className="text-base md:text-lg font-semibold text-gray-600
truncate"
          >
            {cmp?.name}
          </Link>
          <span className="text-sm text-blue-600">{cmp?.email}</span>
        </div>
      </div>

      <div className="hidden w-1/4 h-full md:flex items-center">
        <p className="text-base text-start">{cmp?.location}</p>
      </div>

      <div className="w-1/4 h-full flex flex-col items-center">
        <p className="text-blue-600
font-semibold">{cmp?.jobPosts?.length}</p>
        <span className="text-xs md:base font-normal text-gray-600">
          Jobs Posted
        </span>
      </div>
    </div>
  );
};

export default CompanyCard;
```

Components => CustomButton (For reusable button)

```
import React from "react";

const CustomButton = ({ title, containerStyles, iconRight, type, onClick
}) => {
  return (
    <button
      onClick={onClick}
      type={type || "button"}
      className={`inline-flex items-center ${containerStyles}`}
    >
      {title}

      {iconRight && <div className='ml-2'>{iconRight}</div>}
    </button>
  );
};

export default CustomButton;
```

Components => Footer (Frontend for the footer + newsletter)

```
import { FaFacebookF, FaTwitter, FaLinkedinIn } from "react-icons/fa";
import { FiInstagram } from "react-icons/fi";
import { footerLinks } from "../utils/data";
import { Link } from "react-router-dom";
import TextInput from "./TextInput";
import CustomButton from "./CustomButton";

const Footer = () => {
  return (
    <footer className='text-white mp-20'>
      <div className='overflow-x-hidden -mb-0.5'>
```

```
        <svg
          preserveAspectRatio='none'
          viewBox='0 0 1200 120'
          xmlns='http://www.w3.org/2000/svg'
          style={{
            fill: "#1d4ed8",
            width: "125%",
            height: 112,
            transform: "rotate(180deg)",
          }}
        >
          <path d='M321.39 56.44c58-10.79 114.16-30.13 172-41.86
82.39-16.72 168.19-17.73 250.45-.39C823.78 31 906.67 72 985.66 92.83c70.05
18.48 146.53 26.09 214.34 3V0H0v27.35a600.21 600.21 0 00321.39 29.09z' />
        </svg>
      </div>

      <div className='bg-[#1d4ed8] '>
        <div className='container px-5 py-20 mx-auto '>
          <div className='w-full flex flex-wrap gap-10 justify-between
-mb-10 -px-4'>
            {footerLinks.map(({ id, title, links }) => (
              <div className='w-auto px-4 ' key={id + title}>
                <h2 className='font-medium text-white tracking-widest
text-sm mb-3'>
                  {title}
                </h2>

                <div className='mb-10 flex flex-col gap-3 '>
                  {links.map((link, index) => (
                    <Link
                      key={link + index}
                      to='/'
                      className='text-gray-300 text-sm hover:text-white '
                    >
                      {link}
                    </Link>
                  ))}
                </div>
              </div>
            ))}
          </div>
        </div>

        <div className=''>
          <p className='container px-5 mx-auto text-white mt-2 '>
            Subscribe to our Newsletter
          </p>

          <div className='container mx-auto px-5 pt-6 pb-8 flex flex-wrap
items-center justify-between '>
            <div className='w-full md:w-2/4 lg:w-1/3 h-16 flex
items-center justify-center md:justify-start '>
              <TextInput
                styles='w-full flex-grow md:w-40 2xl:w-64 bg-gray-100
sm:mr-4 md-2'
                type='email'
                placeholder='Email Address'
              />

              <CustomButton
                title='Subscribe'
                containerStyles={
                  "block bg-[#001a36] text-white px-5 py-2.5 text-md
rounded hover:bg-blue-800 focus:potline-none flex-col items-center mt-2"
                }
              />
            </div>

            <span className='inline-flex lg:ml-auto lg:mt-0 mt-6 w-full
justify-center md:justify-start md:w-auto'>
              <a className='text-white text-xl  hover:scale-125
ease-in-out duration-300'>
                <FaFacebookF />
              </a>
              <a className='ml-3 text-white text-xl  hover:scale-125
ease-in-out duration-300'>
                <FaTwitter />
              </a>
              <a className='ml-3 text-white text-xl  hover:scale-125 
ease-in-out duration-300'>
                <FiInstagram />
              </a>

              <a className='ml-3 text-white text-xl  hover:scale-125 
ease-in-out duration-300'>
                <FaLinkedinIn />
              </a>
            </span>
          </div>
        </div>

        <div className='bg-[#001a36]'>
          <div className='container mx-auto py-4 px-5 flex flex-wrap 
flex-col sm:flex-row'>
            <p className='text-gray-300 text-sm text-center sm:text-left'>
              &copy; 2023 Job Finder —
              <a
                href='https://youtube.com/@CodeWaveWithAsante'
                className='text-[#1199e7] ml-1'
                target='_blank'
                rel='noopener noreferrer'
              >
                @Developers of CSE471
              </a>
            </p>

            <span className='sm:ml-auto sm:mt-0 mt-2 sm:w-auto w-full 
sm:text-left text-center text-gray-300 text-sm'>
              &copy; Sillicondoor
            </span>
          </div>
        </div>
      </div>
    </footer>
  );
};

export default Footer;
```

Components => Header (Frontend for the home page → page view during sign in or create account)

```
import React from "react";
import { AiOutlineSearch, AiOutlineCloseCircle } from "react-icons/ai";
import { CiLocationOn } from "react-icons/ci";
import CustomButton from "./CustomButton";
import { popularSearch } from "../utils/data";
import { HeroImage } from "../assets";

const SearchInput = ({ placeholder, icon, value, setValue, styles }) => {
  const handleChange = (e) => {
    setValue(e.target.value);
  };

  const clearInput = () => setValue("");

  return (
    <div className={`flex w-full md:w-1/3 items-center ${styles}`}>
      {icon}

      <input
        value={value}
        onChange={(e) => handleChange(e)}
        type='text'
        className='w-full md:w-64 p-2 outline-none bg-transparent
text-base'
        placeholder={placeholder}
      />

      <AiOutlineCloseCircle
        className='hidden md:flex text-gray-600 text-xl cursor-pointer'
        onClick={clearInput}
      />
    </div>
  );
};

const Header = ({
  title,
  type,
  handleClick,
  searchQuery,
  setSearchQuery,
  location,
  setLocation,
}) => {
  return (
    <div className='bg-[#f7fdfd]'>
      <div
        className={`container mx-auto px-5 ${
          type ? "h-[500px]" : "h-[350px]"
        } flex items-center relative`}
      >
        <div className='w-full z-10'>
          <div className='mb-8'>
            <p className='text-slate-700 font-bold text-4xl'>{title}</p>
          </div>

          <div className='w-full flex items-center justify-around bg-white
px-2 md:px-5 py-2.5 md:py-6 shadow-2xl rounded-full'>
            <SearchInput
              placeholder='Job Title or Keywords'
              icon={<AiOutlineSearch className='text-gray-600 text-xl' />}
              value={searchQuery}
              setValue={setSearchQuery}
            />
            <SearchInput
              placeholder='Add Country or City'
              icon={<CiLocationOn className='text-gray-600 text-xl' />}
              value={location}
              setValue={setLocation}
              styles={"hidden md:flex"}
            />

            <div>
              <CustomButton
                onClick={handleClick}
                title='Search'
                containerStyles={
                  "text-white py-2 md:py3 px-3 md:px-10 focus:outline-none
bg-blue-600 rounded-full md:rounded-md text-sm md:text-base"
                }
              />
            </div>
          </div>

          {type && (
            <div className='w-full lg:1/2 flex flex-wrap gap-3 md:gap-6
py-10 md:py-14'>
              {popularSearch.map((search, index) => (
                <span
                  key={index}
                  className='bg-[#1d4fd826] text-[#1d4ed8] py-1 px-2
rounded-full text-sm md:text-base'
                >
                  {search}
                </span>
              ))}
            </div>
          )}
        </div>

        <div className='w-1/3 h-full absolute top-24 md:-top-6 lg:-top-14
right-16 2xl:right-[18rem]'>
          <img src={HeroImage} className='object-contain' />
        </div>
      </div>
    </div>
  );
};

export default Header;
```

Components => index (Importing and exporting all the components)

```
import Navbar from "./Navbar";
import Footer from "./Footer";
import CustomButton from "./CustomButton";
import TextInput from "./TextInput";
import SignUp from "./SignUp";
import Header from "./Header";
import ListBox from "./ListBox";
import JobCard from "./JobCard";
import Loading from "./Loading";
import CompanyCard from "./CompanyCard";
import JobTypes from "./JobTypes";

export {
  Navbar,
  Footer,
  CustomButton,
  TextInput,
  SignUp,
  Header,
  ListBox,
  JobCard,
  Loading,
  CompanyCard,
  JobTypes,
};
```

Components => JobCard (Frontend for the job detail header)

```
mport { GoLocation } from "react-icons/go";
import moment from "moment";
import { Link } from "react-router-dom";

const JobCard = ({ job }) => {
  return (
    <Link to={`/job-detail/${job?.id}`}>
      <div
        className='w-full md:w-[16rem] 2xl:w-[18rem] h-[16rem]
md:h-[18rem] bg-white flex flex-col justify-between shadow-lg
                rounded-md px-3 py-5 '
      >
        <div className='flex gap-3'>
          <img
            src={job?.company?.profileUrl}
            alt={job?.company?.name}
            className='w-14 h-14'
          />

          <div className=''>
            <p className='text-lg font-semibold
truncate'>{job?.jobTitle}</p>
            <span className='flex gap-2 items-center'>
              <GoLocation className='text-slate-900 text-sm' />
              {job?.location}
            </span>
          </div>
        </div>

        <div className='py-3'>
          <p className='text-sm'>
            {job?.detail[0]?.desc?.slice(0, 150) + "..."}
          </p>
        </div>

        <div className='flex items-center justify-between'>
          <p className='bg-[#1d4fd826] text-[#1d4fd8] py-0.5 px-1.5
rounded font-semibold text-sm'>
            {job?.jobType}
          </p>
          <span className='text-gray-500 text-sm'>
            {moment(job?.createdAt).fromNow()}
          </span>
        </div>
      </div>
    </Link>
  );
};

export default JobCard;
```

Components => JobTypes (For the drop down list for selecting job types in job post form)

```
import { Fragment, useState } from "react";
import { Listbox, Transition } from "@headlessui/react";
import { BsCheck2, BsChevronExpand } from "react-icons/bs";

const types = ["Full-Time", "Part-Time", "Contract", "Intern"];

export default function JobTypes({ jobTitle, setJobTitle }) {
  return (
    <div className='w-full '>
      <Listbox value={jobTitle} onChange={setJobTitle}>
        <div className='relative'>
          <Listbox.Button className='relative w-full cursor-default
rounded bg-white py-2.5 pl-3 pr-10 text-left focus:outline-none border
border-gray-400 focus:border-blue-500 focus:ring-1 focus:ring-blue-500
sm:text-sm'>
            <span className='block truncate'>{jobTitle}</span>
            <span className='pointer-events-none absolute inset-y-0
right-0 flex items-center pr-2'>
              <BsChevronExpand
                className='h-5 w-5 text-gray-500'
                aria-hidden='true'
              />
            </span>
          </Listbox.Button>
          <Transition
            as={Fragment}
            leave='transition ease-in duration-100'
            leaveFrom='opacity-100'
            leaveTo='opacity-0'
          >
            <Listbox.Options className='absolute mt-1 max-h-60 w-full
overflow-auto rounded-md bg-white py-1 text-base shadow-lg ring-1
ring-black ring-opacity-5 focus:outline-none sm:text-sm'>
              {types.map((type, index) => (
                <Listbox.Option
                  key={index}
                  className={({ active }) =>
                    `relative cursor-default select-none py-2 pl-10 pr-4
${
                      active ? "bg-amber-100 text-amber-900" :
"text-gray-900"
                    }`
                  }
                  value={type}
                >
                  {({ selected }) => (
                    <>
                      <span
                        className={`block truncate ${
                          selected ? "font-medium" : "font-normal"
                        }`}
                      >
                        {type}
                      </span>
                      {selected ? (
                        <span className='absolute inset-y-0 left-0 flex
items-center pl-3 text-amber-600'>
                          <BsCheck2 className='h-5 w-5' aria-hidden='true'
/>

                        </span>
                      ) : null}
                    </>
                  )}
                </Listbox.Option>
              ))}
            </Listbox.Options>
          </Transition>
        </div>
      </Listbox>
    </div>
  );
}
```

Components => ListBox (For reusable drop down list and by default it is for for sorting in terms of newest, oldest, A-Z, Z-A)

```
import { Fragment, useState } from "react";
import { Listbox, Transition } from "@headlessui/react";
import { BsCheck2, BsChevronExpand } from "react-icons/bs";

const options = ["Newest", "Oldest", "A-Z", "Z-A"];
const ListBox = ({ sort, setSort }) => {
  return (
    <div className='w-[8rem] md:w-[10rem]'>
      <Listbox value={sort} onChange={setSort}>
        <div className='relative mt-1'>
          <Listbox.Button
            className={
              "relative w-full cursor-default rounded-lg bg-white py-2
pl-3 pr-10 text-left shadow-md focus:outline-none
focus-visible:border-indigo-500 focus-visible:ring-2
focus-visible:ring-white focus-visible:ring-opacity-75
focus-visible:ring-offset-2 focus-visible:ring-offset-orange-300
sm:text-sm"
            }
          >
            <span className='block truncate'>{sort}</span>

            <span className='pointer-events-none absolute inset-y-0
right-0 flex items-center pr-2'>
              <BsChevronExpand
                className='h-5 w-5 text-gray-500'
                aria-hidden='true'
              />
            </span>
          </Listbox.Button>

          <Transition
            as={Fragment}
            leave='transition ease-in duration-100'
            leaveFrom='opacity-100'
            leaveTo='opacity-0'
          >
            <Listbox.Options className='absolute mt-1 max-h-60 w-full
overflow-auto rounded-md bg-white py-1 text-base shadow-lg ring-1
ring-black ring-opacity-5 focus:outline-none sm:text-sm'>
              {options.map((op, index) => (
                <Listbox.Option
                  key={index}
                  className={({ active }) =>
```

```
                    `relative cursor-default select-none py-2 pl-10 pr-4
${
                      active ? "bg-[#1d4fd830] text-[#1d4ed8]" :
"text-gray-900"
                    }`
                  }
                  value={op}
                >
                  {({ selected }) => (
                    <>
                      <span
                        className={`block truncate ${
                          selected ? "font-medium" : "font-normal"
                        }`}
                      >
                        {op}
                      </span>
                      {selected ? (
                        <span className='absolute inset-y-0 left-0 flex
items-center pl-3 text-[#1d4ed8]'>
                          <BsCheck2 className='h-5 w-5' aria-hidden='true'
/>
                        </span>
                      ) : null}
                    </>
                  )}
                </Listbox.Option>
              ))}
            </Listbox.Options>
          </Transition>
        </div>
      </Listbox>
    </div>
  );
};

export default ListBox;
```

Components => Loading (For loading process)

```
const Loading = () => {
  return (
    <div className='dots-container'>
      <div className='dot'></div>
      <div className='dot'></div>
      <div className='dot'></div>
      <div className='dot'></div>
      <div className='dot'></div>
    </div>
  );
};

export default Loading;
```

Components => Navbar (For the navigation bar)

```
import React, {Fragment, useState} from 'react';
import { Menu, Transition } from "@headlessui/react";
import { BiChevronDown } from "react-icons/bi";
import { CgProfile } from "react-icons/cg";
import { HiMenuAlt3 } from "react-icons/hi";
import { AiOutlineClose, AiOutlineLogout } from "react-icons/ai";
import { Link } from "react-router-dom";
import CustomButton from "./CustomButton";
import { users } from "../utils/data";
import { useSelector } from "react-redux";
import { useDispatch } from "react-redux";


const MenuList = ({ user, onClick }) => {

  const dispatch = useDispatch();
  const handleLogout = () => {

  };

  return (
    <div>
      <Menu as='div' className='inline-block text-left'>
        <div className='flex'>
```

```
          <Menu.Button className='inline-flex gap-2 w-full rounded-md
bg-white md:px-4 py-2 text-sm font-medium text-slate-700
hover:bg-opacity-20 '>
            <div className='leading[80px] flex flex-col items-start'>
              <p className='text-sm font-semibold '>
                {user?.firstName ?? user?.name}
              </p>
              <span className='text-sm text-blue-600 '>
                {user?.jobTitle ?? user?.email}
              </span>
            </div>

            <img
              src={user?.profileUrl}
              alt='user profile'
              className='w-10 h-10 rounded-full object-cover '
            />
            <BiChevronDown
              className='h-8 w-8 text-slate-600'
              aria-hidden='true'
            />
          </Menu.Button>
        </div>

        <Transition
          as={Fragment}
          enter='transition ease-out duration-100'
          enterFrom='transform opacity-0 scale-95'
          enterTo='transform opacity-100 scale-100'
          leave='transition ease-in duration-75'
          leaveFrom='transform opacity-100 scale-100'
          leaveTo='transform opacity-0 scale-95'
        >
          <Menu.Items className='absolute z-50 right-2 mt-2 w-56
origin-top-right divide-y dividfe-gray-100 rounded-md bg-white shadow-lg
focus:outline-none '>
            <div className='p-1 '>
              <Menu.Item>
                {({ active }) => (
                  <Link
```

```
                    to={`${
                      user?.accountType ? "user-profile" :
"company-profile"
                    }`}
                    className={`${
                      active ? "bg-blue-500 text-white" : "text-gray-900"
                    } group flex w-full items-center rounded-md p-2
text-sm`}
                    onClick={onClick}
                  >
                    <CgProfile
                      className={`${
                        active ? "text-white" : "text-gray-600"
                      } mr-2 h-5 w-5  `}
                      aria-hidden='true'
                    />
                    {user?.accountType ? "User Profile" : "Company
Profile"}
                  </Link>
                )}
              </Menu.Item>

              <Menu.Item>
                {({ active }) => (
                  <button
                    onClick={() => handleLogout()}
                    className={`${
                      active ? "bg-blue-500 text-white" : "text-gray-900"
                    } group flex w-full items-center rounded-md px-2 py-2
text-sm`}
                  >
                    <AiOutlineLogout
                      className={`${
                        active ? "text-white" : "text-gray-600"
                      } mr-2 h-5 w-5  `}
                      aria-hidden='true'
                    />
                    Log Out
                  </button>
                )}
              </Menu.Item>
            </div>
          </Menu.Items>
        </Transition>
      </Menu>
    </div>
  );
}



const Navbar = () => {
  //get user if user is logged in
  const user = useSelector((state) => state.user);
  const [isOpen, setIsOpen] = useState(false);

  const handleCloseNavbar = () => {
    setIsOpen((prev) => !prev);
  }


  return (
    <>
      <div className='relative bg-[#f7fdfd] z-50'>
        <nav className='container mx-auto flex items-center
justify-between p-5'>
          {/* Logo part */}
          <div>
            <Link to='/' className='text-blue-600 font-bold text-xl'>
              Sillicon<span className='text-[#1677cccb]'>door</span>
            </Link>
          </div>

          {/* all navigation list */}
          <ul className='hidden lg:flex gap-10 text-base'>
            <li>
              <Link to='/'>Explore Jobs</Link>
            </li>
            <li>
              <Link to='/companies'>Companies</Link>
            </li>
            {/* <li>
              <Link to='/upload-job'>Upload Job</Link>
            </li> */}
            <li>
              <Link to='/about-us'>About</Link>
            </li>
          </ul>

          {/* user authentication: check if user logged in */}
          <div className='hidden lg:block'>
            {!user?.token ? (
              <Link to='/user-auth'>
                <CustomButton
                  title='Sign In'
                  containerStyles='text-blue-600 py-1.5 px-5
focus:outline-none hover:bg-blue-700 hover:text-white rounded-full
text-base border border-blue-600'
                />
              </Link>
            ) : (
              <div>
                <MenuList user={user} />
              </div>
            )}
          </div>

          <button
            className='block lg:hidden text-slate-900'
            onClick={() => setIsOpen((prev) => !prev)}
          >
            {isOpen ? <AiOutlineClose size={26} /> : <HiMenuAlt3 size={26}
/>}
          </button>
        </nav>

        {/* Mobile Menu : different view based on user.accountType*/}
        <div
          className={`${
            isOpen ? "absolute flex bg-[#f7fdfd] " : "hidden"
          } container mx-auto lg:hidden flex-col pl-8 gap-3 py-5`}
        >
          <Link to='/' onClick={handleCloseNavbar}>
            Explore Jobs
          </Link>
          <Link to='/companies' onClick={handleCloseNavbar}>
            Companies
          </Link>
          <Link
            onClick={handleCloseNavbar}
            to={"/upload-job"}
              // user?.accountType === "seeker" ? "job-applied" :
"upload-job"
          >
            {/* {user?.accountType === "seeker" ? "Job Applied" : "Upload
Job"} */}Upload
          </Link>
          <Link to='/about-us' onClick={handleCloseNavbar}>
            About
          </Link>

          {/* based on if user logged in ? */}
          <div className='w-full py-10'>
            {!user?.token ? (
              <a href='/user-auth'>
                <CustomButton
                  title='Sign In'
                  containerStyles={`text-blue-600 py-1.5 px-5
focus:outline-none hover:bg-blue-700 hover:text-white rounded-full
text-base border border-blue-600`}
                />
              </a>
            ) : (
              <div>
                <MenuList user={user} onClick={handleCloseNavbar} />
              </div>
            )}
          </div>
        </div>
      </div>
```

```
    </>
  );
};

export default Navbar;
```

Components => SignUp (For the sign up and create account form)

```
import React, { Fragment, useState } from "react";
import { Dialog, Transition } from "@headlessui/react";
import { useForm } from "react-hook-form";
import { useLocation } from "react-router-dom";
import { useDispatch } from "react-redux";
import TextInput from "./TextInput";
import CustomButton from "./CustomButton";
import { endpoints } from "../utils/api_endpoints";
import { _userLogin } from "../utils/axios_controllers";
import {login} from '../redux/userSlice.js'


const SignUp = ({ open, setOpen }) => {
  const dispatch = useDispatch();
  const location = useLocation();

  const [isRegister, setIsRegister] = useState(true);
  const [accountType, setAccountType] = useState("seeker");

  const [errMsg, setErrMsg] = useState("");
  const {
    register,
    handleSubmit,
    getValues,
    watch,
    formState: { errors },
  } = useForm({
    mode: "onChange",
  });
  const from = location.state?.from?.pathname || "/";

  const closeModal = () => setOpen(false);
```

```
//handle login, signup for both user and company
const onSubmit = async (data) => {
  console.log(data)
  const {email, password} = data;

  let api = '';
  if (!isRegister) {
    //handle logins
    if (accountType === "seeker") {
      api = endpoints.userLogin;
    } else {
      api = endpoints.companyLogin;
    }
  } else {
    //handle signups
    if (accountType === "seeker") {
      api = endpoints.userSignup;
    } else {
      api = endpoints.companySignup;
    }
  }

  try {

    const response = await _userLogin({ email, password });
    console.log(response)
    if (response) {
      // Authentication successful, dispatch action to set user
      dispatch(login(response));

      // Close the modal after successful authentication
      closeModal();
    } else {
      // Authentication failed, update the error message state
      setErrMsg(response || 'Authentication failed');
    }
  } catch (error) {
    console.error('Error during form submission:', error);
    // setErrMsg('An unexpected error occurred');
  }
  };

  return (
    <>
      <Transition appear show={open || false}>
        <Dialog as='div' className='relative z-10 ' onClose={closeModal}>
          <Transition.Child
            as={Fragment}
            enter='ease-out duration-300'
            enterFrom='opacity-0'
            enterTo='opacity-100'
            leave='ease-in duration-200'
            leaveFrom='opacity-100'
            leaveTo='opacity-0'
          >
            <div className='fixed inset-0 bg-black bg-opacity-25' />
          </Transition.Child>

          <div className='fixed inset-0 overflow-y-auto '>
            <div className='flex min-h-full items-center justify-center
p-4 text-center '>
              <Transition.Child
                as={Fragment}
                enter='ease-out duration-300'
                enterFrom='opacity-0 scale-95'
                enterTo='opacity-100 scale-100'
                leave='ease-in duration-200'
                leaveFrom='opacity-100 scale-100'
                leaveTo='opacity-0 scale-95'
              >
                <Dialog.Panel className='w-full max-w-md transform
overflow-hidden rounded-2xl bg-white p-6 text-left align-middle shadow-xl
transition-all '>
                  <Dialog.Title
                    as='h3'
                    className='text-xl font-semibold lwading-6
text-gray-900'
                  >
                    {isRegister ? "Create Account" : "Account Sign In"}
                  </Dialog.Title>
                  <div className='w-full flex items-center justify-center
py-4 '>
                    <button
                      className={`flex-1 px-4 py-2 rounded text-sm
outline-none ${
                        accountType === "seeker"
                          ? "bg-[#1d4fd862] text-blue-900 font-semibold"
                          : "bg-white border border-blue-400"
                      }`}
                      onClick={() => setAccountType("seeker")}
                    >
                      User Account
                    </button>
                    <button
                      className={`flex-1 px-4 py-2 rounded text-sm
outline-none ${
                        accountType !== "seeker"
                          ? "bg-[#1d4fd862] text-blue-900 font-semibold"
                          : "bg-white border border-blue-400"
                      }`}
                      onClick={() => setAccountType("company")}
                    >
                      Company Account
                    </button>
                  </div>

                  <form
                    className='w-full flex flex-col gap-5'
                    onSubmit={handleSubmit(onSubmit)}
                  >
                    <TextInput
                      name='email'
                      label='Email Address'
                      placeholder='email@example.com'
                      type='email'
                      register={register("email", {
                        required: "Email Address is required!",
                      })}
                      error={errors.email ? errors.email.message : ""}
                    />

                    {isRegister && (
                      <div className='w-full flex gap-1 md:gap-2'>
                        <div
                          className={`${
                            accountType === "seeker" ? "w-1/2" : "w-full"
                          }`}
                        >
                          <TextInput
                            name={
                              accountType === "seeker" ? "firstName" :
"name"

                            }
                            label={
                              accountType === "seeker"
                                ? "First Name"
                                : "Company Name"
                            }
                            placeholder={
                              accountType === "seeker"
                                ? "eg. James"
                                : "Comapy name"
                            }
                            type='text'
                            register={register(
                              accountType === "seeker" ? "firstName" :
"name",

                              {
                                required:
                                  accountType === "seeker"
                                    ? "First Name is required"
                                    : "Company Name is required",
                              }
                            )}
                            error={
                              accountType === "seeker"
                                ? errors.firstName
                                  ? errors.firstName?.message
                                  : ""
                                : errors.name
                                ? errors.name?.message
                                : ""
                            }
                          />
                        </div>

                        {accountType === "seeker" && isRegister && (
                          <div className='w-1/2'>
                            <TextInput
                              name='lastName'
                              label='Last Name'
                              placeholder='Wagonner'
                              type='text'
                              register={register("lastName", {
                                required: "Last Name is required",
                              })}
                              error={
                                errors.lastName ? errors.lastName?.message
: ""

                              }
                            />
                          </div>
                        )}
                      </div>
                    )}

                    <div className='w-full flex gap-1 md:gap-2'>
                      <div className={`${isRegister ? "w-1/2" :
"w-full"}`}>
                        <TextInput
                          name='password'
                          label='Password'
                          placeholder='Password'
                          type='password'
                          register={register("password", {
                            required: "Password is required!",
                          })}
                          error={
                            errors.password ? errors.password?.message :
""
                          }
                        />
                      </div>

                      {isRegister && (
                        <div className='w-1/2'>
                          <TextInput
                            label='Confirm Password'
                            placeholder='Password'
                            type='password'
                            register={register("cPassword", {
                              validate: (value) => {
                                const { password } = getValues();

                                if (password != value) {
                                  return "Passwords do no match";
                                }
                              },
                            })}
                            error={
                              errors.cPassword &&
                              errors.cPassword.type === "validate"
                                ? errors.cPassword?.message
                                : ""
                            }
                          />
                        </div>
                      )}
                    </div>

                    {errMsg && (
                      <span
                        role='alert'
                        className='text-sm text-red-500 mt-0.5'
                      >
                        {errMsg}
                      </span>
                    )}
                    <div className='mt-2'>
                      <CustomButton
                        type='submit'
                        containerStyles={`inline-flex justify-center
rounded-md bg-blue-600 px-8 py-2 text-sm font-medium text-white
outline-none hover:bg-blue-800`}
                        title={isRegister ? "Create Account" : "Login
Account"}
                      />
                    </div>
                  </form>

                  <div className='mt-4'>
                    <p className='text-sm text-gray-700'>
                      {isRegister
                        ? "Already has an account?"
                        : "Do not have an account"}

                      <span
                        className='text-sm text-blue-600 ml-2
hover:text-blue-700 hover:font-semibold cursor-pointer'
                        onClick={() => setIsRegister((prev) => !prev)}
                      >
                        {isRegister ? "Login" : "Create Account"}
                      </span>
                    </p>
                  </div>
                </Dialog.Panel>
              </Transition.Child>
            </div>
          </div>
        </Dialog>
      </Transition>
    </>
  );
};

export default SignUp;
```

Components => Testcomponent

```
import React from 'react';

const Testcomponent = () => {
  return (
    <div>
      <h2>Test test test</h2>
    </div>
  );
};

export default Testcomponent;
```

Components =>TextInput (For the text input in all the forms)

```
import React from "react";

const TextInput = React.forwardRef(
  ({ type, placeholder, styles, label, register, name, error }, ref) => {
    return (
      <div className='flex flex-col mt-2'>
        <p className='text-gray-600 text-sm mb-1 '>{label}</p>

        <input
          type={type}
          name={name}
          placeholder={placeholder}
          ref={ref}
          className={`rounded border border-gray-400 focus:outline-none
focus:border-blue-500 focus:ring-1 focus:ring-blue-500 text-base px-4 py-2
${styles}`}
          {...register}
          aria-invalid={error ? "true" : "false"}
        />
        {error && <span className='text-xs text-red-500 mt-0.5
'>{error}</span>}
      </div>
    );
  }
);
```

```
export default TextInput;
```

Redux => rootReducers (For combining all the reducers used for checking the type of actions)

```
import { combineReducers } from "@reduxjs/toolkit";
import userReducer from "./userSlice";

const rootReducer = combineReducers({
  user: userReducer,
});

export { rootReducer };
```

Redux => store (For storing all the store used to store all the change in state before sending them in the view stage)

```
import { configureStore } from "@reduxjs/toolkit";

import {
  useDispatch as useAppDispatch,
  useSelector as useAppSelector,
} from "react-redux";
import { rootReducer } from "./rootReducer";

const store = configureStore({
  reducer: rootReducer,
});

const { dispatch } = store;
const useSelector = useAppSelector;
const useDispatch = () => useAppDispatch();

export { store, dispatch, useDispatch, useSelector };
```

Redux => userSlice (For creating action and reducers together)

```
import { createSlice } from "@reduxjs/toolkit";
// import { dispatch } from "./store";
import { users } from "../utils/data";
```

```
const initialState = {
  user: JSON.parse(window?.localStorage.getItem("userInfo")) ?? users[1],
};

export const userSlice = createSlice({
  name: "userInfo",
  initialState,
  reducers: {
    login(state, action) {
      state.user = action.payload.user;
    },
    logout(state) {
      state.user = null;
      localStorage?.removeItem("userInfo");
    },
  },
});


console.log(userSlice.actions) // to check stuffs
export const { login, logout } = userSlice.actions;
export default userSlice.reducer;
```

Utils => api_endpoints (All the API endpoints from the backend are listed)

```
//all api endpoints here
export const base_url = "http://localhost:4000/api/v1"


export const endpoints = {
  userLogin: "/auth/login", //post
  userSignup: "/auth/signup", //post
  getUser: "/users/get-user", //post
  updateUser: "/users/update-user", //put
  showAppliedJobs: "/users/get-applied-jobs", //post


  companyLogin : "/companies/login", //post
  companySignUp: "/companies/signup", //post
  getSingleCompany : "/companies/:id", //get
  getJobList : "/companies/get-company-joblistig", //post
  getCompanyProfileInfo: "/companies/get-company", //get [Where did this
come from?]
  getAllCompanies : "/companies/", //get
  updateCompanyProfile: "/companies/update-company", //Put


  uploadJob : "/jobs/upload-job", //Post
  updateJob : "/jobs/update-job", //Put
  findJobs : "/jobs/find-jobs", //Get
  deleteJobs : "/jobs/delete-job/:id", //Delete
  applyJobs : "/jobs/jobId/apply", //post
  showApplicants : "/jobs/:jobId/applicants", //Get

};
```

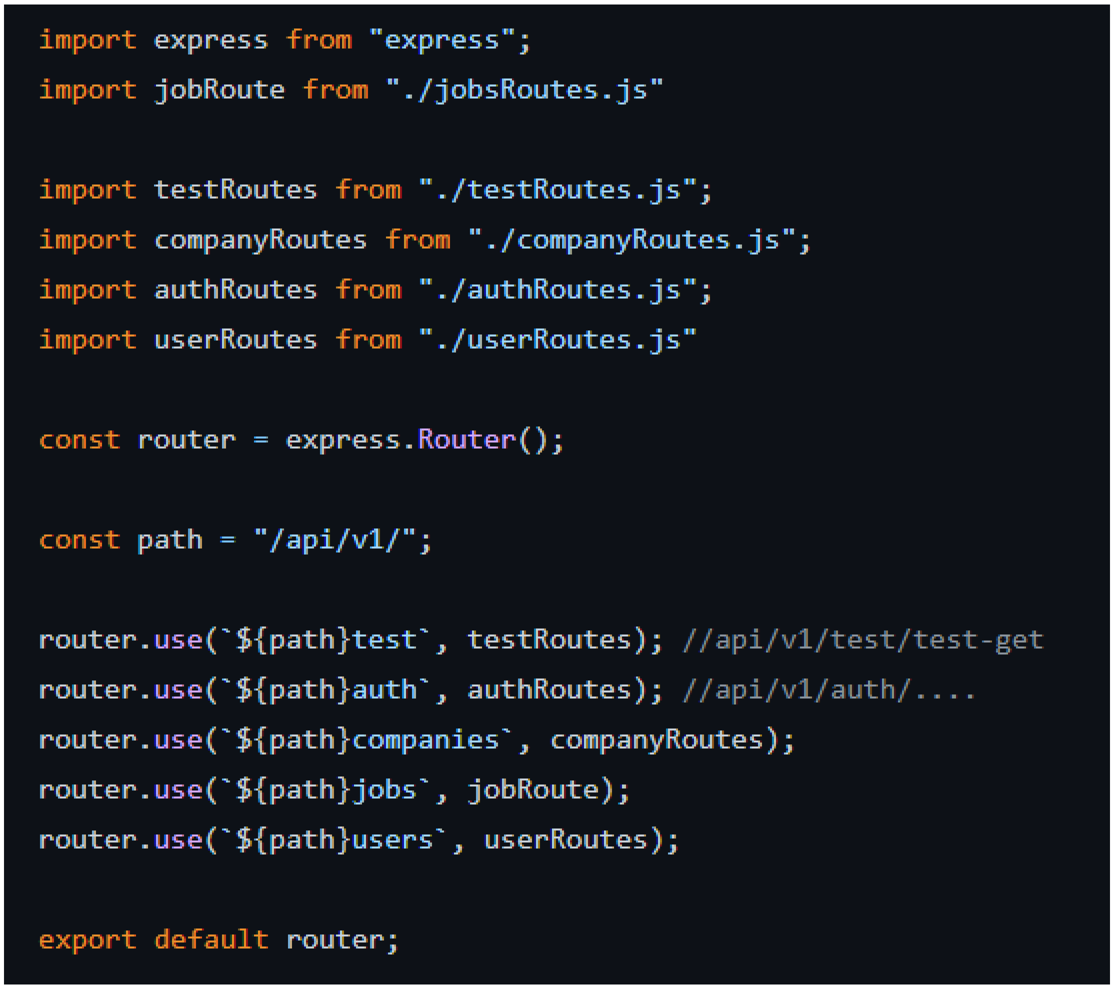
Utils => axios_controllers (connection between frontend and backend are created here)

```
import axios from "axios";

//file or methods import
import { base_url, endpoints } from "./api_endpoints.js";



//user auth//

//signup//
export const _userSignup = async function (user) {
  const api_endpoint = `${base_url}${endpoints.userSignup}`;
  try {
    const response = await axios.post(api_endpoint, user);
      return response.data; // return a json with user
  } catch (err) {
    // console.log(err.response.data.message);
    console.log(err.message)
  }
};
// test signup
// const newUser ={username: "uniquename", firstName: "Faysel", lastName:
"Rajo", email:"unique@gmail.com", password: "122345"}
// _userSignup(newUser).then(data => console.log(data))
```

```
//signup//
export const _userLogin = async function (credential) {
  const api_endpoint = `${base_url}${endpoints.userLogin}`;
  try {
    const response = await axios.post(api_endpoint, credential);
      return response.data;
  } catch (err) {
    // console.log(err.response.data.message);
    console.log(err.message)
  }
};
// const credential ={email:"anika@gmail.com", password: "anika1234"}
// _userLogin(credential).then(data => console.log(data))
```

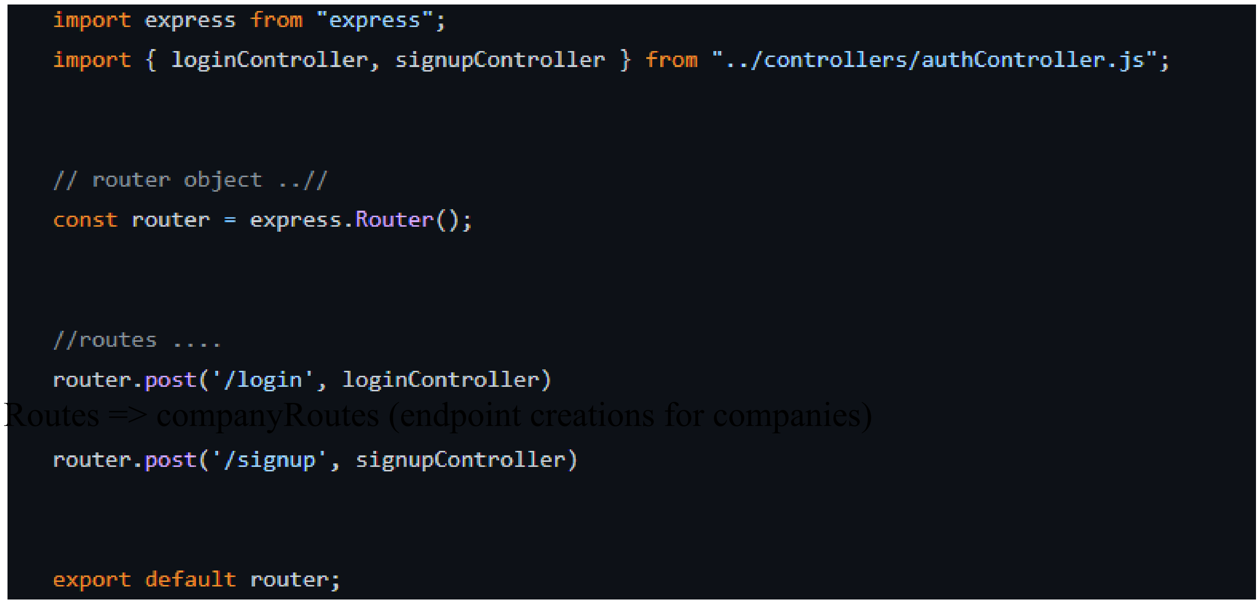
<u>**Backend Development**</u>

Routes => index (all routes are imported here)



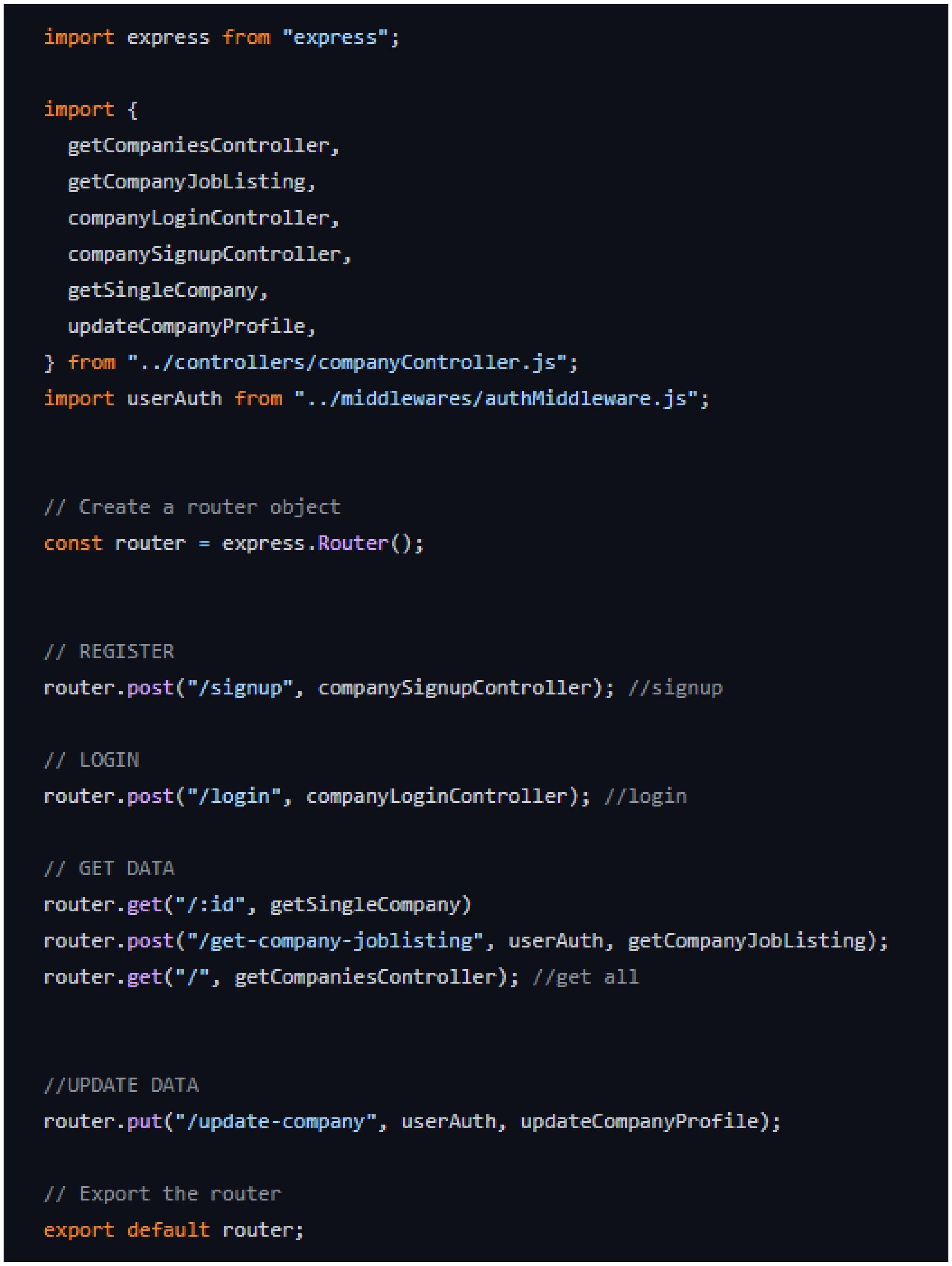
```javascript
import express from "express";
import jobRoute from "./jobsRoutes.js"

import testRoutes from "./testRoutes.js";
import companyRoutes from "./companyRoutes.js";
import authRoutes from "./authRoutes.js";
import userRoutes from "./userRoutes.js"

const router = express.Router();

const path = "/api/v1/";

router.use(`${path}test`, testRoutes); //api/v1/test/test-get
router.use(`${path}auth`, authRoutes); //api/v1/auth/....
router.use(`${path}companies`, companyRoutes);
router.use(`${path}jobs`, jobRoute);
router.use(`${path}users`, userRoutes);

export default router;
```

Routes => authRoutes (endpoint creations for login and signup)



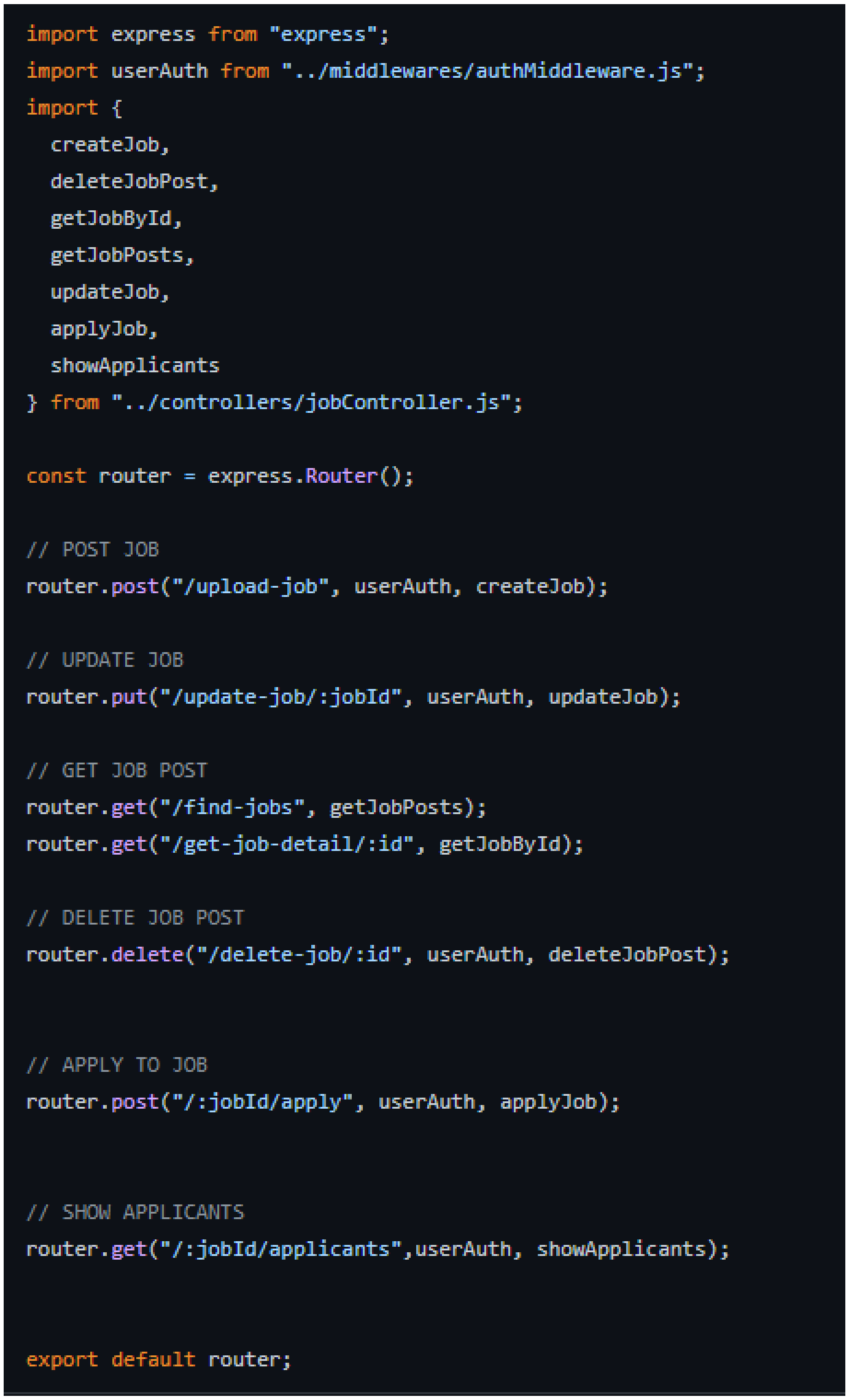
```javascript
import express from "express";
import { loginController, signupController } from "../controllers/authController.js";


// router object ..//
const router = express.Router();


//routes ....
router.post('/login', loginController)

router.post('/signup', signupController)


export default router;
```

Routes => companyRoutes (endpoint creations for companies)



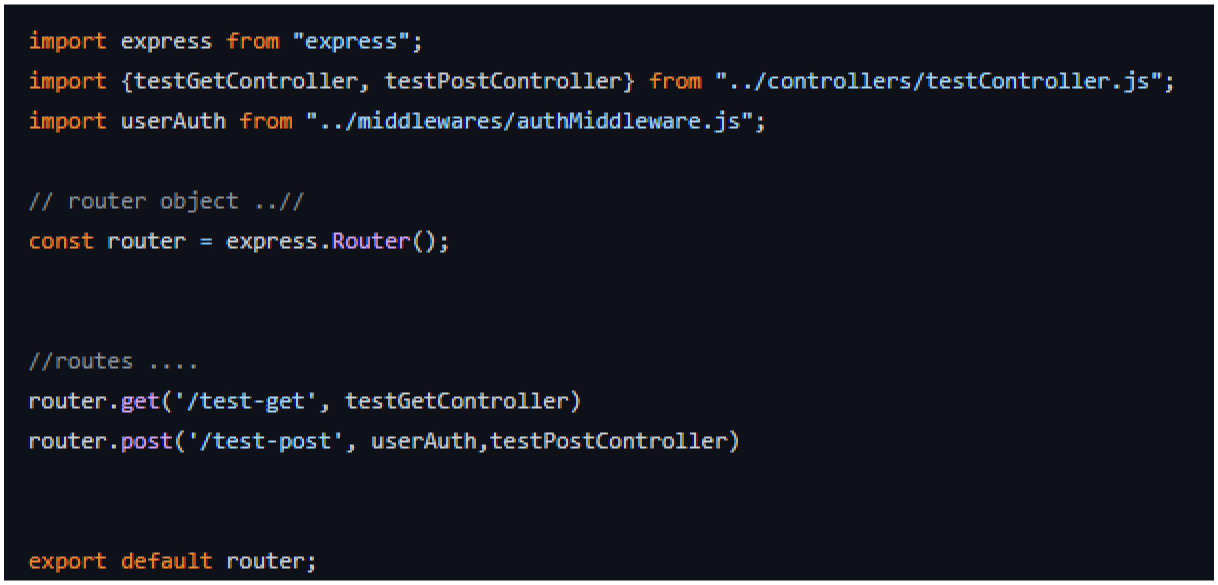
```javascript
import express from "express";

import {
  getCompaniesController,
  getCompanyJobListing,
  companyLoginController,
  companySignupController,
  getSingleCompany,
  updateCompanyProfile,
} from "../controllers/companyController.js";
import userAuth from "../middlewares/authMiddleware.js";


// Create a router object
const router = express.Router();


// REGISTER
router.post("/signup", companySignupController); //signup

// LOGIN
router.post("/login", companyLoginController); //login

// GET DATA
router.get("/:id", getSingleCompany)
router.post("/get-company-joblisting", userAuth, getCompanyJobListing);
router.get("/", getCompaniesController); //get all


//UPDATE DATA
router.put("/update-company", userAuth, updateCompanyProfile);

// Export the router
export default router;
```

Routes => Jobs (endpoint creations for job)



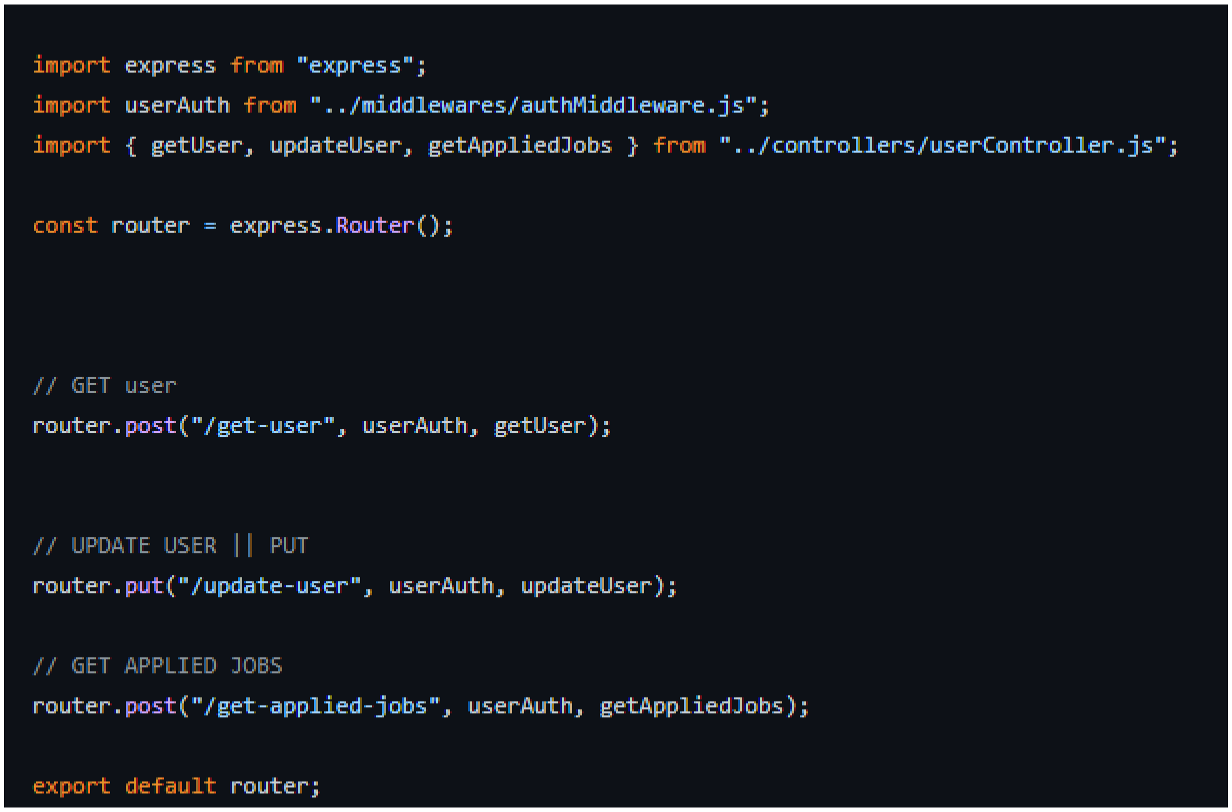
```
import express from "express";
import userAuth from "../middlewares/authMiddleware.js";
import {
  createJob,
  deleteJobPost,
  getJobById,
  getJobPosts,
  updateJob,
  applyJob,
  showApplicants
} from "../controllers/jobController.js";

const router = express.Router();

// POST JOB
router.post("/upload-job", userAuth, createJob);

// UPDATE JOB
router.put("/update-job/:jobId", userAuth, updateJob);

// GET JOB POST
router.get("/find-jobs", getJobPosts);
router.get("/get-job-detail/:id", getJobById);

// DELETE JOB POST
router.delete("/delete-job/:id", userAuth, deleteJobPost);


// APPLY TO JOB
router.post("/:jobId/apply", userAuth, applyJob);


// SHOW APPLICANTS
router.get("/:jobId/applicants",userAuth, showApplicants);


export default router;
```

Routes => testRoutes (endpoint creations for test)



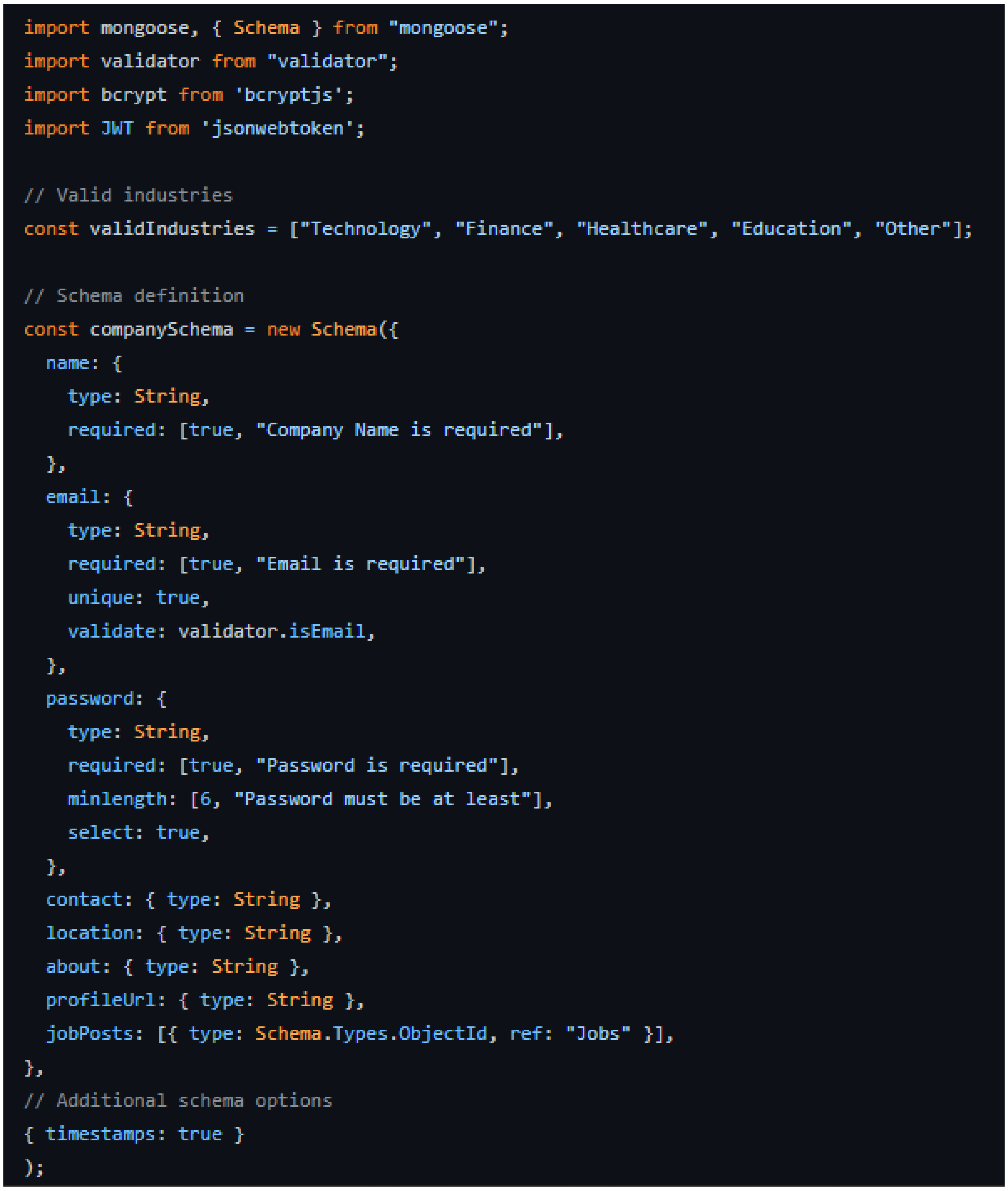
```
import express from "express";
import {testGetController, testPostController} from "../controllers/testController.js";
import userAuth from "../middlewares/authMiddleware.js";

// router object ..//
const router = express.Router();


//routes ....
router.get('/test-get', testGetController)
router.post('/test-post', userAuth,testPostController)


export default router;
```

Routes => userRoutes (endpoint creations for user info)

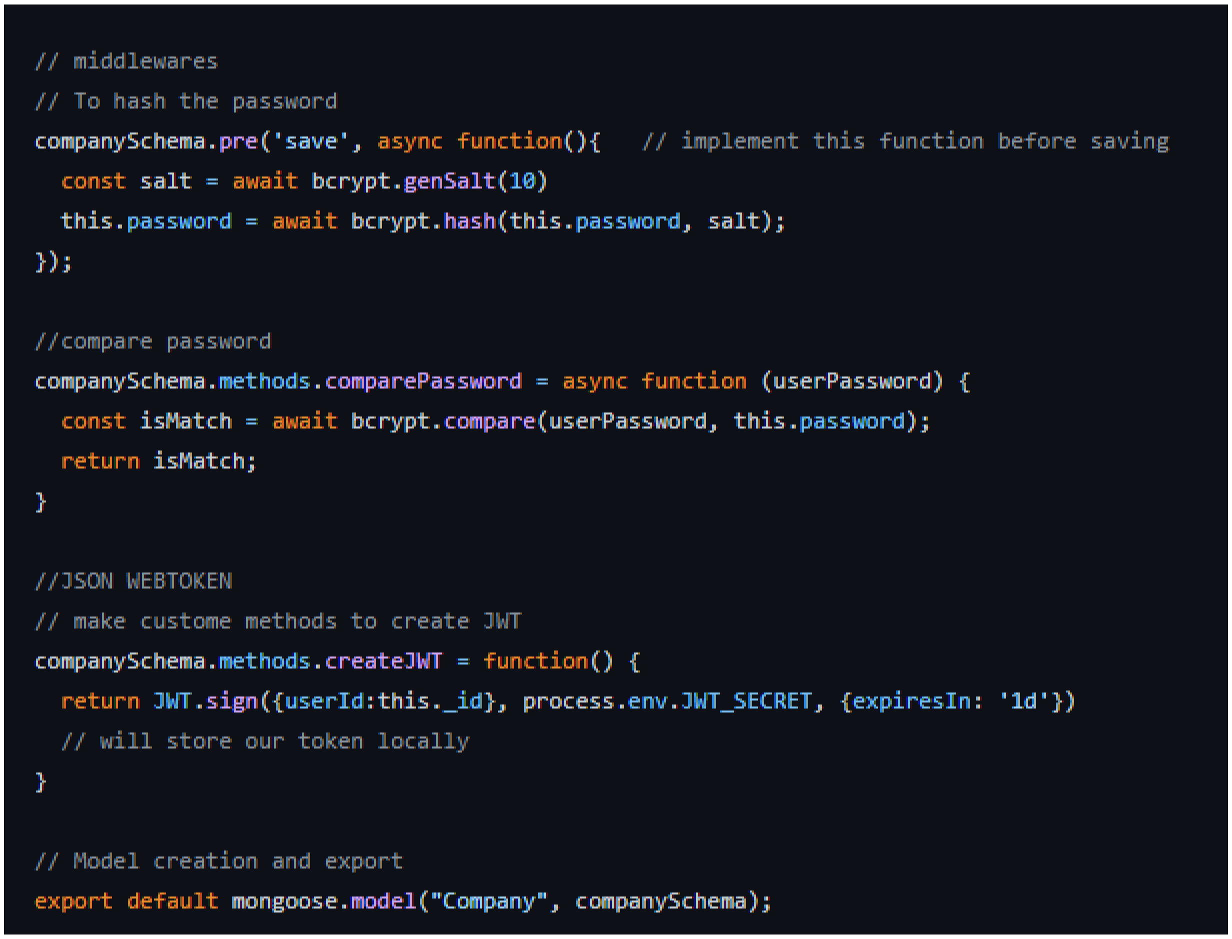


```
import express from "express";
import userAuth from "../middlewares/authMiddleware.js";
import { getUser, updateUser, getAppliedJobs } from "../controllers/userController.js";

const router = express.Router();



// GET user
router.post("/get-user", userAuth, getUser);


// UPDATE USER || PUT
router.put("/update-user", userAuth, updateUser);

// GET APPLIED JOBS
router.post("/get-applied-jobs", userAuth, getAppliedJobs);

export default router;
```

Models => companyModels (Code for handling company features)

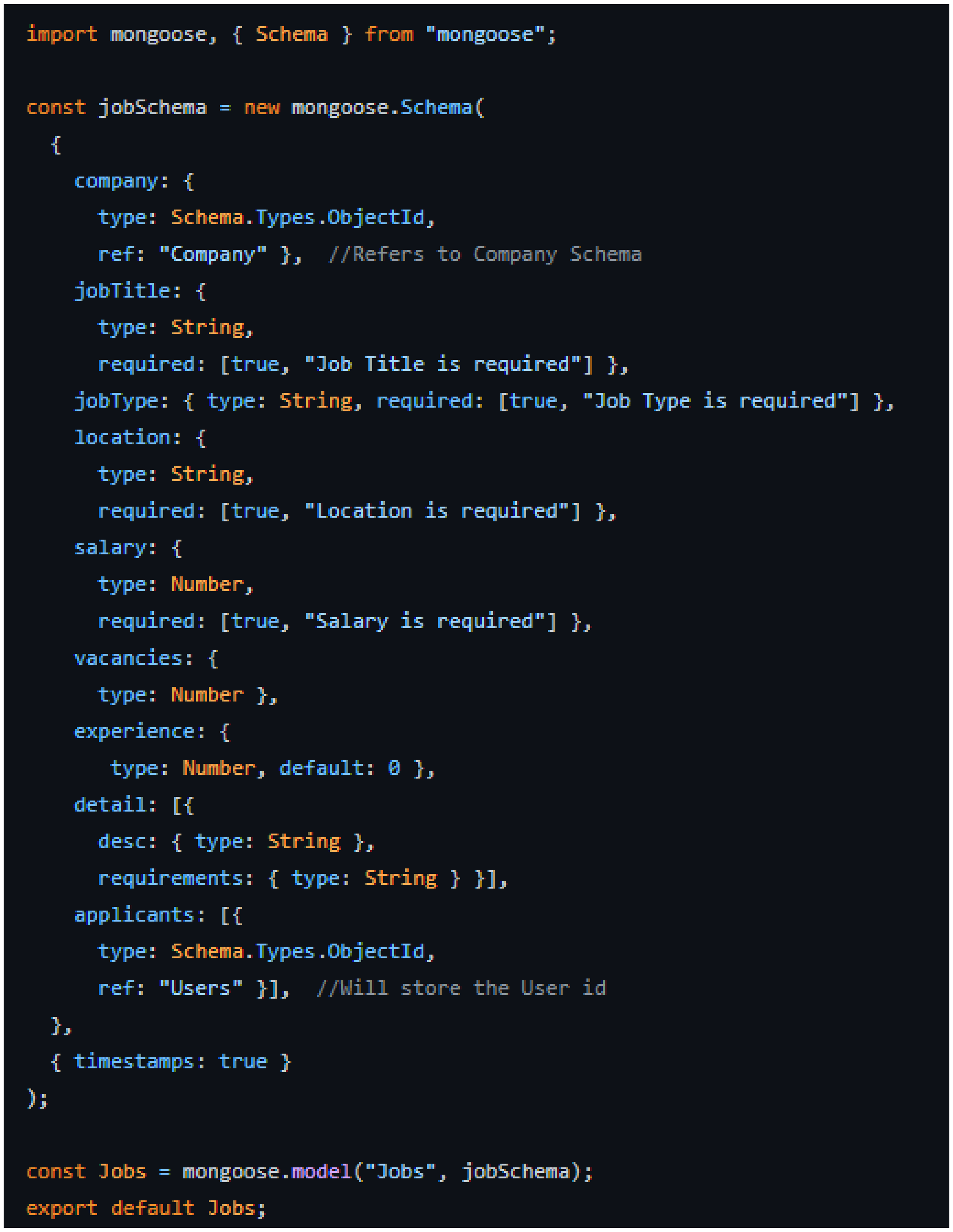


```
import mongoose, { Schema } from "mongoose";
import validator from "validator";
import bcrypt from 'bcryptjs';
import JWT from 'jsonwebtoken';

// Valid industries
const validIndustries = ["Technology", "Finance", "Healthcare", "Education", "Other"];

// Schema definition
const companySchema = new Schema({
  name: {
    type: String,
    required: [true, "Company Name is required"],
  },
  email: {
    type: String,
    required: [true, "Email is required"],
    unique: true,
    validate: validator.isEmail,
  },
  password: {
    type: String,
    required: [true, "Password is required"],
    minlength: [6, "Password must be at least"],
    select: true,
  },
  contact: { type: String },
  location: { type: String },
  about: { type: String },
  profileUrl: { type: String },
  jobPosts: [{ type: Schema.Types.ObjectId, ref: "Jobs" }],
},
// Additional schema options
{ timestamps: true }
);
```



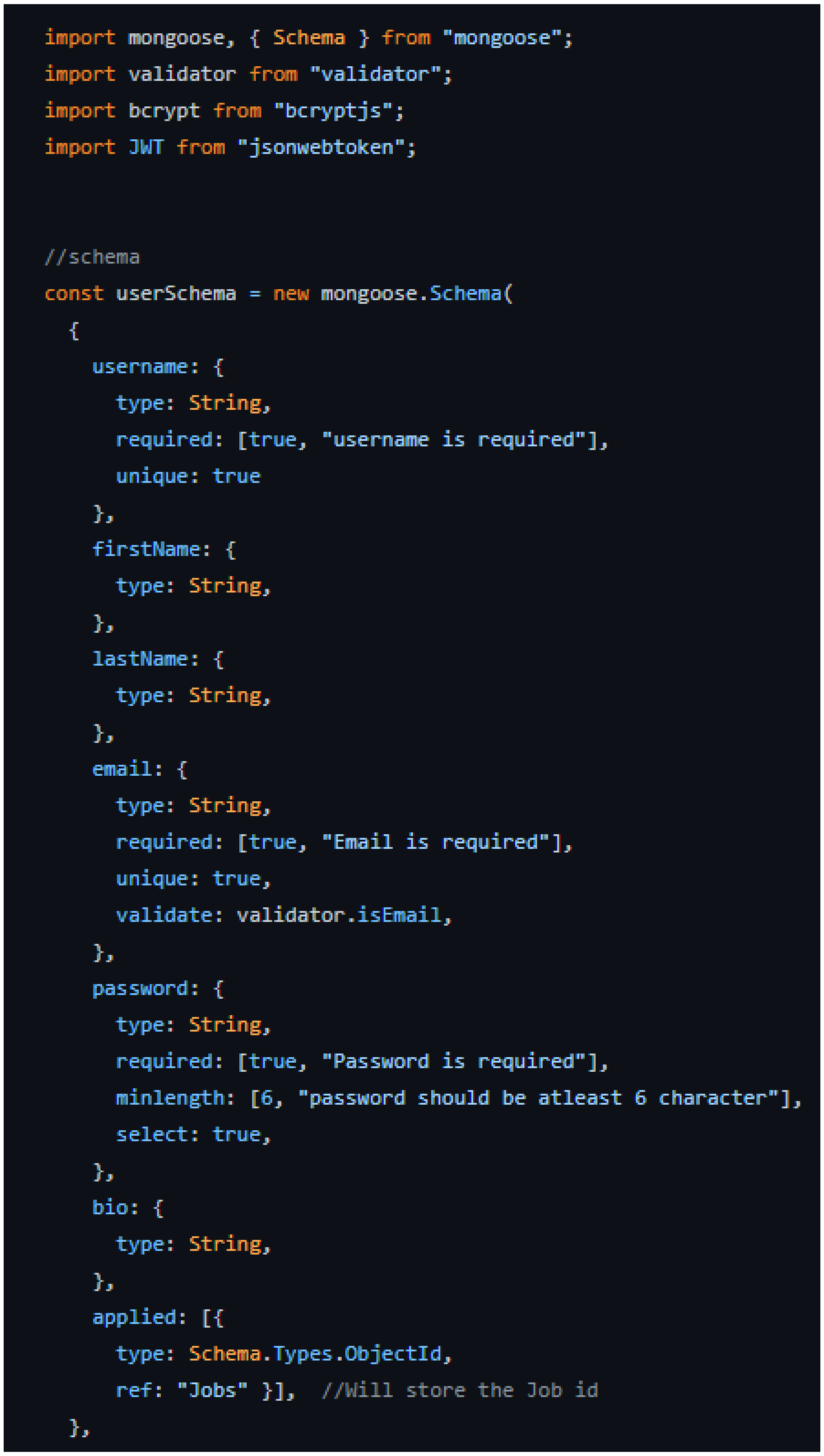
```
// middlewares
// To hash the password
companySchema.pre('save', async function(){   // implement this function before saving
  const salt = await bcrypt.genSalt(10)
  this.password = await bcrypt.hash(this.password, salt);
});

//compare password
companySchema.methods.comparePassword = async function (userPassword) {
  const isMatch = await bcrypt.compare(userPassword, this.password);
  return isMatch;
}

//JSON WEBTOKEN
// make custome methods to create JWT
companySchema.methods.createJWT = function() {
  return JWT.sign({userId:this._id}, process.env.JWT_SECRET, {expiresIn: '1d'})
  // will store our token locally
}

// Model creation and export
export default mongoose.model("Company", companySchema);
```

Models => jobModel (Code for handling job features)

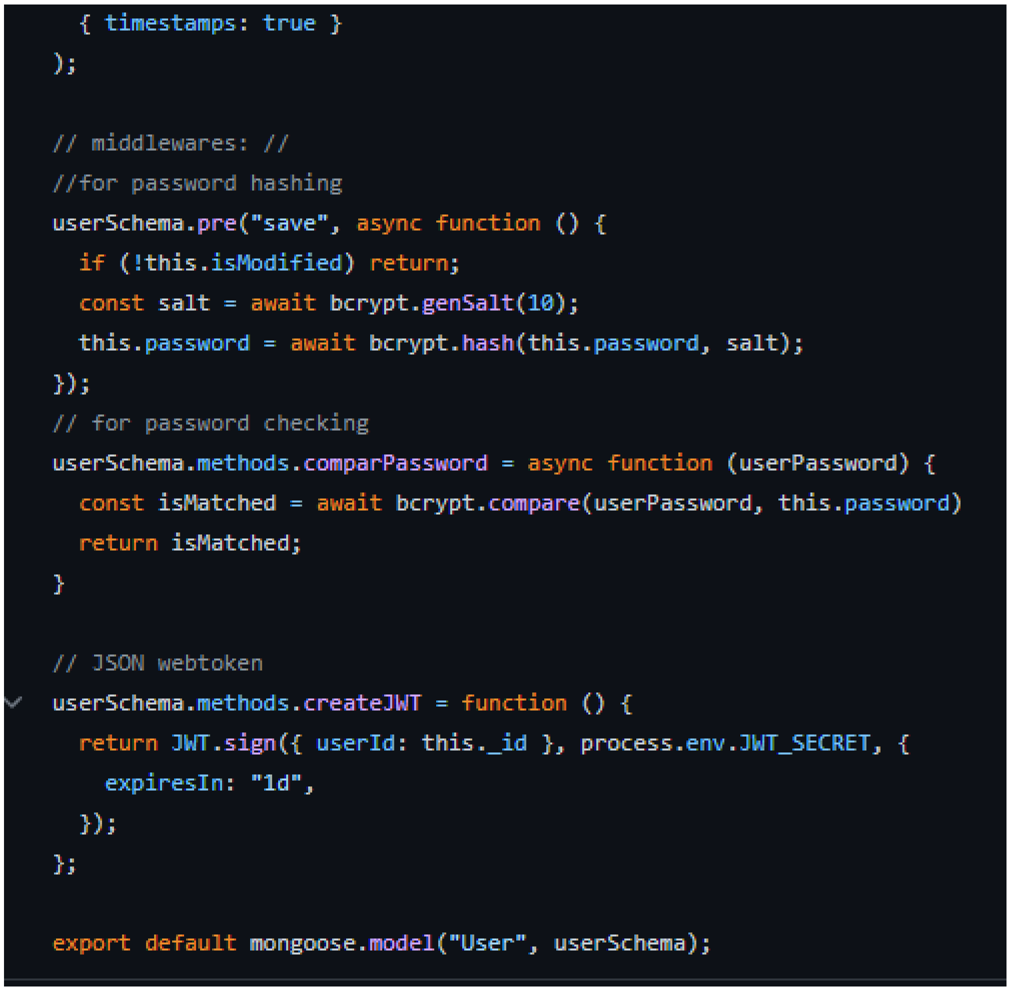


```
import mongoose, { Schema } from "mongoose";

const jobSchema = new mongoose.Schema(
  {
    company: {
      type: Schema.Types.ObjectId,
      ref: "Company" },  //Refers to Company Schema
    jobTitle: {
      type: String,
      required: [true, "Job Title is required"] },
    jobType: { type: String, required: [true, "Job Type is required"] },
    location: {
      type: String,
      required: [true, "Location is required"] },
    salary: {
      type: Number,
      required: [true, "Salary is required"] },
    vacancies: {
      type: Number },
    experience: {
       type: Number, default: 0 },
    detail: [{
      desc: { type: String },
      requirements: { type: String } }],
    applicants: [{
      type: Schema.Types.ObjectId,
      ref: "Users" }],  //Will store the User id
  },
  { timestamps: true }
);

const Jobs = mongoose.model("Jobs", jobSchema);
export default Jobs;
```

Models => userModel (Code for handling user login and signup features)

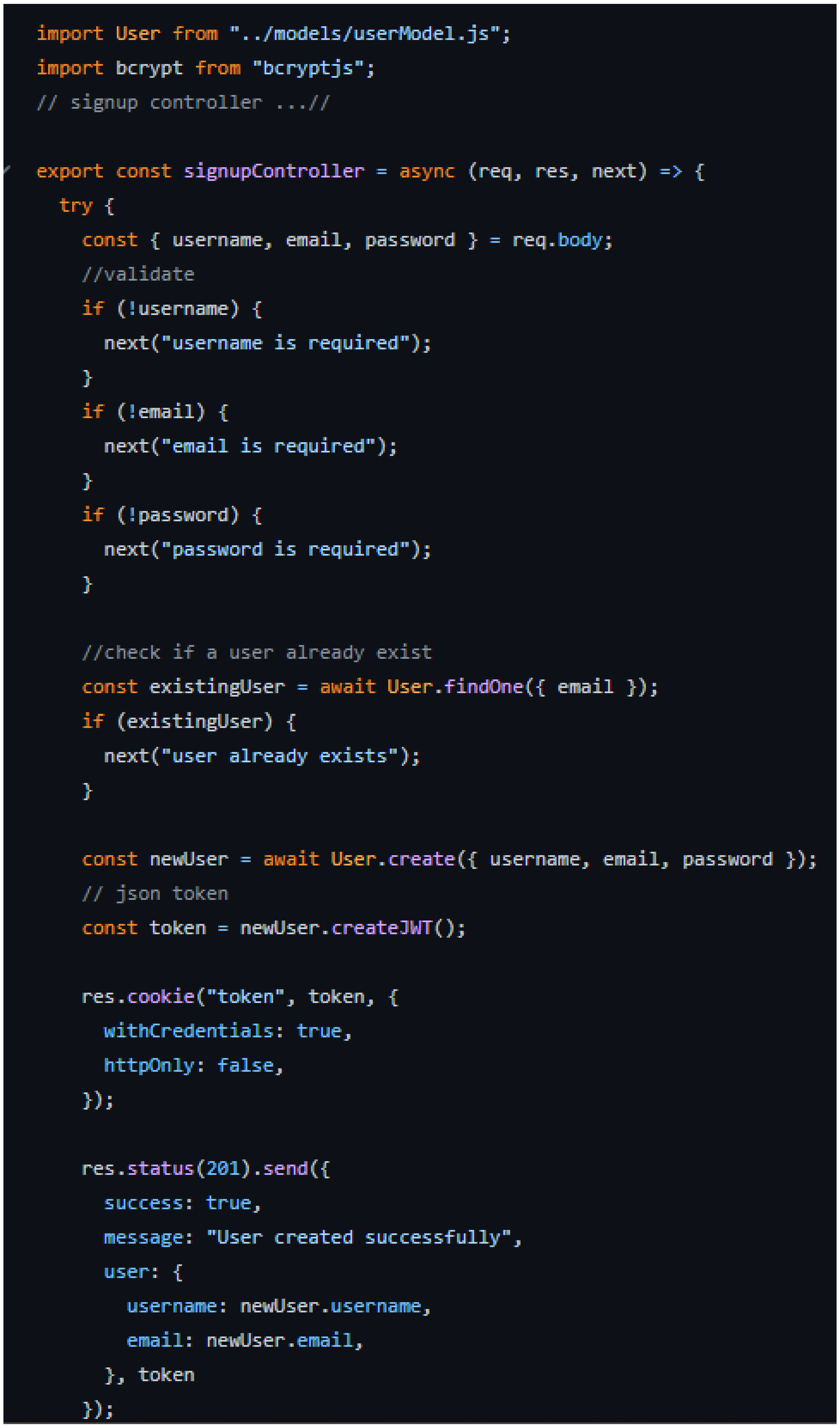


```
import mongoose, { Schema } from "mongoose";
import validator from "validator";
import bcrypt from "bcryptjs";
import JWT from "jsonwebtoken";


//schema
const userSchema = new mongoose.Schema(
  {
    username: {
      type: String,
      required: [true, "username is required"],
      unique: true
    },
    firstName: {
      type: String,
    },
    lastName: {
      type: String,
    },
    email: {
      type: String,
      required: [true, "Email is required"],
      unique: true,
      validate: validator.isEmail,
    },
    password: {
      type: String,
      required: [true, "Password is required"],
      minlength: [6, "password should be atleast 6 character"],
      select: true,
    },
    bio: {
      type: String,
    },
    applied: [{
      type: Schema.Types.ObjectId,
      ref: "Jobs" }],  //Will store the Job id
  },
```

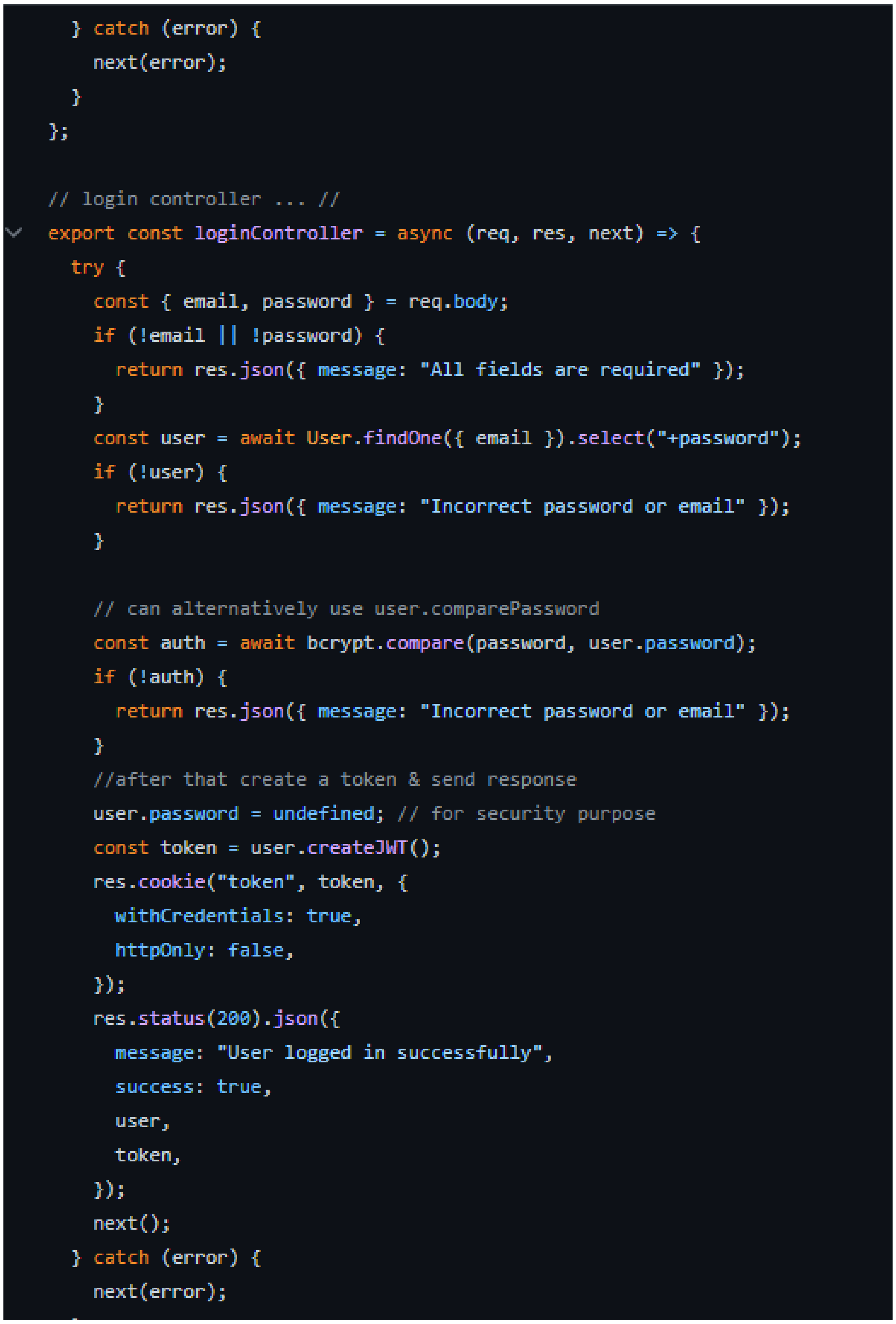


```
  { timestamps: true }
);

// middlewares: //
//for password hashing
userSchema.pre("save", async function () {
  if (!this.isModified) return;
  const salt = await bcrypt.genSalt(10);
  this.password = await bcrypt.hash(this.password, salt);
});
// for password checking
userSchema.methods.comparPassword = async function (userPassword) {
  const isMatched = await bcrypt.compare(userPassword, this.password)
  return isMatched;
}

// JSON webtoken
userSchema.methods.createJWT = function () {
  return JWT.sign({ userId: this._id }, process.env.JWT_SECRET, {
    expiresIn: "1d",
  });
};

export default mongoose.model("User", userSchema);
```

Controllers => autController (conditions for handling correct login and signup)



```
import User from "../models/userModel.js";
import bcrypt from "bcryptjs";
// signup controller ...//

export const signupController = async (req, res, next) => {
  try {
    const { username, email, password } = req.body;
    //validate
    if (!username) {
      next("username is required");
    }
    if (!email) {
      next("email is required");
    }
    if (!password) {
      next("password is required");
    }

    //check if a user already exist
    const existingUser = await User.findOne({ email });
    if (existingUser) {
      next("user already exists");
    }

    const newUser = await User.create({ username, email, password });
    // json token
    const token = newUser.createJWT();

    res.cookie("token", token, {
      withCredentials: true,
      httpOnly: false,
    });

    res.status(201).send({
      success: true,
      message: "User created successfully",
      user: {
        username: newUser.username,
        email: newUser.email,
      }, token
    });
```



```
    } catch (error) {
      next(error);
    }
  };

  // login controller ... //
  export const loginController = async (req, res, next) => {
    try {
      const { email, password } = req.body;
      if (!email || !password) {
        return res.json({ message: "All fields are required" });
      }
      const user = await User.findOne({ email }).select("+password");
      if (!user) {
        return res.json({ message: "Incorrect password or email" });
      }

      // can alternatively use user.comparePassword
      const auth = await bcrypt.compare(password, user.password);
      if (!auth) {
        return res.json({ message: "Incorrect password or email" });
      }
      //after that create a token & send response
      user.password = undefined; // for security purpose
      const token = user.createJWT();
      res.cookie("token", token, {
        withCredentials: true,
        httpOnly: false,
      });
      res.status(200).json({
        message: "User logged in successfully",
        success: true,
        user,
        token,
      });
      next();
    } catch (error) {
      next(error);
```

Controllers => companyController (conditions for handling all the features of the company)

```
import mongoose from "mongoose";
import Company from "../models/companyModel.js";
import bcrypt from "bcryptjs";

// Company Auth //
// Signup Controller
export const companySignupController = async (req, res, next) => {
  try {
    const { name, email, password } = req.body;

    // Validation
    if (!name) {
        next("company name is required");
    }
    if (!email) {
        next("email is required");
    }
    if (!password) {
        next("password is required");
    }

    // Check if a company already exists
    const existingCompany = await Company.findOne({ email });
    if (existingCompany) {
        next("Company already exists");
    }

    // Create a new company
    const newCompany = await Company.create({ name, email, password });
    //token
    const token = newCompany.createJWT();
    res.status(201).json({
      success: true,
      message: "Company created successfully",
      company: {
        name: newCompany.name,
        email: newCompany.email,
      },
      token,
    });
  } catch (error) {
    next(error);
  }
};

// Login Controller //
export const companyLoginController = async (req, res, next) => {
  try {
    const { email, password } = req.body;

    // Validation
    if (!email || !password) {
        return res.json({ message: "All fields are required" });
    }

    // Check if a company exists with the given email
    const company = await Company.findOne({ email }).select("+password");
    if (!company) {
        return res.json({ message: "Incorrect password or email" });
    }

    // Check if the provided password is correct
    // use a library like bcrypt for secure password comparison
    const auth = await bcrypt.compare(password, company.password);
    if (!auth) {
      return res.json({ message: "Incorrect password or email" });
    }
    // For security purposes, you might want to omit the password from the
response
    company.password = undefined;
    const token = company.createJWT()

    res.status(200).json({
      success: true,
      message: "Company logged in successfully",
      company,
      token,
    });
    next();
  } catch (error) {
    next(error)
  }
};


// other routes controller //

// Upadate Company Profile
export const updateCompanyProfile = async (req, res, next) => {
  const { name, contact, location, profileUrl, about } = req.body;

  try {
    // validation
    if (!name || !location || !about || !contact || !profileUrl) {
      next("Please Provide All Required Fields");
      return;
    }

    const id = req.user.userId;

    if (!mongoose.Types.ObjectId.isValid(id))
      return res.status(404).send(`No Company with id: ${id}`);

    const updateCompany = {
      name,
      contact,
      location,
      profileUrl,
      about,
      _id: id,
    };

    const company = await Company.findByIdAndUpdate(id, updateCompany, {
      new: true,
    });

    const token = company.createJWT();
    company.password = undefined;

    res.status(200).json({
      success: true,
      message: "Company Profile Updated SUccessfully",
      company,
      token,
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};



//GET ALL COMPANIES
export const getCompaniesController = async (req, res, next) => {
  try {
    const { search, sort, location } = req.query;

    //conditons for searching filters
    const queryObject = {};

    if (search) {
      queryObject.name = { $regex: search, $options: "i" };
    }

    if (location) {
      queryObject.location = { $regex: location, $options: "i" };
    }

    let queryResult = Company.find(queryObject).select("-password");

    // SORTING
    if (sort === "Newest") {
      queryResult = queryResult.sort("-createdAt");
    }
    if (sort === "Oldest") {
      queryResult = queryResult.sort("createdAt");
    }
    if (sort === "A-Z") {
      queryResult = queryResult.sort("name");
    }
    if (sort === "Z-A") {
      queryResult = queryResult.sort("-name");
    }

    // PAGINATIONS
    const page = Number(req.query.page) || 1;
    const limit = Number(req.query.limit) || 20;

    const skip = (page - 1) * limit;

    // records count
    const total = await Company.countDocuments(queryResult);
    const numOfPage = Math.ceil(total / limit);
    // move next page
    // queryResult = queryResult.skip(skip).limit(limit);

    // show mopre instead of moving to next page
    queryResult = queryResult.limit(limit * page);

    const companies = await queryResult;

    res.status(200).json({
      success: true,
      total,
      data: companies,
      page,
      numOfPage,
    });
  } catch (error) {
    // console.log(error);
    // res.status(404).json({ message: error.message });
    next(error)
  }
};
```

```
//GET  COMPANY JOBS
export const getCompanyJobListing = async (req, res, next) => {
  const { search, sort } = req.query;
  const id = req.user.userId;

  try {
    //conditons for searching filters
    const queryObject = {};

    if (search) {
      queryObject.location = { $regex: search, $options: "i" };
    }

    let sorting;
    //sorting || another way
    if (sort === "Newest") {
      sorting = "-createdAt";
    }
    if (sort === "Oldest") {
      sorting = "createdAt";
    }
    if (sort === "A-Z") {
      sorting = "name";
    }
    if (sort === "Z-A") {
      sorting = "-name";
    }

    let queryResult = await Company.findById({ _id: id }).populate({
      path: "jobPosts",
      options: { sort: sorting },
    });
    const company = await queryResult;

    res.status(200).json({
      success: true,
      company,
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};



// GET SINGLE COMPANY
export const getSingleCompany = async (req, res, next) => {
  try {
    const id = req.params.id;

    const company = await Company.findById({ _id: id });

    if (!company) {
      return res.status(200).send({
        message: "Company Not Found",
        success: false,
      });
    }

    company.password = undefined;
    res.status(200).json({
      success: true,
      data: company,
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};
```

Controllers => jobController (conditions for handling all features of the job)

```
import mongoose from "mongoose";
import Jobs from "../models/jobsModel.js";
import Companies from "../models/companyModel.js";
import Users from "../models/userModel.js";

export const createJob = async (req, res, next) => {
  try {
    const {
      jobTitle,
      jobType,
      location,
      salary,
      vacancies,
      experience,
      desc,
      requirements,
    } = req.body;

    if (
      !jobTitle ||
      !jobType ||
      !location ||
      !salary ||
      !requirements ||
      !desc
    ) {
      next("Please Provide All Required Fields");
      return;
    }

    const companyId = req.user?.userId;
    console.log('companyId:', companyId);
    if (!mongoose.Types.ObjectId.isValid(companyId))
      return res.status(404).send(`No Company with id: ${companyId}`);  //
if the company ID is a valid MongoDB ObjectId

    const jobPost = {
      jobTitle,
      jobType,
      location,
      salary,
      vacancies,
      experience,
      detail: { desc, requirements },
      company: companyId,
    };

    const job = new Jobs(jobPost);
    await job.save();

    const company = await Companies.findById(companyId);

    if (!company) {
        return res.status(404).send(`No Company with id: ${companyId}`);
      }
    console.log('Found job:', job);
    company.jobPosts.push(job._id);
    const updateCompany = await Companies.findByIdAndUpdate(companyId,
company, {
      new: true,
    });

    res.status(200).json({
      success: true,
      message: "Job Posted Successfully",
      job,
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};

export const updateJob = async (req, res, next) => {
  try {
    const {
      jobTitle,
      jobType,
      location,
      salary,
      vacancies,
      experience,
      desc,
      requirements,
    } = req.body;

    if (
      !jobTitle ||
      !jobType ||
      !location ||
      !salary ||
      !desc ||
      !requirements
    ) {
      next("Please Provide All Required Fields");
      return;
    }

    const companyId = req.user?.userId;

    if (!mongoose.Types.ObjectId.isValid(companyId))
      return res.status(404).send(`No Company with id: ${companyId}`);

    const { jobId } = req.params;

    const jobPost = {
      jobTitle,
      jobType,
      location,
      salary,
      vacancies,
      experience,
      detail: { desc, requirements },
      _id: jobId,
    };

    await Jobs.findByIdAndUpdate(jobId, jobPost, { new: true });

    res.status(200).json({
      success: true,
      message: "Job Post Updated Successfully",
      jobPost,
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};


//  to retrieve job posts based on query parameters
export const getJobPosts = async (req, res, next) => {
  try {
    const { search, sort, location, jtype, exp } = req.query;
    const types = jtype?.split(",");
    const experience = exp?.split("-");

    let queryObject = {};

    if (location) {
      queryObject.location = { $regex: location, $options: "i" };
    }

    if (jtype) {
      queryObject.jobType = { $in: types };
    }

    if (exp) {
      queryObject.experience = {
        $gte: Number(experience[0]) - 1,
        $lte: Number(experience[1]) + 1,
      };
    }

    if (search) {
      const searchQuery = {
        $or: [
          { jobTitle: { $regex: search, $options: "i" } },
          { jobType: { $regex: search, $options: "i" } },
        ],
      };
      queryObject = { ...queryObject, ...searchQuery };
    }

    let queryResult = Jobs.find(queryObject).populate({
      path: "company",
      select: "-password",
    });

    if (sort === "Newest") {
      queryResult = queryResult.sort("-createdAt");
    }
    if (sort === "Oldest") {
      queryResult = queryResult.sort("createdAt");
    }
    if (sort === "A-Z") {
      queryResult = queryResult.sort("jobTitle");
    }
    if (sort === "Z-A") {
      queryResult = queryResult.sort("-jobTitle");
    }

    const page = Number(req.query.page) || 1;
    const limit = Number(req.query.limit) || 20;
    const skip = (page - 1) * limit;

    const totalJobs = await Jobs.countDocuments(queryResult);
    const numOfPage = Math.ceil(totalJobs / limit);

    queryResult = queryResult.limit(limit * page);

    const jobs = await queryResult;

    res.status(200).json({
      success: true,
      totalJobs,
      data: jobs,
      page,
      numOfPage,
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};

export const getJobById = async (req, res, next) => {
  try {
    const { id } = req.params;

    const job = await Jobs.findById({ _id: id }).populate({
      path: "company",
      select: "-password",
    });

    if (!job) {
      return res.status(200).send({
        message: "Job Post Not Found",
        success: false,
      });
    }

    const searchQuery = {
      $or: [
        { jobTitle: { $regex: job?.jobTitle, $options: "i" } },
        { jobType: { $regex: job?.jobType, $options: "i" } },
      ],
    };

    let queryResult = Jobs.find(searchQuery)
      .populate({
        path: "company",
        select: "-password",
      })
      .sort({ _id: -1 });

    queryResult = queryResult.limit(6);
    const similarJobs = await queryResult;
    res.status(200).json({
      success: true,
      data: job,
      similarJobs,
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};

export const deleteJobPost = async (req, res, next) => {
  try {
    const { id } = req.params;

    await Jobs.findByIdAndDelete(id);

    res.status(200).send({
      success: true,
      message: "Job Post Deleted Successfully.",
    });
  } catch (error) {
    console.log(error);
    res.status(404).json({ message: error.message });
  }
};


// jobController.js

// ... (previous code)

export const applyJob = async (req, res, next) => {
  try {
    const userId = req.user?.userId;
    const jobId = req.params.jobId;

    if (!mongoose.Types.ObjectId.isValid(userId)) {
      return res.status(404).send(`No User with id: ${userId}`);
    }
```

```
if (!mongoose.Types.ObjectId.isValid(jobId)) {
  return res.status(404).send(`No Job with id: ${jobId}`);
}

// Fetch the job from the database
const job = await Jobs.findById(jobId);

// Check if the job is found
if (!job) {
  return res.status(404).json({
    success: false,
    message: "Job not found.",
  });
}

// Check if the user already applied for the job
if (job.applicants && job.applicants.includes(userId)) {
  return res.status(400).json({
    success: false,
    message: "You have already applied for this job.",
  });
}

// Add the user ID to the job's applications array
job.applicants.push(userId);

// Save the updated job post
const updatedJob = await job.save();


 // Update the user's applied array with the Job ID
 const user = await Users.findByIdAndUpdate(
  userId,
  { $push: { applied: jobId } },
  { new: true }
);

res.status(200).json({
  success: true,
      message: "Application submitted successfully",
      job: updatedJob,
    });
  } catch (error) {
    console.error(error);
    res.status(500).json({ message: "Internal Server Error" });
  }
};


export const showApplicants = async (req, res, next) => {
  try {
    const { jobId } = req.params;

    if (!mongoose.Types.ObjectId.isValid(jobId)) {
      return res.status(404).send(`No Job with id: ${jobId}`);
    }

    // Fetch the job from the database
    const job = await Jobs.findById(jobId);

    // Check if the job is found
    if (!job) {
      return res.status(404).json({
        success: false,
        message: "Job not found.",
      });
    }

    // Fetch users who have applied for the job
    const applicants = await Users.find({ applied: job._id });

    // Exclude sensitive information from the response
    const applicantsInfo = applicants.map(applicant => ({
      _id: applicant._id,
      username: applicant.username,
      email: applicant.email,
      firstName: applicant.firstName,
      lastName: applicant.lastName,
      createdAt: applicant.createdAt,
    }));

    res.status(200).json({
      success: true,
      data: applicantsInfo,
    });
  } catch (error) {
    console.error(error);
    res.status(500).json({ message: "Internal Server Error" });
  }
};
```

Controllers => testController

```
export const testGetController = (req, res) => {
  res.status(200).json({message: "success"});
};

export const testPostController = (req, res) => {
  const { name } = req.body;
  res.status(200).json({ message: "welcome", name: name });
};
```

Controllers => userController

```
import mongoose from "mongoose";
import Users from "../models/userModel.js";
import Jobs from "../models/jobsModel.js";




export const updateUser = async (req, res, next) => {
    const {
      firstName,
      lastName,
      email,
      bio,
    } = req.body;

    try {
      if (!firstName || !lastName || !email ||  !bio) {
        next("Please provide all required fields");
      }

      const id = req.user.userId;

      if (!mongoose.Types.ObjectId.isValid(id)) {
        return res.status(404).send(`No User with id: ${id}`);
      }

      const updateUser = {
        firstName,
        lastName,
        email,
        bio,
        _id: id,
      };

      const user = await Users.findByIdAndUpdate(id, updateUser, { new:
true });

      const token = user.createJWT();

      user.password = undefined;

      res.status(200).json({
        sucess: true,
        message: "User updated successfully",
        user,
        token,
      });
    } catch (error) {
      console.log(error);
      res.status(404).json({ message: error.message });
    }
  };


  export const getUser = async (req, res, next) => {
    try {
```

```
    const id = req.user.userId;

    const user = await Users.findById({ _id: id });

    if (!user) {
      return res.status(200).send({
        message: "User Not Found",
        success: false,
      });
    }

    user.password = undefined;

    res.status(200).json({
      success: true,
      user: user,
    });
  } catch (error) {
    console.log(error);
    res.status(500).json({
      message: "auth error",
      success: false,
      error: error.message,
    });
  }
};


export const getAppliedJobs = async (req, res, next) => {
  try {
    const userId = req.user?.userId;

    if (!mongoose.Types.ObjectId.isValid(userId)) {
      return res.status(404).send(`No User with id: ${userId}`);
    }

    // Fetch the user from the database
    const user = await Users.findById(userId);

    // Check if the user is found
    if (!user) {
      return res.status(404).json({
        success: false,
        message: "User not found.",
      });
    }

    // Fetch jobs based on the applied array in the user document
    const appliedJobs = await Jobs.find({ _id: { $in: user.applied } })
      .populate({
        path: "company",
        select: "-password",
      })
      .sort("-createdAt");

    // Extract relevant information about applied jobs
    const jobsInfo = appliedJobs.map(job => ({
      _id: job._id,
      jobTitle: job.jobTitle,
      jobType: job.jobType,
      location: job.location,
      salary: job.salary,
      company: job.company,
      createdAt: job.createdAt,
    }));

    res.status(200).json({
      success: true,
      data: jobsInfo,
    });
  } catch (error) {
    console.error(error);
    res.status(500).json({ message: "Internal Server Error" });
  }
};
```

Middlewares => authMiddleware

```
import JWT from 'jsonwebtoken';


const userAuth = async (req, res, next) =>{
    const authHeader = req.headers.authorization;
    if (!authHeader || !authHeader.startsWith("Bearer")){
        next("Auth Failed");
    }

    try {
        const token = authHeader.split(' ')[1];
        const payload = JWT.verify(token, process.env.JWT_SECRET);
        req.user = {userId: payload.userId};
        next();
    } catch (error) {
        console.log(error)
        next("Auth Failed");
    }
};


export default userAuth;
```

Middlewares => errrorMiddleware

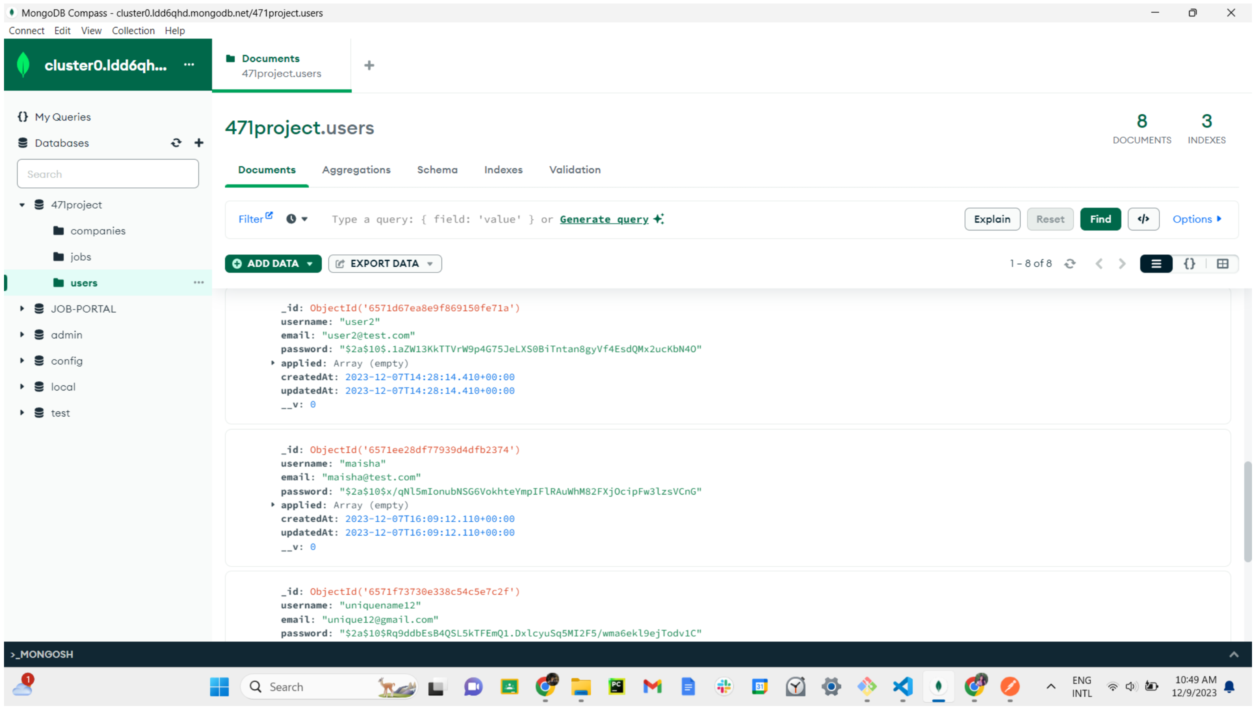
```
// error middleware || NEXT function
const errorMiddleware = (err, req, res, next) => {
  console.log(err);
  const defaultErrors = {
    statusCode: 500,
    message: err
  }

  //missing field error
  if (err.name === 'ValidationError') {
    defaultErrors.statusCode = 400,
    defaultErrors.message = Object.values(err.errors).map(item =>
item.message).join(',')
  }
  //duplicate error
  if (err.code && err.code === 11000) {
    defaultErrors.statusCode = 400;
    defaultErrors.message = `${Object.keys(err.keyValue)} field must be
unique`;
  }

  // handle other cases....

  /// send the response
  res.status(defaultErrors.statusCode).json({message:
defaultErrors.message});

export default errorMiddleware;
```

Config =>db (connection with mongoDB)

```
import mongoose from "mongoose";
import colors from "colors";

const connectDB = async () => {
  try {
    const conn = await mongoose.connect(process.env.MONGO_URL);
    console.log(`Connected to database
${mongoose.connection.host}`.green);
  } catch (error) {
    console.log(`Mongodb erro ${error}`.bgRed.white);
  }
};

export default connectDB;
```

Database

User:

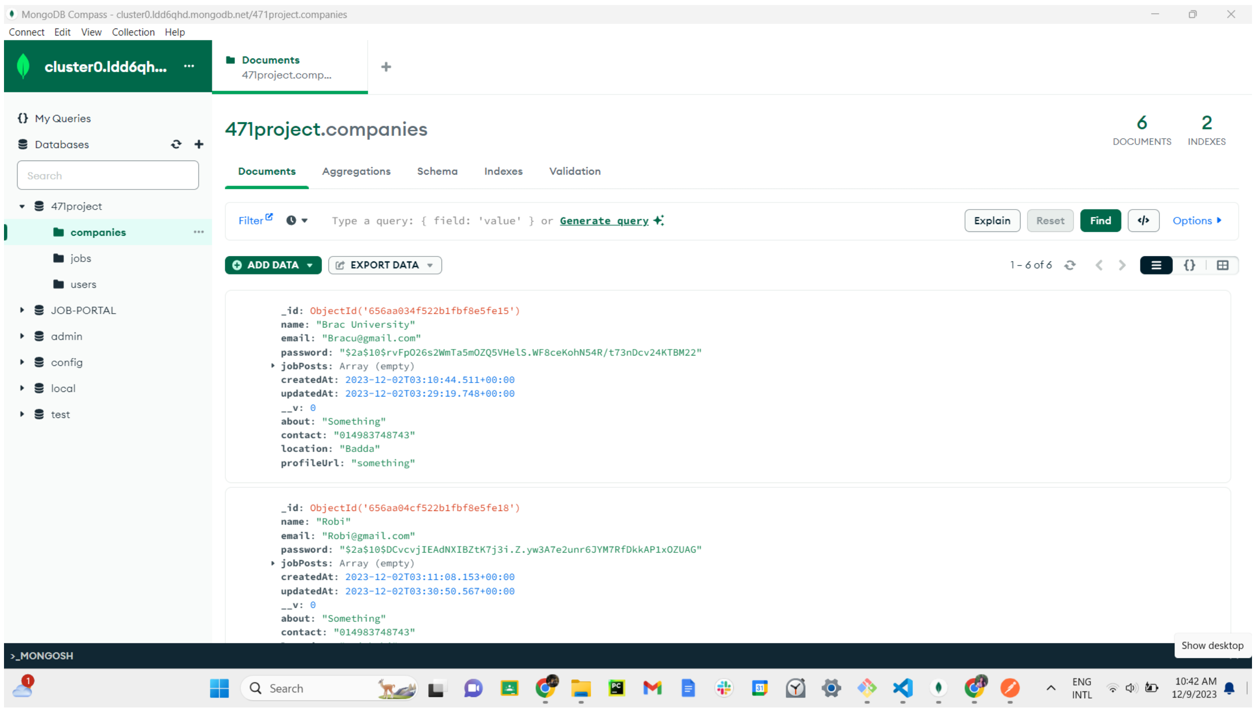

MongoDB Compass - cluster0.ldd6qhd.mongodb.net/471project.users
Connect Edit View Collection Help
cluster0.ldd6qh...
Documents
471project.users
My Queries
Databases
471project
companies
jobs
users
JOB-PORTAL
admin
config
local
test
471project.users
8 DOCUMENTS
3 INDEXES
Documents Aggregations Schema Indexes Validation
Filter
Type a query: { field: 'value' } or Generate query
Explain Reset Find Options
ADD DATA
EXPORT DATA
1 – 8 of 8
>_MONGOSH

```
_id: ObjectId('6571d67ea8e9f869150fe71a')
username: "user2"
email: "user2@test.com"
password: "$2a$10$.1aZW13KkTTVrW9p4G75JeLXS0BiTntan8gyVf4EsdQMx2ucKbN4O"
applied: Array (empty)
createdAt: 2023-12-07T14:28:14.410+00:00
updatedAt: 2023-12-07T14:28:14.410+00:00
__v: 0

_id: ObjectId('6571ee28df77939d4dfb2374')
username: "maisha"
email: "maisha@test.com"
password: "$2a$10$x/qNl5mIonubNSG6VokhteYmpIFlRAuWhM82FXjOcipFw3lzsVCnG"
applied: Array (empty)
createdAt: 2023-12-07T16:09:12.110+00:00
updatedAt: 2023-12-07T16:09:12.110+00:00
__v: 0

_id: ObjectId('6571f73730e338c54c5e7c2f')
username: "uniquename12"
email: "unique12@gmail.com"
password: "$2a$10$Rq9ddbEsB4QSL5kTFEmQ1.DxlcyuSq5MI2F5/wma6ekl9ejTodv1C"
```

Fields of the schema stored in database:

1. id
2. username
3. email
4. password
5. applied
6. createdAt
7. updatedAt

Company:

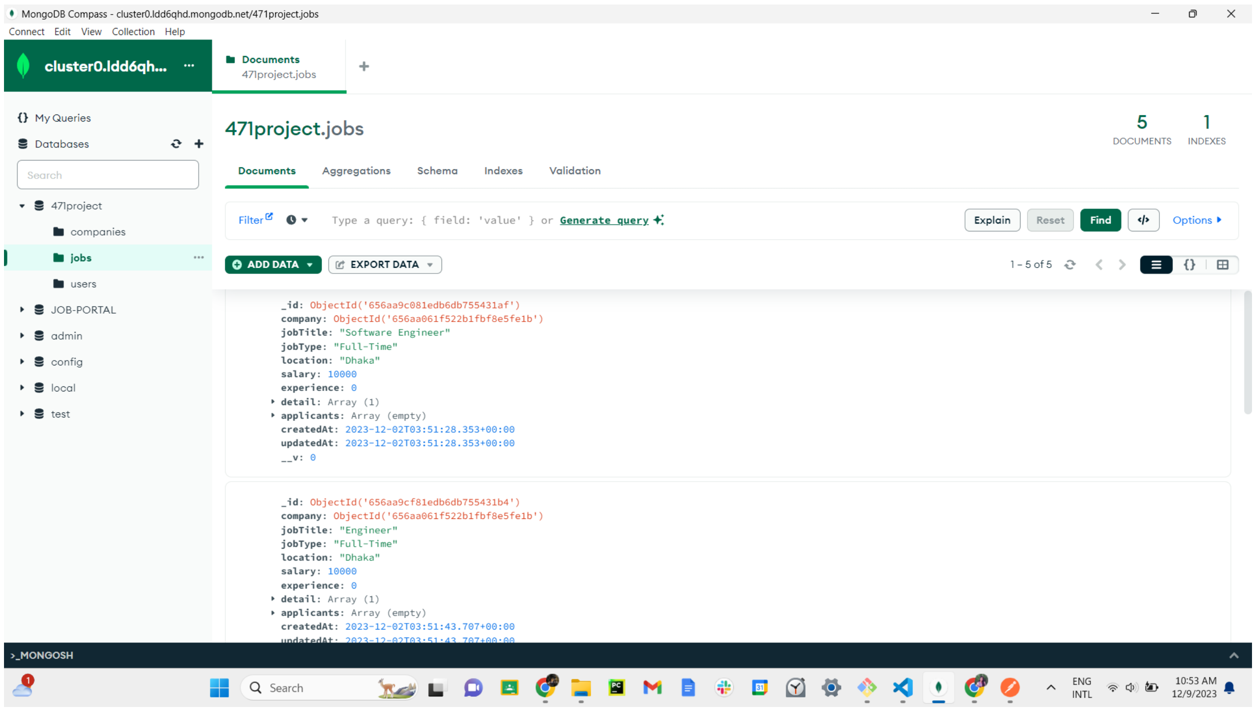


```
_id: ObjectId('656aa034f522b1fbf8e5fe15')
name: "Brac University"
email: "Bracu@gmail.com"
password: "$2a$10$rvFpO26s2WmTa5mOZQ5VHelS.WF8ceKohN54R/t73nDcv24KTBM22"
jobPosts: Array (empty)
createdAt: 2023-12-02T03:10:44.511+00:00
updatedAt: 2023-12-02T03:29:19.748+00:00
__v: 0
about: "Something"
contact: "014983748743"
location: "Badda"
profileUrl: "something"

_id: ObjectId('656aa04cf522b1fbf8e5fe18')
name: "Robi"
email: "Robi@gmail.com"
password: "$2a$10$DCvcvjIEAdNXIBZtK7j3i.Z.yw3A7e2unr6JYM7RfDkkAP1xOZUAG"
jobPosts: Array (empty)
createdAt: 2023-12-02T03:11:08.153+00:00
updatedAt: 2023-12-02T03:30:50.567+00:00
__v: 0
about: "Something"
contact: "014983748743"
```

Fields of the schema stored in database:

1. id
2. name
3. email
4. Password
5. jobPosts
6. createdAt
7. updatedAt
8. about
9. contact
10. location
11. profileUrl

JOB:



```
_id: ObjectId('656aa9c081edb6db755431af')
company: ObjectId('656aa061f522b1fbf8e5fe1b')
jobTitle: "Software Engineer"
jobType: "Full-Time"
location: "Dhaka"
salary: 10000
experience: 0
detail: Array (1)
applicants: Array (empty)
createdAt: 2023-12-02T03:51:28.353+00:00
updatedAt: 2023-12-02T03:51:28.353+00:00
__v: 0

_id: ObjectId('656aa9cf81edb6db755431b4')
company: ObjectId('656aa061f522b1fbf8e5fe1b')
jobTitle: "Engineer"
jobType: "Full-Time"
location: "Dhaka"
salary: 10000
experience: 0
detail: Array (1)
applicants: Array (empty)
createdAt: 2023-12-02T03:51:43.707+00:00
```

Fields of the schema stored in database:

1. id
2. company
3. jobTitle
4. jobType
5. location
6. Salary
7. experience
8. detail
    a. desc
    b. requirements
9. Applicants
10. createdAt
11. updatedAt

Technology

Frontend : React, Redux, Tailwind CSS
Backend: NodeJS, ExpressJS
Database: MongoDB

Github Repository

https://github.com/syedfaysel/471_project.git

Individual Contribution

NAME	ID	Individual Contribution
Syed Faysel Ahammad Rajo	21101078	Handled the styling, data fetching from backend, and the following features in the ***frontend*** : Company sign in, user sign in, company login, user login, view all jobs posted, view applicants' resume applied for a job, update user profile, update CV, view all applied jobs, searching, sorting, filtering.
Naomi Afrin Jalil	21201325	Handled the following features in the ***backend*** : Update user profile, user login, user sign in, apply for jobs, company login, company sign in, update company profile, categorize job, upload job, search job, search company, view applied jobs, show applicants for a job.
Anika Islam	21101298	Handled the following features in the ***frontend*** : Features: show all companies, show individual company page, upload job, update company profile, user profile view, show all jobs, show individual job page.