TABLE OF CONTENTS

INDEX

INTRODUCTION

1. INTRODUCTION

Online Job portal system is an application which connects employer and job seekers where employers are the source of the resources and the jobseeker can find and apply for their targeted job. In Online Job portal system we use NodeJs and MongoDB database. This is the project which keeps records of the employer, jobseeker and administrator. Online Job portal system has three modules i.e. Jobseeker, Employer and Admin.

Admin Module : This module provides administrator related functionalities. Administrator manages entire application and maintains the profiles of applicants and employers.

Employer Module : This module provides functionalities related to employers. Employers can post vacancy details and update the details as and when necessary. Employers can search through applicant resumes based on different criteria.

Jobseeker Module: In this section, Jobseekers can view job which is posted by employer and apply those jobs. Jobseeker can also update his profile, change the password and recover the password.

SYSTEM STUDY

2. SYSTEM STUDY

2.1 EXISTING SYSTEM

Present system is manual. : The present system requires applicants to search through print and visual media for job opportunities.

- Applicants need to apply for jobs using conventional methods and appear for interview on a specified date at a specified location.
- Employers need to advertise the vacancies and sort all applicant details, conduct selection procedures and complete the formalities.
- This approach is tedious and requires much effort and resources.

2.2 PROPOSED SYSTEM

The proposed system is a web based application which allows applicants and employers to register their details.

The following are the advantages of the proposed system:

- Applicants can browse through the vacancy details that are posted and can apply for the jobs online
- Employers can browse through the posted resumes and select suitable candidates.

2.3 PROBLEM DEFINITION AND PROJECT DESCRIPTION

The project titled as "ONLINE JOB PORTAL SYSTEM" is a web based application which provides facilities for getting jobs through online. There is no need of much time because everything about the job will be posted through online. So to reduce the time management, the "ONLINE JOB PORTAL SYSTEM" will be of great help. This software has different modules which enhances the proper working of the system and they are

- Admin
- Employer
- Jobseeker

ADMIN MODULE

- ✓ Manage Employer and Jobseeker details
- ✓ Manage Entire application
- ✓ Admin can change the details presented in the website
- ✓ Maintains the profiles of applicants and employers.

EMPLOYER MODULE

- ✓ It provides functionalities related to employers
- ✓ Employers can post vacancy details and update the details
- ✓ Employers can search through applicant resumes
- ✓ Manage the jobs and post the jobs

JOBSEEKER MODULE

- ✓ To view job which is posted by employer and apply those jobs
- ✓ To view the response of the applied jobs

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1REQUIREMENTS SPECIFICATION

HARDWARE REQUIREMENTS

Processor	:	Intel Core i3 or higher
RAM	:	Minimum 4GB (8 GB recommended)
Hard disk	:	Minimum 40 Gb free space
Keyboard	:	Standard 104-keys Keyboard
Mouse	:	Optical Mouse

SOFTWARE REQUIREMENTS

Operating System	:	Windows 10/11 (64-bit) or equivalent
Local Server	:	Node.js with Express framework
Database	:	MongoDB (NoSQL database)
Front End	:	React.js (HTML, CSS, JavaScript)
Back End	:	Node.js with Express
Database Connectivity	:	Mongoose
Browser	:	Google Chrome / Firefox
Other Tools	:	Visual Studio Code, Postman

3.2 FEASIBILITY STUDY

A feasibility analysis is conducted to decide if the solution considered to meet the criteria is feasible and workable in the software. During the feasibility study, information such as resource availability, cost estimates for software production, advantages of the software to the enterprise after its development, and cost to be expended on its maintenance is determined. The feasibility study aims to getting jobs through online. The system has been tested for feasibility in the following points:

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

TECHNICAL FEASIBILITY

Technical feasibility evaluates the available infrastructure (such as hardware and software) and technologies needed to meet the user needs of software under time and budget constraints. The following are the activities often performed by technical feasibility.

- ✓ Examines whether there are technical guarantees of accuracy, reliability, ease of access and data security.
- ✓ Determines whether the application infrastructure is well-established.
- ✓ Ensures whether the proposed system provides adequate response to inquiries, regardless of the number or location of users

The current system developed is technically feasible as it provides the technical guarantee of accuracy, reliability, security and easy access to the users.

OPERATIONAL FEASIBILITY

The proposed system is beneficial only if it can be turned out into information system which will meet the operating requirements of the organization. The extent to which the required software completes a sequence of steps to address challenges and requirements of the developer and users respectively is measured by operational viability. The following are the operations carried out by operational feasibility:

- ✓ Determines whether sufficient support for the organization is provided from the users.
- ✓ Ensures proper working of the system if it is being developed and implemented.
- ✓ Checks whether there will be any resistance from the users that will ruin the possible benefits of the application.

This Online Job Portal System would ensure the optimal utilization of computer resources and would help in the improvement of performance status.

ECONOMIC FEASIBILITY

A system can be developed technically and that will be used if installed must still be a good investment for the organization. Economic feasibility needs to consider the expenses made on purchasing, such as hardware purchasing and required activities to carry out software development. It is also necessary to consider the benefits that can be achieved by developing the software. Software is economically feasible when it focuses on the issues listed below.

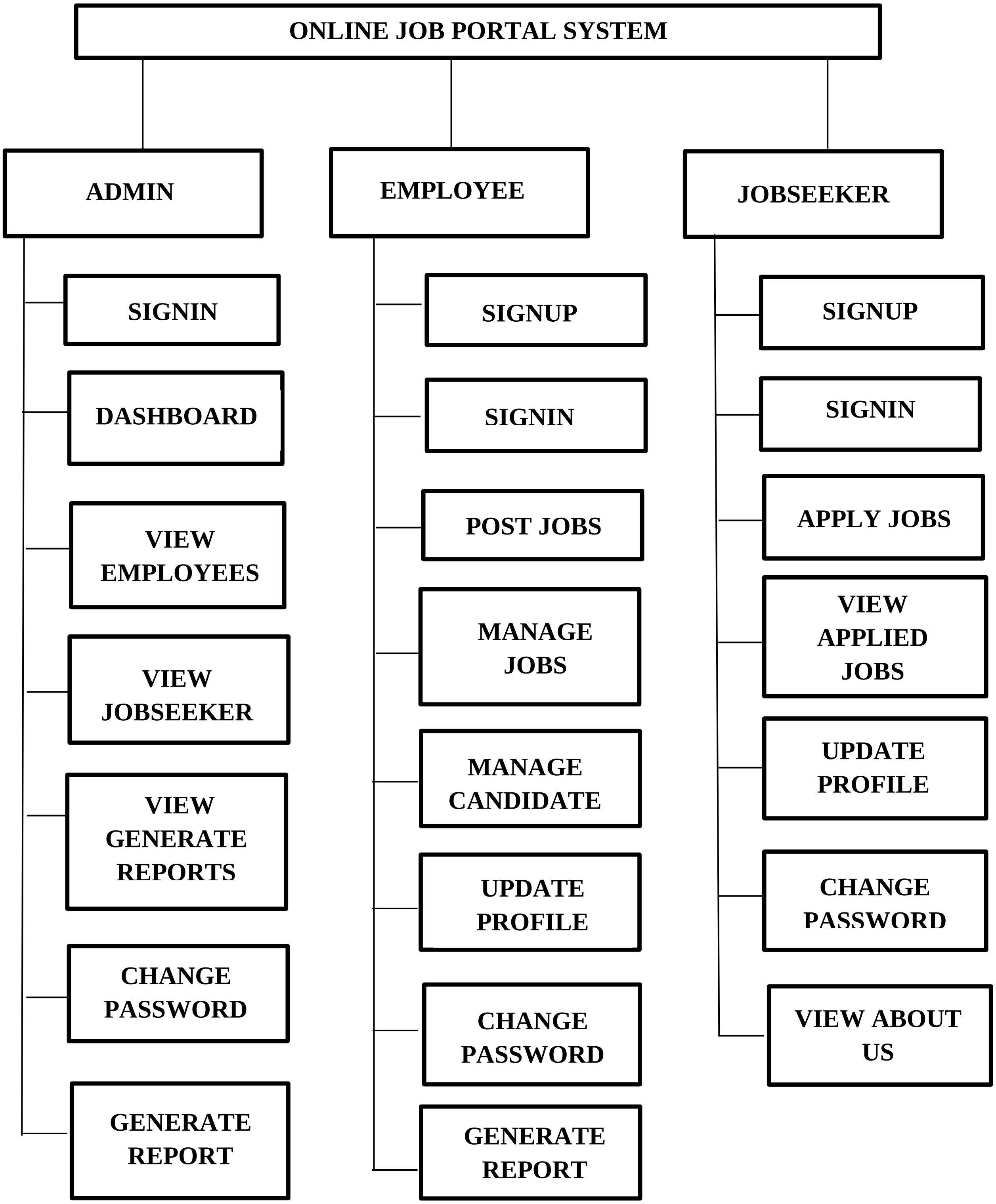
- ✓ Expense incurred on software development for achieving long-term gains for an organization.
- ✓ Expenses required to conduct elicitation and requirements analysis
- ✓ Hardware and software cost, development team, and training cost.

This system is economically feasible. Since this system is developed using the existing resources and technologies, there is nominal expenditure which ensures the economic feasibility of the system.

SYSTEM DESIGN

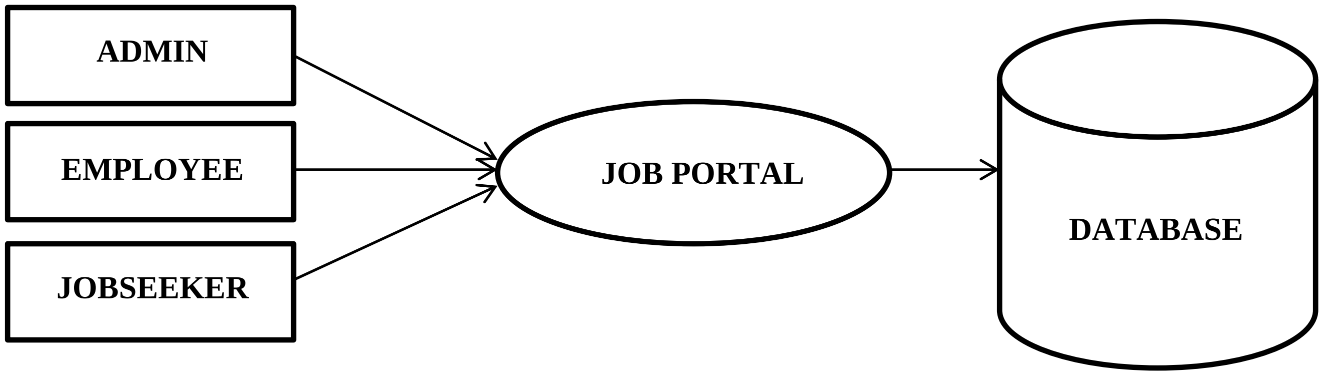
SYSTEM DESIGN

4.1 ARCHITECTURAL DESIGN

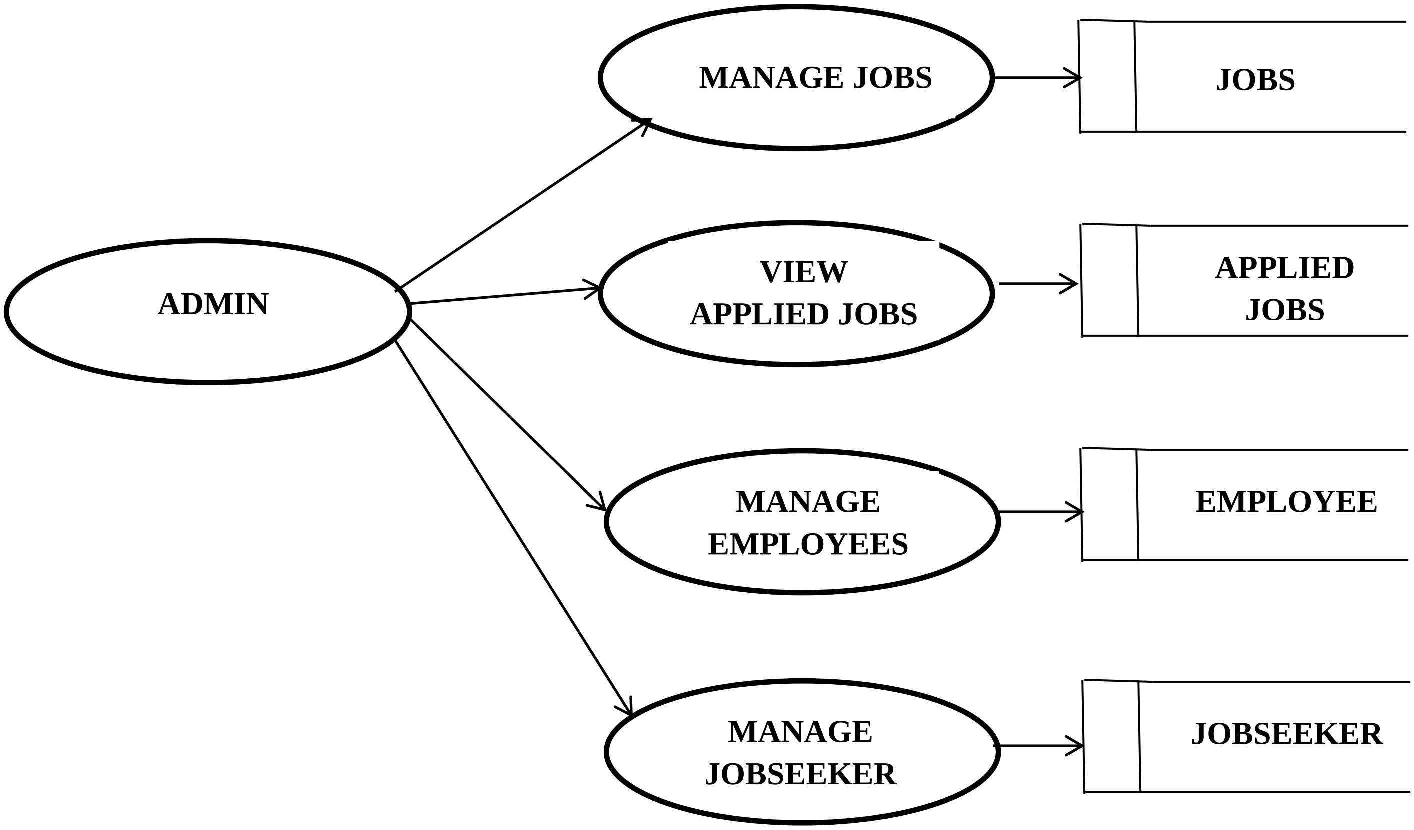


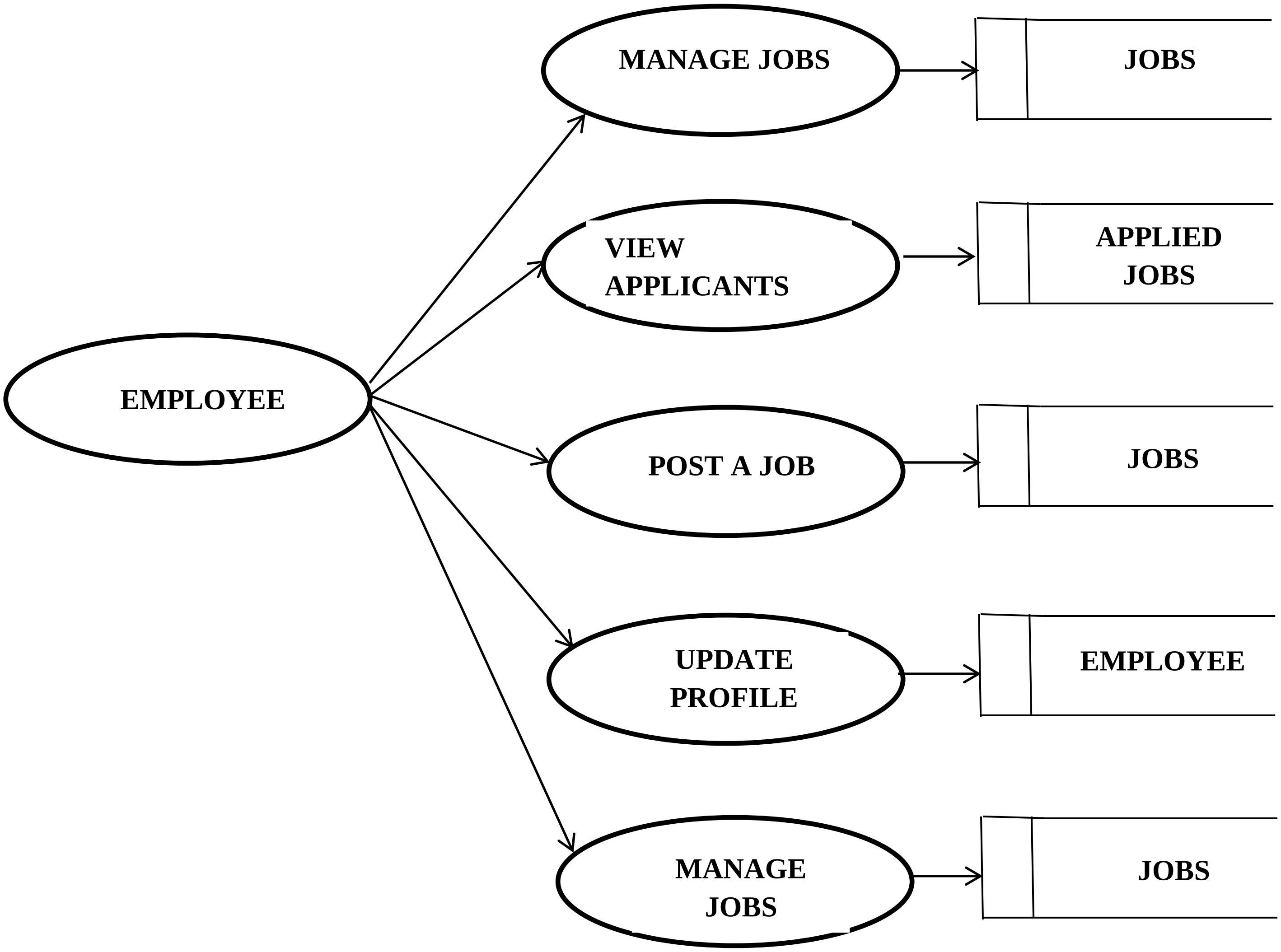
4.2 DATA FLOW DIAGRAM

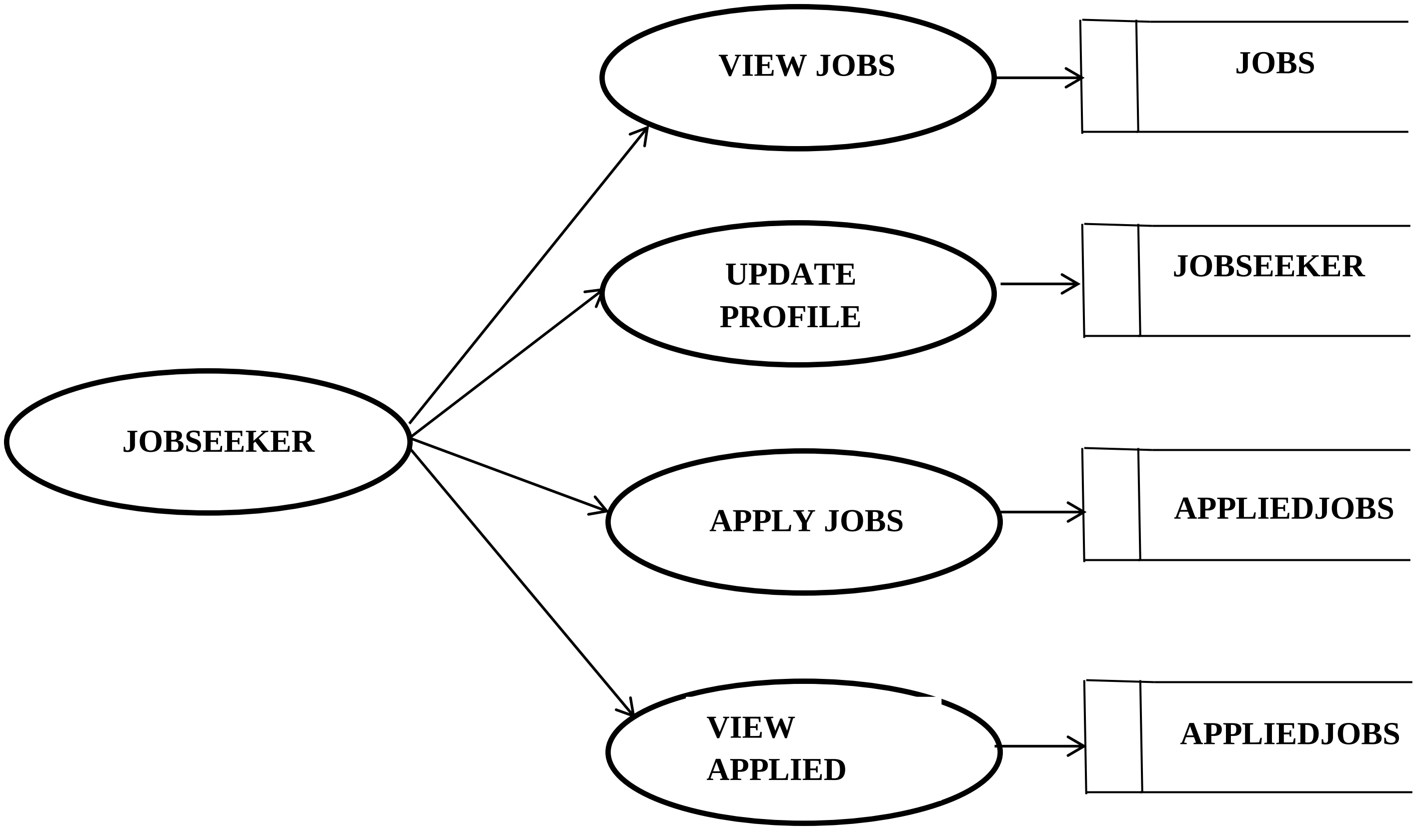
DFD Level 0



DFD Level 1




EMPLOYEE
MANAGE JOBS
JOBS
VIEW APPLICANTS
APPLIED JOBS
POST A JOB
JOBS
UPDATE PROFILE
EMPLOYEE
MANAGE JOBS
JOBS


VIEW JOBS
JOBS
UPDATE PROFILE
JOBSEEKER
JOBSEEKER
APPLY JOBS
APPLIEDJOBS
VIEW APPLIED
APPLIEDJOBS

4.3 DATA DICTIONARY

Collection Name: Admin

Purpose: To Store login details of Admin

S.No	Field Name	Type	Constraint	Description
1	_id	ObjectId	Primary Key	ID
2	admin_name	String	Not Null	Admin Name
3	email	String	Not Null	Admin Email
4	password	String	Not Null	Admin Password

Collection Name: Employees

Purpose: To Store details of Employees

S.No	Field Name	Type	Constraint	Description
1	_id	ObjectId	Primary Key	ID
2	username	String	Not Null	Name of employee
3	email	String	Not Null	Email of employee
4	password	String	Not Null	Employee password
5	company	String	Not Null	Employee’s company
6	position	String	Not Null	Employee’s position
7	company_logo	object	Not Null	Company Logo
8	mobile	number	Not Null	Employee’s number
9	skills	String	Not Null	Employee’s skills

Collection Name: Jobseekers

Purpose: To Store details of Jobseekers

S.No	Field Name	Type	Constraint	Description
1	_id	ObjectId	Primary Key	ID
2	name	String	Not Null	Jobseeker name
3	email	String	Not Null	Jobseeker email
4	password	Date	Not Null	Jobseeker password
5	resume	String	Not Null	Jobseeker resume
6	contactnumber	number	Not Null	Jobseeker number
7	Profilepic	Object	Not Null	Jobseeker picture
8	Aboutme	String	Not Null	About me

Collection Name: Jobs

Purpose: To Store details of Jobs

S.No	Field Name	Type	Constraint	Description
1	_id	ObjectId	Primary Key	ID
2	title	String	Not Null	Job title
3	desc	String	Not Null	Job description
4	company	String	Not Null	Company name
5	companylogo	Object	Not Null	Company logo
6	technology	String	Not Null	Required technology

7	position	number	Not Null	Available positions
8	type	String	Not Null	Job type
9	category	String	Not Null	Job category
10	location	String	Not Null	Job location
11	salary	number	Not Null	salary
12	contactemail	String	Not Null	Contact email
13	createdAt	Date	Not Null	Creation date
14	updatedAt	Date	Not Null	Upadation date

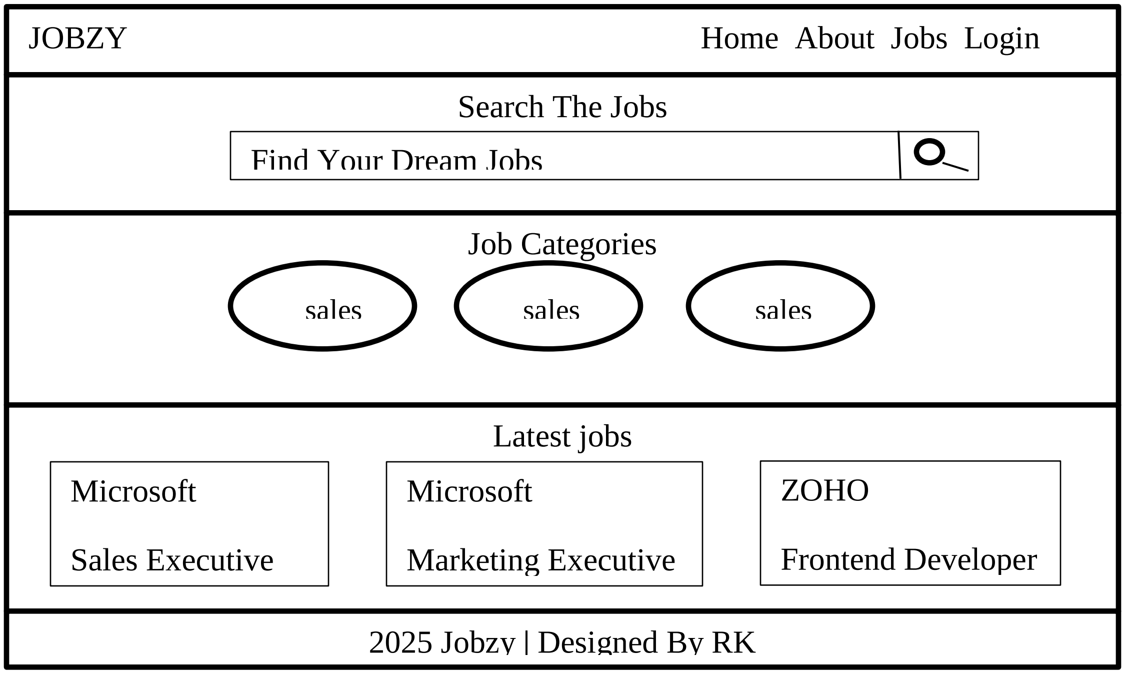
Collection Name: AppliedJobs

Purpose: To Store the details of Applied Jobs

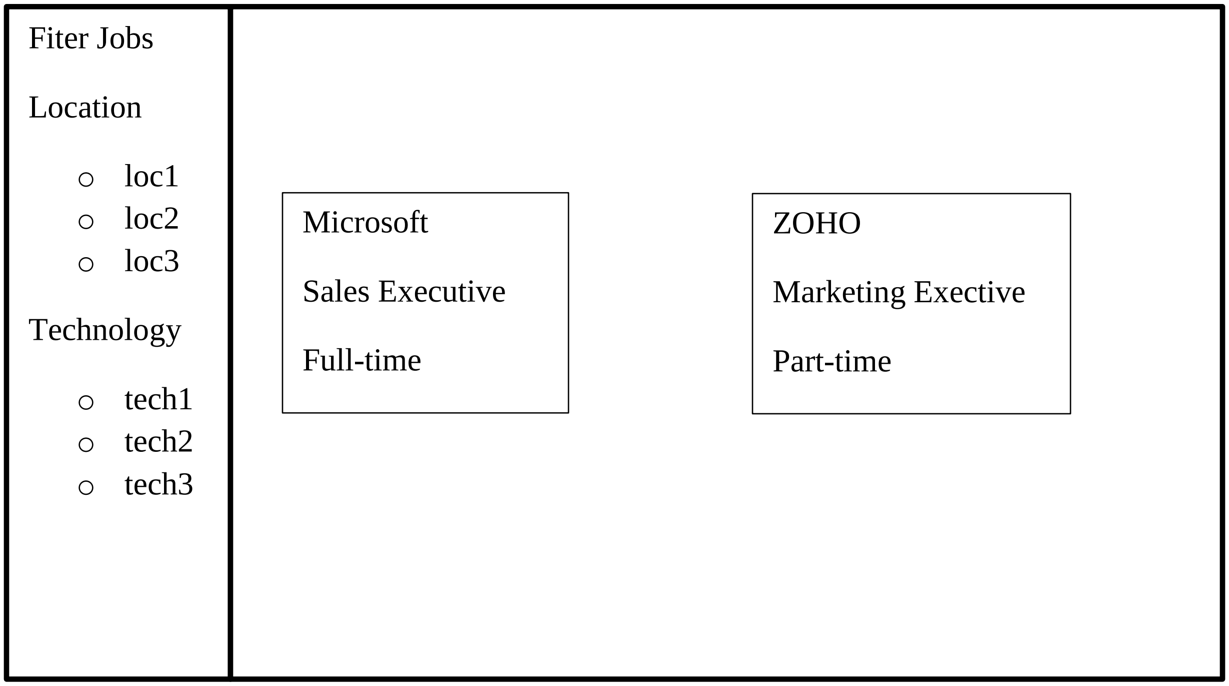
S.No	Field Name	Type	Constraint	Description
1	_id	ObjectId	Primary Key	ID
2	name	String	Not Null	Candidate name
3	resume	String	Not Null	Candidate resume
4	jobtitle	String	Not Null	Job title
5	company	String	Not Null	Company name
6	jscontactemail	String	Not Null	Jobseeker contact email
7	contactemail	String	Not Null	Employee contact email
8	applieddate	Date	Not Null	Job applied date

4.4 USER INTERFACE DESIGN

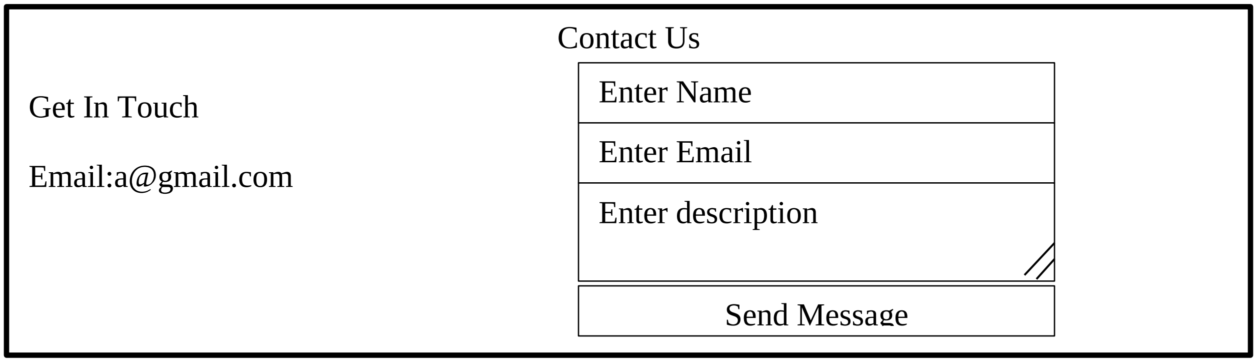
Home Page



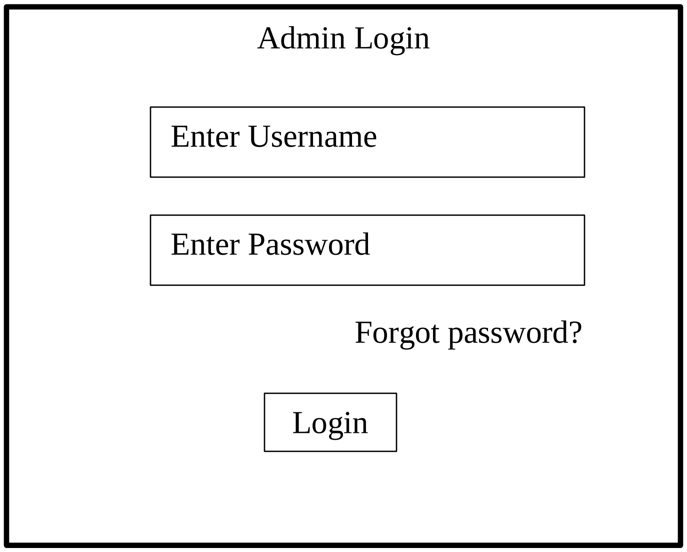
Jobs Page



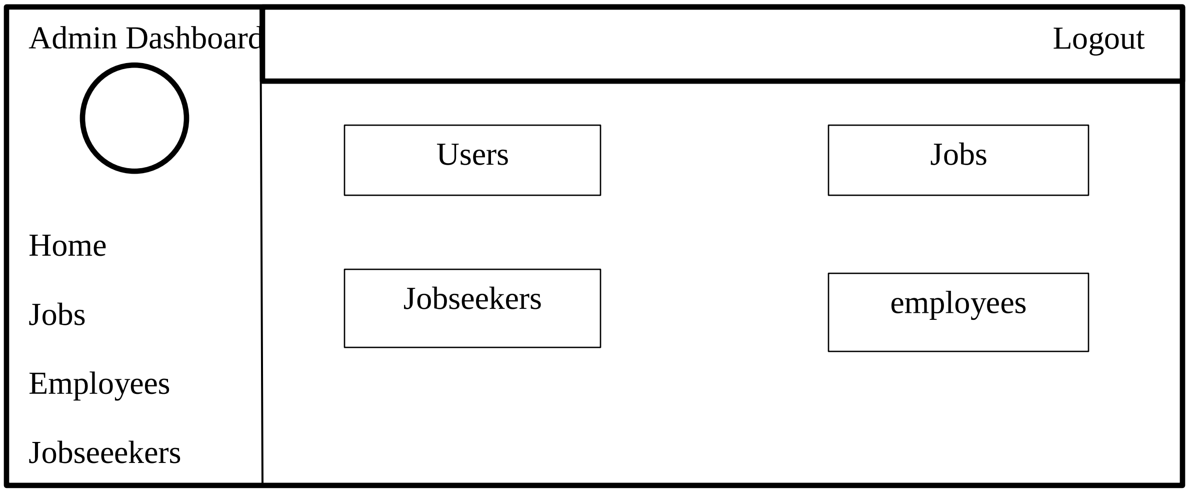
Contact Us



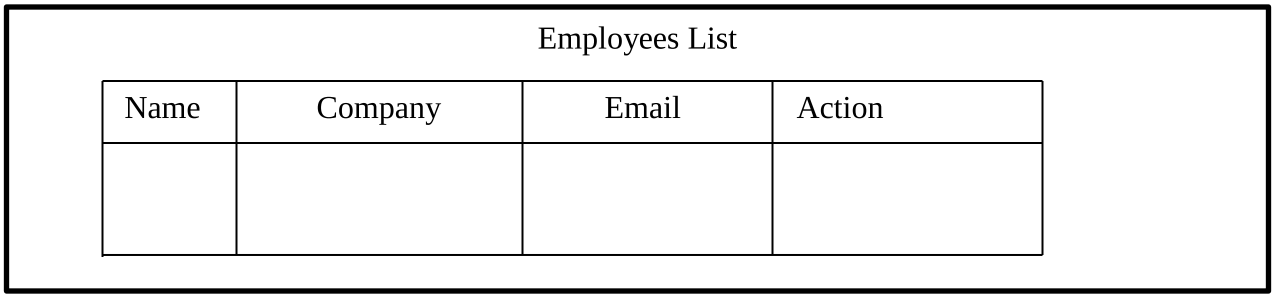
Admin Login



Admin Dashboard



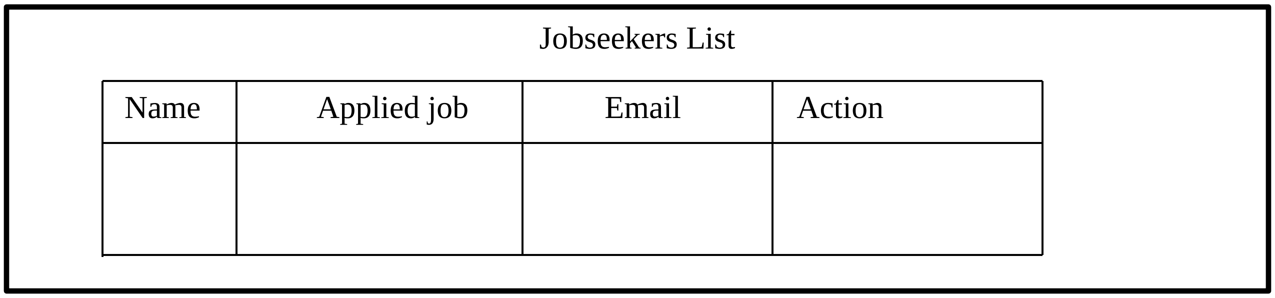
View Employees



Employees List

Name	Company	Email	Action

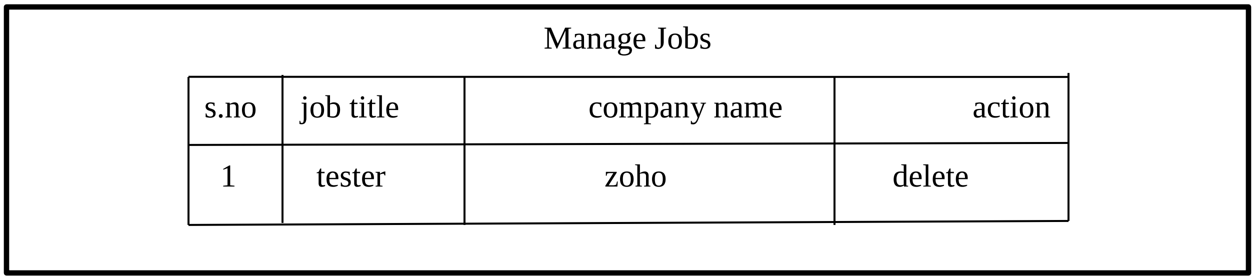
View Jobseekers



Jobseekers List

Name	Applied job	Email	Action

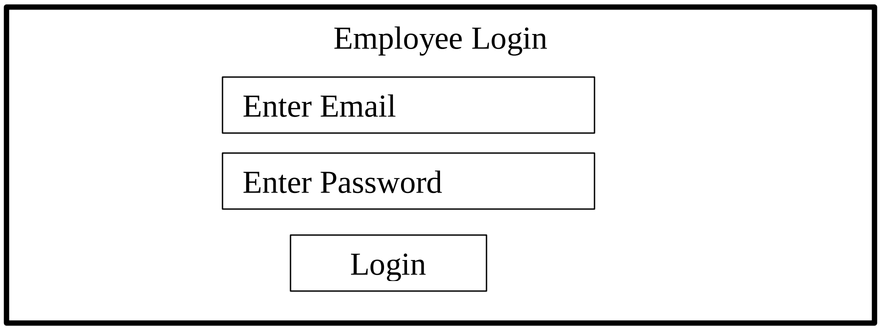
Manage Jobs



Manage Jobs

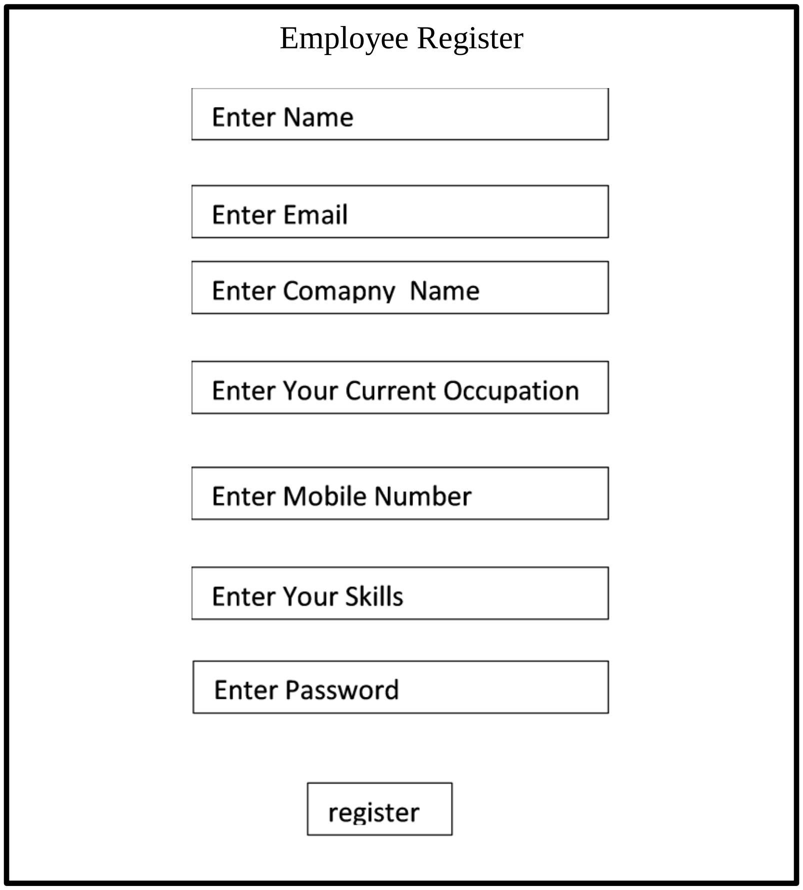
s.no	job title	company name	action
1	tester	zoho	delete

Employee Login

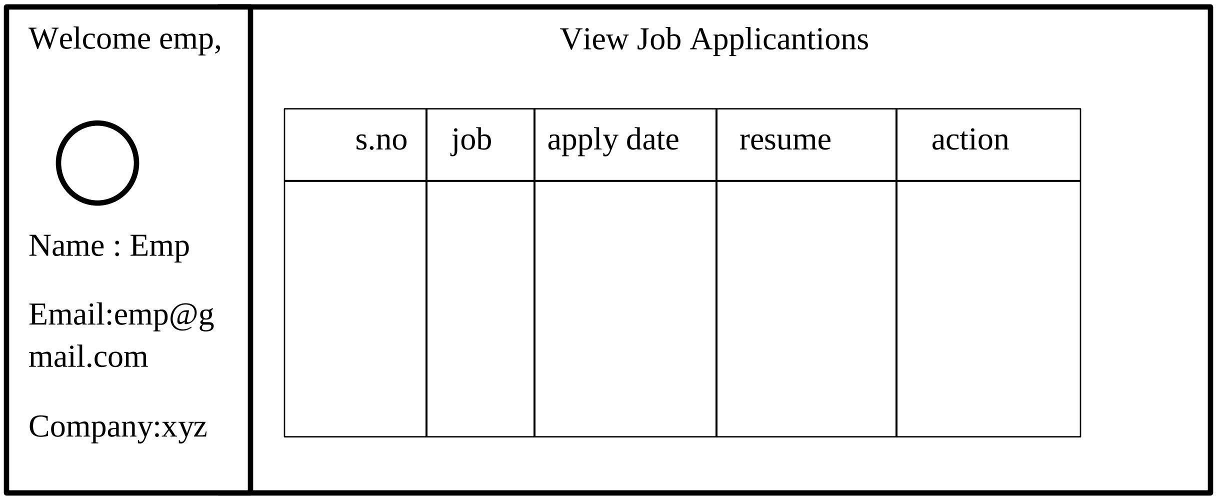


Employee Login

Employee Register



Employee dashboard



Welcome emp,

Name : Emp

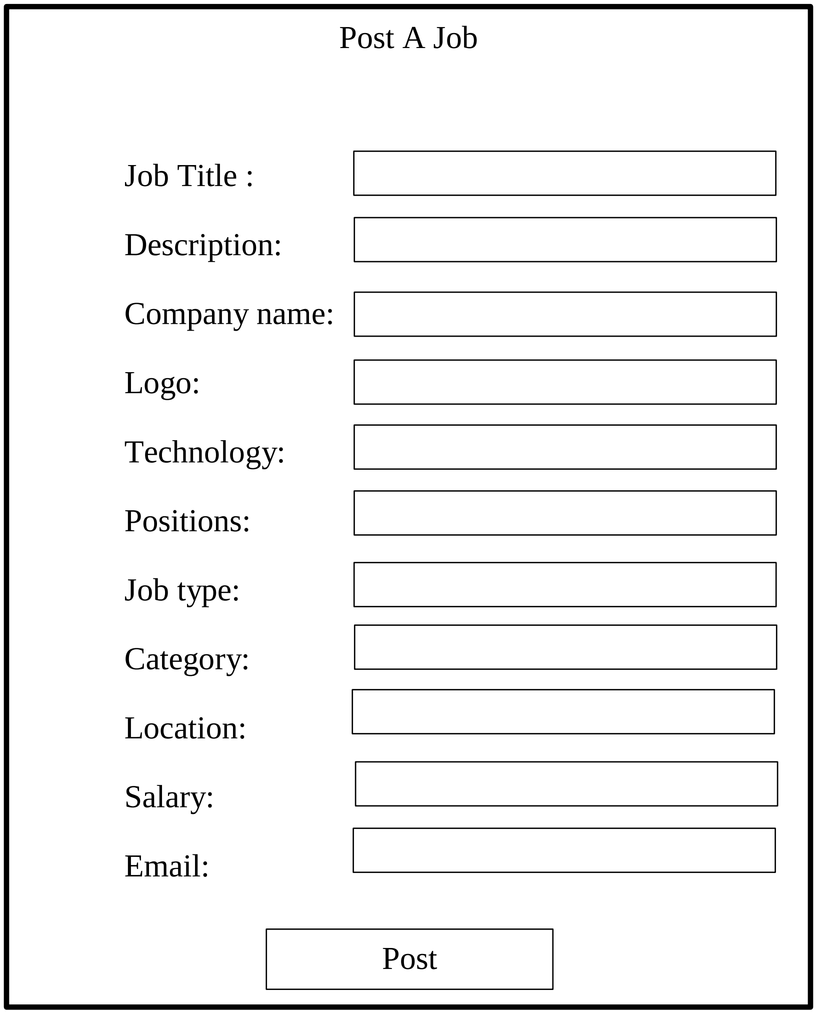
Email:emp@gmail.com

Company:xyz

View Job Applicantions

s.no	job	apply date	resume	action

Post A Job



Post A Job

Job Title :

Description:

Company name:

Logo:

Technology:

Positions:

Job type:

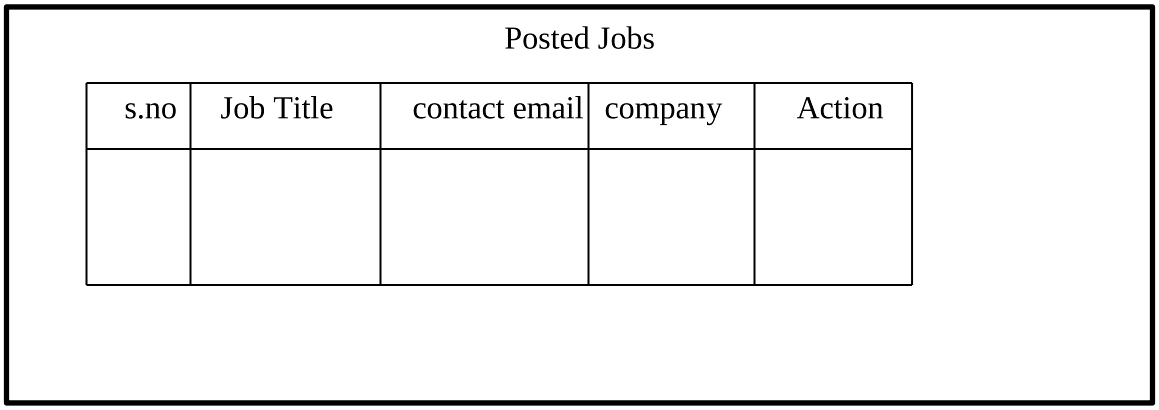
Category:

Location:

Salary:

Email:

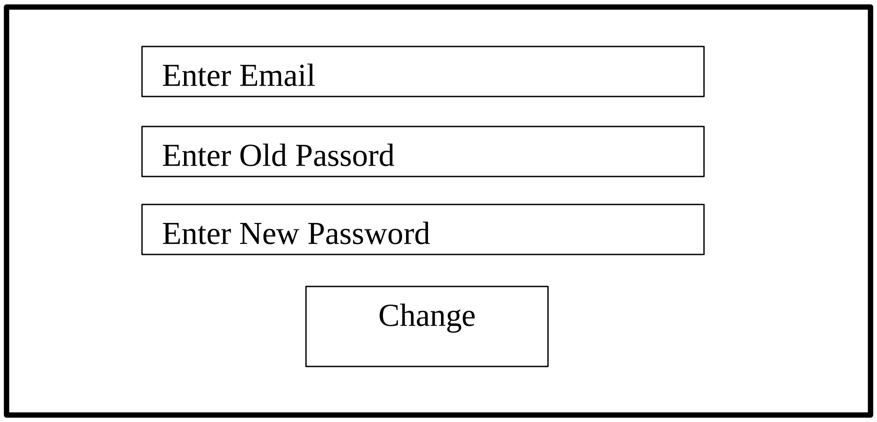
Posted Jobs



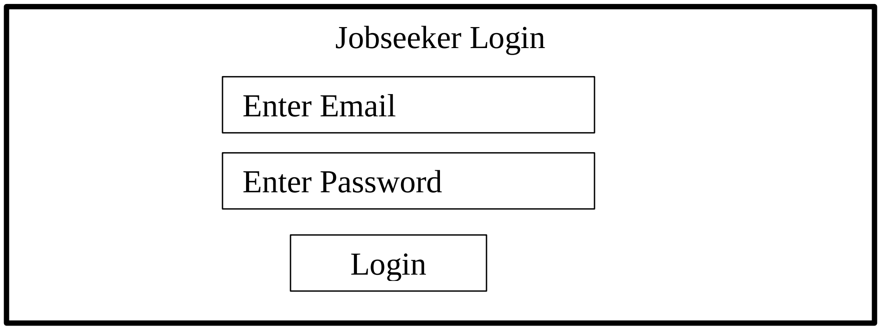
Posted Jobs

s.no	Job Title	contact email	company	Action

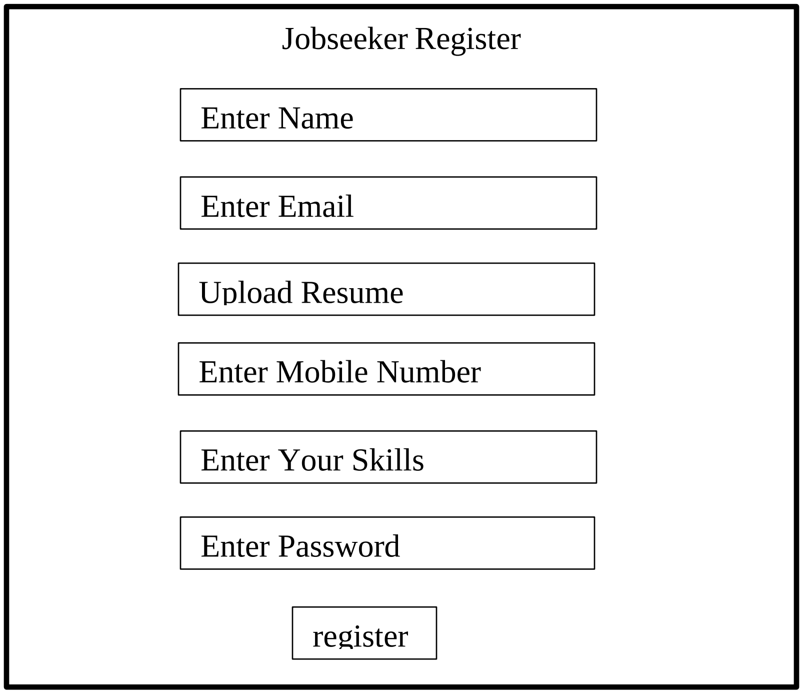
Change Password



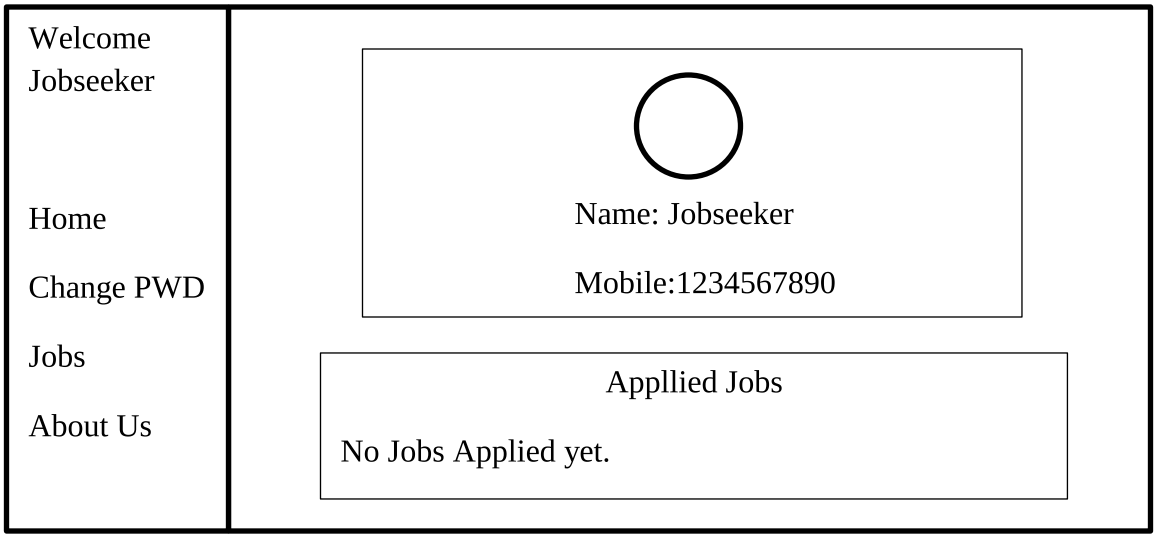
Jobseeker Login



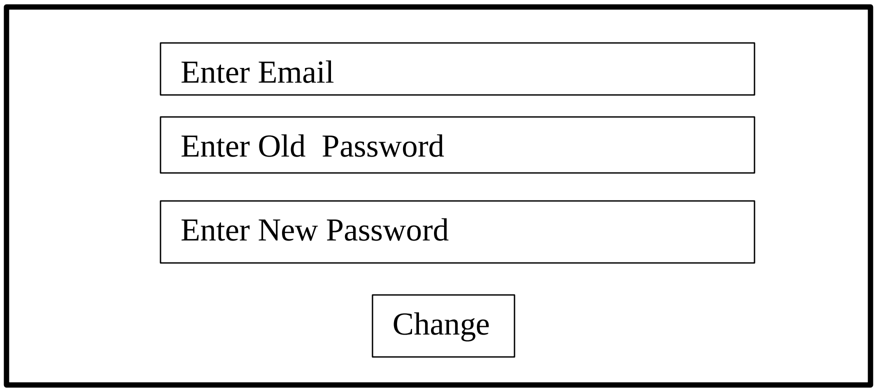
JobSeeker Register



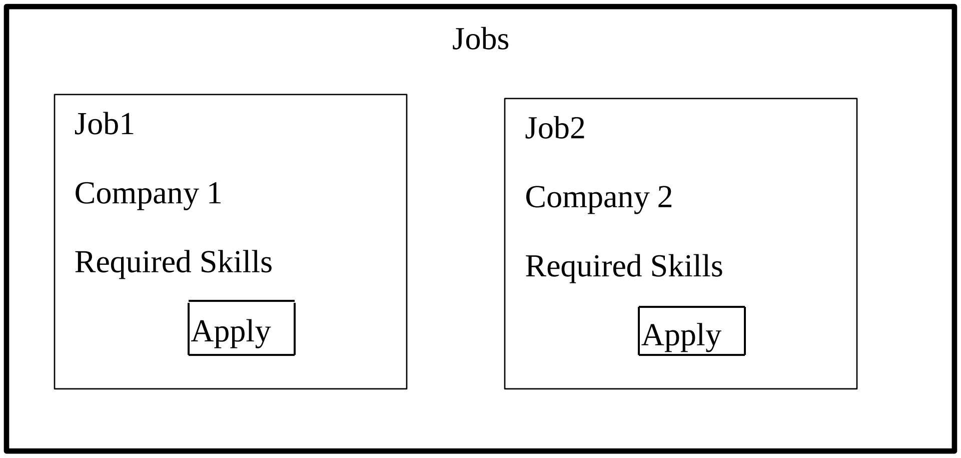
Jobseeker Dashboard



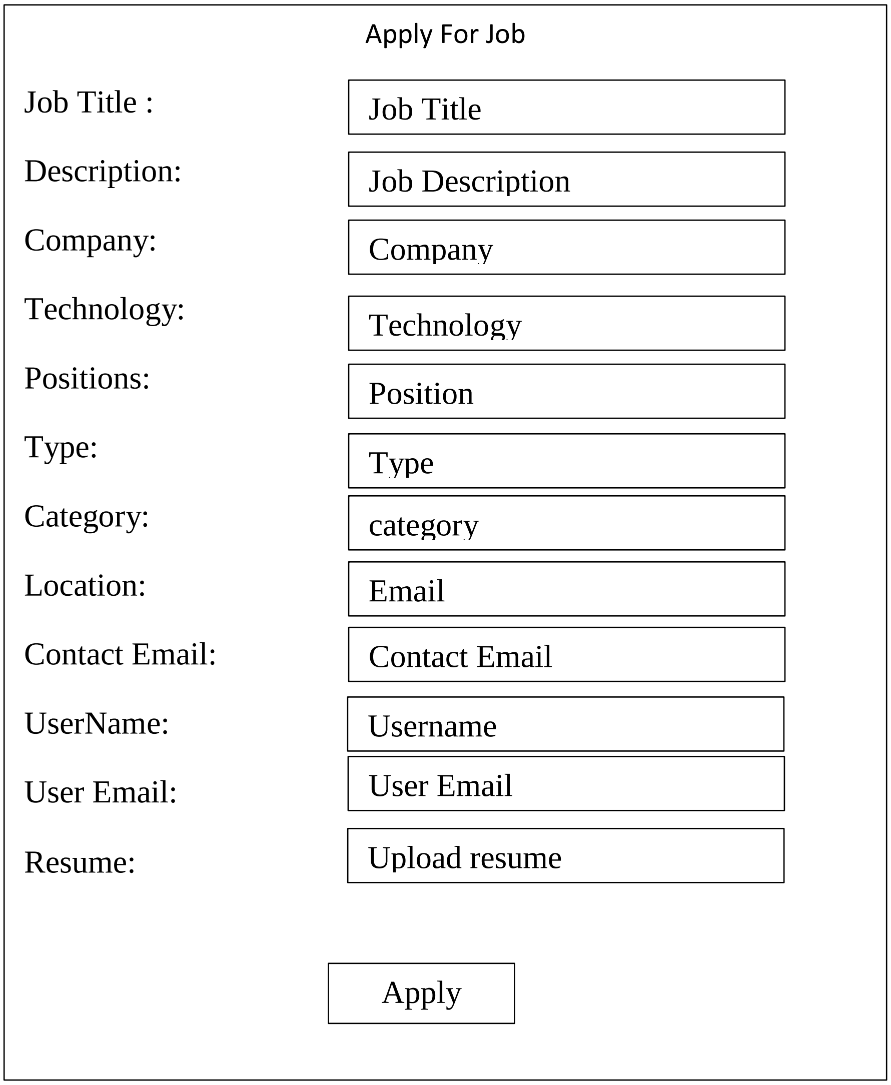
Change Password



View Jobs



Apply For Jobs



Apply For Job

Job Title :	Job Title
Description:	Job Description
Company:	Company
Technology:	Technology
Positions:	Position
Type:	Type
Category:	category
Location:	Email
Contact Email:	Contact Email
UserName:	Username
User Email:	User Email
Resume:	Upload resume

4.5 NORMALIZATION

Definition

Normalization is the process of organizing data in a database to minimize redundancy and avoid anomalies during **insertion**, **update**, or **deletion** operations.In the context of our **Online Job Portal System (MERN Stack)**, normalization ensures that data like **jobseeker details**, **employer details**, **job postings**, and **applications** are stored in separate, well-structured collections and linked using relationships (in MongoDB, usually by storing reference IDs).By applying normalization, we divide large, unorganized collections into smaller, related collections — such as **users**, **jobs**, **applications**, **companies**, and **skills**. This avoids data duplication and makes updates more efficient.

First Normal Form (1NF)

Conditions for 1NF:

1. **Atomic values only** – Each field must hold a single value (e.g., in the jobs collection, location should store one city, not multiple cities in one field).
2. **Same data type in a field** – All values in a field should belong to the same type (e.g., salary should always be stored as a Number, not as a mix of text and numbers).
3. **Unique field names** – Each attribute should have a unique name.
4. **Order of records does not matter** – The sequence of records should not affect meaning.

Example in project:
Instead of storing "technologies": "React, Node.js, MongoDB" in one string field, create an **array** like "technologies": ["React", "Node.js", "MongoDB"] or use a **separate technologies collection** linked to jobs.

Second Normal Form (2NF)

Conditions for 2NF:

1. Must be in **1NF**.
2. **No Partial Dependency** – Non-key attributes should depend on the **whole primary key**, not part of it.

Example in project:
If the applications collection has a composite key (jobId, userId), then fields like jobTitle should not depend only on jobId.
Instead, store only jobId in applications and fetch job details from the jobs collection when needed.

Third Normal Form (3NF)

Conditions for 3NF:

1. Must be in **2NF**.
2. **No Transitive Dependency** – Non-key attributes should not depend on other non-key attributes.

Example in project:

In the users collection, if city determines postalCode, then this is a transitive dependency.
To avoid it, keep city and postalCode independent, or move location data into a separate addresses subdocument.

Boyce–Codd Normal Form (BCNF)

Conditions for BCNF:

1. Must be in **3NF**.
2. For every functional dependency (X → Y), **X** must be a **superkey**.

Example in project:

If in the savedJobs collection, savedJobId determines both userId and jobId, then savedJobId should be the superkey to avoid anomalies.