A

PROJECT REPORT

On

RecruiteX

At TECHPILE TECHNOLOGY PVT. LTD., LUCKNOW

Submitted Towards Partial Fulfillment of

Four-Year B Tech in
Computer Science& Engineering

Under the supervision of
Rahul Soni

SUBMITTED TO:	**SUBMITTED BY:**
AMIT CHAUHAN	**KM SUDHA**

BANSAL INSTITUTE OF ENGINEERING AND TECHNOLOGY LUCKNOW (UP)

Session 2024-2025

COMPLETION CERTIFICATE

This is to certify that **KM SUDHA** of **B TECH (COMPUTER SCIENCE AND ENGINEERING)** from **BANSAL INSTITUTE OF ENGINEERING AND TECHNOLOGY LUCKNOW(UP)** (Institute/University) was working on the project entitled “**RecruiteX**” developed on **"Mern Stack"** in Techpile Technology Pvt. Ltd. She was engaged with us during **10 July** to **10 September** for a period of **60 days.**

She has done an excellent job during her engagement with the Software Development & Testing Division of the company. She has completed her project during the training tenure. Her performance has been good and satisfactory.

I would like to take this opportunity to express my appreciation to **KM SUDHA** her work and wish her all the very best for her future endeavors.

Regards,

DivyaRai

Project Manager

Techpile Technology Pvt. Ltd.

Lucknow(U.P.)

Signature

PREFACE

Summer training is an important part of the engineering curriculum. The B TECH course summer training helps a student in getting acquainted with the manner in which her knowledge is being practically used outside her institute and this is normally different from what she has learnt from books. Hence, when the student switches from the process of learning to that of implementing her knowledge, She finds anabrupt change.This is exactly why summer training session during the B Tech curriculum becomes all the more important. Summer training is prescribed for the student of Technical College as a part of the Four-year degree course of engineering by the AKTU University. We are required to undergo summer training for a period of 60 days after the completion of the 3rdyear.

This training report describes in detail the training after the 3rd year session, which I completed at the ***Techpile Technology Pvt. Ltd...*** This report also gives the information about the organization and it's working along with the project undertaken in the training period.
The fundamental step used in **SDLC** process is based on the ISO 9001 guidelines. My aim was to follow the ISO guidelines and develop a perfect system.

The system development was organized into 5 major parts:

1. **RequirementGathering**
2. **Documentation/Design**
3. **Development**
4. **Coding**
5. **Testing**

ACKNOWLEDGEMENT

Apart from my effort, the success of the project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who express have been instrumental in the successful completion of this project.

I would like to express my deep and sincere gratitude to my supervisor Mr. **Rahul Soni** Sir **(**Techpile Technology Pvt. Ltd.) who gave me his full support and encouraged me to work in an innovative and challenging project for educational field. His wide knowledge and logical thinking gave me right direction all the time.

I am deeply grateful my project coordinator for his help and support provided at every step of the project. Last but not the least, I thank to all employees of **Techpile Technology Pvt. Ltd...** for their support and co-operation.

KM SUDHA

DECLARATION

This is to certify that the project report entitled "**RecruiteX**" is done by me is an authentic work carried out for the partial fulfillment of the requirements for the award of B Tech in**"(COMPUTER SCIENCE AND ENGINEERING)"**under the guidance of **Mr. Rahul Soni**. The matter embodied in this project work has not been submitted earlier for award of any degree or diploma to the best of my knowledge and belief.

KM SUDHA

INDEX

PAGE

LIST OF TABLES

LIST OF FIGURES

PAGE

1. INTRODUCTION

1.1. Overview of Organization

Techpile is founded by some young engineers who have mastered the IT sector, whose objective is to achieve the highest position in the IT sector across the country. Who are trying to achieve this objective by cooperating in various fields.

Techpile is an organization working in both software development and software training which aims to make all the specials of its client successful through their coding as well as to make students more competent to work with a well reputed organization.

We are proud of our high-quality standards. These standards allow us to provide our customers with reliable and error-free software applications, regardless of complexity. Our top-notch developers use the latest software methodologies and technologies. This means that they can concentrate on our client's business goals and keep them involved in every stage through the entire project. Our meticulous approach has helped us build our excellent track record with no failed or aborted projects. We are in the business of change, managing complexity with an unparalleled insight, looking beyond the horizon of IT with resources focused on solutions. Becoming successful is a skill but one cannot perfect it without practice.

Abstract

RecruiteX also supports customized job alerts, ensuring job seekers are notified of new opportunities that match their preferences. Employers can utilize AI-driven candidate matching to identify the best fits for their job postings, reducing time spent on recruitment. The platform's built-in communication tools allow for direct messaging between candidates and recruiters, fostering quicker responses and feedback. In addition, RecruiteX offers flexible subscription plans for companies of all sizes, making it accessible to both small businesses and large enterprises. Ultimately, RecruiteX is designed to create a more dynamic and efficient hiring ecosystem that adapts to the evolving needs of today's workforce.

1.2. Introduction

2. **RecruiteX** is a modern job-finding platform designed to connect job seekers with potential employers quickly and efficiently. The platform offers a streamlined experience, enabling users to

search, apply for jobs, and update their profiles with ease. RecruiteX caters to both job seekers looking for opportunities in various industries and recruiters seeking to fill positions with qualified candidates. With advanced filtering options, personalized job recommendations, and a user-friendly interface, RecruiteX aims to bridge the gap between talent and opportunity, making the job hunt simpler and more effective for everyone involved.

1.3. Objectives:

3. **Talent Empowerment:** Provide job seekers with a platform that empowers them to create comprehensive profiles, showcase their skills, and connect with relevant job opportunities across various industries.
4. **Seamless Recruitment:** Streamline the hiring process for employers by offering tools for posting jobs, filtering candidates, and managing applications efficiently.
5. **Career Growth:** Support career development through personalized job recommendations, resume-building resources, and interview preparation tools, helping candidates advance in their professional journey.
6. **Data Security:** Ensure a secure environment for both job seekers and employers, protecting sensitive information and maintaining confidentiality throughout the recruitment process.
7. **Enhanced Connectivity:** Facilitate direct communication between recruiters and job seekers, promoting faster and more effective interactions to create better employment matches.

1.4. Problem Definition

In this section, we will discuss the limitations and drawbacks of existing recruitment systems that necessitate the development of RecruiteX. The current methods for managing job placements and candidate records are often inefficient and prone to errors. Below are some of the key shortcomings:

- **Low Functionality**: Existing systems often have limited features, making it cumbersome to add or remove job postings and manage candidate records. This inefficiency may require multiple employees to handle simple tasks, leading to unnecessary resource allocation.

- **Erroneous Input and Output**: Human involvement in data entry can lead to errors, resulting in inaccurate records and reports. Such inaccuracies can negatively impact decision-making and overall performance.
- **Portability Issues**: Many current systems rely on manual processes, making data transport cumbersome. Carrying physical records complicates reporting and analysis, especially when generating monthly or yearly insights.
- **Security Concerns**: Data in manual systems is vulnerable to tampering, as physical records can be easily altered. A software solution like RecruiteX can enhance security through password protection, safeguarding sensitive information from unauthorized access.
- **Data Redundancy**: Manual systems often duplicate records across multiple registers, leading to inefficient data management and increased chances of inconsistency.
- **Processing Speed**: Manual record-keeping and calculations are time-consuming, hindering overall performance. RecruiteX can perform these tasks rapidly, allowing for quicker access to information and decisions.
- **Manual Errors**: The likelihood of human error increases with complex tasks, such as report generation and calculations. Automating these processes through software can significantly reduce errors and enhance accuracy.
- **Complexity of Work**: Updating or deleting records in manual systems often requires extensive corrections, leading to complexity and confusion. A digital solution simplifies these operations, making data management straightforward and efficient.

2.SYSTEM ANALYSIS

2.1. Objective:

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

Analysis specifies what the system should do.

2.2 SDLC Phases:

System Development Life Cycle (SDLC) mainly consists of the following 7 phases which can be detailed: -

2.2.2. Preliminary Investigation: -

This is the first phase of the system development life cycle. In this phase we tend to find out the needs of the client –what exactly does the client want? Before the development of any system the important point is to know the needs, objectives and scope of the system

❖ Feasibility Study: -

Feasibility study is thestepof preliminary study of the system development life cycle. Things are always easy at the beginning in any software process. In fact, nothing is in feasible with unlimited time and resources. But it is not the fact. So, practically we have to do in limited resources in a restricted time margin. So, for the system to be feasible, following points we have to consider.

The feasibility study is conducted to check whether the candidate system is feasible. The system which is selected to be the best against the criteria is there after designed and developed. The feasibility study takes in to consideration, the risks involved in the project development beforehand. Therefore, in this phase we have to do feasibility study which is the test of the website according to its work ability, impact on the organization, ability to meet user need and effective use of resources. We do the feasibility study for website to analyze the risks, costs and benefits relating to economics, technology and user organization. There are several types of feasibility depending on the aspect they cover. Import of these includes:

Technical Feasibility:

This is an important outcome of preliminary investigation. It comprises of following questions:

- Can the work of project bed one with the current equipment, existing software and available man power resource?
- If Technology is required what are the possibilities that it can bedeveloped?

Economic Feasibility:

It deals with question related to the economy. It comprises of the following questions: -

- Are there sufficient benefits in creating the system to make the costacceptable?
- Are the costs of not creating the system so great that the project must beundertaken?

Legal Feasibility:

It deals with the question related to the legal issues. It comprises of the following questions: -

- ContractSigning
- Software Licenseagreement
- Issues related to cyberlaws.
- Legal issues relating to the man powercontract.

Operational Feasibility

The operational feasibility consists of the following activity: -

- Will the system be useful if it is developed&implemented?
- Will there be resistance fromemployee?

Social & Behavioral Feasibility

It deals with the various issues related to the human behavior like: -

- Whether the user be able to adapt a new change ornot?
- Whether the ambiance we are providing suits the user or not?

❖ Report Approval: -

Request approval is the preliminary steps of system development lifecycle. Request approval is the phase in which all the requirements which would be provide in the system are stated. The request approval is a sort of agreement between the client and the company which is building this software. Both the parties should be mutually agreed on the stated requirements.

2.2.2 System Analysis: -

System analysis is the phase following the phase of the request approval. In this phase we tend to analyze the overall system which we have to build. System analysis is the crucial part in SDLC.

2.2.3 System Design: -

System design means the designing of the system. The System can be done in either of the following two ways: -

- Logical SystemDesign
- Physical SystemDesign

2.2.4 Coding: -

Coding is the phase in which a developer codes using any programming languages. Coding constitutes only20 % of the whole project and which is easier to write. The coding work is also done in the teams; development of the system is usually done under the modular programming style, which can be either top-down approach or bottom-up approach.

2.2.5 Testing: -

Testing is the phase in which the system that has been developed is tested. Testing comprises of the 60%ofthe overall development of the system. Testing of the system is important because testing aims to uncover the different errors in the system. There are various different testing techniques that can be used for the testing of the system.

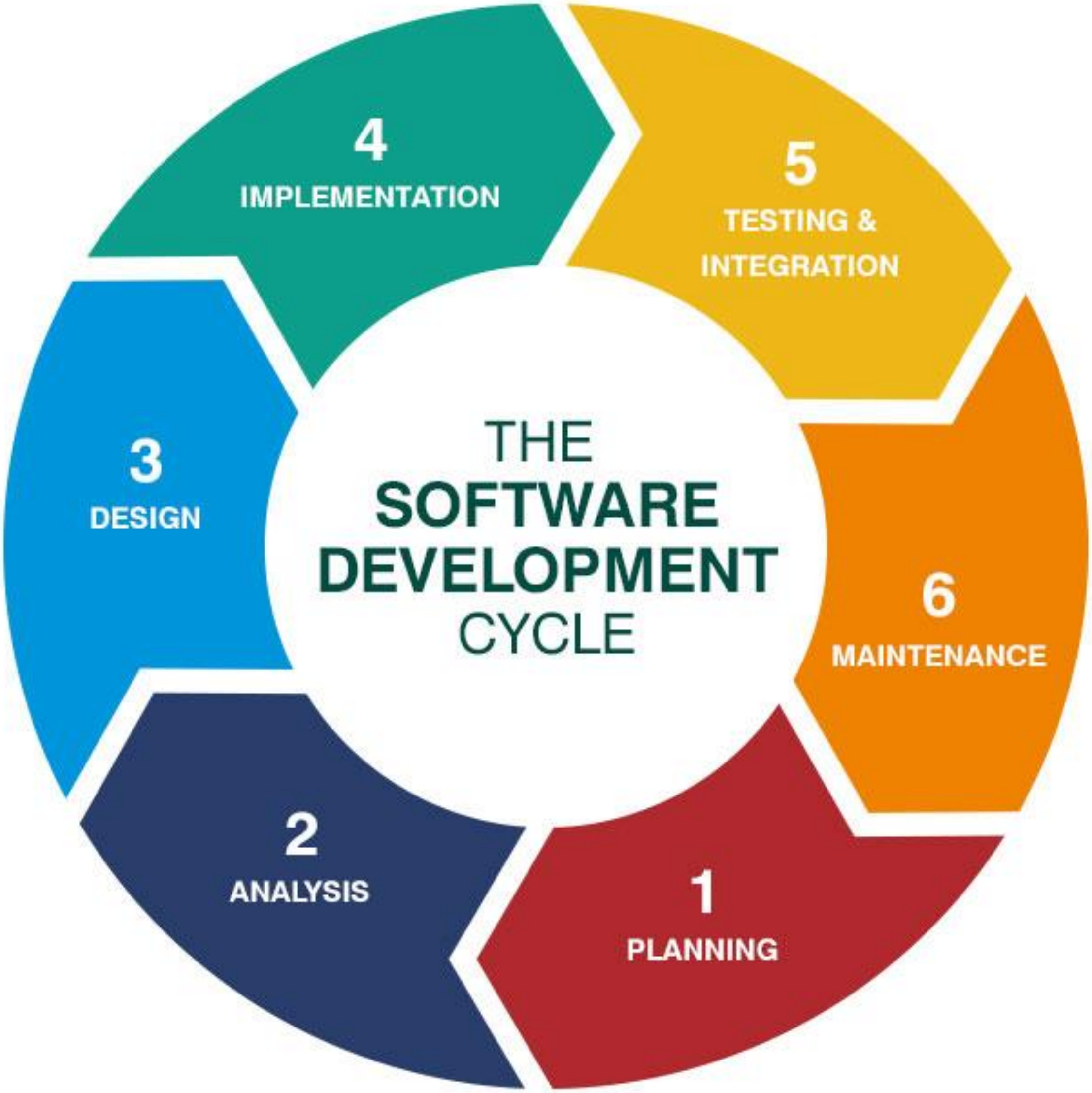
2.2.6 Implementation:

Implementation process involved the installation of software on user’s side. Implementation process actually depends on type of a system & various. Opting for suitable conversion approach is a step implementation. The conversion processes are as follows:-

- ParallelConversion
- Direct ConversionApproach
- Pilot ConversionApproach
- Phase In ConversionApproach

2.2.7 Maintenance: -

Merely developing the system is not important but also maintenance is important. The company that has built the system provides for some time free of cost maintenance to the client and after that period it is usually a paid service.


THE
SOFTWARE
DEVELOPMENT
CYCLE
1
PLANNING
2
ANALYSIS
3
DESIGN
4
IMPLEMENTATION
5
TESTING &
INTEGRATION
6
MAINTENANCE

2.3 Process Description

Gantt charts mainly used to allocate resources to activities. The resources allocated to activities include staff, hardware, and software. Gantt charts (named after its developer Henry Gantt) are useful for resource planning. A Gantt chart is special type of bar chart where each bar represents an activity. The bars are drawn along a timeline. The length of each bar is proportional to the duration of the time planned for the corresponding activity.

Gantt chart is a project scheduling technique. Progress can be represented easily in a Gantt chart, by coloring each milestone when completed. The project will start in the month of January and end after 4 months at the beginning of April.

2.4 PROJECT MODEL USED

Iterative Enhancement Model

➢ This model has the same phases as the waterfall model, but with fewer restrictions.Generally, the phases occur in the same order as in the waterfall model, but they may be conducted in several cycles.

➢ Useable product is released at the end of the each cycle, with each release providing additional functionality. Customers and developers specify as many requirements as possible and prepare a SRS document. Developers and customers then prioritize these requirements. Developers implement the specified requirements in one or more cycles of design, implementation and test based on the definedpriorities.

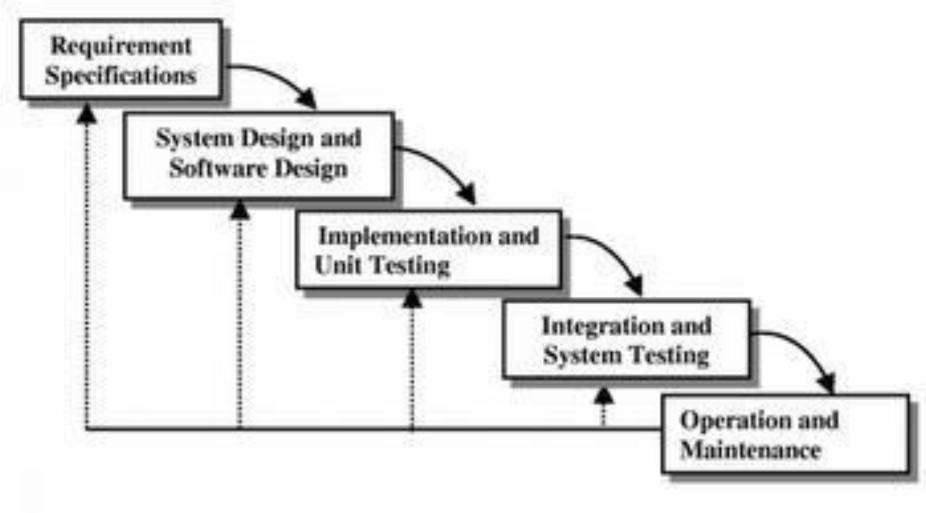
The procedure itself consists of the initialization step, the iteration step, and the Project Control List. The initialization step creates a base version of the system. The goal for this initial implementation is to create a product to which the user can react. It should offer a sampling of the key aspects of the problem and provide a solution that is simple enough to understand and implement easily. To guide the iteration process, a project control list is created that contains a record of all tasks that need to be performed. It includes such items as new features to be implemented and areas of redesign of the existing solution. The control list is constantly being revised as a result of the analysis phase.

The iteration involves the redesign and implementation of iteration is to be simple, straightforward, and modular, supporting redesign at that stage or as a task added to the project control list. The level of design detail is not dictated by the iterative approach. In a light-weight iterative project the code may represent the major source of documentation of the system; however, in a critical iterative project a formal Software Design Document may be used. The analysis of an iteration is based upon user feedback, and the program analysis facilities available. It involves analysis of the structure, modularity, usability, reliability, efficiency, & achievement of goals. The project control list is modified in light of the analysis results.

PHASES:

Incremental development slices the system functionality into increments (portions). In each increment, a slice of functionality is delivered through cross- discipline work, from the requirements to the deployment. The unified process groups increments/iterations into phases: inception, elaboration, construction, and transition.

- Inception identifies project scope, requirements (functional and non-functional) and risks at a high level but in enough detail that work can be estimated.
- Elaboration delivers a working architecture that mitigates the top risks and fulfills the non-functional requirements.
- Construction incrementally fills-in the architecture with production-ready code produced from analysis, design, implementation, and testing of the functional requirements.
- Transition delivers the system into the production operating environment.


Requirement Specifications
System Design and Software Design
Implementation and Unit Testing
Integration and System Testing
Operation and Maintenance

2.5 Data Flow Diagram

Introduction:

DFD is an acronym for the word Data Flow Diagram. DFD is ppictorial representation of the system. DFD is graphical representation of the flow of data through the information system. DFD are also used for the visualization of data processing (structured design). ADFD provides no information about the timings of the process, or about whether process will operate in parallel or sequence. DFD is an important technique for modeling system's high-level detail by showing how input data is transformed to output results through a sequence of functional transformations. DFD reveal relationships among between the various components in a program or system. The strength of DFD lies in the fact that using few symbols we are able to express program design in an easier manner. ADFD can be used to represent the following:

- External Entity sending and receiving data. Process that change the data.
- Flow of data within the system. Data Storage locations.

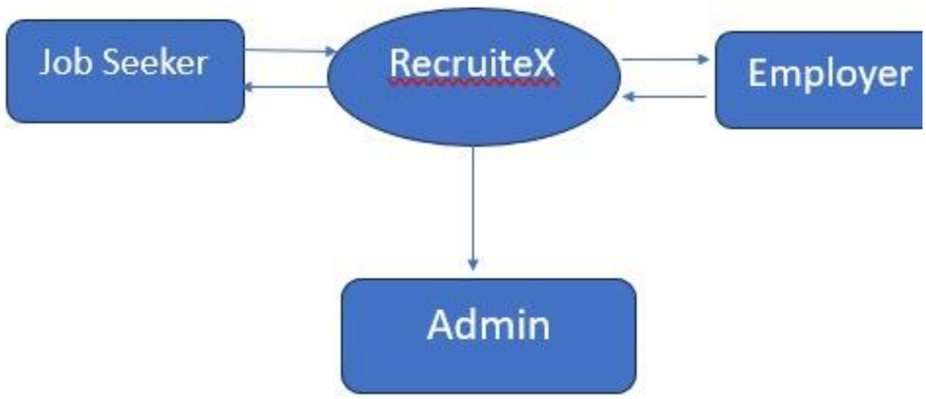
Uses of DFD:

Themain uses of data flow diagrams are as follows: -

DFD is amethodofchoiceforrepresentation of showingof information througha system because of the followingreasons:

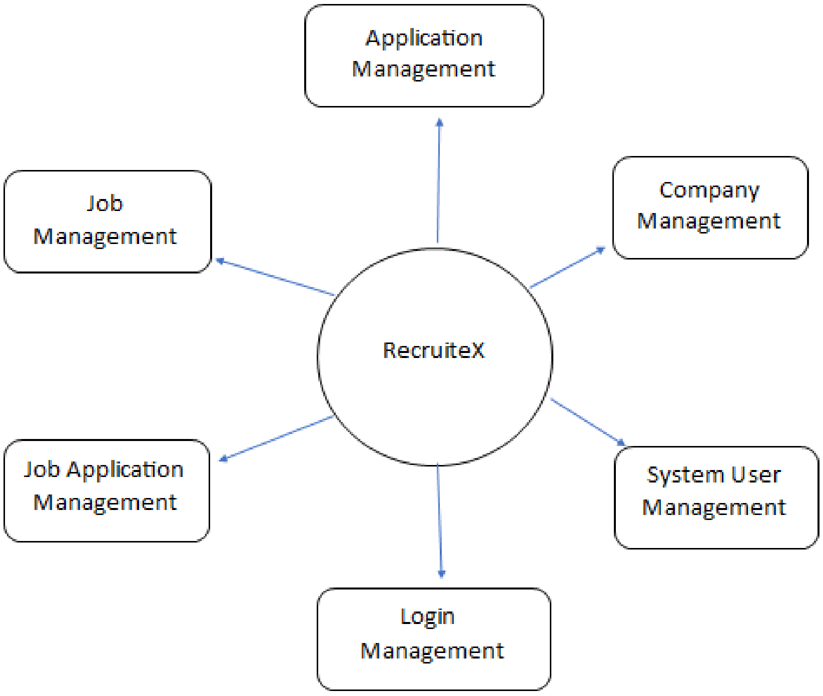
- DFDs are easier to understand bytechnical and non-technicalaudiences.
- DFDs can providea high-level system overview, complete with boundariesand connections toother system
- DFDs can provideadetailed representation of system components.

0 Level DFD


Job Seeker
RecruiteX
Employer
Admin

0 Level DFD

I


Application Management
Job Management
Company Management
RecruiteX
Job Application Management
System User Management
Login Management

0 SOFTWARE HARDWAREREQUIREMENT SPECIFICATION

A requirements specification for a software system is a complete description of the behavior of a system to be developed and it includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements.

Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints) Requirements are a sub-field of software engineering that deals with the elicitation, analysis, specification, and validation of requirements for software.

The software requirement specification document enlists all necessary requirements for project development. To derive the requirements we need to have clear and thorough understanding of the products to be developed. This is prepared after detailed communications with project team and the customer.

a. SERVER SIDE HARDWARE REQUIREMENT:

- AMD Athlon 64 with processor speed 2.8 or more
- 256 DDR Ram
- 40 GB Hard disk
- Network Interface card
- IIS
- CD-Drive

b. SERVER SIDE SOFTWARE REQUIREMENT:

- Windows
- Node Js , Express & React IDE VS Code
- MongoDb

CLIENT SIDE HARDWARE REQUIREMENT:

- Processor Dual core-based computer
- 4 GB Minimum RAM

- 20 GB HDD
- 100 Mbps LAN
- Web Browser

TO DEVELOP THIS PROJECT THE VARIOUS SOFTWARE RESOURCES ARE USED.

Front End	- HTML-CSS , Bootstrap& React
Back End	- MongoDb
Web Server	-Express Server
Technology	- Mern technology
Code-Behind Language	- Node & Express
IDE	- VS Code

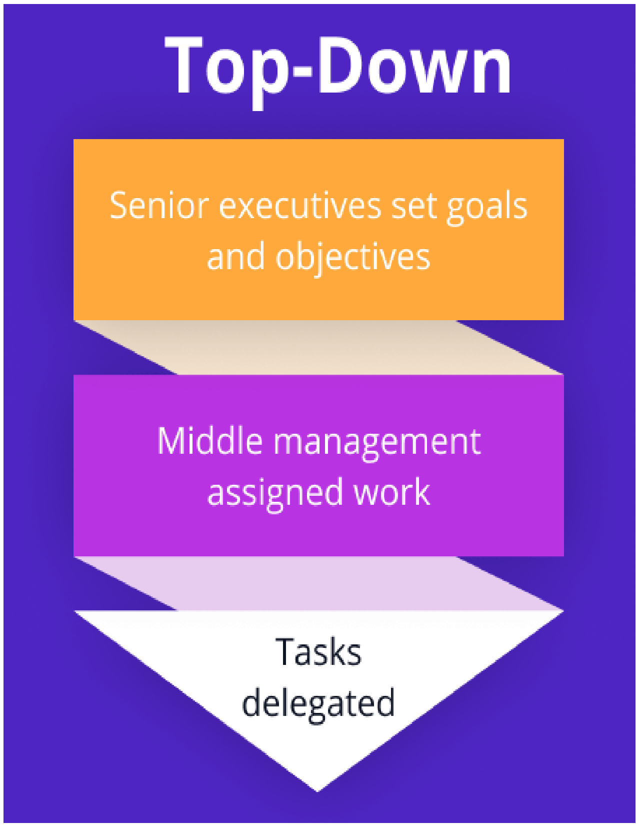
c. SUPPORT AND MAINTENANCE:

One-year free support for rectifying system bugs including front end and beck end will be provided. During warranty period Software Engineers will be responsible for removing bugs and improving it. After one year support can be extended @ 20% of the total product deployment cost.

1.SYSTEM DESIGN APPROACH

d. Top – Down designing:

The top - down designing approach started with major components of the system. It is a stepwise refinement which starts from an abstract design, in each steps the design is refined two or more concrete levels until we reach a level where no – more refinement is possible or not needed.


Top-Down
Senior executives set goals and objectives
Middle management assigned work
Tasks delegated

e. Bottom – Up designing:

In bottom – up designing the most basic and primitive components are designed first, and we proceed to higher level components. We work with layers of abstractions and abstraction are implemented until the stage is reached where the operations supported by the layer is complete.

Bottom-Up
Progress reported
Employees collaborate company-wide
Employees provide input and have decision-making authority

f. Following Approach:

In this project we are following **Mixed Approach** i.e. A combination of top – down and bottom – up. We are developing some of the components using top – down designing approach (e.g. the WebPages) and the some components in bottom – up designing approach (e.g. the middle tier classes).

1.BACKEND DESIGN

5.1 Descriptions of Models :-

5.1.1 AdminModel.js

```
const mongoose=require("mongoose");
let AdminSchema=new mongoose.Schema({

name:{type:String,required:true},
email:{type:String,required:true},
contact:{type:Number,required:true},
password:{type:String,required:true},
location:{type:String,required:true},
img:{type:String,required:true},
createdAt:{type:Date,default:Date.now()},
updatedAt:{type:Date,default:Date.now()}

})

constAdminTable= mongoose.model("Admin",AdminSchema);
module.exports={AdminTable}
```

5.1.2 ReccriuterModel.js

```
const mongoose=require("mongoose");
  let RecruiterSchema=new mongoose.Schema({

    name:{type:String,required:true},
    logo:{type:String,required:true},
    email:{type:String,required:true},
    contact:{type:Number,required:true},
    password:{type:String,required:true},
    location:{type:String,required:true},
    createdAt:{type:Date,default:Date.now()},
    updatedAt:{type:Date,default:Date.now()}

  })

   constrecruiterTable= mongoose.model("recruiters",RecruiterSchema);
   module.exports={recruiterTable}
```

5.1.3 SeekerModel

```
const mongoose=require("mongoose");
let SeekerSchema=new mongoose.Schema({

name:{type:String,required:true},
img:{type:String,required:true},
email:{type:String,required:true},
contact:{type:Number,required:true},
password:{type:String,required:true},
qualification:{type:String,required:true},
location:{type:String,required:true},
preference:{type:String},
resume:{type:String,required:true},
createdAt:{type:Date,default:Date.now()},
updatedAt:{type:Date,default:Date.now()}
})

constseekerTable= mongoose.model("seeker",SeekerSchema);
module.exports={seekerTable}
```

5.2 Routes

5.2.1 AdminRoute.js

```
const express=require("express")
constAdminRoute=express.Router()
const {AdminTable} =require('../models/AdminModel');
const {seekerTable}=require('../models/seekerModel');
const {recruiterTable}=require('../models/recruiterModel');
AdminRoute.post("/Admin-register",async(req,res)=>{
const {name,email,contact,password,location} =req.body;
let img=req.files.img;
img.mv("uploads/"+img.name,(err)=>{
if(err){
res.send(err)
}
})
const data=  newAdminTable({name:
name,img:img.name,email:email,contact:contact,password:password,location:location})
const result= await data.save()
res.json({
code :200,
data:result
})
})


AdminRoute.post("/Admin-login", async (req, res) => {
const {email,password}=req.body

const result = await AdminTable.findOne({email,password});
if(result) {
res.json({
code: 200,
message: "Login Successfull !...",
data:result
});
} else {
res.json({
code: 404,
message: "Invalid Email or Password"

});
}
});

AdminRoute.get("/admin-seekerlist",async(req,res)=>{
try{
const result= await seekerTable.find();
```

```
res.json({
code:200,
message:"Data found Successfull",
data:result
})

} catch(err){
console.log(err)
}
})

AdminRoute.get("/admin-recruiterlist",async(req,res)=>{
try{
const result= await recruiterTable.find();
res.json({
code:200,
message:"Data found Successfull",
data:result
})

} catch(err){
console.log(err)
}
})

AdminRoute.put("/admin-update/:_id",async(req,res)=>{
const _id=req.params._id;
const {name,email,contact,password,location}=req.body;
let img=req.files.img;
img.mv("uploads/"+img.name,(err)=>{
if(err){
res.send(err)
}
})


const result=await AdminTable.findByIdAndUpdate({_id:_id},{
$set:{name,email,contact,password,location,img:img.name}},
{new:true}
)
res.json({
code:200,
message:"Data updated Successfull",
data:result

})

})

AdminRoute.put("/admin-seekerblock/:_id",async(req,res)=>{
const _id=req.params._id;
const status=req.body.status;
const result=await seekerTable.findByIdAndUpdate({_id:_id},{
$set:{isBlock:status}},{new:true})
res.json({
code:200,
message:"Data updated Successfull",
data:result
})
})
AdminRoute.put("/admin-recruiterblock/:_id",async(req,res)=>{
const _id=req.params._id;
const status=req.body.status;
const result=await recruiterTable.findByIdAndUpdate({_id:_id},{
$set:{isBlock:status}},{new:true})
res.json({
code:200,
message:"Data updated Successfull",
data:result
})
})

module.exports={AdminRoute}
```

5.2.2 RecruiterRoute.js

```
const express=require("express")
constrecruiterRoute=express.Router()
const {recruiterTable} =require('../models/recruiterModel');
const {jobPostTable}=require('../models/jobpost');
const {appliedJobTable}=require('../models/appliedJob');
const {seekerTable}=require("../models/seekerModel");
recruiterRoute.post("/recruiter-register",async(req,res)=>{
const name=req.body.name;
const email=req.body.email;
const contact=req.body.contact;
const password=req.body.password;
const location=req.body.location;
let logo=req.files.logo;
logo.mv("uploads/"+logo.name,(err)=>{
if(err){
res.send(err)
}
})
const data=  newrecruiterTable({name:
name,email:email,contact:contact,password:password,location:location,logo:logo.name})
const result= await data.save()
res.json({
code :200,
data:result
})
})
recruiterRoute.post("/recruiter-login", async (req, res) => {
const {email,password}=req.body;

const result = await recruiterTable.findOne({email,password});
if (!result) {
res.json({
code: 302,
message: "Invaild Email or Password",
result: {}
})
} else {
```

```
if (result.isBlock) {
res.json({
code: 203,
message: "Your Account is Blocked",
result: {}
})
}
else {
res.json({
code: 200,
message: "Login successfull",
data: result
})
}
}
});

recruiterRoute.post("/recruiter-jobpost",async(req,res)=>{
try{
const {companyId,category,jobTitle,experience,jobType,vacancies,
jobLocation,salary,applyDate}=req.body;
const data= new  jobPostTable({companyId,category,jobTitle,experience,jobType,
vacancies,jobLocation,salary,applyDate })
const result=await data.save();
res.json({
code:200,
message:"Job posted successfull..",
data:result
})

}catch(err){
console.log(err);
}
})

recruiterRoute.post("/recruiter-postedjob",async(req,res)=>{
const{ companyId }=req.body;
constjobPost=await jobPostTable.find({companyId:companyId}).sort({createdAt:-1});
constfinnalData= await Promise.all(
jobPost.map(async(item)=>{
constcompanyDetails= await recruiterTable.findOne({_id:item.companyId});
return {
_id:item._id,
category:item.category,
jobTitle:item.jobTitle,
experience:item.experience,
jobType:item.jobType,
vacancies:item.vacancies,
jobLocation:item.jobLocation,
salary:item.salary,
applyDate:item.applyDate,
logo:companyDetails.logo,
name:companyDetails.name
}
})
)
res.json({
```

```
code:200,
message:"data found",
data:finnalData
})
})

recruiterRoute.post("/recruiter-applied",async(req,res)=>{
constcompanyId=req.body.companyId;
constappliedList=await appliedJobTable.find({companyId:companyId}).sort({createdAt:-1});
constfinalData=await Promise.all(
appliedList.map(async(item)=>{
constjobData= await jobPostTable.findOne({_id:item.jobId})
constseekerData=await seekerTable.findOne({_id:item.userId})
return {
_id:item?._id,
jobId:item?.jobId,
jobTitle:jobData?.jobTitle,
experience:jobData?.experience,
jobType:jobData?.jobType,
jobLocation:jobData?.jobLocation,
salary:jobData?.salary,
applyDate:jobData?.applyDate,
category:jobData?.category,
vacancies:jobData?.vacancies,
name:seekerData?.name,
email:seekerData?.email,
contact:seekerData?.contact,
resume:seekerData?.resume,
img:seekerData?.img
}
})
)
res.json({
code:200,
message:"data found",
data:finalData
})
})

recruiterRoute.put("/recruiter-update/:_id",async(req,res)=>{
const _id=req.params._id;
const {name,email,contact,location,password }=req.body;
let logo=req.files.logo
logo.mv("uploads/"+logo.name,(err)=>{
if(err){
res.send(err)
}
})
const result=await recruiterTable.findByIdAndUpdate({_id:_id},{
$set:{name,email,contact,location,password,logo:logo.name}},
{new:true}
)
res.json({
code:200,
message:"Data updated Successfull",
data:result
})
```

```
})
module.exports={recruiterRoute}
```

5.2.3 SeekerRoute.js

```
const express = require("express")
constseekerRoute = express.Router()
const{ seekerTable } = require('../models/seekerModel');
const{ jobPostTable } = require('../models/jobpost');
const{ recruiterTable } = require('../models/recruiterModel');
const{ appliedJobTable } = require('../models/appliedJob');
seekerRoute.post("/seeker-register", async (req, res) => {
const{ name, email, contact, password, qualification, location, preference } = req.body;
let img = req.files.img;
let resume = req.files.resume;
img.mv("uploads/" + img.name, (err) => {
if (err) {
res.send(err)
}
})
resume.mv("uploads/" + resume.name, (err) => {
if (err) {
res.send(err)
}
})
const data = new seekerTable({ name: name, img: img.name, email: email,
contact: contact, password: password, qualification: qualification, location: location,
preference: preference, resume: resume.name })
const result = await data.save()
res.json({
code: 200,
data: result
})
})
seekerRoute.post("/seeker-login", async (req, res) => {
const{ email, password } = req.body;
const result = await seekerTable.findOne({ email, password });
if (!result) {
res.json({
code: 302,
message: "Invaild Email or Password",
result: {}
})
} else {
if (result.isBlock) {
res.json({
code: 203,
message: "Your Account is Blocked",
result: {}
})
}
else {
res.json({
code: 200,
message: "Login successfull",
data: result
})
}
}
});

seekerRoute.get("/seeker-joblist", async (req, res) => {

constjobPost = await jobPostTable.find().sort({createdAt:-1});
constfinnalData = await Promise.all(
jobPost.map(async (item) => {
constcompanyDetails = await recruiterTable.findOne({ _id: item.companyId });
return {
_id: item?._id,
companyId: item?.companyId,
category: item?.category,
jobTitle: item?.jobTitle,
experience: item?.experience,
jobType: item?.jobType,
vacancies: item?.vacancies,
jobLocation: item?.jobLocation,
salary: item?.salary,
applyDate: item?.applyDate,
logo: companyDetails?.logo,
name: companyDetails?.name
}
})
)
res.json({
code: 200,
message: "data found",
data: finnalData
})
})

seekerRoute.post("/seeker-apply", async (req, res) => {
const{ jobId, companyId, userId } = req.body;
constisAppleid = await appliedJobTable.findOne({ jobId, userId })
if (isAppleid) {
res.json({
code: 301,
message: "Already Applied",
data: isAppleid
})
} else {
const data = new appliedJobTable({ jobId, companyId, userId });
const result = await data.save();
res.json({
code: 200,
message: "Applied Successfully",
data: result
})

}
})
seekerRoute.post("/seeker-applied", async (req, res) => {
constuserId = req.body.userId;
constappliedList = await appliedJobTable.find({ userId: userId }).sort({createdAt:-1});
```

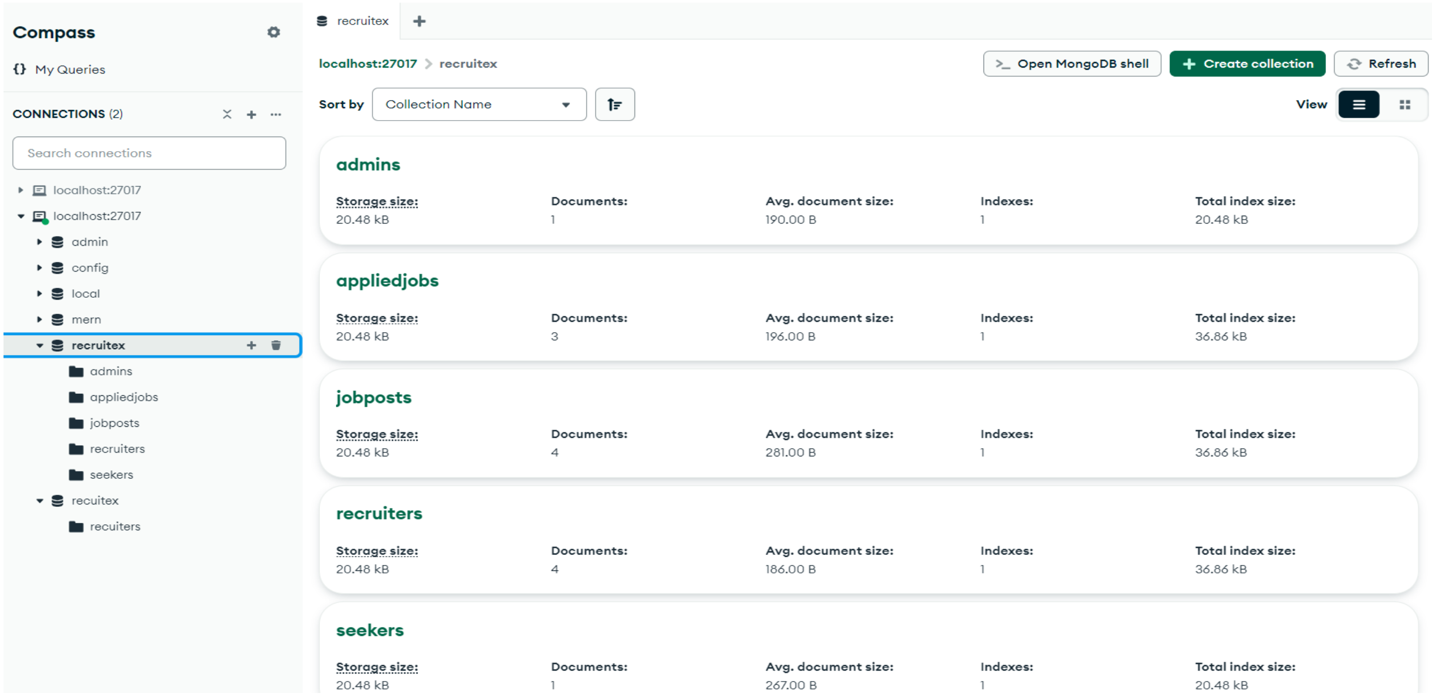
```
constfinalData = await Promise.all(
appliedList.map(async (item) => {
constjobData = await jobPostTable.findOne({ _id: item.jobId })
constcompanyData = await recruiterTable.findOne({ _id: item.companyId })
return {
_id: item?._id,
jobId: item?.jobId,
jobTitle: jobData?.jobTitle,
experience: jobData?.experience,
jobType: jobData?.jobType,
jobLocation: jobData?.jobLocation,
salary: jobData?.salary,
applyDate: jobData?.applyDate,
category: jobData?.category,
vacancies: jobData?.vacancies,
name: companyData?.name,
logo: companyData?.logo
}
})
)
res.json({
code: 200,
message: "data found",
data: finalData
})
})

seekerRoute.put("/seeker-update/:_id", async (req, res) => {
const _id = req.params._id;
const{ name, email, contact, location, password, qualification, preference } = req.body;
let img = req.files.img;
img.mv("uploads/" + img.name, (err) => {
if (err) {
res.send(err)
}
})
let resume = req.files.resume;
resume.mv("uploads/" + resume.name, (err) => {
if (err) {
res.send(err)
}
})
const result = await seekerTable.findByIdAndUpdate({ _id: _id }, {
$set: { name, email, contact, location, password, qualification, preference, img: img.name, resume: resume.name }
},
{ new: true }
)
res.json({
code: 200,
message: "Data updated Successfull",
data: result
})

})


module.exports = { seekerRoute }
```

5.2.4 Index.js

```
const express=require("express");
const mongoose=require("mongoose");
constcors=require("cors");
constexpressFileupload=require('express-fileupload');
const {recruiterRoute} =require('./routes/recruiterRoute')
const {seekerRoute}=require('./routes/seekerRoute')
const {AdminRoute}=require('./routes/AdminRoute')
const app=express();
app.use(express.json());
app.use(cors())
app.use("/upload",express.static("./uploads"));
//http://localhost:9000/upload/daya.jpg
app.use(expressFileupload())
constDbConnect=async()=>{
const con=await mongoose.connect("mongodb://localhost:27017/recruitex");
if(con){
console.log("Connected to MongoDB...");
}
}
DbConnect();
app.use("/api",AdminRoute);
app.use("/api",recruiterRoute);
app.use("/api",seekerRoute);

app.listen(9000,()=>{
console.log("Server is Running at 9000 port")
})
```

1 DATA MODELING

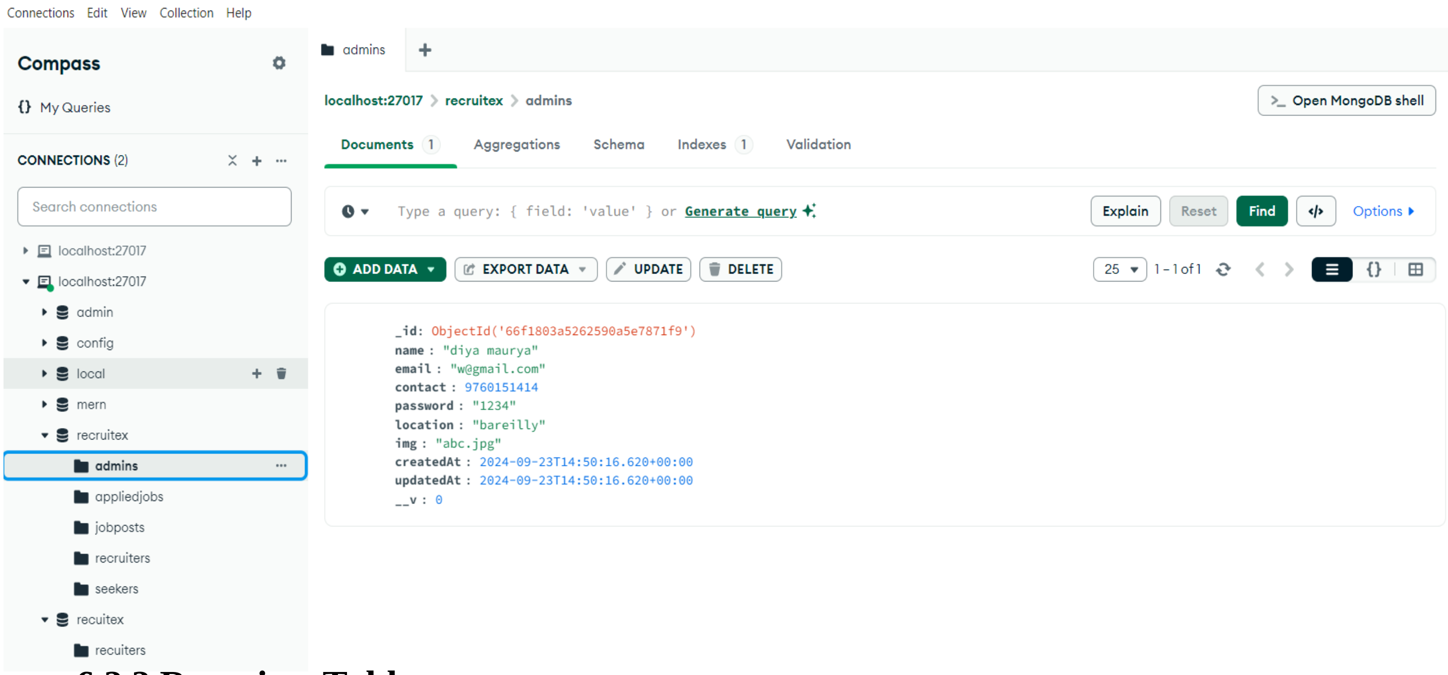
6.1 LIST OF TABLES: Group Table



Collection	Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
admins	20.48 kB	1	190.00 B	1	20.48 kB
appliedjobs	20.48 kB	3	196.00 B	1	36.86 kB
jobposts	20.48 kB	4	281.00 B	1	36.86 kB
recruiters	20.48 kB	4	186.00 B	1	36.86 kB
seekers	20.48 kB	1	267.00 B	1	20.48 kB

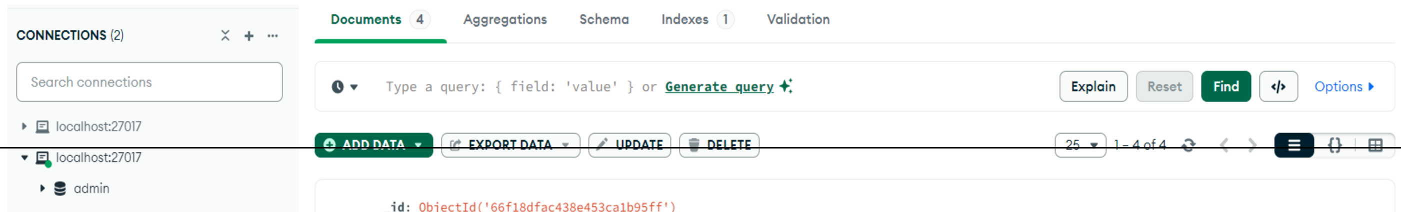
6.2. STRUCTURE OF TABLES:

6.2.1 Admin Table



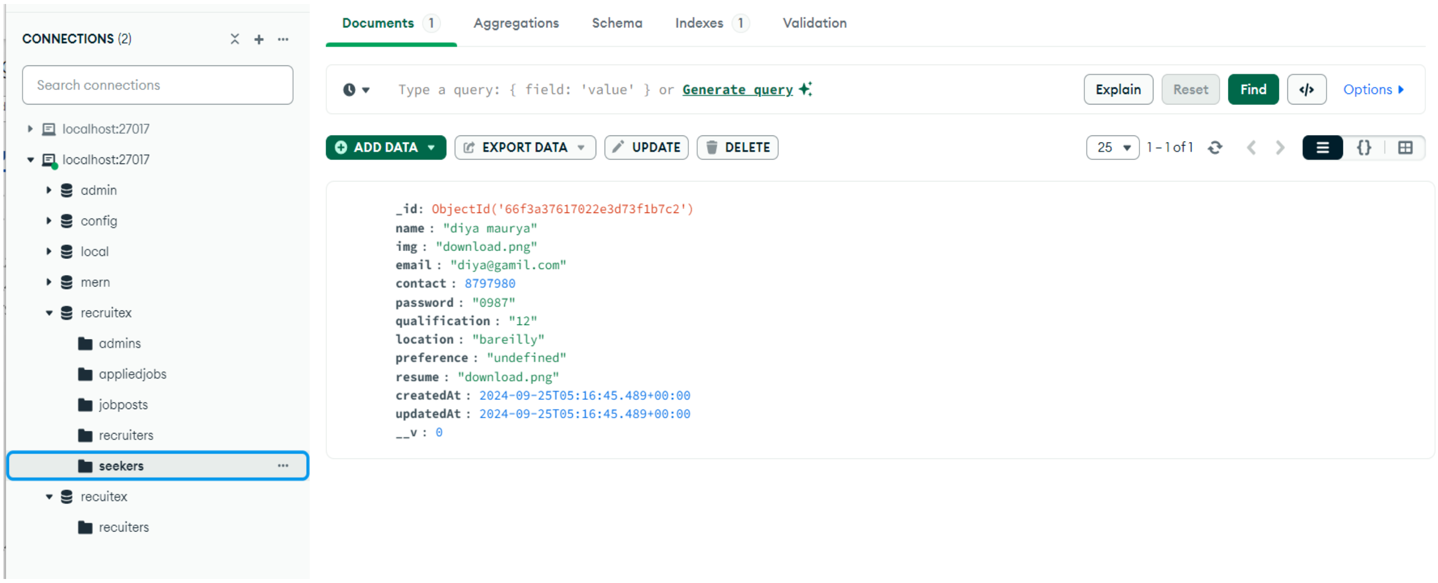
```
_id: ObjectId('66f1803a5262590a5e7871f9')
name : "diya maurya"
email : "w@gmail.com"
contact : 9760151414
password : "1234"
location : "bareilly"
img : "abc.jpg"
createdAt : 2024-09-23T14:50:16.620+00:00
updatedAt : 2024-09-23T14:50:16.620+00:00
__v : 0
```

6.2.2 Recruiter Table



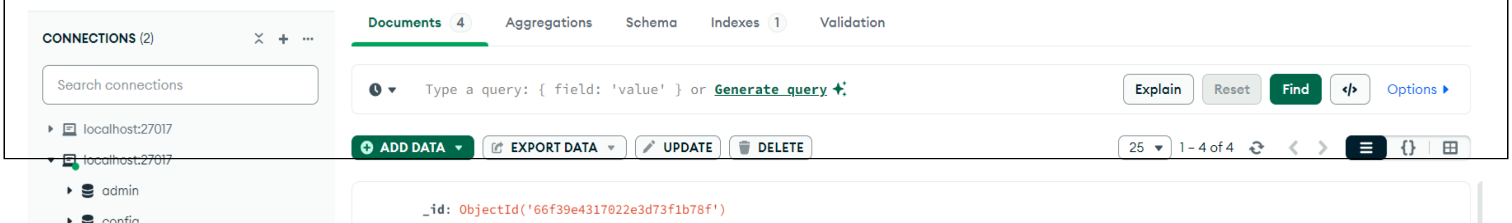
```
_id: ObjectId('66f18dfac438e453ca1b95ff')
```

6.2.3 Seeker



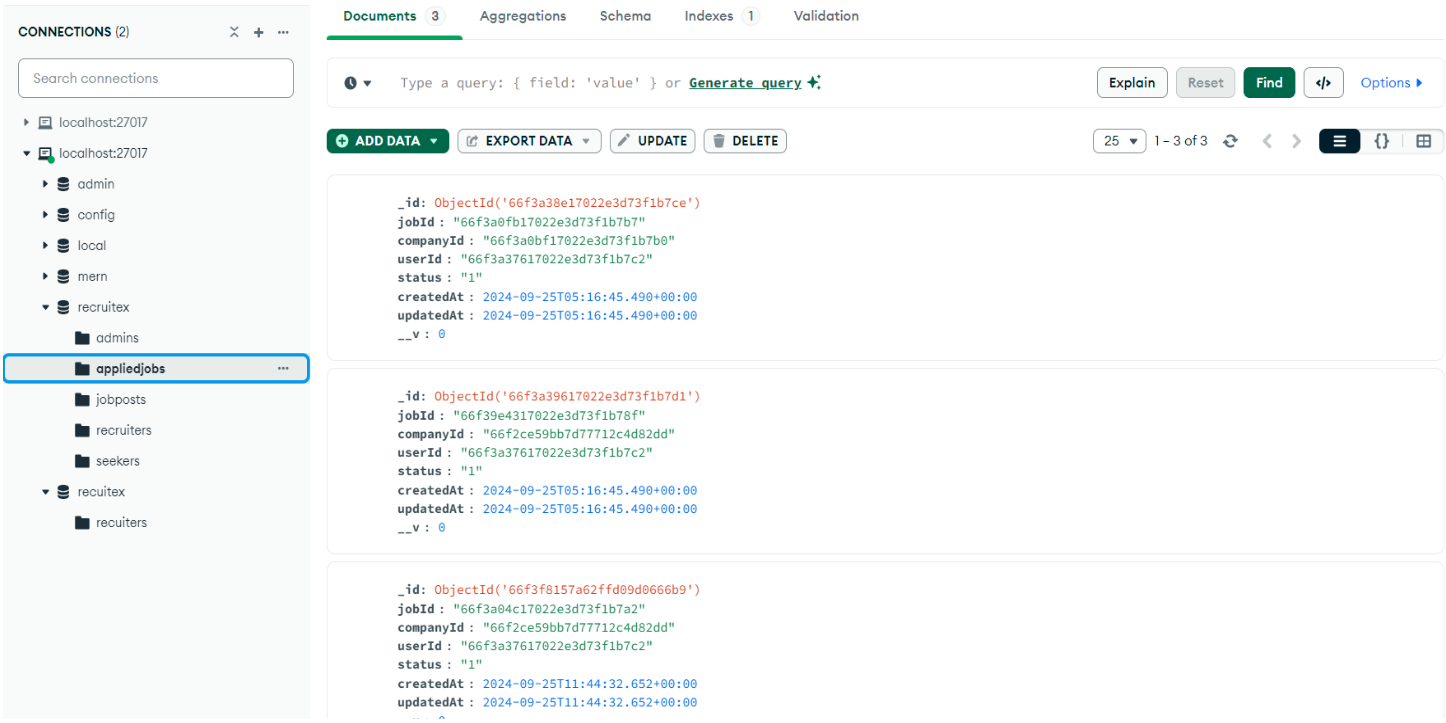
```
_id: ObjectId('66f3a37617022e3d73f1b7c2')
name : "diya maurya"
img : "download.png"
email : "diya@gamil.com"
contact : 8797980
password : "0987"
qualification : "12"
location : "bareilly"
preference : "undefined"
resume : "download.png"
createdAt : 2024-09-25T05:16:45.489+00:00
updatedAt : 2024-09-25T05:16:45.489+00:00
__v : 0
```

6.2.4 Job Post



```
_id: ObjectId('66f39e4317022e3d73f1b78f')
```

6.2.5 Applied Job



```
_id: ObjectId('66f3a38e17022e3d73f1b7ce')
jobId : "66f3a0fb17022e3d73f1b7b7"
companyId : "66f3a0bf17022e3d73f1b7b0"
userId : "66f3a37617022e3d73f1b7c2"
status : "1"
createdAt : 2024-09-25T05:16:45.490+00:00
updatedAt : 2024-09-25T05:16:45.490+00:00
__v : 0
```

```
_id: ObjectId('66f3a39617022e3d73f1b7d1')
jobId : "66f39e4317022e3d73f1b78f"
companyId : "66f2ce59bb7d77712c4d82dd"
userId : "66f3a37617022e3d73f1b7c2"
status : "1"
createdAt : 2024-09-25T05:16:45.490+00:00
updatedAt : 2024-09-25T05:16:45.490+00:00
__v : 0
```

```
_id: ObjectId('66f3f8157a62ffd09d0666b9')
jobId : "66f3a04c17022e3d73f1b7a2"
companyId : "66f2ce59bb7d77712c4d82dd"
userId : "66f3a37617022e3d73f1b7c2"
status : "1"
createdAt : 2024-09-25T11:44:32.652+00:00
updatedAt : 2024-09-25T11:44:32.652+00:00
```

7. TESTING

Testing is the integral part of any System Development Life Cycle insufficient and interested application tends to crash and result in loss of economic and manpower investment besides user’s dissatisfaction and downfall of reputation.

“Software Testing can be looked upon as one among much process, an organization performs, and that provides the last opportunity to correct any flaws in the developed system. Software Testing includes selecting test data that have more probability of giving errors.” The first step in System testing is to develop the plan that all aspect of system Complements, Correctness, Reliability and Maintainability.

Software is to be tested for the best quality assurance, an assurance that system meets the specification and requirement for its intended use and performance.

System Testing is the most useful practical process of executing the program with the implicit intention of finding errors that makes the programfail.

Types of Testing:

Black Box (Functional) Testing:

Testing against specification of system or component. Study it by examining its inputs and related outputs. Key is to devise inputs that have a higher likelihood of causing outputs that reveal the presence of defects. Use experience and knowledge of domain to identify such test cases. Failing this a systematic approach may be necessary. Equivalence partitioning is where the input to a program falls into a number of classes, e.g. positive numbers vs. negative numbers. Programs normally behave the same way for each member of a class. Partitions exist for both input and output. Partitions may be discrete or overlap. Invalid data (i.e. outside the normal partitions) is one or more partitions that should be tested.

Internal System design is not considered in this type of testing. Tests are based on requirements and functionality.

This type of test case design method focuses on the functional requirements of the software, ignoring the control structure of the program. Black box testing attempts to find errors in the following categories:

- Incorrect or missing functions.
- Interface errors.
- Errors in data structures or external database access.
- Performance errors.
- Initialization and termination errors.

White Box (Structural) Testing:

Testing based on knowledge of structure of component (e.g., by looking at source code). Advantage is that structure of code can be used to find out how many test case need to be performed. Knowledge of the algorithm (examination of the code) can be used to identify the equivalence partitions. Path testing is where the tester aims to exercise every independent execution path through the component. All conditional statements tested for both true and false cases. If a unit has n control statements, there program units than large ones. Flow graphs are a pictorial representation of the paths of control through a program (ignoring assignments, procedure calls and I/O statements).

Use flow graph to design test cases that execute each path. Static tools may be used to make this easier in programs that have a complex branching structure. Tools support. Dynamic program analyzers instrument a program with additional code. Typically, this will count how many times each statement is executed. At end print out report showing which statements have and have not been executed. Problems with flow graph derived testing:

- ❖ Data complexity could not take into account.
- ❖ We cannot test all paths in combination.
- ❖ In really only possible at unit and module testing stages because beyond that complexity is too high.

This testing is based on knowledge of the internal logic of an application's code. Also known as a Glass Box Testing Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.

Unit Testing:

Unit testing concentrates on each unit of the software as implemented in the code. This is done to check syntax and logical errors in programs. At this stage, the test focuses on each module individually, assuring that it functions properly as a unit. In our case, we used extensive white-box testing at the unit testing stage.

A developer and his team typically do the unit testing do the unit testing is done in parallel with coding; it includes testing each function and procedure.

Incremental Integration Testing:

Bottom-up approach for testing i.e. continuous testing of an application as new functionality is added; Application functionality and modules should be independent enough to test separately done by programmers or by testers.

Integration Testing:

Testing of integration modules to verify combined functionality after integration . Modules are typically code modules, individual applications, client and server and distributed systems.

Functional Testing:

This type of testing ignores the internal parts and focus on the output is as per requirement or not. Black box type testing geared to functionality requirements of an application.

System Testing:

Entire system is tested as per the requirements. Black box type test that is based on overall requirement specifications covers all combined parts of a system.

End-to-End Testing:

Similar to system testing involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database using network communications, or interacting with hardware, applications, or system if appropriate.

Regression Testing:

Testing the application as a whole for the modification in any module or functionality .Difficult to cover all the system in regression testing so typically automation tools are used for these testing types.

Acceptance Testing:

Normally this type of testing is done to verify if system meets the customer specified requirements. User or customers do this testing to determine whether to accept application.

Performance Testing:

Term often used interchangeably with “stress” and “load” testing, To check whether system meets performance requirements, Used different performance and load tools to do this.

Alpha Testing:

In house virtual user environment can be created for this type of testing. Testing is done at the end of development still minor design changes may be made as a result of such testing.

Beta Testing:

Testing typically done by end-users or others. This is final testing before releasing application for commercial purpose.

8. **Input-output Forms (SCREENSHOTS AND CODING)**

8.1 Coding Section

App.jsx

```
import "bootstrap/dist/css/bootstrap.css";

import "bootstrap/dist/js/bootstrap.bundle";

import { BrowserRouter, Route, Routes } from "react-router-dom";

import "./App.css";

import AdminLogin from "./Components/Admin/AdminLogin";

import AdminSeeker from "./Components/Admin/AdminSeekerList";

import AdminRecruiter from "./Components/Admin/AdminRecruiterList";

import SeekerLogin from "./Components/JobSeeker/SeekerLogin";

import SeekerRegister from "./Components/JobSeeker/SeekerRegister";

import About from "./Components/LandingPage/About";

import Contact from "./Components/LandingPage/Contact";

import FindJob from "./Components/LandingPage/FindJob";

import Footer from "./Components/LandingPage/Footer";

import Home from "./Components/LandingPage/Home";

import NavBar from "./Components/LandingPage/NavBar";

import PostJob from "./Components/Recruiter/PostJob";

import RecruiterLogin from "./Components/Recruiter/RecruiterLogin";

import RecruiterRegister from "./Components/Recruiter/RecruiterRegiter";

import AdminDasboard from "./Components/Admin/AdminDashboard";

import RecruiterPostedJobs from "./Components/Recruiter/PostedJobs"

import  Rprofile from "./Components/Recruiter/RecruiterProfile";

import SeekerApplyJobList from './Components/JobSeeker/SeekerApplyJobList'

import AdminUpdate from './Components/Admin/AdminProfile'
import SeekerUpdate from "./Components/JobSeeker/SeekerProfile";
import SeekerAppliedJob from './Components/JobSeeker/SeekerAppliedJob';
import RecruiterAppliedJob from './Components/Recruiter/RecruiterAppliedJob';
// import ListApplier from "./Components/Recruiter/ListofAppliers"
function App(){
return (
<>
<BrowserRouter>
<NavBar />
<Routes>
<Route path="/" element={<Home/>} />
<Route path="/findjob" element={<FindJob/>} />
<Route path="/about" element={<><About/></>}/>
<Route path="/contact" element={<><Contact/></>} />
{/* admin routes */}
<Route path="/admin" element={<AdminUpdate/>} />
<Route path="/admin/seekerlist" element={<><AdminSeeker/></>} />
<Route path="/admin/recruiterlist" element={<><AdminRecruiter/></>} />
<Route path="/admin/login" element={<><AdminLogin/></>}/>
<Route path="/admin/update" element={<AdminUpdate/>}/>
{/* Seeker */}
<Route path="/seeker/register" element={<><SeekerRegister/></>}/>
<Route path="/seeker/login" element={<><SeekerLogin/></>} />
<Route path="/seeker" element={<SeekerUpdate/>}/>
<Route path="/seeker/jobapply" element={<SeekerApplyJobList/>} />
<Route  path="/seeker/update" element={<SeekerUpdate/>}/>
<Route path="/seeker/appliedjob" element={<SeekerAppliedJob/>} />
{/* Recruiter */}
<Route  path="/recruiter/register" element={<RecruiterRegister/>}/>
<Route  path="/recruiter/login" element={<><RecruiterLogin/></>}/>
<Route  path="/recruiter" element={<Rprofile/>}/>
<Route path="/recruiter/postedjob" element={<RecruiterPostedJobs/>}/>
<Route  path="/recruiter/postjob" element={<><PostJob/></>}/>
<Route  path="/recruiter/appliedjob" element={<RecruiterAppliedJob/>}/>
<Route path="*" element={<h1>404 Page not found</h1>}/>
</Routes>
<Footer/>
</BrowserRouter>
</>
)
}
export default App;
```

Main.jsx:

```
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import App from './App.jsx'
import './index.css'
createRoot(document.getElementById('root')).render(
<StrictMode>
<App />
</StrictMode>,
)
```

App.css:

```
.form-control {
transition: border-color 0.3s ease, box-shadow 0.3s ease; /* Smooth transition for hover effect */
```

```
}
.form-control:hover {
border-color: #007bff; /* Change border color on hover */
box-shadow: 0 0 5px rgba(0, 123, 255, 0.5); /* Slight box shadow */
outline: none; /* Remove outline */
}
body{
overflow-x: hidden !important;
}
.navbar_main_row {
box-shadow: 0px 0px 10px #dfd8d8;
background: rgba(255, 255, 255, 0.8);
backdrop-filter: blur(10px); /* Adjust the blur value as needed */
min-height: 50px;
}
.nav_font{
font-size: 20px;
color:#212529 !important;
margin-left:30px;
}

.nav_font:hover {
color: #ff2642 !important;
}

.hero_section{
background: url("/h1_hero.jpg");
background-size: 100% 100%;
```

```
background-repeat: no-repeat;
background-attachment: fixed;
min-height: 550px !important;
}

.typewriter_hero{
font-size: 75px !important;
padding-top:70px;
color: #1f2b7b !important;
}

.category_section{
min-height: 600px;
}
.category_top_section{
color:#ff2642;
}
.category_heading{
color: #2b2b5f;
margin-top: 10px;
}
.category_box{
min-height: 400px;
border:2px solid red;
}
.hover-effect:hover {
background-color: #0056b3; /* Slightly darker blue */
color: white; /* White text color */
```

```
transition: background-color 0.3s ease, color 0.3s ease;
}

/* Optional hover effect for the secondary button */
.hover-effect.btn-secondary:hover {
background-color: #343a40; /* Slightly darker gray for secondary */
color: white;
}
/* Dropdown menu styles */
.custom-dropdown {
border-radius: 8px;
box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.15);
padding: 10px;
background-color: #f8f9fa; /* Light background */
}

/* Dropdown item hover */
.dropdown-item:hover {
background-color: #e2e6ea;
color: #0056b3;
border-radius: 5px;
padding-left: 15px;
transition: background-color 0.3s ease, padding-left 0.2s ease;
}

/* Dropdown arrow style */
.custom-btn::after {
margin-left: 10px;
```

```
}
.categoty_outer{
min-height: 550px;
background:#ffffff;
}
.child_row{
min-height: 600px;
}
.category_feature{
color: #fc82a2 ;
font-weight: 500;
}
.category_h1{
color: #283957;
}
.category_outer_box{
min-height: 600px;
}
.card_category{
min-height: 200px;
}
.child_div{
min-height: 200px;
padding: 10px 10px 10px ;
box-sizing: border-box;
background: #ffffff;
border-radius: 7px;
transition: all ease 1s;
```

```
border:2px solid #f7f7f9;
}
.child_div:hover{
transform: scale(1.03);
transition: all ease 1s;
box-shadow: 1px 1px 7px 5px #f7f7f9;
}
.category_icon{
font-size: 50px;
color: #00417e;
}
.category_content{
font-size: 20px;
color: #5f8eb3;}
.category_num{
font-size: 15px;
color: #fc82a2;
font-weight: 600;
}



.resume_image
{
/* background-image: url('/cv_bg.jpg'); */
background-color: #4c5595;
background-size: 100% 100%;
background-repeat: no-repeat;
```

```
min-height:350px ;
}
.blur
{
background: rgba(69,69,158,0.6);
min-height: 350px;
}

.upload_resume
{
height: 40px;
width: 200px;
border: 5px;
border-radius: 3px;
box-shadow: 0px 0px 2px white;
background: transparent;
color:white;
}
.upload_resume:hover{
color:white
}
.text_color
{
color:#fc82a2;
font-weight: 400;
font-size:20px;
}
```

```
.job_search
{

min-height: 250px;
margin-bottom: 30px;
padding-top: 1px;
border-radius: 7px;
transition: all 1s;
background-color: #26317f;
}
.job_search:hover
{
min-height: 285px;
margin-bottom: 30px;
box-shadow: 0px 0px 3px white;
transform:scale(1.01);
transition: all 1s;
}
.tex{
color:#7f74a3;
text-transform: capitalize;
text-align: justify;
}
/* hii*/
.jobs_outer{
min-height: 600px;
background: #ffffff;
}
```

```
.jobchild_row{
min-height: 600px;
}
.jobs_feature{
color: #fc82a2 ;
text-align: center;
}
.jobs_h1{
color: #28395a;
text-align: center;
}
.jobs_outer_box{
min-height: 600px;
}
.jobcard_category{
min-height: 110px;
box-sizing: border-box;
}
.jobchild_div{
min-height: 100px;
padding: 10px;
box-sizing: border-box;
border-radius: 3px;
transition: all ease .8s;
}
.jobchild_div:hover{
transform: scale(1.01);
box-shadow: 1px 1px 7px 5px #f7f7f9;
```

```
transition: all ease .8s;
}
.jobs_icon{
font-size: 60px;
color: #5f8eb3;
padding: 5px;
}
.jobs_title{
box-sizing: border-box;
}
.jobs_name{
float: left;
font-size: 15px;
font-weight: 300;
box-sizing: border-box;
}
.jobs_location{
float: left;
font-size: 15px;
font-weight: 300;
box-sizing: border-box;
}

.jobs_salary{
float: left;
font-size: 15px;
font-weight: 300;
box-sizing: border-box;
```

```
}
.jobs_time{
font-size: 15px;
font-weight: 400;
}

.jobs_btn {
border: 1px solid #8d94dd;
color: #8d94dd;
border-radius:20px;
height: 35px;
width:120px;
line-height: 30px;
}
.content_col{
min-height: 130px;
box-sizing: border-box;
}
.btn_job{
min-width: 150px;
min-height: 40px;
background: none;
color: rgb(71, 71, 180);
font-size: 20px;
border: 2px solid rgb(71, 71, 180);
border-radius: 50px;
}
@media (max-width: 576px) {
```

```
.jobs_icon img {
width: 80px; /* Adjust the image size for smaller screens */
}
.jobs_title {
font-size: 1.2rem; /* Adjust the title size */
}
.jobs_btn {
padding: 0.5rem 1rem; /* Adjust button padding */
}
}
.btn_job:hover{
color: white;
background: #5f8eb3
}
.apply_image
{
background-image: url('/how-applybg.png');
min-height: 600px;
}
.child_div_apply{
color: white;
}
.child_div_apply_ic{
color: white;
font-size: 60px;
}
.testimonial_outer{
min-height: 400px;
```

```
}
.main_testimonial{
min-height: 500px;
text-align: center;
}
/* .support_container{
background: rgb(230, 228, 228);
} */
.btn_support{
height: 40px;
width: 150px;
background: #db0249;
border: none;
color: white;
margin-left: 80px;
margin-bottom: 10px;
}
.btn_support:hover{
background: white;
color: #db0249;
border: 2px solid #db0249;
}
.supportImage_section{
background: url('/support-img.jpg');
min-height: 600px;
background-size: 100% 100%;
padding-right: 30px;
position: relative;
```

```
}
.supportcontent_section{
text-align: justify;
}

.supportImage_sectionContent{
min-height: 100px;
min-width:120px;
position: absolute;
bottom: 0px;
left:-40px;
background: #1f2c7b;
color: white;
text-align: center;
}
.support_sinceSection{
font-size: 35px;
font-family: bold;
margin-top: -20px;
}
.blog_container{
min-height: 600px;
}
.blog_row{
transform: translateY(-25px);
}
.img_content{
width: 100px;
```

```
height: 70px;
padding-top: 10px;
font-size: 20px;
background: #db0249;
color: white;
text-align: center;
transform: translateY(187px);
}
.blog_btn{
border: none;
background: none;
padding-left: 20px;
}
.blog_btn:hover{
color: white;
border: 1px solid #5f8eb3;
border-radius: 10px;
background: #5f8eb3;
padding-left: 5px;
}
.top_footer-container{
min-height: 450px;
background: #020a1d;
}
.topFooter_content{
color:#eeeeee;
font-weight: 400;
font-size: 16px;
```

```
text-align: justify;
}
.f{
text-align: right;
}
.footer_btn{
width: 50px;
background: #db0249;
padding-left: 15px;
padding-bottom: 5px;
font-size: 25px;
}
.foot_content{
color: white;
}
.p{
color:#d9d9d9;
font-size: 18px;
font-weight: 400;
}
/*Find a Job*/
.findjob_header{
background: url('/cv_bg.jpg');
background-size: 100% 100%;
background-repeat: no-repeat;
min-height: 350px;
text-align: center;
color: white;
```

```
}
/*-----------About-----------------*/
.about_header{
background: url('/how-applybg.png');
background-size: 100% 100%;
min-height: 200px;
}

.about_header_content{
color: white;
text-align: center;
}
.about_intro{
text-align: center;
}
.about_image{
min-height: 400px;
}
.team_header{
text-align: center;
}
.team_card{
min-height: 200px;
}
.teamchild_div{
min-height: 200px;
padding: 10px;
box-sizing: border-box;
```

```
border-radius: 10px;
box-shadow: 0px 0px 3px #5f8eb3;
transition: all ease 1s;
}
.teamchild_div:hover{
transform: scale(1.05);
transition: all ease 1s;
}
.team_name{
font-size: 25px;
font-weight: bold;
color: #5f8eb3;
}
.team_content{
font-size: 15px;
color: gray;
}
.team_review{
font-size: 15px;
}
/*contact*/
.contact_row{
min-height: 450px;
}
.contact_container{
background: linear-gradient(120deg,#536bdf,#db0249);
border-radius: 15px;
```

```
box-shadow: 0px 0px 5px black;
min-height: 450px;
}
.contact_img{
width: 100%;
height: 100%;
}
.contact_msg{
height: 200px;
}
.contact_heading{
color: rgb(150, 193, 200);
}
.contact_para{
color: rgb(202, 203, 201);
}
.contact_submit{
background: #0056b3;
color: white;
border: none;
}
/*-----------RegisterRecruiter-----------------*/
.recuiter_register{
min-height: 550px;
}
.register_img{
background: url('/recruiterRegister.jpg');
background-size: 100% 100%;
```

```
min-height: 600px;
}
.form_div_register{
min-height: 600px;
border: 1px solid #5f8eb3 ;
}
.recruiter_signUp_text{
color: #0056b3;
text-align: center;
padding: 20px;
}
.register_submit{
background: #0056b3;
color: white;
}
.error_msg{
color: red !important;
}
/* ------------Login------------- */

.form_div_login{
min-height: 600px;
border: 1px solid #5f8eb3 ;
padding-top: 80px;
background: linear-gradient(120deg,#db0249,#536bdf);
}
```

```
.seeker_list{
min-height: 200px;
border-radius: 10px;
}
.seeker_img{
border-radius: 50%;
}
.seeker_name{
text-align: center;
font-size: 40px;
}
.seeker_content{
text-align: center;
font-size: 15px;
}
.seeker_btn{
min-width: 180px;
min-height: 40px;
line-height: 40px;
border-radius: 20px;
background: #1f2c7b;
color: white;
text-align: center;
}
/* -----------Recruiter Post Job---------- */
.form_h2 {
text-align: center;
margin-bottom: 25px;
```

```
font-size: 20px;
color: #1f2b7b;
text-transform:uppercase;
}
.form_button {
width: 100%;
padding: 10px;
background: #fb246a;
border: none;
border-radius: 5px;
color: white;
font-size: 16px;
cursor: pointer;
transition: background 0.3s ease;
font-weight: 700;
}
.form_button:hover {
background: #074f88;
}
.err_span{
color:red !important;
}
.blog-container {
margin-top: 40px;
}
.blog-subheading {
color: #db0249;
font-weight: bold;
```

```
padding-top: 20px;
}
.blog-heading {
font-size: 32px;
font-weight: bold;
margin-bottom: 40px;
}
.blog-row {
display: flex;
justify-content: space-around;
flex-wrap: wrap;
}
.blog-card {
margin-bottom: 40px;
}
.blog-image-wrapper {
position: relative;
width: 100%;
overflow: hidden;
}
.date-badge1 {
position: absolute;
top: 156px;
left: 0px;
background-color: #db0249;
color: white;
text-align: center;
}
```

```
.blog-img {
width: 100%;
height: auto;
border-radius: 5px;
}
.blog-category {
padding-top: 10px;
color: gray;
font-style: italic;
}
.blog-title {
padding-top: 10px;
font-size: 20px;
}

.blog-btn {
background-color: #db0249;
color: white;
border: none;
padding: 10px 20px;
margin-top: 20px;
cursor: pointer;
font-size: 16px;
border-radius: 5px;
}
.blog-btn:hover {
background-color: #bf0242;
}
```

```
/* Responsive styling */
@media (max-width: 768px) {
.blog-card {
margin-bottom: 20px;
}
.blog-title {
font-size: 18px;
}
.blog-btn {
font-size: 14px;
}
}
@media (max-width: 576px) {
.blog-title {
font-size: 16px;
}
.blog-btn {
font-size: 12px;
padding: 8px 16px;
}
}
.postedjob_h{
color: #074f88;
font-weight: 700;
font-size: 18px;
}
.postedjob_p1{
```

```
color: #074f88;
font-size: 17px;
font-weight: 600;
}
.postedjob_p2{
/* color: #fa236a; */
font-size: 17px;
font-weight: 600;
}
.postedjob_card:hover{
transform: scale(1.02);
box-shadow: 0px 0px 5px rgba(179, 176, 176, 0.5);
transition: all 0.3s;
}
.postedjob_card{
transition: all 0.3s;
}
.form_button {
width: 100%;
padding: 10px;
background: #fb246a;
border: none;
border-radius: 5px;
color: white;
font-size: 16px;
cursor: pointer;
transition: background 0.3s ease;
font-weight: 700;
```

```css
}
/* profile css */
.form_label{
font-weight: 600;
color: #074f88;
}
.form_card{
border: 1.5px solid #074f88 !important;
}
.form_cardheader{
border-bottom: 1.5px solid #074f88 !important;
}
.form_button {
width: 100%;
padding: 10px;
background: #fb246a;
border: none;
border-radius: 5px;
color: white;
font-size: 16px;
cursor: pointer;
transition: background 0.3s ease;
font-weight: 700;
}
.form_button:hover {
background: #074f88;
}
```

```
.nav_bar_logo{
height: 80px;
width: 80px;
border-radius: 50%;
border: 2px solid black;
}
.btn_block_unblock{
background-color:#fb246a;
color: white;
text-align: center;
border-radius: 10px;
width: 100%;
}
.btn_block_unblock:hover{
background-color:#fb246a;
color: white;
text-align: center;
border-radius: 10px;
width: 100%;
}
```

AdminDashboard.jsx:

```
import AdminSeeker from "./AdminSeekerList"
import AdminRecruiter from "./AdminRecruiterList"
function AdminDasboard(){
return (
<>
<AdminSeeker/>
<AdminRecruiter/>
</>
)
}
export default AdminDasboard;
```

AdminLogin.jsx:

```
import { useForm } from 'react-hook-form';
import { yupResolver } from '@hookform/resolvers/yup';
import * as yup from 'yup';
import axios from 'axios';
import { useNavigate } from 'react-router-dom';
const schema = yup
.object()
.shape({
email: yup.string().required().email(),
password: yup.string().required()
})
function AdminLogin(){
const navigate=useNavigate()
const {register,handleSubmit, formState:{errors}}=useForm({
resolver:yupResolver(schema)
})
const handleData=async(data)=>{
const payLoad={
email:data.email,
password:data.password
}
const response=await   axios.post("http://localhost:9000/api/Admin-login",payLoad,{
headers:{
```

```
"Content-Type":"application/json"
}
})
if(response.data.code==200){

localStorage.setItem("data",JSON.stringify(response.data.data))
localStorage.setItem("userType",JSON.stringify('admin'))
alert("Login Successfull !...")
navigate('/admin')
}else{
alert("Invalid Email or Password !...")
}
}
return(<>
<div className="row">
<div className="col-sm-1"></div>
<div className="col-sm-10">
<div className="row mt-3">
<div className="col-sm-6 form_div_register recuiter_register">
<div className="register_img"></div>
</div>
<div className="col-sm-6 form_div_login">
<h2 className="recruiter_signUp_text">Admin <span style={{color:
"#db0249"}}>Login</span></h2>
<div className="p-4">
<form onSubmit={handleSubmit(handleData)}>
<div className="row mb-4 pt-3 ">
<input  className="form-control "
```

```
placeholder="Enter Your email"
type="text"
{...register('email')}
/>
{errors.email?.message && <span  className='error_msg'>{errors.email?.message}</span>}
</div>
<div className="row mb-4 pt-3">
<input  className="form-control "
placeholder="Enter Your password"
type="password"
{...register('password')}
/>
{errors.password?.message && <span  className='error_msg'>{errors.password?.message}</span>}
</div>
<div className="row mb-4 pt-3">
<input  className="register_submit form-control "
type="submit"
value="Login"
/>
</div>
</form>
</div>
</div>
</div>
</div>
<div className="col-sm-1"></div>
</div>
<br />
```

```
<br />
</>)
}
export default AdminLogin;
```

AdminProfile.jsx:

```
import React, { useEffect } from 'react';
import { useForm } from 'react-hook-form';    // Form Validation
import { yupResolver } from '@hookform/resolvers/yup';   // Form Validation
import * as yup from 'yup';   // Form Validation
import '../../app.css'
import axios from 'axios';
const schema = yup
.object()
.shape({
name: yup.string().required().min(2).max(30),
email: yup.string().required().email(),
contact: yup.string().required(),
password: yup.string().required(),
location: yup.string().required(),
img: yup.mixed().required()
})

const AdminUpdate = () => {
useEffect(()=>{
const userdetails= JSON.parse(localStorage.getItem("data"));
if(userdetails){
setValue("name",userdetails.name);
setValue("email",userdetails.email);
setValue("contact",userdetails.contact);
setValue("password",userdetails.password);
setValue("location",userdetails.location)
}
},[])

const { register, handleSubmit,setValue, formState: { errors } } = useForm({
resolver: yupResolver(schema),
});

const handleData = async (data) => {
if(!data || data.img.length==0){
alert("Please select an image");
return ;
}
const temData=JSON.parse(localStorage.getItem("data"));
const formData = new FormData();
formData.append('name', data.name);
formData.append('email', data.email);
formData.append('contact', data.contact);
formData.append('password', data.password);
formData.append('location', data.location);
formData.append('img', data.img[0]);
const response=await axios.put(`http://localhost:9000/api/admin-update/${temData._id}`,formData,{
headers:{
"Content-Type":"multipart/form-data"
}
})
```

```
if(response.data.code==200){
alert("Admin Updated Successfully");
}
}
return(
<div className="container my-5">
<div className="row justify-content-center">
<div className="col-md-8">
<div className="card shadow-sm form_card">
<div className="card-header form_cardheader text-center">
<h4 className="mb-0 form_h2">Admin Profile Update</h4>
</div>
<div className="card-body p-4">
<form onSubmit={handleSubmit(handleData)}>
<div className="row mb-3">
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1 form_label">Name :</label>
<input {...register('name')} type="text" className="form-control" placeholder="Enter Your Name" />
{errors.name?.message && <span className='err_span'>{errors.name?.message}</span>}
</div>
</div>
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1 form_label">Email :</label>
<input {...register('email')} type="text" readOnly={true} className="form-control" placeholder="Enter Your Email" />
{errors.email?.message && <span className='err_span'>{errors.email?.message}</span>}
```

```
</div>

</div>

</div>

<div className="row mb-3">

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Contact :</label>

<input {...register('contact')} type="number" className="form-control" placeholder="Enter Your Contact"/>

{errors.contact?.message && <span className='err_span'>{errors.contact?.message}</span>}

</div>

</div>

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Password :</label>

<input type='password' {...register('password')} className="form-control" placeholder='Enter Your Password'/>

{errors.password?.message && <span className='err_span'>{errors.password?.message}</span>}

</div>

</div>

</div>

<div className="row mb-3">

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Location :</label>

<input {...register('location')} type="text" className="form-control" placeholder="Enter Your Password"/>

{errors.location?.message && <span className='err_span'>{errors.location?.message}</span>}

</div>
```

```
</div>
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1 form_label">img :</label>
<input {...register('img')} type="file" className="form-control" accept='img/*' />
{errors.img?.message && <span className='err_span'>{errors.img?.message}</span>}
</div>
</div>
</div>
<div className="d-grid">
<input type="submit" value='UPDATE PROFILE' className="form_button"/>
</div>
</form>
</div>
</div>
</div>
</div>
</div>
)
}
export default AdminUpdate
```

AdminRecruiter.jsx;

```
import { useEffect ,useState} from "react"
import axios from "axios"
function AdminRecruiter(){
const [recruiter, setrecruiter] = useState([])
```

```
useEffect(()=>{
fetchData()
},[])
const fetchData=async()=>{
const response=await axios.get("http://localhost:9000/api//admin-recruiterlist",{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
setrecruiter(response.data.data)
}
}
const handleBlock=async(item)=>{
const payload={
status:item.isBlock ? false : true
}
const response=await axios.put(`http://localhost:9000/api/admin-recruiterblock/${item._id}`,payload,{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
fetchData()
alert("Status Updaetd Successfully..")
}
}
return (
<>
<div className="container">
<div className="row justify-content-center">
{
recruiter.map((item, index) => (
<div className="col-12 col-md-6 col-lg-4 seeker_list my-2 mx-2" key={index}>
<div className="card p-2 h-100">
<div className="text-center">
<img className="seeker_img img-fluid rounded-circle" src={`http://localhost:9000/upload/${item?.logo}`} alt={item.name} style={{ maxWidth: '100px', height: '100px' }} />
</div>
<div className="seeker_name text-center mt-2 font-weight-bold">{item.name}</div>
<table className="table table-borderless mt-3">
<tbody>
<tr>
<td className="text-right font-weight-bold">Email:</td>
<td className="text-left">{item.email}</td>
</tr>
<tr>
<td className="text-right font-weight-bold">Contact:</td>
<td className="text-left">{item.contact}</td>
</tr>
<tr>
<td className="text-right font-weight-bold">Location:</td>
<td className="text-left">{item.location}</td>
</tr>
<tr>
<div className="row">
<td className="text-right font-weight-bold">
<center>

<div className="btn  btn_block_unblock" onClick={()=>handleBlock(item)} >{item.isBlock ? "UnBlock" :"Block"}</div>

</center>
</td>
</div>
</tr>
</tbody>
</table>
<div className="text-center mt-auto">
{/* <button className=" seeker_btn ">{item.blockBtn}</button> */}
</div>
</div>
</div>
))
}
</div>
</div>
</>
)
}
export default AdminRecruiter
```

AdminSeeker.jsx:

```
import { useEffect ,useState} from "react"
import axios from "axios"
```

```
function AdminSeeker(){
const [seeker, setSeeker] = useState([])
useEffect(()=>{
fetchData()
},[])
const fetchData=async()=>{
const response=await axios.get("http://localhost:9000/api/admin-seekerlist",{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
setSeeker(response.data.data)
}
}
const handleBlock=async(item)=>{
const payload={
status:item.isBlock ? false : true
}
const response=await axios.put(`http://localhost:9000/api/admin-seekerblock/${item._id}`,payload,{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
fetchData()
alert("Status Updaetd Successfully..")
}
```

```
}

return (

<>

<div className="container">

<div className="row justify-content-center">

{

seeker.map((item, index) =>

(

<div className="col-12 col-md-6 col-lg-4 seeker_list my-2 mx-2" key={index}>

<div className="card border border-none p-2 h-100">

<div className="text-center">

<img className="seeker_img img-fluid rounded-circle"
src={`http://localhost:9000/upload/${item.img}`} alt={item.name} style={{ maxWidth: '100px', height:
'100px' }} />

</div>

<div className="seeker_name text-center mt-2 font-weight-bold">{item.name}</div>

<table className="table table-borderless mt-3">

<tbody>

<tr>

<td className="text-right font-weight-bold">Email:</td>

<td className="text-left">{item.email}</td>

</tr>

<tr>

<td className="text-right font-weight-bold">Contact:</td>

<td className="text-left">{item.contact}</td>

</tr>

<tr>

<td className="text-right font-weight-bold">Location:</td>

<td className="text-left">{item.location}</td>
</tr>

<tr>

<td className="text-right font-weight-bold">Qualification:</td>

<td className="text-left">{item.qualification}</td>

</tr>

<tr>

<td className="text-right font-weight-bold">Preference:</td>

<td className="text-left">{item.preference}</td>

</tr>

<tr>

<div className="row">

<td className="text-right font-weight-bold">

<center>


<div className="btn  btn_block_unblock" onClick={()=>handleBlock(item)} >{item.isBlock ?
"UnBlock" :"Block"}</div>

</center>

</td>

</div>

</tr>

</tbody>

</table>

<div className="text-center mt-auto">

{/* <button className=" seeker_btn ">{item.blockBtn}</button> */}

</div>

</div>

</div>

))
```

```
}
</div>
</div>
</>
)
}
export default AdminSeeker
```

JobSeeker:

SeekerAppliedJob.jsx:

```
import axios from "axios";
import { useState } from "react";
import { useEffect } from "react";

function SeekerAppliedJob(){
const [dataId,setData]=useState()
const [jobData,setJobData]=useState([])
useEffect(()=>{
const temData=JSON.parse(localStorage.getItem("data"));
setData(temData)
getData()

},[])



const getData=async()=>{
const temData=JSON.parse(localStorage.getItem("data"));
const payload={
```

```
userId:temData._id

}

const response= await axios.post("http://localhost:9000/api/seeker-applied",payload,{

headers:{

"Content-Type":"application/json"

}

})

if(response.data.code==200){

setJobData(response.data.data)

}


}


return (<>

<div className="container my-3">


{jobData.map((el) => {

// console.log(el,"################################################")

return (

<div className="card p-3 mb-3 postedjob_card">

<div className="row d-flex justify-content-center align-items-center">

{/* Logo Column */}

<div className="col-md-3 d-flex justify-content-center align-items-center">

<img src={`http://localhost:9000/upload/${el.logo}`} alt="Company Logo" className="img-fluid bg-dark" style={{ maxHeight: '100px' }} />

</div>


{/* Company Name, Job Title, and Job Type Column */}
<div className="col-md-3 d-flex justify-content-start flex-column my-3">

<h5 className='postedjob_h'> {el.name}</h5>

<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobTitle}</span></div>

<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobType}</span></div>

</div>


{/* Job Category, Location, and Salary Column */}

<div className="col-md-3">

<div className='postedjob_p2 mb-2'>Category: <span className='postedjob_p1'>
{el.category}</span></div>

<div className='postedjob_p2 mb-2'>Location: <span className='postedjob_p1'>
{el.jobLocation}</span></div>

<div className='postedjob_p2 mb-2'>Salary: <span className='postedjob_p1'>
{el.salary}</span></div>

</div>


{/* Apply Date and Vacancies Column */}

<div className="col-md-3">

<div className='postedjob_p2 mb-2'>Vacancies: <span className='postedjob_p1'>
{el.vacancies}</span></div>

<div className='postedjob_p2 mb-2'>Apply By: <span className='postedjob_p1'>
{el.applyDate}</span></div>

</div>

</div>

</div>

)

})}

</div>

</>)

}
```

```
export default SeekerAppliedJob;
```

SeekerApplyJobList.jsx:

```
import axios from "axios";
import { useState } from "react";
import { useEffect } from "react";
function SeekerApplyJobList(){
const [dataId,setData]=useState()
const [jobData,setJobData]=useState([])
useEffect(()=>{
const temData=JSON.parse(localStorage.getItem("data"));
setData(temData)
getData()
},[])
const getData=async()=>{
const response= await axios.get("http://localhost:9000/api/seeker-joblist",{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
setJobData(response.data.data)
}

}
const handleApply=async(element)=>{
const companyId=element.companyId;
const jobId=element._id;
const userId=dataId._id;
const payload={
companyId:companyId,
userId:userId,
jobId:jobId
}
const response=await axios.post("http://localhost:9000/api/seeker-apply",payload,{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
alert("Job Applied Successfully")
}
else if(response.data.code==301){
alert("You have already applied for this role");
}
}
return (<>
<div className="container my-3">
{jobData.map((el) => {
// console.log(el,"#############################################")
return (
<div className="card p-3 mb-3 postedjob_card">
<div className="row d-flex justify-content-center align-items-center">
{/* Logo Column */}
<div className="col-md-3 d-flex justify-content-center align-items-center">
<img src={`http://localhost:9000/upload/${el.logo}`} alt="Company Logo" className="img-fluid bg-dark" style={{ maxHeight: '100px' }} />
```

```
</div>


{/* Company Name, Job Title, and Job Type Column */}

<div className="col-md-3 d-flex justify-content-start flex-column my-3">

<h5 className='postedjob_h'> {el.name}</h5>

<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobTitle}</span></div>

<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobType}</span></div>

</div>


{/* Job Category, Location, and Salary Column */}

<div className="col-md-3">

<div className='postedjob_p2 mb-2'>Category: <span className='postedjob_p1'>
{el.category}</span></div>

<div className='postedjob_p2 mb-2'>Location: <span className='postedjob_p1'>
{el.jobLocation}</span></div>

<div className='postedjob_p2 mb-2'>Salary: <span className='postedjob_p1'>
{el.salary}</span></div>

</div>


{/* Apply Date and Vacancies Column */}

<div className="col-md-3">

<div className='postedjob_p2 mb-2'>Vacancies: <span className='postedjob_p1'>
{el.vacancies}</span></div>

<div className='postedjob_p2 mb-2'>Apply By: <span className='postedjob_p1'>
{el.applyDate}</span></div>

<input className='form_button mt-3' onClick={()=>handleApply(el)} type='submit' value='APPLY
NOW' style={{width:'150px', fontSize:"0.8em"}}/>

</div>

</div>

</div>
```

```
)
})}
</div>
</>)
}
export default SeekerApplyJobList;
```

SeekerLogin.jsx:

```
import { useForm } from 'react-hook-form';
import { yupResolver } from '@hookform/resolvers/yup';
import * as yup from 'yup';
import axios from 'axios';
import { useNavigate } from 'react-router-dom';
const schema = yup
.object()
.shape({
email: yup.string().required().email(),
password: yup.string().required()
})
function SeekerLogin(){
const navigate=useNavigate()
const {register,handleSubmit, formState:{errors}}=useForm({
resolver:yupResolver(schema)
})
const handleData=async(data)=>{
const payLoad={
email:data.email,
password:data.password
}
```

```
const response=await   axios.post("http://localhost:9000/api/Seeker-login",payLoad,{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
localStorage.setItem("data",JSON.stringify(response.data.data))
localStorage.setItem("userType",JSON.stringify('seeker'))
alert("Login Successfull !...")
navigate('/seeker')
}else if(response.data.code==203){
alert("Your Account is Blocked contact to Admin")
}
else if (response.data.code==302){
alert("Invaild Email or password")
}
}
return(<>
<div className="row">
<div className="col-sm-1"></div>
<div className="col-sm-10">
<div className="row mt-3">
<div className="col-sm-6 form_div_register recuiter_register">
<div className="register_img"></div>
</div>
<div className="col-sm-6 form_div_login">
<h2 className="recruiter_signUp_text">Seeker <span style={{color:
"#db0249"}}>Login</span></h2>
```

```
<div className="p-4">
<form onSubmit={handleSubmit(handleData)}>
<div className="row mb-4 pt-3 ">
<input  className="form-control "
placeholder="Enter Your email"
type="text"
{...register('email')}
/>
{errors.email?.message && <span  className='error_msg'>{errors.email?.message}</span>}
</div>
<div className="row mb-4 pt-3">
<input  className="form-control "
placeholder="Enter Your password"
type="password"
{...register('password')}
/>
{errors.password?.message && <span  className='error_msg'>{errors.password?.message}</span>}
</div>

<div className="row mb-4 pt-3">
<input  className="register_submit form-control "
type="submit"
value="Login"
/>
</div>
</form>
</div>
</div>
```

```
</div>
</div>
<div className="col-sm-1"></div>
</div>
<br />
<br />
</>)
}
export default SeekerLogin;
```

SeekerProfile.jsx:

```
import React, { useEffect } from 'react';
import { useForm } from 'react-hook-form';
import * as Yup from 'yup';
import { yupResolver } from '@hookform/resolvers/yup';
import axios from 'axios';

// Validation schema using Yup
const schema = Yup.object().shape({
name: Yup.string().required('Name is required'),
email: Yup.string().email('Invalid email').required('Email is required'),
contact: Yup.string().matches(/^[0-9]{10}$/, 'Contact must be 10 digits').required('Contact is required'),
location: Yup.string().required('Location is required'),
password: Yup.string().min(6, 'Password must be at least 6 characters').required('Password is required'),
qualification: Yup.string().required('Qualification is required'),
preference: Yup.string().required('Preference is required'),
img: Yup.mixed().required('Profile image is required'),
resume: Yup.mixed().required('Resume is required'),
});
```

```
const SeekerUpdate = () => {
const { register, handleSubmit, setValue, formState: { errors } } = useForm({
resolver: yupResolver(schema),
});
useEffect(() => {
const userdetails = JSON.parse(localStorage.getItem("data"));
if (userdetails) {
setValue("name", userdetails.name);
setValue("email", userdetails.email);
setValue("contact", userdetails.contact);
setValue("location", userdetails.location);
setValue("password", userdetails.password);
setValue("qualification", userdetails.qualification);
setValue("preference", userdetails.preference);
}
}, [setValue]);
const handleData = async (data) => {
const formData = new FormData();
formData.append('name', data.name);
formData.append('email', data.email);
formData.append('contact', data.contact);
formData.append('location', data.location);
formData.append('password', data.password);
formData.append('qualification', data.qualification);
formData.append('preference', data.preference);
formData.append('img', data.img[0]);
formData.append('resume', data.resume[0]);
```

```
if (!data.img || !data.resume) {
alert("Please select both image and resume");
return;
}
const temData = JSON.parse(localStorage.getItem("data"));
try {
const response = await axios.put(`http://localhost:9000/api/seeker-update/${temData._id}`, formData, {
headers: {
"Content-Type": "multipart/form-data",
},
});
if (response.data.code === 200) {
alert("Profile updated successfully");
}
} catch (error) {
console.error("Error updating profile", error);
}
};
return (
<div className="container my-5">
<div className="row justify-content-center">
<div className="col-md-8">
<div className="card shadow-sm form_card">
<div className="card-header form_cardheader text-center">
<h4 className="mb-0 form_h2">Seeker Profile Update</h4>
</div>
<div className="card-body p-4">
```

```
<form onSubmit={handleSubmit(handleData)}>

<div className="row mb-3">

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Name :</label>

<input {...register('name')} type="text" className="form-control" placeholder="Enter Your Name" />

{errors.name?.message && <span className='err_span'>{errors.name?.message}</span>}

</div>

</div>

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Email :</label>

<input {...register('email')} type="text" className="form-control" placeholder="Enter Your Email" />

{errors.email?.message && <span className='err_span'>{errors.email?.message}</span>}

</div>

</div>

</div>

<div className="row mb-3">

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Contact :</label>

<input {...register('contact')} type="number" className="form-control" placeholder="Enter Your
Contact" />

{errors.contact?.message && <span className='err_span'>{errors.contact?.message}</span>}

</div>

</div>

<div className="col-md-6">

<div className="form-group">
```

```
<label className="mb-2 ms-1 mt-1 form_label">Password :</label>
<input {...register('password')} type='password' className="form-control" placeholder='Enter Your Password' />
{errors.password?.message && <span className='err_span'>{errors.password?.message}</span>}
</div>
</div>
</div>
<div className="row mb-3">
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1 form_label">Location :</label>
<input {...register('location')} type="text" className="form-control" placeholder="Enter Your Location" />
{errors.location?.message && <span className='err_span'>{errors.location?.message}</span>}
</div>
</div>
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1 form_label">Qualification :</label>
<input {...register('qualification')} type="text" className="form-control" placeholder="Enter Your Qualification" />
{errors.qualification?.message && <span className='err_span'>{errors.qualification?.message}</span>}
</div>
</div>
</div>

<div className="row mb-3">
<div className="col-md-6">
```

```
<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Preference :</label>

<input {...register('preference')} type="text" className="form-control" placeholder="Enter Your
Preference" />

{errors.preference?.message && <span className='err_span'>{errors.preference?.message}</span>}

</div>

</div>

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Profile Image :</label>

<input {...register('img')} type="file" className="form-control" accept="image/*" />

{errors.img?.message && <span className='err_span'>{errors.img?.message}</span>}

</div>

</div>

</div>

<div className="row mb-3">

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Resume :</label>

<input {...register('resume')} type="file" className="form-control" accept=".pdf,.doc,.docx" />

{errors.resume?.message && <span className='err_span'>{errors.resume?.message}</span>}

</div>

</div>

</div>


<div className="d-grid">

<input type="submit" value='UPDATE PROFILE' className="form_button" />

</div>
```

```
</form>
</div>
</div>
</div>
</div>
</div>
);
}
export default SeekerUpdate;
```

SeekerRegister.jsx:

```
import { yupResolver } from '@hookform/resolvers/yup';
import axios from 'axios';
import { useForm } from 'react-hook-form';
import * as yup from 'yup';
const schema = yup
.object()
.shape({
name: yup.string().required().min(2).max(20),
img: yup.mixed().required(),
email: yup.string().required().email(),
contact: yup.string().required(),
password: yup.string().required(),
qualification: yup.string().required(),
location: yup.string().required(),
preference: yup.string().required(),
resume: yup.mixed().required()
})
function SeekerRegister() {
```

```
const { register, handleSubmit, formState: { errors } } = useForm({
resolver: yupResolver(schema),
});
const handleData = async (data) => {
const formData = new FormData();
formData.append("name", data.name);
formData.append("img", data.img[0]);
formData.append("email", data.email)
formData.append("contact", data.contact)
formData.append("password", data.password)
formData.append("qualification", data.qualification)
formData.append("location", data.location)
formData.append("preference", data.jobPreference)
formData.append("resume", data.resume[0])
await axios.post("http://localhost:9000/api/seeker-register", formData, {
headers: {
"Content-Type": "multipart/form-data"
}
})
alert("Registation SuccessFull !.")

}

return (<>
<div className="row">
<div className="col-sm-1"></div>
<div className="col-sm-10">
<div className="row">
```

```
<div className="col-sm-6 form_div_register recuiter_register">

<div className="register_img"></div>

</div>

<div className="col-sm-6 form_div_register ">

<h2 className="recruiter_signUp_text">Seeker <span style={{ color: "#db0249"
}}>SingUp</span></h2>

<div className="p-4">

<form onSubmit={handleSubmit(handleData)}>

<div className="row mb-4">

<input className="form-control "

placeholder="Enter Your Name"

type="text"

{...register('name')}

/>

{errors.name?.message && <span className='error_msg'>{errors.name?.message}</span>}

</div>

<div className="row mb-4">

<input className="form-control "

type="file"

{...register('img')}

/>

{errors.img?.message && <span className='error_msg'>{errors.img?.message}</span>}

</div>

<div className="row mb-4 ">

<input className="form-control "

placeholder="Enter Your email"

type="text"

{...register('email')}
```

```
/>
{errors.email?.message && <span className='error_msg'>{errors.email?.message}</span>}
</div>
<div className="row mb-4">
<input className="form-control "
placeholder="Enter Your contact"
type="number"
{...register('contact')} />
{errors.contact?.message && <span className='error_msg'>{errors.contact?.message}</span>}
</div>
<div className="row mb-4">
<input className="form-control "
placeholder="Enter Your password"
type="password"
{...register('password')}
/>
{errors.password?.message && <span className='error_msg'>{errors.password?.message}</span>}
</div>
<div className="row mb-4">
<input className="form-control "
placeholder="Enter Your Qualification"
type="text"
{...register('qualification')}
/>
{errors.qualification?.message && <span
className='error_msg'>{errors.qualification?.message}</span>}
</div>
<div className="row mb-4">
```

```
<input className="form-control "
placeholder="Enter Your location"
type="text"
{...register('location')} />
{errors.location?.message && <span className='error_msg'>{errors.location?.message}</span>}
</div>
<div className="row mb-4">
<input className="form-control "
placeholder="Enter Your Job Preference"
type="text"
{...register('preference')}
/>
{errors.preference?.message && <span className='error_msg'>{errors.preference?.message}</span>}
</div>
<div className="row mb-4">
<input
className="form-control"
type="file"
accept="application/pdf"
{...register('resume')}
/>
{errors.resume?.message && <span className='error_msg'>{errors.resume?.message}</span>}
</div>
<div className="row mb-4">
<input className="register_submit form-control "
type="submit"
value="SignUp"
/>
```

```
</div>
</form>
</div>
</div>
</div>
</div>
<div className="col-sm-1"></div>
</div>
<br />
<br />
</>)
}
export default SeekerRegister;
```

LandingPage

About.jsx:

```
function About(){

const about_arr=[
{
img:"job-list1.png",
name: 'Arlene McCoy',
title: 'Frontend Devloper',
location: 'New Yark, US',
review: '$3500 - $4000',

},
{
```

```
img:"job-list2.png",
name: 'Floyed Milws',
title: 'UI/UX designer',
location: 'Athens, Greece',
review: '$3500 - $4000',

},
{
img:"job-list1.png",
name: 'Devon Lane',
title: 'Frontend Devloper',
location: 'Chicago, US',
review: '$3500 - $4000',

},
{
img:"job-list2.png",
name: 'Jerome Bell',
title: 'Backend Devloper',
location: 'Mumbai, India',
review: '$3500 - $4000',
}
}
return (
<>
<div className="container-fluid about_header">
<div className="row about_header_content text-center pt-5">
<h1>About Us</h1>
```

```
<h5>Get the latest jobs, updates, and tips</h5>
</div>
</div>
<div className="row">
<div className="col-0 col-sm-2"></div>
<div className="col-12 col-sm-8">
<div className="row about_intro pt-4 text-center text-md-start">
<span>OUR COMPANY</span>
<h2><b>About Our Company</b></h2>
<p>
TalentHunt Job Portal is a specialized platform designed to connect job seekers with potential employers.
It serves as a bridge between professionals looking for new opportunities and companies searching for skilled talent.
TalentHunt streamlines the job search and recruitment process, making it easier for both sides to find the perfect match efficiently.
</p>
<br />
<div className="about_image text-center">
<img src="about.jpg" alt="About Company" className="img-fluid" />
</div>
</div>
<div className="row pt-3">
<h2>What we can do?</h2>
<p>
Lorem ipsum dolor sit amet consectetur adipisicing elit. Id voluptas vel voluptatem nobis, quod commodi velit,
assumenda laudantium sint quos sapiente, suscipit deleniti laboriosam facilis iste earum alias dicta quo repellat accusamus nemo ipsum!
</p>
```

```
<p>
Doloremque quo sapiente accusantium, aut tempora, incidunt nulla iusto magni ex maiores qui nemo consectetur voluptate.
</p>
<p>
Lorem ipsum dolor sit amet consectetur adipisicing elit. Id voluptas vel voluptatem nobis, quod commodi velit,
assumenda laudantium sint quos sapiente, suscipit deleniti laboriosam facilis iste earum alias dicta quo repellat accusamus nemo ipsum!
</p>
</div>
</div>
<div className="col-0 col-sm-2"></div>
</div>
<br />
<div className="row">
<div className="col-0 col-sm-2"></div>
<div className="col-12 col-sm-8 ps-3 team_header">
<span>OUR COMPANY</span>
<h2>Meet Our Team</h2>
<p>
Lorem ipsum dolor sit amet consectetur adipisicing elit. Id voluptas vel voluptatem nobis, quod commodi velit,
assumenda laudantium sint quos sapiente, suscipit deleniti laboriosam facilis iste earum alias dicta quo repellat accusamus nemo ipsum!
</p>
<div className="row">
{about_arr.map((card) => {
return (
<div className="col-12 col-md-5 ms-md-2 mt-5 team_card">
<div className="teamchild_div text-center">
<div className="team_icon">
<img src={card.img} alt="Team Member" className="img-fluid" />
</div>
<div className="team_name">{card.name}</div>
<div className="team_content">{card.title}</div>
<div className="team_content">{card.location}</div>
<div className="team_review">{card.review}</div>
</div>
</div>
);
})}
</div>
</div>
<div className="col-0 col-sm-2"></div>
</div>

<div className="row">
<div className="col-0 col-sm-2"></div>
<div className="col-12 col-sm-8 ps-3 pt-5">
<h2>News and Blog</h2>
<p>Get the latest news, updates, and blogs</p>
<div className="row blog_row">
<div className="col-12 col-md-6">
<div className="img_content">24 <br /> Now</div>
<img src="home-blog1.jpg" alt="Blog" className="img-fluid" />
<p className="pt-3">| Properties</p>
<h2>Footprints in Time is the perfect House in Kurashiki</h2>
```

```
<button className="blog_btn mt-3" type="button">READ MORE</button>
</div>
<div className="col-12 col-md-6">
<div className="img_content">24 <br /> Now</div>
<img src="home-blog2.jpg" alt="Blog" className="img-fluid" />
<p className="pt-3">| Properties</p>
<h2>Footprints in Time is the perfect House in Kurashiki</h2>
<button className="blog_btn mt-3" type="button">READ MORE</button>
</div>
</div>
</div>
<div className="col-0 col-sm-2"></div>
</div>
</>
)
}
export default About
```

Apply.jsx:

```
import { GoArchive } from "react-icons/go";
function Apply() {
return (
<>
<div className="row apply_image text-center mt-1 pt-2">
<div className="text_color mt-5">Apply Process</div>
<div className="h1 text-light">How It Works</div>
<div className="row ">
{/* Use responsive column classes */}
<div className="col-12 col-sm-2"></div>
```

```
<div className="col-12 col-sm-8">
<div className="row">
{/* Adjust each step's column size for better responsiveness */}
<div className="col-12 col-sm-4 mb-4">
<div className="box_border">
<div className="job_search text-center ">
<h1 className="child_div_apply_ic mt-3"><GoArchive /></h1>
<div className="h4 child_div_apply mt-3">1. Search A Job</div>
<div className="tex px-4  mt-3 pb-4">Text editing and text formatting, etc., fonts, and sizes. It was developed to integrate the features included.</div>
</div>
</div>
</div>
<div className="col-12 col-sm-4 mb-4">
<div className="box_border">
<div className="job_search text-center ">
<h1 className="child_div_apply_ic mt-3"><GoArchive /></h1>
<div className="h4 child_div_apply mt-3">2. Search A Job</div>
<div className="tex px-4  mt-3 pb-4">Text editing and text formatting, etc., fonts, and sizes. It was developed to integrate the features included.</div>
</div>
</div>
</div>

<div className="col-12 col-sm-4 mb-4">
<div className="box_border">
<div className="job_search text-center ">
<h1 className="child_div_apply_ic mt-3"><GoArchive /></h1>
<div className="h4 child_div_apply mt-3">3. Search A Job</div>
```

```
<div className="tex px-4  mt-3 pb-4">Text editing and text formatting, etc., fonts, and sizes. It was developed to integrate the features included.</div>
</div>
</div>
</div>
</div>
</div>
<div className="col-12 col-sm-2"></div>
</div>
</div>
</>
)
}
export default Apply;
```

BlogSection.jsx:

```
function BlogSection(){
return (
<>
<div className="container blog-container">
<div className="row text-center">
<p className="blog-subheading">OUR LATEST BLOG</p>
<h1 className="blog-heading">Our recent news</h1>
</div>
<div className="row blog-row">
<div className="col-md-2 col-sm-12 blog-card"></div>
<div className="col-md-4 col-sm-12 blog-card">
<div className="blog-image-wrapper">
<div className="date-badge1 w-25">
```

```
<p className="mt-2">24</p>

<p>Now</p>

</div>

<img src="/home-blog1.jpg" alt="Blog 1" className="blog-img" />

</div>

<p className="blog-category ms-5 mt-2">| Properties</p>

<h3 className="blog-title ms-5">

Footprints in Time is perfect House in Kurashiki

</h3>

<button className="blog-btn">READ MORE</button>

</div>

<div className="col-md-4 col-sm-12 blog-card">

<div className="blog-image-wrapper">

<div className="date-badge1 w-25">

<p className="mt-2">24</p>

<p>Now</p>

</div>

<img src="home-blog2.jpg" alt="Blog 2" className="blog-img" />

</div>

<p className="blog-category ms-5 mt-2">| Properties</p>

<h2 className="blog-title ms-5">

Footprints in Time is perfect House in Kurashiki

</h2>

<button className="blog-btn">READ MORE</button>

</div>

<div className="col-md-2 col-sm-12 blog-card"></div>

</div>

</div>
```

```
</>}
}
export default BlogSection
```

Category.jsx:

```
import { FaUserSecret } from "react-icons/fa";
import { TbDeviceDesktopStar } from "react-icons/tb";
import { BsFillFileEarmarkBarGraphFill } from "react-icons/bs";
import { TbDeviceMobileCog } from "react-icons/tb";
import { GrUserWorker } from "react-icons/gr";
import { GiCircuitry } from "react-icons/gi";
import { BsFillBuildingsFill } from "react-icons/bs";
import { MdOutlineLibraryBooks } from "react-icons/md";
function Category(){
const category_arr=[
{
img:<FaUserSecret/>,
title:"Design & Creative",
count:"(653)"
},
{
img:<TbDeviceDesktopStar/>,
title:"Design & Devlopment",
count:"(658)"
},
{
img:<BsFillFileEarmarkBarGraphFill/>,
title:"Sales & Marketing",
count:"(658)"
},
{
img:<TbDeviceMobileCog/>,
title:"Mobile Application",
count:"(658)"
},
{
img:<GrUserWorker/>,
title:"Construction",
count:"(658)"
},
{
img:<GiCircuitry/>,
title:"Information Technology",
count:"(658)"
},
{
img:<BsFillBuildingsFill/>,
title:"Real Estate",
count:"(658)"
},
{
img:<MdOutlineLibraryBooks/>,
title:"Content Writing",
count:"(658)"
}
]
return(<>
<div className="row categoty_outer ps-5 pe-5">
<div className="col-12 ps-5 pe-5">
<div className="row child_row ps-5 pe-5 pt-5">

<div className="category_feature mb-3 text-center">FEATURED TOUR PACKAGES</div>
<div className="h1 category_h1 text-center">Browse Top Categories</div>
<div className="row category_outer_box pt-5 text-center">
{category_arr.map((Category) => {
return (
<div key={Category.title} className="col-sm-3 text-center card_category">
<div className="child_div">
<div className="category_icon mt-2">{Category.img}</div>
<div className="category_content mt-2">{Category.title}</div>
<div className="category_num mt-2">{Category.count}</div>
</div>
</div>
);
})}
</div>

</div>
</div>
</div>
</>)
}
export default Category;
```

Contact.jsx:

```
import '../../App.css'
```

```
function Contact(){
return (
<>
<div className="row contact_row mb-3">
{/* Add padding or margin to center the content and make it responsive */}
<div className="col-0 col-sm-1"></div>
<div className="col-12 col-sm-10">
<div className="row contact_container mt-5 pt-2">
<div className="col-12 col-md-6 text-center text-md-start">
<img className="contact_img img-fluid" src="contact.webp" alt="Contact" />
</div>
<div className="col-12 col-md-6 mt-4 pt-5 px-5 mt-md-0">
<h2 className="contact_heading">Get in touch</h2>
<p className="contact_para">
The right move at the right time saves your investment. Live the dream of expanding your business.
</p>
<div className="contact_form p-5 m-0 m-md-4">
{/* Responsive Form Fields */}
<div className="mb-3">
<input type="text" className="form-control" placeholder="Enter Your Name" />
</div>
<div className="mb-3">
<input type="text" className="form-control" placeholder="Enter Your Company (optional)" />
</div>
<div className="mb-3">
<input type="email" className="form-control" placeholder="Enter Your Email" />
</div>
<div className="mb-3">
<input type="number" className="form-control" placeholder="Enter Your Number" />
</div>
<div className="mb-3">
<textarea type="text" className="form-control" placeholder="Message"></textarea>
</div>
<div className="mb-3">
<input className="contact_submit form-control btn btn-primary" type="submit" value="Submit" />
</div>
</div>
</div>
</div>
</div>
<div className="col-0 col-sm-1"></div>
</div>

</>
)
}
export default Contact
```

FindJob.jsx:

```
function FindJob(){
const jobs_arr=[
{
img:"job-list1.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list2.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list3.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list4.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list1.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list2.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
}
]
return (
<>
<div className="container-fluid findjob_header ">
<div className="row blur p-5">
<div className="col-sm-2"></div>
<div className="col-sm-8 p-3">
<div className=" p-5">
```

```
<div className="h2 mb-4">Many jobs Available Here</div>
<div className="p mb-3">Lorem, ipsum dolor sit amet consectetur adipisicing elit. Temporibus nihil rerum sunt nemo at natus voluptates voluptatibus voluptatem similique deleniti?</div>
<h4>Select your Prefrenced Job</h4>
</div>
</div>
<div className="col-sm-2"></div>
</div>
</div>
<div className="row">
<div className="col-sm-2"></div>
<div className="col-sm-8 px-5">
<div className="row job_outer_box pt-3">
{
jobs_arr.map((item)=>{
return(<>
<div className="jobcard_category mb-0 pb-3">
<div className="jobchild_div ps-3">
<div className="row">
{/* Image Section */}
<div className="col-12 col-sm-2 content_col text-center">
<div className="jobs_icon">
<img src={item.img} alt={item.jobTitle} className="img-fluid" />
</div>
</div>
{/* Job Details Section */}
<div className="col-12 col-sm-8 content_col">
<div className="h3 jobs_title mt-3">
{item.jobTitle}
</div>
<div className="jobs_name mt-2 me-5">{item.companyName}</div>
<div className="jobs_location mt-2 me-5">{item.location}</div>
<div className="jobs_salary mt-2 me-5">{item.salary}</div>
</div>
{/* Button and Time Section */}
<div className="col-12 col-sm-2 content_col text-center">
<div className="btn btn-outline-primary rounded-pill mt-4">
{item.jobBtn}
</div>
<div className="jobs_time mt-4">
{item.postedTime}
</div>
</div>
</div>
</div>
</div>
</>)
})
}
</div>
</div>
<div className="col-sm-2"></div>
</div>
</>
)
}
```

```
export default FindJob
```

Footer.jsx:

```
import { BsFillSendFill } from "react-icons/bs";
import { FaFacebookF, FaTwitter, FaInstagram, FaLinkedinIn } from 'react-icons/fa'; // Import necessary icons
function Footer(){
return (
<>
<div className="container-fluid px-5 py-3 top_footer-container">
<div className="container" style={{ paddingTop: "75px" }}>
<div className="row">
<div className="col-12 col-sm-6 col-md-3 px-4 topFooter_content mb-4">
<div className="h4 text-light">ABOUT US</div>
<p className="mt-4">Heaven frucvitful doesn't cover lesser dvsays appear creeping seasons so behold.</p>
</div>
<div className="col-12 col-sm-6 col-md-3 px-4 topFooter_content mb-4">
<div className="h4 text-light">CONTACT INFO</div>
<p className="mt-4">Address: Lucknow, <br /> Uttar Pradesh, 226022. <br /> <br /> Phone: +91-xx-xxxx-xxxx <br />Email: abc123@gmail.com</p>
</div>
<div className="col-12 col-sm-6 col-md-3 px-4 topFooter_content mb-4">
<div className="h4 text-light">IMPORTANT LINK</div>
<p className="mt-4">View Project <br /> Contact us <br /> Testimonial <br /> Properties <br /> Support</p>
</div>
<div className="col-12 col-sm-6 col-md-3 px-4 topFooter_content mb-4">
<div className="h4 text-light">NEWSLETTER</div>
<p className="mt-4">Heaven frucvitful doesn't cover lesser in  days appear creeping.</p>
```

```
<div className="input-group flex-nowrap">
<input
type="email"
className="form-control"
placeholder="Email Address"
aria-label="Email Address"
aria-describedby="addon-wrapping"
/>
<span className="footer_btn" id="addon-wrapping">
<BsFillSendFill />
</span>
</div>
</div>
</div>
<div className="row text-center mt-4 pe-5 mb-2">
<div className="col-12 col-sm-6 col-md-3 pe-5 bg-light rounded-pill">
<img src="logo.png" alt="Logo" className="img-fluid" />
</div>
<div className="col-12 col-sm-6 col-md-3 pe-5 foot_content">
<div className="h4 text-light">5000+ <span style={{ color:"#eeeeee", fontSize:18,fontWeight:300 }}>Talented Hunters</span></div>
</div>
<div className="col-12 col-sm-6 col-md-3 pe-5 foot_content">
<div className="h4 text-light">451 <span style={{ color:"#eeeeee", fontSize: 18,fontWeight:300 }}>Talented Hunters</span></div>
</div>
<div className="col-12 col-sm-6 col-md-3 pe-5 foot_content">
<div className="h4 text-light">568 <span style={{ color:"#eeeeee", fontSize: 18,fontWeight:300 }}>Talented Hunters</span></div>
```

```
</div>
</div>
</div>
<hr className="mt-4" style={{ color: "gray" }} />
<div className="row text-secondary px-5 mb-3 mt-4">
<div className="col-sm-6">Copyright &copy; 2024 RecruiterX | This is developed by ...</div>
<div className="col-sm-6 f">
<a href="https://www.facebook.com" className="me-3" target="_blank" rel="noopener noreferrer">
<FaFacebookF />
</a>
<a href="https://www.twitter.com" className="me-3" target="_blank" rel="noopener noreferrer">
<FaTwitter />
</a>
<a href="https://www.instagram.com" className="me-3" target="_blank" rel="noopener noreferrer">
<FaInstagram />
</a>
<a href="https://www.linkedin.com" target="_blank" rel="noopener noreferrer">
<FaLinkedinIn />
</a>
</div>
</div>
</div>


</>
)
}
export default Footer
```

Home.jsx:

```
import Typewriter from 'typewriter-effect';
import Category from './Category';
import Resumes from './Resumes';
import  Apply from './Apply';
import Testimonial from './Testimonial';
import Support from './Support';
import BlogSection from './BlogSection';
import RecentJobs from './RecentJobs';
import { useEffect } from 'react';
function Home(){
useEffect(()=>{
localStorage.removeItem("data");
localStorage.removeItem("userType");
},[])
return (<>
<div className="container-fluid">
{/* start of hero section */}
<div className="row hero_section">
<div className="col-lg-2 col-md-1 col-sm-12"></div>
<div className="col-lg-5 col-md-6 col-sm-12 pt-5 text-center text-lg-start">
<h1 className="typewriter_hero">
<Typewriter
options={{
strings: ['Find the most exciting startup jobs...'],
autoStart: true,
loop: true,
}}
/>
</h1>
</div>
<div className="col-lg-5 col-md-5 col-sm-12"></div>
</div>

{/* category section */}
<Category />
{/* resume section */}
<Resumes />
{/* apply section */}
<Apply />
{/* recent jobs section */}
<RecentJobs />
{/* testimonial section */}
<Testimonial />
{/* support section */}
<Support />
{/* blog section */}
<BlogSection />
</div>
</>)
}

export default Home;
```

NavBar.jsx:

```
import { useEffect, useState } from 'react';
import { Link, useNavigate } from 'react-router-dom'; // Import useLocation
import { FaHome,FaBriefcase, FaCheckCircle, FaInfoCircle, FaPhoneAlt,  FaUserShield, FaUserTie, FaUser,FaUserPlus, FaUserAlt, FaUsers,  FaSignOutAlt } from "react-icons/fa";

function NavBar() {

const [data, setData] = useState("");

const [userType, setUserType] = useState("");

const nav = useNavigate();

useEffect(() => {

const temData = JSON.parse(localStorage.getItem("data"));

setData(temData);

const temUserType = JSON.parse(localStorage.getItem("userType"));

setUserType(temUserType);

}, [location.pathname]);

const adminLogout = () => {

localStorage.removeItem("data");

localStorage.removeItem("userType");

nav("/admin/login");

};

const seekerLogout = () => {

localStorage.removeItem("data");

localStorage.removeItem("userType");

nav("/seeker/login");

};

const recruiterLogout = () => {

localStorage.removeItem("data");

localStorage.removeItem("userType");

nav("/recruiter/login");

};

if (userType === "admin") {
```

```
return (
<>
<div className="row navbar_main_row mb-1 sticky-top mb-1 ps-5 pe-4">
<div className="col-12">
<nav className="navbar navbar-expand-lg navbar-light">
<div className="container-fluid">
<Link className="navbar-brand" to="">
<img className='nav_bar_logo' src={`http://localhost:9000/upload/${data.img}`} />
</Link>
<button
className="navbar-toggler"
type="button"
data-bs-toggle="collapse"
data-bs-target="#navbarNav"
aria-controls="navbarNav"
aria-expanded="false"
aria-label="Toggle navigation"
>
<span className="navbar-toggler-icon" />
</button>
<div className="collapse navbar-collapse" id="navbarNav">
<ul className="navbar-nav">
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/admin">
<FaHome className="menu-icon" />&nbsp; Dashboard
</Link>
</li>
<li className="nav-item">
```

```
<Link className="nav-link nav_font" aria-current="page" to="/admin/seekerlist">
<FaUsers className="menu-icon" />&nbsp;Seeker List
</Link>
</li>
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/admin/recruiterlist">
<FaUserTie className="menu-icon" />&nbsp;Recruiter List
</Link>
</li>
<li className="nav-item" onClick={adminLogout}>
<Link className="nav-link nav_font" aria-current="page">
<FaSignOutAlt className="menu-icon" />&nbsp;LogOut
</Link>
</li>
</ul>
</div>
</div>
</nav>
</div>
</div>
</>
);
} else if (userType === "seeker") {
return (
<>
<div className="row navbar_main_row mb-1 sticky-top mb-1 ps-5 pe-4">
<div className="col-12">
<nav className="navbar navbar-expand-lg navbar-light">
```

```
<div className="container-fluid">
<Link className="navbar-brand" to="">
<img className='nav_bar_logo' src={`http://localhost:9000/upload/${data.img}`} />
</Link>
<button
className="navbar-toggler"
type="button"
data-bs-toggle="collapse"
data-bs-target="#navbarNav"
aria-controls="navbarNav"
aria-expanded="false"
aria-label="Toggle navigation"
>
<span className="navbar-toggler-icon" />
</button>
<div className="collapse navbar-collapse" id="navbarNav">
<ul className="navbar-nav">
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/seeker">
<FaHome className="menu-icon" />&nbsp;Dashboard
</Link>
</li>
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/seeker/jobapply">
<FaBriefcase className="menu-icon" />&nbsp;Apply Job
</Link>
</li>
<li className="nav-item">
```

```
<Link className="nav-link nav_font" aria-current="page" to="/seeker/appliedjob">
<FaCheckCircle className="menu-icon" />&nbsp;Applied Job
</Link>
</li>
<li className="nav-item" onClick={seekerLogout}>
<Link className="nav-link nav_font" aria-current="page">
<FaSignOutAlt className="menu-icon" />&nbsp;LogOut
</Link>
</li>
</ul>
</div>
</div>
</nav>
</div>
</div>
</>
);
} else if (userType === "recruiter") {
return (
<>
<div className="row navbar_main_row mb-1 sticky-top mb-1 ps-5 pe-4">
<div className="col-12">
<nav className="navbar navbar-expand-lg navbar-light">
<div className="container-fluid">
<Link className="navbar-brand" to="">
<img className='nav_bar_logo' src={`http://localhost:9000/upload/${data.logo}`} />
</Link>
<button
```

```
className="navbar-toggler"
type="button"
data-bs-toggle="collapse"
data-bs-target="#navbarNav"
aria-controls="navbarNav"
aria-expanded="false"
aria-label="Toggle navigation"
>
<span className="navbar-toggler-icon" />
</button>
<div className="collapse navbar-collapse" id="navbarNav">
<ul className="navbar-nav">
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/recruiter">
<FaHome className="menu-icon" />&nbsp;Dashboard
</Link>
</li>
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/recruiter/postedjob">
<FaUser className="menu-icon" />&nbsp; postedJobs
</Link>
</li>
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/recruiter/PostJob">
<FaBriefcase />&nbsp;Job Post
</Link>
</li>
<li className="nav-item">
```

```
<Link className="nav-link nav_font" aria-current="page" to="/recruiter/appliedjob">
<FaCheckCircle className="menu-icon" />&nbsp;Applied Job
</Link>
</li>
<li className="nav-item" onClick={recruiterLogout}>
<Link className="nav-link nav_font" aria-current="page">
<FaSignOutAlt className="menu-icon" />&nbsp;LogOut
</Link>
</li>
</ul>
</div>
</div>
</nav>
</div>
</div>
</>
);
} else {
return (
<>
<div className="row navbar_main_row sticky-top mb-1 ps-5 pe-4">
<div className="col-12">
<nav className="navbar navbar-expand-lg navbar-light">
<div className="container-fluid">
<Link className="navbar-brand" to="">
<img src="/logo.png" alt="logo" height={"70px"} />
</Link>
<button
```

```
className="navbar-toggler"
type="button"
data-bs-toggle="collapse"
data-bs-target="#navbarNav"
aria-controls="navbarNav"
aria-expanded="false"
aria-label="Toggle navigation"
>
<span className="navbar-toggler-icon" />
</button>
<div className="collapse navbar-collapse" id="navbarNav">
<ul className="navbar-nav">
<li className="nav-item">
<Link className="nav-link nav_font" aria-current="page" to="/">
<FaHome />&nbsp;Home
</Link>
</li>
<li className="nav-item">
<Link className="nav-link nav_font" to="/findjob">
<FaBriefcase />&nbsp;Find a Job
</Link>
</li>
<li className="nav-item">
<Link className="nav-link nav_font" to="/about">
<FaInfoCircle />&nbsp;About
</Link>
</li>
<li className="nav-item">
```

```
<Link className="nav-link nav_font" to="/contact" tabIndex={-1} aria-disabled="true">
<FaPhoneAlt />&nbsp;Contact
</Link>
</li>
</ul>
<div className="dropdown ms-auto mt-2">
<button
className="btn btn-outline-secondary dropdown-toggle custom-btn"
type="button"
id="dropdownMenuButton1"
data-bs-toggle="dropdown"
aria-expanded="false"
>
<b>Registration</b>
</button>
<ul className="dropdown-menu custom-dropdown" aria-labelledby="dropdownMenuButton1">
<li>
<Link className="dropdown-item" to="recruiter/register">
<FaUserPlus />&nbsp;Recruiter
</Link>
</li>
<li>
<Link className="dropdown-item" to="/seeker/register">
<FaUserAlt />&nbsp;Job Seeker
</Link>
</li>
</ul>
</div>
```

```
<div className="dropdown ms-4 mt-2 me-5">
<button
className="btn btn-outline-primary dropdown-toggle custom-btn"
type="button"
id="dropdownMenuButton2"
data-bs-toggle="dropdown"
aria-expanded="false"
>
<b>Login</b>
</button>
<ul className="dropdown-menu custom-dropdown" aria-labelledby="dropdownMenuButton2">
<li>
<Link className="dropdown-item" to="/admin/login">
<FaUserShield />&nbsp;Admin Login
</Link>
</li>
<li>
<Link className="dropdown-item" to="/recruiter/login">
<FaUserTie />&nbsp;Recruiter Login
</Link>
</li>
<li>
<Link className="dropdown-item" to="/seeker/login">
<FaUser />&nbsp; Seeker Login
</Link>
</li>
</ul>
</div>
```

```
</div>
</div>
</nav>
</div>
</div>
</>
);
}
}
export default NavBar;
```

RecentJob.jsx:

```
function RecentJobs(){
const jobs_arr=[
{
img:"job-list1.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list2.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list3.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
},
{
img:"job-list4.png",
jobTitle: 'Digital Marketer',
companyName: 'Creative Agency',
location: 'Athens, Greece',
salary: '$3500 - $4000',
jobBtn: 'Full Time',
postedTime: '7 hours ago',
}
]

return (
<>
<div className="row jobs_outer">
<div className="col-12 col-sm-2"></div>
<div className="col-12 col-sm-8">
```

```
<div className="row jobchild_row pt-3">
<span className="jobs_feature mt-4 mb-3">RECENT JOB</span>
<h1 className="jobs_h1 mb-4">Featured Jobs</h1>
<div className="row job_outer_box ">
{jobs_arr.map((item) => {
return (
<div className="col-12  jobcard_category pb-3" key={item.jobTitle}>
<div className="jobchild_div ">
<div className="row">
<div className="col-12 col-sm-2 content_col text-center">
<div className="jobs_icon">
<img src={item.img} alt={item.jobTitle} />
</div>
</div>
<div className="col-12 col-sm-8 content_col">
<div className="h3 jobs_title mt-3">
{item.jobTitle}
</div>
<div className="jobs_name mt-2 me-5">{item.companyName}</div>
<div className="jobs_location mt-2 me-5">{item.location}</div>
<div className="jobs_salary mt-2 me-5">{item.salary}</div>
</div>
<div className="col-12 col-sm-2 pe-2 content_col text-center">
<div className="jobs_btn mt-4 ">{item.jobBtn}</div>
<div className="jobs_time mt-4">{item.postedTime}</div>
</div>
</div>
</div>
</div>
);
})}
</div>
</div>
</div>
<div className="col-12 col-sm-2"></div>
</div>
</>
)
}
export default RecentJobs
```

Resumes.jsx:

```
function Resumes()
{
return(
<>
<div className="row resume_image mt-1">
<div className="blur">
<div className="row">
{/* Use responsive column classes */}
<div className="col-12 col-sm-3"></div>
<div className="col-12 col-sm-6 text-center">
<div className="mt-5 text-light fs-4">Featured Tour Package</div>
<div className="h1 mt-3 text-light">Make A Different Online Resume</div>
<button className="btn upload_resume mt-5">Upload Your CV</button>
</div>
<div className="col-12 col-sm-3"></div>
</div>
</div>
</div>
</>
)
}
export default Resumes;
```

Support.jsx:

```
function Support(){
return (
<>
<div className="container support_container ">
<div className="row mt-5 ps-5">
{/* Left content section */}
<div className="col-12 col-md-1 "></div>
<div className="col-12 col-md-5">
<div className="mt-5 supportcontent_section" style={{ textAlign: "left", color: "#fc82a2", 
fontWeight:"500" }}>
WHAT WE ARE DOING
</div>
<div className="h3 mt-4  supportcontent_section" style={{ textAlign: "left", color: "#28395a" }}>
24k Talented people are getting jobs
</div>
<div className="supportcontent_section mt-4 pe-5" style={{ textAlign: "justify" }}>
<b>
Mollit anim laborum duis au dolor in voluptate velit
ess cillum dolore eu lore dsu quality mollit anim
laborumuis au dolor in voluptate velit cillum. Mollit anim laborum duis au dolor in voluptate velit
```

```
ess cillum dolore eu lore dsu quality mollit anim
laborumuis au dolor in voluptate velit cillum.
</b>
</div>
<div className="supportcontent_section mt-4 pe-5" style={{ textAlign: "justify" }}>
Mollit anim laborum. Duis aute irufo dhjkolohr in re voluptate
velit esscillumlore eu quife nrulla parihatur. Excghcepteur
signjnt occa cupidatat non inulpadeserunt mollit aboru.
temnthp incididunt ut labore mollit anim laborum suis aute
velit esscillumlore eu quife nrulla parihatur.
temnthp incididunt ut labore mollit anim laborum suis aute
velit esscillumlore eu quife nrulla parihatur.
</div>
<button className="btn_support mt-5" type="button">
Post a job
</button>
</div>

{/* Right image section */}
<div className="col-12 col-md-4 supportImage_section text-center mt-4 mt-md-0">
<div className="supportImage_sectionContent p-3">
<p className="mt-2 ">Since</p>
<p className="support_sinceSection">2024</p>
</div>
</div>
<div className="col-12 col-md-2 "></div>

</div>
```

```
</div>

</>

)

}

export default Support
```

Testinomial.jsx:

```
import Testimonials from 'react-testimonials';

function Testimonial() {

const img1 = 'testimonial-founder.png'; // Put the Image URLs

const img2 = 'testimonial-founder.png'; // Put the Image URLs

const img3 = 'testimonial-founder.png'; // Put the Image URLs

const img4 = 'testimonial-founder.png'; // Put the Image URLs

const img5 = 'testimonial-founder.png'; // Put the Image URLs

const review1=[img1, "Margaret Lawson", "Creative Director", "I am at an age where I just want to be fit and healthy our bodies are our responsibility! So start caring for your body and it will care for you. Eat clean it will care for you and workout hard."];

const review2=[img2, "Margaret Lawson", "Creative Director", "I am at an age where I just want to be fit and healthy our bodies are our responsibility! So start caring for your body and it will care for you. Eat clean it will care for you and workout hard."];

const review3=[img3, "Margaret Lawson", "Creative Director", "I am at an age where I just want to be fit and healthy our bodies are our responsibility! So start caring for your body and it will care for you. Eat clean it will care for you and workout hard."];

const review4=[img4, "Margaret Lawson", "Creative Director", "I am at an age where I just want to be fit and healthy our bodies are our responsibility! So start caring for your body and it will care for you. Eat clean it will care for you and workout hard."];

const review5=[img5, "Margaret Lawson", "Creative Director", "I am at an age where I just want to be fit and healthy our bodies are our responsibility! So start caring for your body and it will care for you. Eat clean it will care for you and workout hard."];

const items=[review1,review2,review3,review4,review5]

return (<>

<div className="row testimonial_outer" >

<div className="col-sm-1"></div>
<div className="col-sm-10">
<div className="row main_testimonial">
<Testimonials items={items} />
</div>
</div>
<div className="col-sm-1"></div>
</div>
</>)
}
export default Testimonial;
```

Recruiter:

ListofAppliers.jsx:

```
import React from 'react'
import './ApplierList.css';
function ListApplier()
{
// Sample data for applicants
const applicants = [
{
id: 1,
name: 'John Doe',
position: 'Frontend Developer',
experience: '3 years',
status: 'Pending',
},
{
id: 2,
name: 'Jane Smith',
position: 'Backend Developer',
experience: '5 years',
status: 'Reviewed',
},
{
id: 3,
name: 'Mike Johnson',
position: 'Full Stack Developer',
experience: '4 years',
status: 'Accepted',
},
{
id: 4,
name: 'John Doe',
position: 'Frontend Developer',
experience: '3 years',
status: 'Pending',
},
{
id: 5,
name: 'Jane Smith',
position: 'Backend Developer',
experience: '5 years',
status: 'Reviewed',
},
{
id: 6,
name: 'Mike Johnson',
position: 'Full Stack Developer',
experience: '4 years',
status: 'Accepted',
},
{
id: 7,
name: 'John Doe',
position: 'Frontend Developer',
experience: '3 years',
status: 'Pending',
},
{
id: 8,
name: 'Jane Smith',
position: 'Backend Developer',
experience: '5 years',
status: 'Reviewed',
},
{
id: 9,
name: 'Mike Johnson',
position: 'Full Stack Developer',
experience: '4 years',
status: 'Accepted',
}
];
return (
<div className="containe-fluid" style={{backgroundColor: '#f5f5f5'}}>
<div className="applier-list-container">
<h2 className="title">List of Appliers</h2>
<div className="card-grid">
{applicants.map(applicant => (
<div className="applier-card" key={applicant.id}>
<h3 className="applicant-name">{applicant.name}</h3>
<p><strong>Position Applied:</strong> {applicant.position}</p>
<p><strong>Experience:</strong> {applicant.experience}</p>
<p><strong>Status:</strong> <span className={`status
${applicant.status.toLowerCase()}`}>{applicant.status}</span></p>
</div>
))}
</div>
</div>
</div>
);
}
export default ListApplier;
```

PostedJobs.jsx:

```
import axios from "axios";
import { useState } from "react";
import { useEffect } from "react";


function RecruiterPostedJobs(){
const [dataId,setData]=useState()
const [jobData,setJobData]=useState([])
useEffect(()=>{
```

```
const temData=JSON.parse(localStorage.getItem("data"));
setData(temData)
getData()
},[])
const getData=async()=>{
const temData=JSON.parse(localStorage.getItem("data"));
const payload={
companyId:temData._id
}
const response= await axios.post("http://localhost:9000/api/recruiter-postedjob",payload,{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
setJobData(response.data.data)
}
}
return (<>
<div className="container my-3">
{jobData.map((el) => {
return (
<div className="card p-3 mb-3 postedjob_card">
<div className="row d-flex justify-content-center align-items-center">
{/* Logo Column */}
<div className="col-md-3 d-flex justify-content-center align-items-center">
<img src={`http://localhost:9000/upload/${el.logo}`} alt="Company Logo" className="img-fluid bg-dark" style={{ maxHeight: '100px' }} />
</div>

{/* Company Name, Job Title, and Job Type Column */}

<div className="col-md-3 d-flex justify-content-start flex-column my-3">

<h5 className='postedjob_h'> {el.name}</h5>

<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobTitle}</span></div>

<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobType}</span></div>

</div>

{/* Job Category, Location, and Salary Column */}

<div className="col-md-3">

<div className='postedjob_p2 mb-2'>Category: <span className='postedjob_p1'>
{el.category}</span></div>

<div className='postedjob_p2 mb-2'>Location: <span className='postedjob_p1'>
{el.jobLocation}</span></div>

<div className='postedjob_p2 mb-2'>Salary: <span className='postedjob_p1'>
{el.salary}</span></div>

</div>

{/* Apply Date and Vacancies Column */}

<div className="col-md-3">

<div className='postedjob_p2 mb-2'>Vacancies: <span className='postedjob_p1'>
{el.vacancies}</span></div>

<div className='postedjob_p2 mb-2'>Apply By: <span className='postedjob_p1'>
{el.applyDate}</span></div>

{/* <input className='form_button mt-3' type='submit' value='APPLY NOW' style={{width:'150px',
fontSize:"0.8em"}}/> */}

</div>

</div>

</div>

)

})}

</div>
```

```
</>)
}
export default RecruiterPostedJobs;
```

PostJob.jsx:

```
import React, { useEffect, useState } from 'react'
import { useForm } from 'react-hook-form';    // Form Validation
import { yupResolver } from '@hookform/resolvers/yup';   // Form Validation
import * as yup from 'yup';   // Form Validation
import axios from 'axios';  // fetch data

// form validation schema
const schema = yup
.object()
.shape({
jobTitle: yup.string().required(),
experience: yup.number().required().min(0).max(35),
jobType: yup.string().required(),
vacancies: yup.string().required().min(1).max(100),
jobLocation: yup.string().required(),
salary: yup.number().required().min(5000).max(250000),
applyDate: yup.string().required(),
category: yup.string().required()
})
const Postjob = () => {
const [dataId,setData]=useState("")
useEffect(()=>{
const temData=JSON.parse(localStorage.getItem("data"));
setData(temData)
},[])
// form validation useform Hook
const { register, handleSubmit,reset, formState: { errors } } = useForm({
resolver: yupResolver(schema),
});
const handleData = async(data) =>{
const payload={
companyId:dataId._id,
jobTitle:data.jobTitle,
experience:data.experience,
jobType:data.jobType,
vacancies:data.vacancies,
jobLocation:data.jobLocation,
salary:data.salary,
applyDate:data.applyDate,
category:data.category
}
const response=await axios.post("http://localhost:9000/api/recruiter-jobpost",payload,{
headers: {
"Content-Type":"application/json"
}
})
if(response.data.code==200){
alert("Job posted successfully");
reset()
}
}
return (
<>
<div className="container my-5">
<div className="row justify-content-center">
<div className="col-md-10 col-lg-8">
<div className="card shadow-sm">
<div className="card-header text-center">
<h4 className="mb-0 form_h2">Post a New Job</h4>
</div>
<div className="card-body p-4">
<form onSubmit={handleSubmit(handleData)}>
{/* Job Category & Job Title */}
<div className="row mb-3">
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1">Job Category:</label>
<input
{...register('category')}
type="text"
className="form-control"
placeholder="Enter Job Category"
/>
{errors.category?.message && (
<span className='err_span'>{errors.category?.message}</span>
)}
</div>
</div>
<div className="col-md-6">
<div className="form-group">
```

```
<label className="mb-2 ms-1 mt-1">Job Title:</label>
<input
{...register('jobTitle')}
type="text"
className="form-control"
placeholder="Enter Job Title"
/>
{errors.jobTitle?.message && (
<span className='err_span'>{errors.jobTitle?.message}</span>
)}
</div>
</div>
</div>
{/* Experience & Job Type */}
<div className="row mb-3">
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1">Experience (in years):</label>
<input
{...register('experience')}
type="number"
className="form-control"
placeholder="Enter Experience Needed To Apply"
/>
{errors.experience?.message && (
<span className='err_span'>{errors.experience?.message}</span>
)}
</div>
```

```
</div>
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1">Job Type:</label>
<input
{...register('jobType')}
type="text"
className="form-control"
placeholder="Enter Job Type"
/>
{errors.jobType?.message && (
<span className='err_span'>{errors.jobType?.message}</span>
)}
</div>
</div>
</div>
{/* Vacancies & Job Location */}
<div className="row mb-3">
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1">Number of Vacancies:</label>
<input
{...register('vacancies')}
type="number"
className="form-control"
placeholder="Enter Number of Vacancies"
/>
{errors.vacancies?.message && (
<span className='err_span'>{errors.vacancies?.message}</span>
)}
</div>
</div>
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1">Job Location:</label>
<input
{...register('jobLocation')}
type="text"
className="form-control"
placeholder="Enter Job Location"
/>
{errors.jobLocation?.message && (
<span className='err_span'>{errors.jobLocation?.message}</span>
)}
</div>
</div>
</div>

{/* Salary & Last Apply Date */}
<div className="row mb-3">
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1">Salary:</label>
<input
{...register('salary')}
type="number"
```

```
className="form-control"
placeholder="Enter Salary"
/>
{errors.salary?.message && (
<span className='err_span'>{errors.salary?.message}</span>
)}
</div>
</div>
<div className="col-md-6">
<div className="form-group">
<label className="mb-2 ms-1 mt-1">Last Apply Date:</label>
<input
{...register('applyDate')}
type="date"
className="form-control"
/>
{errors.applyDate?.message && (
<span className='err_span'>{errors.applyDate?.message}</span>
)}
</div>
</div>
</div>

<div className="d-grid ms-1 gap-2 d-md-flex justify-content-md-start">
<input
type="submit"
value='Post Job'
className="btn btn-outline-primary"
```

```
/>
</div>
</form>
</div>
</div>
</div>
</div>
</div>
</>
)
}
export default Postjob
```

RecruiterAppliedJob.jsx:

```
import axios from "axios";
import { useState } from "react";
import { useEffect } from "react";
function RecruiterAppliedJob(){
const [dataId,setData]=useState()
const [jobData,setJobData]=useState([])
useEffect(()=>{
const temData=JSON.parse(localStorage.getItem("data"));
setData(temData)
getData()
},[])


const getData=async()=>{
const temData=JSON.parse(localStorage.getItem("data"));
const payload={

companyId:temData._id

}

const response= await axios.post("http://localhost:9000/api/recruiter-applied",payload,{

headers:{

"Content-Type":"application/json"

}

})

if(response.data.code==200){

setJobData(response.data.data)

}

}

return (<>

<div className="container my-3">

{jobData.map((el) => {

console.log(el,"##############################################")

return (

<div className="card p-3 mb-3 postedjob_card">

<div className="row d-flex justify-content-center align-items-center">

{/* Logo Column */}

<div className="col-md-3 d-flex justify-content-center align-items-center">

<img src={`http://localhost:9000/upload/${el.img}`} alt="Company Logo" className="img-fluid bg-dark" style={{ maxHeight: '100px' }} />

</div>


{/* Company Name, Job Title, and Job Type Column */}

<div className="col-md-3 d-flex justify-content-start flex-column my-3">

<h5 className='postedjob_h'> {el.name}</h5>
<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobTitle}</span></div>
<div className='postedjob_p2 mb-2'><span className='postedjob_p1'> {el.jobType}</span></div>
</div>
{/* Job Category, Location, and Salary Column */}
<div className="col-md-3">
<div className='postedjob_p2 mb-2'>Category: <span className='postedjob_p1'>
{el.category}</span></div>
<div className='postedjob_p2 mb-2'>Location: <span className='postedjob_p1'>
{el.jobLocation}</span></div>
<div className='postedjob_p2 mb-2'>Salary: <span className='postedjob_p1'>
{el.salary}</span></div>
</div>
{/* Apply Date and Vacancies Column */}
<div className="col-md-3">
<div className='postedjob_p2 mb-2'>Contact: <span className='postedjob_p1'>
{el?.contact}</span></div>
<div className='postedjob_p2 mb-2'>Email: <span className='postedjob_p1'>
{el?.email}</span></div>
<div className='postedjob_p2 mb-2'>
<a href={`http://localhost:9000/upload/${el.resume}`} target="_blank">Download</a></div>
</div>
</div>
</div>
)
})}

</div>
</>)
}
export default RecruiterAppliedJob;
```

RecruiterLogin.jsx:

```
import { useForm } from 'react-hook-form';
import { yupResolver } from '@hookform/resolvers/yup';
import * as yup from 'yup';
import axios from 'axios';
import { useNavigate } from 'react-router-dom';
const schema = yup
.object()
.shape({
email: yup.string().required().email(),
password: yup.string().required()
})
function RecruiterLogin(){
const navigate=useNavigate()
const {register,handleSubmit, formState:{errors}}=useForm({
resolver:yupResolver(schema)
})
const handleData=async(data)=>{
const payLoad={
email:data.email,
password:data.password
}
const response=await   axios.post("http://localhost:9000/api/Recruiter-login",payLoad,{
headers:{
"Content-Type":"application/json"
}
})
if(response.data.code==200){
```

```
localStorage.setItem("data",JSON.stringify(response.data.data))
localStorage.setItem("userType",JSON.stringify('recruiter'))
alert("Login Successfull !...")
navigate('/recruiter')
}else if(response.data.code==203){
alert("Your Account is Blocked contact to Admin")
}
else if (response.data.code==302){
alert("Invaild Email or password")
}
}
return(<>
<div className="row">
<div className="col-sm-1"></div>
<div className="col-sm-10">
<div className="row mt-3">
<div className="col-sm-6 form_div_register recuiter_register">
<div className="register_img"></div>
</div>
<div className="col-sm-6 form_div_login">
<h2 className="recruiter_signUp_text">Recruiter <span style={{color:
"#db0249"}}>Login</span></h2>
<div className="p-4">
<form onSubmit={handleSubmit(handleData)}>
<div className="row mb-4 pt-3 ">
<input  className="form-control "
placeholder="Enter Your email"
type="text"
```

```
{...register('email')}
/>
{errors.email?.message && <span  className='error_msg'>{errors.email?.message}</span>}
</div>
<div className="row mb-4 pt-3">
<input  className="form-control "
placeholder="Enter Your password"
type="password"
{...register('password')}
/>
{errors.password?.message && <span  className='error_msg'>{errors.password?.message}</span>}
</div>

<div className="row mb-4 pt-3">
<input  className="register_submit form-control "
type="submit"
value="Login"
/>
</div>
</form>
</div>
</div>
</div>
</div>
<div className="col-sm-1"></div>
</div>
<br />
<br />
```

```
</>)
}
export default RecruiterLogin;
```

RecruiterProfile.jsx:

```
import React ,{useEffect}from 'react';
import { useForm } from 'react-hook-form';    // Form Validation
import { yupResolver } from '@hookform/resolvers/yup';   // Form Validation
import * as yup from 'yup';   // Form Validation
import '../../app.css'
import axios from 'axios';
const schema = yup
.object()
.shape({
name: yup.string().required().min(2).max(30),
email: yup.string().required().email(),
contact: yup.string().required(),
password: yup.string().required(),
location: yup.string().required(),
logo: yup.mixed().required()
})
const RecruiterUpdate = () => {
useEffect(()=>{
const userdetails= JSON.parse(localStorage.getItem("data"));
if(userdetails){
setValue("name",userdetails.name);
setValue("email",userdetails.email);
setValue("contact",userdetails.contact);
setValue("password",userdetails.password);
setValue("location",userdetails.location)
}
},[])
// form validation useform Hook
const { register, handleSubmit,setValue, formState: { errors } } = useForm({
resolver: yupResolver(schema),
});
const handleData = async (data) => {
const formData = new FormData();
formData.append('name', data.name);
formData.append('email', data.email);
formData.append('contact', data.contact);
formData.append('password', data.password);
formData.append('location', data.location);
formData.append('logo', data.logo[0]);
if(!data || data.logo.length==0){
alert("Please select an image");
return ;
}
const temData= JSON.parse(localStorage.getItem("data"));
const response=await axios.put(`http://localhost:9000/api/recruiter-
update/${temData._id}`,formData,{
headers:{
        "Content-Type":"multipart/form-data"
}
})
if(response.data.code==200){
alert("Admin Updated Successfully");
```

```
}

}


return(
<div className="container my-5">
<div className="row justify-content-center">
        <div className="col-md-8">
        <div className="card shadow-sm form_card">
        <div className="card-header form_cardheader text-center">
        <h4 className="mb-0 form_h2">Recruiter Profile Update</h4>
        </div>
        <div className="card-body p-4">
        <form onSubmit={handleSubmit(handleData)}>
                  <div className="row mb-3">
                  <div className="col-md-6">
                  <div className="form-group">
                  <label className="mb-2 ms-1 mt-1 form_label">Name :</label>
                  <input {...register('name')} type="text" className="form-control"
                  placeholder="Enter Your Name" />
                  {errors.name?.message && <span
                  className='err_span'>{errors.name?.message}</span>}
                  </div>
                  </div>
                  <div className="col-md-6">
                  <div className="form-group">
                  <label className="mb-2 ms-1 mt-1 form_label">Email :</label>
                  <input {...register('email')} type="text" className="form-control"
                  placeholder="Enter Your Email" />
```

```
{errors.email?.message && <span className='err_span'>{errors.email?.message}</span>}

</div>

</div>

</div>

<div className="row mb-3">

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Contact :</label>

<input {...register('contact')} type="number" className="form-control" placeholder="Enter Your Contact"/>

{errors.contact?.message && <span className='err_span'>{errors.contact?.message}</span>}

</div>

</div>

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Password :</label>

<input type='password' {...register('password')} className="form-control" placeholder='Enter Your Password'/>

{errors.password?.message && <span className='err_span'>{errors.password?.message}</span>}

</div>

</div>

</div>

<div className="row mb-3">

<div className="col-md-6">

<div className="form-group">

<label className="mb-2 ms-1 mt-1 form_label">Location :</label>
```

```
            <input {...register('location')} type="text" className="form-control" placeholder="Enter Your Password"/>
            {errors.location?.message && <span className='err_span'>{errors.location?.message}</span>}
            </div>
            </div>
            <div className="col-md-6">
            <div className="form-group">
            <label className="mb-2 ms-1 mt-1 form_label">logo :</label>
            <input {...register('logo')} type="file" className="form-control" accept='logo/*' />
            {errors.logo?.message && <span className='err_span'>{errors.logo?.message}</span>}
            </div>
            </div>
            </div>
            <div className="d-grid">
            <input type="submit" value='UPDATE PROFILE' className="form_button"/>
            </div>
      </form>
      </div>
      </div>
      </div>
  </div>
  </div>
)}
  export default RecruiterUpdate
```

RecruiterRegiter.jsx:

```
  import { useForm } from 'react-hook-form';
  import { yupResolver } from '@hookform/resolvers/yup';
```

```
import * as yup from 'yup';
import axios from 'axios';
const schema = yup
.object()
.shape({
name: yup.string().required().min(2).max(15),
email: yup.string().required().email(),
contact: yup.string().required(),
password: yup.string().required(),
location: yup.string().required(),
logo: yup.mixed().required(),
})

function RecruiterRegister(){

const { register, handleSubmit, formState: { errors } } = useForm({
resolver: yupResolver(schema),
});
const handleData=async(data)=>{
const formData=new FormData();
formData.append("name",data.name);
formData.append("email",data.email)
formData.append("contact",data.contact)
formData.append("password",data.password)
formData.append("location",data.location)
formData.append("logo",data.logo[0])
await   axios.post("http://localhost:9000/api/recruiter-register",formData,{
            headers:{
```

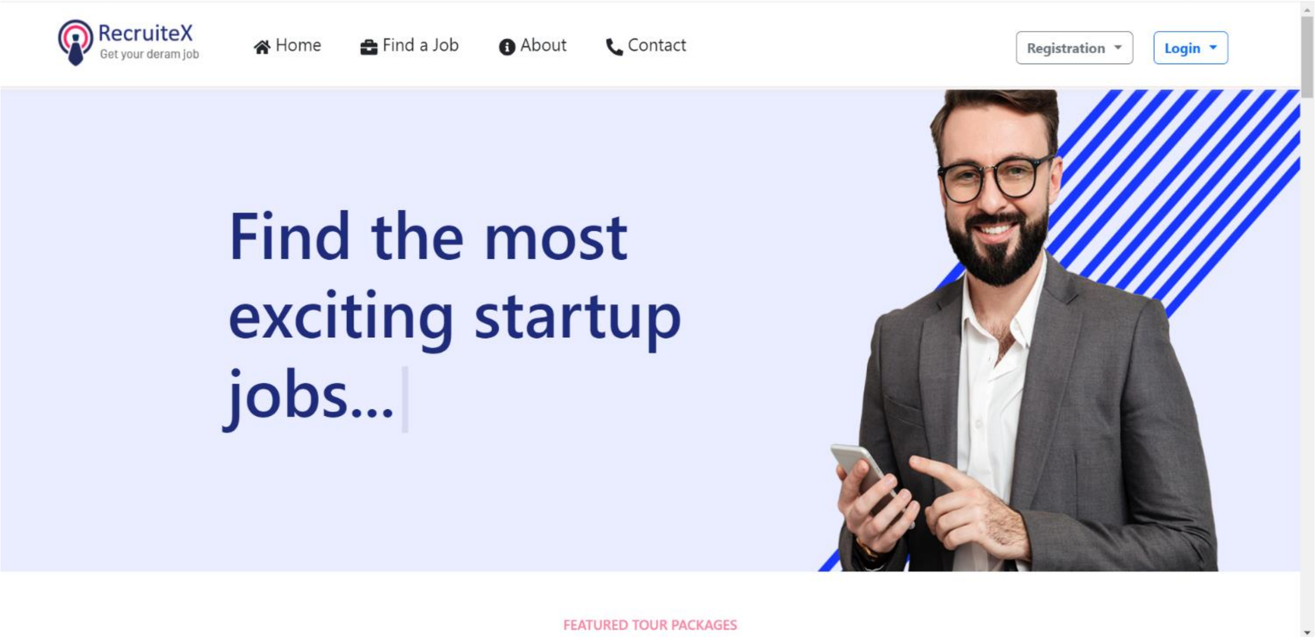
```
        "Content-Type":"multipart/form-data"
        }
})
alert("Registation SuccessFull !.")
}
return(<>
<div className="row">
<div className="col-sm-1"></div>
<div className="col-sm-10">
        <div className="row">
        <div className="col-sm-6 form_div_register recuiter_register">
        <div className="register_img"></div>
        </div>
        <div className="col-sm-6 form_div_register ">
                <h2 className="recruiter_signUp_text">Recruiter <span style={{color:
                "#db0249"}}>SingUp</span></h2>
        <div className="p-4">
        <form onSubmit={handleSubmit(handleData)}>
        <div className="row mb-4">
                <input  className="form-control "
                placeholder="Enter Your Name"
                type="text"
                {...register('name')}
                />
                {errors.name?.message &&
                <span  className='error_msg'>{errors.name?.message}</span>}
        </div>
        <div className="row mb-4 ">
                <input  className="form-control "
```

```
            placeholder="Enter Your email"

            type="text"

            {...register('email')}

            />

            {errors.email?.message &&
            <span  className='error_msg'>{errors.email?.message}</span>}

        </div>

        <div className="row mb-4">

            <input  className="form-control "

            placeholder="Enter Your contact"

            type="number"

            {...register('contact')}/>

            {errors.contact?.message &&
            <span  className='error_msg'>{errors.contact?.message}</span>}

        </div>

        <div className="row mb-4">

            <input  className="form-control "

            placeholder="Enter Your password"

            type="password"

            {...register('password')}

            />

            {errors.password?.message &&
            <span  className='error_msg'>{errors.password?.message}</span>}

        </div>

        <div className="row mb-4">

            <input  className="form-control "

            placeholder="Enter Your location"

            type="text"

            {...register('location')}/>
```

```
                {errors.location?.message &&
                <span  className='error_msg'>{errors.location?.message}</span>}

            </div>

            <div className="row mb-4">

                <input  className="form-control "

                type="file"

                {...register('logo')}

                />

                {errors.logo?.message &&
                <span  className='error_msg'>{errors.logo?.message}</span>}

            </div>

            <div className="row mb-4">

                <input  className="register_submit form-control "

                type="submit"

                value="SignUp"/></div>

            </form>

            </div>

            </div>

            </div>

        </div>

        <div className="col-sm-1"></div>

        </div>

        <br />

        <br />

        </>)}

    export default RecruiterRegister;
```

8.2 SCREENSHOTS

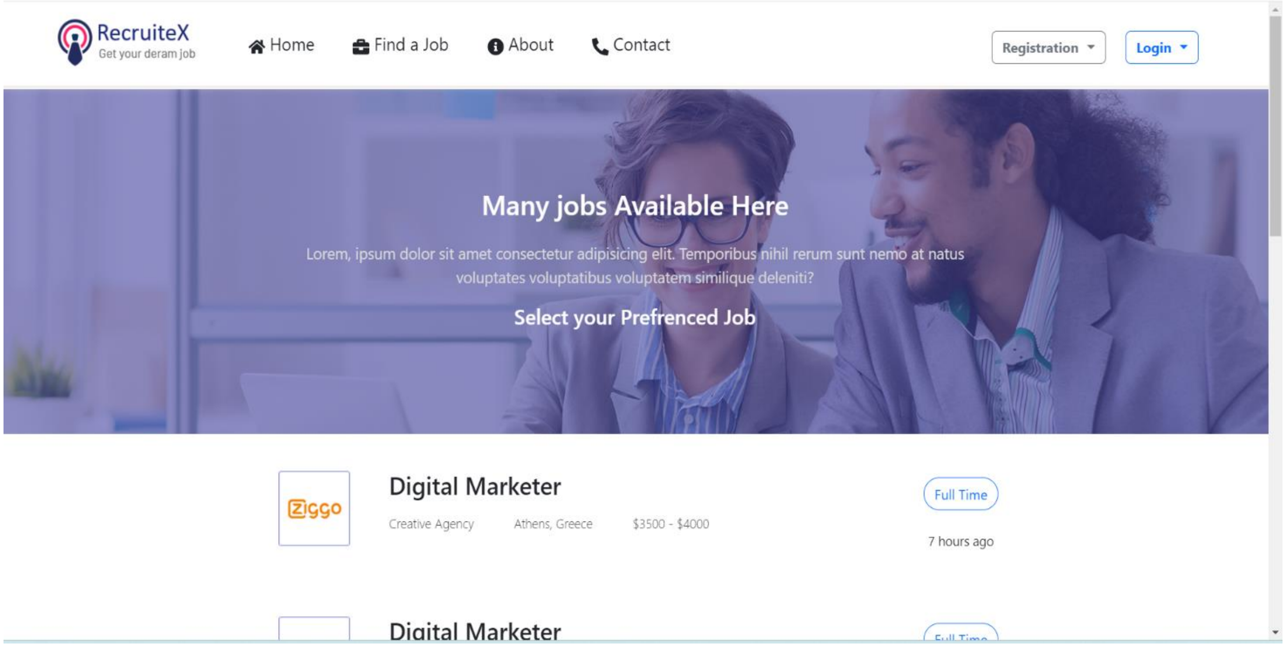
HOME



Find the most exciting startup jobs...

FEATURED TOUR PACKAGES

FindsJob



Many jobs Available Here

Lorem, ipsum dolor sit amet consectetur adipisicing elit. Temporibus nihil rerum sunt nemo at natus voluptates voluptatibus voluptatem similique deleniti?

Select your Prefrenced Job

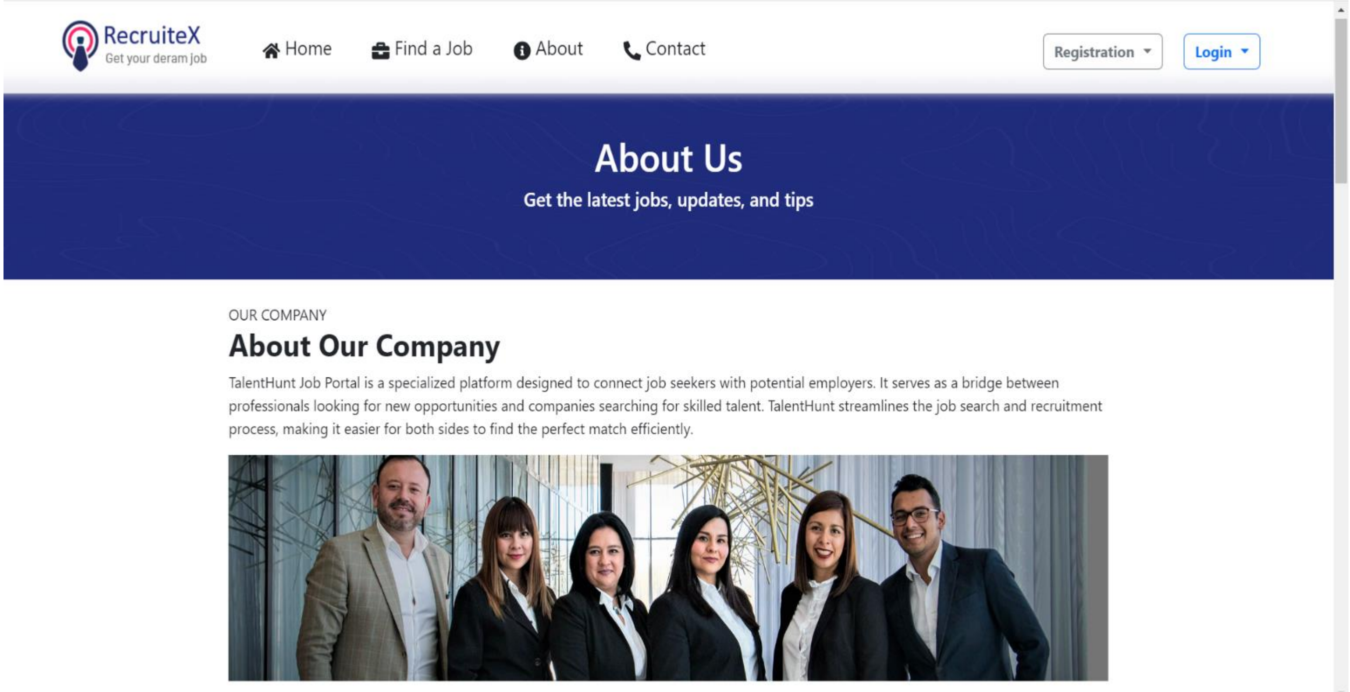
Digital Marketer

Creative Agency Athens, Greece $3500 - $4000

7 hours ago

Digital Marketer

About



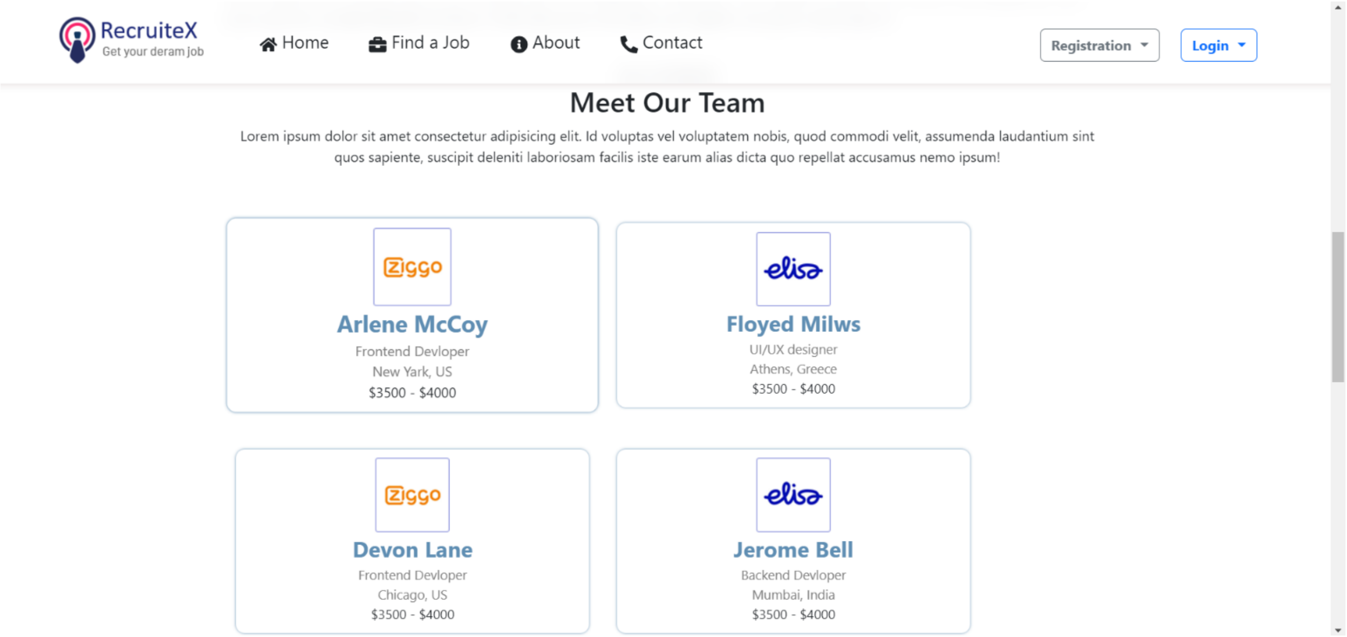
About Us

Get the latest jobs, updates, and tips

OUR COMPANY

About Our Company

TalentHunt Job Portal is a specialized platform designed to connect job seekers with potential employers. It serves as a bridge between professionals looking for new opportunities and companies searching for skilled talent. TalentHunt streamlines the job search and recruitment process, making it easier for both sides to find the perfect match efficiently.



Meet Our Team

Lorem ipsum dolor sit amet consectetur adipisicing elit. Id voluptas vel voluptatem nobis, quod commodi velit, assumenda laudantium sint quos sapiente, suscipit deleniti laboriosam facilis iste earum alias dicta quo repellat accusamus nemo ipsum!

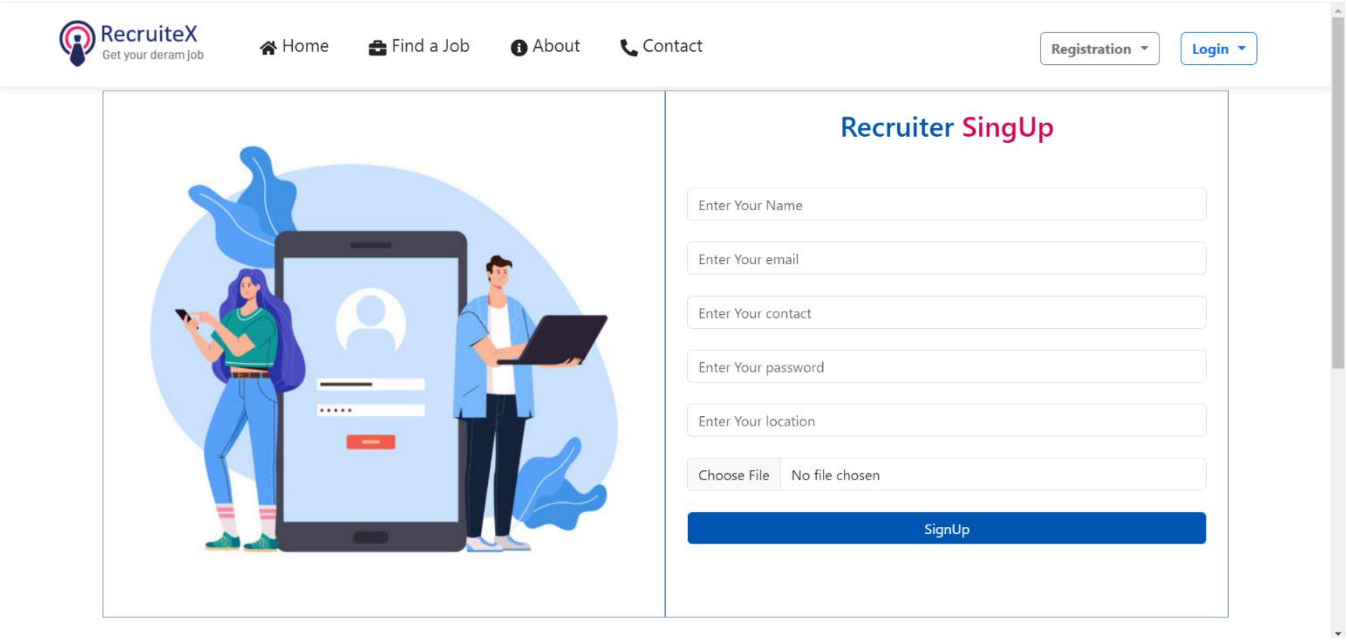
Arlene McCoy
Frontend Devloper
New Yark, US
$3500 - $4000

Floyed Milws
UI/UX designer
Athens, Greece
$3500 - $4000

Devon Lane
Frontend Devloper
Chicago, US
$3500 - $4000

Jerome Bell
Backend Devloper
Mumbai, India
$3500 - $4000

Recruiter Signup



Recruiter SingUp

Enter Your Name

Enter Your email

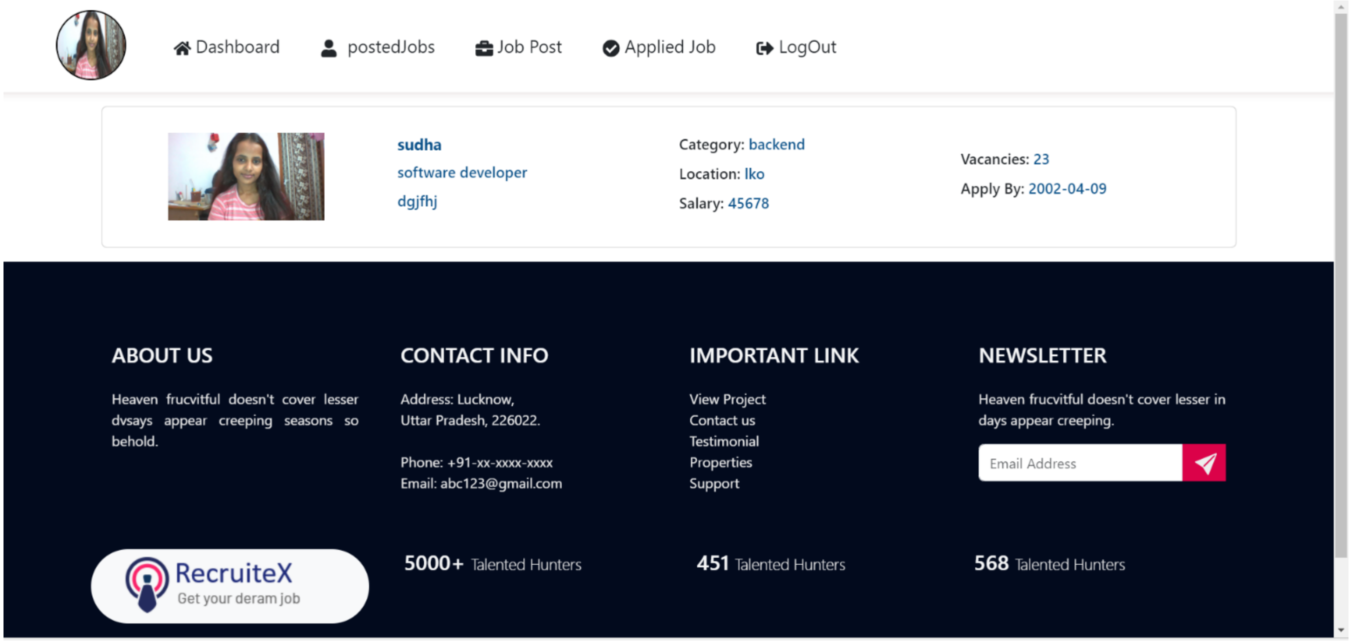
Enter Your contact

Enter Your password

Enter Your location

Choose File No file chosen

PostedJobs



sudha
software developer
dgjfhj

Category: backend
Location: lko
Salary: 45678

Vacancies: 23
Apply By: 2002-04-09

ABOUT US

Heaven frucvitful doesn't cover lesser dvsays appear creeping seasons so behold.

CONTACT INFO

Address: Lucknow,
Uttar Pradesh, 226022.

Phone: +91-xx-xxxx-xxxx
Email: abc123@gmail.com

IMPORTANT LINK

View Project
Contact us
Testimonial
Properties
Support

NEWSLETTER

Heaven frucvitful doesn't cover lesser in days appear creeping.

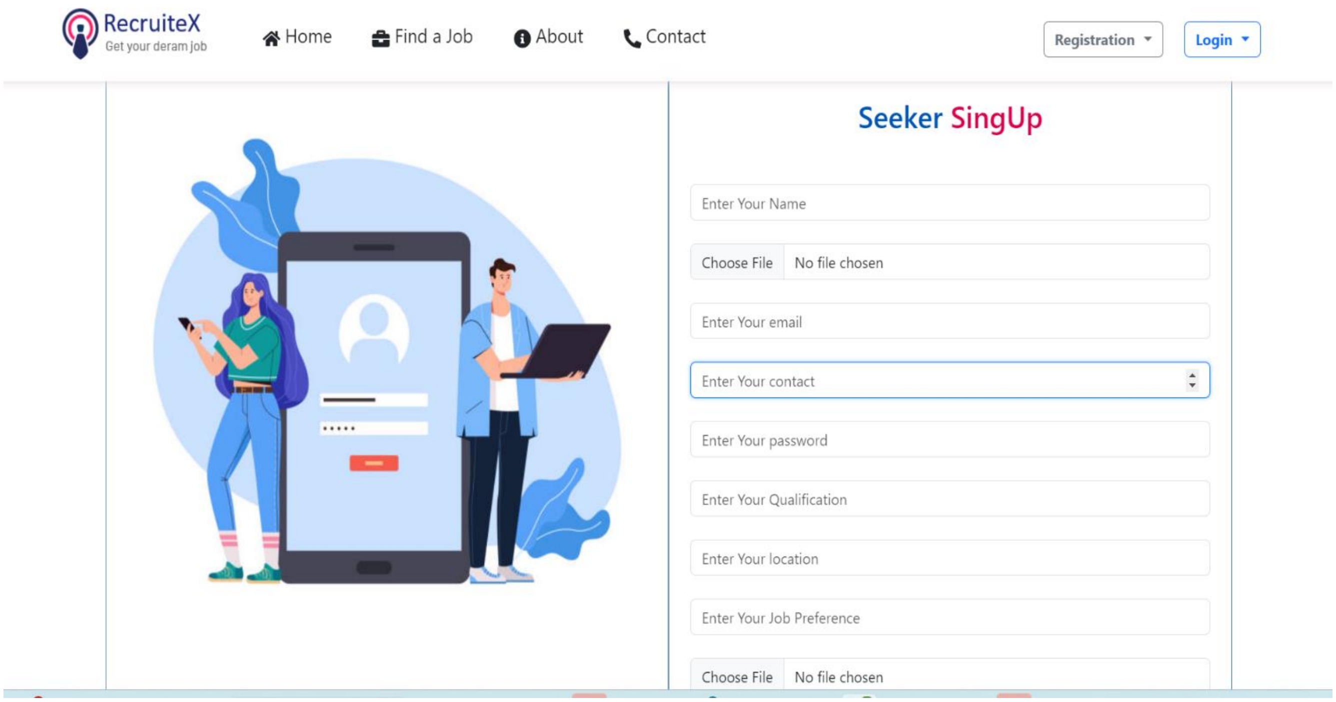
Email Address

5000+ Talented Hunters

451 Talented Hunters

568 Talented Hunters

Seeker SignUp



Seeker SingUp

Enter Your Name

Choose File No file chosen

Enter Your email

Enter Your contact

Enter Your password

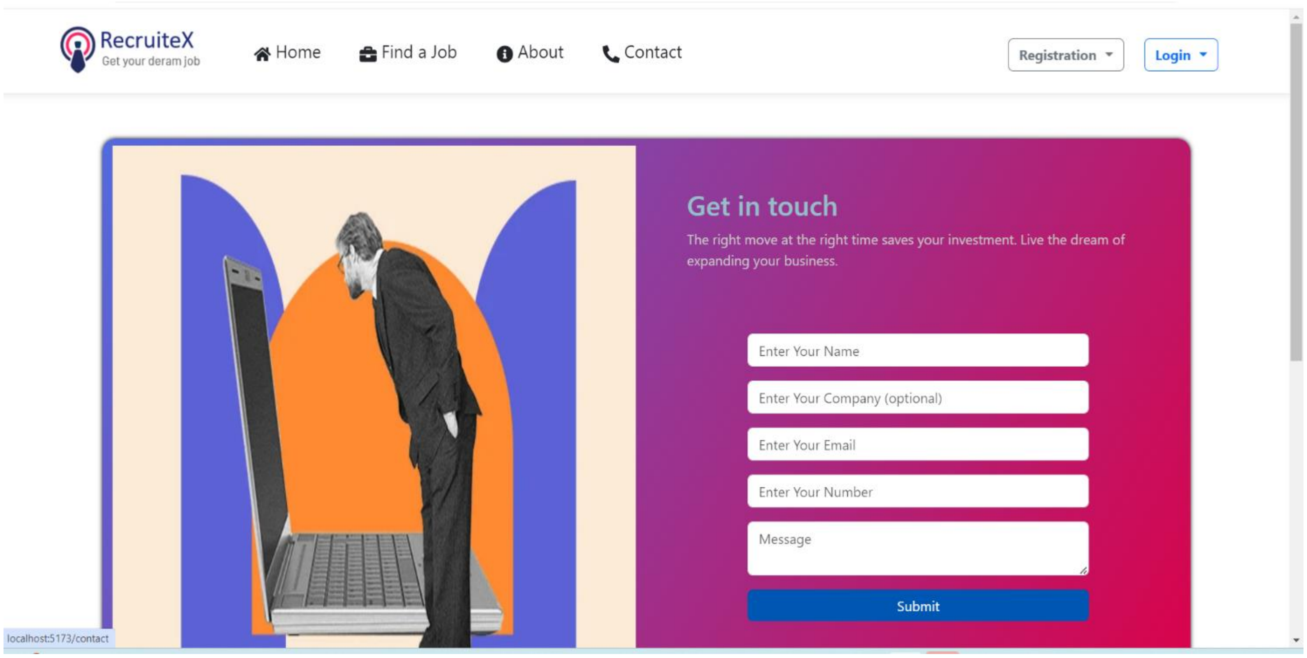
Enter Your Qualification

Enter Your location

Enter Your Job Preference

Choose File No file chosen

Contact us



Get in touch

The right move at the right time saves your investment. Live the dream of expanding your business.

Enter Your Name

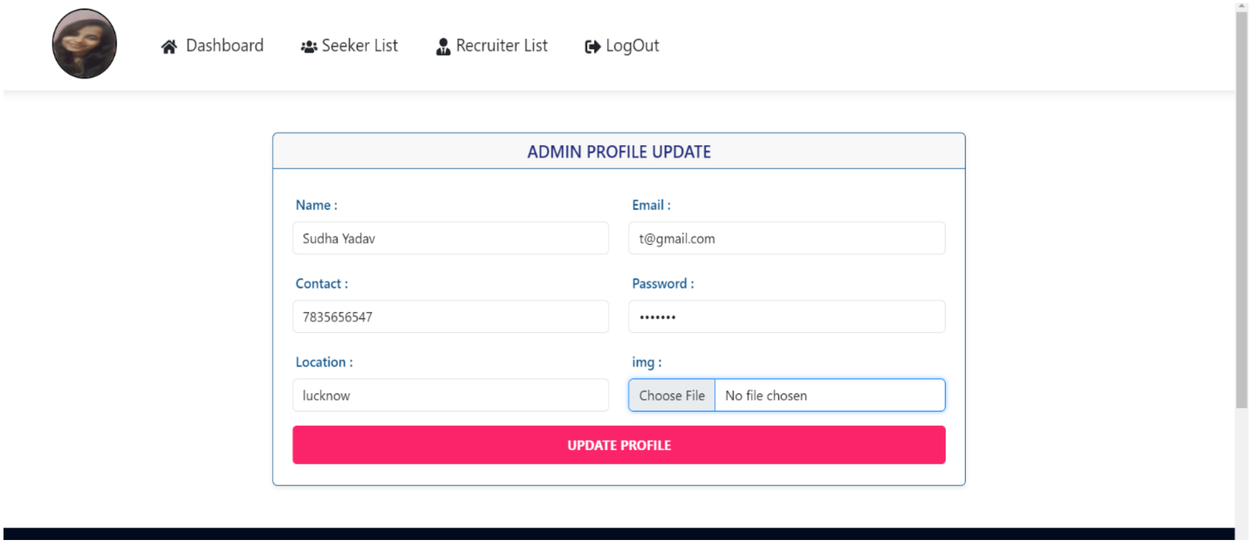
Enter Your Company (optional)

Enter Your Email

Enter Your Number

Message

Admin login



ADMIN PROFILE UPDATE

Name : Sudha Yadav

Email : t@gmail.com

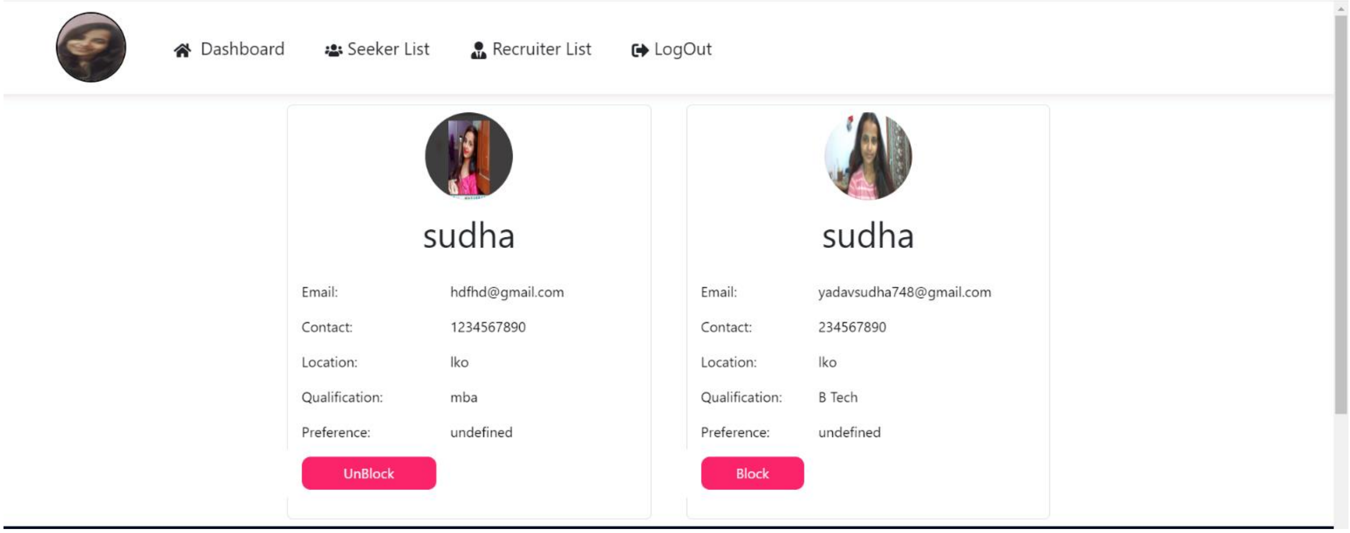
Contact : 7835656547

Password : •••••••

Location : lucknow

img : No file chosen

SeekerList



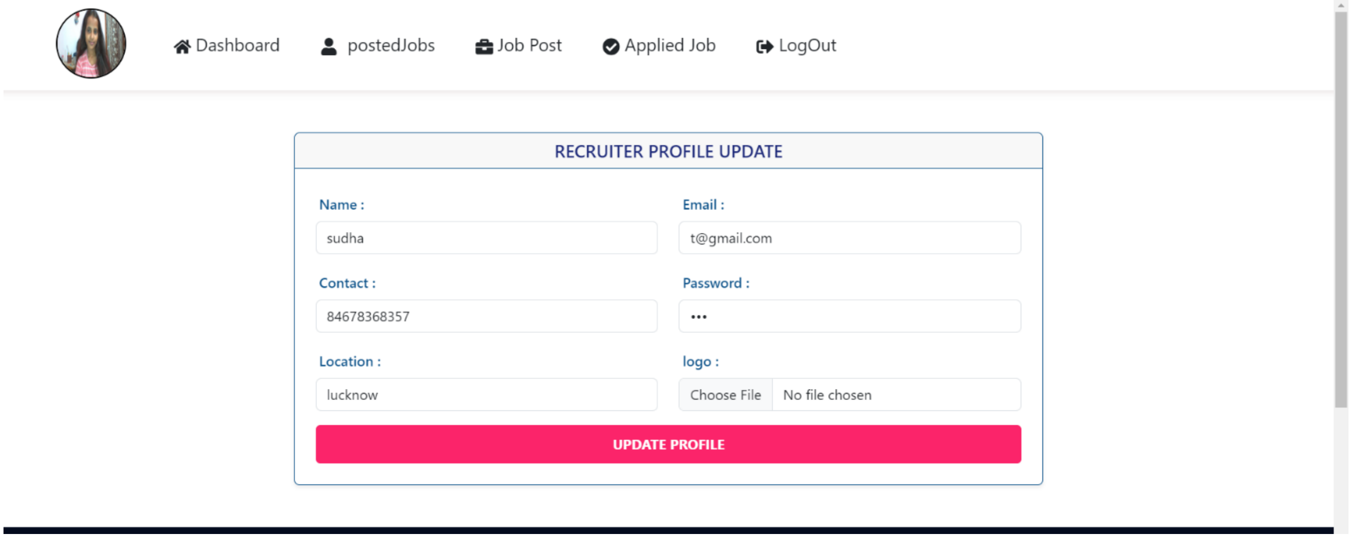
sudha

Email:	hdfhd@gmail.com
Contact:	1234567890
Location:	lko
Qualification:	mba
Preference:	undefined

sudha

Email:	yadavsudha748@gmail.com
Contact:	234567890
Location:	lko
Qualification:	B Tech
Preference:	undefined

Recruiter Login



RECRUITER PROFILE UPDATE

Name : sudha

Email : t@gmail.com

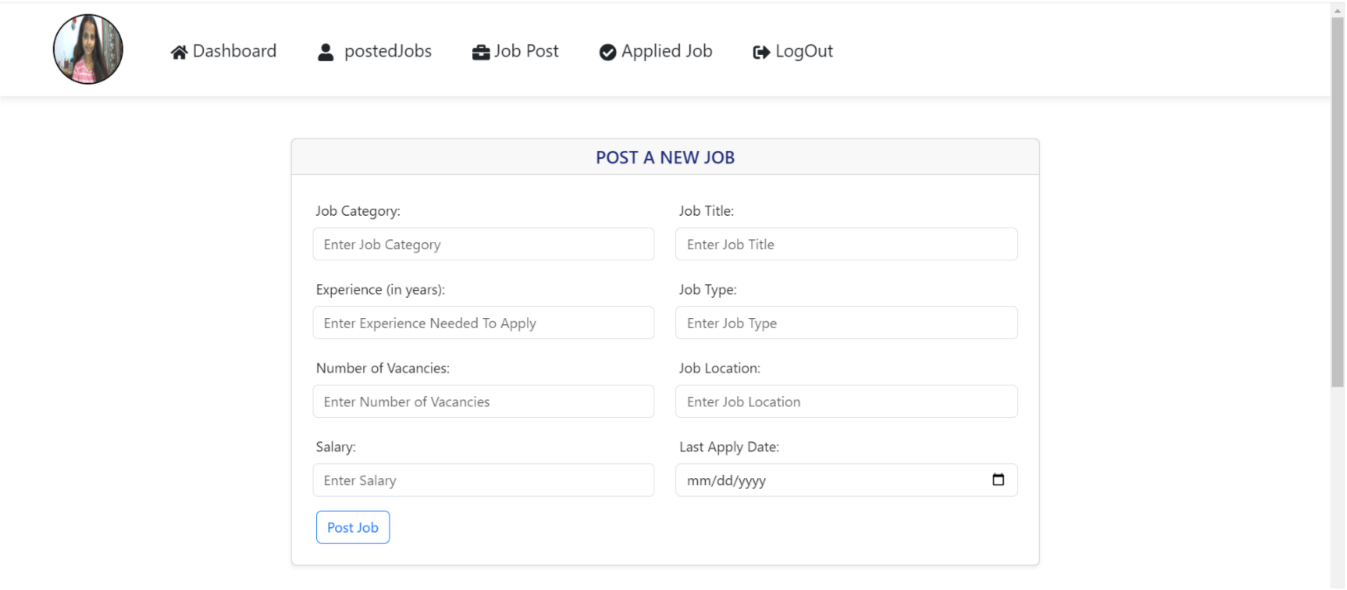
Contact : 84678368357

Password : •••

Location : lucknow

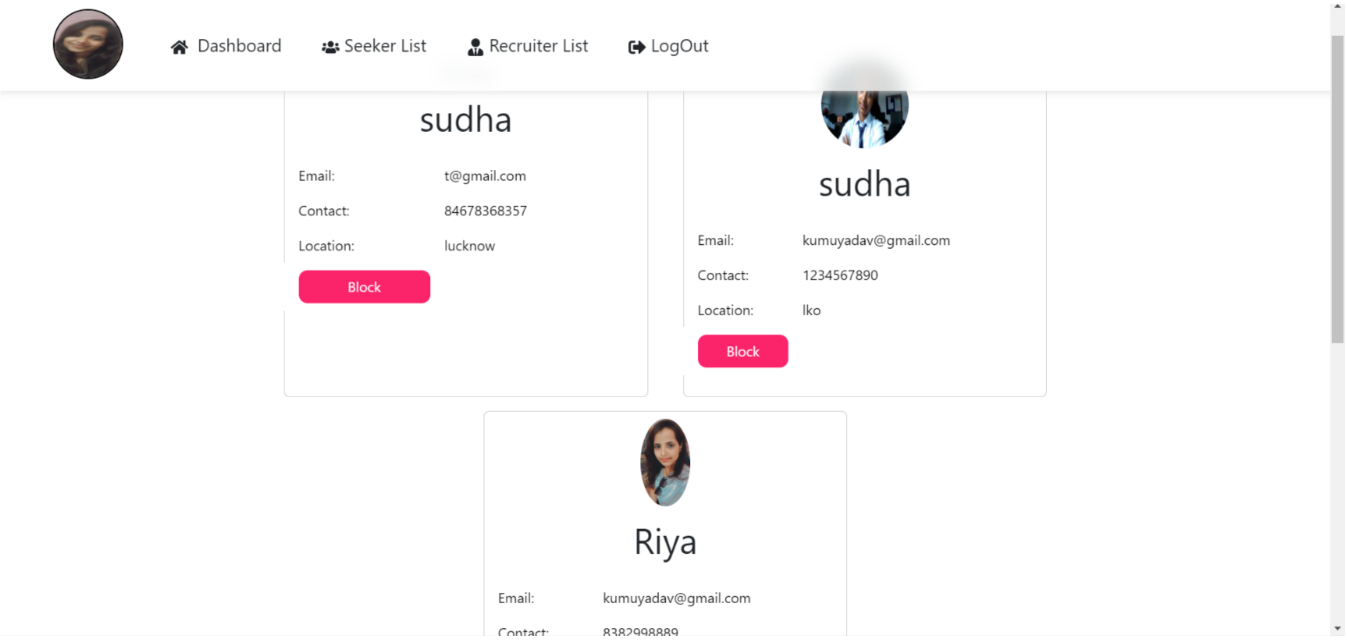
logo : No file chosen

JobPost



POST A NEW JOB

Job Category: Enter Job Category	Job Title: Enter Job Title
Experience (in years): Enter Experience Needed To Apply	Job Type: Enter Job Type
Number of Vacancies: Enter Number of Vacancies	Job Location: Enter Job Location
Salary: Enter Salary	Last Apply Date: mm/dd/yyyy



sudha

Email:	t@gmail.com
Contact:	84678368357
Location:	lucknow

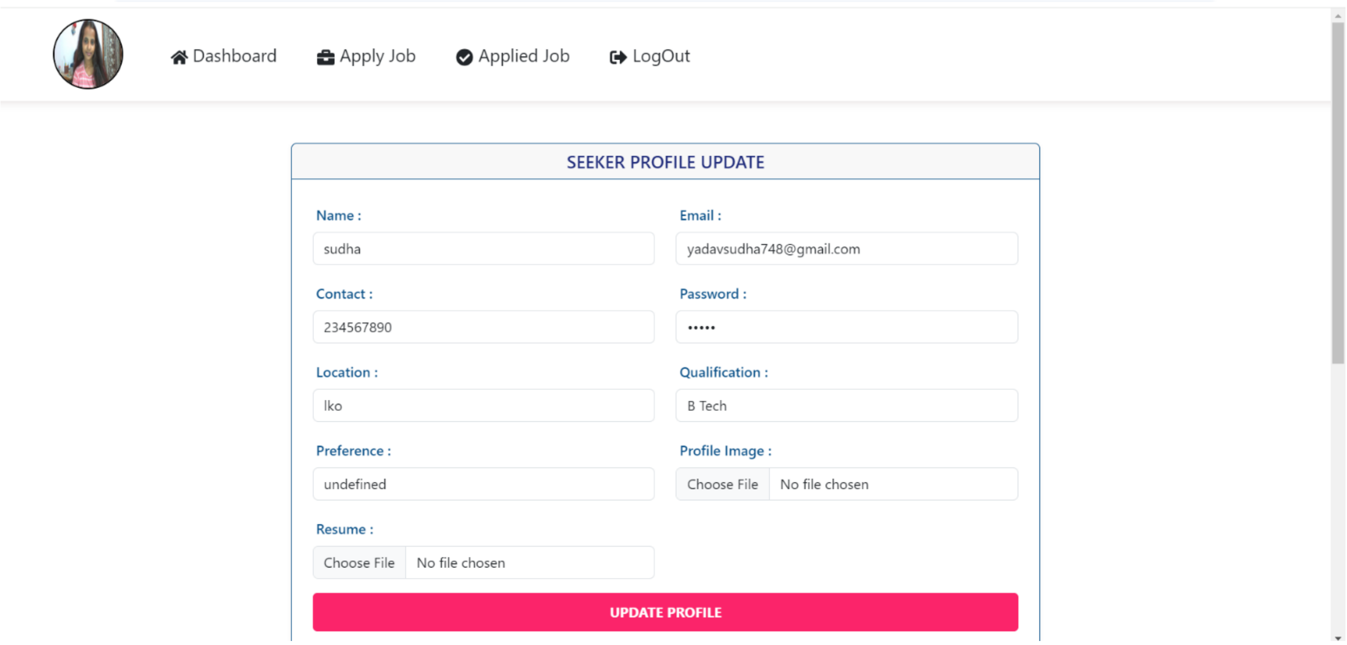
sudha

Email:	kumuyadav@gmail.com
Contact:	1234567890
Location:	lko

Riya

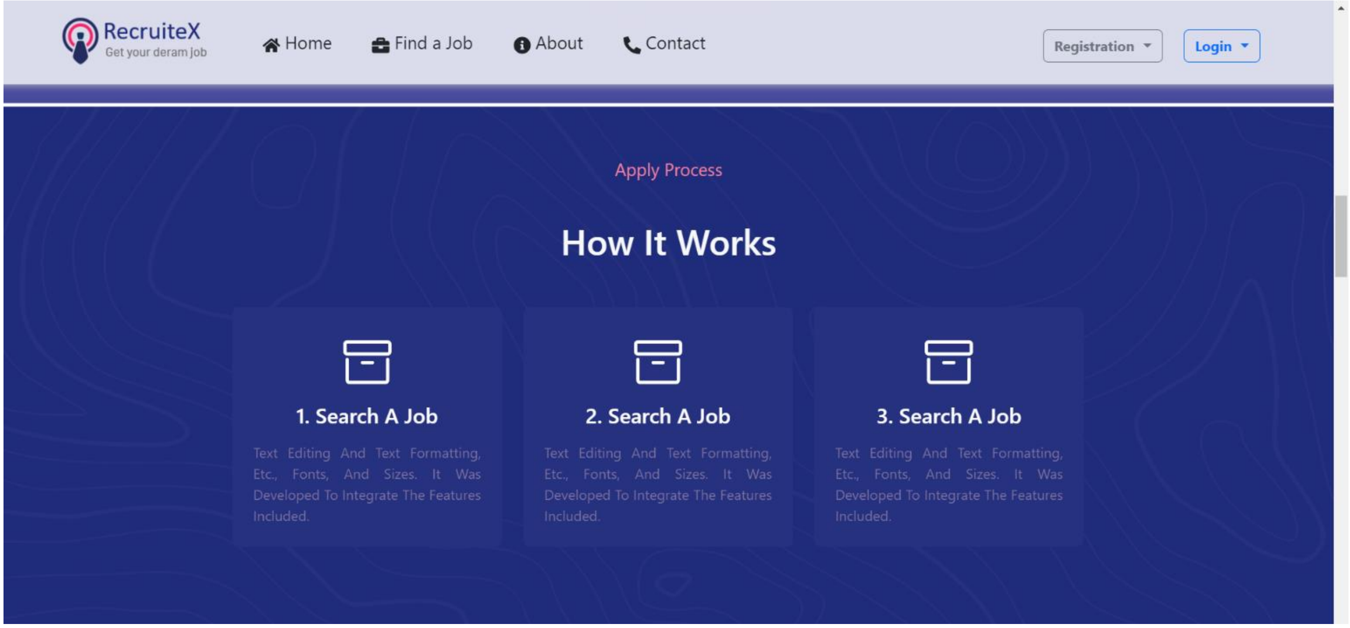
Email:	kumuyadav@gmail.com
Contact:	8382998889

Seeker Login



SEEKER PROFILE UPDATE

Name : sudha	Email : yadavsudha748@gmail.com
Contact : 234567890	Password : •••••
Location : lko	Qualification : B Tech
Preference : undefined	Profile Image : Choose File No file chosen
Resume : Choose File No file chosen	



Apply Process

How It Works

1. Search A Job

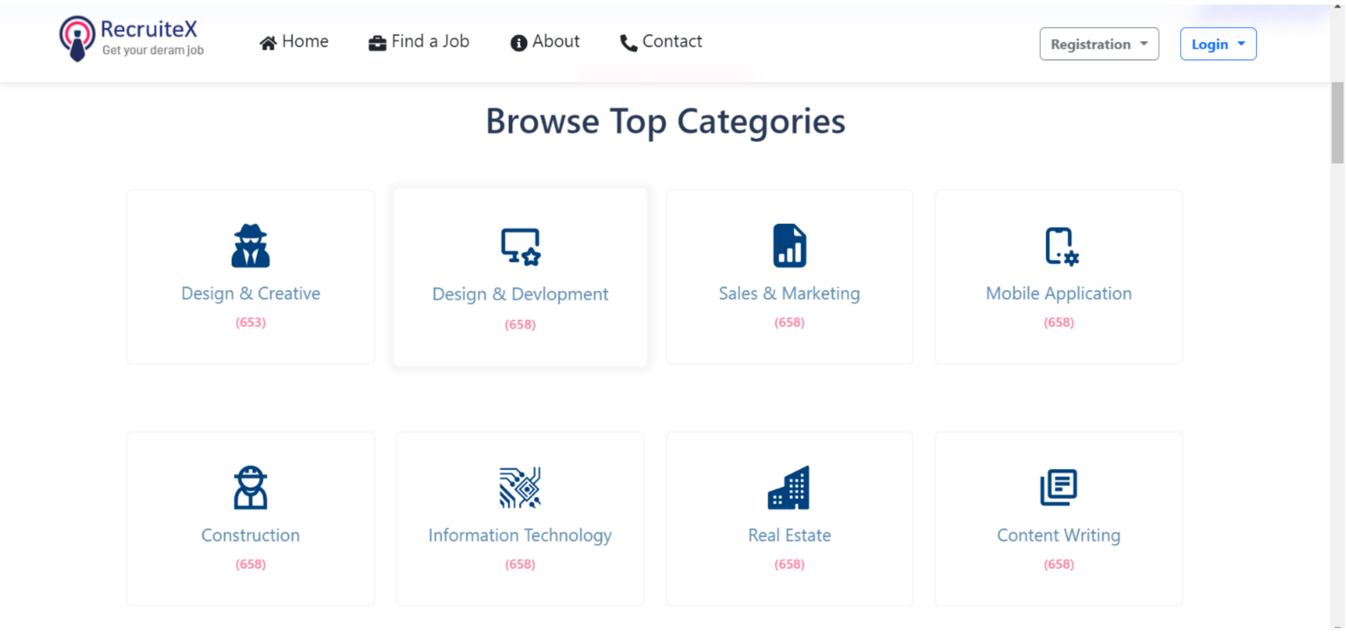
Text Editing And Text Formatting, Etc., Fonts, And Sizes. It Was Developed To Integrate The Features Included.

2. Search A Job

Text Editing And Text Formatting, Etc., Fonts, And Sizes. It Was Developed To Integrate The Features Included.

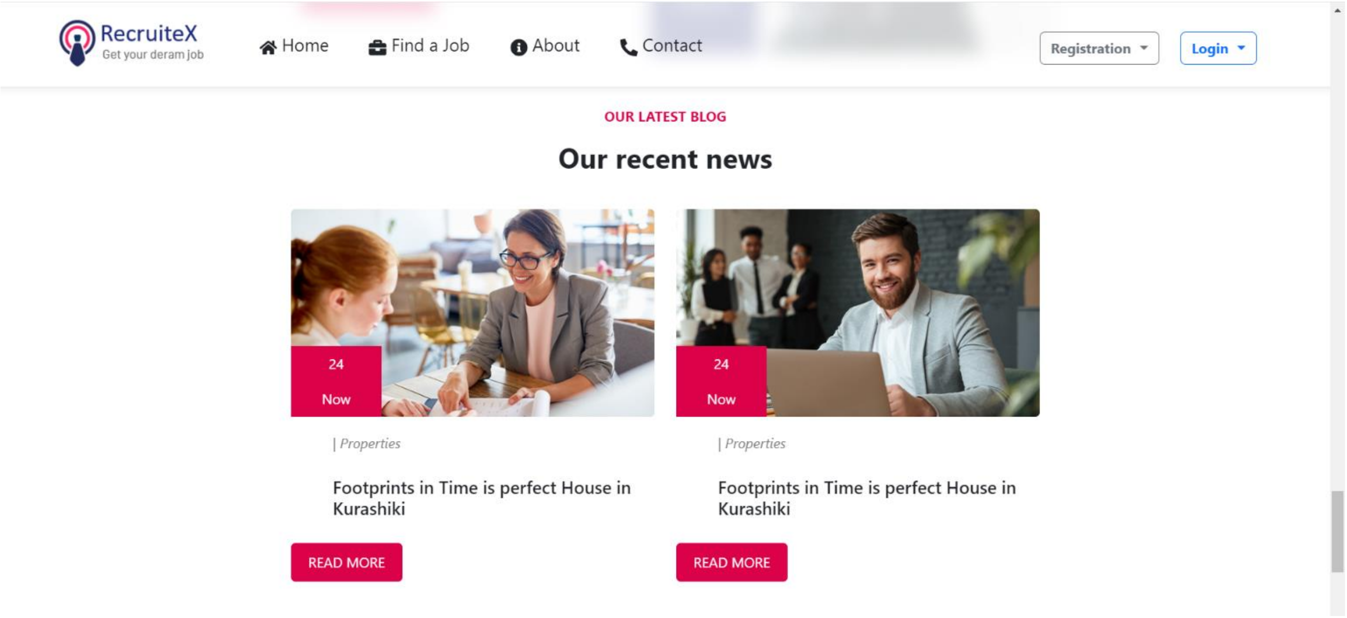
3. Search A Job

Text Editing And Text Formatting, Etc., Fonts, And Sizes. It Was Developed To Integrate The Features Included.



Browse Top Categories

Design & Creative (653)	Design & Devlopment (658)	Sales & Marketing (658)	Mobile Application (658)
Construction (658)	Information Technology (658)	Real Estate (658)	Content Writing (658)



OUR LATEST BLOG

Our recent news

24
Now

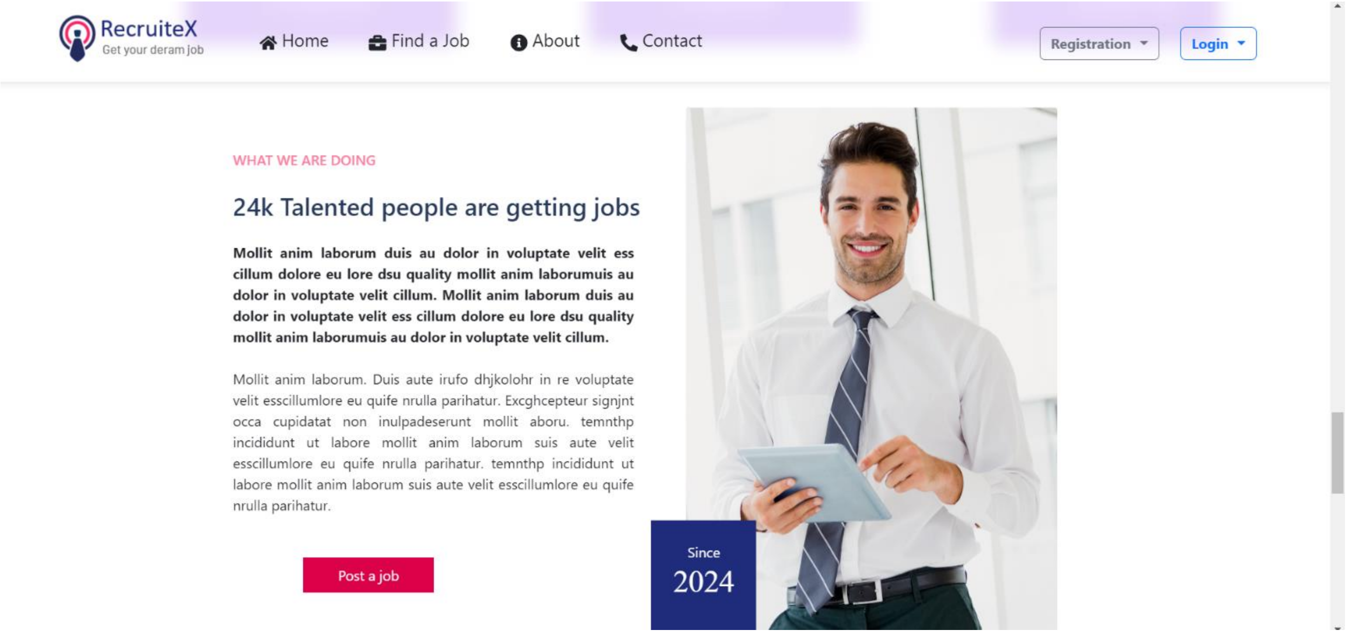
| Properties

Footprints in Time is perfect House in Kurashiki

24
Now

| Properties

Footprints in Time is perfect House in Kurashiki

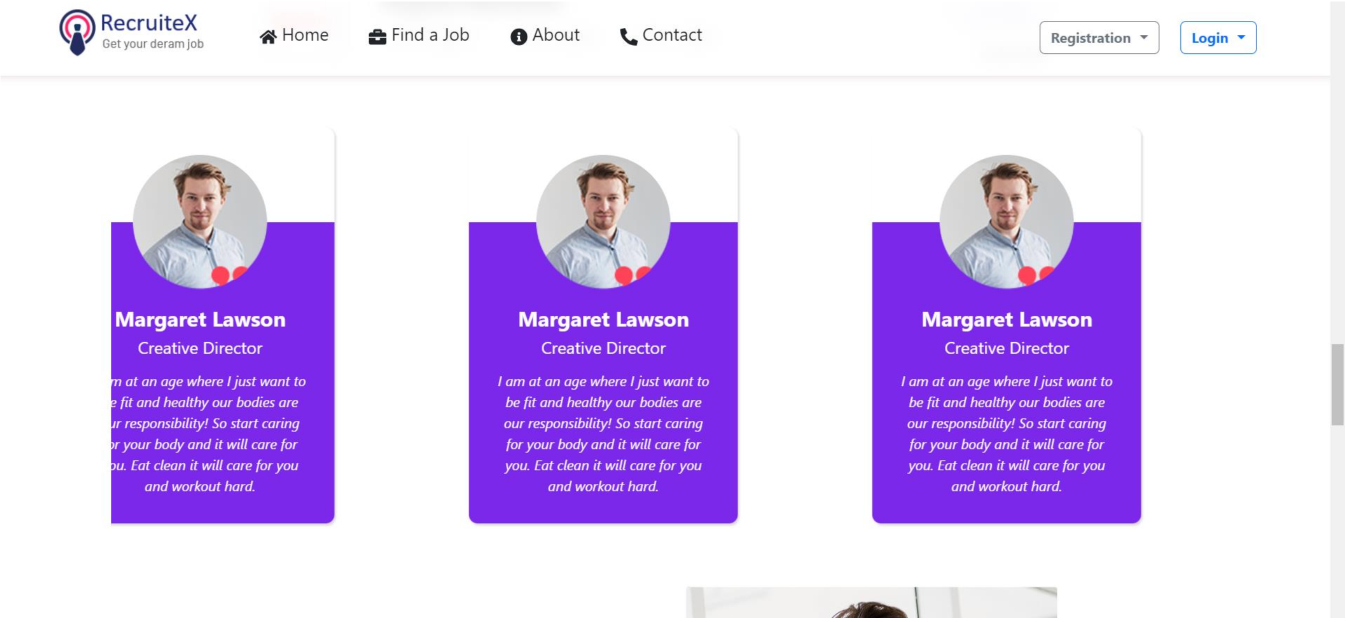


WHAT WE ARE DOING

24k Talented people are getting jobs

Mollit anim laborum duis au dolor in voluptate velit ess cillum dolore eu lore dsu quality mollit anim laborumuis au dolor in voluptate velit cillum. Mollit anim laborum duis au dolor in voluptate velit ess cillum dolore eu lore dsu quality mollit anim laborumuis au dolor in voluptate velit cillum.

Mollit anim laborum. Duis aute irufo dhjkolohr in re voluptate velit esscillumlore eu quife nrulla parihatur. Excghcepteur signjnt occa cupidatat non inulpadeserunt mollit aboru. temnthp incididunt ut labore mollit anim laborum suis aute velit esscillumlore eu quife nrulla parihatur. temnthp incididunt ut labore mollit anim laborum suis aute velit esscillumlore eu quife nrulla parihatur.



Margaret Lawson
Creative Director

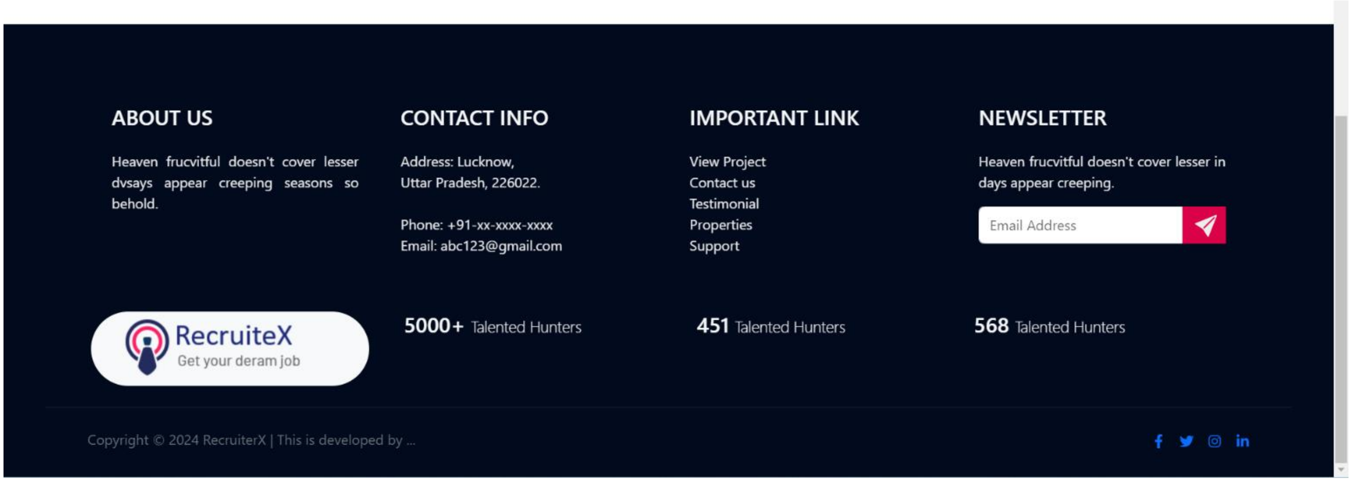
m at an age where I just want to e fit and healthy our bodies are ur responsibility! So start caring or your body and it will care for ou. Eat clean it will care for you and workout hard.

Margaret Lawson
Creative Director

I am at an age where I just want to be fit and healthy our bodies are our responsibility! So start caring for your body and it will care for you. Eat clean it will care for you and workout hard.

Margaret Lawson
Creative Director

I am at an age where I just want to be fit and healthy our bodies are our responsibility! So start caring for your body and it will care for you. Eat clean it will care for you and workout hard.



ABOUT US

Heaven frucvitful doesn't cover lesser dvsays appear creeping seasons so behold.

CONTACT INFO

Address: Lucknow,
Uttar Pradesh, 226022.

Phone: +91-xx-xxxx-xxxx
Email: abc123@gmail.com

IMPORTANT LINK

View Project
Contact us
Testimonial
Properties
Support

NEWSLETTER

Heaven frucvitful doesn't cover lesser in days appear creeping.

5000+ Talented Hunters

451 Talented Hunters

568 Talented Hunters

Copyright © 2024 RecruiterX | This is developed by ...

9.FUTURE SCOPE

Following modification or upgrades can be done in system.

1. More than one company can be integrated through this software.

2. Web services can be used to know exact donation status of packets.

3. Client can check there donation delivery status online

10. CONCLUSION:

In conclusion, RecruiteX aims to provide a user-friendly web application that connects job seekers with employers, streamlining the recruitment process. By fostering a collaborative community, the platform allows users to share resources and insights, making job hunting and hiring more efficient.

Designed to be accessible and affordable, RecruiteX will continually evolve, incorporating new features based on user feedback. Our goal is to empower both job seekers and employers, creating a dynamic recruitment ecosystem that meets their needs.