

VIETNAMESE GERMAN UNIVERISTY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEER



IT Jobs Website Application Report

Student 1: Nguyen Dai Minh - 10421037
Student 2: Nguyen Khoi Nguyen - 10421097
Student 3: Pham Nguyen Quy Nam - 10421093
Student 4: Pham Tuan Kiet - 10421033

August 12, 2024

Content

1	Introduction	4
2	Requirements Gathering	6
2.1	Use Case Diagram	6
2.2	Use Case Analysis	7
2.3	Sequence Diagram	19
2.3.1	Search Jobs	19
2.3.2	Register as Employer	20
2.3.3	Register as Jobseeker	21
2.3.4	Login to the website	22
2.3.5	Edit profile	23
2.3.6	Search Jobs for registered user	24
2.3.7	View jobs application details	25
2.3.8	Job application	25
2.3.9	Withdraw job application	26
2.3.10	Post jobs	26
2.3.11	Delete jobs	27
2.3.12	View jobs	27
2.3.13	View applicants info	28
2.4	Software Architecture	28
2.5	Deployment Diagram	29
2.6	Entity-Relationship Diagram	29
2.7	User Interface Mock-up	30
2.7.1	Main Page	30
2.7.2	Login Page	31
2.7.3	Jobseeker Registration Page	32
2.7.4	Employer Registration Page	33
2.7.5	Employer Page	34
2.7.6	Jobseeker Page	34
3	Back-end Programming	35
3.1	Login	35
3.2	Register	36
3.2.1	Register for Employer	36
3.2.2	Register as Jobseeker	37
3.3	Manage applicants	39
3.4	Manage jobs	41
3.5	Manage applications	43
3.6	Manage employers	44

4 Front-end Programming	47
4.1 Main Page	47
4.2 Login Page	48
4.3 Jobseeker Registration Page	49
4.4 Employer Registration Page	50
4.5 Jobseeker Profile	51
4.6 Employer Profile	54
5 Back-end Testing	57
5.1 Search Jobs	57
5.1.1 Test case 1: Keyword search successful	57
5.1.2 Test case 2: Keyword search return null	58
5.2 Register as Employer	58
5.2.1 Test case 1: Register Successful	58
5.2.2 Test case 2: Register fail with missing information	59
5.3 Register as Jobseeker	60
5.3.1 Test case 1: Register Successful	60
5.3.2 Test case 2: Register fail with missing information	61
5.4 Edit profile	61
5.4.1 Test case 1: Edit Profile Jobseeker	61
5.4.2 Test case 2: Edit Profile Employer	62
5.5 Job application	62
5.5.1 Test case 1: Job application is already in	62
5.5.2 Test case 2: Job application is new	63
5.6 Post jobs	64
5.6.1 Test case 1: Post Jobs successful	64
5.6.2 Test case 2: Post Jobs unsuccessful	64
5.7 Delete jobs	64
5.7.1 Test case 1: Delete successful	64
5.7.2 Test case 2: Delete fail	65
6 Front-end Testing	66
6.1 Main Page	66
6.2 Login Page	66
6.3 Jobseeker Registration Page	66
6.4 Employer Registration Page	67
6.5 Employer Page	68
6.6 Jobseeker Page	69
7 Security	70

8 Change Management	72
8.1 Updated Use Case Diagram	73
8.2 Updated Use Case Analysis	74
8.3 Updated Sequence Diagram	75
8.3.1 Updated ER Diagram	76
8.4 Updated UI	76
8.4.1 Updated Employer Page	76
8.4.2 Updated Jobseeker Page	77
8.5 Updated Notification Page	77
8.6 Revision History	78

1 Introduction

Application: Online IT job seeking application

The journey from graduation to securing one's first job in the IT industry can be fraught with obstacles. Several factors contribute to the high rate of unemployment among IT freshers, including:

1. Decreasing Demand for Entry-Level Positions: In an increasingly competitive job market, companies may prioritize hiring candidates with prior experience or specialized skills, leaving entry-level positions in high demand but low supply.
2. Focus on Senior and Junior Positions: While there may be ample opportunities for senior and junior positions within the IT industry, fresh graduates often struggle to compete with candidates who possess more experience or advanced qualifications.
3. Difficulty in Establishing Contact with Employers: Many IT freshers face challenges in effectively presenting themselves to potential employers, whether due to lack of networking opportunities, inadequate resume or interview preparation, or limited access to job postings.

ITJ Ltd recognizes the pressing need to address these challenges and empower both job seekers and employers in the IT industry. Through its comprehensive job seeking services, ITJ Ltd offers innovative solutions to facilitate meaningful connections and streamline the recruitment process.

Employers partnering with ITJ Ltd can expect a range of benefits, including:

- Job Posting Services: Employers can post detailed job listings, including information about the position, required qualifications, and desired skills, ensuring that job seekers have access to relevant and up-to-date job opportunities.
- Applicant Management Tools: ITJ Ltd provides employers with robust applicant management tools, allowing them to view applicant profiles, track application statuses, and make informed hiring decisions. Employers can easily accept or reject applicants, schedule interviews, and communicate with candidates directly through the platform.
- Profile Management: Employers have the flexibility to update their profiles, showcasing their company culture, values, and job offerings. This allows job seekers to gain insights into potential employers and make informed decisions about their applications.

Job seekers utilizing ITJ Ltd's services can expect comprehensive support throughout their job search journey, including:

- Job Search and Application: Job seekers have access to a vast database of job listings, searchable by keywords, location, and industry. They can apply to jobs directly through the platform, streamlining the application process and increasing their visibility to potential employers.
- Profile Enhancement: Job seekers can create personalized profiles, highlighting their skills, education, and experience. They have the option to upload resumes, cover letters, and other relevant documents, allowing them to present a comprehensive overview of their qualifications to potential employers.
- Job Details and Updates: ITJ Ltd provides job seekers with detailed job descriptions,

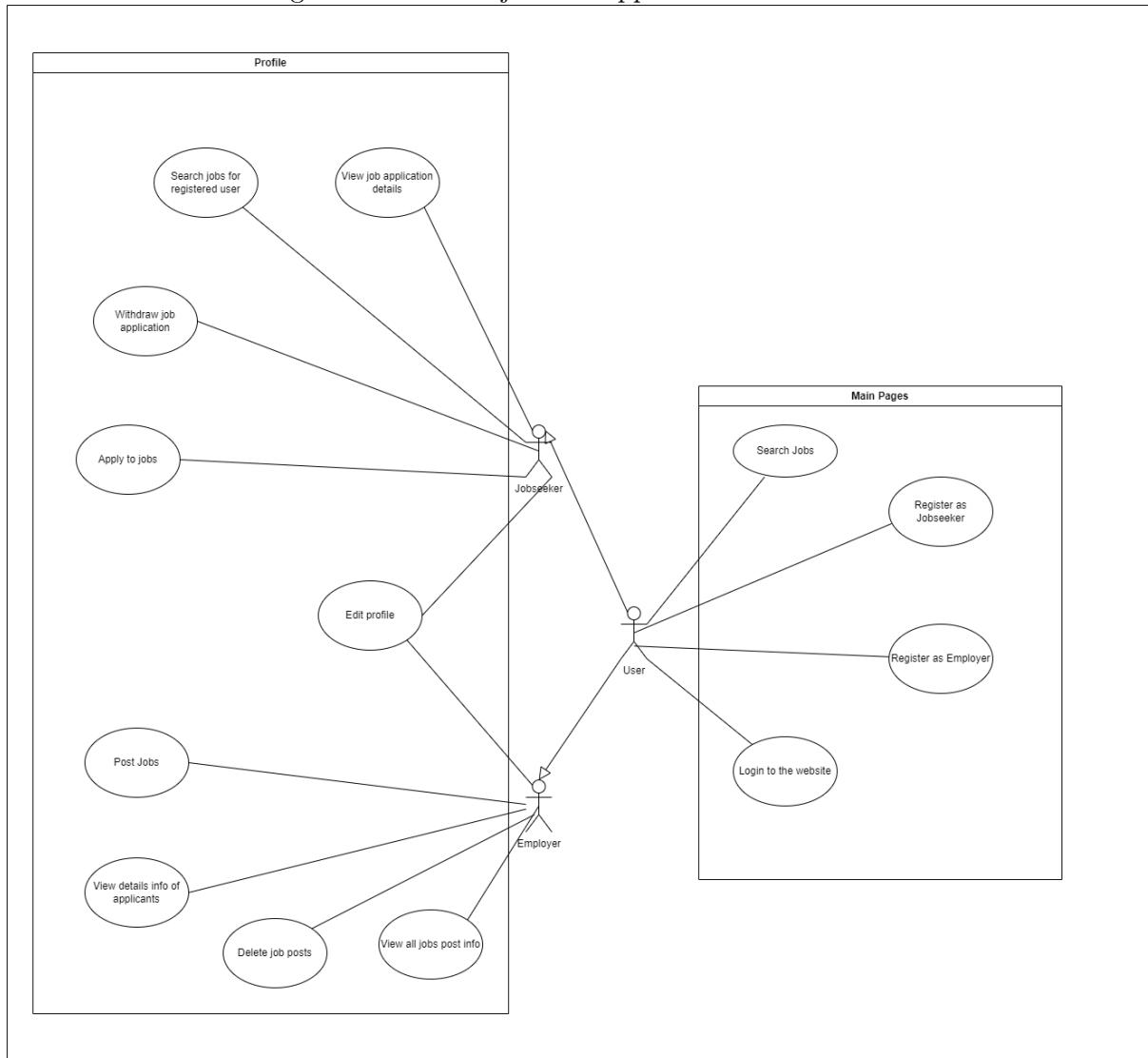
including information about job responsibilities, qualifications, and application deadlines. Job seekers receive notifications about new job postings and updates, ensuring they stay informed about relevant opportunities. By bridging the gap between job seekers and employers, ITJ Ltd's job seeking services have the potential to transform the IT job market and address the challenges faced by IT freshers. Some of the key benefits and impacts of these services include:

1. Increased Access to Job Opportunities: IT freshers gain access to a wider range of job opportunities, including entry-level positions and internships, through ITJ Ltd's platform. This increased accessibility enhances their chances of securing meaningful employment opportunities within the IT industry.
2. Streamlined Recruitment Process: Employers benefit from a streamlined recruitment process, facilitated by ITJ Ltd's applicant management tools. By providing a centralized platform for posting jobs, reviewing applications, and communicating with candidates, employers can save time and resources while ensuring they attract and hire the best talent for their organizations.
3. Improved Candidate-Employer Matching: ITJ Ltd's platform utilizes advanced algorithms and personalized profiles to match candidates with relevant job opportunities. This improved matching process increases the likelihood of successful hires and fosters long-term relationships between employers and employees.
4. Enhanced Professional Development: Job seekers utilizing ITJ Ltd's services gain access to resources and support to enhance their professional development. From resume writing tips to interview preparation workshops, ITJ Ltd empowers job seekers to present themselves effectively to potential employers and advance their careers within the IT industry.

2 Requirements Gathering

2.1 Use Case Diagram

Here is our use case diagrams for our IT job web application:



2.2 Use Case Analysis

Name	Search jobs
Description	This use case is for user to type a keyword into the search bar to find the jobs that match their search words.
Data Return	List of strings
Trigger	Users must enter a search term and press Search
Pre-conditions	Users must be on the main page of the web application
Post-conditions	A list of jobs matching the search terms will appear on the screen or none in case there are no jobs matching the search terms.
Basic Flow	<ol style="list-style-type: none"> 1. The user enter the keyword in the search bar 2. The application database will compare the keyword with jobs.title or employer.ename and output result to the front end 3. The web app will output a list of jobs matching the user search terms 4. User can click on login to login or create an account
Alternative Flow	<ol style="list-style-type: none"> 1. At step 3, if there are no jobs matching, the front end will display sorry, there are no jobs matching the search terms

Name	Register as Employer
Description	This use case is for user to register as an employer.
Data Return	None
Trigger	Users must click on become an Employer
Pre-conditions	Users must be on the employer registration page of the web application
Post-conditions	User will successfully register as an employer and transfer back to the main page of the application.
Basic Flow	<ol style="list-style-type: none"> 1. The user navigate to the employer registration page of the web application 2. A form containing company information, login information, contact person, industry, location as well as brief description about the company will appear for the user to fill in and choose (employer.e_name, employer.executive, employer.address, employer.Industry, employer.phone, employer.location) 3. User can click on register to finish the registration 4. The database will receive information and store it in the employer table 5. Upon successful registration, the user will be redirect to the main page
Alternative Flow	<ol style="list-style-type: none"> 1. At step 2, if the information entered is not in the correct format users will be warned.

Name	Register as Jobseeker
Description	This use case is for user to register as a job seeker.
Data Return	None
Trigger	Users must click on become a Jobseeker
Pre-conditions	Users must be on the job seeker registration page of the web application
Post-conditions	User will successfully register as a job seeker and transfer back to the main page of the application.
Basic Flow	<ol style="list-style-type: none"> 1. The user navigate to the jobseeker registration page of the web application 2. A form containing personal information such as jobseeker.log_id, jobseeker.name, jobseeker.phone, jobseeker.location, jobseeker.master_edu, jobseeker.skills, jobseeker.basic_edu, jobseeker.experience 3. User can click on register to finish the registration 4. The database will receive information and store it in the jobseeker table 5. Upon successful registration, the user will be redirect to the main page
Alternative Flow	<ol style="list-style-type: none"> 1. At step 2, if the information entered is not in the correct format users will be warned.

Name	Login to the website
Description	This use case is for user to login to the website.
Data Return	None
Trigger	Users must click on login
Pre-conditions	Users must be on the login page of the web application
Post-conditions	User will successfully login and redirect to account profile.
Basic Flow	<ol style="list-style-type: none"> 1. The user navigate to the login page of the web application 2. User will enter their login credentials (login.email, login.password) 3. The database will check if the credentials entered is correct 4. User will be redirected to the account profile of either jobseeker or employer
Alternative Flow	<ol style="list-style-type: none"> 1. At step 3, if the credentials entered is not correct, users will have to enter the credentials again

Name	Edit profile
Description	This use case involves updating the basic information of both employers and jobseekers in the database.
Data Return	None
Trigger	Users must click on "Edit profile" in the menu task bar.
Pre-conditions	Users must register as either employers or jobseekers to updating the information in the database
Post-conditions	Employers or jobseekers will successfully update their account information after clicking on the "Confirm" button.
Basic Flow	<ol style="list-style-type: none"> 1. The employer or jobseeker click on the "update" button on the profile UI. 2. The employer or jobseeker will able to see the option which can be modified modify in the profile (employer.e_name, employer.industry, employer.address, employer.executive, employer.phone, employer.e_type, jobseeker.log_id, jobseeker.name, jobseeker.phone, jobseeker.location, jobseeker.master_edu, jobseeker.skills, jobseeker.basic_edu, jobseeker.experience). 3. The employer or jobseeker can click confirm to finish edit profile.

Name	Search job for registered user
Description	This use case describes the scenario when the jobseeker looks for a job.
Data Return	List of objects containing strings
Trigger	User must type the job name on the search bar and hit on "Search" button or user can click on "Advanced search"
Pre-conditions	Users must register as jobseeker and is logged in.
Post-conditions	None
Basic Flow	<ol style="list-style-type: none"> 1. The user searches for a specific job based on what they type into the search bar. 2. The app retrieves the jobs based on the jobseeker's input (employer.e_name, employer.industry, employer.e_type) 3. User will be presented with a list of jobs matching the input
Alternative Flow	<p>On step 2, user can use advance search.</p> <ol style="list-style-type: none"> 1. The user searches for a specific job based on industries, skills needed, etc. (employer.e_name, employer.industry, employer.e_type, job.title, job.Industry). 2. The app filters out jobs that match jobseeker's desire.

Name	View job application details
Description	This use case describes the scenario when the jobseeker want to view their job application details.
Data Return	Pop up objects containing strings
Trigger	More than one application of jobseeker exist.
Pre-conditions	Users must register as jobseeker and is logged in.
Post-conditions	Application details will pop up as a form
Basic Flow	<ol style="list-style-type: none"> 1. The jobseeker view the list of jobs applied. Jobs will include job.title, job.industry, job.experience, job.basic_pay, job.location, job.job_desc. 2. Jobseeker can click on the job to see more details 3. The job details will pop up as well as application status

Name	Job application
Description	This use case describes the scenario when the jobseeker want to apply for a job.
Data Return	None
Trigger	The jobseeker press apply in the job page
Pre-conditions	Jobseeker has already found his or her desired job.
Post-conditions	The job application is added to the query.
Basic Flow	<ol style="list-style-type: none"> 1. The user found a job 2. The user click to see the job description. 3. The user click on apply button. 4. The system will add the job application.

Name	Withdraw job application
Description	This use case describes the scenario when the jobseeker want to withdraw their job application .
Data Return	None
Trigger	The jobseeker press withdraw in job application details.
Pre-conditions	One application or more have to exist.
Post-conditions	The job application is removed.
Basic Flow	<ol style="list-style-type: none"> 1. The user click to see application details. Details will include application.status, application.date_applied, application.apply_id 2. The user click on withdraw 3. The system will remove the job application

Name	Post Jobs
Description	Employer posts a job vacancy.
Data return	None
Trigger	Employer selects the option to post a job
Pre-conditions	Employer is logged into their account
Post-conditions	Job is successfully posted on the web app
Basic Flow	<ol style="list-style-type: none"> 1. Employer selects the option to post a job. 2. Employer fills out the job details such as job title, description, requirements, etc. (job.title, job.location, job.vac_no, job.basic_pay, job.experience, job_Industry) 3. Employer submits the job posting. 4. System validates the job posting and adds it to the database.
Alternate Flow	<ol style="list-style-type: none"> 1. At step 4, if the required fields are not filled, the system prompts the employer to complete all required fields before submitting the job posting.

Name	Delete Jobs Post
Description	Employer deletes a job posting.
Data return	None
Trigger	Employer selects the option to manage job postings
Pre-conditions	Employer is logged into their account
Post-conditions	Job posting is removed from the system
Basic Flow	<ol style="list-style-type: none"> 1. Employer selects the option to manage job postings. 2. Employer selects the job posting they want to delete. 3. Employer confirms the deletion. 4. System removes the job posting from the database.

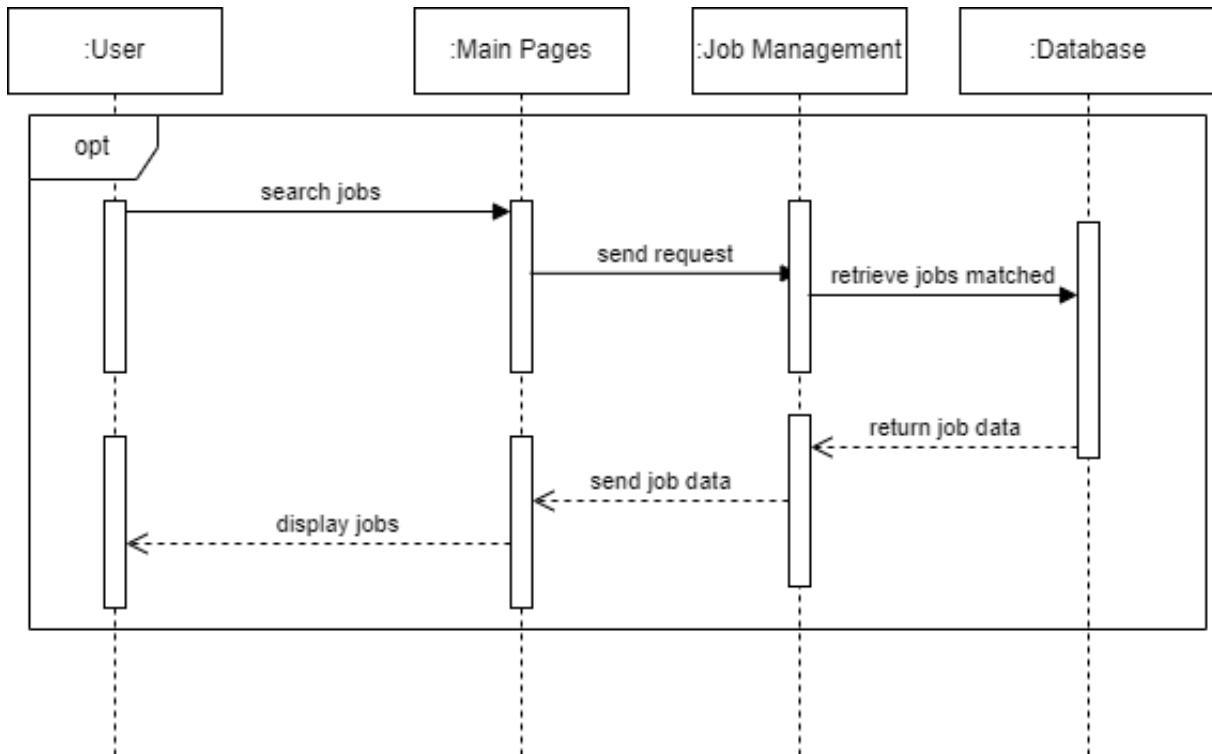
Name	View Jobs
Description	Employer views the list of jobs they have posted.
Data return	Pop up objects containing strings
Trigger	Employer navigates to the section to view posted jobs
Pre-conditions	Employer is logged into their account
Post-conditions	Employer sees the list of jobs they have posted
Basic Flow	<ol style="list-style-type: none"> 1. Employer navigates to the section to view posted jobs. 2. System retrieves and displays the list of jobs posted by the employer. 3. Employer chooses to view a specific job. 4. System retrieves and displays the specified job details.

Name	View Applicants Info
Description	Employer views information about applicants who applied to their job postings
Data return	Pop up objects containing strings
Trigger	Employer selects a specific job posting
Pre-conditions	Employer is logged into their account
Post-conditions	Employer sees the information about applicants
Basic Flow	<ol style="list-style-type: none"> 1. Employer selects a specific job posting. Job posting include information such as (job.title, job.location, job.vac_no, job.basic_pay, job.experience, job_Industry) 2. System retrieves and displays the list of applicants for that job posting. 3. Employer can view individual applicant details.

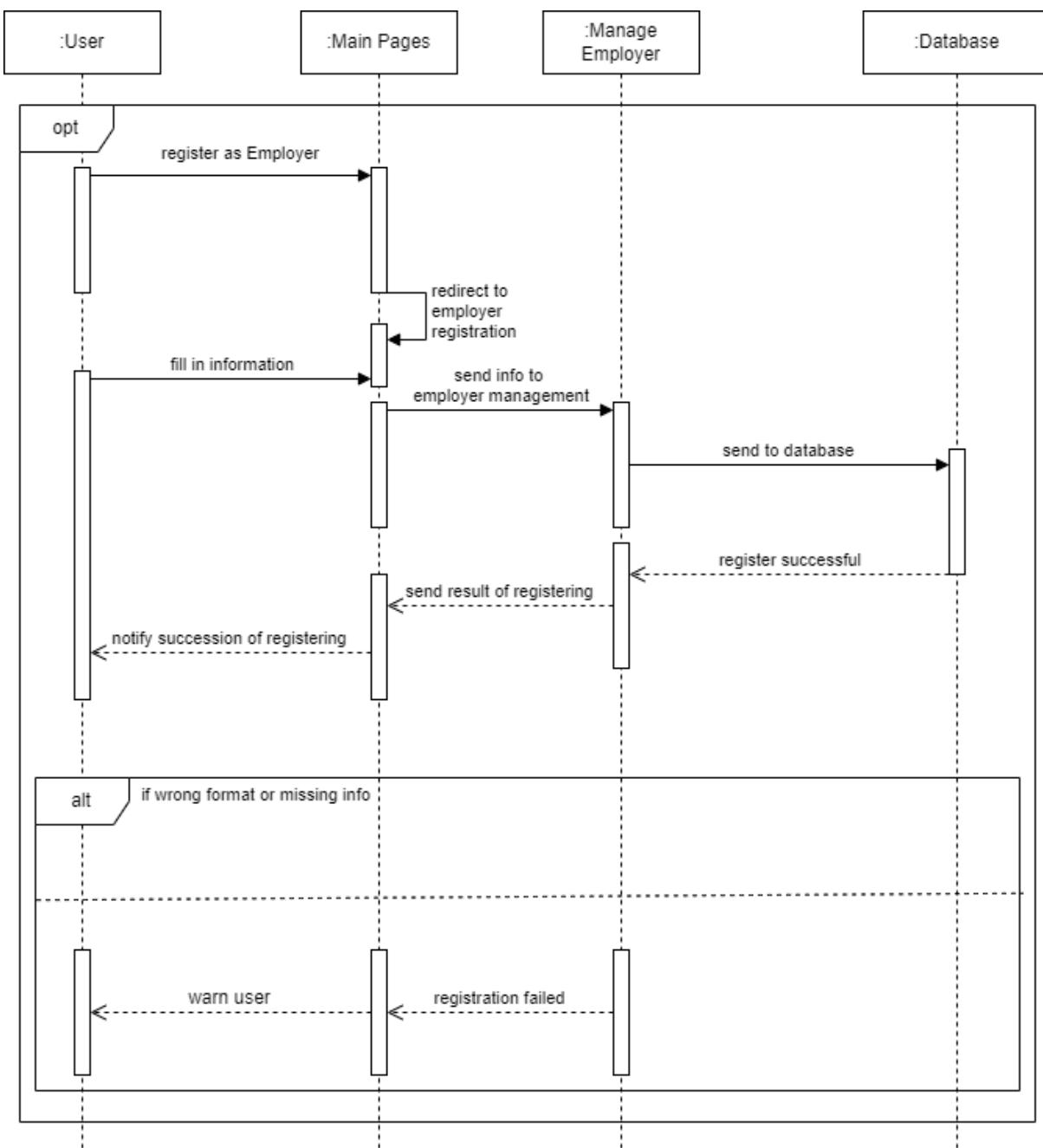
2.3 Sequence Diagram

In this section, we will present sequence diagrams respective of the aforementioned use case tables

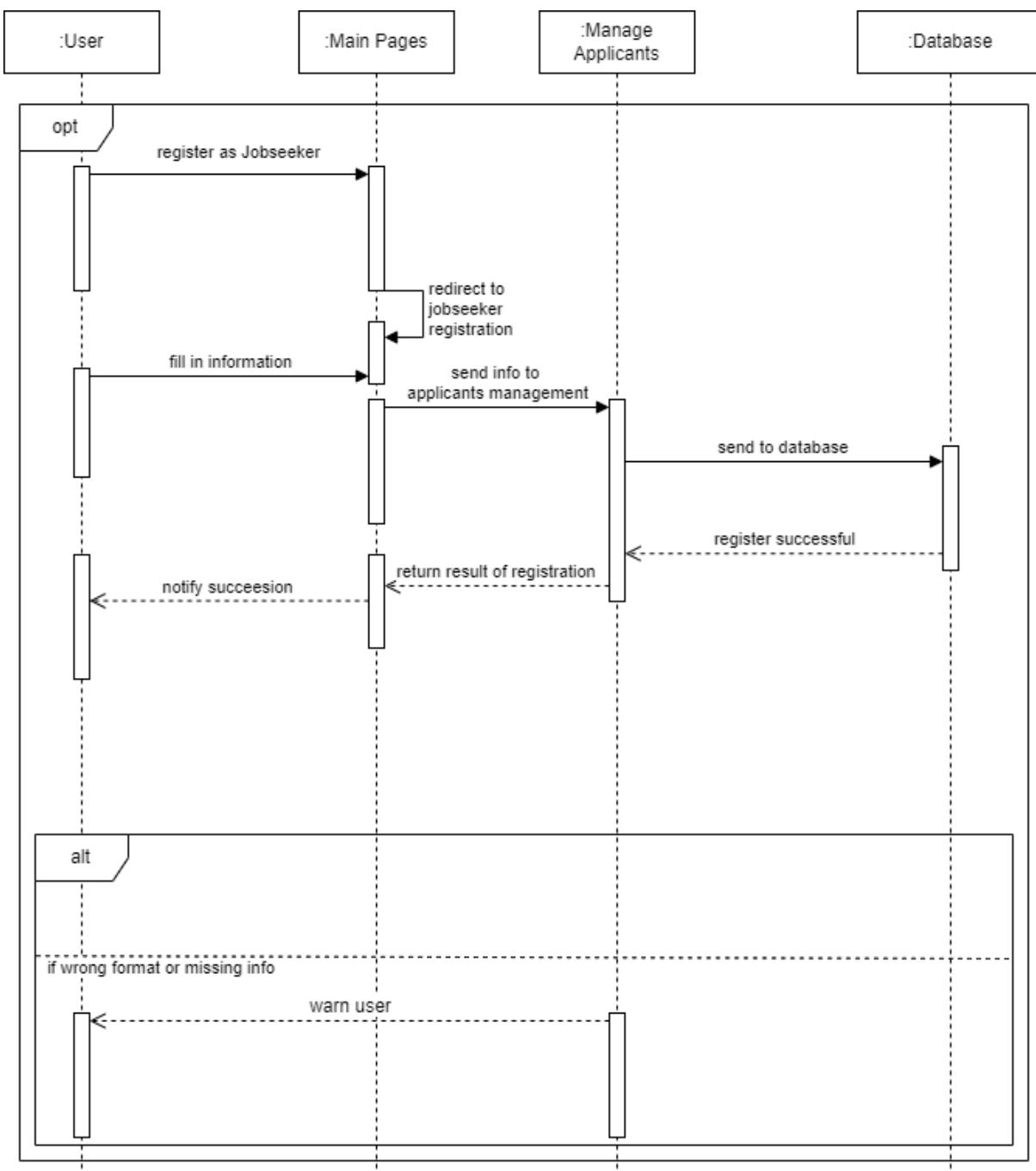
2.3.1 Search Jobs



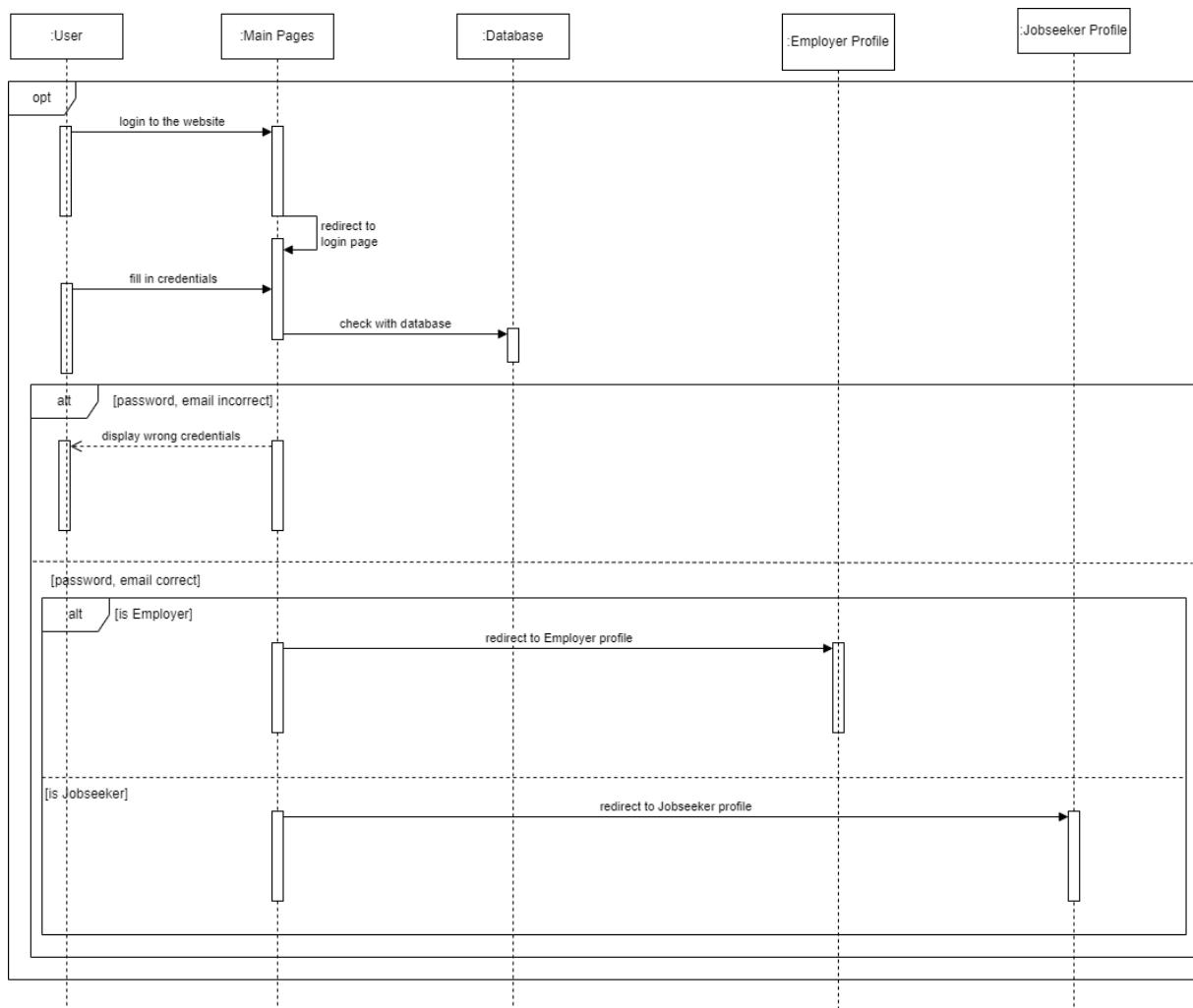
2.3.2 Register as Employer



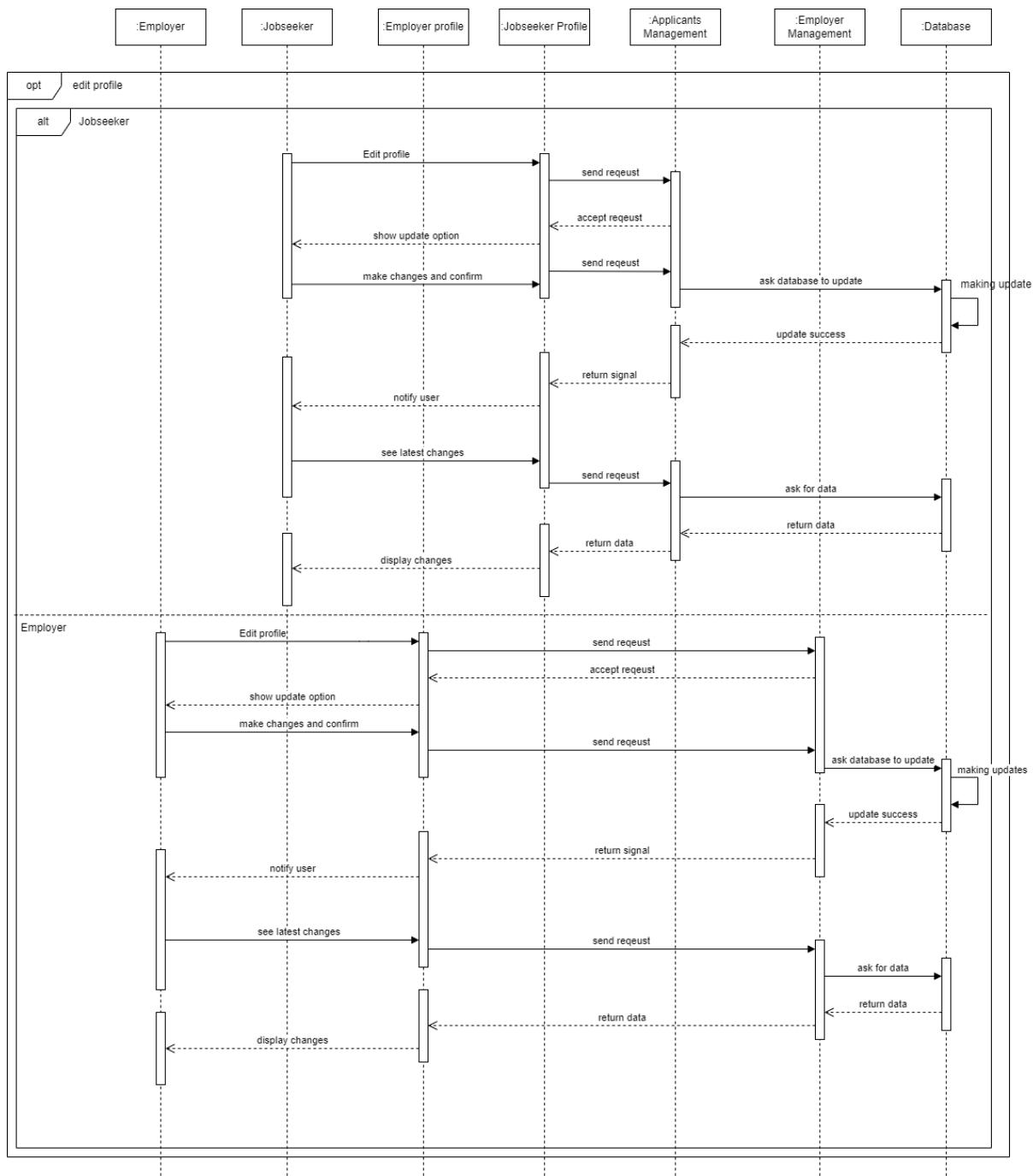
2.3.3 Register as Jobseeker



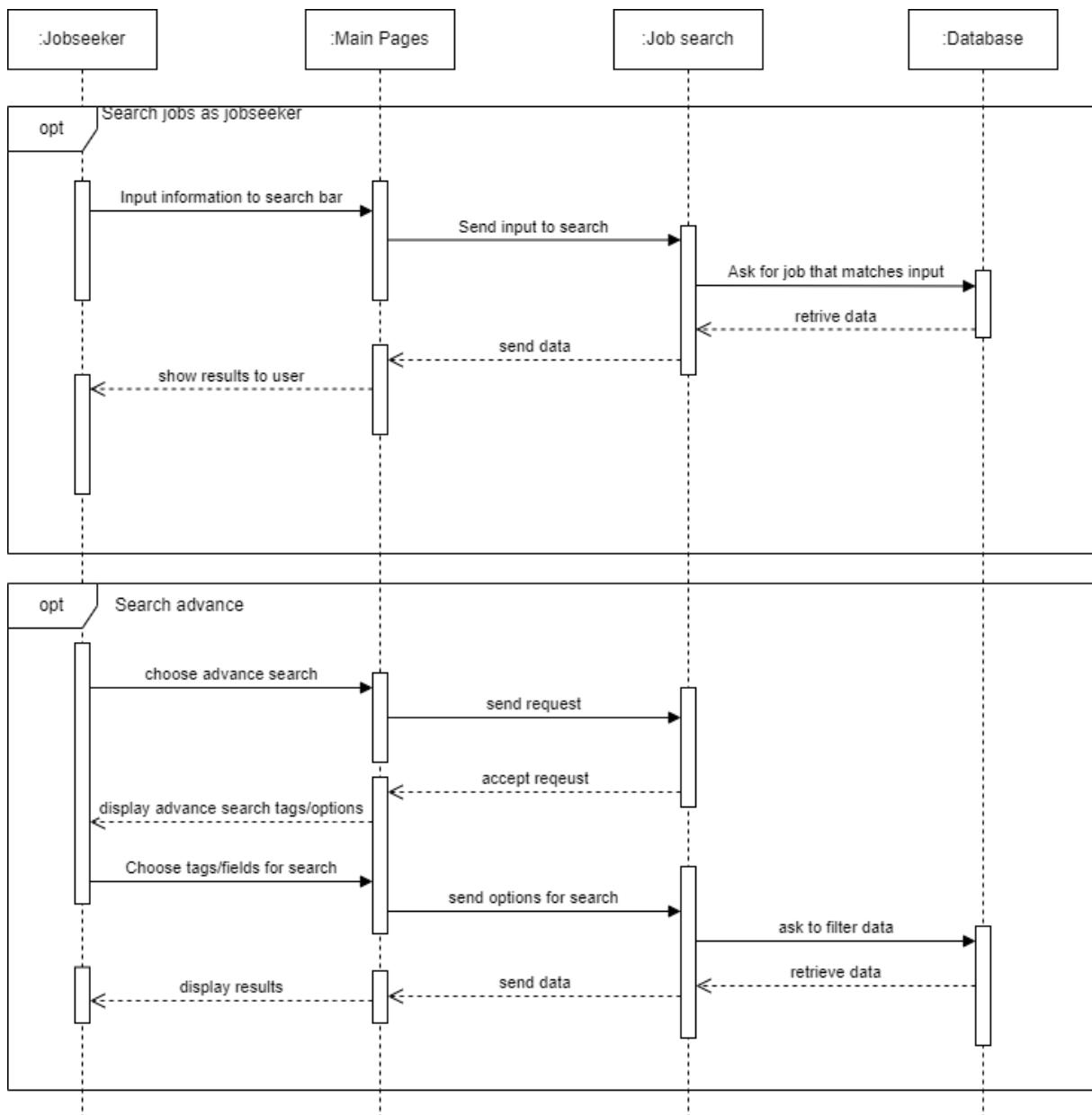
2.3.4 Login to the website



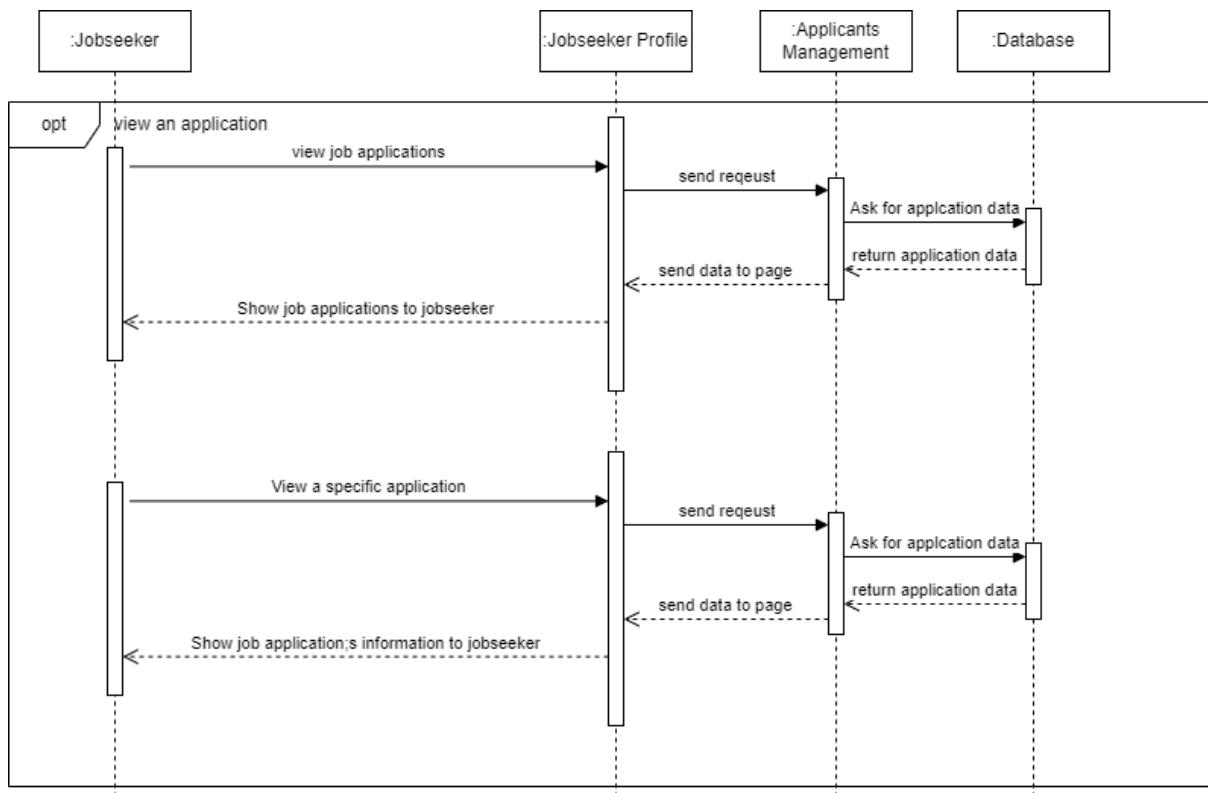
2.3.5 Edit profile



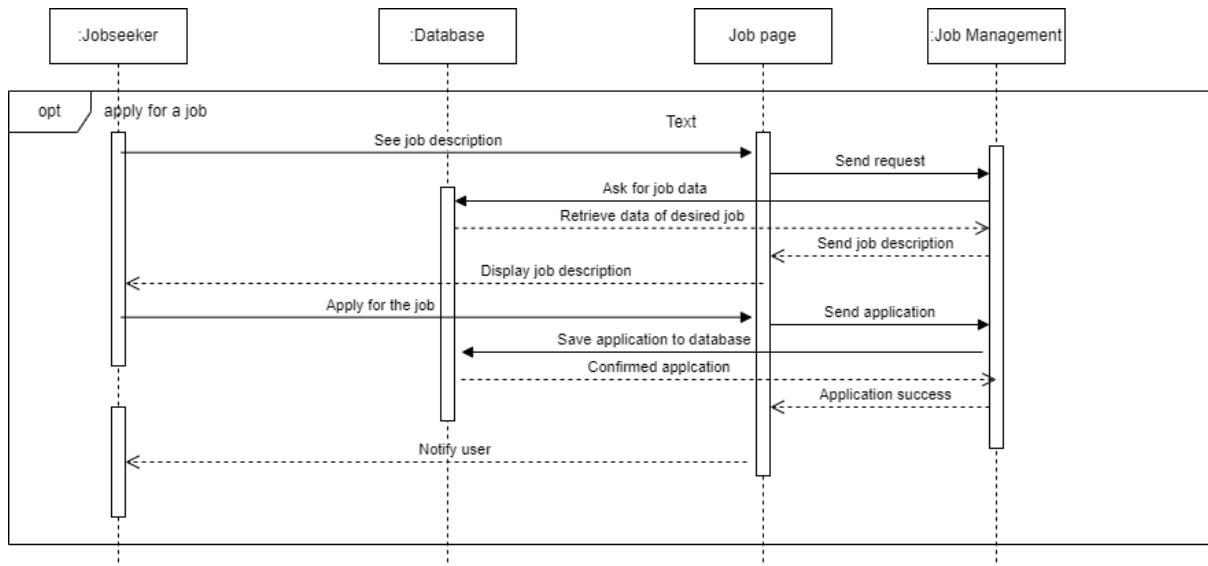
2.3.6 Search Jobs for registered user



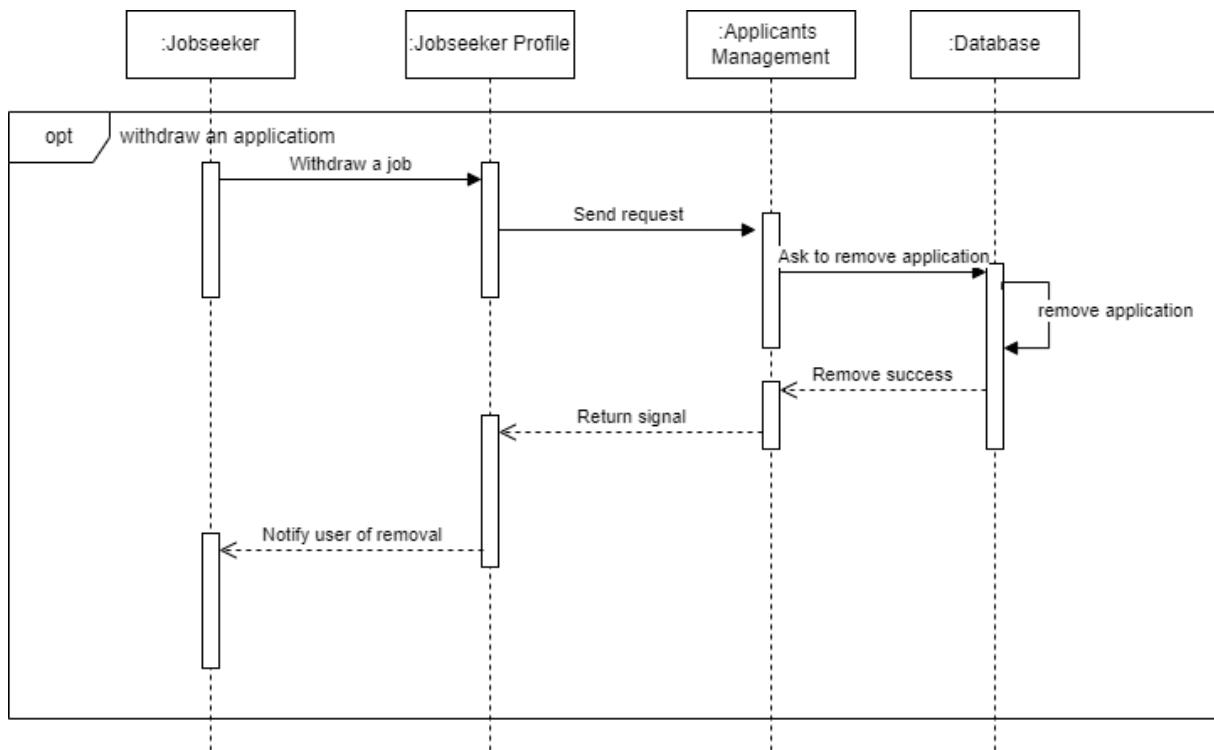
2.3.7 View jobs application details



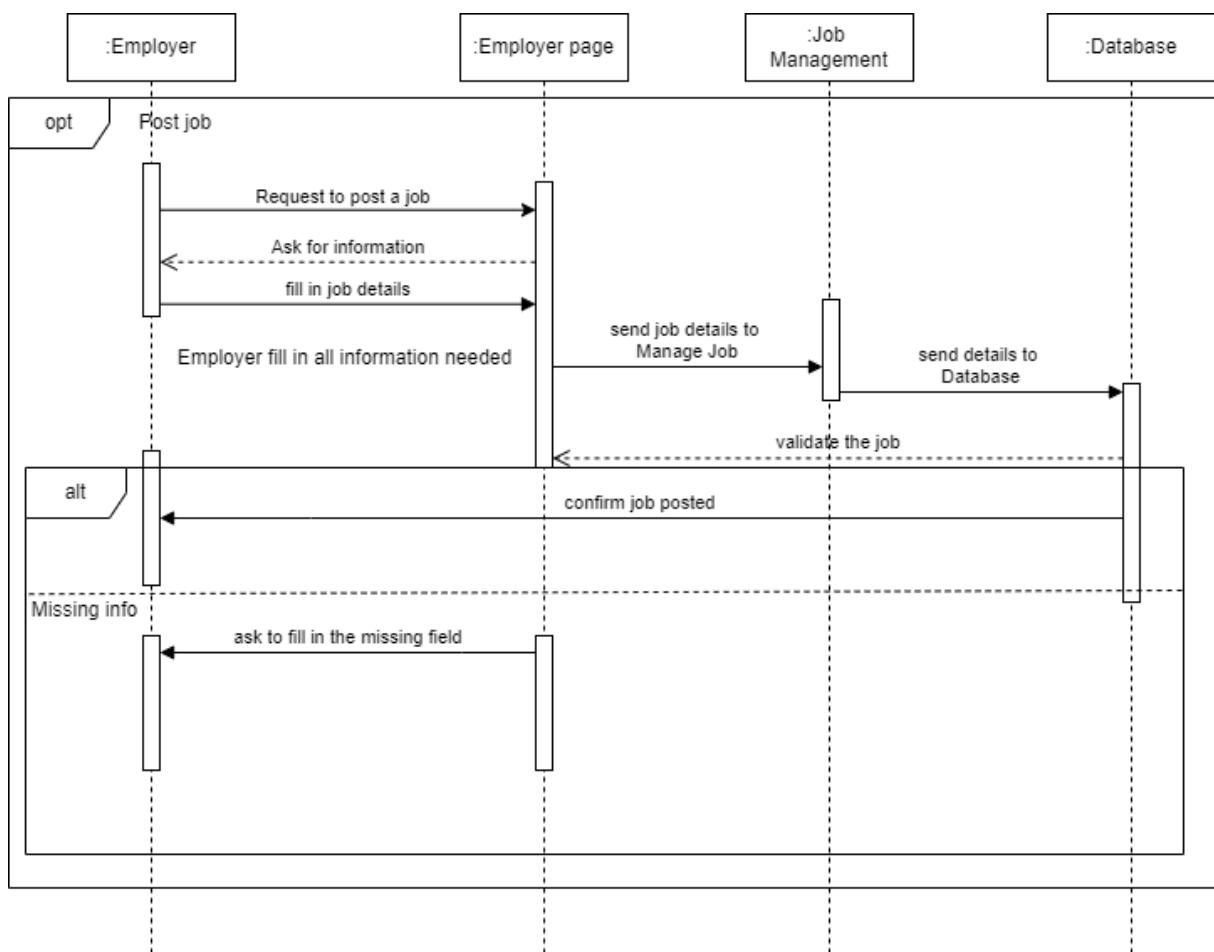
2.3.8 Job application



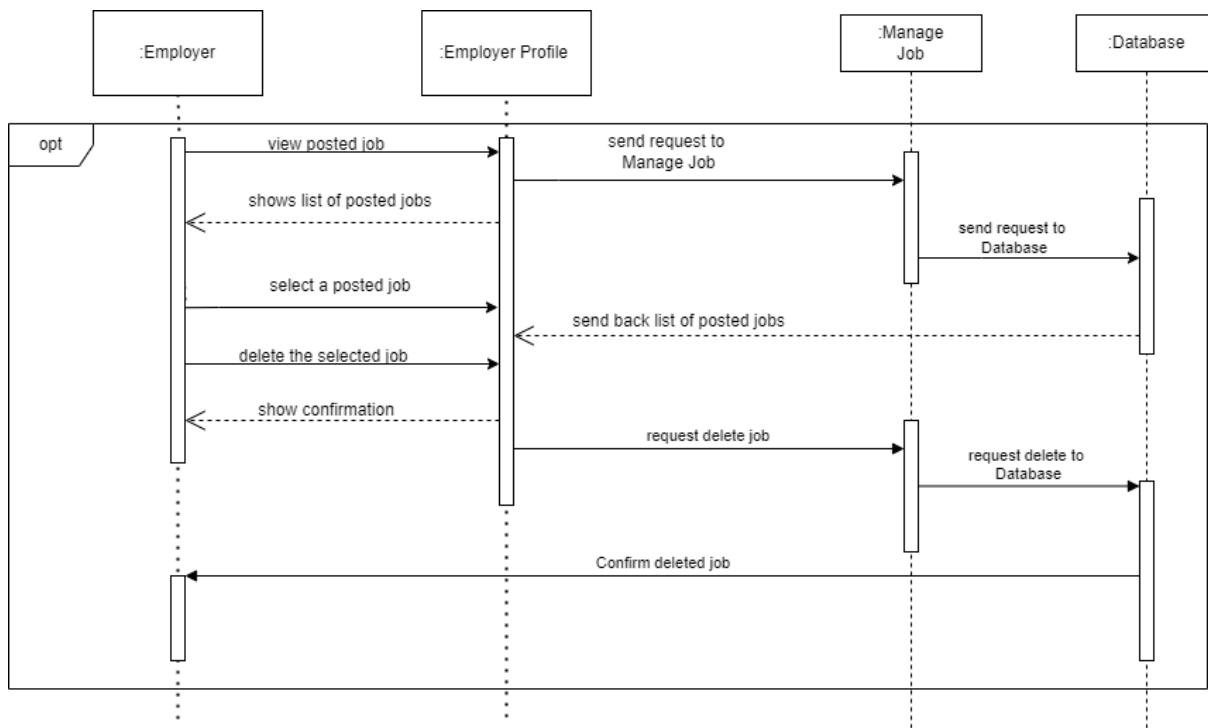
2.3.9 Withdraw job application



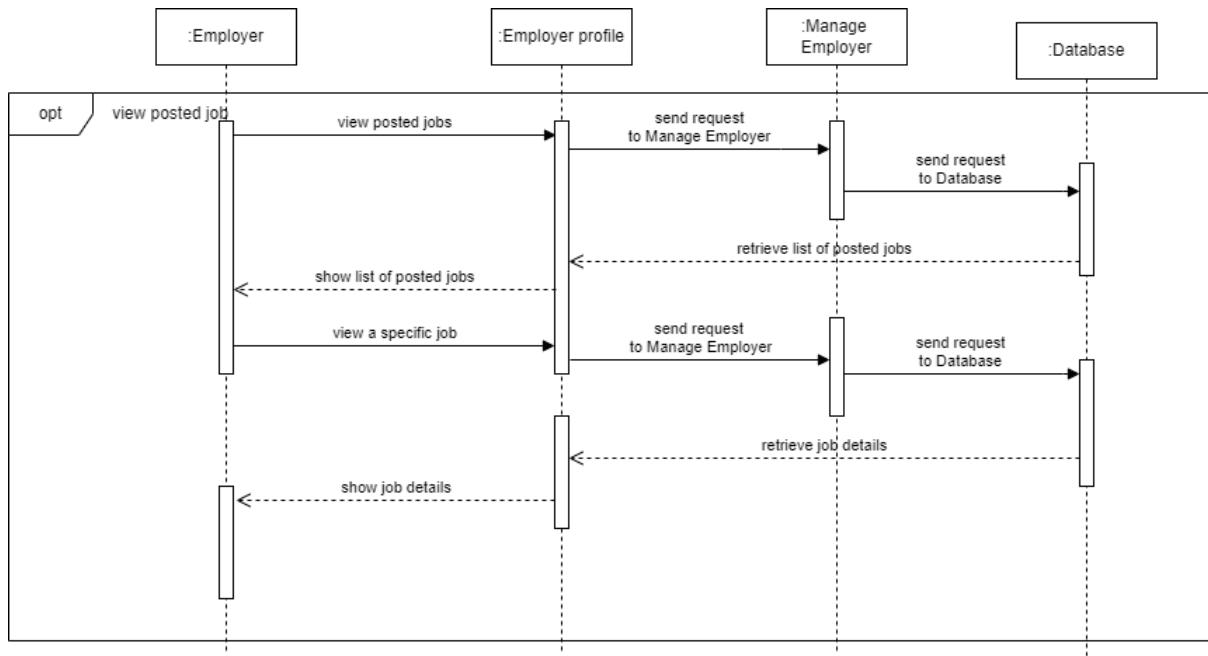
2.3.10 Post jobs



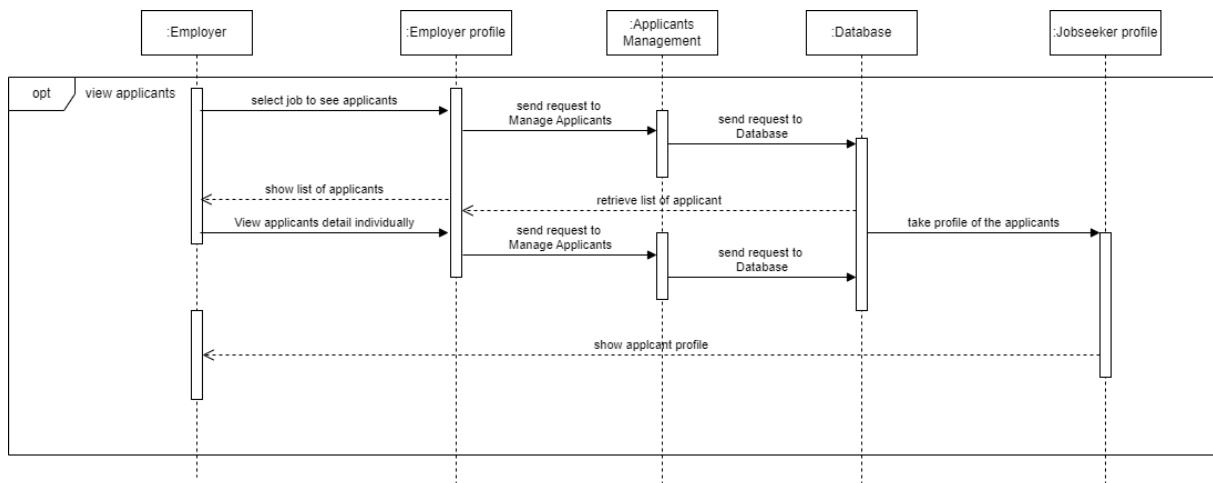
2.3.11 Delete jobs



2.3.12 View jobs

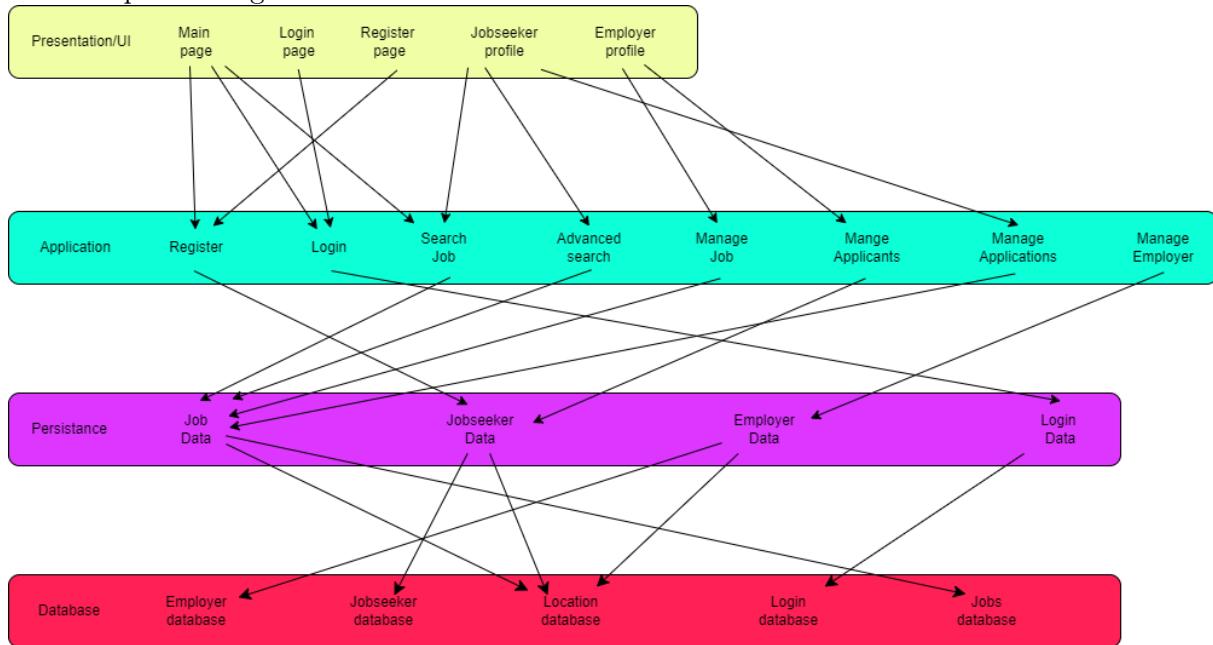


2.3.13 View applicants info



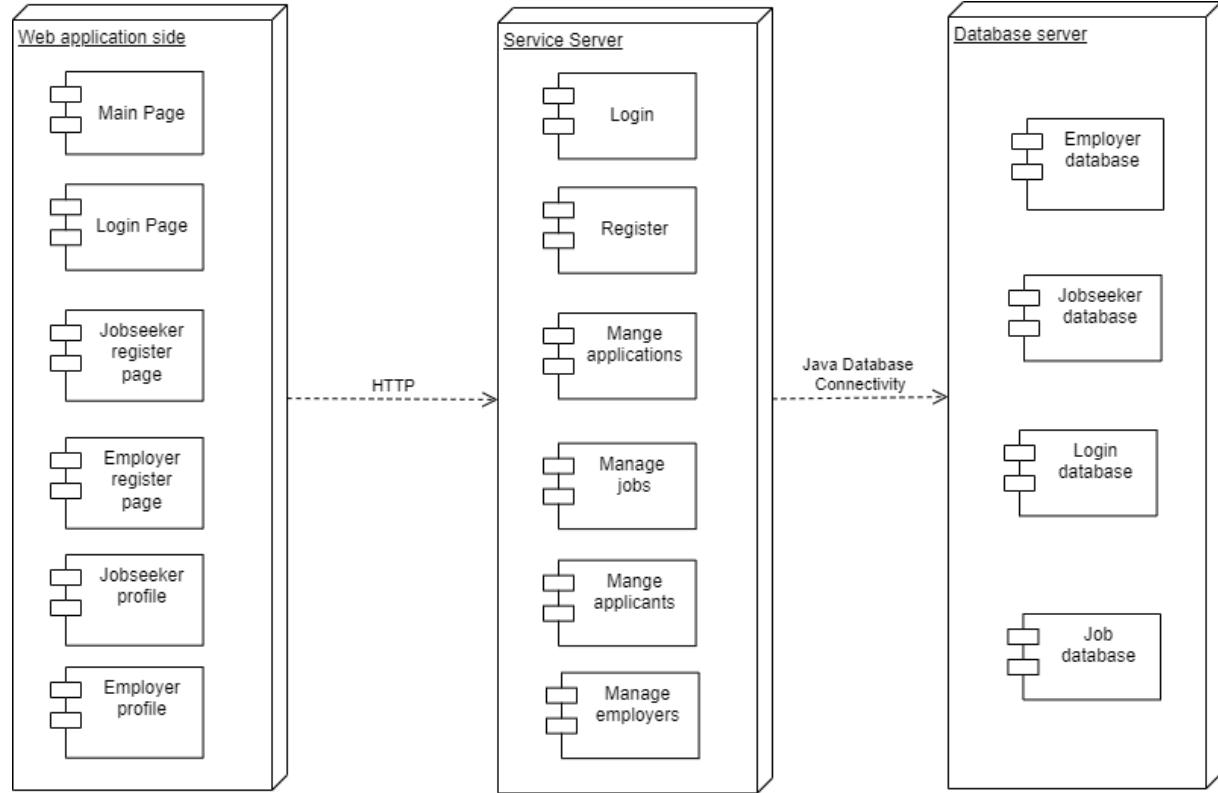
2.4 Software Architecture

Here we will present our software architecture corresponding to the components and functions in the sequence diagrams.



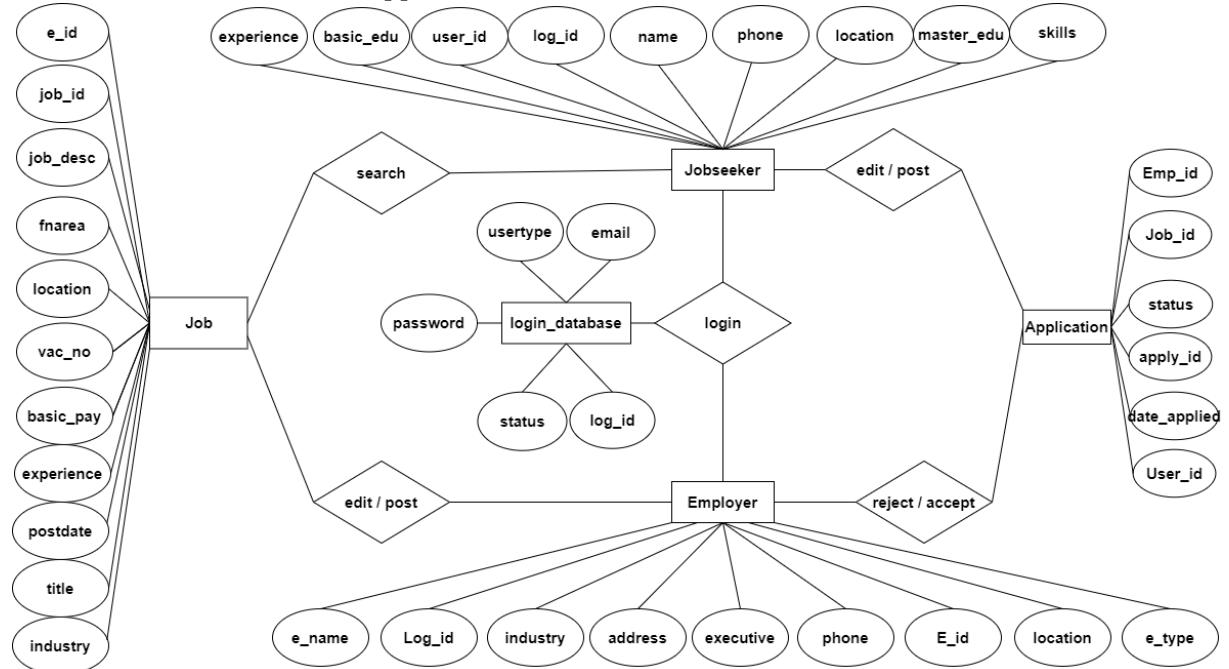
2.5 Deployment Diagram

Based on the architecture diagram, we can implement as well an UML Deployment diagram.



2.6 Entity-Relationship Diagram

In this section, we will attempt to brainstorm an entity relationship diagram to show the schema of our database for the web application.



2.7 User Interface Mock-up

2.7.1 Main Page

ITJob logo	Home		Register	Login
------------	------	--	----------	-------

See what jobs are available:
Jobs available:

Search

The best part? Everything.

Easily find jobs
Easily find employees

Become a Jobseeker

Become a Employer

Contact



back to top

2.7.2 Login Page

ITJob logo	Login	Sign Up
------------	-------	---------

Please sign in

Email address

▼ Password

Sign in

Find IT Jobs

Register today

Looking for the right employees ?

Register Today

2.7.3 Jobseeker Registration Page

ITJob logo	Jobseeker Registration		Login
------------	------------------------	--	-------

Your Login Details

Enter your Email ID:

Create new Password:

Confirm the Password:

Your Contact Information

Your full name:

Where are you currently located?

Enter your Mobile number:

Your Current Employment Details

How much work experience do you have

What are your Key skills

Your Education Qualifications

Your Basic education

Your Masters Education

2.7.4 Employer Registration Page

ITJob logo	Employer Registration	Login
<div style="border: 2px solid #ccc; border-radius: 10px; height: 150px;"></div>		

Your Login Details

Email:

Password

Confirm password

Your Company Details

Company Name:

Company Type: Company Consultant

Industry:

Address:

Pincode:

Contact person:

Contact number:

Where are you currently located?

Check for errors before submitting the form!!

2.7.5 Employer Page

ITJob logo Profile Menu Account+ Logout

Welcome "Company_name" Post jobs and find the right candidates

You can post a new job, manage your jobs and update your profile

Company Profile
The following information are visible to job seekers. We recommend you to always update these information

Picture

Company_name

Change company logo

Name:
Type:
Industry:
Address:
Pincode:
Executive name:
Phone number:
Email:
Main location:
About company:

2.7.6 Jobseeker Page

ITJob logo Profile Options Search Jobs Search Logout

Welcome "Jobseeker_name" Find jobs, edit your profile or update your current resume for better jobs

Your Profile Recommended jobs Update resume Advanced Search

Picture

Jobseeker_name

Change image

Your Profile
Full name:
Email:
Phone:
Location:
Experience(Years):
Skills:
UG Qualification:
PG Qualification:

3 Back-end Programming

3.1 Login

```
1 $email=$_POST['email'];
2 $passd=$_POST['password'];
```

Retrieve User Input: The script starts by retrieving the user's email and password from the POST request. These values are stored in the '\$email' and '\$passd' variables, respectively.

```
1 $query=mysqli_query($db1,"select * from login where email='".$email"");
```

Database Query: It then performs a database query to find a user in the login table with the provided email. The result of this query is stored in the \$query variable.

```
1 $result=mysqli_fetch_array($query ,MYSQLI_ASSOC);
```

Fetch Query Result: The script fetches the result of the query as an associative array using "mysqli_fetch_array" and stores it in the "\$result" variable.

```
1 if(($result>0) && ( password_verify( $passd, $result['password'] ) ) ){
```

Verify User and Password: The script checks if the query returned a result and if the provided password matches the hashed password stored in the database using "password_verify".

```
1 if($result['usertype']=="jobseeker")
2 {
3     session_start();
4     $_SESSION["id"] = $result['log_id'];
5     $_SESSION["type"] = $result['usertype'];
6     header('location:jobseeker/profile.php?msg=success');
7 }
8 elseif($result['usertype']=="employer")
9 {
10    session_start();
11    $_SESSION["elogid"] = $result['log_id'];
12    $_SESSION["type"] = $result['usertype'];
13    $_SESSION["status"]=$result['status'];
14    header('location:employer/profile.php?msg=success');
15 }
```

User Type Handling: If the user is found and the password is correct, the script checks the user type.

If the user is a jobseeker, it starts a new session, sets session variables for the user ID and type, and redirects the user to the jobseeker profile page with a success message.

If the user is an employer, it also starts a new session, sets session variables for the user ID, type, and status, and redirects the user to the employer profile page with a success message.

```

1 else
2 {
3 header('location:login.php?msg=failed');
4 }

```

Login Failure: If the user is not found or the password does not match, the script redirects the user back to the login page with a failed message.

3.2 Register

3.2.1 Register for Employer

```

1 include_once('../config.php');
2 $email=$_POST['email'];
3 $password=$_POST['pass1'];
4 $hash = password_hash($password, PASSWORD_DEFAULT);
5 $name=$_POST['compname'];
6 $type=$_POST['comtype'];
7 //echo $type;
8 $industry=$_POST['indtype'];
9 //echo $industry;
10 $addr=$_POST['addr'];
11 $pin=$_POST['pin_code'];
12 $person=$_POST['person'];
13 $phone=$_POST['phone'];
14 $countryid=$_POST['country'];
15 $stateid=$_POST['state'];
16 $cityid=$_POST['city'];
17
18 mysqli_select_db($db2,"location");
19
20 $query1=mysqli_query($db2,"select name from countries WHERE id = '$countryid'" or die("Wrong Query"));
21 $row = mysqli_fetch_assoc($query1);
22 $country= $row['name'];
23
24 $query2=mysqli_query($db2,"select name from states WHERE id = '$stateid'" or die("Wrong Query"));
25 $row = mysqli_fetch_assoc($query2);
26 $state= $row['name'];
27 //echo $state;
28
29 $query3=mysqli_query($db2,"select name from cities WHERE id = '$cityid'" or die("Wrong Query"));
30 $row = mysqli_fetch_assoc($query3);
31 $city= $row['name'];
32
33 $location=$country.", ".$state.", ".$city;
34

```

```

35 $query4= "INSERT INTO login (email,password,userstype,status) VALUES ('$email' ,
36     '$hash','employer',0)";
37 $result1 = mysqli_query($db1,$query4) or die("Cant Register , The user email
38     may be already existing");
39 $query5 = "INSERT INTO employer (log_id,ename,phone,location,etype,address,
40     pincode,executive,industry)
41         VALUES ((SELECT log_id FROM login WHERE email='$email'),'$name
42     ','$phone','$location','$type','$addr','$pin','$person','$industry')";
43 // $result2 = mysqli_query($db1,$query5);
44 if (!mysqli_query($db1,$query5))
45 {
46     echo("Error description: " . mysqli_error($db1));
47 }
48 else{
49     header('location:../login.php');
50 }

```

This PHP script, part of a registration process for employers, performs the following actions:

1. Includes the database configuration file.
2. Retrieves employer registration data (email, password, company name, company type, industry type, address, pin code, contact person, phone number, country, state, and city IDs) from a form submission via POST method.
3. Hashes the password using a default algorithm for security.
4. Selects the location database for subsequent queries.
5. Executes three queries to the location database to retrieve the names of the country, state, and city based on their IDs. Each query:
 - Is executed against the db2 database connection.
 - Uses the mysqli_query function to perform the SQL query.
 - Utilizes or die("Wrong Query") for basic error handling, stopping script execution if a query fails.
 - Fetches the result with mysqli_fetch_assoc to get the name of the location (country, state, city) from the respective tables (countries, states, cities).
6. Stores the retrieved location names in variables for further use.

3.2.2 Register as Jobseeker

```

1 include_once('../config.php');
2 // Data retrieved begins here
3 $email=$_POST['useremail'];
4 //echo $email;
5 $password=$_POST['pass1'];
6 $hash = password_hash($password, PASSWORD_DEFAULT);

```

```

7 //echo $password;
8 $name=$_POST['uname'];
9 $mobile=$_POST['mobno'];
10 $experience=$_POST['experience'];
11 $skills=$_POST['skill1'].".", ". $_POST['skill2'].".", ". $_POST['skill3'].".", ". $_POST
    ['skill4'];
12 $ug=$_POST['ugcourse'];
13 $pg=$_POST['pgcourse'];
14 $countryid=$_POST['country'];
15
16 $location="";
17 $type="jobseeker";
18 // data retreived ends here
19
20 // now wants to fetch data from location db
21
22 mysqli_select_db($db2,"location");
23 $query1=mysqli_query($db2,"select name from countries WHERE id = '$countryid'" or die("Wrong Query"));
24 $row = mysqli_fetch_assoc($query1);
25 $country= $row['name'];
26
27 $query2=mysqli_query($db2,"select name from states WHERE id = '$stateid'" or die("Wrong Query"));
28 $row = mysqli_fetch_assoc($query2);
29 $state= $row['name'];
30 //echo $state;
31
32 $query3=mysqli_query($db2,"select name from cities WHERE id = '$cityid'" or die("Wrong Query"));
33 $row = mysqli_fetch_assoc($query3);
34 $city= $row['name'];
35 //echo $city;
36 $location=$country.", ".$state.", ".$city;
37 //echo $location;
38 mysqli_close($db2);
39 mysqli_select_db($db1,"jobportal");
40
41 $query4="INSERT INTO login (email,password,usertype,status) VALUES ('$email','
    $hash','$type',1)";
42 $result1 = mysqli_query($db1,$query4) or die("Cant Register , The user email
    may be already existing");
43 $query5 = "INSERT INTO jobseeker (log_id,name,phone,location,experience,skills,
    basic_edu,master_edu)
        VALUES ((SELECT log_id FROM login WHERE email='$email'),'$name
    ','$mobile','$location','$experience','$skills','$ug','$pg')";
44
45
46 // $result2 = mysqli_query($db1,$query5);
47 if (!mysqli_query($db1,$query5))

```

```

48 {
49 echo("Error description: " . mysqli_error($db1));
50 }
51 else{
52 header('location:../login.php?msg=registered');
53 }

```

Include Configuration: The script starts by including a configuration file (config.php). This file likely contains database connection settings and other global configurations.

Retrieve Form Data: It retrieves data submitted via a POST request from a registration form. This includes the user's email, password, name, mobile number, experience, skills, undergraduate course, postgraduate course, and country ID. The password is then hashed using PHP's password_hash function for secure storage.

Concatenate Skills: The skills are received from individual fields (skill1, skill2, skill3, skill4) and concatenated into a single string, separated by commas. This suggests the form allows the user to enter up to four skills.

Set Default Values: It sets default values for location and type. The location is initially an empty string, and type is set to "jobseeker", indicating the role of the user in the system.

Database Selection for Location Data: The script selects a database named "location" using mysqli_select_db. This indicates that location-related data (countries, states) are stored in a separate database or a separate schema within the same database.

Fetch Country Name: It executes a query to fetch the name of the country based on the 'countryid' provided by the user. The country name is retrieved from the countries table. If the query fails, it terminates the script and prints "Wrong Query", which is a basic error handling mechanism.

Fetch State Name: Similar to fetching the country name, it attempts to fetch the state name using an undefined variable \$stateid. This is a mistake in the script as \$stateid is not defined anywhere in the provided code. This would likely result in a PHP notice or warning and could cause the query to fail or return unexpected results.

3.3 Manage applicants

```

1 function selectJs(user, job, emp) {
2     var xmlhttp;
3     if (window.XMLHttpRequest) { // for IE7+, Firefox, Chrome, Opera,
4         Safari
5             xmlhttp = new XMLHttpRequest();
6         } else { // for IE6, IE5
7             xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
8         }
9         xmlhttp.onreadystatechange = function() {
10             if (xmlhttp.readyState != 4 && xmlhttp.status == 200) {
11                 document.getElementById("message").innerHTML = "Processing
Request..";
12             } else if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
287
288
289
289
290
291
292
293
294
295
296
297
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
317
318
319
319
320
321
322
323
324
325
326
327
327
328
329
329
330
331
332
333
334
335
336
337
337
338
339
339
340
341
342
343
344
345
346
346
347
348
348
349
349
350
351
352
353
354
355
356
356
357
358
358
359
359
360
361
362
363
364
365
365
366
367
367
368
368
369
369
370
371
372
373
374
375
375
376
377
377
378
378
379
379
380
381
382
383
384
385
385
386
387
387
388
388
389
389
390
391
392
393
394
395
395
396
397
397
398
398
399
399
400
401
402
403
404
405
405
406
407
407
408
408
409
409
410
411
412
413
414
414
415
416
416
417
417
418
418
419
419
420
421
422
423
424
424
425
426
426
427
427
428
428
429
429
430
431
432
433
434
434
435
436
436
437
437
438
438
439
439
440
441
442
443
444
444
445
446
446
447
447
448
448
449
449
450
451
452
453
454
454
455
456
456
457
457
458
458
459
459
460
461
462
463
464
464
465
466
466
467
467
468
468
469
469
470
471
472
473
474
474
475
476
476
477
477
478
478
479
479
480
481
482
483
484
484
485
486
486
487
487
488
488
489
489
490
491
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
501
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
```

```
12         document.getElementById("message").innerHTML = xmlhttp.
13         responseText;
14     } else {
15         document.getElementById("message").innerHTML = "Error
16 Occurred. <a href='manage_applicants.php'>Reload Or Try Again</a> the page.."
17     ;
18     }
19     xmlhttp.open("GET", "process_select.php?user=" + user + "&job=" + job +
20 "&emp=" + emp, true);
21     xmlhttp.send();
22 }
23
24 function rejectJs(user, job, emp) {
25     var xmlhttp;
26     if (window.XMLHttpRequest) { // for IE7+, Firefox, Chrome, Opera,
27 Safari
28         xmlhttp = new XMLHttpRequest();
29     } else { // for IE6, IE5
30         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
31     }
32     xmlhttp.onreadystatechange = function() {
33         if (xmlhttp.readyState != 4 && xmlhttp.status == 200) {
34             document.getElementById("message").innerHTML = "Processing
35 Request..";
36         } else if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
37             document.getElementById("message").innerHTML = xmlhttp.
38         responseText;
39         } else {
40             document.getElementById("message").innerHTML = "Error
41 Occurred. <a href='manage_applicants.php'>Reload Or Try Again</a> the page.."
42     ;
43     }
44     xmlhttp.open("GET", "process_reject.php?user=" + user + "&job=" + job +
45 "&emp=" + emp, true);
46     xmlhttp.send();
47 }
```

Initialization of XMLHttpRequest: Both functions start by creating an XMLHttpRequest object. This object allows you to request data from a server without reloading the page. For compatibility with older versions of Internet Explorer (IE5 and IE6), an ActiveXObject is used as a fallback.

Setting up the request: The `.open()` method initializes a newly-created request, or reinitializes an existing one. In this case, it's configured to make a GET request to either `process_select.php` or `process_reject.php`, passing `user`, `job`, and `emp` as query parameters. The `true` parameter indicates that the request should be made asynchronously.

Handling the response: The onreadystatechange event listener is set up to handle the response

from the server. This function checks the readyState and status of the request:

- If readyState is not 4 (request finished and response is ready) and status is 200 (OK), it updates the inner HTML of an element with the ID message to "Processing Request..", indicating that the request is in progress.
- If readyState is 4 and status is 200, it updates the message element's inner HTML to display the response from the server, which could be a success message, details about the operation, or any other server-generated content.
- For any other case, it displays an error message with a link to reload the page or try again, indicating that something went wrong with the request.

Sending the request: Finally, the .send() method is called to send the request to the server.

3.4 Manage jobs

```

1 include_once('../config.php');
2 session_start();
3 $eid=$_SESSION['eid'];
4 $desig=$_POST['desig'];
5 $vacno=$_POST['vacno'];
6 $desc=$_POST['jobdesc'];
7 $exp=$_POST['exp'];
8 $money=$_POST['money'];
9 $salary=$_POST['pay'];
10 $fnarea=$_POST['fnarea'];
11 $countryid=$_POST['country'];
12 $stateid=$_POST['state'];
13 $cityid=$_POST['city'];
14 $indtype=$_POST['indtype'];
15 $ug=$_POST['ugcourse'];
16 $pg=$_POST['pgcourse'];
17 $profile=$_POST['profile'];
18 $date=date('d-m-y');
19 $pay=$money." ".$salary;
20 mysqli_select_db($db2,"location");
21 $query1=mysqli_query($db2,"select name from countries WHERE id = '$countryid'" )
    or die("Wrong Query");
22 $row = mysqli_fetch_assoc($query1);
23 $country= $row['name'];
24
25 $query2=mysqli_query($db2,"select name from states WHERE id = '$stateid'" ) or
    die("Wrong Query");
26 $row = mysqli_fetch_assoc($query2);
27 $state= $row['name'];
28 //echo $state;
29

```

```

30 $query3=mysqli_query($db2,"select name from cities WHERE id = '$cityid'" ) or
     die("Wrong Query");
31 $row = mysqli_fetch_assoc($query3);
32 $city= $row['name'];
33
34 $location=$country . "," . $state . "," . $city;
35 mysqli_close($db2);
36 mysqli_select_db($db1,"jobportal");
37
38 $query4="insert into jobs (eid,title,jobdesc,vacno,experience,basicpay,fnarea,
     location,industry,ugqual,pgqual,profile,postdate)VALUES ('$eid','$desig',
     '$desc','$vacno','$exp','$pay','$fnarea','$location','$indtype','$ug',
     '$pg','$profile','$date')";
39
40 if (!mysqli_query($db1,$query4))
41 {
42 echo("Error description: " . mysqli_error($db1));
43 }
44 else{
45 header('location:profile.php?msg=jobposted');
46 }

```

Configuration and Session Start: include_once('..../config.php'); includes a configuration file that likely contains database connection settings and other global configurations. session_start(); starts a new session or resumes an existing one, allowing the use of session variables.

Retrieving Form Data: The script retrieves data submitted via a POST request from a form where an employer fills out details about a job vacancy. This includes the employer's ID (\$eid from session), job designation (\$desig), number of vacancies (\$vacno), job description (\$desc), experience required (\$exp), salary details (\$money and \$salary), functional area (\$fnarea), location details (\$countryid, \$stateid, \$cityid), industry type (\$indtype), required qualifications (\$ug for undergraduate, \$pg for postgraduate), job profile (\$profile), and the posting date (\$date).

Building the Location String: The script selects the location database to fetch the names of the country, state, and city based on their IDs. These names are concatenated to form a single string (\$location) representing the job's location.

Inserting Job Details into Database: After closing the connection to the location database and selecting the jobportal database, it attempts to insert the job details into the jobs table using the mysqli_query function. The query includes all the retrieved and processed information.

Error Handling and Redirection: If the insertion fails, an error message is displayed using mysqli_error(\$db1). If the insertion is successful, the employer is redirected to their profile page with a message indicating that the job has been posted (header('location:profile.php msg=jobposted')).

```

1 include_once('..../config.php');
2 session_start();
3 if(!isset($_SESSION['id'])){
4     header('location:..../login.php?msg=please_login');

```

```
5 }
6 elseif(!isset($_GET['jid'])){
7     header('location:managejobs.php?msg=selectjob');
8 }
9 $query = "DELETE FROM jobs WHERE jobid=$_GET[jid]";
10 $result = mysqli_query($db1,$query);
11 if($result) {
12     echo "<h3 style='color: green;'> Selected Job Is Successfully Deleted</h3>";
13 }
14 else{
15     echo "<h3 style='color: red;'> Failed to delete the selected job!</h3>";
16 }
```

Start a session: `session_start()`; initializes a session or resumes the current one based on a session identifier passed via a GET or POST request, or passed via a cookie.

Check if a user is logged in: The script checks if there's an id set in the session with `if(!isset($_SESSION['id']))....` If not, it redirects the user to the login page with a message indicating that the user needs to log in.

Check if a job ID is provided: It then checks if a job ID (jid) is provided in the GET request with elseif(!isset(\$_GET['jid'])).... If not, it redirects the user to the manage jobs page with a message indicating that a job should be selected.

Delete the job: If both checks pass, it constructs a SQL query to delete the job from the jobs table where the jobid matches the provided job ID from the GET request: `$query = "DELETE FROM jobs WHERE jobid=$_GET[jid]"`. This line is vulnerable to SQL injection as it directly includes a GET parameter in the SQL query without sanitization or prepared statements.

Execute the query: The script then attempts to execute the query using `mysqli_query($db1, $query);`. It checks if the query was successful.

Feedback to the user: If the query was successful, it displays a message indicating that the job was successfully deleted. If not, it displays a message indicating failure to delete the job.

3.5 Manage applications

```
1 function rejectjob(jobid) {
2     // alert(keyword);
3     var xmlhttp;
4     if (window.XMLHttpRequest) { // for IE7+, Firefox, Chrome, Opera,
Safari
5         xmlhttp = new XMLHttpRequest();
6     } else { // for IE6, IE5
7         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
8     }
9     xmlhttp.onreadystatechange = function() {
10        if (xmlhttp.readyState != 4 && xmlhttp.status == 200) {
11            document.getElementById("message").innerHTML = "Processing..
";
12        } else if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
13        }
14    }
15 }
```

```

13         document.getElementById("message").innerHTML = xmlhttp.
14         responseText;
15     } else {
16         document.getElementById("message").innerHTML = "Error
17         Occurred. <a href='profile.php'>Reload Or Try Again</a> the page.";
18     }
19     xmlhttp.open("GET", "reject.php?jid=" + jobid, true);
20     xmlhttp.send();
}

```

The **rejectjob function** in the provided PHP file sends an asynchronous GET request to reject.php with a job ID as a query parameter. It updates the content of an element with the ID message on the webpage based on the response from the server.

If the request is in process and successful, it displays "Processing..". Once completed successfully, it shows the server's response.

If there's an error, it displays an error message with a link to reload or try again.

```

1 function apply(jobid) {
2     // alert(keyword);
3     var xmlhttp;
4     if (window.XMLHttpRequest) { // for IE7+, Firefox, Chrome, Opera, Safari
5         xmlhttp = new XMLHttpRequest();
6     } else { // for IE6, IE5
7         xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
8     }
9     xmlhttp.onreadystatechange = function() {
10        if (xmlhttp.readyState != 4 && xmlhttp.status == 200) {
11            document.getElementById("applydiv").innerHTML = "Processing..";
12        } else if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
13            document.getElementById("applydiv").innerHTML = xmlhttp.responseText;
14        } else {
15            document.getElementById("applydiv").innerHTML = "Error Occurred. <a href='
16            profile.php'>Reload Or Try Again</a> the page.";
17        }
18    }
19    xmlhttp.open("GET", "apply_job.php?jid=" + jobid, true);
20    xmlhttp.send();
}

```

The **apply function** sends an asynchronous GET request to apply_job.php with a job ID (jobid) as a query parameter. It updates the content of an element with the ID applydiv on the webpage based on the response from the server. If the request is in process, it displays "Processing..". Once completed successfully, it shows the server's response. If there's an error, it displays an error message with a link to reload or try again.

3.6 Manage employers

```

1 // Start the session

```

```

2 session_start();
3
4 // Include your database connection file here
5 include('../config.php');
6
7
8 // Check if form is submitted
9 if ($_SERVER["REQUEST_METHOD"] == "POST") {
10     $name=$_POST['ename'];
11     $type=$_POST['etype'];
12
13     //echo $type;
14     $industry=$_POST['industry'];
15
16     //echo $industry;
17     $addr=$_POST['address'];
18     $pin=$_POST['pincode'];
19     $person=$_POST['executive'];
20     $phone=$_POST['phone'];
21     $countryid=$_POST['country'];
22     $stateid=$_POST['state'];
23     $cityid=$_POST['city'];
24     $location="";
25
26     // now wants to fetch data from location db
27     mysqli_select_db($db2,"location");
28     $query1=mysqli_query($db2,"select name from countries WHERE id = '$countryid'");
29     or die("Wrong Query");
30     $row = mysqli_fetch_assoc($query1);
31     $country= $row['name'];
32
33     $query2=mysqli_query($db2,"select name from states WHERE id = '$stateid'");
34     or die("Wrong Query");
35     $row = mysqli_fetch_assoc($query2);
36     $state= $row['name'];
37     //echo $state;
38
39     $query3=mysqli_query($db2,"select name from cities WHERE id = '$cityid'");
40     or die("Wrong Query");
41     $row = mysqli_fetch_assoc($query3);
42     $city= $row['name'];
43     //echo $city;
44     $location=$country." ".$state." ".$city;
45     //echo $location;
46     mysqli_close($db2);
47     mysqli_select_db($db1,"jobportal");
48
49     // Prepare an UPDATE statement
50     $sql = "UPDATE employer SET ename=? , etype=? , industry=? , address=? , pincode=?";

```

```

=?, executive=?, phone=?, location=? WHERE eid=?";
48
49
50     if ($stmt = mysqli_prepare($db1, $sql)) {
51         // Bind variables to the prepared statement as parameters
52         mysqli_stmt_bind_param($stmt, "ssssisssi", $name, $type, $industry,
53         $addr, $pin, $person, $phone, $location, $_SESSION['eid']);
54
55         // Attempt to execute the prepared statement
56         if (mysqli_stmt_execute($stmt)) {
57             // Redirect to profile page
58             header("location: profile.php");
59             exit();
60         } else {
61             echo "Something went wrong. Please try again later.";
62         }
63     }
64
65     // Close statement
66     mysqli_stmt_close($stmt);
67
68 // Close connection
69 mysqli_close($db1);

```

Session Start: The script begins by starting a new session or resuming the current one using `session_start()`. This is necessary for accessing session variables, which are used for maintaining state across different pages.

Database Connection: It includes the database configuration file `../config.php` to establish a connection to the database. This file likely contains database connection details and is placed one directory up from the current script.

Form Submission Check: The script checks if the form has been submitted using the POST method. This is done by checking the `$_SERVER["REQUEST_METHOD"]` variable.

Retrieving Form Data: Upon form submission, it retrieves the employer's profile information from the POST request. This includes the employer's name, type, industry, address, pin code, executive name, phone number, and location details (country, state, city IDs).

Fetching Location Data: The script then selects the location database using `mysqli_select_db($db2, "location")` and fetches the names of the country, state, and city based on the IDs provided in the form. These names are concatenated to form a complete location string.

Updating the Employer's Profile: After fetching the location details, the script switches the database context to `jobportal` using `mysqli_select_db($db1, "jobportal")`. It prepares an SQL UPDATE statement to update the employer's profile with the new data. The `mysqli_prepare()` function is used to prepare the SQL statement, and `mysqli_stmt_bind_param()` is used to bind the form data to the prepared statement's parameters.

Executing the Update Statement: The script attempts to execute the prepared statement

using mysqli_stmt_execute(). If the execution is successful, it redirects the user to the profile page using the header("location: profile.php") function. If the execution fails, it displays an error message.

Closing Resources: Finally, the script closes the prepared statement and the database connection using mysqli_stmt_close(\$stmt) and mysqli_close(\$db1), respectively, to free up resources.

4 Front-end Programming

4.1 Main Page

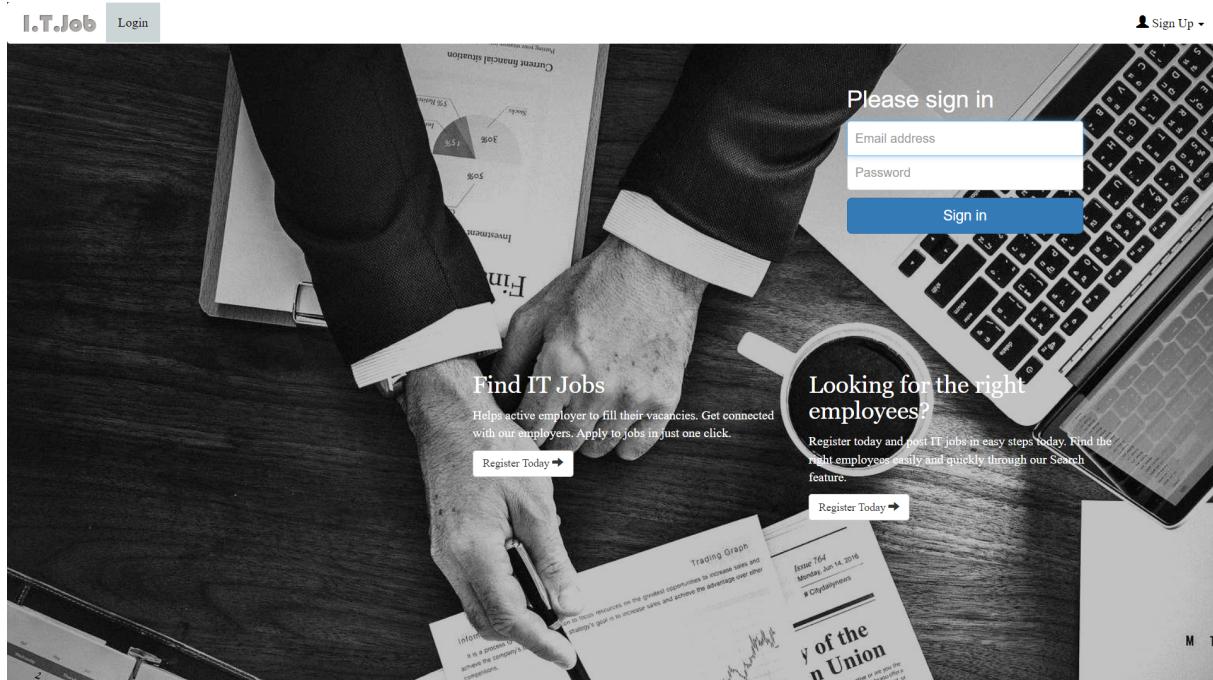
The screenshot shows the homepage of a job search platform named "I.T.Job". At the top, there is a navigation bar with links for "Home", "Register", and "Login". The main heading is "See what jobs are available:" followed by a sub-heading "Jobs available:". Below this is a search bar with a placeholder "Enter your search keyword" and a "Search" button. A promotional section below the search bar claims "The best part? Everything." and lists three benefits: "Easily find jobs.", "Easily find employees.", and "Support always open.". Two large buttons are present: "Become an Employer" on the left and "Become a Jobseeker" on the right. Each button has a corresponding "Sign Up" button below it. At the bottom of the page, there is a "CONTACT US" section with contact information: "Contact us and we'll get back to you within 24 hours.", "Binh Duong, Viet Nam", "+84 976123456", and "knghitv@gmail.com". A "Back to top" link is located at the very bottom.

The main page has the fewest functions of all pages. Its only significance function is the search function. This function will take input from user as keyword to search for the name of the job and when user hit the search button, it will make a GET HTTP(AJAX) request to a server-side script (home_search.php) with the keyword as the parameter. The function will display "Searching.." as it executes the request in various states. Upon completion, the results or an error message will pop up Employment of the search function:

```
1 function search() {
```

```
2     var keyword=document.getElementById("keyword").value;
3     // alert(keyword);
4     var xmlhttp;
5     if (window.XMLHttpRequest) { // for IE7+, Firefox, Chrome, Opera, Safari
6         xmlhttp = new XMLHttpRequest();
7     } else { // for IE6, IE5
8         ...
9     }
```

4.2 Login Page



The login page, as its name states, it has the function for users to enter their credentials to access the page. Also, there are hyperlinks to go for registration. These function will be expressed later as it is related to the back-end rather than front-end

4.3 Jobseeker Registration Page

The screenshot shows the 'Jobseeker Registration' page of the I.T.Job website. At the top, there is a navigation bar with the logo 'I.T.Job', the text 'Jobseeker Registration', and a 'Login' button. The main heading 'Register & Find IT jobs' is displayed prominently in a large, bold font. Below it, a sub-instruction reads 'Helps active employer to fill their vacancies. Get connected with our employers. Apply to jobs in just one click'. The form is divided into several sections:

- Your Login Details:** Contains fields for 'Enter your Email ID:' (with placeholder 'Your E-mail'), 'Create new Password:' (with placeholder 'New Password'), and 'Confirm the Password:' (with placeholder 'Confirm Password').
- Your Contact Information:** Contains fields for 'Mention your Full Name:' (with placeholder 'Your Name'), 'Where are you currently located?' (with dropdowns for 'Select Country', 'Select State', and 'Select City'), and 'Enter your Mobile number:' (with placeholder 'Mobile number').
- Your Current Employment Details:** Contains fields for 'How much work experience do you have:' (with dropdown placeholder 'select') and 'What are your Key Skills:' (with four dropdowns labeled 'Select Skill').
- Your Educational Qualifications:** Contains fields for 'Your Basic Education:' (with dropdown placeholder 'Select') and 'Your Masters Education:' (with dropdown placeholder 'Select').
- At the bottom, there are two buttons: a green 'Register' button and a red 'Reset' button.

The front-end allows user to input their information such as names, phone number, level of education, etc., in order to complete their registration as well as set up their profile. Also, there are two buttons for submitting and resetting the form. However, apart from inputting, other functions that are relevant to registration is more on the side of back-end.

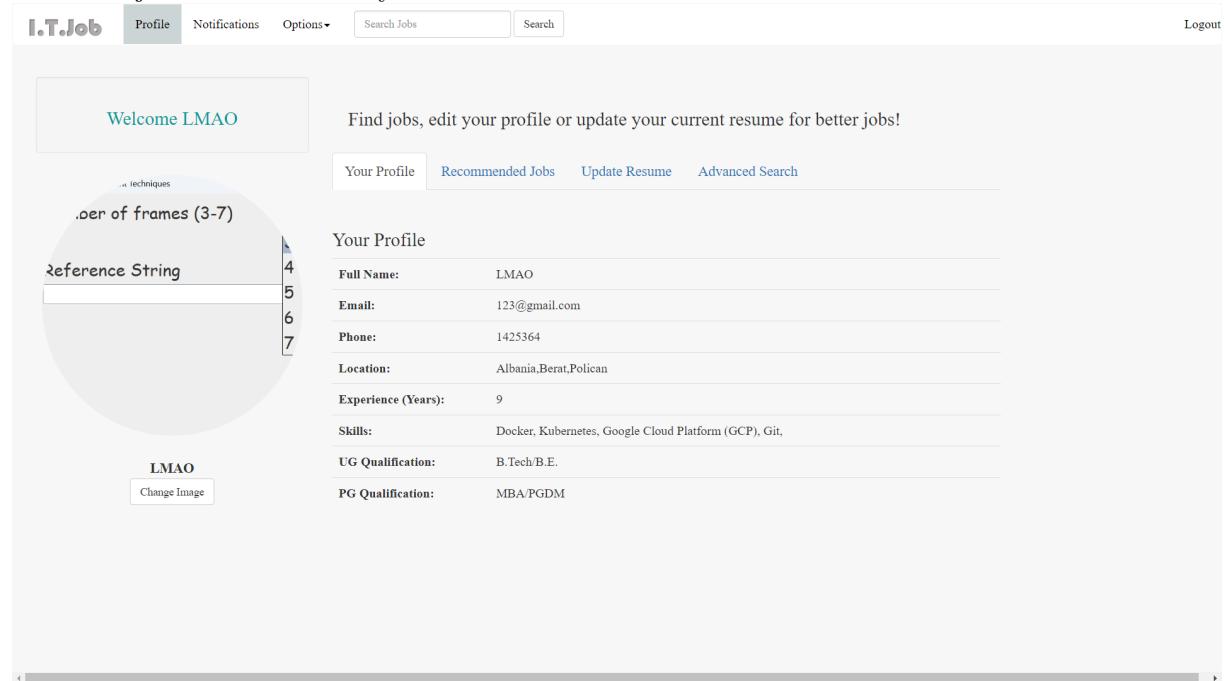
4.4 Employer Registration Page

The screenshot shows the 'Employer Registration' page. At the top left is the 'I.T.Job' logo, and at the top right is a 'Login' link. The main title 'Register as Employer' is centered above a brief description: 'Register today and post IT jobs in easy steps today. Find the right employees easily and quickly through our Search feature.' Below this, under 'Your Login details', there are three input fields: 'E-mail:' (with placeholder 'Enter your email'), 'Password:' (with placeholder 'Enter your password'), and 'Confirm Password:' (with placeholder 'Confirm your password'). Under 'Your Company Details', there are several more fields: 'Company Name:' (placeholder 'Enter Company Name'), 'Company Type:' (radio buttons for 'Company' and 'Consultant'), 'Industry:' (dropdown menu with placeholder 'Select an Industry'), 'Address:' (text area), 'Pincode:' (placeholder 'Enter the pincode'), 'Contact Person:' (placeholder 'Enter Executive Name'), 'Contact Number:' (placeholder 'Enter Contact Number'), and 'Where are you currently located?' (three dropdown menus for 'Select Country', 'Select State', and 'Select City'). At the bottom, a note says 'Check for errors before submitting the form!' followed by 'Register' and 'Reset' buttons.

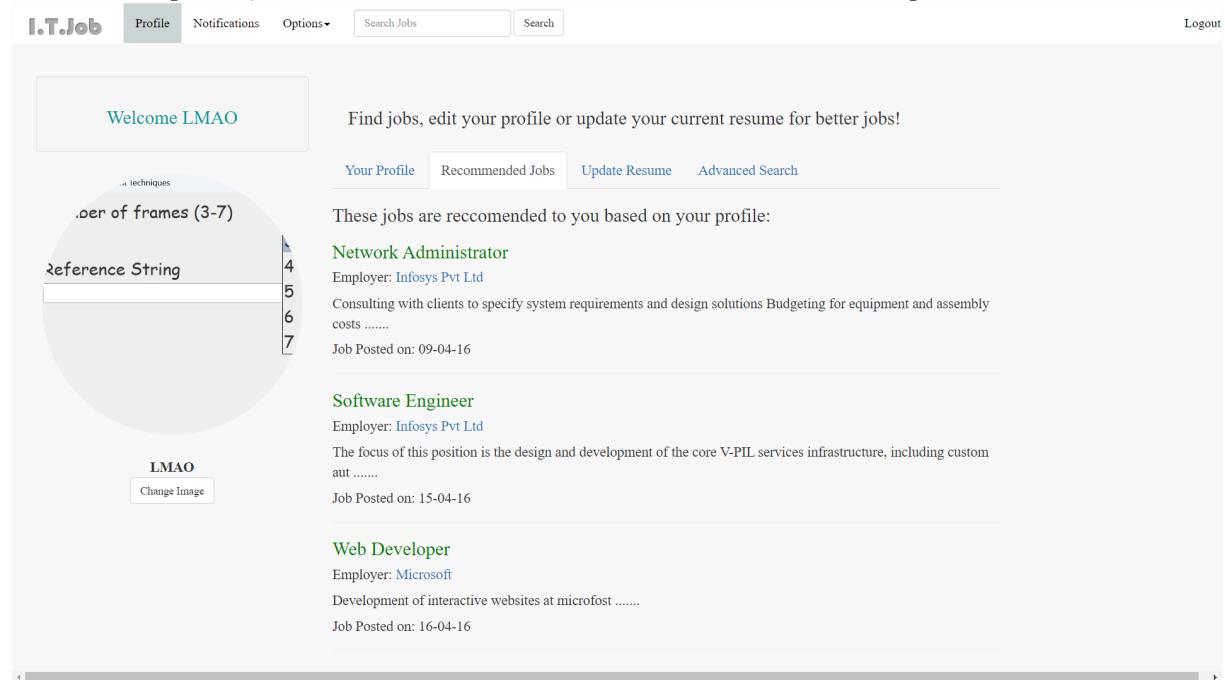
Similar to the Jobseekers' registration, the one of employers also include a form to fill in, buttons to submit and reset while the other functions lean to back-end.

4.5 Jobseeker Profile

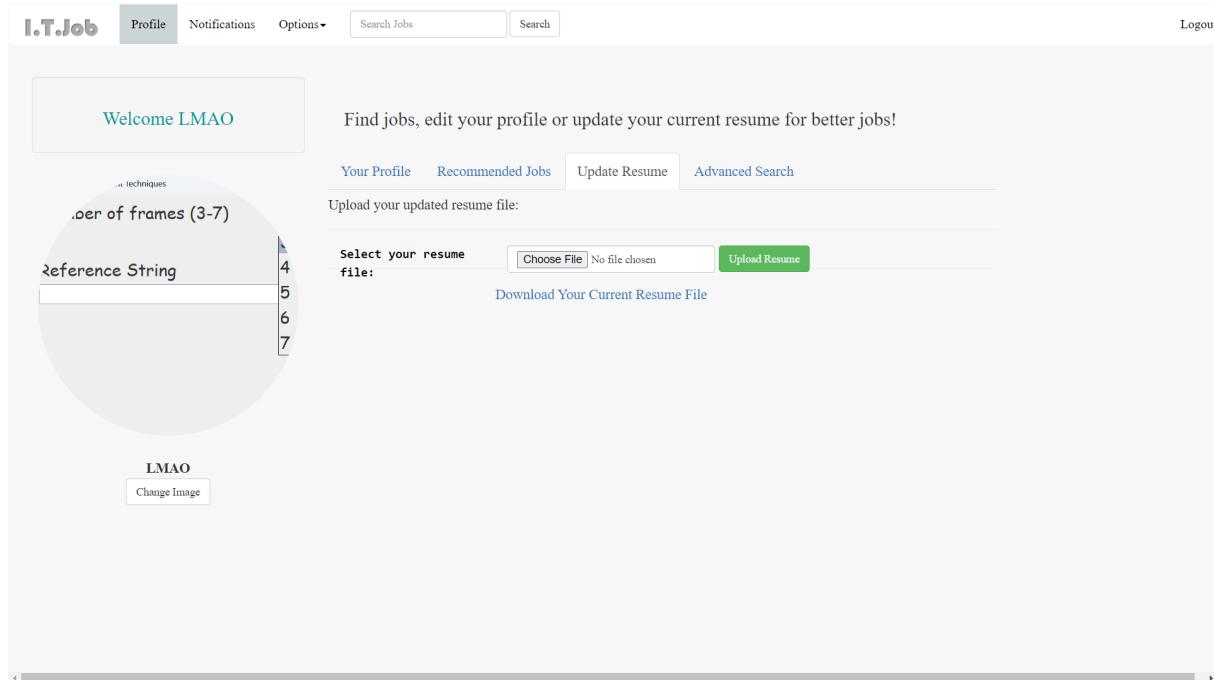
Profile of jobseekers has many function in front-end.



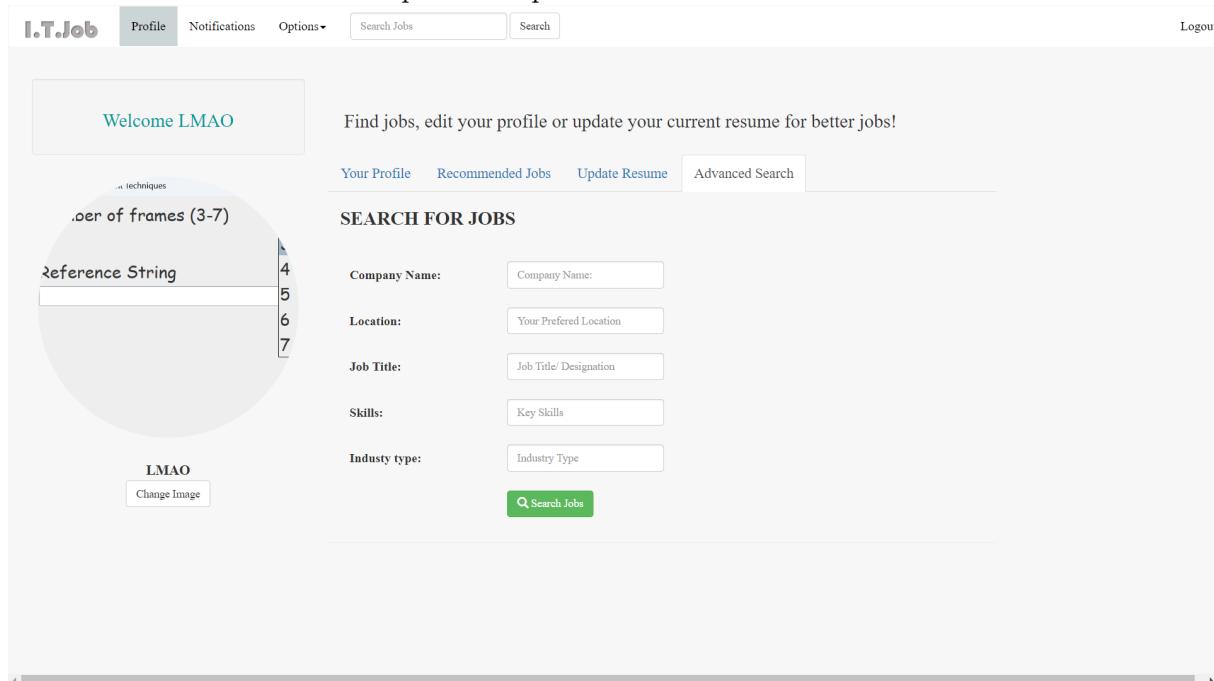
In the main profile, user can view their information as well as choose options for other tasks.



This profile function allow users to see recommended jobs or position based on their provided information. How the recommendation work will be explained in later in this report



This function enables user to upload or update their CV.



The advance search function, is used to perform an asynchronous HTTP (AJAX) request to a server-side script (adv_search.php) with parameters for company, location, designation, and skills. It dynamically updates the content of an element with the ID subcontent on the webpage without reloading the page. The function handles different states of the request by displaying "Searching.." during the process and showing the result or an error message upon completion.

```

1 function advsearch(){
2     var comp=document.getElementById("company").value;
3     var location=document.getElementById("location").value;
4     ...
5 }
```

I.T.Job [Update Information](#) [Back](#)

Update your information

Your Contact Information

Mention your Full Name:

Where are you currently located?

Enter your Mobile number:

Your Current Employment Details

How much work experience do you have:

What are your Key Skills:

Your Educational Qualifications

Your Basic Education:

Your Masters Education:

[submit](#) [Reset](#)

This function allows user to update their profile by fill in the information the same as the registration form. After that they can choose to proceed to completion, or reset. User also can choose to go back.

I.T.Job LMAO Notifications [View Applied Job](#) Options [Search](#) Logout

You Applied for these jobs

Project Management
Employer: CTTNHHMTV
eheqbeqbewgagaegageqbbxbxxb.....
Job Posted on: 18-06-24
Applied on: 18-06-24

Software Engineer
Employer: Infosys Pvt Ltd
The focus of this position is the design and development of the core V-PIL services infrastructure, including custom aut
Job Posted on: 15-04-16
Applied on: 18-06-24

Users can view the list of jobs they applied for along with a brief description, time posted and time applied via this function. And by clicking to a specific job, user can see the job details.

The screenshot shows a user interface for managing job offers. At the top, there are navigation links: I.T.Job, LMAO, Notifications, Selected Jobs, Options, a search bar, and a Logout link. A message in the center says, "You have got selection for these job" and "You can reject a job offer if you are not interested." Below this is a table with one row of data:

Employer	Job Title	Job Detail	Selection Date	Action
CTTNHHMTV	Project Management	eheqbeqbewgaaeageqbxxfbxxxx.....	18-06-24	<button>Reject Job</button>

The view selected job function helps users to view a job offer made to them from an employer, user can also choose to reject or accept the offer.

4.6 Employer Profile

Similar to the jobseeker profile, employer profile also has many functions.

The screenshot shows the employer profile section. At the top, there are navigation links: I.T.Job, Profile, Notifications (0), Menu, Account, and Logout. On the left, there's a sidebar with a welcome message, job posting instructions, and a dropdown menu for page replacement techniques. The main area displays company information:

Name:	CTTNHHMTV
Type:	Consultant
Industry:	Banking/Financial Services/Stockbroking/Securities
Address:	NOJFVOJENVNJEJVJEFNVJWGFSSBFB
Pincode:	12312431241
Executive Name:	MEWEGFWF
Phone Number:	13302352657259
Email:	12345@gmail.com
Main Location:	Chad.Lac.Bol

Unlike the jobseeker profile, the employer does not have as many tabs. However, it still allows the user to view their information as well as perform other functions.

I.T.Job Employer Registration Back

Update your Employer's information

Your Company Details

Company Name:

Company Type: Company Consultant

Industry:

Address:

Pincode:

Contact Person:

Contact Number:

Where are you currently located?

About Company:

Check for errors before submitting the form!

This function of employer is close to that of jobseeker, user fill in a form and finish by submitting the updated form. They can also proceed to reset or go back.

IT.Job CTTNHHMTV Job Posting Notifications Menu Account Logout

POST JOBS

Job Details:

Job Title/ Designation:	<input type="text"/>		
Number of vacancies:	<input type="text"/>		
Job Description:	<input type="text"/>		
Work Experience:	<input type="text"/> Minimum Years		
Basic Pay:	<input type="text"/> Rs		
Functional Area:	<input type="text"/>		
Location:	<input type="text"/> Select Country	<input type="text"/> Select State	<input type="text"/> Select City
Industry:	<input type="text"/> - Select an Industry -		

Desired Candidate Profile:

Specify UG Qualification:	<input type="text"/> Select
Specify PG Qualification:	<input type="text"/> Select
Desired Candidate Profile	<input type="text"/>

Are you sure to submit the job! Check for errors before submitting the job

Post Job

This function helps employer to post a job by fill in information and requirements in the form of a form. The user then hit the post job button to submit the post.

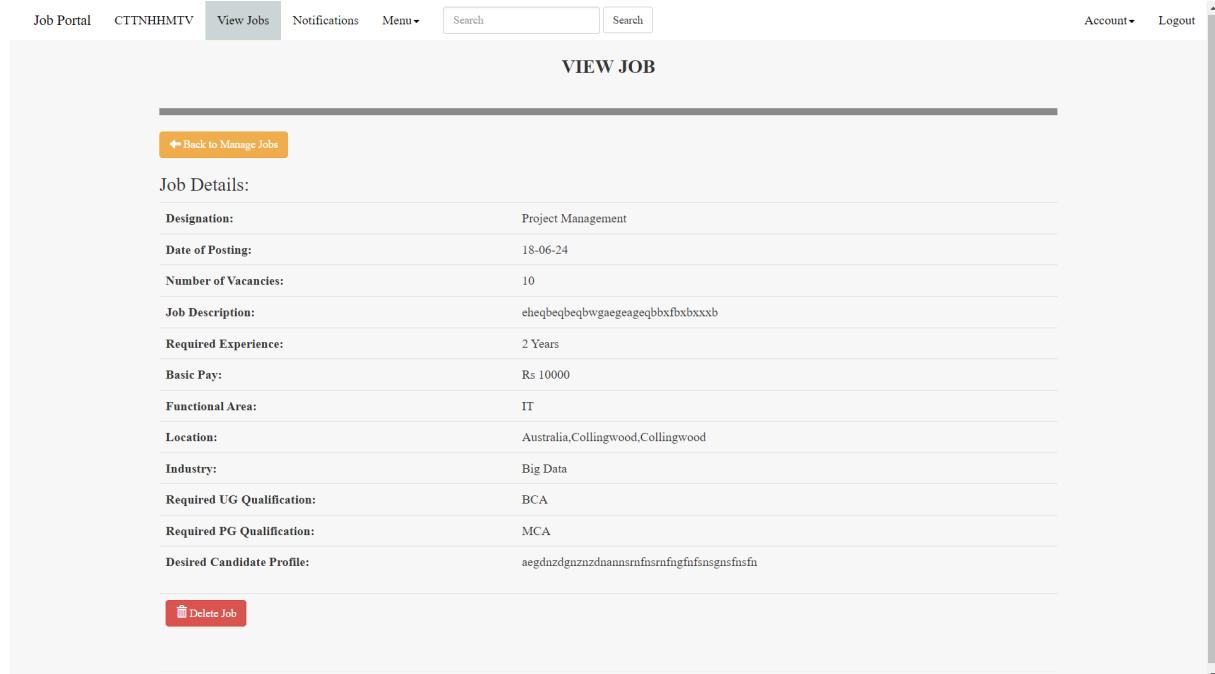
IT.Job CTTNHHMTV Manage Jobs Notifications Menu Account Logout

Manage Your Posted Jobs

Job Title	Job Description	Date of Posting	Actions
Project Management	eheqbqbeqbwgaegeageqbfbxbxxb	18-06-24	View Job View Applicants

By using this function, user can view the list of jobs with a brief description which they have

already posted. Employer can further proceed by choosing to view job details or view applicants.



VIEW JOB

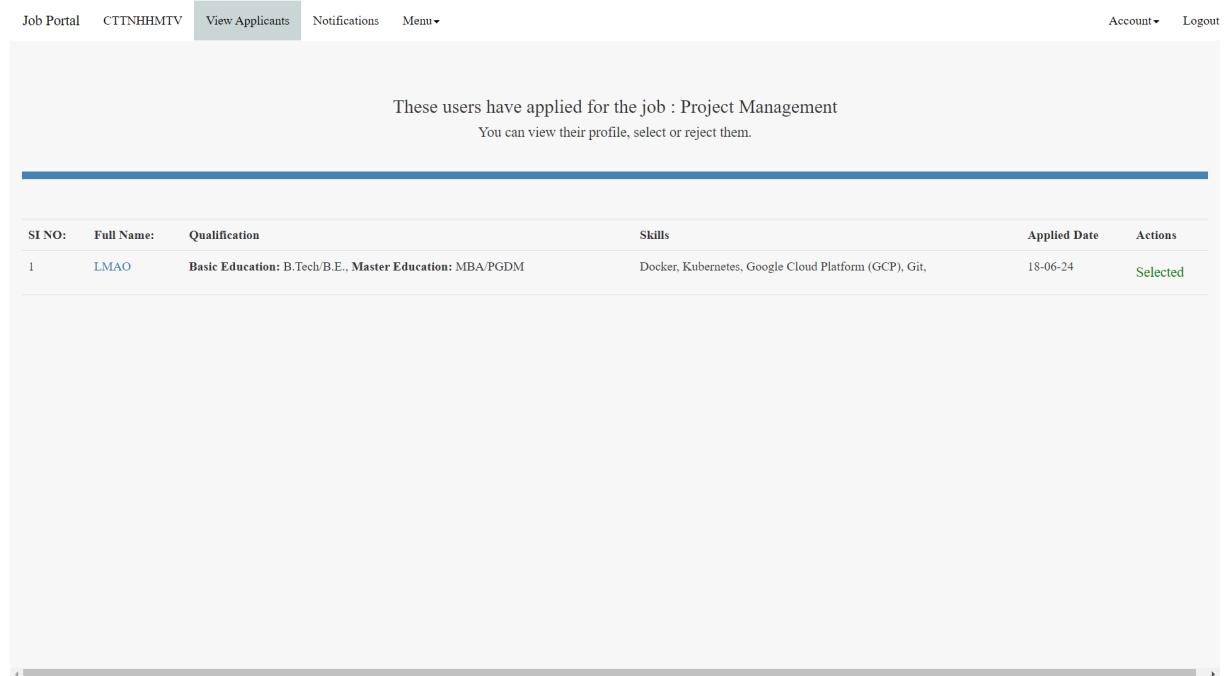
[Back to Manage Jobs](#)

Job Details:

Designation:	Project Management
Date of Posting:	18-06-24
Number of Vacancies:	10
Job Description:	eheqbeqbewgaaegeaqbbxbxxxx
Required Experience:	2 Years
Basic Pay:	Rs 10000
Functional Area:	IT
Location:	Australia,Collingwood,Collingwood
Industry:	Big Data
Required UG Qualification:	BCA
Required PG Qualification:	MCA
Desired Candidate Profile:	aegdnzdgaznzdhaansrnfnfnsfngfifnsngnsfnsfa

[Delete Job](#)

The function shows the employer their job's detailed description. It also allow user to delete the posted job or go back to the list.



These users have applied for the job : Project Management
You can view their profile, select or reject them.

SI NO:	Full Name:	Qualification	Skills	Applied Date	Actions
1	LMAO	Basic Education: B.Tech/B.E., Master Education: MBA/PGDM	Docker, Kubernetes, Google Cloud Platform (GCP), Git,	18-06-24	Selected

The employer can see the list of the applicants for their jobs. They can also view the applicants education, qualifications and skills. Employer can either reject or accept the jobseeker

5 Back-end Testing

5.1 Search Jobs

5.1.1 Test case 1: Keyword search successful

```

1 $keyword= $_GET['key'];
2 $query = "select * from jobs join employer on jobs.eid = employer.eid where
    title LIKE '%" . $keyword . "%' or employer.ename LIKE '%". $keyword . "%' or
    employer.profile LIKE '%" . $keyword . "%' ";
3 $result = mysqli_query($db1, $query);

```

Listing 1: Code for querying database for search function

This code will take the keyword enter by the user and send a query with that keyword to the database to search for the jobs. In this test case the database should return a instances containing the keyword.

Assuming our keyword is Microsoft, we will use the following sql query to query the database:

```

1 select * from jobs join employer on jobs.eid = employer.eid where title LIKE
    'Microsoft' or employer.ename LIKE 'Microsoft' or employer.profile LIKE '
    Microsoft';

```

Listing 2: SQL Code for querying database for search function

The screenshot shows the MySQL Workbench interface with the following details:

- Query Result:** Shows a green checkmark and the message "Showing rows 0 - 0 (1 total, Query took 0.0008 seconds.)".
- SQL Query:** The query used is: `select * from jobportal.jobs join jobportal.employer on jobs.eid = employer.eid where title LIKE 'Microsoft' or employer.ename LIKE 'Microsoft' or employer.profile LIKE 'Microsoft';`
- Result Table:** A table with columns: jobid, eid, title, jobdesc, vacno, experience, basicpay, fnarea, location, industry, ugqual, pgqual. One row is shown for a job with eid 2, titled "Web Developer", located in India, Kerala, Ernakulam, with a basic pay of Rs 25000, belonging to the Software Services industry, and having a B.Tech/B.E. qualification.

5.1.2 Test case 2: Keyword search return null

In this test case the database should return nothing.

Assuming our keyword is google, we will use the following sql query to query the database:

```

1 select * from jobs join employer on jobs.eid = employer.eid where title LIKE
    'Google' or employer.ename LIKE 'Google' or employer.profile LIKE 'Google';

```

Listing 3: SQL Code for querying database for search function

The screenshot shows the MySQL Workbench interface with the following details:

- Query Result:** Shows a green checkmark and the message "MySQL returned an empty result set (i.e. zero rows). (Query took 0.0004 seconds.)".
- SQL Query:** The query used is: `select * from jobs join employer on jobs.eid = employer.eid where title LIKE 'Google' or employer.ename LIKE 'Google' or employer.profile LIKE 'Google';`
- Result Table:** An empty table with columns: jobid, eid, title, jobdesc, vacno, experience, basicpay, fnarea, location, industry, ugqual, pgqual, profile, postdate, eid, log.
- Operations:** A button labeled "Query results operations" with a dropdown menu showing "Create view".

5.2 Register as Employer

5.2.1 Test case 1: Register Successful

```

1 $query4= "INSERT INTO login (email,password,usertype,status) VALUES ('$email','
    $hash','employer',0)";

2

3 $query5 = "INSERT INTO employer (log_id,ename,phone,location,etype,address,
    pincode,executive,industry)
        VALUES ((SELECT log_id FROM login WHERE email='$email'),'$name
    ','$phone','$location','$type','$addr','$pin','$person','$industry')";
```

Listing 4: Code for querying database for register process

This code inserts a new employer record into the database. In this test case, a new user should be added as an employer to the database.

When a new employer registers on our website by creating an account:

```

1 INSERT INTO login (email,password,usertype,status) VALUES ('khoinguyen@gmail.com
    ','123','employer',0);
```

Listing 5: SQL Code for querying database for register process

✓ 1 row inserted.
Inserted row id: 41 (Query took 0.0067 seconds.)

INSERT INTO login (email,password,usertype,status) VALUES ('khoinguyen@gmail.com','123','employer',0);

log_id	email	password	usertype	status
41	khoinguyen@gmail.com	123	employer	0

```

1 INSERT INTO employer (log_id,ename,phone,location,etype,address,pincode,
    executive,industry)
        VALUES ((SELECT log_id FROM login WHERE email='khoinguyen@gmail.
    com'),'TNHH','0767471717','Vietnam','Company','Vietnam','103123','DaiMinh',
    'Steel');
```

Listing 6: SQL Code for querying database for register process

✓ 1 row inserted.
Inserted row id: 7 (Query took 0.0050 seconds.)

INSERT INTO employer (log_id,ename,phone,location,etype,address,pincode,executive,industry)VALUES
((SELECT log_id FROM login WHERE
email='khoinguyen@gmail.com'),'TNHH','0767471717','Vietnam','Company','Vietnam','103123','DaiMinh','Ste
el');

eid	log_id	ename	etype	industry	address	pincode	executive	phone	location
7	41	TNHH	Company	Steel	Vietnam	103123	DaiMinh	0767471717	Vietnam

5.2.2 Test case 2: Register fail with missing information

In this test case, a new user shouldn't be added as an employer to the database.

```
1 INSERT INTO login (email,password,userstype,status) VALUES ('khoinguyen@gmail.com'
, '123', ,0);
```

Listing 7: SQL Code for querying database for register process

Error

SQL query: [Copy](#)

```
INSERT INTO login (email,password,userstype,status) VALUES ('minhtri@gmail.com','123', ,0);
```

5.3 Register as Jobseeker

5.3.1 Test case 1: Register Successful

```
1 $query4="INSERT INTO login (email,password,userstype,status) VALUES ('$email','
$query4=$hash','$type',1)";
2 $result1 = mysqli_query($db1,$query4) or die("Cant Register , The user email
may be already existing");
3 $query5 = "INSERT INTO jobseeker (log_id,name,phone,location,experience,skills,
basic_edu,master_edu)
VALUES ((SELECT log_id FROM login WHERE email='$email'),'$name
','$mobile','$location','$experience','$skills','$ug','$pg')";
```

Listing 8: Code for querying database for register process

This code inserts a new jobseeker record into the database. In this test case, a new user should be added as an jobseeker to the database.

When a new jobseeker registers on our website by creating an account:

```
1 INSERT INTO login (email,password,userstype,status) VALUES ('minhtri@gmail.com','
123','jobseeker',1);
```

Listing 9: SQL Code for querying database for register process

```
1 INSERT INTO jobseeker (log_id,name,phone,location,experience,skills,basic_edu ,
master_edu) VALUES ((SELECT log_id FROM login WHERE email='minhtri@gmail.
com'),'Minh Tri','0767470100','Vietnam','3 Years','C++','BA','CS');
```

Listing 10: SQL Code for querying database for register process

✓ 1 row inserted.

Inserted row id: 44 (Query took 0.0056 seconds.)

```
INSERT INTO login (email,password,userstype,status) VALUES ('Daiminh@gmail.com','123','CEO',1);
```

✓ 1 row inserted.

Inserted row id: 20 (Query took 0.0060 seconds.)

```
INSERT INTO jobseeker (log_id,name,phone,location,experience,skills,basic_edu,master_edu) VALUES
((SELECT log_id FROM login WHERE email='Daiminh@gmail.com'),'Dai Minh','0767879198','Vietnam','3
Years','C++','B.A','CS');
```

5.3.2 Test case 2: Register fail with missing information

In this test case, a new user shouldn't be added as an employer to the database.

```
1 INSERT INTO login (email,password,userstype,status) VALUES ('minhtri1@gmail.com',
  '123','jobseeker',,1);
```

Listing 11: SQL Code for querying database for register process

Error

SQL query: [Copy](#)

```
INSERT INTO login (email,password,userstype,status) VALUES ('minhtri1@gmail.com','123','jobseeker',,1);
```

5.4 Edit profile

5.4.1 Test case 1: Edit Profile Jobseeker

```
1 // Prepare an UPDATE statement
2 $sql = "UPDATE jobseeker SET name=?, location=?, phone=?, experience=?,
skills=?, basic_edu=?, master_edu=? WHERE user_id=?";
3 if ($stmt = mysqli_prepare($db1, $sql)) {
4     // Bind variables to the prepared statement as parameters
5     mysqli_stmt_bind_param($stmt, "sssisssi", $name, $location, $phone,
6     $experience, $skills, $basic_edu, $master_edu, $_SESSION['jsid']);
7
8     // Attempt to execute the prepared statement
9     if (mysqli_stmt_execute($stmt)) {
10         // Redirect to profile page
11         header("location: profile.php");
12         exit();
13     } else {
14         echo "Something went wrong. Please try again later.";
15     }
}
```

Listing 12: Code for querying database for edit process

This code update a new jobseeker record into the database. In this test case, information of old user should be updated.

```
1 UPDATE jobseeker SET name='Minh Dai', location='Vietnam', phone='0976854344',
experience='3', skills='C++', basic_edu='B.A' WHERE user_id=20;
```

Listing 13: SQL Code for querying database for edit process

```
✓ 1 row affected. (Query took 0.0076 seconds.)

UPDATE jobseeker SET name='Minh Dai', location='Vietnam', phone='0976854344', experience='3', skills='C++', basic_edu='B.A' WHERE user id=20;
```

5.4.2 Test case 2: Edit Profile Employer

```

1 // Prepare an UPDATE statement
2 $sql = "UPDATE employer SET ename=?, etype=?, industry=?, address=?, pincode=?,
3 , executive=?, phone=?, location=? WHERE eid=?";
4
5 if ($stmt = mysqli_prepare($db1, $sql)) {
6     // Bind variables to the prepared statement as parameters
7     mysqli_stmt_bind_param($stmt, "ssssisssi", $name, $type, $industry,
8     $addr, $pin, $person, $phone, $location, $_SESSION['eid']);
9
10    // Attempt to execute the prepared statement
11    if (mysqli_stmt_execute($stmt)) {
12        // Redirect to profile page
13        header("location: profile.php");
14        exit();
15    } else {
16        echo "Something went wrong. Please try again later.";
17    }
}
```

Listing 14: Code for querying database for edit process

This code update a new employer record into the database. In this test case, information of old user should also be updated.

```
1 UPDATE employer SET ename='LMFAO', etype='asdfasdf', industry='asdfasdf',
2 address='adsasdf', pincode='1231', executive='MAO', phone='0978586456',
3 location='Vietnam' WHERE eid=5;
```

Listing 15: SQL Code for querying database for edit process

```
✓ 1 row affected. (Query took 0.0048 seconds.)

UPDATE employer SET ename='LMFAO', etype='asdfasdf', industry='asdfasdf', address='adsasdf', pincode='1231', executive='MAO',
phone='0978586456', location='Vietnam' WHERE eid=5;
```

5.5 Job application

5.5.1 Test case 1: Job application is already in

```

1 $q1=mysqli_query($db1,"select * from application where job_id = $jobid AND
2 user_id = $jsid");
3 if(mysqli_num_rows($q1)>=1){
4     echo " <div class='alert alert-danger alert-dismissible' role='alert'>
5 
```

```

4         <button type='button' class='close' data-dismiss='alert' aria-label
5             ='Close'><span
6                 aria-hidden='true'>&times;</span></button>
7             <p style='font-size: 20px'><strong>Note:</strong> You have already
8             applied for this job!</p>
9         </div>";
}

```

Listing 16: Code for querying database for application process

This code will test job application. In this case, there is already a job application so the query should return a result and the frontend will display the alert.

```
1 select * from application where job_id =2 AND user_id = 8;
```

Listing 17: SQL Code for querying database for application process

✓ Showing rows 0 - 0 (1 total, Query took 0.0025 seconds.)

```
select * from application where job_id =2 AND user_id = 8;
```

5.5.2 Test case 2: Job application is new

```

1 else{
2     $q2=mysqli_query($db1,"insert into application (user_id,emp_id,job_id,
3         date_applied) VALUES ($jsid,(SELECT eid from jobs where jobid = $jobid),
4         $jobid,'$date')");
5     if($q2){
6         echo " <div class='alert alert-success alert-dismissible' role='alert'>
7             <button type='button' class='close' data-dismiss='alert' aria-label
8                 ='Close'><span
9                     aria-hidden='true'>&times;</span></button>
10            <p style='font-size: 20px'><strong>Note:</strong> You have
11            successfully applied for this job!</p>
12        </div>";
13    }
}

```

Listing 18: Code for querying database for application process

This code will insert into the database application the user and will notify about the successful application.

```
1 insert into application (user_id,emp_id,job_id,date_applied) VALUES (10,(SELECT
2     eid from jobs where jobid = 2),2,'18-04-16');
```

Listing 19: SQL Code for querying database for register process

✓ 1 row inserted.

Inserted row id: 13 (Query took 0.0060 seconds.)

```
insert into application (user_id,emp_id,job_id,date_applied) VALUES (10,(SELECT eid from jobs where jobid = 2),2,'18-04-16');
```

5.6 Post jobs

5.6.1 Test case 1: Post Jobs successful

```

1 $query4="insert into jobs (eid,title,jobdesc,vacno,experience,basicpay,fnarea,
2   location,industry,ugqual,pgqual,profile,postdate )VALUES ('$eid','$desig',
3   '$desc','$vacno','$exp','$pay','$fnarea','$location','$indtype','$ug','$pg',
4   '$profile','$date')";

5 if (!mysqli_query($db1,$query4))
6 {
7 echo("Error description: " . mysqli_error($db1));
8 }

```

Listing 20: Code for querying database for post jobs process

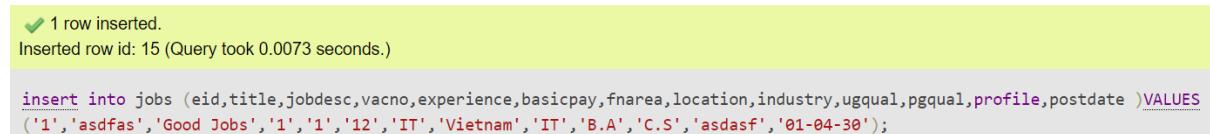
This code will insert the job listing into the database. In this case, a successful insert will appear.

```

1 insert into jobs (eid,title,jobdesc,vacno,experience,basicpay,fnarea,location,
2   industry,ugqual,pgqual,profile,postdate )VALUES ('1','asdfas','Good Jobs','1',
3   '1','12','IT','Vietnam','IT','B.A','C.S','asdASF','01-04-30');

```

Listing 21: SQL Code for querying database for post jobs process



1 row inserted.
Inserted row id: 15 (Query took 0.0073 seconds.)

insert into jobs (eid,title,jobdesc,vacno,experience,basicpay,fnarea,location,industry,ugqual,pgqual,profile,postdate)VALUES ('1','asdfas','Good Jobs','1','1','12','IT','Vietnam','IT','B.A','C.S','asdASF','01-04-30');

5.6.2 Test case 2: Post Jobs unsuccessful

In this test case, the query will have some parameter missing so it should return error.

```

1 insert into jobs (eid,title,jobdesc,vacno,experience,basicpay,fnarea,location,
2   industry,ugqual,pgqual,profile,postdate )VALUES ('1','asdfas','Good Jobs','1',
3   '1','12','IT','Vietnam','IT','B.A','C.S','asdASF',);

```

Listing 22: SQL Code for querying database for post jobs process



Error
SQL query: Copy

insert into jobs (eid,title,jobdesc,vacno,experience,basicpay,fnarea,location,industry,ugqual,pgqual,profile,postdate)VALUES ('1','asdfas','Good Jobs','1','1','12','IT','Vietnam','IT','B.A','C.S','asdASF',);

5.7 Delete jobs

5.7.1 Test case 1: Delete successful

```

1 $query = "DELETE FROM jobs WHERE jobid=$_GET[jid]";
2 $result = mysqli_query($db1,$query);
3 if($result) {
4   echo "<h3 style='color: green;'> Selected Job Is Successfully Deleted</h3>";
5 }

```

```
6 else{
7     echo "<h3 style='color: red;'> Failed to delete the selected job!</h3>";
8 }
```

Listing 23: Code for querying database for delete process

This code will delete the job in the database. In this case, a successful delete will appear

```
1 DELETE FROM jobs WHERE jobid=13;
```

Listing 24: SQL Code for querying database for delete process

✓ 1 row deleted. (Query took 0.0076 seconds.)

```
DELETE FROM jobs WHERE jobid=13;
```

5.7.2 Test case 2: Delete fail

In this case, a failure delete will show 0 rows deleted

```
1 DELETE FROM jobs WHERE jobid=13;
```

Listing 25: SQL Code for querying database for delete process

✓ 0 rows deleted. (Query took 0.0002 seconds.)

```
DELETE FROM jobs WHERE jobid=13;
```

6 Front-end Testing

6.1 Main Page

These are the functions of the Main Page.

URL: http://localhost/job_portal/login.php

These are the commands:

Command	Target	Value
1 ✓ open	/job_portal/	
2 ✓ set window size	652x720	
3 ✓ click	id=keyword	
4 ✓ type	id=keyword	Micro
5 ✓ click	css=.glyphicon-search	
6 ✓ mouse over	css=.glyphicon-search	
7 ✓ mouse out	css=.glyphicon-search	
8 ✓ click	css=.glyphicon-chevron-up	
9 ✓ click	css=.col-sm-6:nth-child(1).btn	
10 ✓ run script	window.scrollTo(0,713.3333129882812)	
11 ✓ click	css=.col-sm-6:nth-child(2).btn	
12 ✓ click	css=.img-fluid	

Command	Target	Value
12 ✓ click	css=.img-fluid	
13 ✓ click	linkText=Register	
14 ✓ click	linkText=Become a Jobseeker	
15 ✓ click	css=.img-fluid	
16 ✓ click	linkText=Register	
17 ✓ click	linkText=Become an Employer	
18 ✓ click	css=.img-fluid	
19 ✓ click	linkText=Login	
20 ✓ mouse over	linkText=Sign Up	
21 ✓ mouse out	linkText=Sign Up	
22 ✓ click	css=.img-fluid	

6.2 Login Page

These are the functions of the Login Page.

URL: http://localhost/job_portal/login.php

These are the commands:

Command	Target	Value
1 ✓ open	http://localhost/job_portal/login.php	
2 ✓ set window size	652x726	
3 ✓ click	linkText=Register Today	
4 ✓ click	linkText=Login	
5 ✓ click	css=.col-lg-4:nth-child(1).btn	
6 ✓ click	linkText=Login	
7 ✓ click	id=inputEmail	
8 ✓ type	id=inputEmail	12345@gmail.com
9 ✓ type	id=inputPassword	12345678
10 ✓ click	css=.btn-lg	

6.3 Jobseeker Registration Page

These are the functions of the Jobseeker Registration Page.

URL: http://localhost/job_portal/login.php

These are the commands:

	Command	Target	Value
✓ JobseekerRegistration*	1 ✓ open	http://localhost/job_portal/jobseeker/register_user.php	
✓ mainPage	2 ✓ set window size	652x726	
	3 ✓ type	id=email	Nguyen@gmail.com
	4 ✓ type	id=passnew	12345678
	5 ✓ type	id=passconf	12345678
	6 ✓ click	id=name	
	7 ✓ click	id=name	
	8 ✓ type	id=name	Nguyen Khoi Nguyen
	9 ✓ click	id=countryId	
	10 ✓ select	id=countryId	label=Vietnam
	11 ✓ click	id=stateId	
	12 ✓ select	id=stateId	label=Mien Nui Va Trung Du
✓ JobseekerRegistration*	Command	Target	Value
✓ mainPage	13 ✓ click	id=cityId	
	14 ✓ click	id=mobno	
	15 ✓ type	id=mobno	0767470100
	16 ✓ click	id=experience	
	17 ✓ select	id=experience	label=2 year
	18 ✓ click	id=skill1	
	19 ✓ select	id=skill1	label=C++
	20 ✓ click	id=skill2	
	21 ✓ select	id=skill2	label=SQL
	22 ✓ click	id=skill3	
	23 ✓ select	id=skill3	label=Ruby
	24 ✓ click	id=skill4	
✓ JobseekerRegistration*	Command	Target	Value
✓ mainPage	20 ✓ click	id=skill2	
	21 ✓ select	id=skill2	label=SQL
	22 ✓ click	id=skill3	
	23 ✓ select	id=skill3	label=Ruby
	24 ✓ click	id=skill4	
	25 ✓ select	id=skill4	label=HTML/CSS
	26 ✓ click	id=ugcourse	
	27 ✓ select	id=ugcourse	label=B.A
	28 ✓ click	id=pgcourse	
	29 ✓ select	id=pgcourse	label=CS
	30 ✓ click	id=reg	

6.4 Employer Registration Page

These are the functions of the Employer Registration Page.

URL: http://localhost/job_portal/login.php

These are the commands:

	Command	Target	Value
✓ EmpRegistration*	1 ✓ open	http://localhost/job_portal/employer/register_emp.php	
✓ JobseekerRegistration*	2 ✓ set window size	652x720	
✓ mainPage	3 ✓ click	id=email	
	4 ✓ type	id=email	Nghi@gmail.com
	5 ✓ type	id=pass1	12345678
	6 ✓ type	id=pass2	12345678
	7 ✓ type	id=compname	KhanhNghiTNHH
	8 ✓ click	css=.radio-inline:nth-child(1)	
	9 ✓ click	id=indtype	
	10 ✓ select	id=indtype	label=Brewery/Distillery
	11 ✓ click	id=indtype	
	12 ✓ click	css=form-group:nth-child(10)	

	Command	Target	Value
✓ EmpRegistration*			
✓ JobseekerRegistration*	✓ click	id=addr	
✓ mainPage	✓ type	id=addr	D9 Vanh dai 4, Binh Duong, Vietnam
	✓ click	id=recomp	
	✓ click	id=pincode	
	✓ type	id=pincode	22456
	✓ click	id=person	
	✓ type	id=person	Dai Minh
	✓ click	id=phone	
	✓ type	id=phone	0767470100
	✓ click	id=countryId	
	✓ select	id=countryId	label=Bangladesh
	✓ click	id=stateId	
✓ EmpRegistration*			
✓ JobseekerRegistration*	✓ click	id=person	
✓ mainPage	✓ type	id=person	Dai Minh
	✓ click	id=phone	
	✓ type	id=phone	0767470100
	✓ click	id=countryId	
	✓ select	id=countryId	label=Bangladesh
	✓ click	id=stateId	
	✓ select	id=stateId	label=Gopalganj
	✓ click	id=cityId	
	✓ select	id=cityId	label=Gopalganj
	✓ click	id=reg	

6.5 Employer Page

These are the functions of the Employer Page.

URL: http://localhost/job_portal/login.php

These are the commands:

	Command	Target	Value
✓ EmpProfile*			
	http://localhost/job_portal/employer/profile.php		
1	✓ open	http://localhost/job_portal/employer/profile.php	
2	✓ set window size	1296x736	
3	✓ click	linkText=Notifications	
4	✓ click	css=.glyphicon	
5	✓ click	linkText=Menu	
6	✓ click	linkText=Post Jobs	
7	✓ click	id=desig	
8	✓ type	id=desig	IT-security
9	✓ click	id=desig	
10	✓ type	id=desig	IT-security 2
11	✓ click	id=vac_no	
12	✓ type	id=vac_no	12
✓ EmpProfile*			
	http://localhost/job_portal/employer/profile.php		
	Command	Target	Value
13	✓ run script	window.scrollTo(0,124.99999237060547)	
14	✓ click	id=job_desc	
15	✓ type	id=job_desc	Require every customers
16	✓ click	css=form-inline:nth-child(4) > .col-sm-4	
17	✓ click	id=job_desc	
18	✓ type	id=job_desc	Require every customers and person with good IT skill
19	✓ click	id=exp	
20	✓ select	id=exp	label=2
21	✓ click	id=pay	
22	✓ type	id=pay	2000000
23	✓ click	id=fnarea	
24	✓ click	id=pay	

✓ EmpProfile*	http://localhost/job_portal/employer/profile.php	Command	Target	Value
25	✓ click	id=pay		
26	✓ double click	id=pay		
27	✓ type	id=pay		200000
28	✓ send keys	id=pay		\$(KEY_ENTER)
29	✓ click	id=fnarea		
30	✓ type	id=fnarea		District 7
31	✓ click	id=countryId		
32	✓ select	id=countryId		label=Saint Helena
33	✓ click	id=stateId		
34	✓ select	id=stateId		label=Ascension
35	✓ click	id=cityId		
36	✓ select	id=cityId		label=Georgetown
✓ EmpProfile*	http://localhost/job_portal/employer/profile.php	Command	Target	Value
37	✓ click	id=indtype		
38	✓ select	id=indtype		label=Quality Assurance (QA) and Testing
39	✓ click	id=ugcourse		
40	✓ select	id=ugcourse		label=B.A
41	✓ click	id=pgcourse		
42	✓ select	id=pgcourse		label=CS
43	✓ click	id=profile		
44	✓ click	css=#[job_post > .form-group:nth-child(3)]		
45	✓ click	id=profile		
46	✓ click	id=profile		
47	✓ type	id=profile		Require every customers and person with good IT skill Require every customers and person with good IT skill
✓ EmpProfile*	http://localhost/job_portal/employer/profile.php	Command	Target	Value
47	✓ type	id=profile		Require every customers and person with good IT skill Require every customers and person with good IT skill
48	✓ mouse over	id=postbtn		
49	✓ click	id=postbtn		
50	✓ click	linkText=Menu		
51	✓ click	linkText=Manage Jobs		
52	✓ click	css=id:nth-child(4).btn		
53	✓ click	css=.btn-warning		
54	✓ click	css=id:nth-child(5).btn		
55	✓ click	css=.nav:nth-child(2) > li:nth-child(1) > a		
56	✓ mouse over	css=.active > a		
57	✓ mouse out	css=.active > a		

6.6 Jobseeker Page

These are the functions of the Jobseeker Page.

URL: http://localhost/job_portal/login.php

These are the commands:

✓ EmpProfile*	:	Command	Target	Value
✓ EmpRegistration		1 ✓ open	http://localhost/job_portal/jobseeker/profile.php	
✓ JobSeekerProfile*		2 ✓ set window size	1296x736	
✓ JobseekerRegistration		3 ✓ click	linkText=Recommended Jobs	
✓ LoginPage		4 ✓ click	linkText=Update Resume	
✓ mainPage		5 ✓ click	linkText=Advanced Search	
		6 ✓ click	linkText=Your Profile	
		7 ✓ click	id=keyword	
		8 ✓ type	id=keyword	a
		9 ✓ click	css=.btn-default:nth-child(2)	
		10 ✓ click	linkText=LMFAO	
		11 ✓ click	css=.nav:nth-child(2) > li:nth-child(1) > a	
		12 ✓ click	linkText=Notifications	

	Command	Target	Value
✓ EmpProfile*			
✓ EmpRegistration	13 ✓ click	linkText=Back	
✓ JobSeekerProfile*	14 ✓ click	linkText=Options	
✓ JobseekerRegistration	15 ✓ click	linkText=Update Profile	
✓ LoginPage	16 ✓ click	id=name	
✓ mainPage	17 ✓ type	id=name	Quy Lam
	18 ✓ click	id=countryId	
	19 ✓ select	id=countryId	label=Bangladesh
	20 ✓ click	id=stateId	
	21 ✓ select	id=stateId	label=Faridpur
	22 ✓ click	id=cityId	
	23 ✓ select	id=cityId	label=Bhanga
	24 ✓ click	id=phone	
✓ EmpProfile*			
✓ EmpRegistration	25 ✓ type	id=phone	09320888222
✓ JobSeekerProfile*	26 ✓ run script	window.scrollTo(0,249.99998474121094)	
✓ JobseekerRegistration	27 ✓ click	id=experience	
✓ LoginPage	28 ✓ click	id=experience	
✓ mainPage	29 ✓ select	id=experience	label=2 year
	30 ✓ click	id=skill0	
	31 ✓ select	id=skill0	label=Amazon Web Services (AWS)
	32 ✓ click	id=skill1	
	33 ✓ select	id=skill1	label=C++
	34 ✓ click	id=skill2	
	35 ✓ select	id=skill2	label=Microsoft Azure
	36 ✓ click	id=skill3	
✓ EmpProfile*			
✓ EmpRegistration	37 ✓ select	id=skill3	label=HTML/CSS
✓ JobSeekerProfile*	38 ✓ click	id=basic_edu	
✓ JobseekerRegistration	39 ✓ select	id=basic_edu	label=B.A
✓ LoginPage	40 ✓ click	id=master_edu	
✓ mainPage	41 ✓ select	id=master_edu	label=CA
	42 ✓ click	id=reg	
	43 ✓ click	linkText=Options	
	44 ✓ click	linkText=View Applied Jobs	
	45 ✓ click	linkText=Software Engineer	
	46 ✓ click	css=.btn-success	
	47 ✓ click	css=.btn-warning	
	48 ✓ click	linkText=Options	
✓ EmpProfile*			
✓ EmpRegistration	49 ✓ click	id=master_edu	
✓ JobSeekerProfile*	50 ✓ select	id=master_edu	label=CA
✓ JobseekerRegistration	42 ✓ click	id=reg	
✓ LoginPage	43 ✓ click	linkText=Options	
✓ mainPage	44 ✓ click	linkText=View Applied Jobs	
	45 ✓ click	linkText=Software Engineer	
	46 ✓ click	css=.btn-success	
	47 ✓ click	css=.btn-warning	
	48 ✓ click	linkText=Options	
	49 ✓ click	linkText=View Selected Jobs	
	50 ✓ click	css=.nav:nth-child(2) > li:nth-child(1) > a	

7 Security

In today's digital age, user account security is paramount, and one common method to ensure this security is by storing user accounts in a local database with hashed passwords. This approach has several advantages and disadvantages, which are crucial for organizations to consider when designing their systems.

Pros:

1. Enhanced Security: Storing passwords in a hashed format significantly enhances security. Hashing algorithms convert plain-text passwords into fixed-size strings of characters, which are irreversible. This means that even if an attacker gains access to the database, they cannot easily retrieve the original passwords. Using strong, modern hashing algorithms like bcrypt or Argon2 adds layers of security, making it extremely difficult for attackers to crack the hashed passwords.

2. Control and Customization: A local database offers full control over the data and how it is managed. Organizations can implement custom security policies, such as specific hashing algorithms, salting (adding random data to passwords before hashing), and periodic rehashing to enhance security. This control also extends to access permissions, backup procedures, and data recovery processes tailored to the organization's specific needs.

3. Performance: Storing user data locally can lead to improved performance. Local databases typically offer faster data retrieval and response times compared to remote servers, especially for applications with high traffic or requiring quick authentication processes. This speed can enhance user experience and reduce latency issues.

4. Cost-Effective: Utilizing a local database can be cost-effective for small to medium-sized organizations. It eliminates the need for expensive third-party services and allows for more predictable budgeting. The costs associated with maintaining local servers and databases can be lower compared to subscription fees for cloud services, especially if the required infrastructure is already in place.

Cons:

1. Maintenance and Management: Managing a local database requires ongoing maintenance, including software updates, security patches, and hardware upkeep. Organizations must have dedicated IT staff to handle these tasks, which can be resource-intensive and costly. Failure to maintain the database properly can lead to vulnerabilities and potential data breaches.

2. Scalability Issues: As the organization grows, so does the amount of data. Scaling a local database to accommodate increasing user accounts and data can be challenging. It often requires additional hardware, storage, and network resources, which can be expensive and logistically complex. Cloud-based solutions, in contrast, offer more seamless scalability options.

3. Security Risks: While hashed passwords provide a layer of security, local databases are still susceptible to physical and cyber threats. Physical threats include theft or damage to the server hardware, while cyber threats involve hacking attempts, malware, and other forms of cyberattacks. Comprehensive security measures, such as firewalls, encryption, and intrusion detection systems, are essential but add to the complexity and cost of managing local databases.

Password Recovery:

If there is a situation where users forgot their password it may be recover by authentication via the provided email of the user. This might involve working with the database and a library where SMTP connection is allowed and email connection is available. An email containing a code to recover password will be available and once user receive it they can recover and update their passwords.

Access Control:

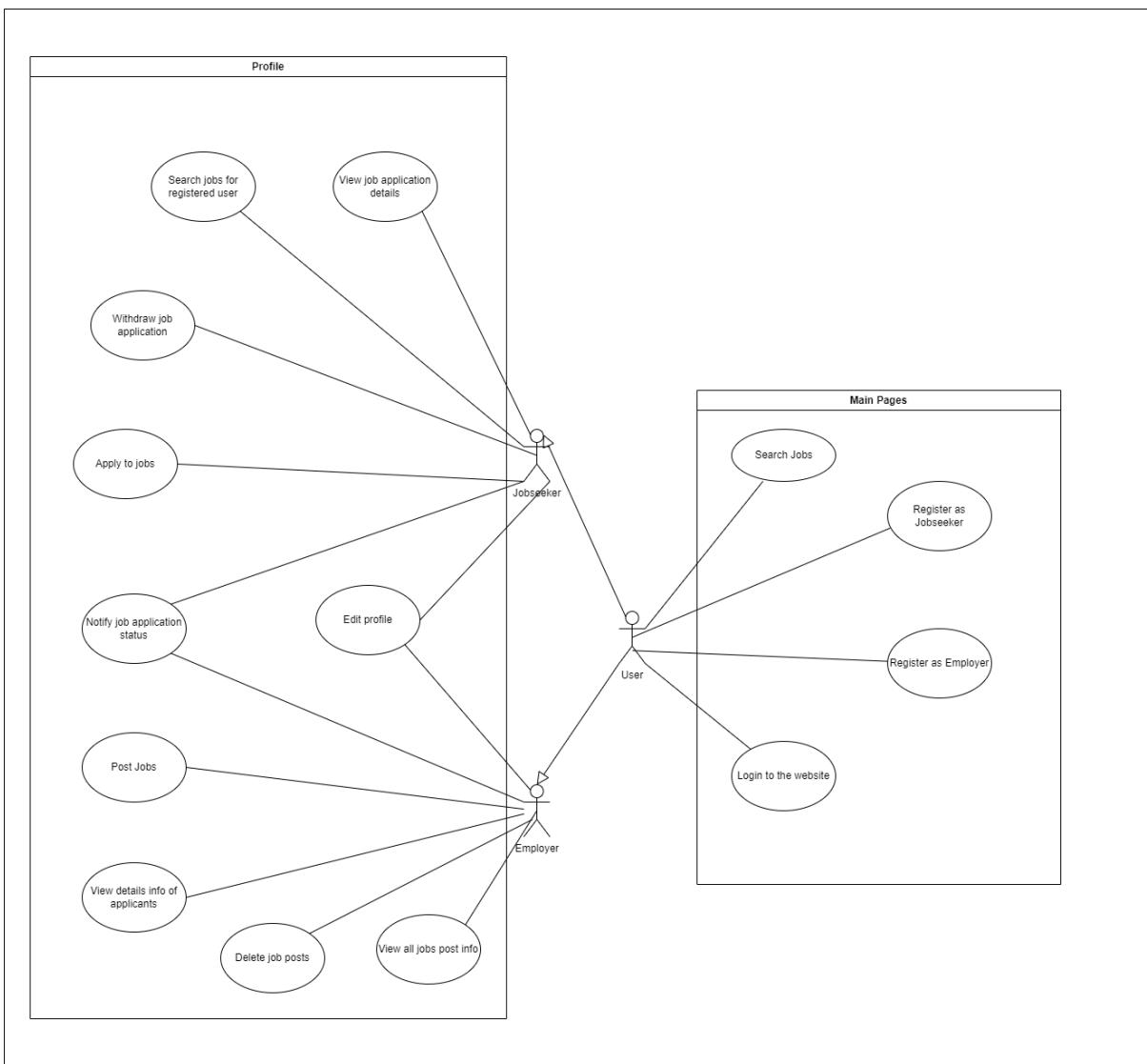
There are also certain components only available to certain users.

1. Jobseeker: Jobseeker will have access to their own user information as well as the management of their own profile page and application.
2. Employer: Employer will have control over jobs management abilities such as post jobs, delete jobs,... as well as having management over their applicants for example reject or accept and applicants.
3. Non-register User: Users who have not registered is only allowed to view available jobs and search for them but can not look into any further information unless they register.

8 Change Management

On June 11th, the customers have reviewed our project progression thoroughly and we have received request to add in a notification system for both the employer as well as the jobseeker. We have decided to update 1 more use case to our use case diagrams, do a thorough use case analysis of the use case as well as create a new sequence diagram for such use case. Aside from that we also slightly modify our mockup UI to incorporate the use case and we also updated our code to match.

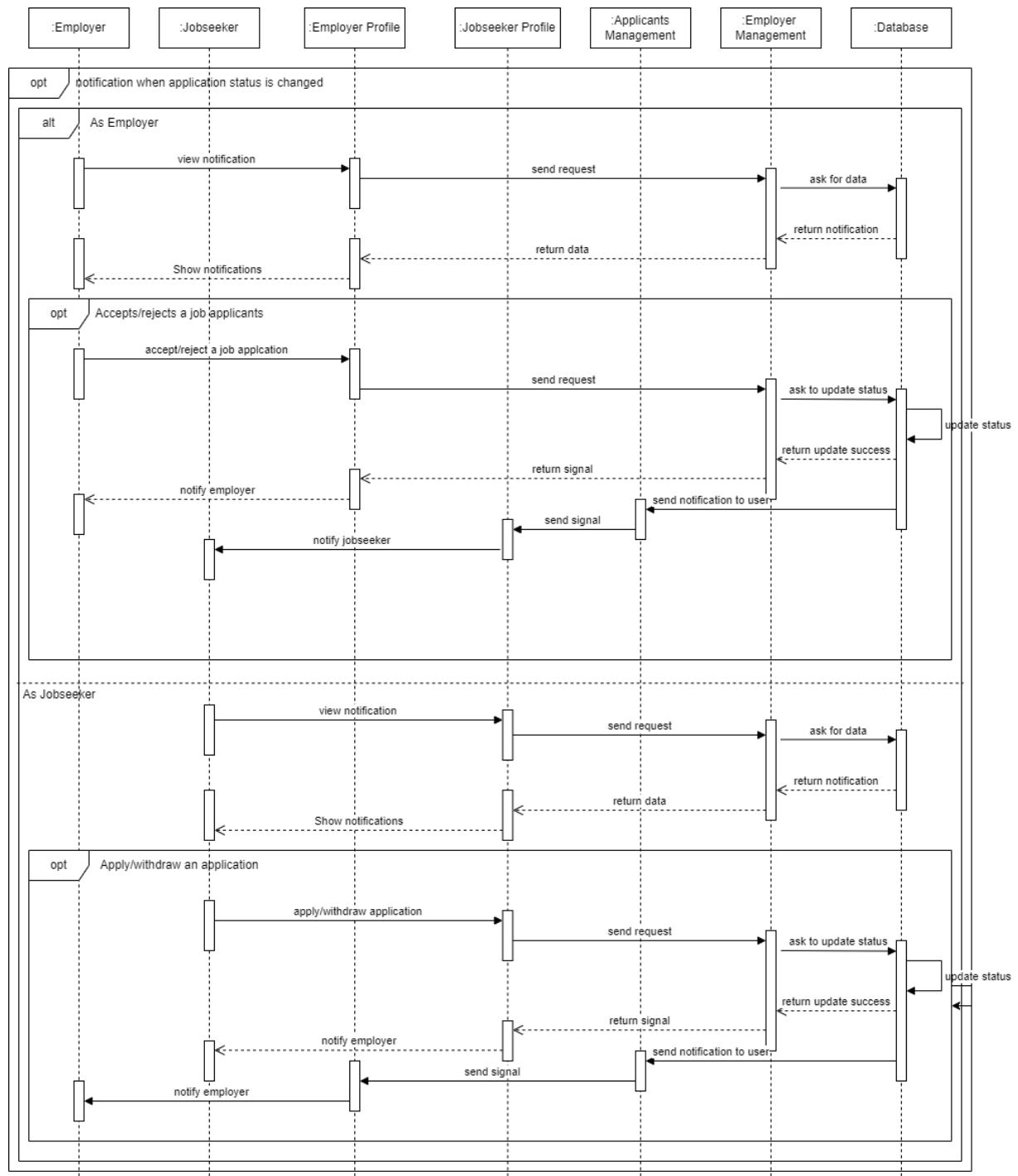
8.1 Updated Use Case Diagram



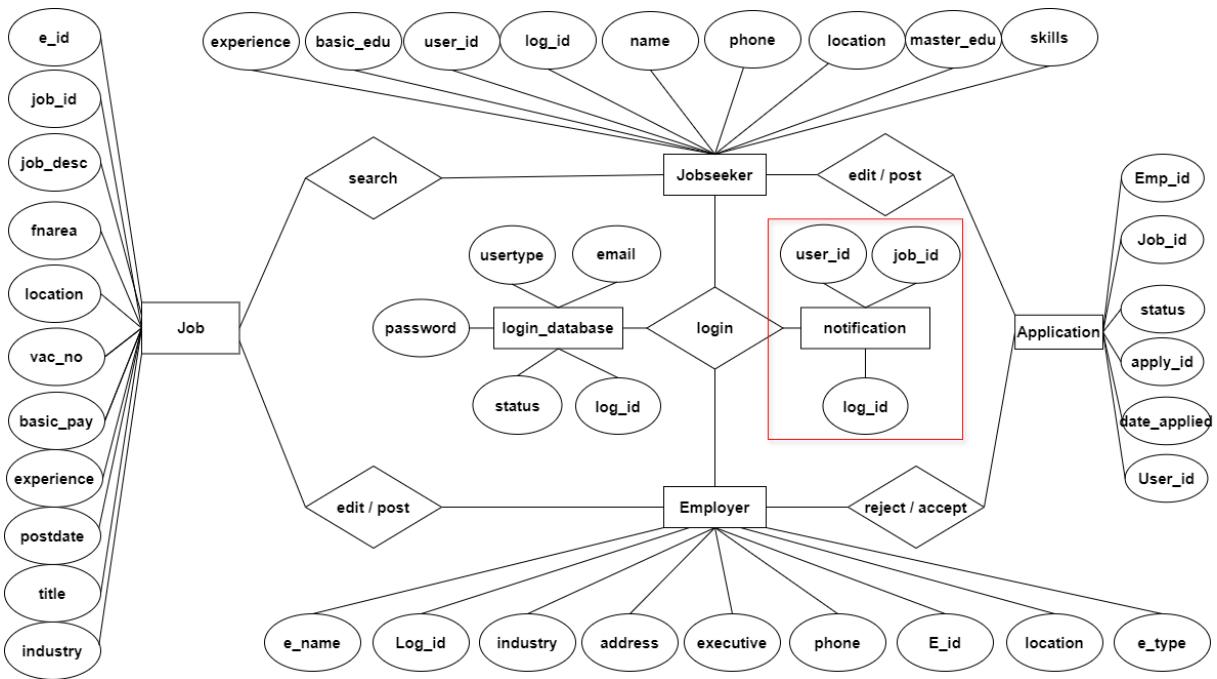
8.2 Updated Use Case Analysis

Name	Notify job application status
Description	This use case is for notifying both employers and jobseekers of the status of job applications.
Data Return	Pop up objects containing strings
Trigger	Users must click on "Notification" in the menu task bar.
Pre-conditions	Users must register as either employers or jobseekers to receive notifications related to their status
Post-conditions	Employer will receive notification when there are new applicants or when there are no vacancies available. Jobseeker will receive notification when there application status changed
Basic Flow	<ol style="list-style-type: none"> 1. The Employer or jobseeker click on the notification in menu task bar. 2. The employer or jobseeker will able to see the latest change in application status or job posted.
Alternative Flow	<ol style="list-style-type: none"> 1. When there are changes in application status or job posted the notification button on the task bar will have a red dot and the number representing the new notification 2. Employer and jobseeker can click on notification to see the latest change in application status or job posted.

8.3 Updated Sequence Diagram



8.3.1 Updated ER Diagram



8.4 Updated UI

8.4.1 Updated Employer Page

The screenshot shows the employer's profile page with the following elements:

- Header:** I.T.Job, Profile, Notifications, Menu, Account, Logout
- Welcome Message:** Welcome LMFAO
- Post Jobs Message:** You can post a new job, manage your jobs and update your profile.
- Company Logo:** MEGAH logo (yellow background with blue text).
- Change Company Logo:** Button to change the company logo.
- Company Profile:** Post jobs and find the right candidates!
- Table of Information:**

Name:	LMFAO
Type:	asdfasdf
Industry:	asdfasdf
Address:	adsasdf
Pincode:	1231
Executive Name:	MAO
Phone Number:	0978586456
Email:	12345@gmail.com
Main Location:	Vietnam

8.4.2 Updated Jobseeker Page

The screenshot shows the I.T.Job platform interface. At the top, there is a navigation bar with tabs for Profile, Notifications (which is highlighted with a red border), Options, and a search bar. A 'Logout' link is also present. Below the navigation, a welcome message 'Welcome LMAO' is displayed, along with a user profile picture of a person named 'Đà Quỳnh'. The profile section includes fields for Full Name (LMAO), Email (123@gmail.com), Phone (1425364), Location (Albania,Berat,Polican), Experience (Years) (9), Skills (Docker, Kubernetes, Google Cloud Platform (GCP), Git, ...), UG Qualification (B.Tech/B.E.), and PG Qualification (MBA/PGDM). A search bar at the top right allows users to 'Find jobs, edit your profile or update your current resume for better jobs!'.

8.5 Updated Notification Page

The screenshot shows the Notifications page. At the top, there is a 'Back' button. The main title is 'NOTIFICATIONS'. A message box displays the text 'A new job application has been submitted.' This indicates that the user has received a notification about a new job application being submitted.

8.6 Revision History

Revision History

Current
3:40:49 PM
3:38:15 PM
6/26/2024 4:33:24 PM
6/26/2024 4:29:40 PM
6/18/2024 2:37:21 PM
6/18/2024 2:19:49 PM
6/18/2024 2:02:34 PM
6/18/2024 2:00:29 PM
6/18/2024 1:47:45 PM
6/18/2024 1:45:51 PM
6/18/2024 1:44:57 PM
6/18/2024 1:40:03 PM
6/18/2024 1:35:01 PM

```

usecaseDiagram
    actor Jobsseeker
    actor User
    actor Employer
    usecase SearchJobs
    usecase RegisterJobseeker
    usecase RegisterEmployer
    usecase LoginWebsite

    Jobsseeker --> SearchJobs : "Search jobs for registered user"
    Jobsseeker --> RegisterJobseeker : "Register as jobseeker"
    Jobsseeker --> LoginWebsite : "Login to the website"
    User --> SearchJobs : "Search Jobs"
    User --> RegisterEmployer : "Register as Employer"
    User --> LoginWebsite : "Login to the website"
    Employer --> PostJobs
    Employer --> EditProfile
    Employer --> ViewApplicants
    Employer --> DeleteJobPosts
    Employer --> ViewAllJobPostInfo
    Employer --> EditProfile
    Employer --> ViewApplicants
    Employer --> DeleteJobPosts
    Employer --> ViewAllJobPostInfo
    
```

Minh Nguyen Dai 7/2/2024 3:41:01 PM ↵ 🔍 ⚖️ 🔎 Use-Case D { }

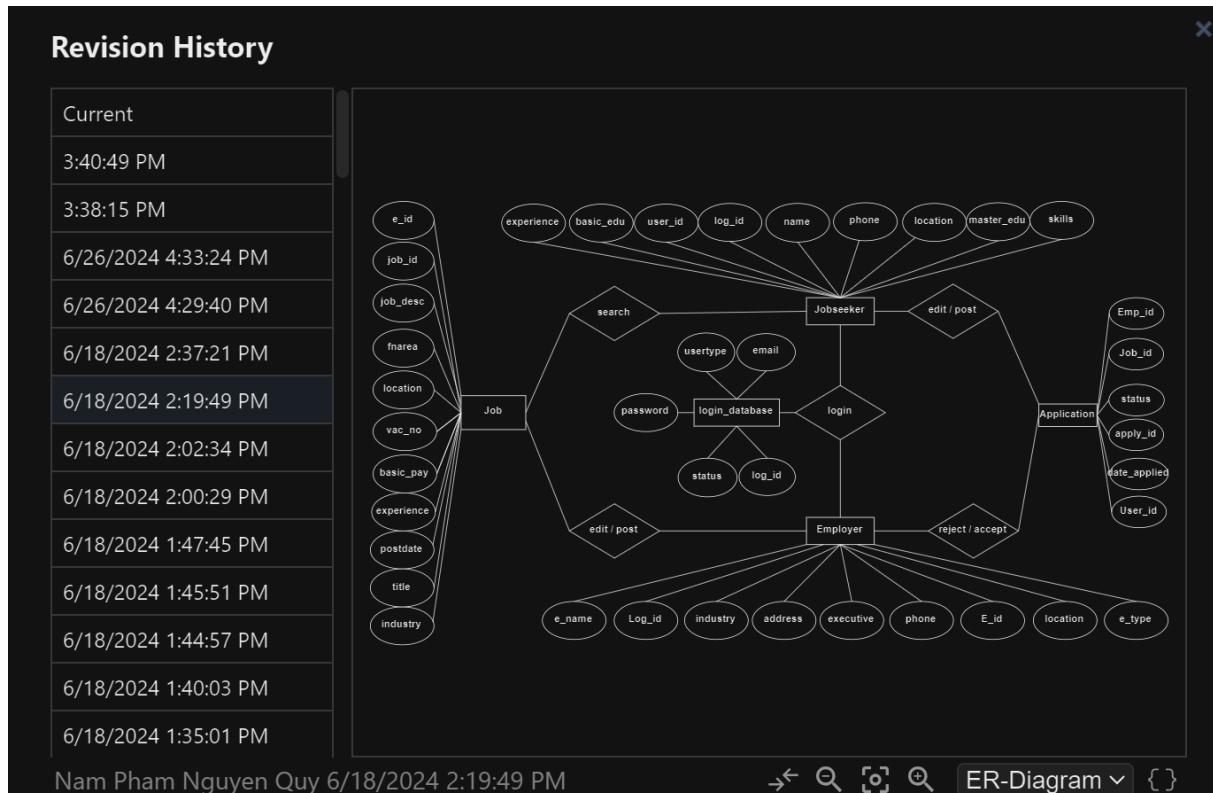
Our previous version use case diagram dont have the new use case but we added it in our current version

Revision History

Current
3:40:49 PM
3:38:15 PM
6/26/2024 4:33:24 PM
6/26/2024 4:29:40 PM
6/18/2024 2:37:21 PM
6/18/2024 2:19:49 PM
6/18/2024 2:02:34 PM
6/18/2024 2:00:29 PM
6/18/2024 1:47:45 PM
6/18/2024 1:45:51 PM
6/18/2024 1:44:57 PM
6/18/2024 1:40:03 PM
6/18/2024 1:35:01 PM

Minh Nguyen Dai 7/2/2024 3:41:01 PM ↵ 🔍 ⚖️ 🔎 Sequence D { }

Same goes for our sequence diagrams we have added 1 more diagram to our whole project



In the ER diagram you can see that our old version does not have the notification entities