COMP1682 Final Report

Project Name: Application Solve Employment Problem

Student Name: Ngo Hoang Thanh

Student ID: 001197080

Supervisor: Tran Trong Minh

Date Submit: 05/09/2022

Final Year Report

COMP1682 Final Year Project

Program Title: BSc Hons Computing

Word count: 10.000 (Minimum)

Abstract

In this project, I designed a recruiting website to provide a shared space for companies and employees to solve difficulties. Employers will post positions here to locate applicants for their companies, and workers will look for employment here. The website will give users with the majority of the essential features, such as job search, CV submission, business search... (for employees) or job posting, candidate search. (for employers). Aside from that, the website will provide users a few additional features such as chat, comment, and notification.

Acknowledgements

First and foremost, I would want to express my heartfelt gratitude to Mr. Tran Trong Minh for constantly accompanying and assisting me in completing and managing this project over the course of a year. He assisted me in improving my ideas, recognizing my own limitations, and providing extremely valuable comments and recommendations, as well as providing really kind counsel. In addition, I'd like to thank Mr. Hoang Nhu Vinh for assisting me in improving my knowledge by teaching me new knowledge and demonstrating me the hazards experienced during the project's development process to repair it in a timely manner. Finally, I'd want to express my gratitude to my family and friends for their support and assistance during this extremely stressful academic year.

Table of Contents

Table of Tables

Table of Figures

1. Introduction

1.1 Background

The issue of employment and unemployment has always received great attention from most countries in the world. Most countries strive to develop policies geared toward economic growth, stabilizing prices for services and goods, improving job supply, and reducing underemployment.

However, the COVID-19 pandemic has caused the world's economic recession, the prolonged epidemic situation has caused significant negative impacts on major economies such as the US, China, Europe, and the US. etc. The situation of the Covid-19 epidemic in Vietnam is no exception and is evolving very complicatedly. According to a July 2020 report by the Vietnamese General Statistics Office (GSO), 30.8 million individuals across the whole country were directly and seriously harmed by COVID-19. Based on the article (Huong T. T. Nguyen, 2020) can found that Vietnam's job market has been severely damaged, leading to a serious increase in unemployment rate (2.73%) and this also leads to the highest unemployment rate in 10 years.

In this context, one of the primary policy objectives is to keep unemployment low. To make it easier for people to find jobs, there are a number of websites on the online market where companies can post jobs and employees can go there to apply for jobs. Some popular job search websites in Vietnam such as: vietnamworks.com, vietsinworks.com, mywork.com.vn or Careerlink.com (Team, 2017). The above websites provide a lot of convenient functions for employees can easily find the job they want. Not only making it easy for users to find jobs, the above websites also provide users with the ability to create CVs directly and submit resumes easily. In addition, there are many other functions such as: suggesting to users about top companies and jobs, receiving notifications of new jobs, saving jobs, supporting multi-language functions, etc. However, besides the good sides of the websites on them, they also have some limitations such as: it is difficult for employees and employers to communicate with each other, employees may not trust reviews about the company or recruitment, etc.

Therefore, this project will focus to develop a website that can inherit the advantages of already existing websites in the market, while at the same time improving their shortcomings. To provide an environment for employees to find a suitable workplace for themselves, to help the company find the right piece of the puzzle for their company, and also to solve the problem of employment for Vietnam.

1.2 Aim

This project was built with the aim of developing a recruitment website to provide an environment that helps employers and employees easily communicate and recruit/find jobs. Moreover, to ensure that the project can meet the user requirements, the project will improve the shortcomings of existing recruitment websites on the market and develop some more cool features. Further, the project also aims to solve a part of the unemployment issue in Vietnam.

2. Objectives

2.1. Research

2.1.1. Research customer requirements

In this step, I will use Google Forms to collect the opinions of both the company's human resources department and job seeker to determine what their requirements are through which I will summarize list good and feasible ideas to apply to my project.

2.1.2. Research on the interface and function for the system

From the requests received from customers and I will also refer to other similar websites to conduct research, develop the interface and function of the system to this project.

2.1.3. Research about technical for the system

For the technology part, for the Front-end array I will use HTML, CSS, Bootstraps, JavaScript, jQuery, ReactJS. These are the technologies most commonly used today on the Front-end side. As for Back-end array, I choose use C# language with .NET Core Framework and database I will use SQL Server for this project.

2.2. Application Development

2.2.1. Design the interface and functionality for the application

I will check and refer to websites available online with the goal of building a beautiful, friendly interface and helping users have the best experience when using. Along with that, I will also develop the necessary functions for the application such as: login, register, search, post, submit CV and so on.

2.2.2. Design

There are different designs with different functions applied in this project which are specifically described as follows:

Use Case Diagram: I use to describe user interaction with system features.

Data Flow Diagram (DFD): It provides an overview of the system data flow. It also shows us the input and output of the data, giving us a more in-depth look at how the data is being processed.

Entity Relationship Diagram (ERD): It is a diagram showing the entities in the database and the relationship between them.

Wireframe: I use Wireframe to visually create a blueprint from my project ideas for easy development and modification.

2.2.3. Implementation

Once I have completed all the design steps and selected a development technology, I will start developing my application. I will try to develop the application in accordance with the above designs to create the most complete website.

2.3. Testing

After completing the project, I will proceed to check that the system functions are working according to the original requirements. After that, if there is an error or something is not right, I will repair it to improve the product better.

2.4. Analyze

In this section, I will summarize what I have done and seriously evaluate whether my product has met the initial targets set out or not. If there's something unfinished, I'll fix it right away. After completing the initial targets, the system is basically considered complete but I will also learn and receive contributions from the community so that the product can be more and more perfect.

For each software project, it is necessary to analyse and choose the suitable SDLC (Software Development Life Cycle) model to easily plan development, maintenance, change or upgrade during project development. In this project also needs a suitable SDLC model, so after I researched and analysed some popular software development methodology such as: Agile, DevOps, Rapid Application Development, Waterfall Model. I found this project suitable for applying the Waterfall model to the project development process.

3. Approach

3.1. Justification of the suitability of a Methodology or a Framework followed.

After researching and analysing some popular methodologies as well as analysing and evaluating the high-level functional requirements of the project. I have decided to include waterfall model in this project. And the reason I choose it is:

- First, the phases of the model are clear:
    - Requirements
    - System Design
    - Implementation
    - Integration and Testing
    - Deployment of System
    - Maintenance

  With clearly divided phases makes it easy for me to work and manage the project progress.

- Second, because the waterfall model is afraid to face changing requirements or project scope, but this project does not include the customer - who often causes changes to the requirements and project scope. Therefore, the weakness of the requirement change, and the project scope at later stages of the project is non-existent, which makes it easy to use this methodology in the project.
- Third, because the stages don't overlap, I can easily develop without fear of conflict.

- Finally, the tasks in the previous stage must be completely completed before moving on to the next stage, so if there are any barriers, they will come to light immediately to be dealt with.

4. Literature Review

4.1. Approach to literature searching

Research can be time-consuming, but it will provide the insight and information needed to be able to assess the needs of the market to consider the idea of developing a product that can meet the needs of the market or not. For this study to be successful, it is of utmost importance to ensure that all information gathered is from reputable and authoritative authors. Because this project mainly focuses on the job market in Vietnam, I will only focus on reports, studies, and official websites to avoid inappropriate data leading to incorrect requirements and project scope. All information collected for the research of this project will be collected by me in the form of Journal Article, Report, Book, Academic report, Document from official websites.

4.2. Market analysis

In 2020, net foreign direct investment inflows into Vietnam will reach about US$15.8 billion (Ngoc, June 2021) that is a potential figure for the market in Vietnam. Based on (Das, n.d.), it has been reported that around 40 percent of FDI firms in Vietnam find it difficult to recruit skilled employees. This shows that with a large amount of investment in the Vietnamese market, and if employees can meet the requirements of foreign-invested companies in Vietnam, this will be a very good opportunity for companies to invest in Vietnam. Employees can find suitable jobs for themselves with high salaries.

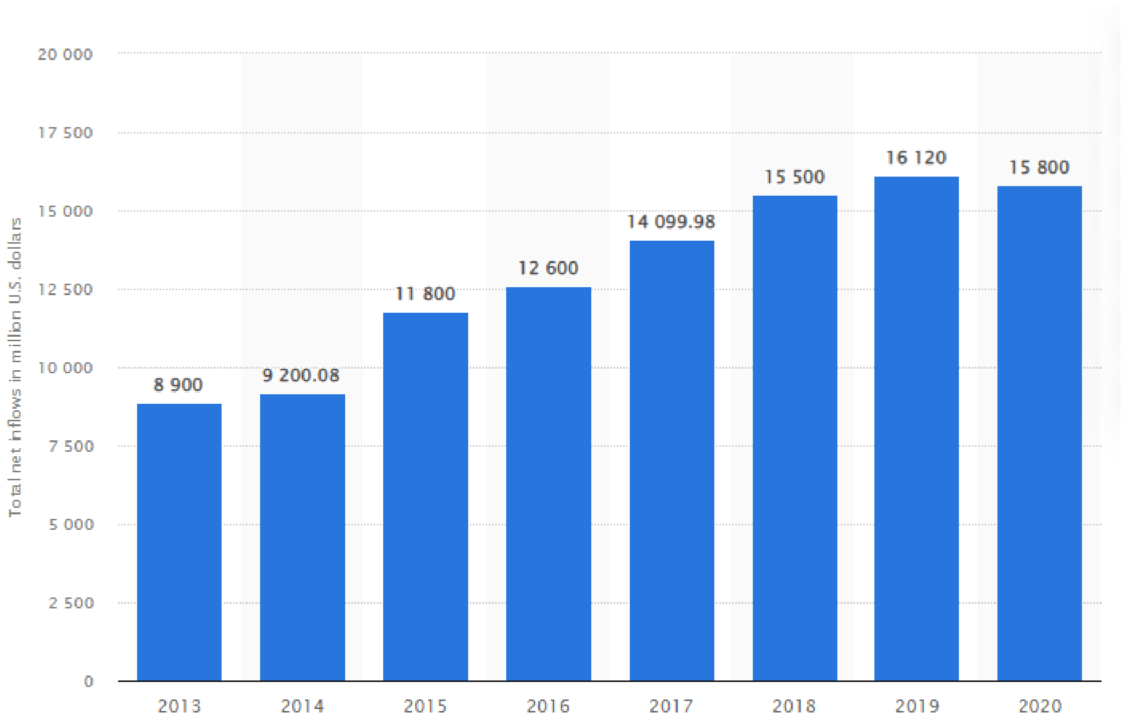


FIGURE 1: FOREIGN DIRECT INVESTMENT NET INFLOWS IN VIETNAM FROM 2013 TO 2020 (NGOC, JUNE 2021)

The above section has presented the potential of international investment capital for Vietnam's labor market. And next will research for the government as well as companies with capital in Vietnam.

According to (Cekindo Vietnam, 2021), the Vietnamese government took moves in March 2018 to enhance technical and vocational training in order to meet labor market demands. Young adults in Vietnam now have the ability to obtain vocational education accreditation based on Decree No. 49/2018/ND-CP.

The government hopes to reach its goal of providing vocational training to 2.2 million Vietnamese individuals in 2018. The outcome was encouraging: as of the first quarter of 2018, Vietnam had almost 1,900 vocational training centers, including 545 vocational schools and 395 colleges.

Thus, it can be seen that now as well as in the near future, the demand for job search will be increased.

Along with the growth of the digital industry, many people will look to job brokerage sites like LinkedIn to find jobs or recruit employees. Adrien Bizouard, country manager of Robert Walters Vietnam said that "We see opportunities for offline people to do more online projects allowing them to grow their skills. Vietnam remains transitional – people are still making purchases offline even as they move online". It can understand that online recruitment cannot replace offline but it can work at the same time to increase productivity. Through this study, it can be seen that the development of a recruitment website is highly feasible.

4.3. Technology used for the project

The project is developed based on the ideas of some existing recruitment websites on the market and aims to increase the user experience. In this section I will list the technologies selected for the project.

4.3.1. Web app

Web applications are programs that allow organizations and their consumers to communicate more effectively. Companies are changing their working practices and adopting more web applications as the Internet becomes more widely used.

According to (Team, 2021), Web apps can be developed for many different reasons and used by companies or individuals. Individuals need it to facilitate their communication or purchase things online. Also, employees can collaborate on projects and work on shared documents with web applications. They can create reports, files, and share information from anywhere and with any device.

And for this project, the web app will provide an environment for companies and individuals to communicate to exchange information related to recruitment. In addition, the web app also has benefits such as:

- User don't need to install it on the hard drive, so it doesn't cause space limitations.
- A web application reduces costs for both the end-user and the business.
- All users can access the same version so it eliminates any compatibility issues.
- Web apps can operate on numerous platforms, independent of the operating system or device, as long as the browser is suitable.
- Easy UI design with more features than mobile apps.

4.3.2. Front-end (ReactJS)

ReactJS is a component-based library that may be used to create interactive user interfaces. It is currently the most widely used front-end JS library. The view (V) layer is included in the M-V-C (Model View Controller) pattern. Facebook, Instagram, and a network of individual developers and organizations all support it. React enables the creation of huge and complicated online applications that can update their contents without requiring further page refreshes. It aims to improve user experiences by developing web apps that are lightning fast and stable. Other JavaScript libraries or frameworks, such as AngularJS, can be used with ReactJS in MVC.

This article (Aggarwal, 2018) analysed about the performance of ReactJS as follows:

> ReactJS is a JavaScript library for creating reusable user interface (UI) components. The following is the definition from the official React documentation: React is a library that allows you to create modular user interfaces. React enables the creation of huge and complicated online applications that can update their contents without requiring further page refreshes. It's the View (V) in the Model-View-Controller system (MVC). React encapsulates the Document Object Model (DOM), making application development easier, faster, and more reliable. React mostly renders on the server using NodeJS, with

React Native providing support for native mobile apps. React uses unidirectional data flow, which reduces boilerplate and makes it easier to use than traditional data binding.

ReactJS' virtual DOM (Virtual document object model) is another important feature. It's comparable to the document object model generated by the browser, except it's stored in memory instead. The operation of virtual DOM is pretty straightforward. When a request to alter the page content is made, the changes are initially reflected in the virtual DOM that is stored in memory. After that, instead of rendering the complete DOM, a diff() algorithm compares the two, i.e. the virtual DOM and the browser DOM, and only the appropriate modifications are reflected to the browser DOM. This significantly improves the application's performance, especially when thousands of data changes are required.

4.3.3. Back-end (.NET Core)

There are several languages and frameworks accessible in the field of backend development, but .NET Core, Node.JS, Spring, and Symphony emerge as the frontrunners. In this project, though, I chose.NET Core for the back-end. Because it provides a very dependable environment with excellent performance, enabling for application growth, testing, and hosting or cloud migration. It can be used in even the most secure applications due to its high level of security.

In addition, Lukasz and Pawel 2 senior .NET Core developers also shared about the advantages of .NET Core as follows (Lukasz, 2020):

- Free IDE: VS Code, VS Community, or virtually any text editor used with CLI.
- Community: Centered around Stack Overflow (over 10k answered topics according to 2020 data)
- Multiple supported OSes: macOS, Linux, and Windows, also with Docker support!
- Lightweight: .NET Core apps are modular, meaning they can be built only with needed components.

4.3.4. Database (SQL Server)

SQL Server is a relational database management system (RDBMS) that was created and released by Microsoft. It includes its SQL language as well as Transact-SQL (T-SQL), Microsoft's proprietary language with exception handling, variable declaration, and stored procedures. SQL Server Database Engine is the core component of SQL Server that controls, processes, and secures data storage. The database engine is divided into two sections: the relational engine, which processes commands and queries, and the storage engine. The second component is the storage engine, which is responsible for managing database features such as tables, pages, files, indexes, and transactions.

Author Nabil DIDAOUI (DIDAOUI, 2018) listed benefits when using SQL Server:

- Eliminate the need for data movement
- Run real-time analytics on operational data
- Reduce database maintenance and increase business uptime
- Boost security and protect data in use

- Optimize with choice and flexibility

5. Legal, Social, Ethical and Professional Issues and Considerations

5.1. Legal

According to (Nishith Desai Associates, 2005), First, this project will ensure compliance with the core legal issues of an e-commerce website: contracts, security, authentication, privacy and data protection, intellectual property rights, domain names, jurisdiction and taxation. In addition to the above problems, I will also make sure that there are no other problems arising in this project.

5.2. Social

Currently, the issue of employment in Vietnam is one of the issues that need top attention. According to (Chong, et al., 2021), Vietnam's unemployment rate was 2.22% in the first quarter of 2020 but rose to 2.73% in the second quarter (Terence Tai Leung Chong, 2020) and now that percentage is even higher. Therefore, finding jobs for people or human resources for companies is also a rather urgent issue. I think this project will be the best way to solve the above problem. In today's internet age, by creating a job website, it will help companies easily find human resources and people also easily find their desired job easily. Especially during the current period of limited travel.

5.3. Ethical and Professional

In order to create a professional working environment for everyone, ethics is also a top priority in this project. I will ensure that my project will cover basic ethical issues such as: trust, security and privacy and loyalty according to (Lijuan, 2013). Besides, it also does not violate the prohibited acts for the website: perform fraudulent acts, violate intellectual property rights, use false information to deceive others or steal and save money. Disclosing confidential information of individuals or organizations, etc (Tri minh law corporation, 2021).

5.4. Considerations

Data storage requirements may rise, requiring innovative ways to data management. To ensure that users' data is safe, data storage systems must keep up with the rate of development. Furthermore, backups must be kept isolated from the primary server to avoid being harmed in the event of a data breach affecting users. To make evaluating the technology easier, I needed to appropriately define and categorize the application's content. Because data theft can do substantial harm to a corporation, the project requires risk assessment and security. To improve security, the system requires user access control. If the system does not have needless access privileges, a breach will not cause significant damage.

6. Requirements

Requirements help the developer understand what is required to complete the project and make sure that everyone working on the project understands what must be done to meet the needs of the project. It is important to start a project with a requirements analysis because it help stakeholders to explain their difficulties with the current system as well as what they want the system to do. Understanding the customer's problem is essential to determining what the new system will require. However, there are several criteria that support different aspects of the project (examples: functional requirements, non-functional requirements, business requirements). Therefore, it is important to identify all the high-level requirements for this project.

6.1. Analysis of requirements

Before developing a project, the most important thing is to think about the system requirements. A high-level requirements list will be analyzed before the implementation phase begins and it should contain all the required system requirements.

ID	High Level Requirements	Description
1	Login	The user must be logged in to the system in order to utilize the application. This is to assist the system in determining who is logged in and what role to log in to the system in order to allow that individual to access the functions associated with his or her role in the system.
2	Register	To handle user information, the system must provide a registration function that collects and stores user information. Furthermore, in order to use the application, users must create an account, so the registration feature is required.
3	Logout	The logout function will delete all user data from the browser to protect the account or log in to the system with another account.
4	Forgot Password	In some cases, users may forget their password so the forgot password function helps them to recover their password.
5	Change password	This function helps users to change their password to protect their account when there is a problem.
6	Manager user profile	Users can access manage their profile such as name, phone number or email,...
7	Create recruitment postings	Companies can create recruitment postings to find employees for their company
8	View recruitment posting	Users can view recruitment posting and edit or delete if they want
9	Employee submit CV to the recruitment	Employees can submit their CV files to the job posting so that employers can review their CV

	posting	
10	Comment on the recruitment posting	Company and employee can comment to discuss, ask some questions or answer questions together.
11	Chat between company and employee	The company and the employee can chat to discuss about the problems related to the recruitment. This is a function as a bridge to communicate between employee and company.
12	Search recruitment information with options	Users can search for information to support their job application with many options such as list of companies are hiring, list of jobs that are recruiting,...
13	Company view and downloads employee's CV file pdf	The company can view or download the CV file pdf of the employee for easy storage, classification and evaluation.
14	Create a new career	The administrator can create a new career.
15	Manage career	The administrator can manage career such as: view list, edit or delete.
16	Create a new branch	The administrator can create a new branch.
17	Manage branch	The administrator can manage branch such as: view list, edit or delete.
18	Search by branch or career	The users can search to view detail of the branch or career they want.

TABLE 1: ANALYSIS OF REQUIREMENTS

6.2. Existing Solutions

6.2.1 CareerLink.vn

CareerLink.vn, which was founded in 2006, has progressively grown into a reliable link between employees and companies. It specialize in senior management people searches as well as internet recruiting. As a result, businesses will be happy with the fact that they were able to discover qualified people for their needs, and job seekers will be fulfilled with their desire to advance in their professions.

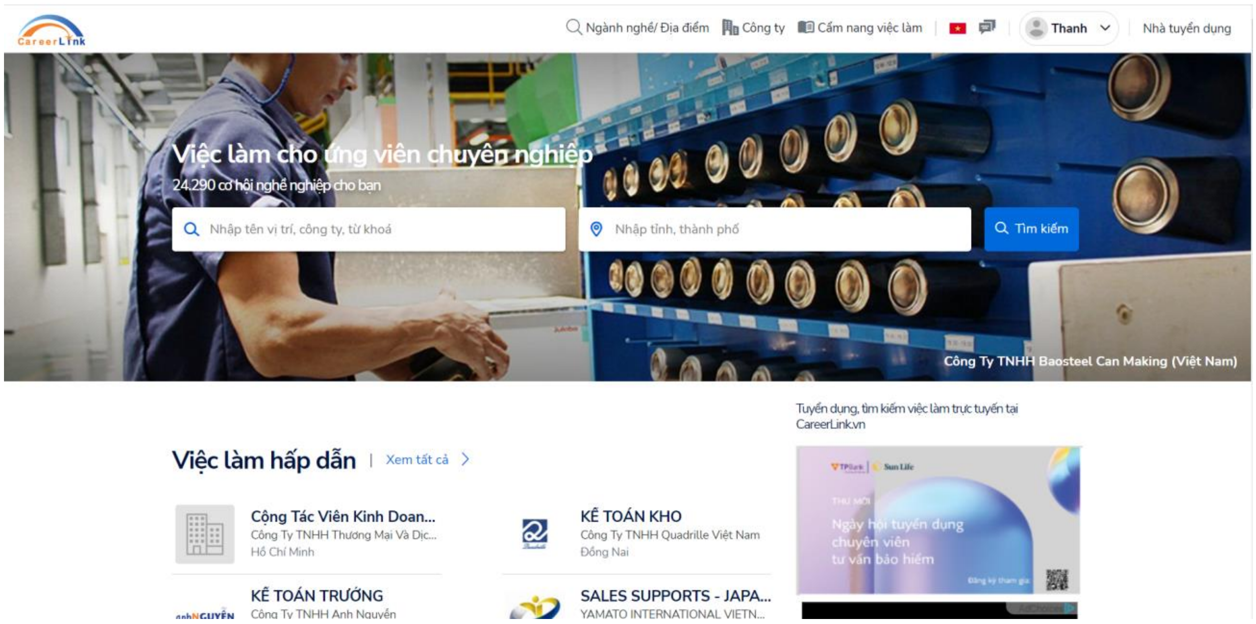
Pros

- It job segmentation by geographical location and field makes it easier for candidates and employers to find the correct job.
- Make it possible to create CVs for employee immediately on the website.

- Provide a large source of candidates, diverse in fields and qualifications.
- The candidate's CV and profile were reviewed, restricting the status of virtual CVs and rubbish CVs while providing maximum help to businesses.

Cons

- The majority of them are for skilled and experienced candidates. Small and medium-sized businesses frequently struggle to locate the suitable applicant.
- Companies must pay a fee to publish employment advertisements.
- Because there are a large number of candidates' virtual CVs on the site, employers must check them before getting resumes from this site.



Việc làm cho ứng viên chuyên nghiệp

24.290 cơ hội nghề nghiệp cho bạn

Tuyển dụng, tìm kiếm việc làm trực tuyến tại CareerLink.vn

Việc làm hấp dẫn

Cộng Tác Viên Kinh Doan...
Công Ty TNHH Thương Mại Và Dịc...
Hồ Chí Minh

KẾ TOÁN KHO
Công Ty TNHH Quadrille Việt Nam
Đồng Nai

KẾ TOÁN TRƯỞNG
Công Ty TNHH Anh Nguyên

SALES SUPPORTS - JAPA...
YAMATO INTERNATIONAL VIETN...

Ngày hội tuyển dụng chuyên viên tư vấn bảo hiểm

FIGURE 2: CAREERLINK.VN INTERFACE

6.2.2 Vietnamworks.com

VietnamWorks is the firm Navigos's oldest recruiting website in Vietnam. Understanding the demands of job seekers, VietnamWorks not only provides people with the chance to find work, but also provides prospects for advancement and growth at work. With a large number of job listings each year, it may be called the leading website in Vietnam for assisting job searchers and businesses.

Pros

- Vietnamworks has a long history of job posting and job search. The site boasts a large number of visitors, a diversified and high-quality staff, and is suitable for both local and international enterprises.
- Candidates with extensive coverage, who are highly qualified, educated, and experienced

Cons

- The site is primarily intended for recruiters with advanced professional qualifications and expertise. Suitable for firms searching for officers or employees with high wages.
- The page collects fees for some services such as file filtering packages, file clipping at locations, etc.

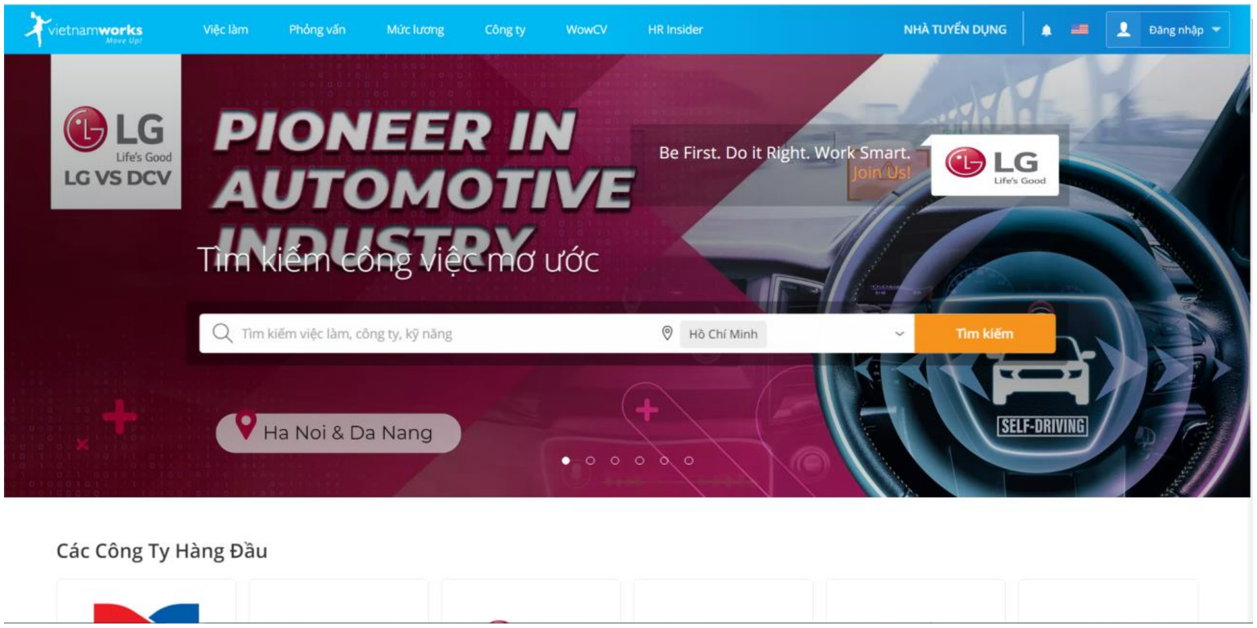


FIGURE 3: VIETNAMWORKS.COM INTERFACE

6.2.3 Conclusion

After analyzing the benefits and drawbacks of the apps on the market, I chose and built the features that are truly important for this project, while minimizing the drawbacks. The program will solve the market's remaining obstacles while optimizing basic functionality to improve user experience. In addition, in order to attract more users, I create new features such as chat, comments between employers and workers, and a notification mechanism.

7. Business Requirements

7.1. Overall Picture

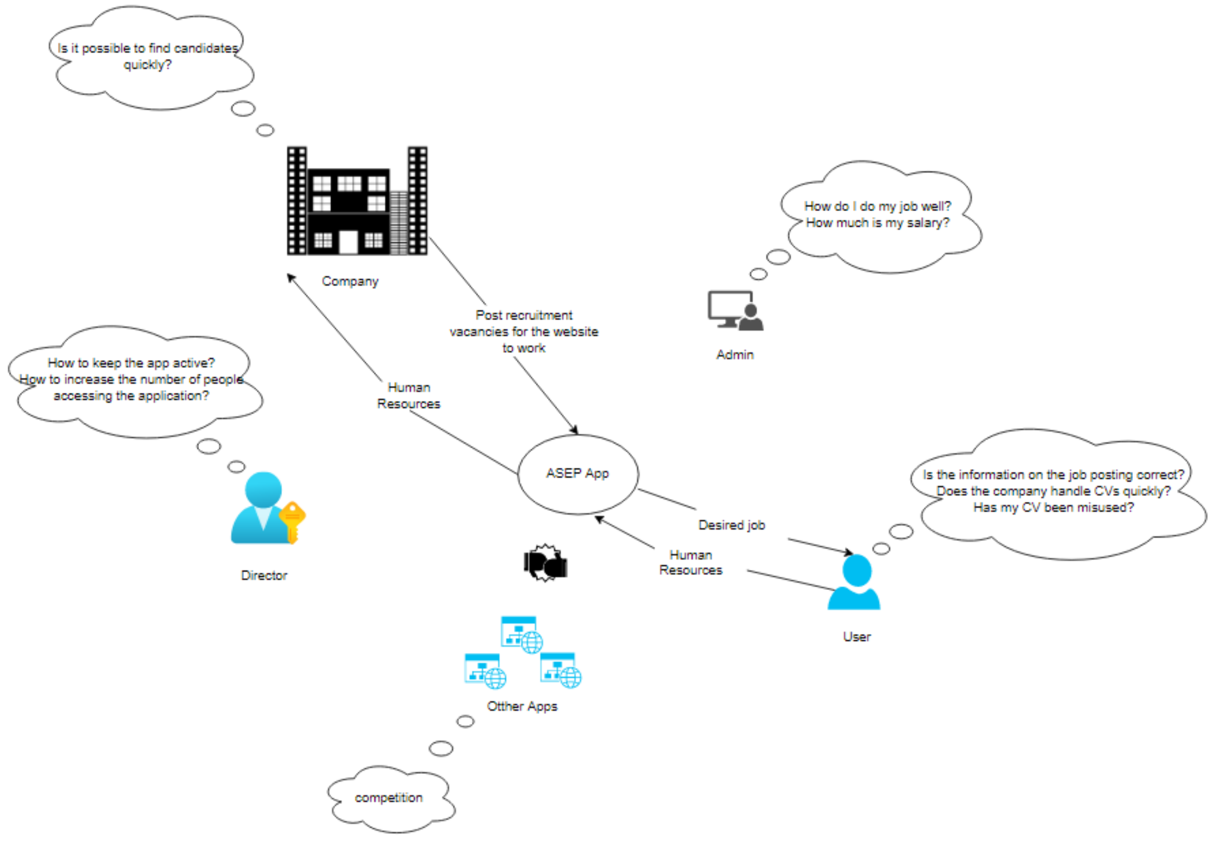

Is it possible to find candidates quickly?
Company
Post recruitment vacancies for the website to work
How do I do my job well? How much is my salary?
Admin
How to keep the app active? How to increase the number of people accessing the application?
Human Resources
ASEP App
Is the information on the job posting correct? Does the company handle CVs quickly? Has my CV been misused?
Desired job
Human Resources
Director
User
Otther Apps
competition

FIGURE 4: OVERALL PICTURE

7.2. Functional Requirements with MoSCoW prioritisation

The following is a list of all the functional requirements for the project at the same time also evaluate the priority and estimate time for each requirement. In there:

Must have: This first category comprises all of the prerequisites required for the project to be completed successfully. These should be non-negotiable aspects that satisfy the bare minimum of needs (MUST).

Should have: This second type of needs is one step below must have; it may be used to prepare criteria for future releases while not interfering with the present project. Should-have aspects are useful for project completion, but they are not required. In other words, even if should-have requirements are not present in the final product, it will still operate. Should-have aspects, on the other hand, significantly boost the value of the product.

Could have: This category includes requirements that have a much smaller impact when left out of the project.

ID	Functional Requirements	Description	Priority	Estimate time (Day)

1	Login	The user must be logged in to the system in order to utilize the application. This is to assist the system in determining who is logged in and what role to log in to the system in order to allow that individual to access the functions associated with his or her role in the system.	Must have	4
2	Register	To handle user information, the system must provide a registration function that collects and stores user information. Furthermore, in order to use the application, users must create an account, so the registration feature is required.	Must have	3
3	Logout	The logout function will delete all user data from the browser to protect the account or log in to the system with another account.	Must have	2
4	Forgot Password	In some cases, users may forget their password so the forgot password function helps them to recover their password.	Must have	3
5	View user profile	Users can access to view their personal information.	Must have	4
6	Edit user profile	User can edit their information.	Must have	3
7	Change password	This function helps users to change their password to protect their account when there is a problem.	Must have	2
8	Create recruitment	Companies can create recruitment postings to find	Must have	4

	postings	employees for their company		
9	View recruitment posting	Users can view recruitment posting	Must have	4
10	Edit recruitment posting	Companies can edit their recruitment posting	Must have	3
11	Delete recruitment posting	Companies can delete their recruitment posting	Must have	2
12	Employee submit CV to the recruitment posting	Employees can submit their CV files to the job posting so that employers can review their CV	Must have	4
13	Comment on the recruitment posting	Company and employee can comment to discuss, ask some questions or answer questions together.	Should have	12
14	Chat between company and employee	The company and the employee can chat to discuss about the problems related to the recruitment. This is a function as a bridge to communicate between employee and company.	Must have	8
15	Follow company	Employee can follow the company they need to easily track information about recruitment of that company.	Should have	4
16	Get notifications	Usually, companies will not regularly check recruitment website, so sometimes they will miss employee's job applications. Therefore, it is necessary to have a notification function for send notifications when employee submits a CV to the recruitment posting,	Should have	8

		comment or chatting,...		
17	Search recruitment information with options	Users can search for information to support their job application with many options such as list of companies are hiring, list of jobs that are recruiting,...	Must have	8
18	Company downloads employee's CV file pdf	The company can download the CV file pdf of the employee for easy storage, classification and evaluation.	Must have	4
19	Company views detail employee's CV file	The company can view CV file of the employee in the browser.	Could have	4
20	Support multi-languages	Supports multiple languages will help expand the market and attract more potential users, increase competition and easily for marketing.	Could have	12
21	Generate template CV	Some users are inexperienced in creating a standard CV and they need assistance such as showing example CVs or templates so they can edit suitable with themselves.	Could have	8
22	Create a new career	The administrator can create the career.	Must have	3
23	View list career	The users can view the career.	Must have	4
24	Search career	The users can search the career.	Should have	3
25	Edit career	The administrator can edit the career if they want.	Must have	3
26	Delete career	The administrator can delete the career if they feel it unnecessary.	Must have	2
27	Create a new branch	The administrator can create a new branch.	Must have	3

28	View list branch	The users can view list branch.	Must have	4
29	Search branch	The users can search the branch.	Should have	3
30	Edit branch	The administrator can edit the branch if they want.	Must have	3
31	Delete branch	The administrator can delete the branch if he/ she feels it unnecessary.	Must have	2
32	Create forum	Create a forum for people to discuss and share experiences in find job, hire employee or interview.	Could have	8
Total Estimate time				144 days

TABLE 2: FUNCTIONAL REQUIREMENTS WITH MOSCOW PRIORITISATION

7.3. Non-functional Requirements

Factorial	Requirement
Security	The user's password must be hashed before store into the database.
	The system must protect personal information for users.
Performance	System response time should be fast.
	Do not allow storing large files to avoid overload.
Usability	Easy to use
	Design descriptions, icons to help users understand functions without having to try.
Maintainability	The system must be maintained regularly so that errors can be detected and patched in time
Scalability	The system must be able to expand into the future, ensuring that future features do not overload the hardware.
Mobile Responsive	Have to develop responsive UI for mobile devices
3rd Party Integrations	Ability to link with 3rd parties such as Facebook, Google.

TABLE 3: NON-FUNCTIONAL REQUIREMENTS

8. Analysis and Design

8.1. Architecture

This project uses Client-server architecture for application development. After researched about architecture through this report (Roomi, 2020), I decided that should choose this architecture because it provides me with the following advantages:

- It provides the ability to save all the important information in one location. This facilitates the management and updating of resources or data.
- In client-server architecture, data is centralized in a single location, it making client data security easier and safer. Furthermore, if data is lost, it is quite simple to retrieve data from a single backup table.
- Using the client-server paradigm allows me to easily upgrade the application in the future.
- Without needing terminal mode or a processor, any client may effortlessly access the system.

8.2. High Level Design

8.2.1. Assumptions

- Resources - I assume I can do all the roles on the project my health always good during the project and my computer is always working fine. However, I can be overloaded with completing tasks leading to reduced health, slow project progress, or my computer crashes during project implementation.
- Delivery - On time delivery, high level requirements are fulfilled and some additional functionality is also developed. However, due to reasons such as lack of knowledge or procrastination, the project is delivered late or the committed functions are not fully delivered.
- Budget - The project will use all free resources so budget should be no problem. However, there may be costs such as damaged computers that need to be purchased new or software and services that require licensing, leading to out of budget.
- Scope - the scope of the project will remain unchanged throughout the implementation. However, because the analysis of the project's requirements is incorrect, the project scope can be change.
- Schedule: all tasks and deadlines are delivered on time. The order of each scheduled task does not change throughout development. However, due to non-procedural operation, the schedule can be disturbed.
- Methodology - Apply the waterfall model well to all phases of the project. However, due to the lack of experience in organization, management and operation, it is no longer true to fulfil the commitments at each stage.
- Technology - All software works well, there are no problems such as: version error, expired license... The environment on the computer is compatible with the installed

software. However, because installing the wrong version leads to an environment error, a software error that causes a serious system failure.

8.2.2. Endpoints

An endpoint is a vulnerable point that can be used by cybercriminals to infiltrate the system. Therefore, clearly defining the endpoints is essential for the project. Project will have 4 connection endpoints that are laptops, desktop computers, smartphones, and tablets.

8.2.3. 3rd party Services

In this project, it has a function is sends a Gmail notification to the company or employee when there are activities related to their account such as: an employee sends a CV to the company, or when the If the company posts a job posting, users who follow the construction will receive a notification...

8.2.4. Overall Picture

The technologies used in the ASEP project are summarized below. First, I utilize React Js to develop the front-end to receive user requests. Then, for the system's backend, I use.NET Core Framework to process the requests provided from the front end. Finally, SQL Server is used to store information in a database.

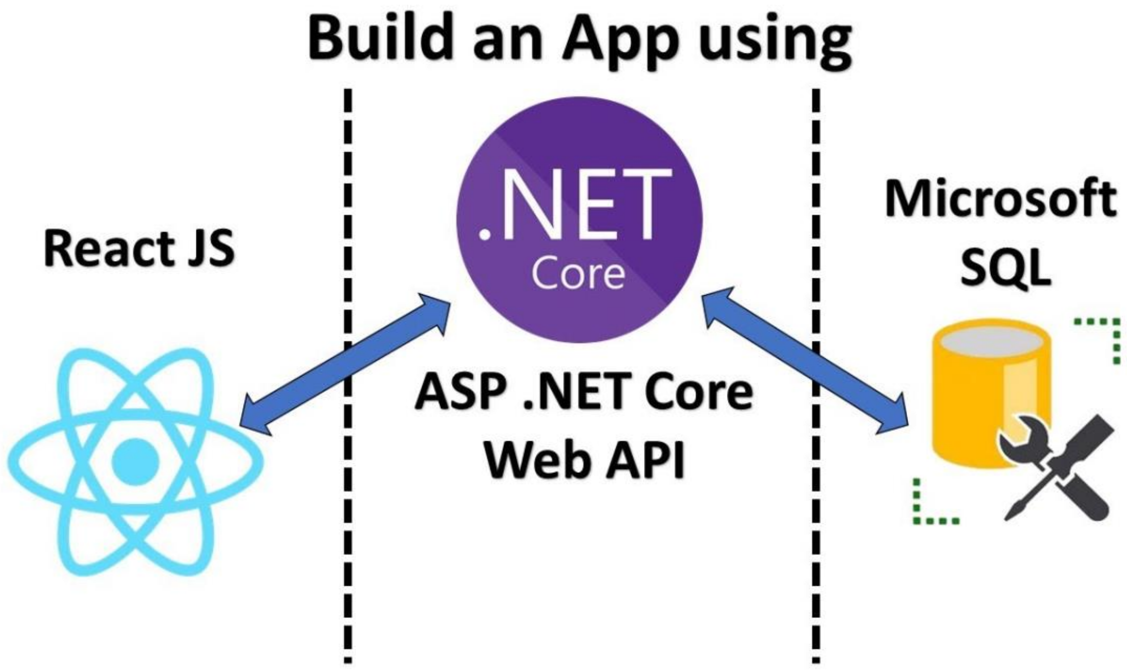


FIGURE 5: OVERALL PICTURE

8.3. Technology choices

8.3.1. Front End

After investigating front-end development technologies, I selected to utilize ReactJS in my project for the following reasons:

- The main reason I selected ReactJS for my project is because it is much easier to learn and use than popular frontend frameworks like AngularJS and VueJS. Therefore, it makes simple for me to construct and develop my project.
- It also includes a number of libraries that assist users in easy and simply creating interfaces such as Bootstrap, Material UI, Chakra, Tailwind, and others.
- React also provides users with a virtual DOM functionality, which is a virtual version of the DOM that allows me to easily analyze the risks whenever I modify the UI.
- At the moment, ReactJS is the most popular framework, and it is liked and used by many people. As a result, its support network is rather large, and I can readily obtain assistance anytime I encounter an issue.
- Using react allows me to construct my project using so-called reusable elements. This allows me to easily change and reuse my code.

8.3.2. Back End

I use .NET core framework for my project for the following reasons:

- It is an open source framework with a large community that can support and help me in developing this project swiftly and smoothly.
- .NET allows for cross-platform deployment across Windows, macOS, and Linux. Furthermore, the Visual Studio IDE provides a plethora of capabilities to assist me in designing and developing applications quickly.
- .NET Core gives me with exceptional performance as well as flexibility and extensibility, allowing me to build maintainable, reusable, and efficient code by using optimizations tools and support libraries

8.3.2.1. Database

After considerable consideration, I opted to utilize Microsoft SQL Server database for my project because:

- Microsoft SQL is simple to use and install, requiring only a few mouse clicks. It is also incredibly simple to use due to its user-friendly UI.
- It has several editions to satisfy a wide range of demands including: Enterprise, Standard, Express, and Developer. In this project, I picked the developer version since it best suited my project's requirements and scope.
- SQL Server databases are very secure and employ advanced encryption methods that make it nearly hard to breach security levels. Therefore, the user's data will be completely secure.
- SQL Server has a lost data recovery capability that allows me to restore my data if something goes wrong.
- Connecting and retrieving data between .NET core and SQL Server is very easy using Enity Framework Core

8.3.3. Web App

Currently, the most popular ways to reach users through the internet are web applications and mobile applications. After researching and analyzing the feasibility, I realized that the project should be implement as a web app. The reason is as follows:

- Users do not need to be downloaded or installed to use —web apps function in-browser.
- Easy to maintain—they have a common codebase regardless of mobile platform.
- Web app will update themselves.
- Faster and easier to build than mobile app.
- Do not need app store approval to publish, so easy to launch.

8.3.4. Operation System

After doing some study on operating systems, I opted to utilize Windows for my project because:

- Windows is the operating system that best supports web application development using .NET Core framework and SQL Server.
- Installing visual studio, visual studio code and SSMS on windows is also pretty simple and quick.
- Windows offers good anti-virus protection that is incorporated into the operating system to tackle online threats. This protects my code safe during project development.

8.4. Use case Diagram

8.4.1. Use case diagram, primary and secondary use-case scenario for admin

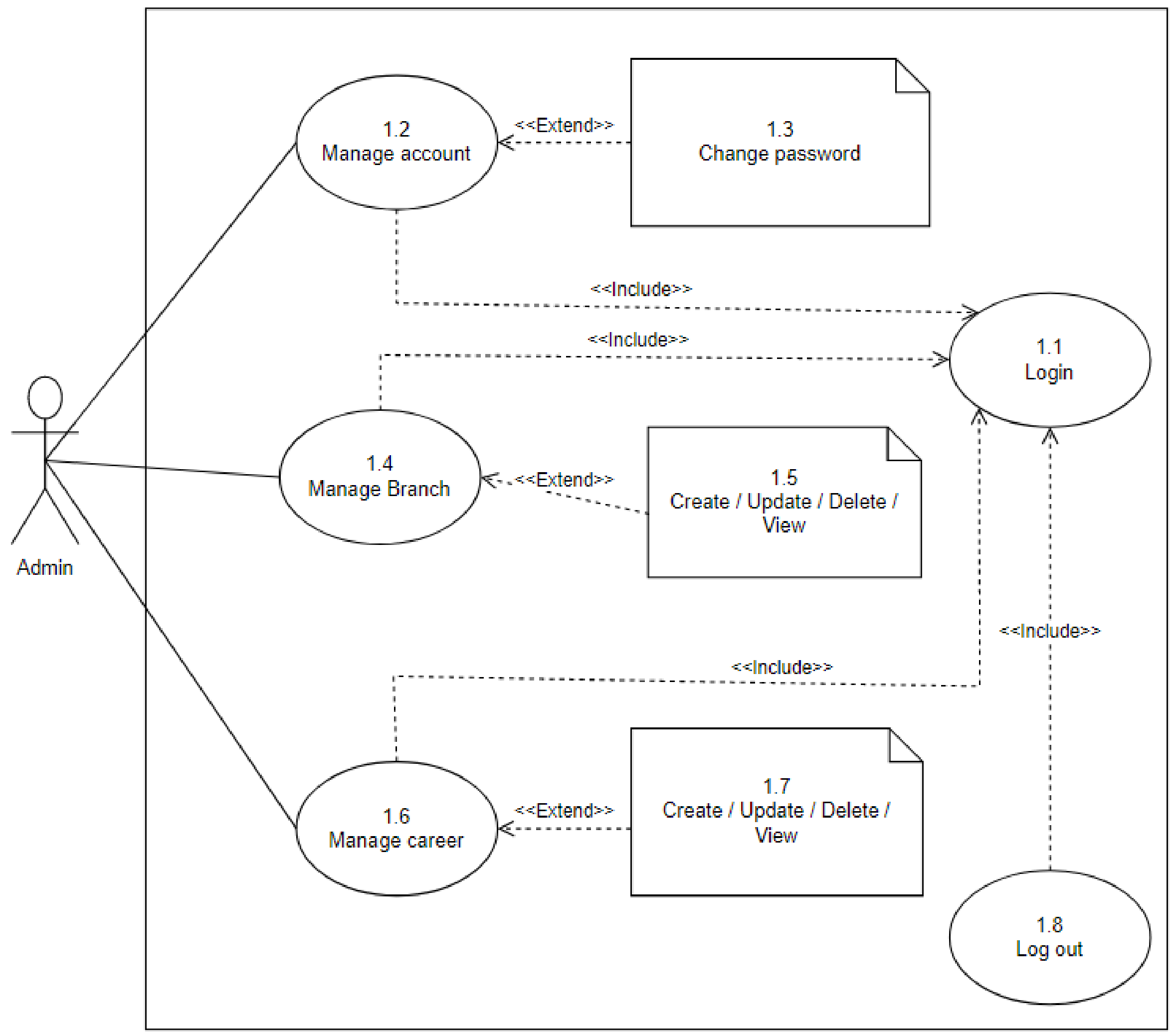


FIGURE 6: USE CASE DIAGRAM FOR ADMIN

Primary Use-case scenario 1:

Use case ID	UC-1.1
Use case name	Login
Actors	Admin
Pre-Condition(s)	• Admin has to Admin. • Separate account is preferred. • A separate account is recommended. • The admin account has been granted access.

Flow of events: **Primary scenario** . 1. Admin enters password account with an existing admin account. . 2. The system verifies the successful login and allows the administrator to access the application. 3. Admin goes to the admin home page.	
Post-Condition(s)	• Admin successfully logged into the system.

Possible secondary scenario which could occur:

- User entered incorrect login information.
- The account role isn’t admin.

Primary Use-case scenario 2:

Use case ID	UC-1.2
Use case name	Manager Account
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to manage account. • User has to Admin.

Flow of events: **Primary scenario** 1. Admin login to the system. 2. The system validates login information successfully and allows the admin to access the application. 3. Admin click on the button “Manager account” in navbar to see all account.	
Post-Condition(s)	• Admin manages the account.

Possible secondary scenario which could occur:

- Login session has expired.
- The system checks to ensure that the admin account is not incorrectly logged in and is unable to access the application.
- The database has no information.

Primary Use-case scenario 3:

Use case ID	UC-1.3
Use case name	Change password
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to change password account. • User must be Admin.
Flow of events: **Primary scenario** 1. Admin must log in to the system. 2. Admin clicks "Manage account " button in the nav bar, the system will go to view all account page. 3. Admin clicks "Change password" button to change password account. 4. Admin enters all information of new password in the data form to change password account. 5. Change password account successfully and the system saves data to the database.	
Post-Condition(s)	• Admin successfully changed password (if existing user)

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the admin account is not logged in correctly and is unable to access the application.
- Admin who enters the confirm password that does not match the new password will notbe able to change the password.
- The account that does not exist.

Primary Use-case scenario 4:

Use case ID	UC-1.4
Use case name	Manager Branch
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to manage branch. • User must be Admin.
Flow of events: **Primary scenario** 1. Admin login to the system. 2. The system validates login information successfully and allows the admin to access the application. 3. Admin click on the button "Manager branch" in the nav bar to see all car (is existing).	
Post-Condition(s)	• Admin manages all branch.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the admin account is not logged in correctly and is unable to access the application.
- No data in database.

Primary Use-case scenario 5:

Use case ID	UC-1.5
Use case name	Create (Create branch)
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to create a new branch. • User must be Admin.
Flow of events: **Primary scenario** 1. Admin must log in to the system. 2. Admin clicks "Manage branch" button, the system will go to view all branch page. 3. Admin clicks "Create a new branch" button to create a branch. 4. Admin enters all the information in the data form to create a new branch. 5. Create a new branch successfully and the system saves data to the database.	
Post-Condition(s)	• Admin successfully created a new branch. • The data is saved to the database.

Possible secondary scenario which could occur:

- Login session has expired.
- The system checks to ensure that the admin account is not incorrectly logged in and is unable to access the application.

- A branch information authentication system already exists, but a new branch that does not match an existing branch must be created.
- Wrong data entered in the form.

Primary Use-case scenario 6:

Use case ID	UC-1.5
Use case name	Update (Update branch)
Actors	Admin
Pre-Condition(s)	• Admin must be login into the system if admin want to update branch. • User must be Admin.

Flow of events: **Primary scenario** 1. Admin must log in to the system. 2. Admin clicks "Manage branch" button, the system will go to view all branch page. 3. Admin clicks "Update" button to create a branch. 4. Admin enters all the information in the data form to update new branch. 5. Update car successfully and the system saves data to the database.	
Post-Condition(s)	• Admin successfully updated branch (if existing user)

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the admin account is not logged in correctly and is unable to access the application.
- Branch that does not exist.
- Wrong data entered in the form.

Primary Use-case scenario 7:

Use case ID	UC-1.5
Use case name	View (View branch)
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to view branch • User must be Admin.
Flow of events: **Primary scenario** 6. Admin must log in to the system. 7. Admin clicks "Manage branch" button, the system will go to view all car page. 8. Admin clicks "View" button to view a branch. 9. Admin enters all the information in the data form to view branch. 10. Update car successfully and the system saves data to the database.	
Post-Condition(s)	• Admin successfully viewed all branch

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the admin account is not logged in correctly and is unable to access the application.
- No data in database.

Primary Use-case scenario 8:

Use case ID	UC-1.5
Use case name	Delete (Delete branch)
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to delete branch. • User must be Admin.
Flow of events: **Primary scenario** 1. Admin must log in to the system. 2. Admin clicks "Manage branch" button, the system will go to view all branch. 3. Admin clicks "Delete" button (if existing car) to delete branch.	
Post-Condition(s)	• Admin successfully deleted branch (if existing branch)

Possible secondary scenario which could occur:

- Login session has expired.
- The system validates that the admin account is not properly logged in and so cannot access the application.
- Branch that does not exist.

Primary Use-case scenario 9:

Use case ID	UC-1.6
Use case name	Manage career
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to manage career. • User must be Admin
Flow of events: **Primary scenario** 1. Admin login to the system. 2. The system validates login information successfully and allows the admin to access the application. 3. Admin click on the button “Manager career” to see all car (is existing).	
Post-Condition(s)	• Admin manages all career.

Possible secondary scenario which could occur:

- Login session has expired.

- The system validates that the admin account is not properly logged in and so cannot access the application.
- The database has no information.

Primary Use-case scenario 10:

Use case ID	UC-1.7
Use case name	Create (Create career)
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to create a new career. • User must be Admin.
Flow of events: **Primary scenario** 6. Admin must log in to the system. 7. Admin clicks "Manage career" button, the system will go to view all car page. 8. Admin clicks "Create a new career" button to create a career. 9. Admin enters all the information in the data form to create a new career. 10. Create a new career successfully and the system saves data to the database.	
Post-Condition(s)	• Admin successfully created a new career. • The data is saved to the database.

Possible secondary scenario which could occur:

- Login session has expired.
- The system validates that the admin account is not properly logged in and so cannot access the application.

Primary Use-case scenario 11:

Use case ID	UC-1.7
Use case name	View (view career)
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to view all career). • User must be Admin.

Flow of events: **Primary scenario** 1. Admin must log in to the system. 2. Admin clicks "Manage career" button, the system will go to view all car page. Admin view career.	
Post-Condition(s)	• Admin successfully viewed career.

Possible secondary scenario which could occur:

- Login session has expired.
- The system validates that the admin account is not properly logged in and so cannot access the application.

- The database has no information.

Primary Use-case scenario 12:

Use case ID	UC-1.7
Use case name	Update (Update career)
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to update career. • User must be Admin.
Flow of events: **Primary scenario** 1. Admin must log in to the system. 2. Admin clicks "Manage career" button, the system will go to view all career page. 3. Admin clicks "Update" button to create a career. 4. Admin enters all the information in the data form to update new career. Update car successfully and the system saves data to the database.	
Post-Condition(s)	• Admin successfully updated career

Possible secondary scenario which could occur:

- Login session has expired.

- The system validates that the admin account is not properly logged in and so cannot access the application.
- Career that does not exist.
- Data incorrect entered in the form.

Primary Use-case scenario 13:

Use case ID	UC-1.7
Use case name	Delete (Delete career)
Actors	Admin
Pre-Condition(s)	• Admin must be logged into the system if admin want to delete career. • User must be Admin.
Flow of events: **Primary scenario** 3. Admin must log in to the system. 4. Admin clicks "Manage career" button, the system will go to view all career 3. Admin clicks "Delete" button (if existing career) to delete career.	
Post-Condition(s)	• Admin successfully deleted branch

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the admin account is not logged in correctly and is unable to access the application.
- Career that does not exist.

Primary Use-case scenario 14:

Use case ID	UC-1.8
Use case name	Logout
Actors	Admin
Pre-Condition(s)	• User must be logged into the system.
Flow of events: **Primary scenario** 1. Admin must log in to the system. 2. Admin clicks "Logout" button, the system will logout and go to home page.	
Post-Condition(s)	• Admin successfully logout.

Possible secondary scenario which could occur:

- Login session has expired.
- The system validates that the admin account is not properly logged in and so cannot access the application.

8.4.2. Use case diagram, primary and secondary use-case scenario for user

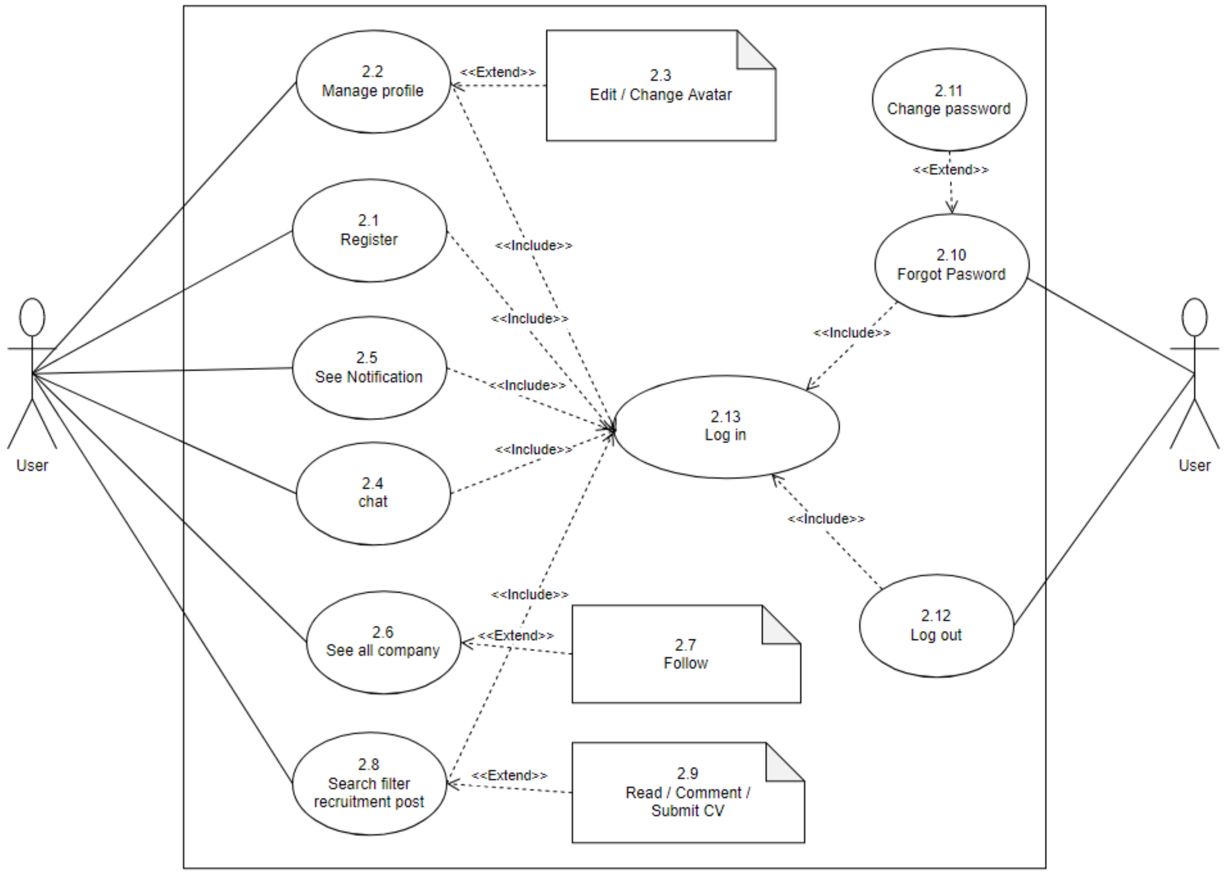


FIGURE 7: USE CASE DIAGRAM FOR USER

Primary Use-case scenario 1:

Use case ID	UC-2.1
Use case name	Register
Actors	User
Pre-Condition(s)	• User must use real email. • The admin device has been connected to the internet whenperforming register.
Flow of events: **Primary scenario** 1. User clicks "Register" button, system will go to register page. 2. User enters all information in the data form to register account. 3. User go to login page.	
Pre-Condition(s)	User successfully registers.

Possible secondary scenario which could occur:

- User entered incorrect information form register.

Primary Use-case scenario 2:

Use case ID	UC-2.2
Use case name	Manage profile
Actors	User
Pre-Condition(s)	• User must login to the system if user manages their profile. • User must be user.
Flow of events: **Primary scenario** 1. User login to the system. 2. The system validates login information successfully and allows user to access the application. 3. User click on the button “Manager profile” to manage their profile.	
Pre-Condition(s)	User see their profile (if existing user)

Possible secondary scenario which could occur:

- Login session has expired.
- The system confirms that the user account is not properly logged in and hence cannot access the application.

Primary Use-case scenario 3:

Use case ID	UC-2.3
Use case name	Edit (Update my profile)
Actors	User
Pre-Condition(s)	• User must be logged into the system if user want to update their profile. • User must be user.
Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks "Edit profile" button, the system will go to update profile page. 3. User enters all information in the data form to update their profile. 4. Update profile successfully and the system saves data to the database.	
Pre-Condition(s)	User successfully updated profile (if existing user).

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the staff account is not logged in correctly and is unable to access the application.
- User does not enter data in the form.

Primary Use-case scenario 4:

Use case ID	UC-2.3
Use case name	Change avatar
Actors	User
Pre-Condition(s)	• User must be logged into the system.
Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks "profile" button, the system will go to change profile page. 3. User upload new avatar they want to change avatar. 5. Change avatar successfully and the system saves image URL to the database.	
Pre-Condition(s)	User successfully change avatar.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that user account is not logged in correctly and is unable to access the application.
- User does not choose new avatar.

Primary Use-case scenario 5:

Use case ID	UC-2.4
Use case name	Chat
Actors	User
Pre-Condition(s)	• User must be logged into the system if user want to chat.
Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks "Message" button, the system will go to chat message page. 3. User can find people	
Pre-Condition(s)	User successfully registers.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the driver account is not logged in correctly and is unable toaccess the application.
- The system cannot find other users.

Primary Use-case scenario 6:

Use case ID	UC-2.5
Use case name	See Notification
Actors	User
Pre-Condition(s)	• User must be logged into the system.
Flow of events: **Primary scenario** User clicks "notification" Icon, system will go to page notification.	
Pre-Condition(s)	User seen notification.

Possible secondary scenario which could occur:

- Login session expired.
- The system verifies that the user account is not logged in correctly and is unable toaccess the application.

Primary Use-case scenario 7:

Use case ID	UC-2.6
Use case name	See all company
Actors	User
Pre-Condition(s)	• User must be logged into the system if user want to see all company.
Flow of events:	

Primary scenario 1. User must log in to the system. 2. User clicks "See all company" button, the system will go to view all company.	
Pre-Condition(s)	User successfully viewed all company.

Possible secondary scenario which could occur:

- Login session expired.
- The system verifies that the driver account is not logged in correctly and is unable toaccess the application.

Primary Use-case scenario 8:

Use case ID	UC-2.7
Use case name	Follow
Actors	User
Pre-Condition(s)	• User must be logged into the system if user want to follow company.
Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks "See all company" button, the system will go to view all company. 3. User clicks "Follow" button, if they want follow a company. 4. Follow successfully and the system saves data to the database.	
Pre-Condition(s)	User successfully registers.

Possible secondary scenario which could occur:

- Login session expired.
- The system verifies that the user account is not logged in correctly and is unable toaccess the application.

Primary Use-case scenario 9:

Use case ID	UC-2.8
Use case name	Search filter recruitment post
Actors	User
Pre-Condition(s)	• User must be logged into the system if user want to Search filter recruitment post.
Flow of events: **Primary scenario** 1. User go to home page. 2. User select list by occupation, experience, salary, and so on. 3. The system filters and displays relevant recruitment post to users	
Pre-Condition(s)	User seen recruitment post

Possible secondary scenario which could occur:

Primary Use-case scenario 10:

Use case ID	UC-2.9
Use case name	Read (read recruitment post)
Actors	User
Pre-Condition(s)	• User must be logged into the system.
Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks recruitment post, the system will go to view recruitment post.	
Pre-Condition(s)	User successfully viewed recruitment post.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the user account is not logged in correctly and is unable toaccess the application.

Primary Use-case scenario 11:

Use case ID	UC-2.9
Use case name	Comment (comment recruitment post)
Actors	User
Pre-Condition(s)	• User must be logged into the system if user want to comment recruitment post.
Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks recruitment post, the system will go to view recruitment post. 3.User choose 1 recruitment post, user comment under recruitment post.	
Pre-Condition(s)	User successfully comment.

Possible secondary scenario which could occur:

- Login session expired.

Primary Use-case scenario 12:

Use case ID	UC-2.9
Use case name	Submit CV
Actors	User
Pre-Condition(s)	• User must be logged into the system if driver want to submit Cv.

Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks recruitment post, the system will go to view recruitment post. 3. User clicks "submit CV" button, choose file and upload file PDF	
Pre-Condition(s)	User successfully submitted CV

Possible secondary scenario which could occur:

- Login session has expired.
- User cannot upload CV.
- User does not submit files in a format PDF

Primary Use-case scenario 13:

Use case ID	UC-2.10
Use case name	Forgot password
Actors	User
Pre-Condition(s)	• User must access the application. • User is received new password from email.
Flow of events: **Primary scenario** 1. User go to login page. 2. User clicks "Forgot password" button, the system will go to forgot password page. 3. User enters all the information in the data form of forgot password page. 4. User receive link reset password from email	
Pre-Condition(s)	User successfully reset password.

Possible secondary scenario which could occur:

- Account that does not exist.
- Data incorrect entered in the form.
- User does not receive email.

Primary Use-case scenario 14:

Use case ID	UC-2.11
Use case name	Change password
Actors	User
Pre-Condition(s)	• User must be logged into the system if user want to change their password.
Flow of events: **Primary scenario** 1. Staff must log in to the system. 2. User clicks "Change password" button, the system will go to change password page. 3. User enters all information of new password in the data form to change password.	

4. Change password successfully and the system saves data to the database.	
Pre-Condition(s)	User successfully changed password (if existing user)

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the user account is not logged in correctly and is unable toaccess the application.
- User does not enter data in the form.
- User who enters the confirm password that do not match the new password will not beable to change the password.

Primary Use-case scenario 15:

Use case ID	UC-2.12
Use case name	Log out
Actors	User
Pre-Condition(s)	• User must be logged into the system.
Flow of events: **Primary scenario** 1. User must log in to the system. 2. User clicks "Logout" button, the system will logout and go to home page.	
Pre-Condition(s)	User successfully logout.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the user account is not logged in correctly and is unable to access the application.

Primary Use-case scenario 16:

Use case ID	UC-2.13
Use case name	Login
Actors	User
Pre-Condition(s)	• User must be user. • Separate account is preferred. • The user account has been authorized. • The user device has been connected to the internet when performing login.
Flow of events: **Primary scenario** 1. User enters password account or login with an existing User account. 2. The system verifies the successful login and allows the User to access the application.	

3. User goes to the dashboard page.	
Pre-Condition(s)	User successfully logged into the system.

Possible secondary scenario which could occur:

- User entered incorrect login information.
- The account role is not user.

8.4.3. Use case diagram, primary and secondary use-case scenario for company

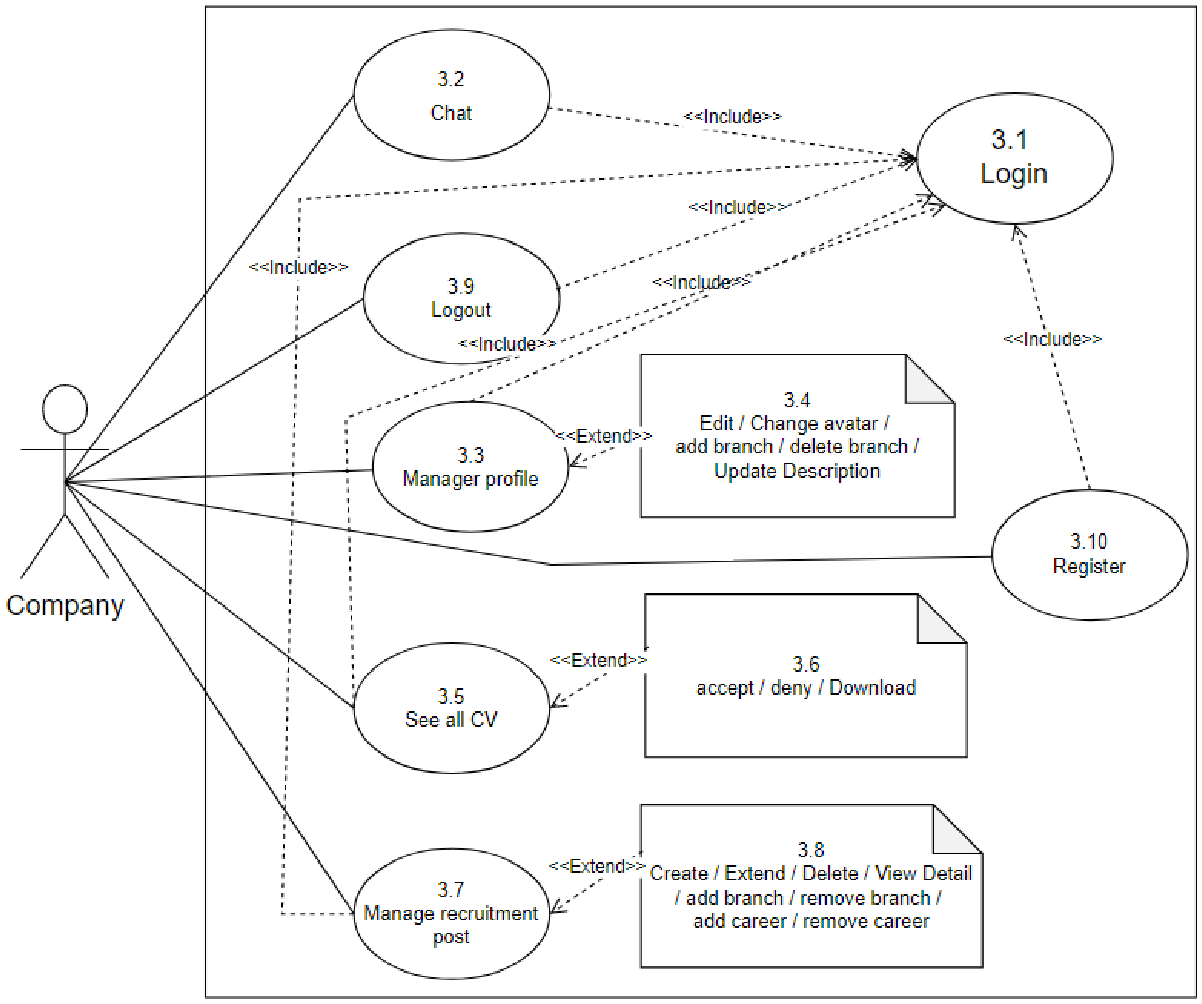


FIGURE 8: USE CASE DIAGRAM FOR COMPANY

Primary Use-case scenario 1:

Use case ID	UC-3.1
Use case name	Login
Actors	Company
Pre-Condition(s)	• User must be company. • Separate account is preferred. • The Company account has been authorized. • The Company device has been connected to the internet when performing login.

Flow of events: **Primary scenario** 1. User company account and enters password. 2. The system verifies the successful login and allows the company to access the application. 3. User goes to the dashboard page.	
Pre-Condition(s)	Company successfully logged into the system.

Possible secondary scenario which could occur:

- Company entered incorrect login information.
- The account role is not company.

Primary Use-case scenario 2:

Use case ID	UC-3.2
Use case name	Chat
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to chatmessage. • User must be Company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Message" button, the system will go to chat message page. 3. Company can find people 4. Company message with people who have permission in the system	
Pre-Condition(s)	Company chat message.

Possible secondary scenario which could occur:

- Login session expired.
- The system verifies that the company account is not logged in correctly and is unable toaccess the application.
- The system cannot find other users.

Primary Use-case scenario 3:

Use case ID	UC-3.3
Use case name	Manager profile
Actors	Company
Pre-Condition(s)	• Company must login to the system if company manages their profile. • User must be company.
Flow of events: **Primary scenario** 1. Company login to the system. 2. The system validates login information successfully and allows the company to	

access theapplication. 3. Company clicks on the button “Manager profile” to manage their profile.	
Pre-Condition(s)	Company seen their profile (if existing user).

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable toaccess the application.

Primary Use-case scenario 4:

Use case ID	UC-3.4
Use case name	Edit (update profile)
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to update theirprofile. • User must be company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Edit profile" button, the system will go to update profile page. 3. Company enters all information in the data form to update their profile. 4. Update profile successfully and the system saves data to the database.	
Pre-Condition(s)	Company successfully updated profile (if existing user)

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable toaccess the application.
- Company does not enter data in the form.

Primary Use-case scenario 5:

Use case ID	UC-304
Use case name	Change avatar
Actors	Company
Pre-Condition(s)	• Company must be logged into the system.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "profile" button, the system will go to change profile page. 3. Company uploads new avatar they want to change avatar. 4. Change avatar successfully and the system saves URL image to the database.	

Pre-Condition(s)	Company successfully changed avatar.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that user account is not logged in correctly and is unable to access the application.
- User does not choose new avatar.

Primary Use-case scenario 6:

Use case ID	UC-3.4
Use case name	Add branch (add branch to company)
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to branch to company. • User must be company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Edit profile" button, the system will go to update profile page. 3. Company clicks " Add branch" icon to add branch to company. System saves data to the database.	
Pre-Condition(s)	Company successfully added branch to company.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- Branch that does not exist.

Primary Use-case scenario 7:

Use case ID	UC-3.4
Use case name	Delete branch
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to delete branch to company. • User must be Company.
Flow of events: **Primary scenario** Company must log in to the system. Company clicks "Edit profile" button, the system will go to update profile page. Company clicks "Delete branch" icon to delete branch to company. The system saves data to the database.	

Pre-Condition(s)	Company successfully logged into the system.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- Branch that does not exist.

Primary Use-case scenario 8:

Use case ID	UC-3.4
Use case name	Description (description company)
Actors	Company
Pre-Condition(s)	• User must be company. • Separate account is preferred. • The Company account has been authorized. • The Company device has been connected to the internet when performing login.
Flow of events: **Primary scenario** 4. User enters company account and password. 5. The system verifies the successful login and allows the company to access the application. 6. User goes to the dashboard page.	
Pre-Condition(s)	Company successfully logged into the system.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- No data in database.

Primary Use-case scenario 9:

Use case ID	UC-3.5
Use case name	See all CV
Actors	Company
Pre-Condition(s)	• User must be company. • Company has to login into the system if company want to see all CV
Flow of events: **Primary scenario** 1. Company has to login to the system. 2. Company click "see all cv " button, the system will go to see cv page.	
Pre-Condition(s)	Company successfully seen all CV.

Possible secondary scenario which could occur:

- Login session has expired
- The account role is not company.
- No data in system

Primary Use-case scenario 10:

Use case ID	UC-3.6
Use case name	Accept CV
Actors	Company
Pre-Condition(s)	• User must be company. • Company has to login into the system if company want to accept CV
Flow of events: **Primary scenario** 1. Company click "see all cv" button, the system will go to see cv page. 2. Company clicks "accept" icon, The system saves data to the database.	
Pre-Condition(s)	Company successfully accepted CV.

Possible secondary scenario which could occur:

- The account role is not company.

Primary Use-case scenario 11:

Use case ID	UC-3.6
Use case name	Deny CV
Actors	Company
Pre-Condition(s)	• User must be company. • Company has to login into the system if company want to deny the CV
Flow of events: **Primary scenario** 1. Company click "see all cv" button, the system will go to see cv page. 2. Company clicks "deny" icon, The system saves data to the database.	
Pre-Condition(s)	Company successfully denied Cv.

Possible secondary scenario which could occur:

- The account role is not company.

Primary Use-case scenario 12:

Use case ID	UC-3.6
Use case name	Download CV
Actors	Company
Pre-Condition(s)	• User must be company. • Company has to login into the system if company want to download Cv.

Flow of events: **Primary scenario** 1. Company click “see all cv“ button, the system will go to see cv page. 2. Company clicks “download” icon to download file zip.	
Pre-Condition(s)	Company successfully download CV.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- CV that does not exist.

Primary Use-case scenario 13:

Use case ID	UC-3.7
Use case name	Manger recruitment post
Actors	Company
Pre-Condition(s)	• Company has to login into the system if company want to manager recruitment post. • User must be company
Flow of events: **Primary scenario** 1. Company login to the system. 2. The system validates login information successfully and allow the company to access the system. 3. Company clicks “manager recruitment post” button to see all recruitment post.	
Pre-Condition(s)	Company successfully seen all recruitment post.

Possible secondary scenario which could occur:

- Login session has expired.
- No data in database

Primary Use-case scenario 14:

Use case ID	UC-3.8
Use case name	Create
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if admin want to create a new recruitment post. • User must be Company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Manage recruitment post " button, the system will go to view all	

recruitment post page. 3. Company clicks "Create a new recruitment post " button to create a recruitment post. 4. Company enters all the information in the data form to create a new recruitment post. 5. Create a new recruitment post successfully and the system saves data to the database.	
Pre-Condition(s)	Company successfully created recruitment post.

Possible secondary scenario which could occur:

- Login session has expired.
- The system checks to ensure that the company account is not incorrectly logged in and is unable to access the system.

Primary Use-case scenario 15:

Use case ID	UC-3.8
Use case name	Extend
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to extend arecruitment post. • User must be Company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Manage recruitment post " button, the system will go to view all recruitment post page. 3. Company clicks "Extend recruitment post " button to extend a recruitment post. 4. Company enters all the date in the data form to extend a recruitment post.	
Pre-Condition(s)	Company successfully extended recruitment post

Possible secondary scenario which could occur:

- Login session has expired.
- User is not a company role.

Primary Use-case scenario 16:

Use case ID	UC-3.8
Use case name	Delete (Delete recruitment post)
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company wants to delete recruitment post. • User must be Company.
Flow of events: **Primary scenario**	

1. Company must log in to the system. 2. Company clicks "Manage recruitment post." button, the system will go to view all recruitment post. 3. Company clicks "Delete" button to delete recruitment post.	
Pre-Condition(s)	Company successfully deleted recruitment post.

Possible secondary scenario which could occur:

- Login session has expired.
- The system validates that the company account is not properly logged in and so cannot access the application.
- Recruitment post that does not exist.

Primary Use-case scenario 17:

Use case ID	UC-3.8
Use case name	View Detail
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to view recruitment post. • User must be company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Manage recruitment post " button, the system will go to view all recruitment post page. 3. Company clicks "View" button to view a recruitment post. 4. Company enters all the information in the data form to view recruitment post. 5. Update recruitment post successfully and the system saves data to the database.	
Pre-Condition(s)	Company successfully logged into the system.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the Company account is not logged incorrectly and is unable to access the application.
- No data in database.

Primary Use-case scenario 18:

Use case ID	UC-3.8
Use case name	Add branch
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to add new branch. • User must be company.

Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Manage recruitment post." button, the system will go to view all recruitment post. 3. Company clicks "add branch" button to add branch into a recruitment post.	
Pre-Condition(s)	Company successfully added new branch into a recruitment post.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- Branch that does not exist.

Primary Use-case scenario 19:

Use case ID	UC-3.8
Use case name	Remove branch
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to remove branch. • User must be company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Manage recruitment post." button, the system will go to view all recruitment post. 3. Company clicks "remove branch" button to remove branch in a recruitment post.	
Pre-Condition(s)	Company successfully logged remove branch in a recruitment post.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- Branch that does not exist.

Primary Use-case scenario 20:

Use case ID	UC-3.8
Use case name	Add career
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to add new career. • User must be company.

Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Manage recruitment post." button, the system will go to view all recruitment post. 3. Company clicks "add career" button to add career into a recruitment post.	
Pre-Condition(s)	Company successfully logged into the system.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- Branch that does not exist.

Primary Use-case scenario 21:

Use case ID	UC-3.8
Use case name	Remove career
Actors	Company
Pre-Condition(s)	• Company must be logged into the system if company want to remove career. • User must be company.
Flow of events: **Primary scenario** 1. Company must log in to the system. 2. Company clicks "Manage recruitment post." button, the system will go to view all recruitment post. 3. Company clicks "remove career" button to remove career in a recruitment post.	
Pre-Condition(s)	Company successfully logged into the system.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.
- Career that does not exist.

Primary Use-case scenario 22:

Use case ID	UC-3.9
Use case name	Log out
Actors	Company
Pre-Condition(s)	• User must be logged into the system.
Flow of events:	

Primary scenario 1. Company must log in to the system. 2. Company clicks "Logout" button, the system will logout and go to home page.	
Pre-Condition(s)	User successfully logout.

Possible secondary scenario which could occur:

- Login session has expired.
- The system verifies that the company account is not logged in correctly and is unable to access the application.

Primary Use-case scenario 22:

Use case ID	UC-3.9
Use case name	Register
Actors	Company
Pre-Condition(s)	• User must use real email. • The company device has been connected to the internet when performing register.
Flow of events: **Primary scenario** 3. User clicks "Register" button, system will go to register page. 4. Company enters all information in the data form to register account. 5. Company go to login page.	
Pre-Condition(s)	Company successfully registers.

Possible secondary scenario which could occur:

User entered incorrect information form register.

8.5. Entity Relationship Diagrams

8.5.1. Physical design (Capacity)

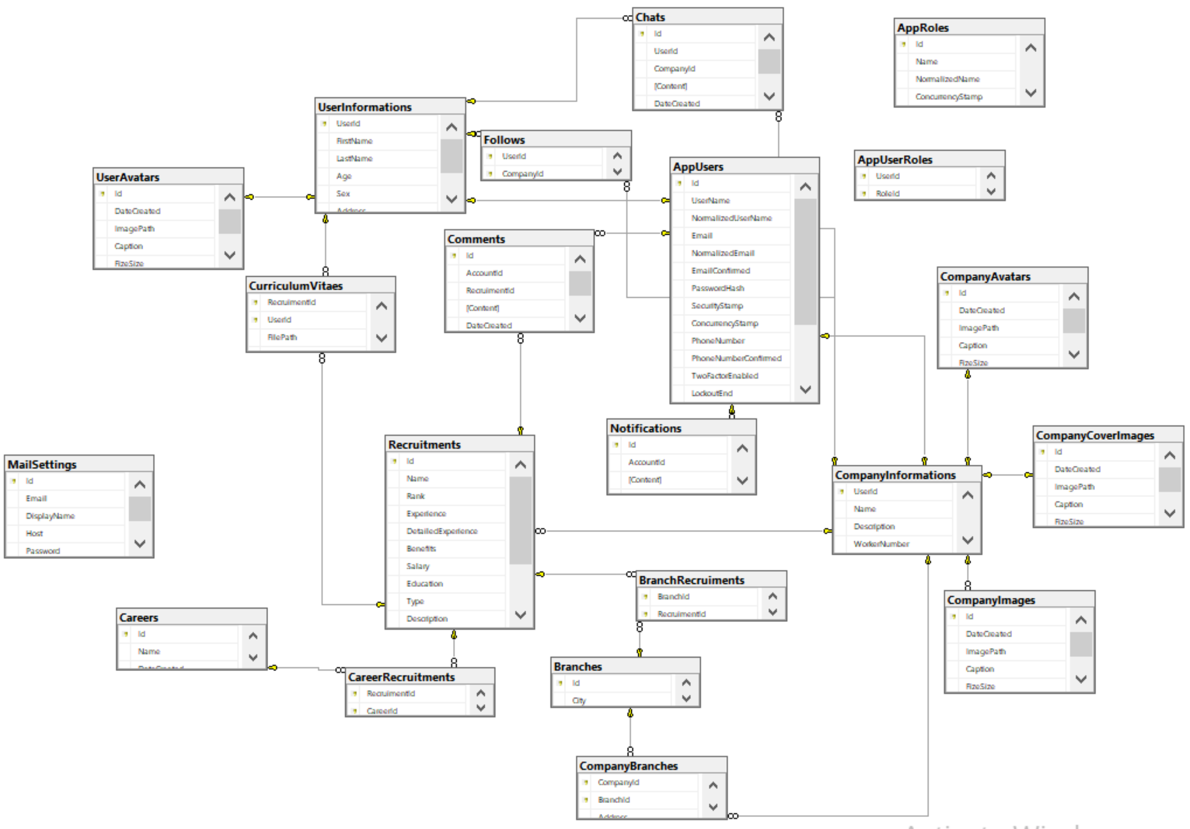


FIGURE 9: PHYSICAL DESIGN

8.6. Wireframes for prototypes

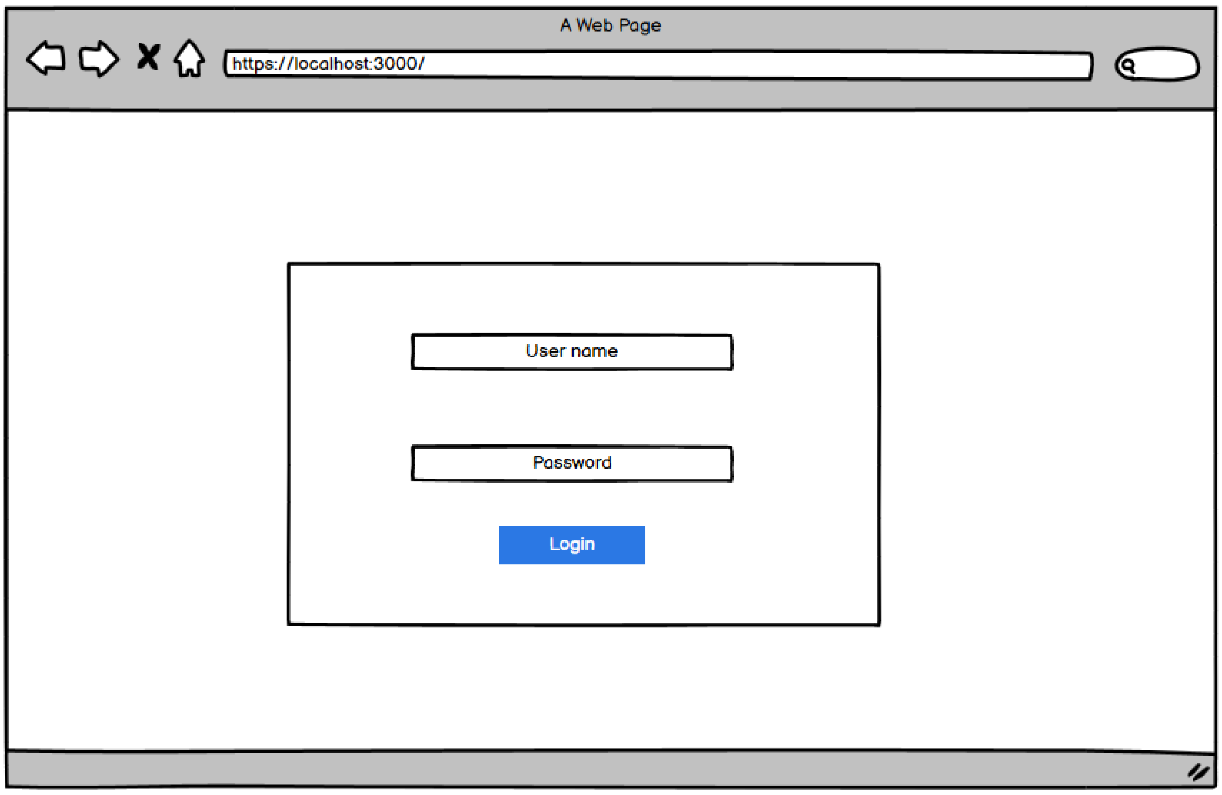
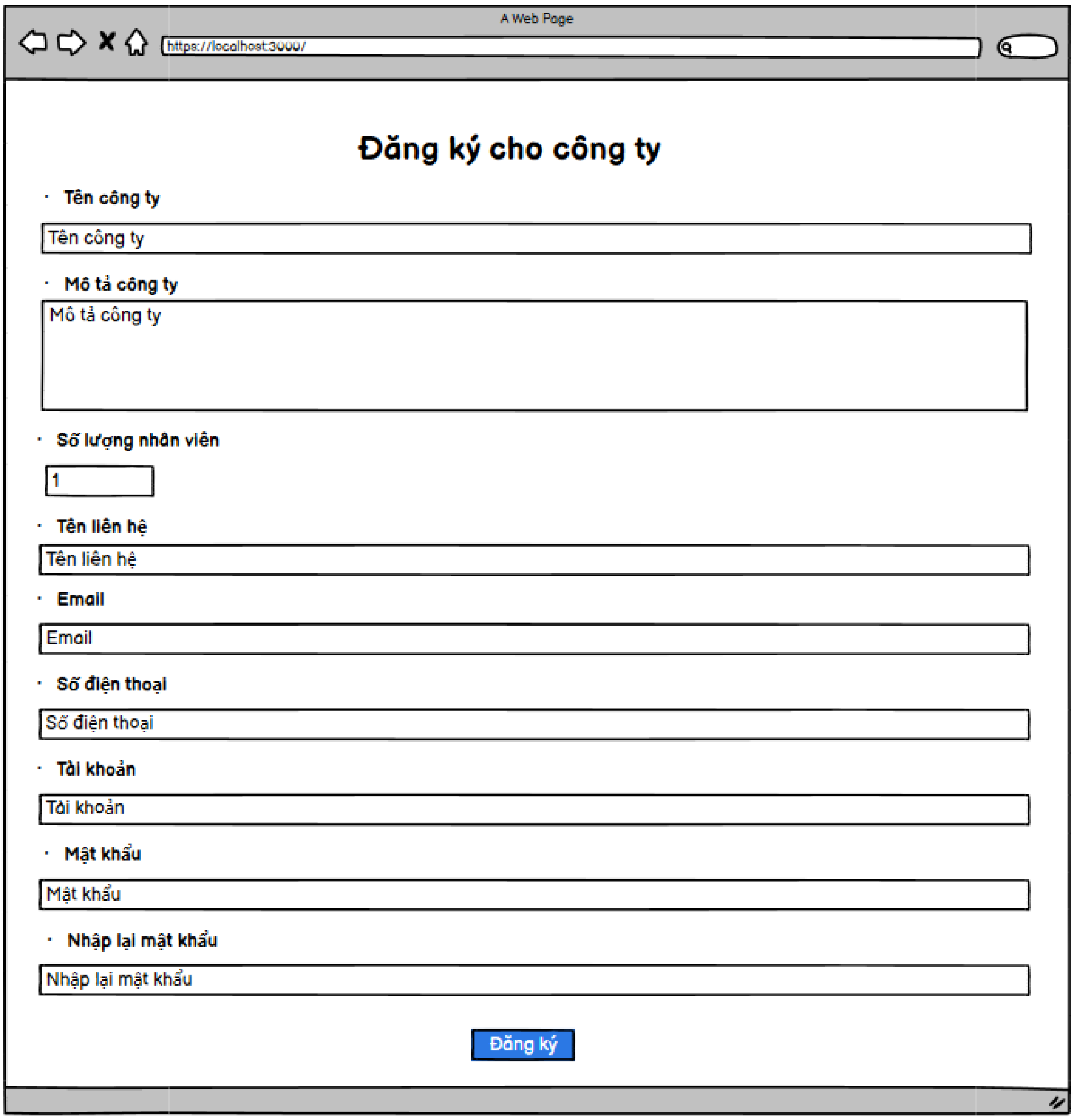


FIGURE 10: LOGIN



Đăng ký cho công ty

- Tên công ty

Tên công ty

- Mô tả công ty

Mô tả công ty

- Số lượng nhân viên

1

- Tên liên hệ

Tên liên hệ

- Email

Email

- Số điện thoại

Số điện thoại

- Tài khoản

Tài khoản

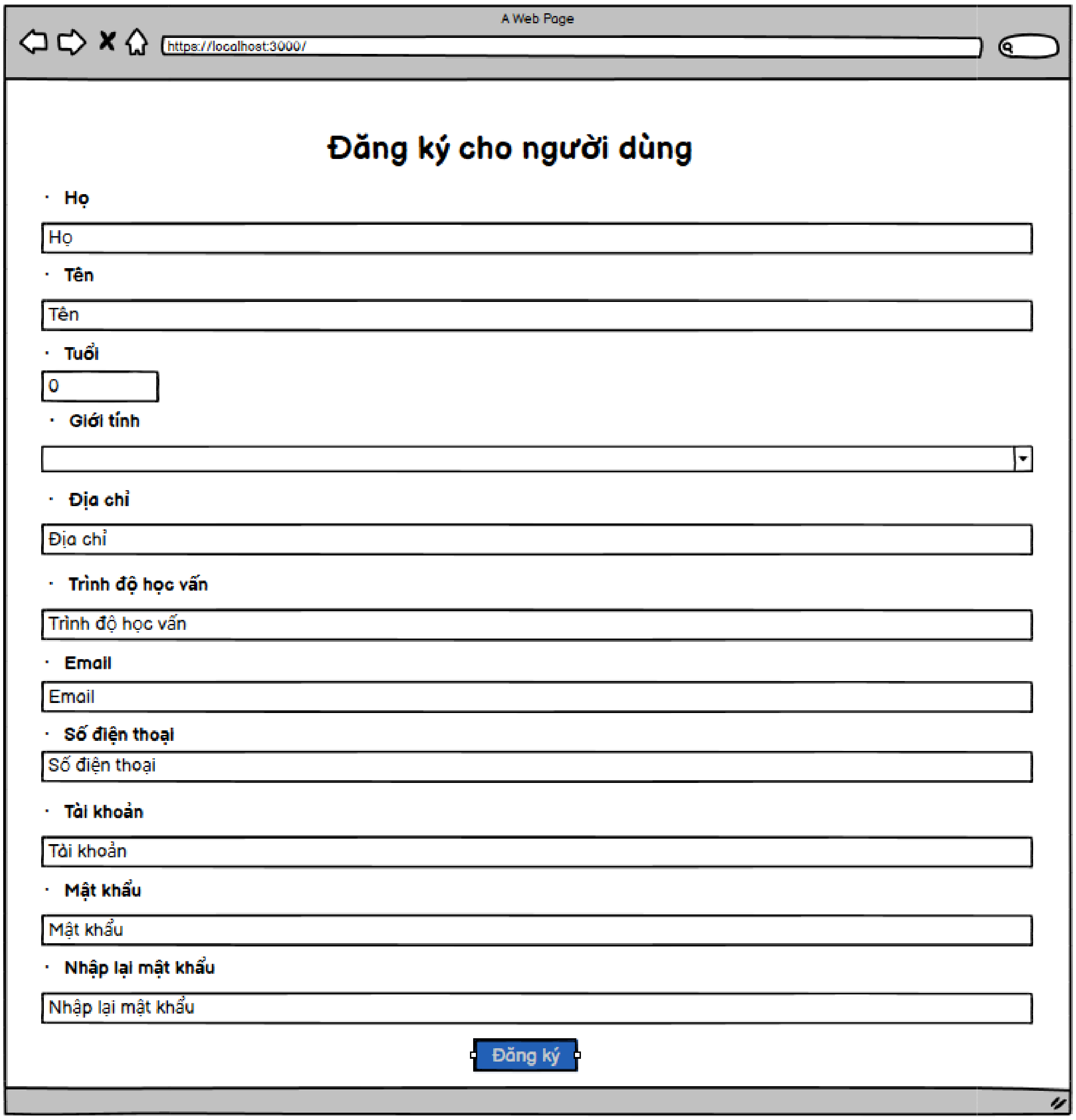
- Mật khẩu

Mật khẩu

- Nhập lại mật khẩu

Nhập lại mật khẩu

FIGURE 11: REGISTER A NEW COMPANY ACCOUNT


A Web Page

https://localhost:3000/

Đăng ký cho người dùng

- Họ

Họ

- Tên

Tên

- Tuổi

0

- Giới tính

- Địa chỉ

Địa chỉ

- Trình độ học vấn

Trình độ học vấn

- Email

Email

- Số điện thoại

Số điện thoại

- Tài khoản

Tài khoản

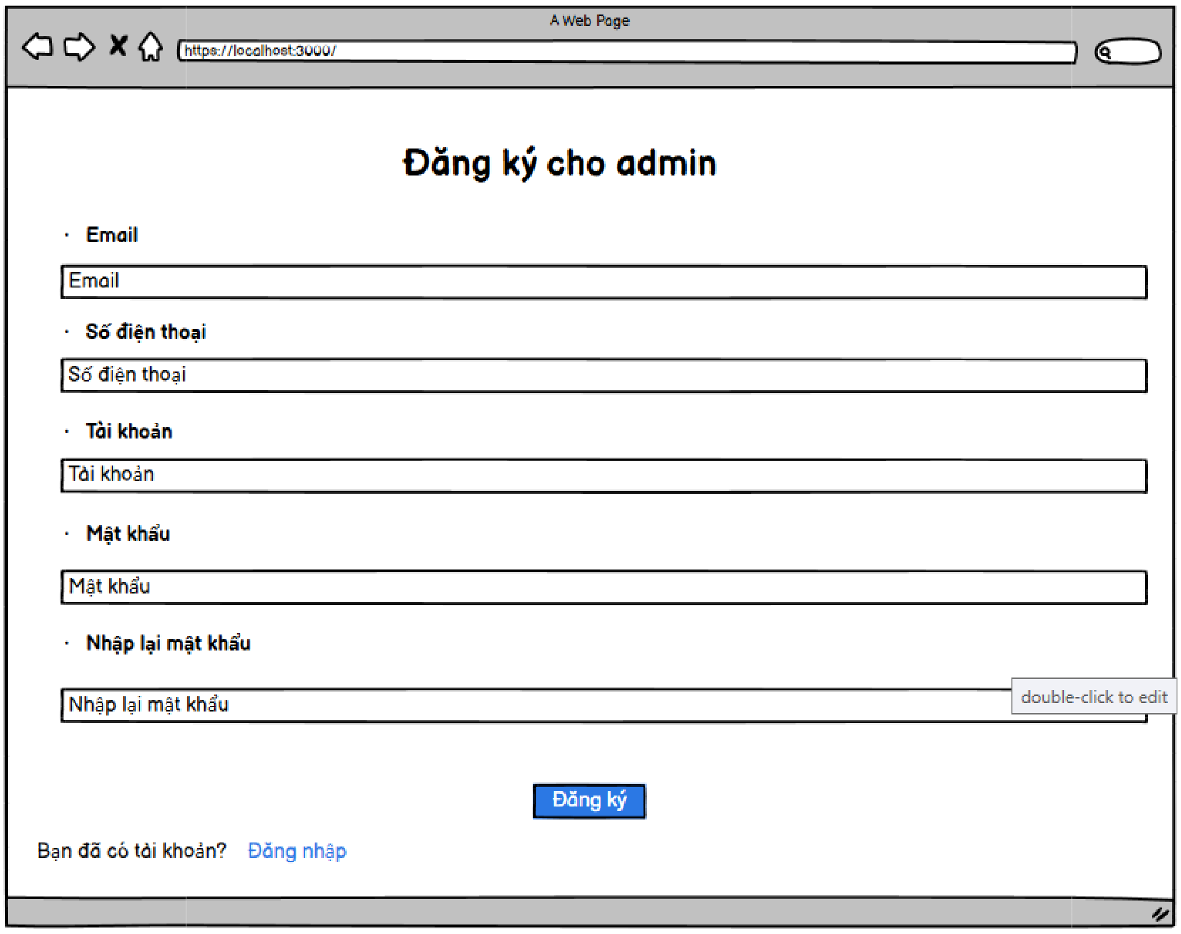
- Mật khẩu

Mật khẩu

- Nhập lại mật khẩu

Nhập lại mật khẩu

FIGURE 12: REGISTER A NEW USER ACCOUNT


A Web Page

https://localhost:3000/

Đăng ký cho admin

- Email

Email

- Số điện thoại

Số điện thoại

- Tài khoản

Tài khoản

- Mật khẩu

Mật khẩu

- Nhập lại mật khẩu

Nhập lại mật khẩu

double-click to edit

Đăng ký

Bạn đã có tài khoản? Đăng nhập

FIGURE 13: REGISTER A NEW ADMIN ACCOUNT

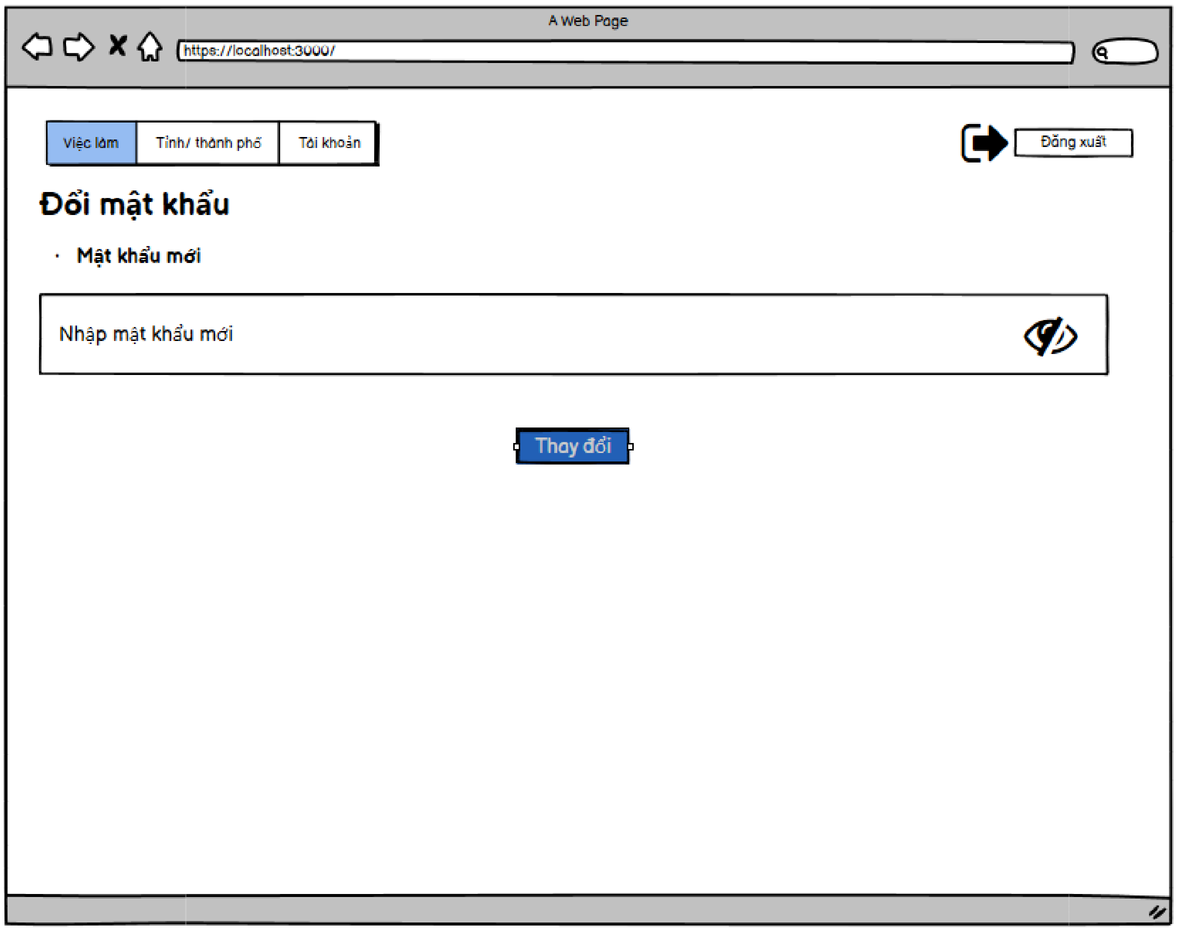


FIGURE 14: CHANGE PASSWORD

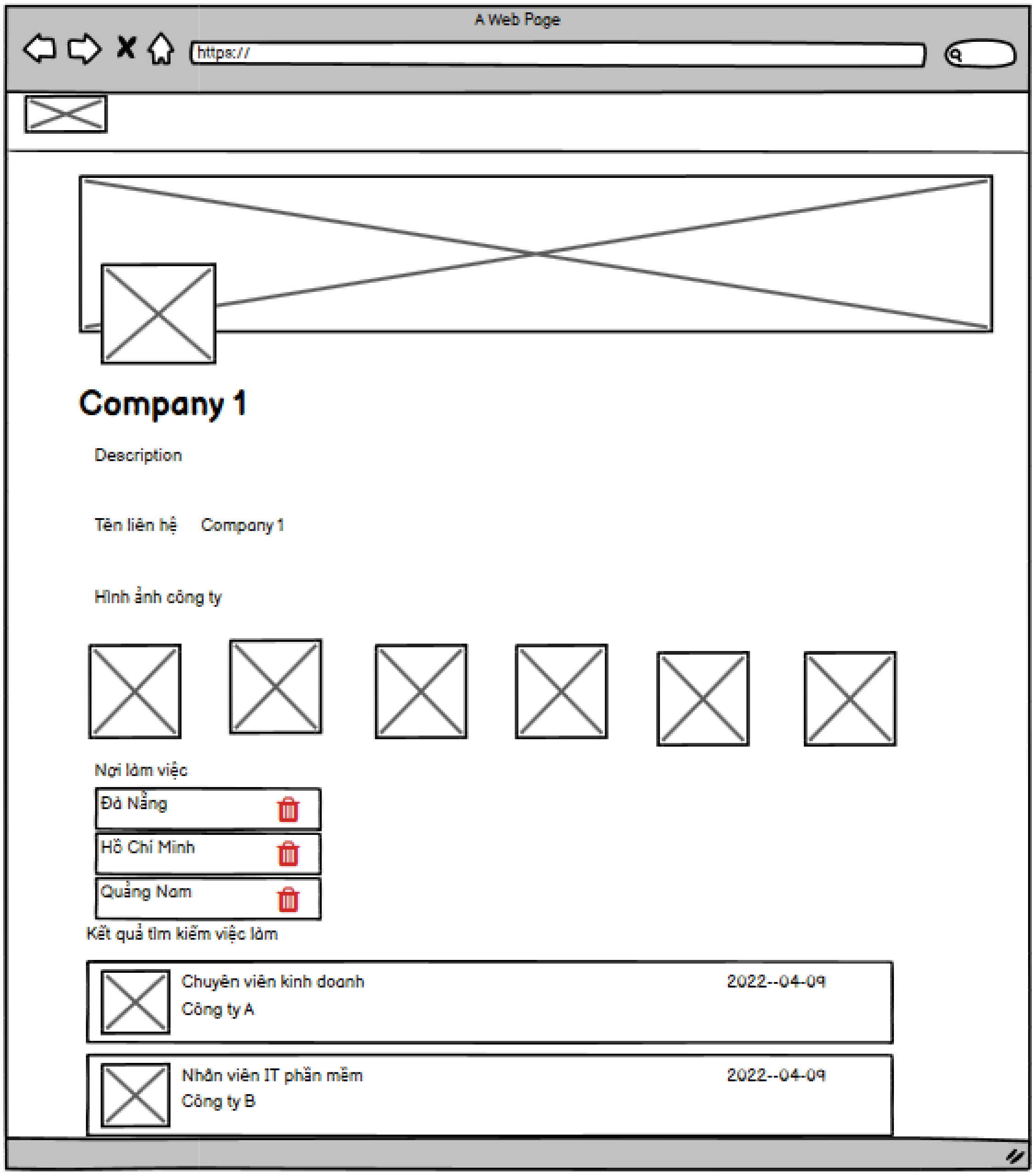


FIGURE 15: PROFILE COMPANY

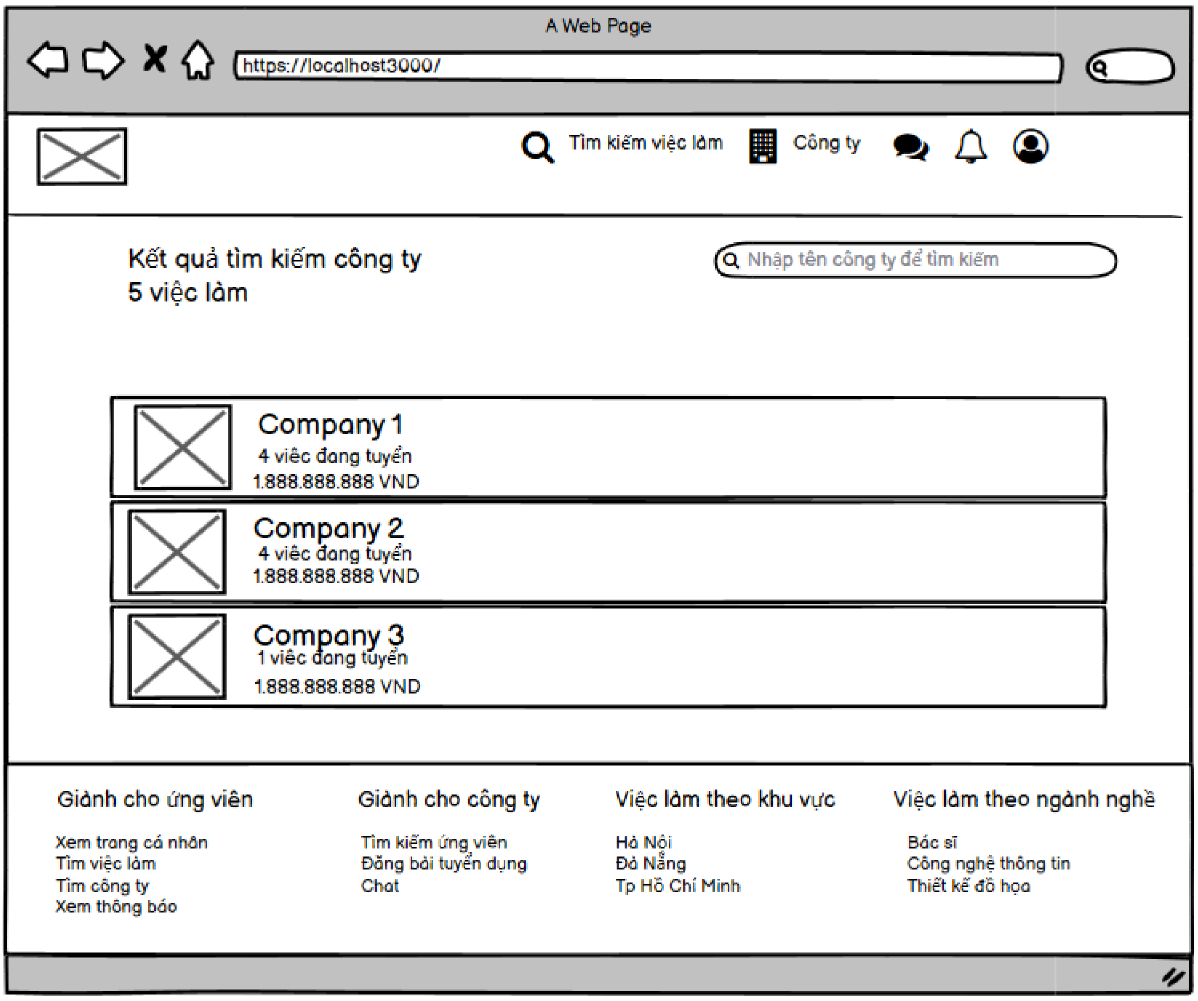


FIGURE 16: SEARCH COMPANY

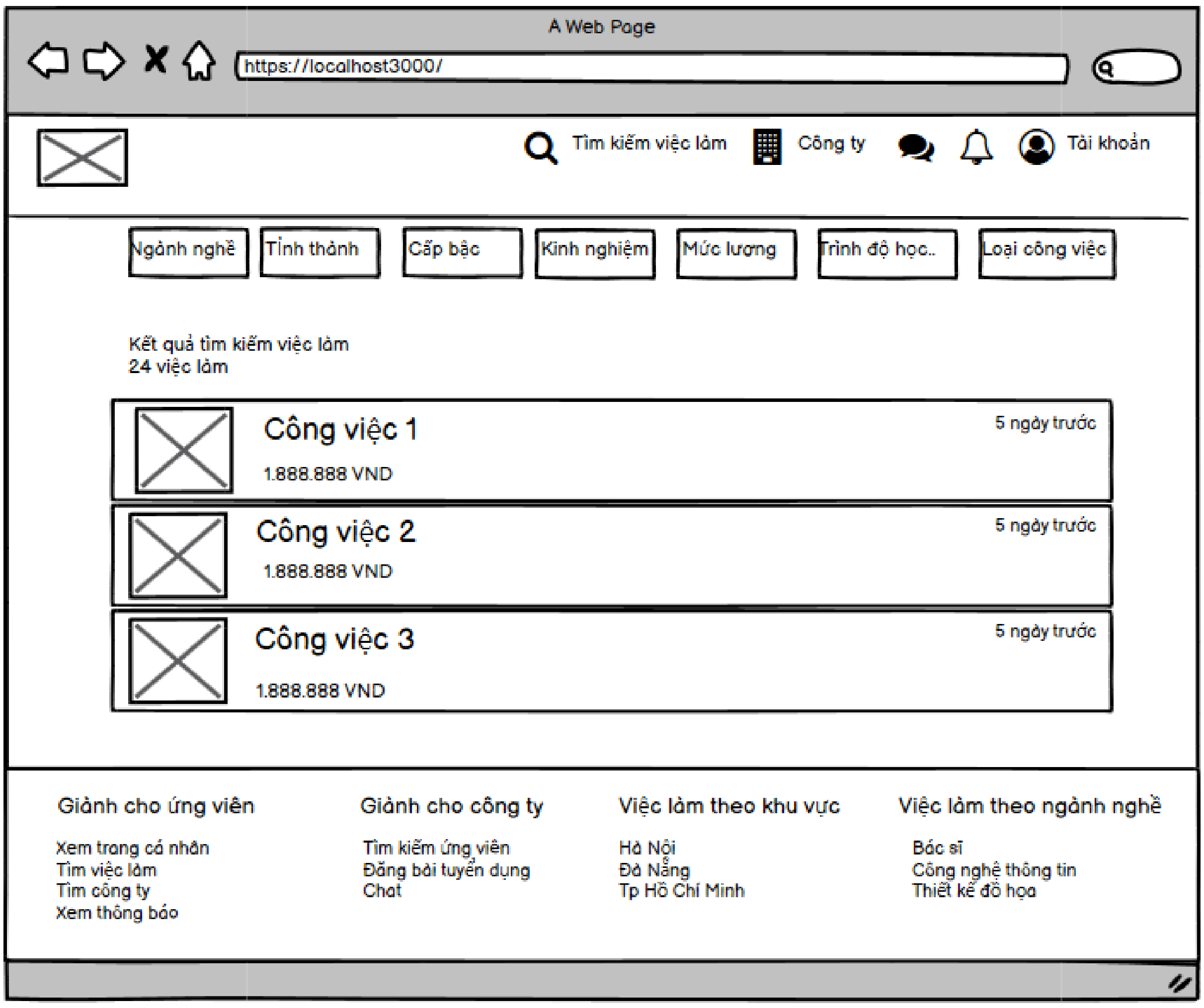
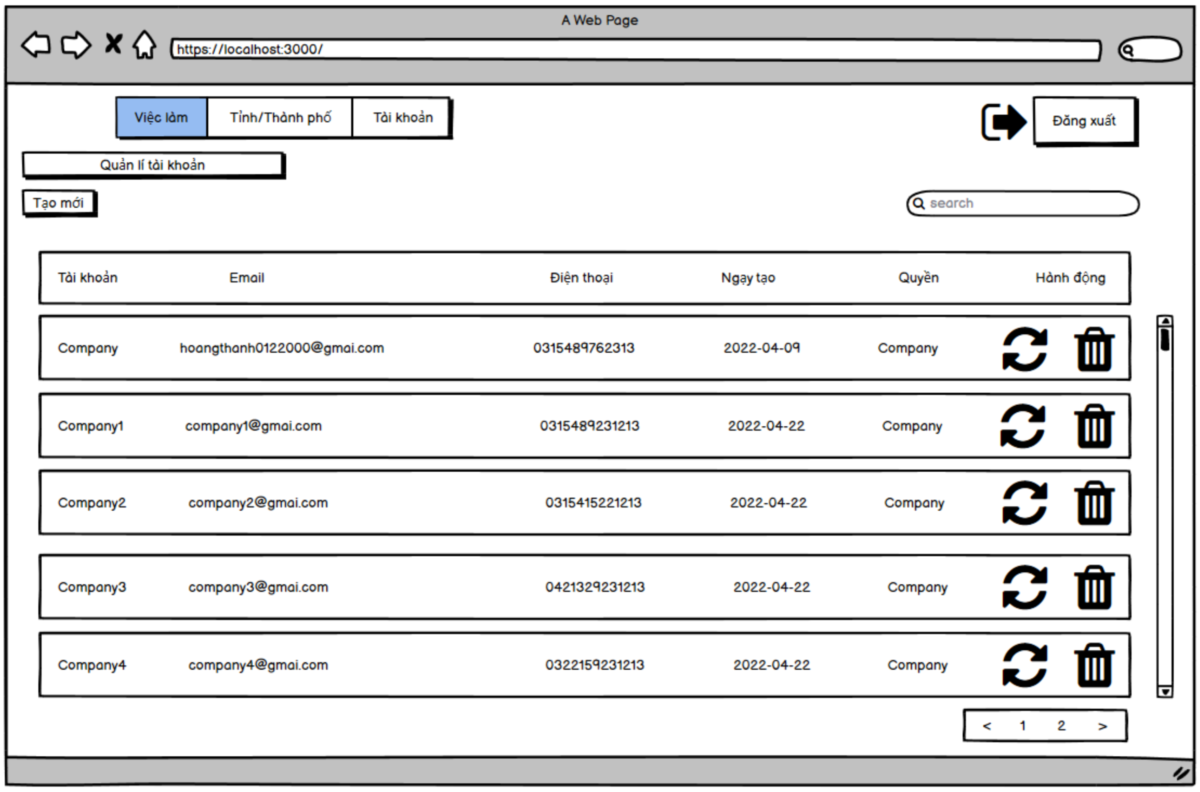


FIGURE 17: SEARCH RECRUITMENT


A Web Page

https://localhost:3000/

Việc làm | Tỉnh/Thành phố | Tài khoản

Đăng xuất

Quản lí tài khoản

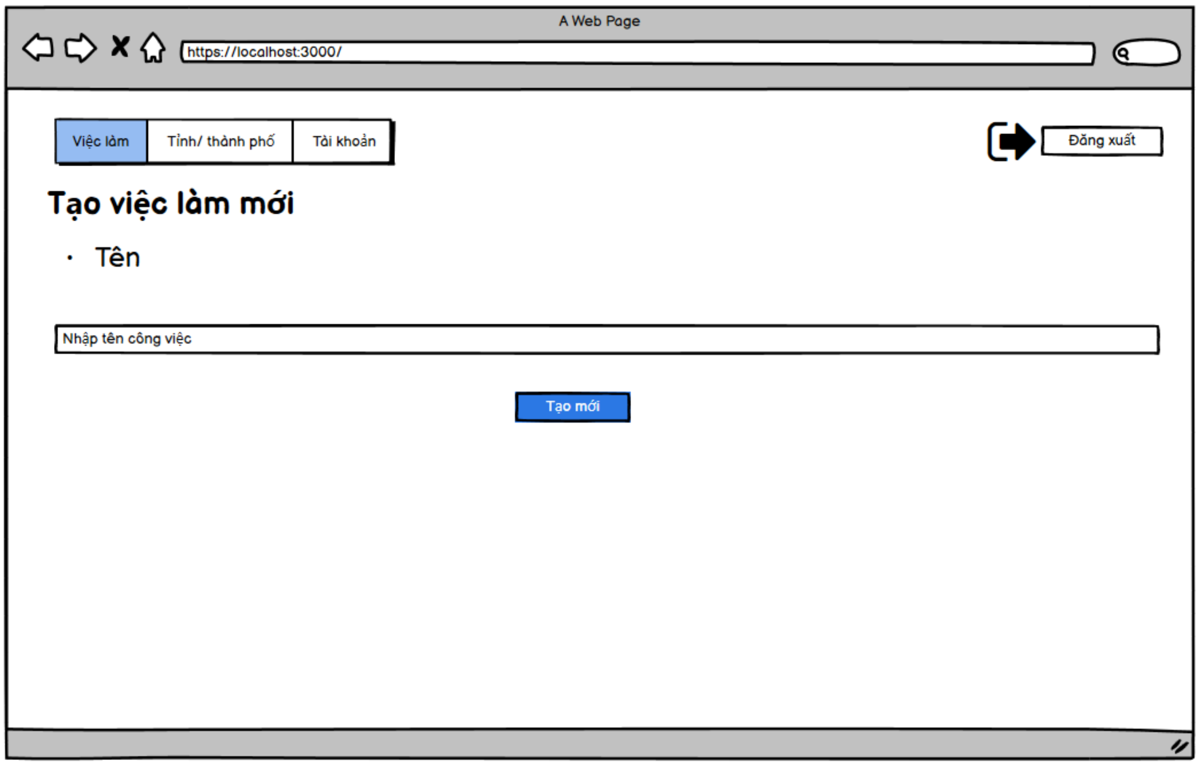
Tạo mới

search

Tài khoản	Email	Điện thoại	Ngày tạo	Quyền	Hành động
Company	hoangthanh0122000@gmai.com	0315489762313	2022-04-09	Company	
Company1	company1@gmai.com	0315489231213	2022-04-22	Company	
Company2	company2@gmai.com	0315415221213	2022-04-22	Company	
Company3	company3@gmai.com	0421329231213	2022-04-22	Company	
Company4	company4@gmai.com	0322159231213	2022-04-22	Company	

< 1 2 >

FIGURE 18: USER MANAGEMENT


A Web Page

https://localhost:3000/

Việc làm | Tỉnh/ thành phố | Tài khoản

Đăng xuất

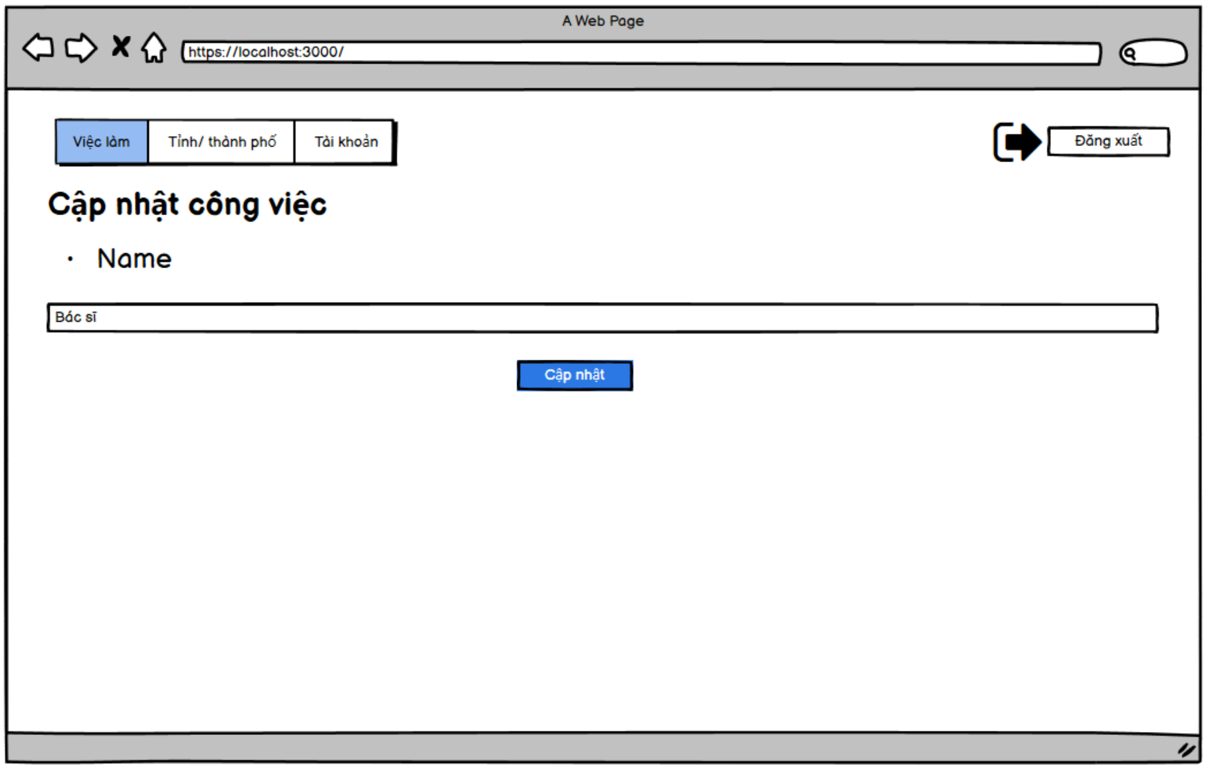
Tạo việc làm mới

- Tên

Nhập tên công việc

Tạo mới

FIGURE 19: CREATE A NEW RECRUITMENT


A Web Page

https://localhost:3000/

Việc làm | Tỉnh/ thành phố | Tài khoản

Đăng xuất

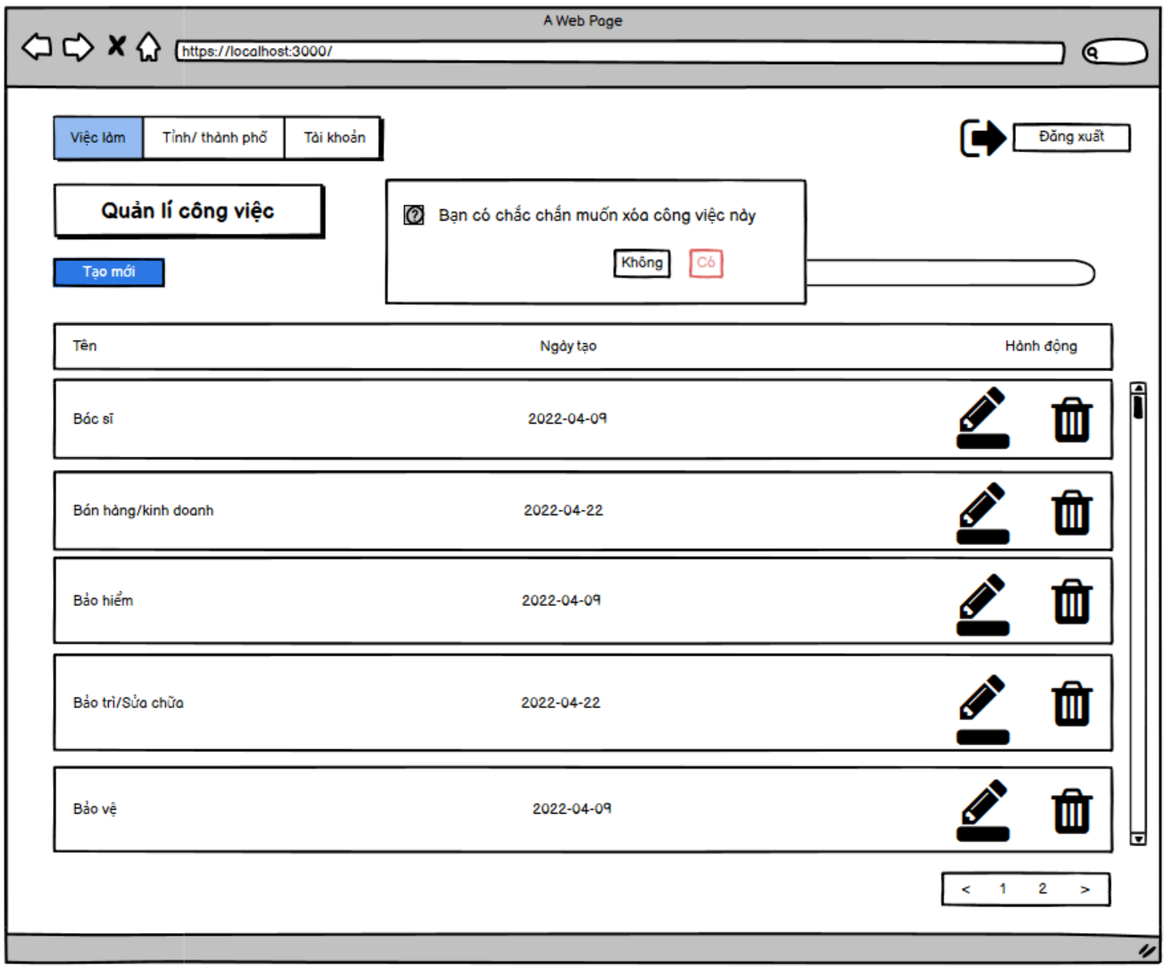
Cập nhật công việc

- Name

Bác sĩ

Cập nhật

FIGURE 20: UPDATE RECRUITMENT


A Web Page

https://localhost:3000/

Việc làm | Tỉnh/ thành phố | Tài khoản

Đăng xuất

Quản lí công việc

Tạo mới

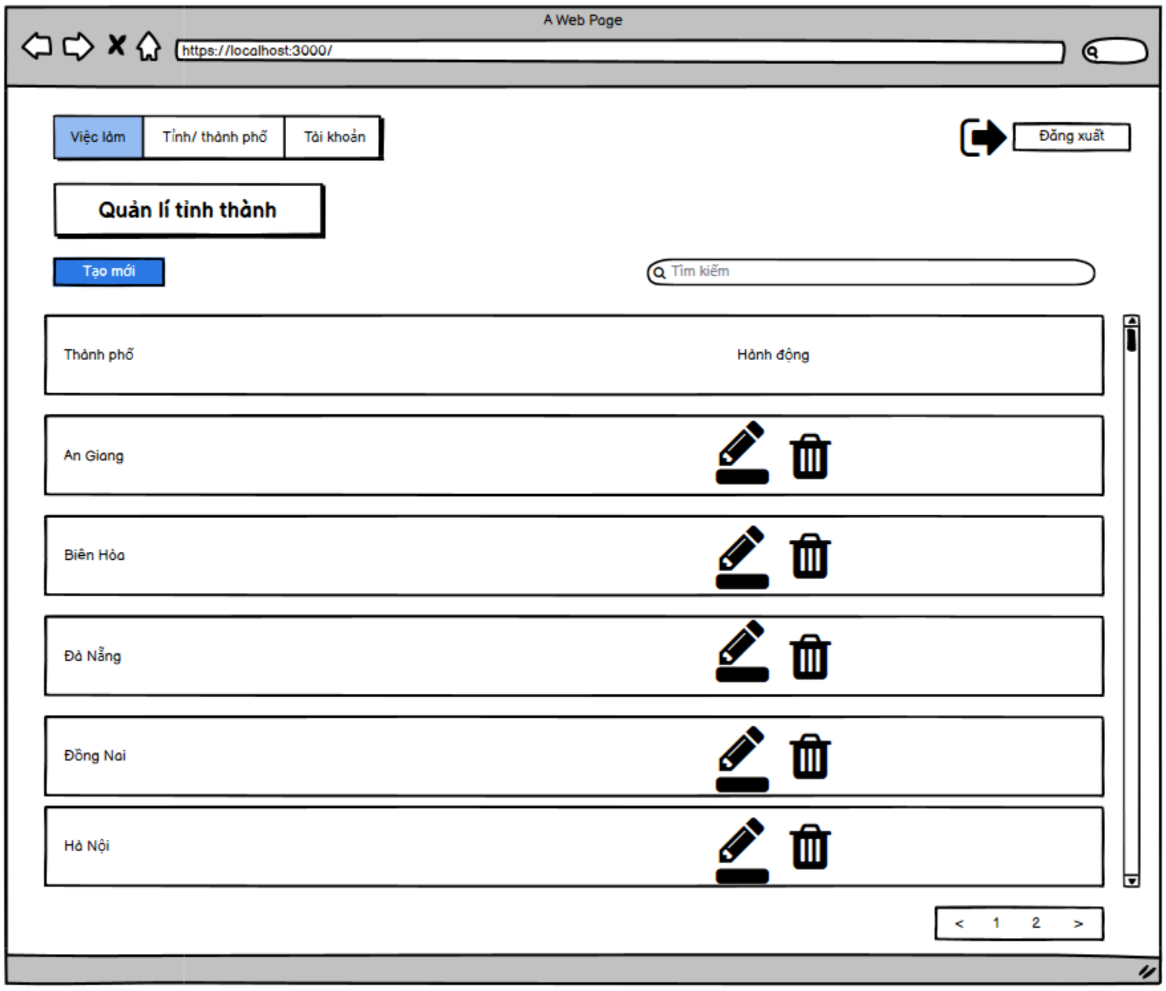
Bạn có chắc chắn muốn xóa công việc này

Không | Có

Tên	Ngày tạo	Hành động
Bác sĩ	2022-04-09	
Bán hàng/kinh doanh	2022-04-22	
Bảo hiểm	2022-04-09	
Bảo trì/Sửa chữa	2022-04-22	
Bảo vệ	2022-04-09	

< 1 2 >

FIGURE 21: DELETE RECRUITMENT



Thành phố	Hành động
An Giang	
Biên Hòa	
Đà Nẵng	
Đồng Nai	
Hà Nội	

FIGURE 22: BRANCH MANAGEMENT

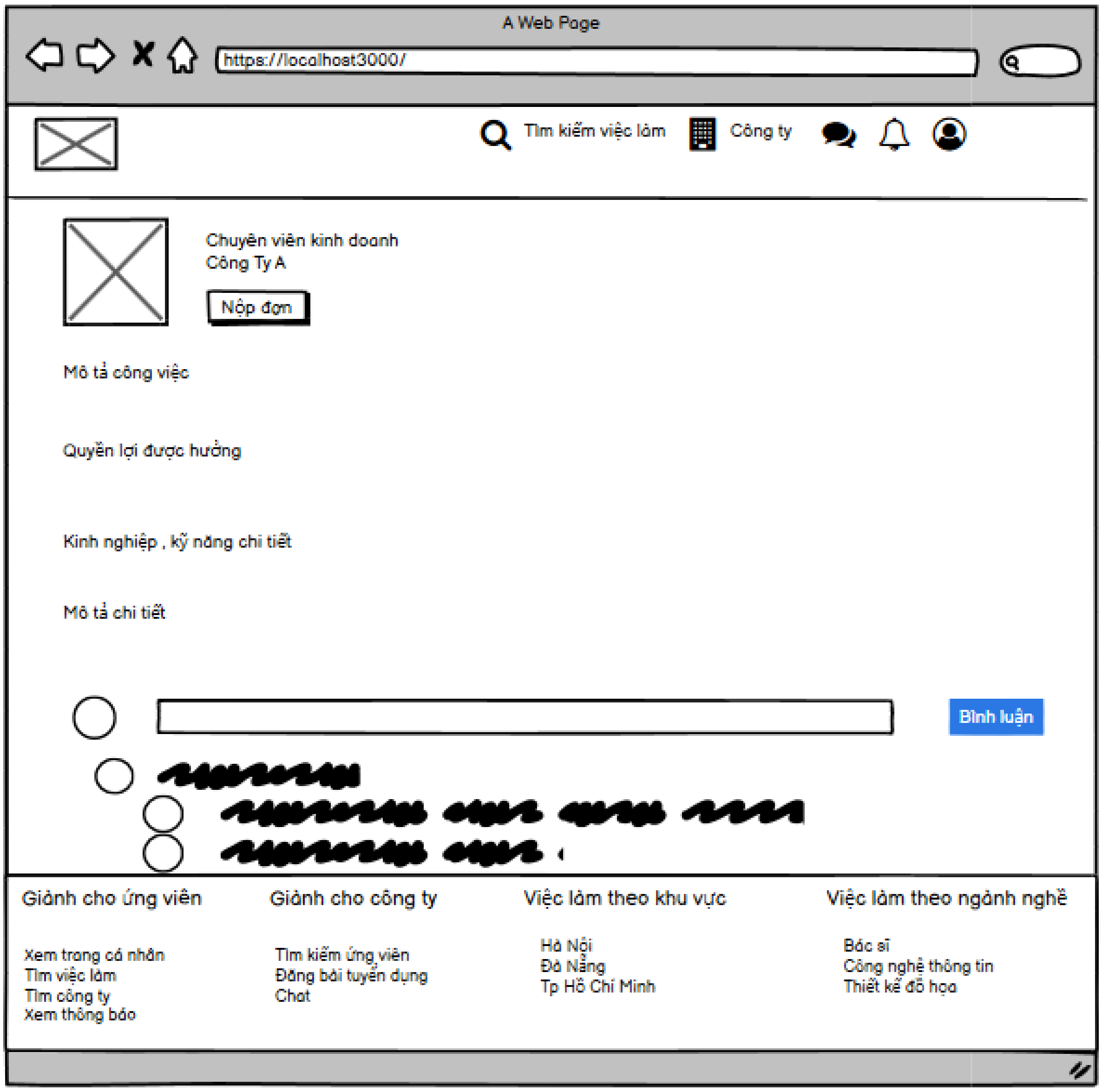


FIGURE 23: DETAIL RECRUITMENT

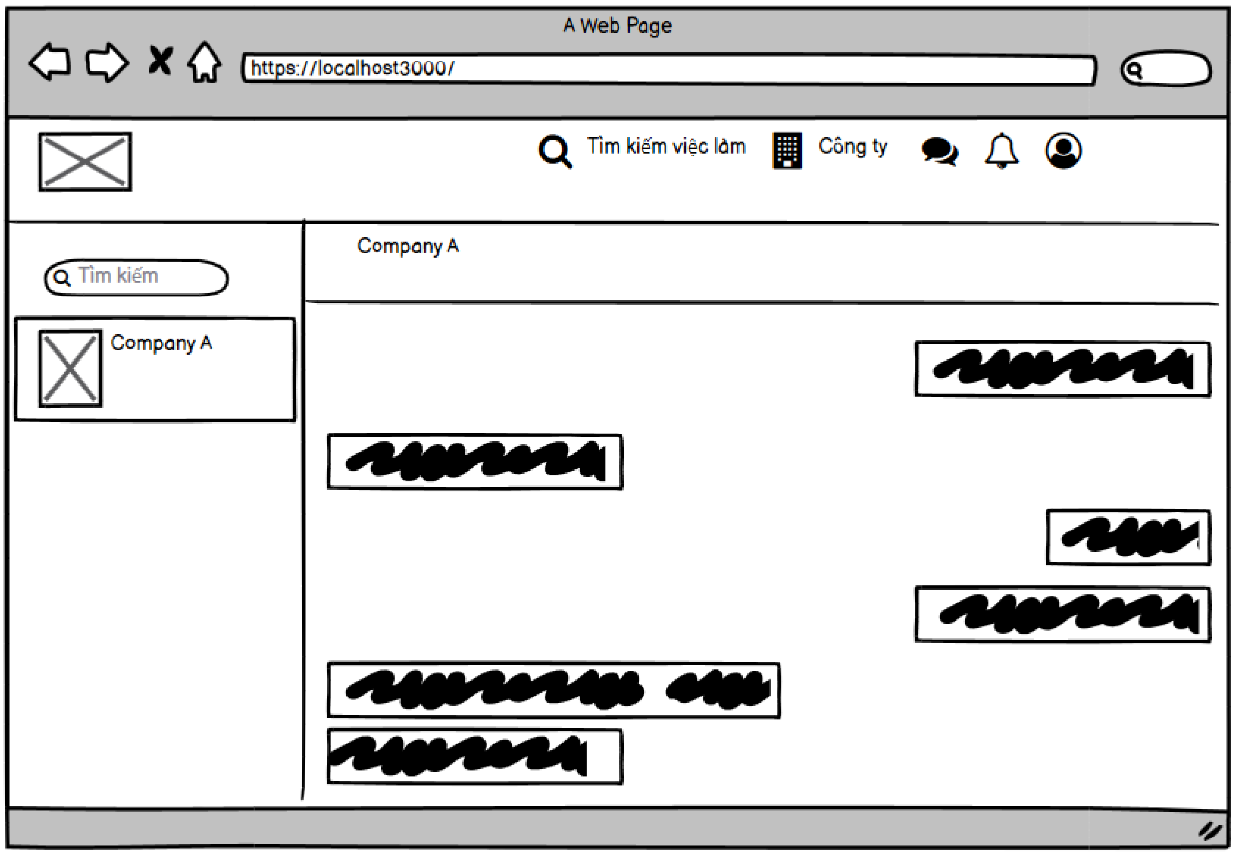


FIGURE 24: CHAT FUNCTION

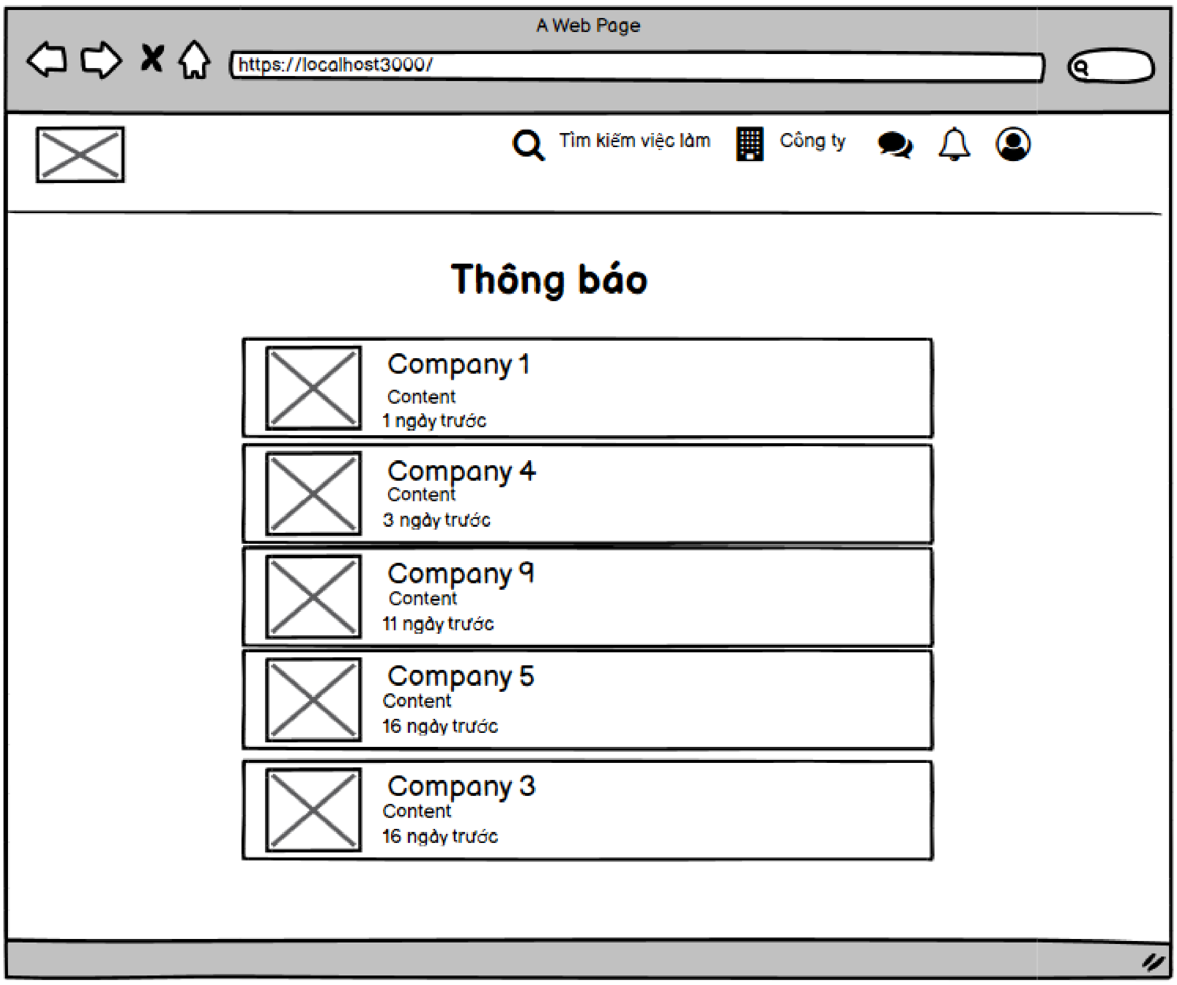


FIGURE 25: NOTIFICATION

8.7. Activity Diagram / Sequence Diagram

Activities for admin

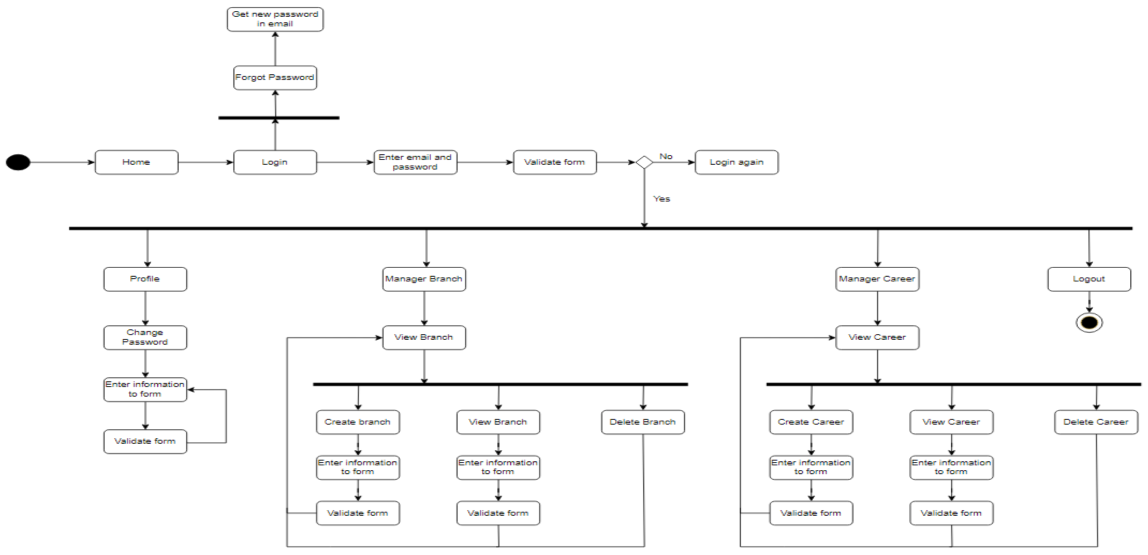


FIGURE 26: ACTIVITIES FOR ADMIN

Activities for User

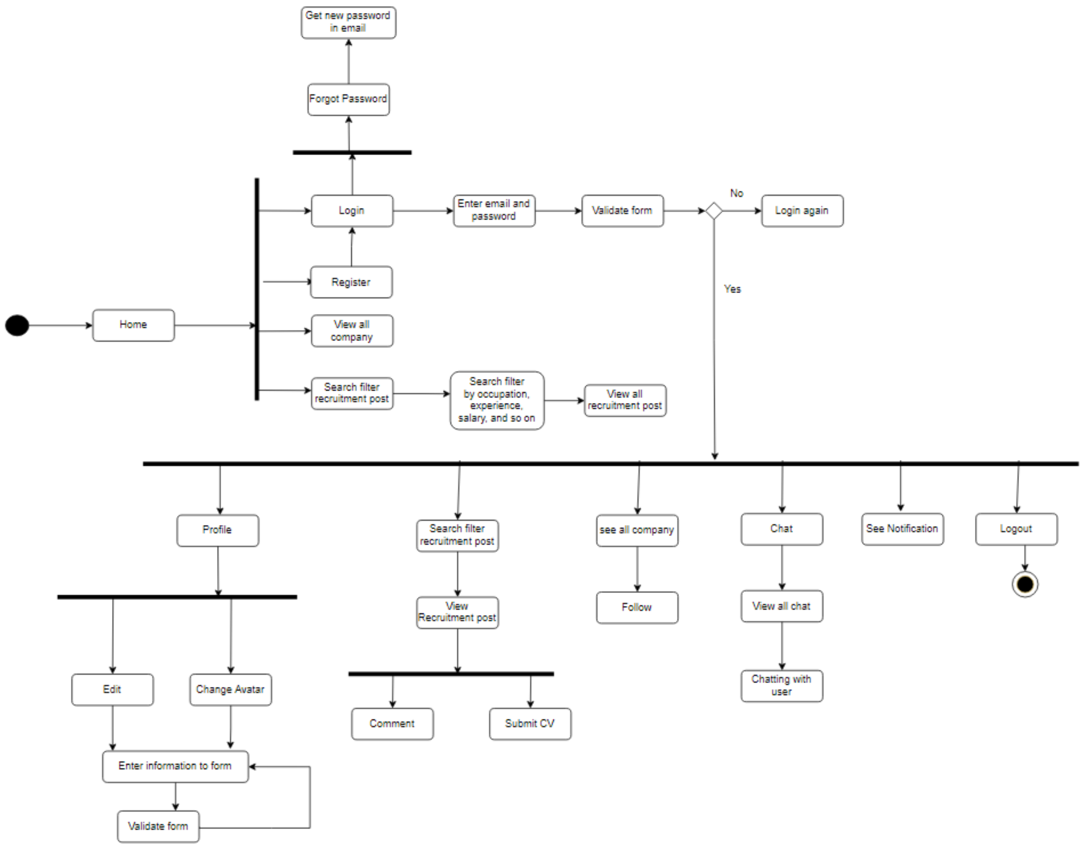


Figure 27: Activities for User

Activities for Company

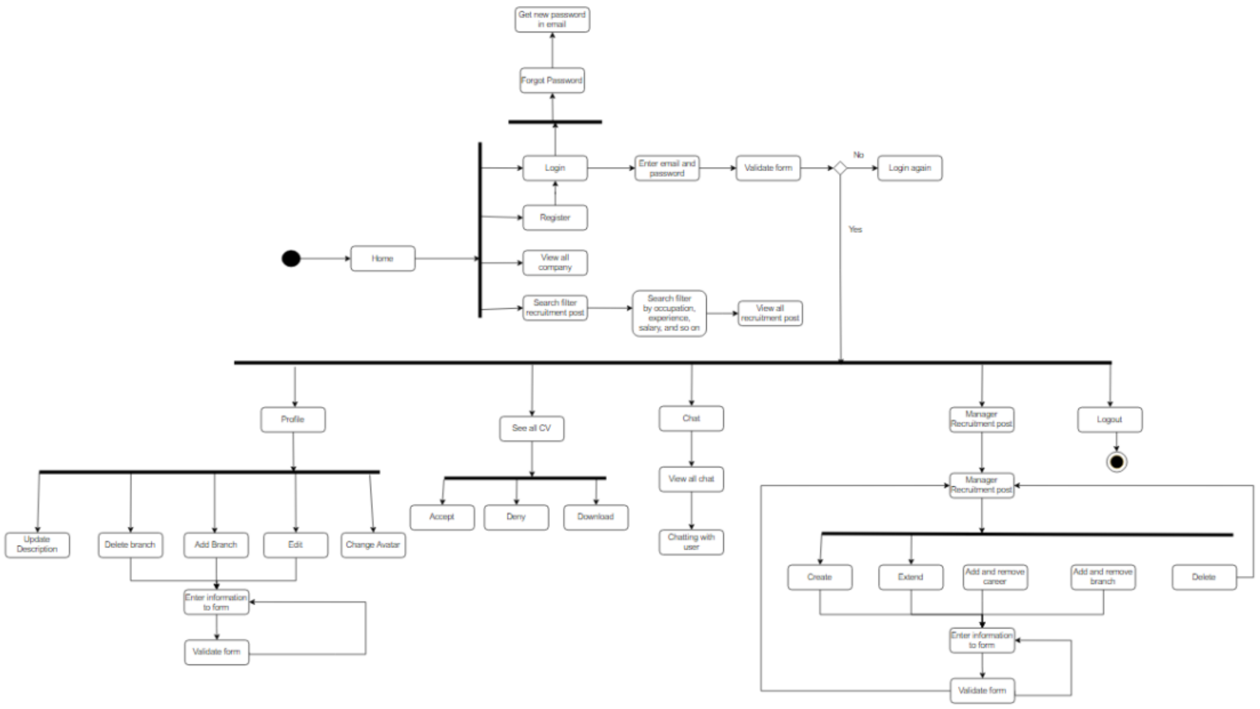
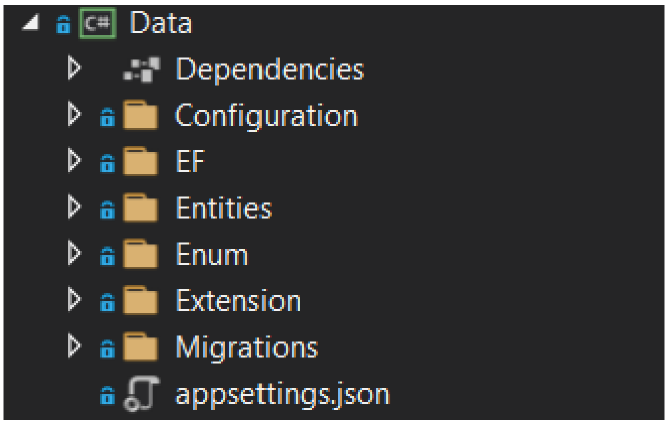


Figure 28: Activities for Company

9. Implementation

9.1. Database



```
Data
    Dependencies
    Configuration
    EF
    Entities
    Enum
    Extension
    Migrations
    appsettings.json
```

FIGURE 29: CONSTRUCTOR OF DATABASE DESIGN

In this project, I will first construct the database for the application using Entity Framework Code First described as shown above. The primary folders and their functions are shown below:

• Entities: This is the folder that contains the entities that are used to hold information from the database's tables.

• Configuration: This folder contains classes that use the Fluent API to configure the entities in the Entities folder

• Extension: This is the folder containing the code needed to make data available when migration database. This implies that when the database is built, the data is immediately available with no extra any impact.

• EF: This folder contains the DbContext class, which is used to query and store instances of entities

• Enum: This is the folder containing the enums

• Migrations: I used two major commands to establish a database: Add-Migration and Update-Database. The Migration folder is created once the I execute the first command. The I then execute the Update-Database command, which creates a new database based on the Migrations folder.

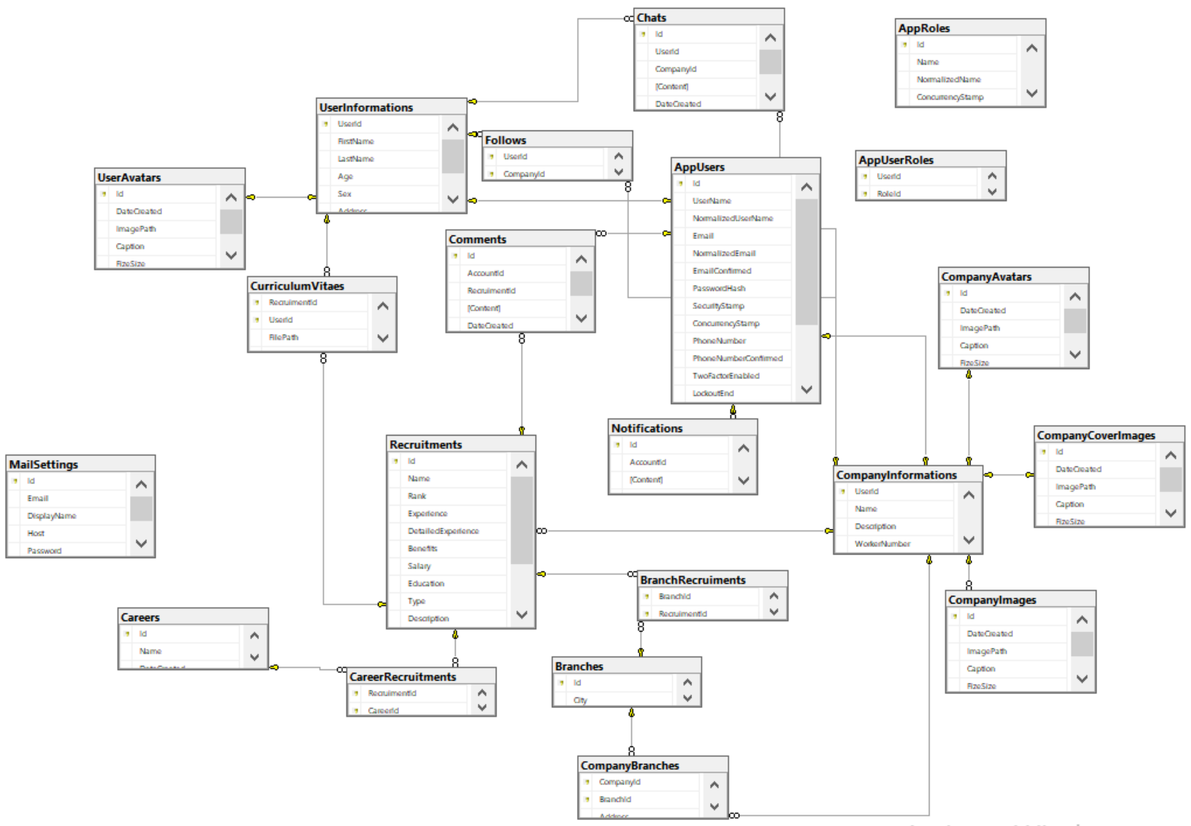


FIGURE 30: FINAL ERD

The image above illustrates the system's final ERD, and the below explains how it was implemented:

About the function

- AppUsers: Contains login information of users in the system such as: user name, password etc
- AppRoles: Contains information about system roles
- AppUserRoles: Determine which user belongs to which role
- UserInformations: Contains user details information
- UserAvatars: Contains avatar details information of user
- CompanyInformations: Contains company details information
- CompanyAvatars: Contains avatar details information of company
- CompanyCoverImages: Contains cover image details information of company
- CompanyImages: Contains images details information of company
- Follow: Determine which user followed which company
- Notifications: Store your account's notifications.
- Chat: Contains information about chat between user account and company account

- Recruitment: This table contains descriptions of recruitment postings
- Branch: Contains information about cities
- Career: Contains information about careers
- CompanyBranch: Determine the cities in which the company's branches are situated, as well as the particular addresses of those branches.
- BranchRecruitment: Identify the branches that need to be recruited by the company
- CareerRecruitment: Identify the careers that need to be recruited by the company
- Comments: Contains information about user or company account comments on the recruitment posting
- CurriculumVitaes: Contains information about the user's CV submitted to the company's recruitment posting
- MailSettings: Contains information of email used for the system's mail sending function.

About the Relationship

- The AppUserRoles table is a many-to-many relationship of the AppUsers and AppRoles table.
- The Chat and Follow table are a many-to-many relationship of the UserInformations and CompanyInformations table
- The Comment table is a many-to-many relationship of the AppUsers and Recruitments table
- The CurriculumVitaes table is a many-to-many relationship of the UserInformations and Recruitments table
- The CompanyBranches table is a many-to-many relationship of the CompanyInformations and Branches table
- The BranchesRecruitment table is a many-to-many relationship of the Recruitments and Branches table
- The CareersRecruitment table is a many-to-many relationship of the Recruitments and Careers table
- The Notifications table has a foreign key to the AppUsers table and they have a one-to-many relationship
- The UserAvatars table has a foreign key to the UserInformations table and they have a one-to-one relationship
- The CompanyAvatars and CompanyCoverImages table has a foreign key to the CompanyInformations table and they have a one-to-one relationship
- The CompanyImages table has a foreign key to the CompanyInformations table and they have a one-to-many relationship

9.2. Front End

9.2.1. Project Folder Structure

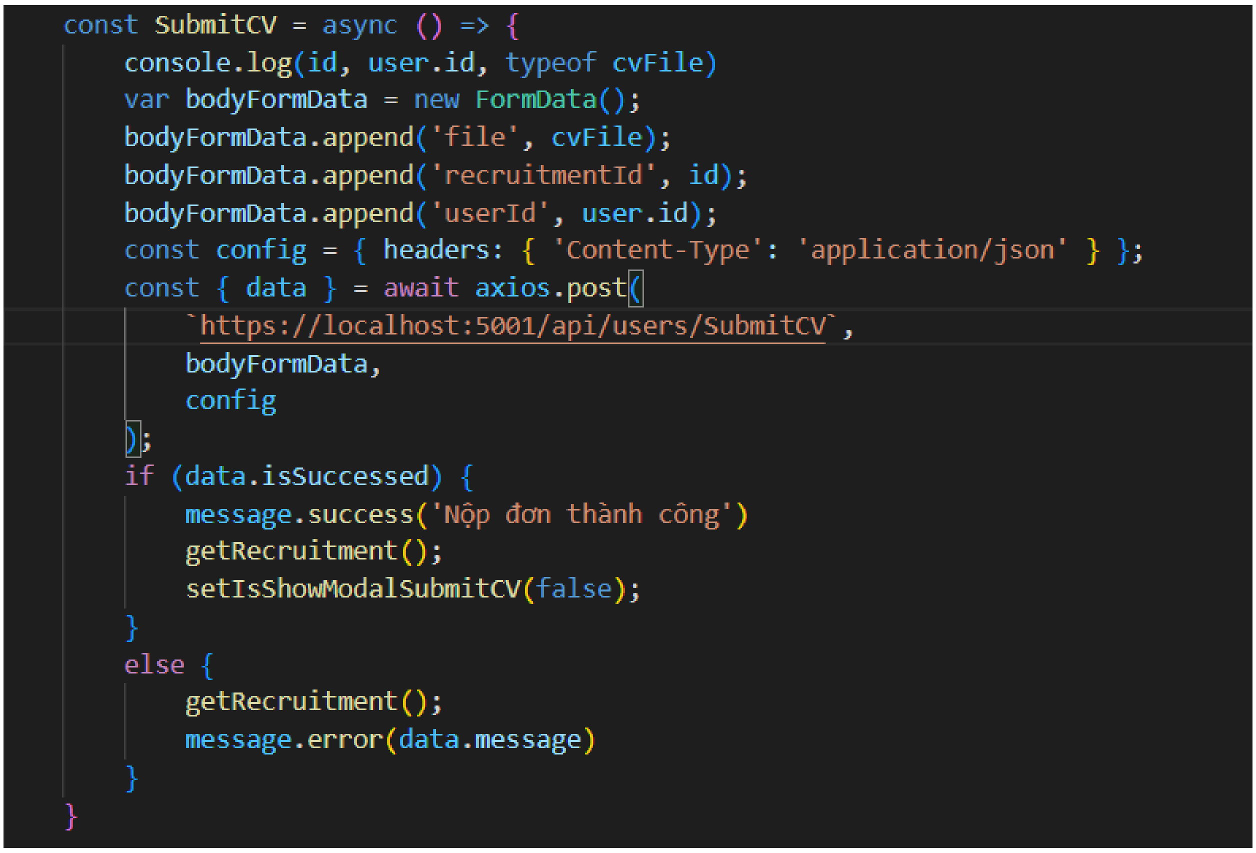
The diagram above displays the project's front-end directory structure. The first is the node module folder, which is used to store modules or libraries used in the project's development. Following that is the public folder, which contains static files such as index.html, javascript libraries, graphics, and so on that I do not want to handle with webpack. Next is the src folder, which is the main folder on the front-end side of the project. It includes the processing code that allows the system's pages to be displayed. It is divided into three main folders: assets folder, which stores pictures; layouts folder, which stores layouts shared by the whole system; pages folder, which includes the application's primary pages. Finally, the App.js file will include all of the material from the layouts and pages folders that will be displayed to the user.

9.2.2. Source code samples

Here are a few important and difficult-to-implement system features:

Submit CV

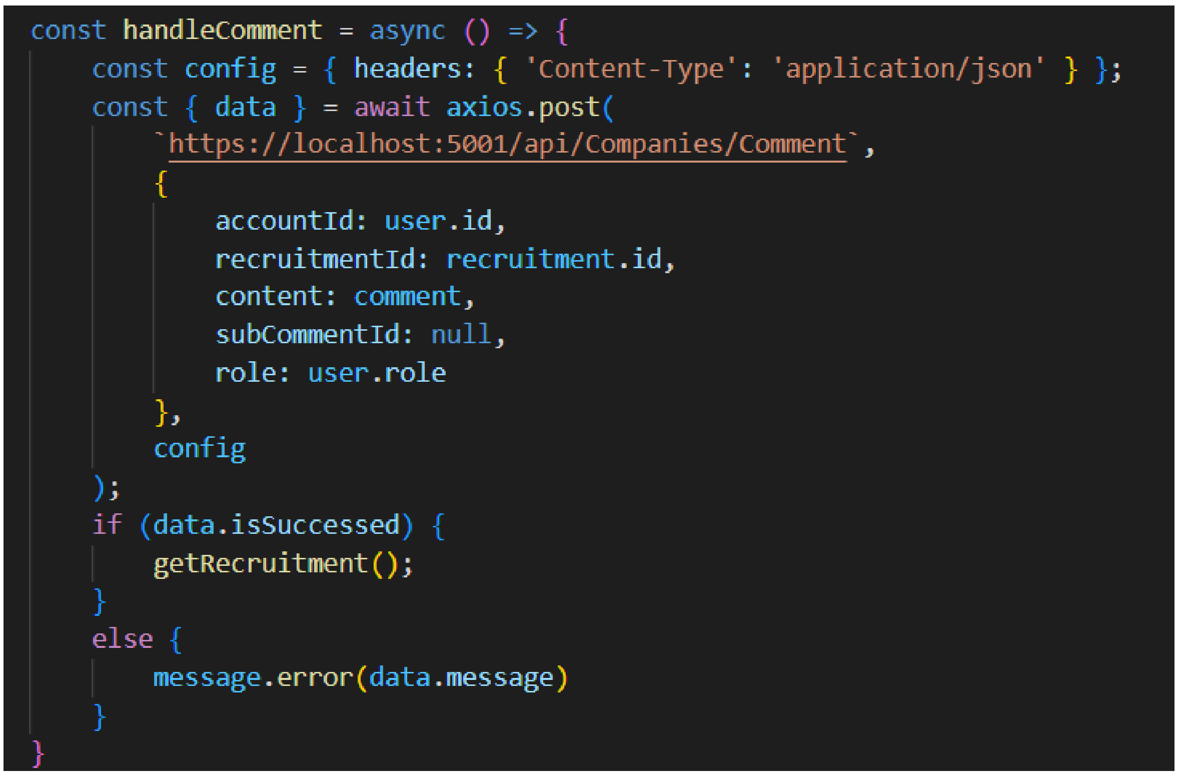
This is the feature of submitting CV to the user's job posting. The front end will receive the pdf file that the user has selected, and determine if the user's id and the selected job post is, then send the data to the backend. After receiving the results from the back end, it will check if it succeeds, it will show a success message, if not, it will display the error returned by the back end.



```
const SubmitCV = async () => {
    console.log(id, user.id, typeof cvFile)
    var bodyFormData = new FormData();
    bodyFormData.append('file', cvFile);
    bodyFormData.append('recruitmentId', id);
    bodyFormData.append('userId', user.id);
    const config = { headers: { 'Content-Type': 'application/json' } };
    const { data } = await axios.post(
        `https://localhost:5001/api/users/SubmitCV`,
        bodyFormData,
        config
    );
    if (data.isSuccessed) {
        message.success('Nộp đơn thành công')
        getRecruitment();
        setIsShowModalSubmitCV(false);
    }
    else {
        getRecruitment();
        message.error(data.message)
    }
}
```

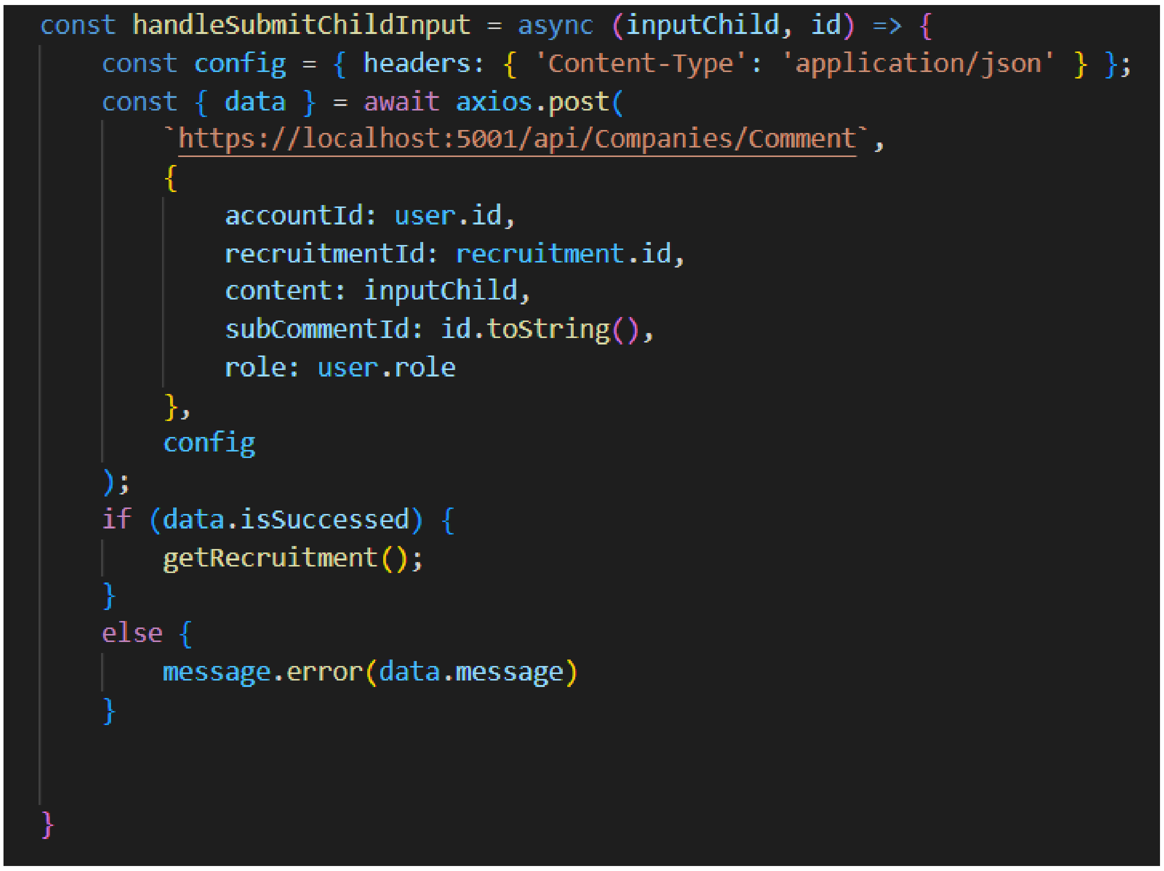
Comment

The comment function will receive the user's comment text and check to see whether the user id, job post id, and user role information are accessible before returning the data to the back end. If the results are successful, the page will be refreshed to display the new comment; otherwise, the error provided from the back end will be displayed.



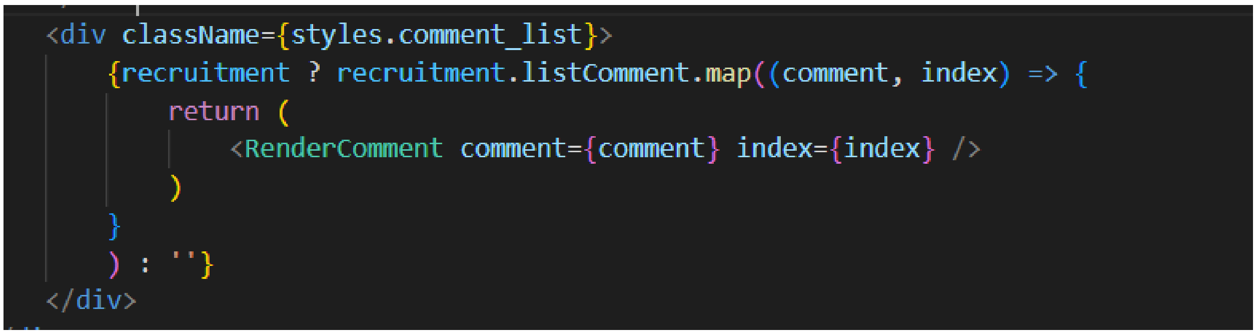
```
const handleComment = async () => {
    const config = { headers: { 'Content-Type': 'application/json' } };
    const { data } = await axios.post(
        `https://localhost:5001/api/Companies/Comment`,
        {
            accountId: user.id,
            recruitmentId: recruitment.id,
            content: comment,
            subCommentId: null,
            role: user.role
        },
        config
    );
    if (data.isSuccessed) {
        getRecruitment();
    }
    else {
        message.error(data.message)
    }
}
```

Besides the normal comment feature, employer or employee can also comment to reply to the old comments of the person who commented before. This feature is similar to the comment function but there is one other thing that it will receive an additional id comment parent.

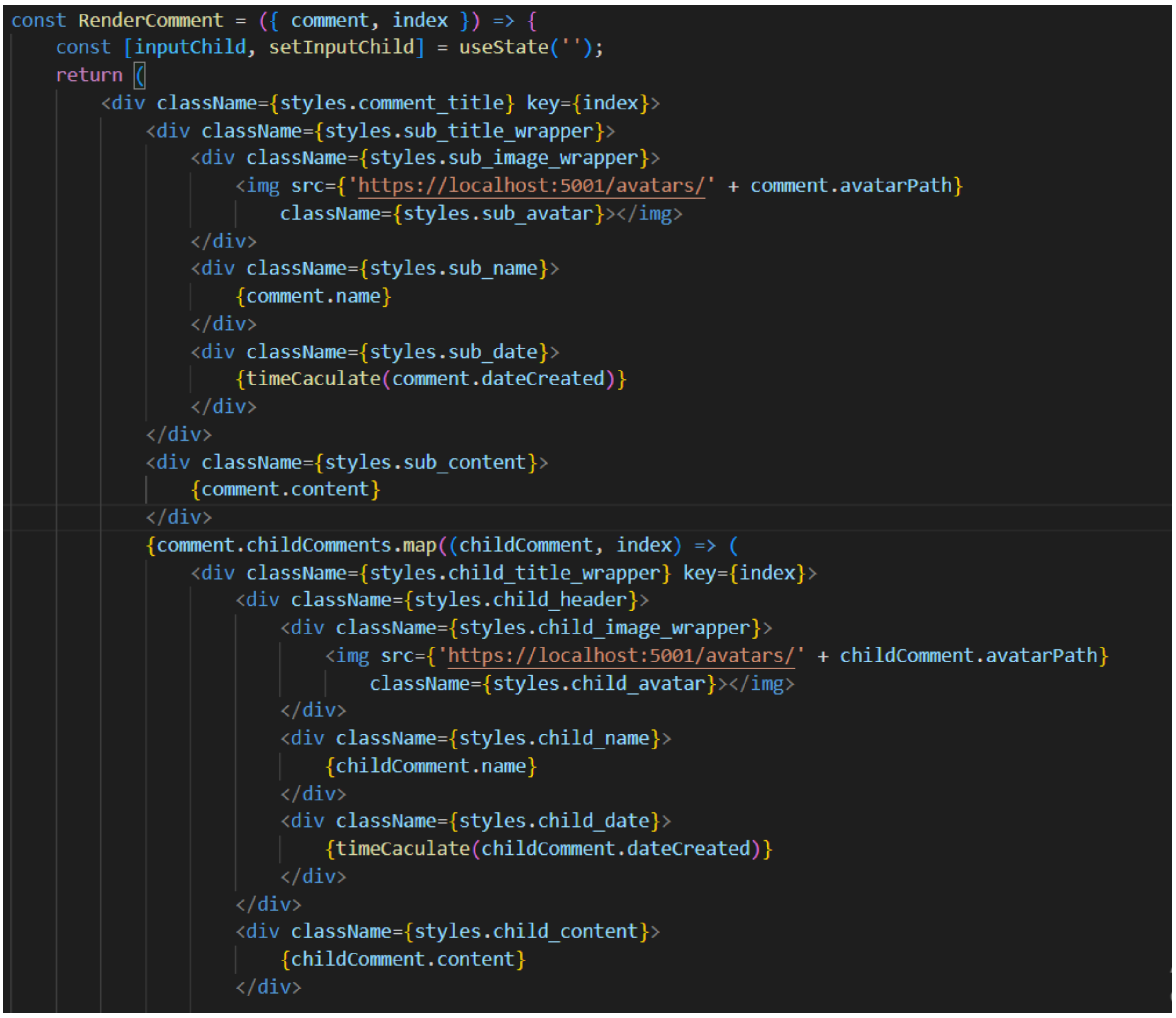


```
const handleSubmitChildInput = async (inputChild, id) => {
    const config = { headers: { 'Content-Type': 'application/json' } };
    const { data } = await axios.post(
        `https://localhost:5001/api/Companies/Comment`,
        {
            accountId: user.id,
            recruitmentId: recruitment.id,
            content: inputChild,
            subCommentId: id.toString(),
            role: user.role
        },
        config
    );
    if (data.isSuccessed) {
        getRecruitment();
    }
    else {
        message.error(data.message)
    }


}
```

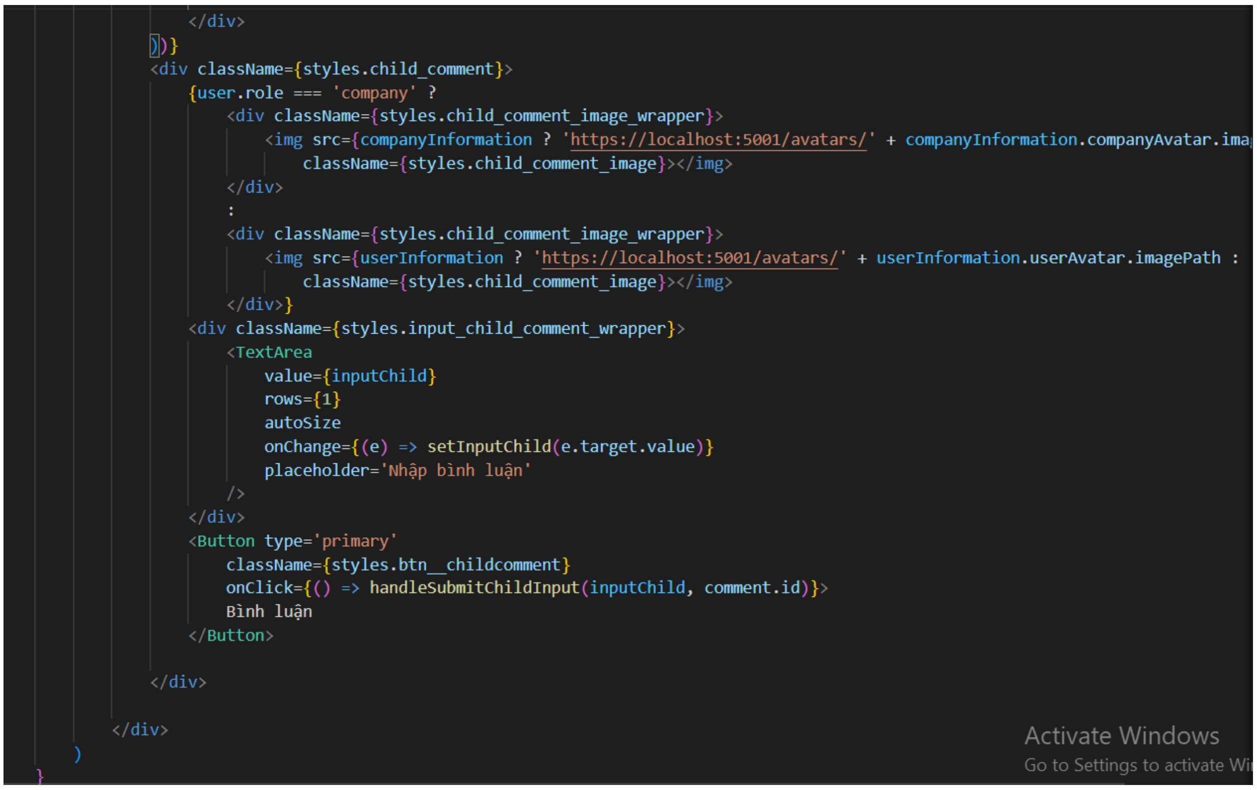
To be able to do these two features is also very cool that I will have to go through all the next large comments, then have to go through its child comments and display from there to be able to display all. both comments and at the same time determine the id of the parent comment.



```
<div className={styles.comment_list}>
    {recruitment ? recruitment.listComment.map((comment, index) => {
        return (
            <RenderComment comment={comment} index={index} />
        )
    }
    ) : ''}
</div>
```



```
const RenderComment = ({ comment, index }) => {
    const [inputChild, setInputChild] = useState('');
    return (
        <div className={styles.comment_title} key={index}>
            <div className={styles.sub_title_wrapper}>
                <div className={styles.sub_image_wrapper}>
                    <img src={'https://localhost:5001/avatars/' + comment.avatarPath}
                        className={styles.sub_avatar}></img>
                </div>
                <div className={styles.sub_name}>
                    {comment.name}
                </div>
                <div className={styles.sub_date}>
                    {timeCaculate(comment.dateCreated)}
                </div>
            </div>
            <div className={styles.sub_content}>
                {comment.content}
            </div>
            {comment.childComments.map((childComment, index) => (
                <div className={styles.child_title_wrapper} key={index}>
                    <div className={styles.child_header}>
                        <div className={styles.child_image_wrapper}>
                            <img src={'https://localhost:5001/avatars/' + childComment.avatarPath}
                                className={styles.child_avatar}></img>
                        </div>
                        <div className={styles.child_name}>
                            {childComment.name}
                        </div>
                        <div className={styles.child_date}>
                            {timeCaculate(childComment.dateCreated)}
                        </div>
                    </div>
                    <div className={styles.child_content}>
                        {childComment.content}
                    </div>
```

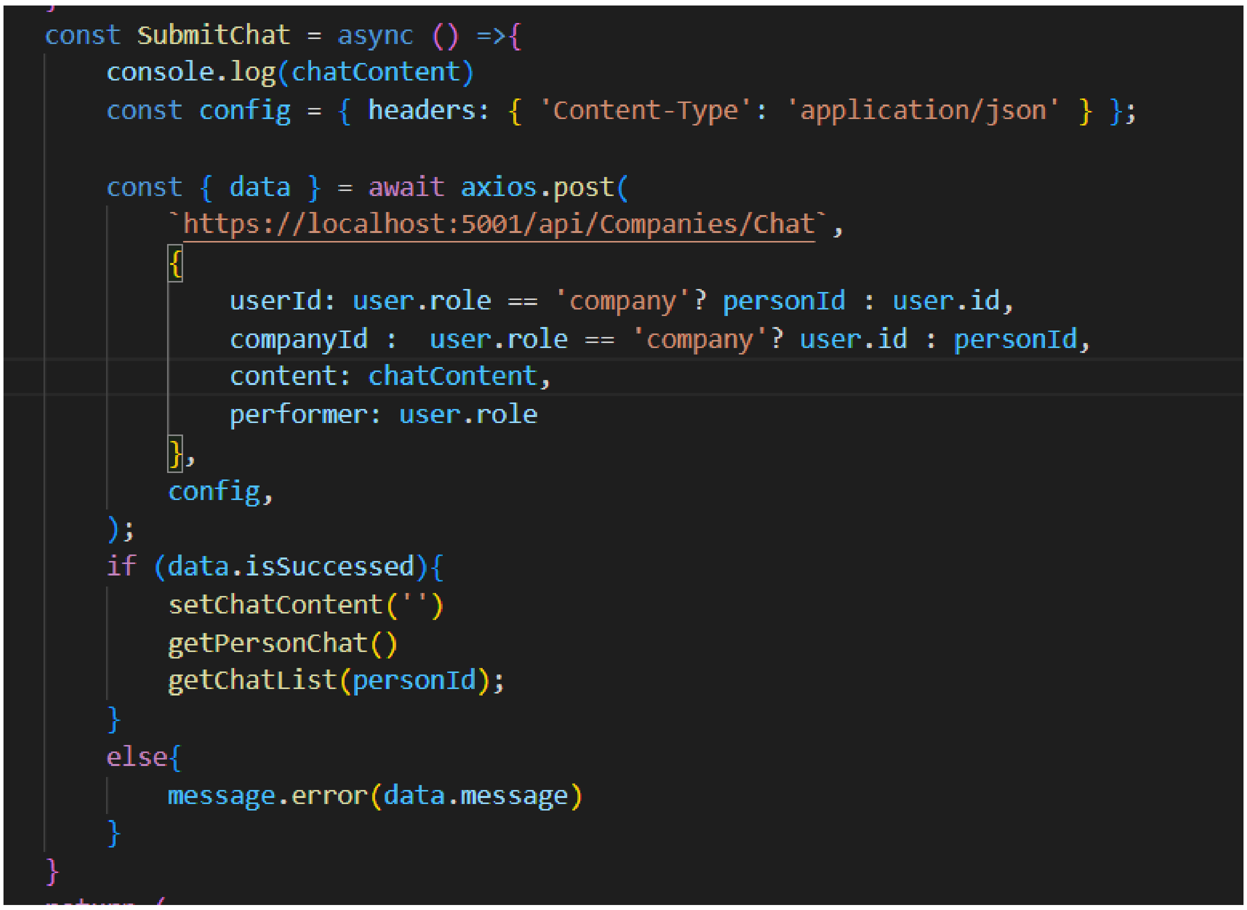


```
                </div>
            })}
            <div className={styles.child_comment}>
                {user.role === 'company' ?
                    <div className={styles.child_comment_image_wrapper}>
                        <img src={companyInformation ? 'https://localhost:5001/avatars/' + companyInformation.companyAvatar.ima
                            className={styles.child_comment_image}></img>
                    </div>
                    :
                    <div className={styles.child_comment_image_wrapper}>
                        <img src={userInformation ? 'https://localhost:5001/avatars/' + userInformation.userAvatar.imagePath :
                            className={styles.child_comment_image}></img>
                    </div>}
                <div className={styles.input_child_comment_wrapper}>
                    <TextArea
                        value={inputChild}
                        rows={1}
                        autoSize
                        onChange={(e) => setInputChild(e.target.value)}
                        placeholder='Nhập bình luận'
                    />
                </div>
                <Button type='primary'
                    className={styles.btn__childcomment}
                    onClick={() => handleSubmitChildInput(inputChild, comment.id)}>
                    Bình luận
                </Button>

            </div>

        </div>
    )
}
```

And this is the result:

Chat

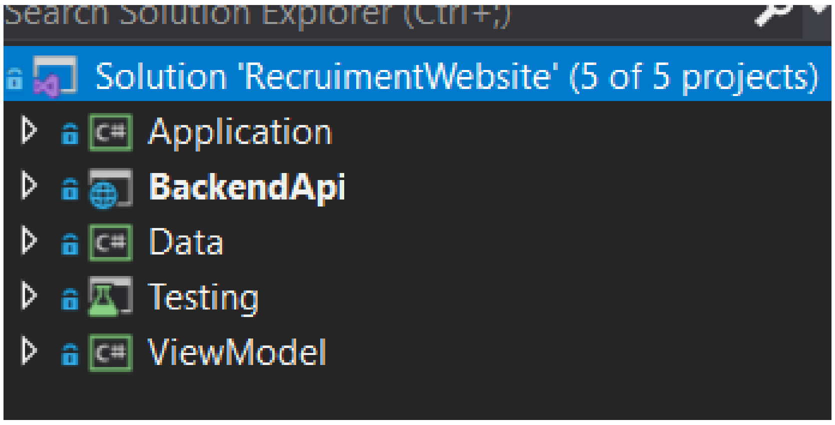
The chat feature is similar to the comment feature, when the user enters the content and presses send, the system will receive the user's data and determine more information such as the id of the two people chatting, the role of the real person. Show chat action and data back to the back end. After receiving the results, if successful, the page will be reloaded to display the chat content, otherwise, it will display the error returned by the back end.



```
const SubmitChat = async () =>{
    console.log(chatContent)
    const config = { headers: { 'Content-Type': 'application/json' } };

    const { data } = await axios.post(
        `https://localhost:5001/api/Companies/Chat`,
        {
            userId: user.role == 'company'? personId : user.id,
            companyId :  user.role == 'company'? user.id : personId,
            content: chatContent,
            performer: user.role
        },
        config,
    );
    if (data.isSuccessed){
        setChatContent('')
        getPersonChat()
        getChatList(personId);
    }
    else{
        message.error(data.message)
    }
}
```

9.3. Back End

9.3.1. Project Folder Structure



```
Solution 'RecruimentWebsite' (5 of 5 projects)
  Application
  BackendApi
  Data
  Testing
  ViewModel
```

The figure above displays the system's folder structure on the backend. As we can see, it consists of five sub-projects and specialized functions, which are as follows:

- **Data:** This part was introduced in section 9.1, and it provides code to assist with the creation and configuration of the system's database.
- **ViewModel:** This section comprises the ViewModels, which are primarily used to accept and return data to the client side in a straightforward and easy-to-understand manner.
- **Application:** This section contains the services used to query data from the database then process it and return it to the client.
- **Testing:** This section provides unit tests that are designed to determine whether or not the services written in the application are functioning properly.

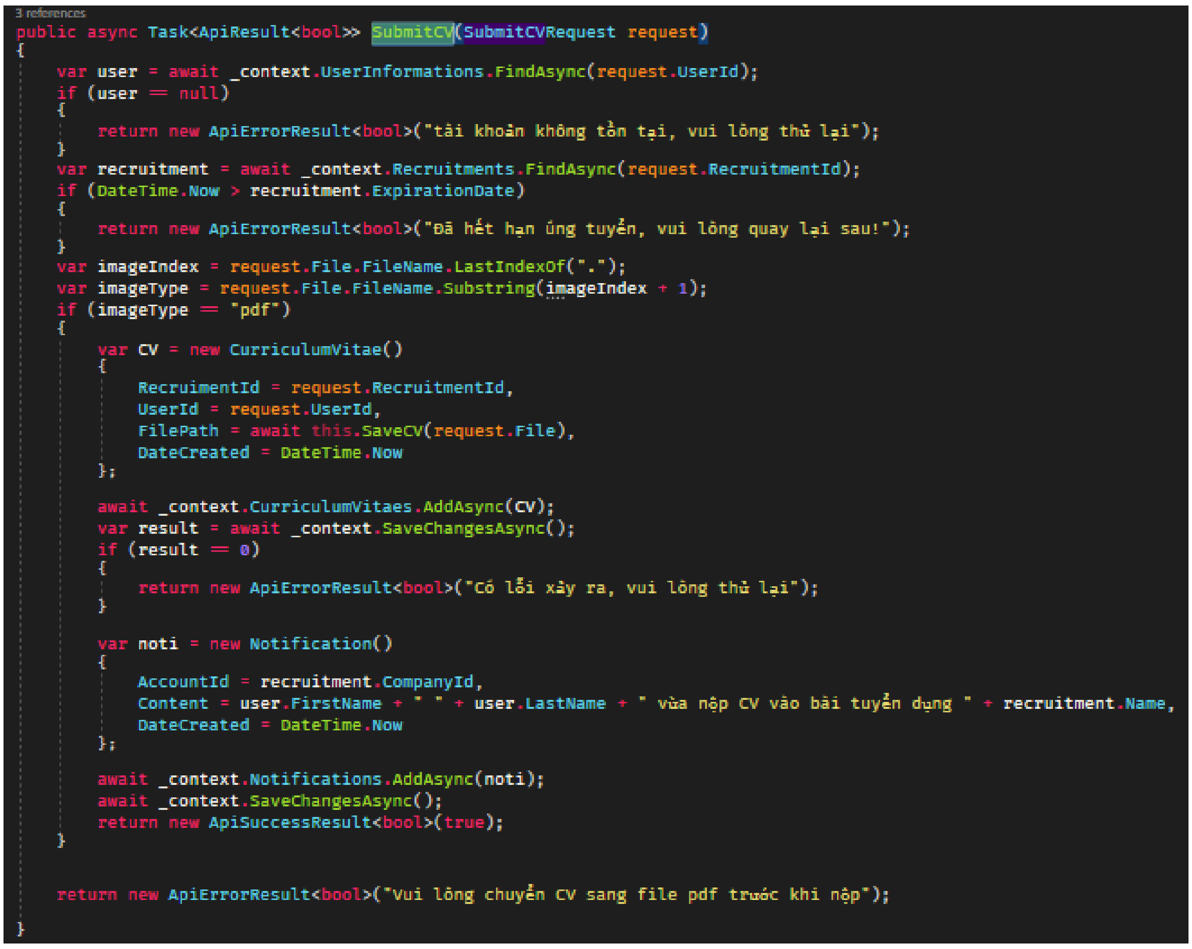
- **BackendApi:** This is the section where the Api is returned to the client. It will utilize the application's services and return the API to the client.

9.3.2. Source code samples

In this part, I will demonstrate the code that implements the system's back end functionalities as described in Section 9.2.2:

SubmitCV

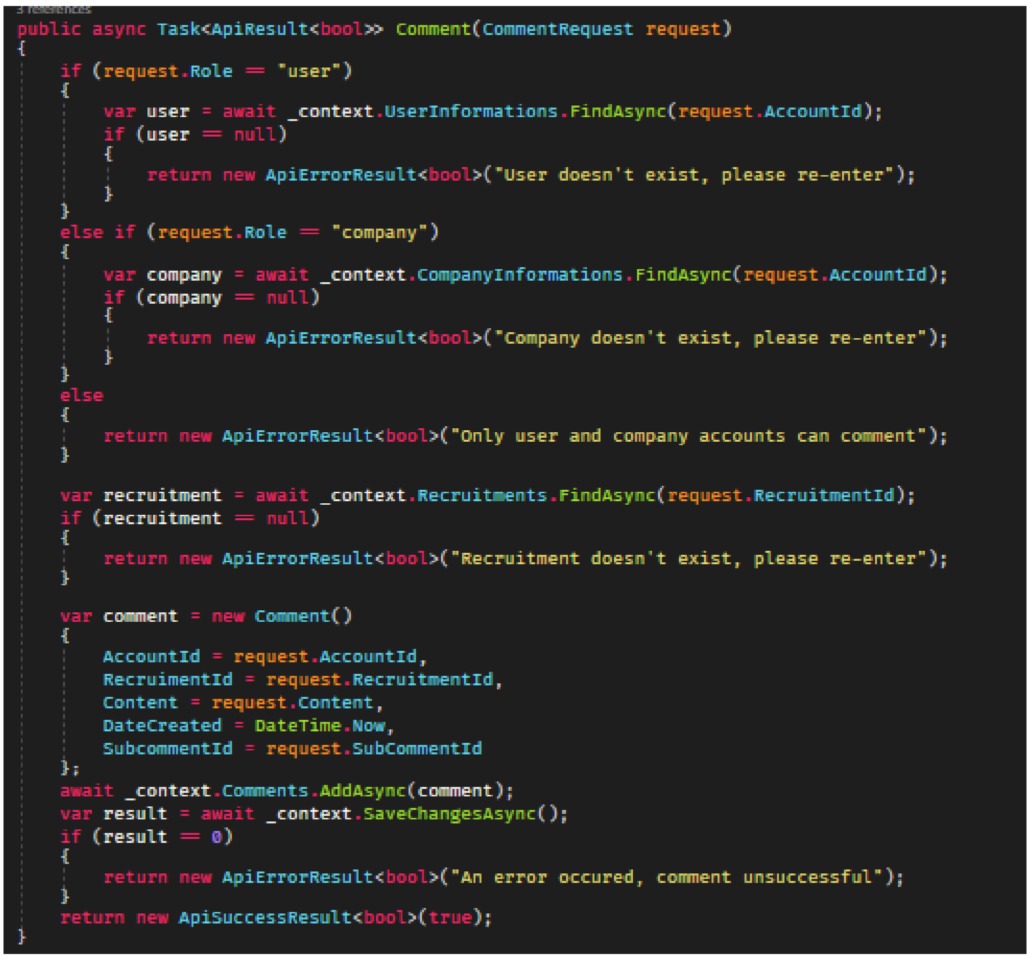
The system will first check if the person who submitted the CV exists or not, otherwise it will return an error to the client. Continue to check if the job posting has expired or not, if it has expired, it will also return an error to the client. If the above 2 checks are passed, the system will continue to check if the file the user chooses is a pdf or not, if not, then return it to the client asking the employee to convert to a pdf file before submitting. Conversely, if the file is in pdf format, it will proceed to create information to save the employee's CV and at the same time create a notice to the company that this candidate has submitted a CV for them.



```
3 references
public async Task<ApiResult<bool>> SubmitCV(SubmitCVRequest request)
{
    var user = await _context.UserInformations.FindAsync(request.UserId);
    if (user == null)
    {
        return new ApiErrorResult<bool>("tải khoản không tồn tại, vui lòng thử lại");
    }
    var recruitment = await _context.Recruitments.FindAsync(request.RecruitmentId);
    if (DateTime.Now > recruitment.ExpirationDate)
    {
        return new ApiErrorResult<bool>("Đã hết hạn ứng tuyển, vui lòng quay lại sau!");
    }
    var imageIndex = request.File.FileName.LastIndexOf(".");
    var imageType = request.File.FileName.Substring(imageIndex + 1);
    if (imageType == "pdf")
    {
        var CV = new CurriculumVitae()
        {
            RecruimentId = request.RecruitmentId,
            UserId = request.UserId,
            FilePath = await this.SaveCV(request.File),
            DateCreated = DateTime.Now
        };

        await _context.CurriculumVitaes.AddAsync(CV);
        var result = await _context.SaveChangesAsync();
        if (result == 0)
        {
            return new ApiErrorResult<bool>("Có lỗi xảy ra, vui lòng thử lại");
        }

        var noti = new Notification()
        {
            AccountId = recruitment.CompanyId,
            Content = user.FirstName + " " + user.LastName + " vừa nộp CV vào bài tuyển dụng " + recruitment.Name,
            DateCreated = DateTime.Now
        };

        await _context.Notifications.AddAsync(noti);
        await _context.SaveChangesAsync();
        return new ApiSuccessResult<bool>(true);
    }

    return new ApiErrorResult<bool>("Vui lòng chuyển CV sang file pdf trước khi nộp");
}
```

Comment

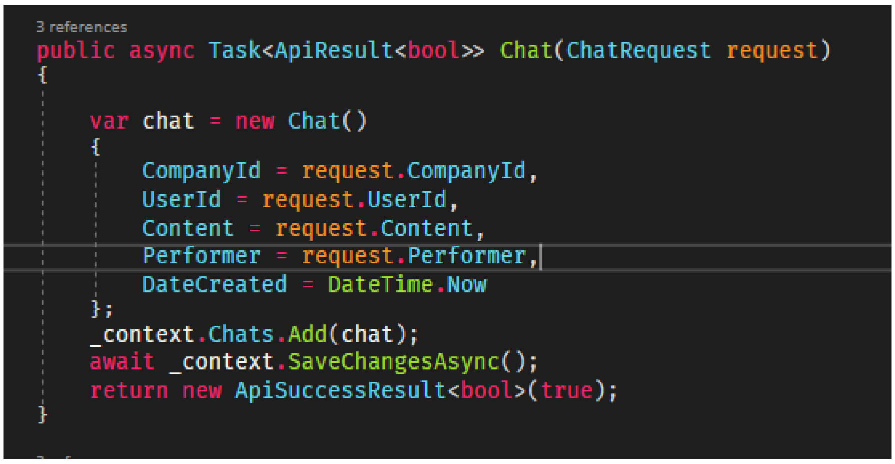
For the comment feature, after receiving data from the client, the system will check if that user or company or recruitment post exists or not, otherwise it will return an error to the client. Besides, the system also checks if the commenter is an employee or a company. Finally, if the conditions mentioned above are satisfied, the comment will be saved to the database and successfully returned to the client.



```
public async Task<ApiResult<bool>> Comment(CommentRequest request)
{
    if (request.Role == "user")
    {
        var user = await _context.UserInformations.FindAsync(request.AccountId);
        if (user == null)
        {
            return new ApiErrorResult<bool>("User doesn't exist, please re-enter");
        }
    }
    else if (request.Role == "company")
    {
        var company = await _context.CompanyInformations.FindAsync(request.AccountId);
        if (company == null)
        {
            return new ApiErrorResult<bool>("Company doesn't exist, please re-enter");
        }
    }
    else
    {
        return new ApiErrorResult<bool>("Only user and company accounts can comment");
    }

    var recruitment = await _context.Recruitments.FindAsync(request.RecruitmentId);
    if (recruitment == null)
    {
        return new ApiErrorResult<bool>("Recruitment doesn't exist, please re-enter");
    }

    var comment = new Comment()
    {
        AccountId = request.AccountId,
        RecruimentId = request.RecruitmentId,
        Content = request.Content,
        DateCreated = DateTime.Now,
        SubcommentId = request.SubCommentId
    };
    await _context.Comments.AddAsync(comment);
    var result = await _context.SaveChangesAsync();
    if (result == 0)
    {
        return new ApiErrorResult<bool>("An error occured, comment unsuccessful");
    }
    return new ApiSuccessResult<bool>(true);
}
```

Chat

This function is quite simple, after receiving data from the client, just save that information in the database and notify the client successfully.



```
3 references
public async Task<ApiResult<bool>> Chat(ChatRequest request)
{

    var chat = new Chat()
    {
        CompanyId = request.CompanyId,
        UserId = request.UserId,
        Content = request.Content,
        Performer = request.Performer,
        DateCreated = DateTime.Now
    };
    _context.Chats.Add(chat);
    await _context.SaveChangesAsync();
    return new ApiSuccessResult<bool>(true);
}
```

9.4. Images

Here is a picture of the main features of the application:

The first is the system's login function, the user must input the right account and password to log in, after which the system will check the user's role and direct them to the appropriate page.

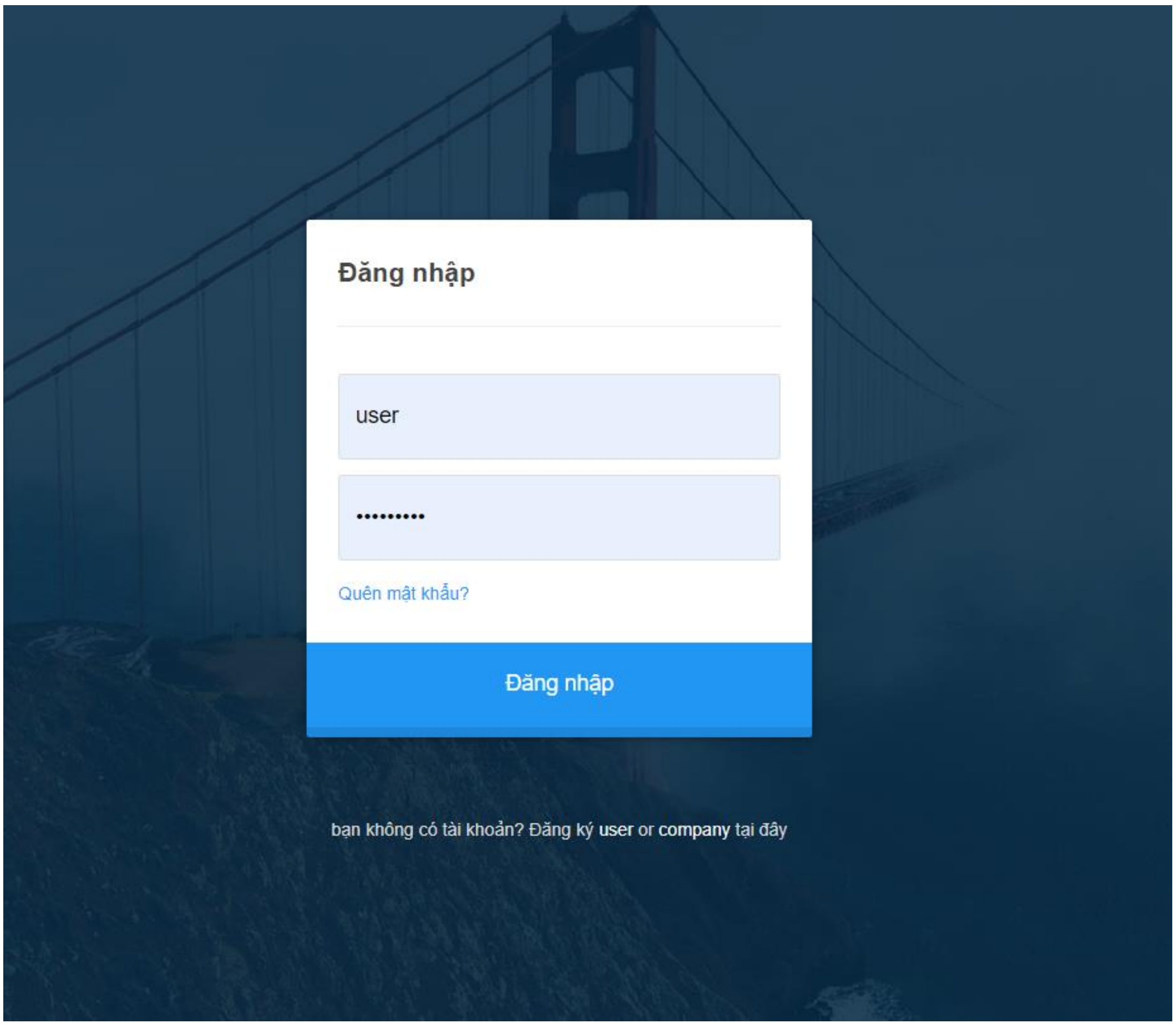
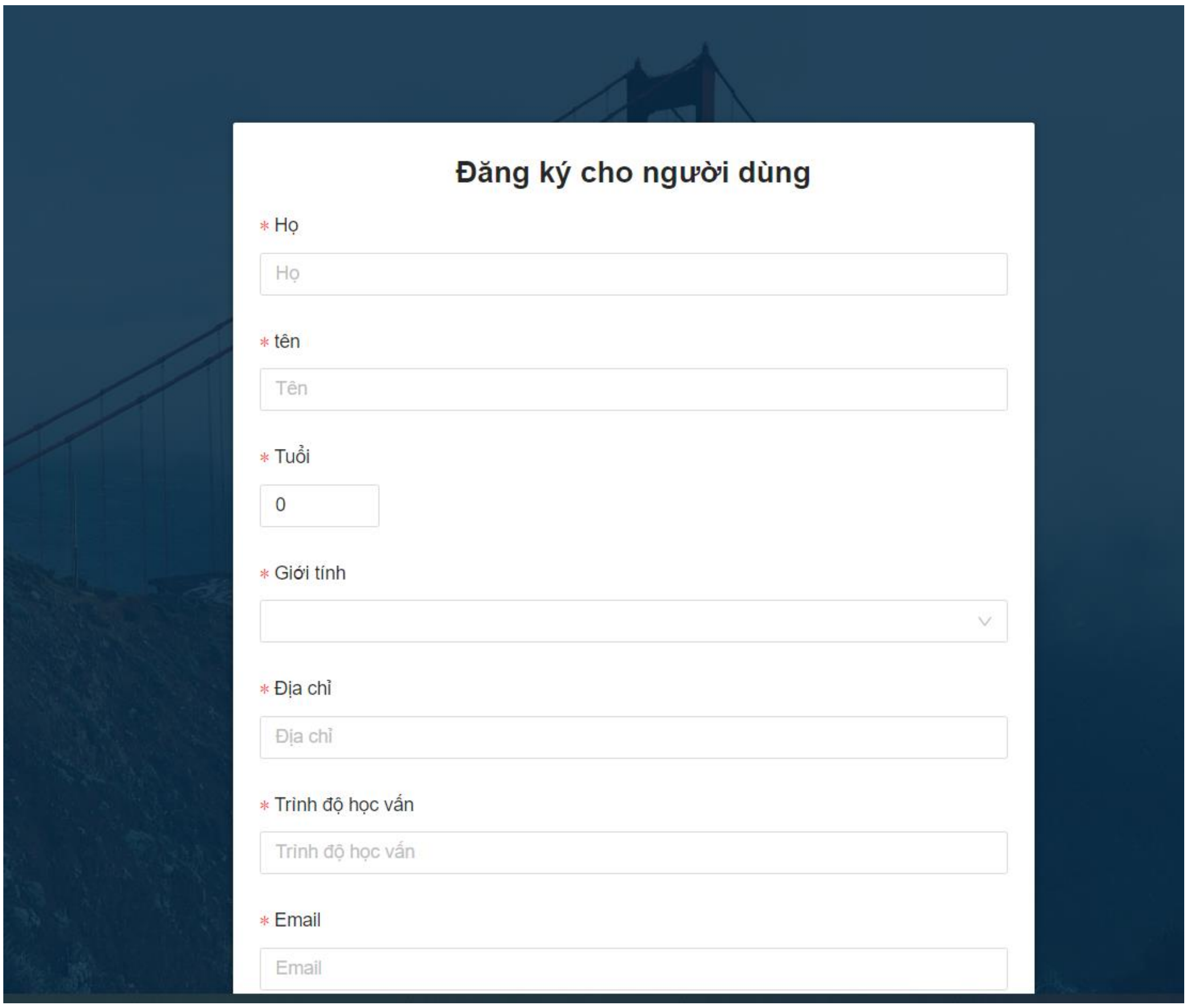


FIGURE 31: LOGIN PAGE

Next is the registration feature, depending on what type of account the user wants to create, choose the appropriate registration page. Then they only need to enter the correct information required to be able to log into the system.


Đăng ký cho người dùng

* Họ

Họ

* tên

Tên

* Tuổi

0

* Giới tính

* Địa chỉ

Địa chỉ

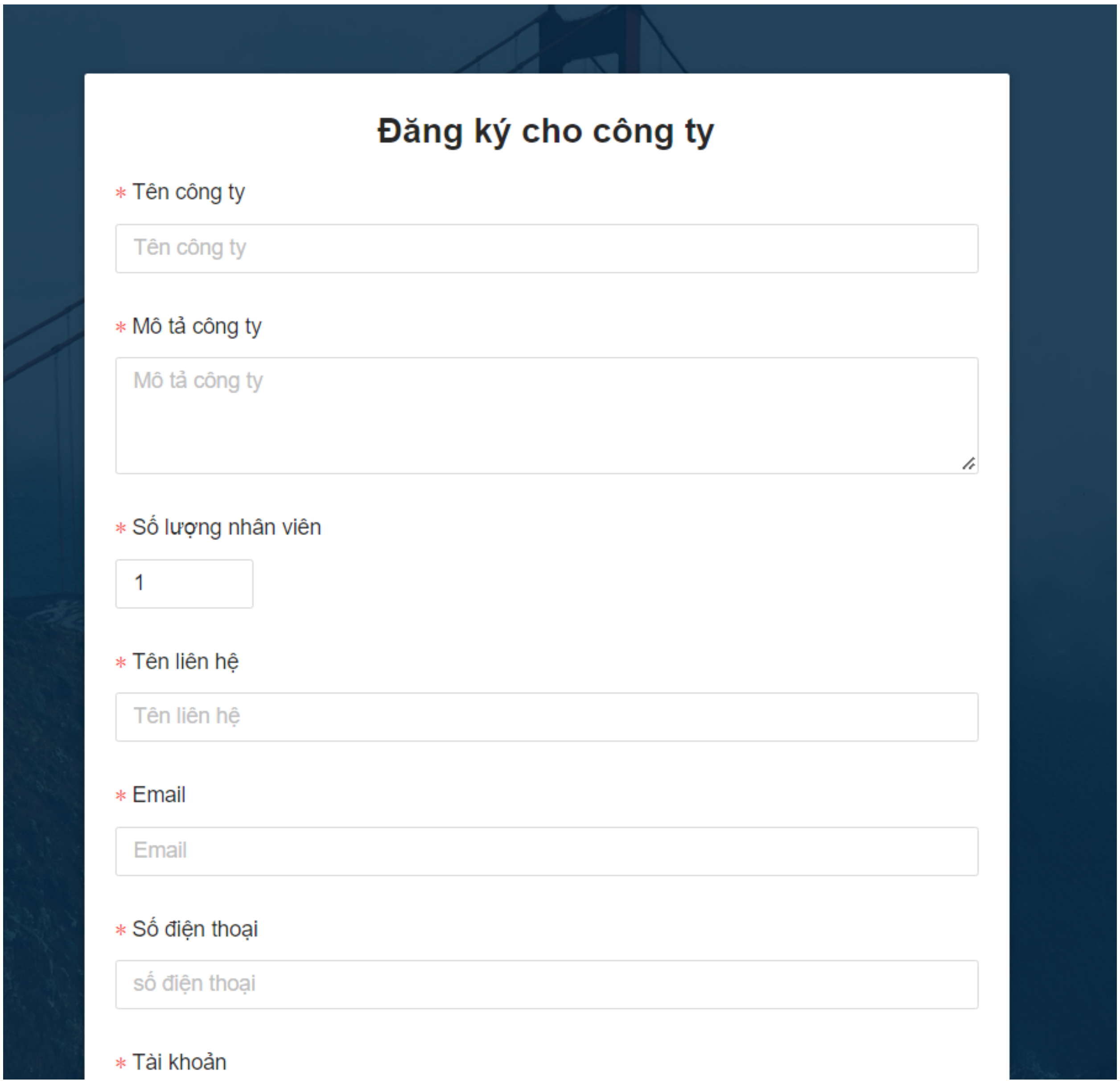
* Trinh độ học vấn

Trinh độ học vấn

* Email

Email

FIGURE 32: REGISTER FOR USER PAGE


Đăng ký cho công ty

* Tên công ty

Tên công ty

* Mô tả công ty

Mô tả công ty

* Số lượng nhân viên

1

* Tên liên hệ

Tên liên hệ

* Email

Email

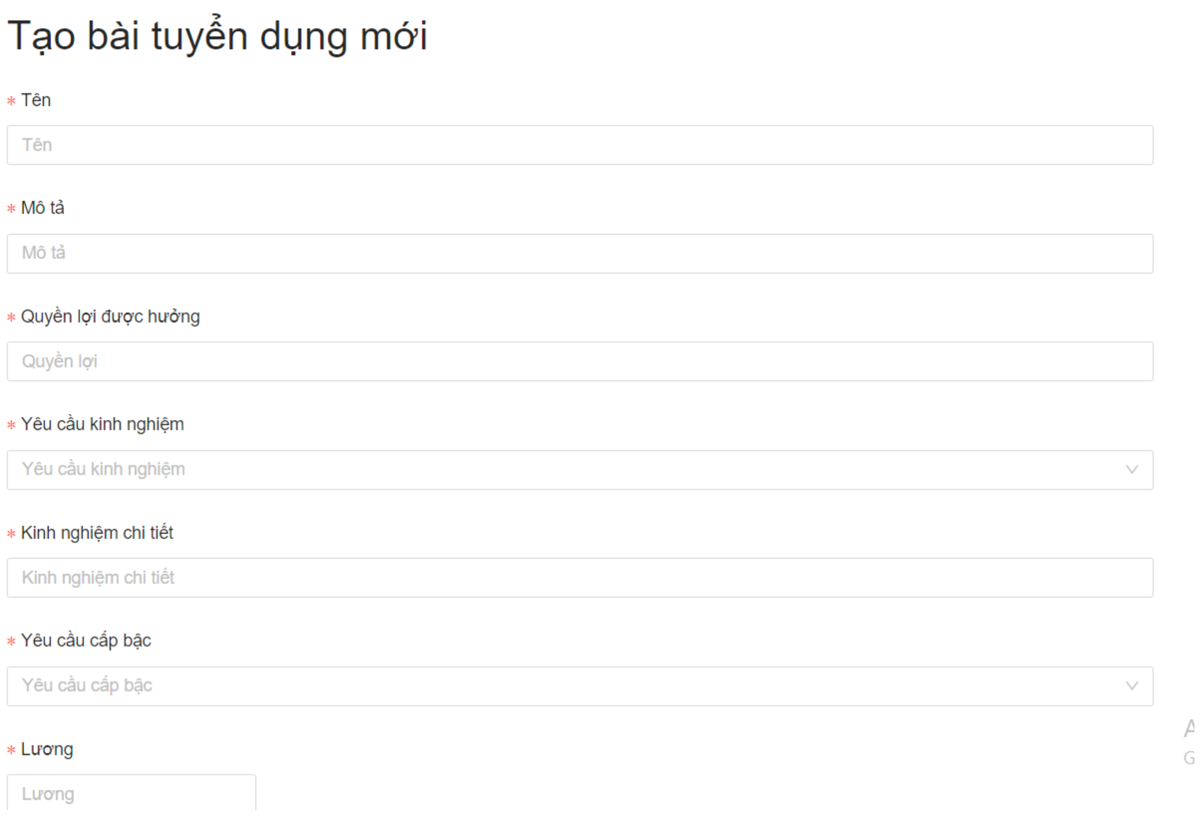
* Số điện thoại

số điện thoại

* Tài khoản

FIGURE 33: REGISTER FOR COMPANY PAGE

Next is the feature of creating new recruitment postings for the company. The company will fill in the desired recruitment requirements to be able to create a new recruitment posting

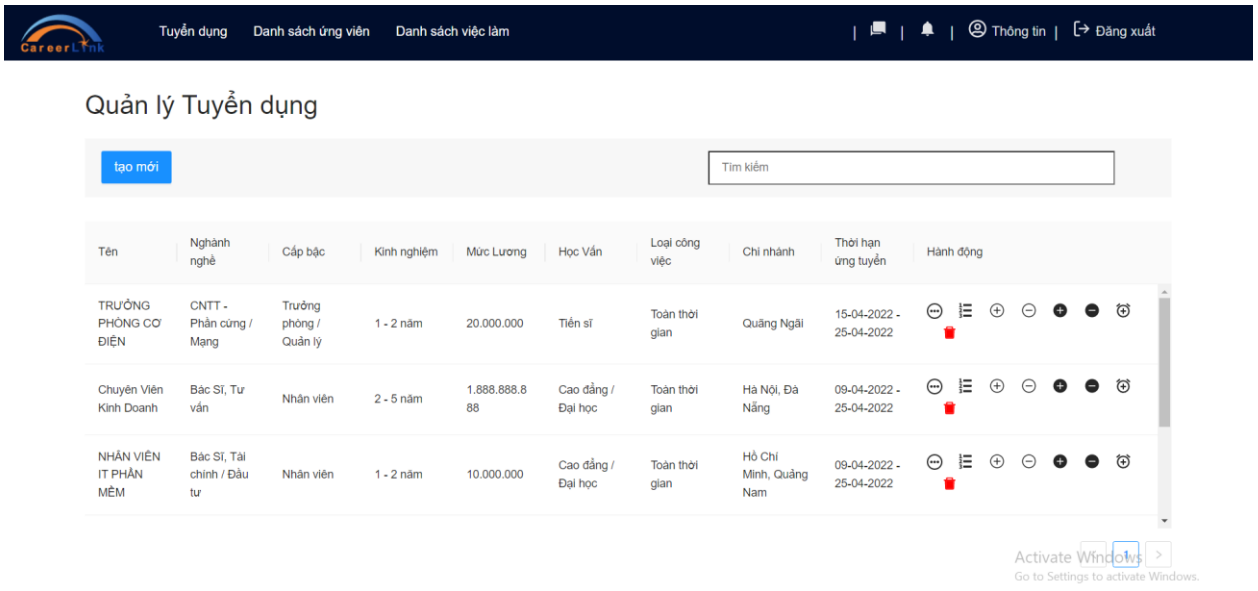


Tạo bài tuyển dụng mới

* Tên	Tên
* Mô tả	Mô tả
* Quyền lợi được hưởng	Quyền lợi
* Yêu cầu kinh nghiệm	Yêu cầu kinh nghiệm
* Kinh nghiệm chi tiết	Kinh nghiệm chi tiết
* Yêu cầu cấp bậc	Yêu cầu cấp bậc
* Lương	Lương

FIGURE 34: CREATE RECRUITMENT PAGE

After creating the job recruitment, there will be a page where the company can manage their job recruitments.



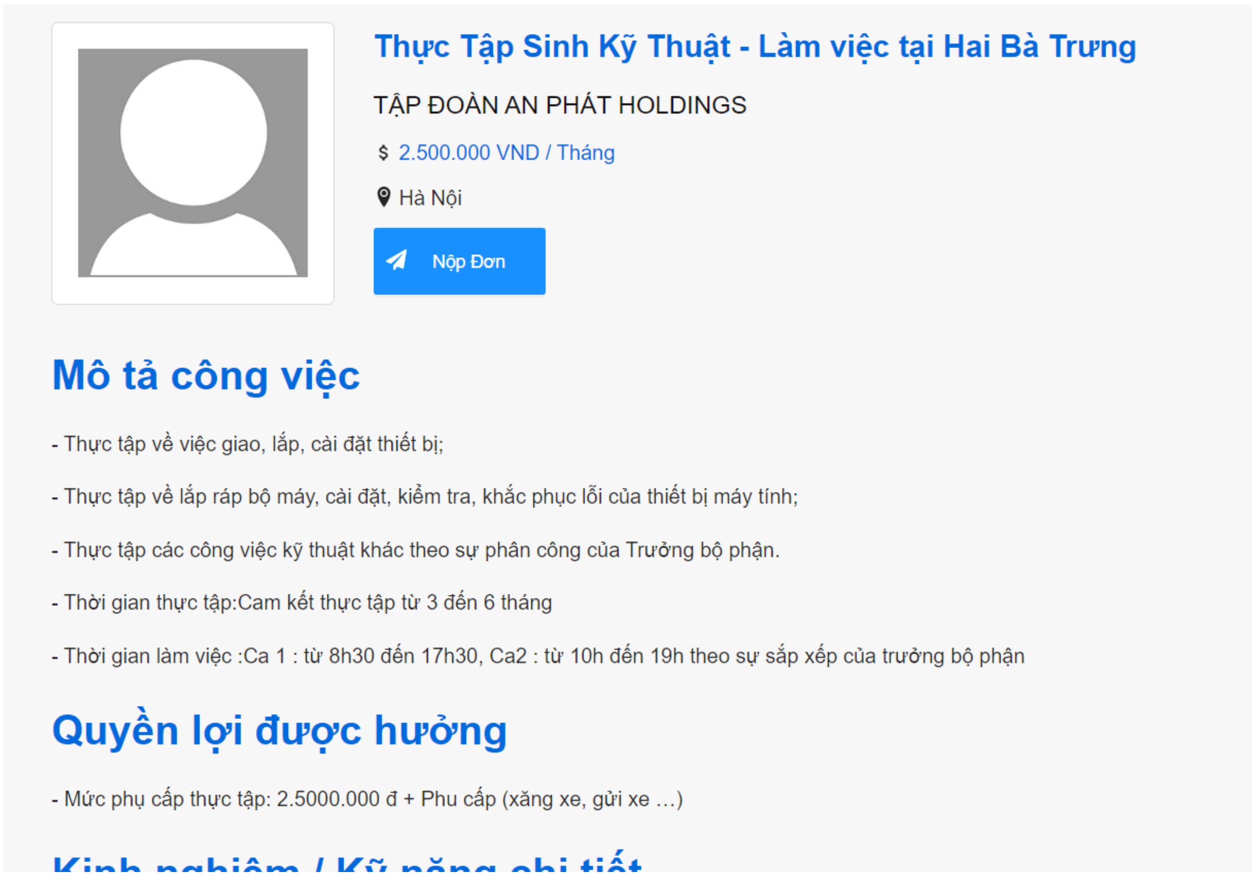
Quản lý Tuyển dụng

Tên	Nghành nghề	Cấp bậc	Kinh nghiệm	Mức Lương	Học Vấn	Loại công việc	Chi nhánh	Thời hạn ứng tuyển	Hành động
TRƯỞNG PHÒNG CƠ ĐIỆN	CNTT - Phần cứng / Mạng	Trưởng phòng / Quản lý	1 - 2 năm	20.000.000	Tiến sĩ	Toàn thời gian	Quãng Ngãi	15-04-2022 - 25-04-2022	
Chuyên Viên Kinh Doanh	Bác Sĩ, Tư vấn	Nhân viên	2 - 5 năm	1.888.888.888	Cao đẳng / Đại học	Toàn thời gian	Hà Nội, Đà Nẵng	09-04-2022 - 25-04-2022	
NHÂN VIÊN IT PHẦN MỀM	Bác Sĩ, Tài chính / Đầu tư	Nhân viên	1 - 2 năm	10.000.000	Cao đẳng / Đại học	Toàn thời gian	Hồ Chí Minh, Quảng Nam	09-04-2022 - 25-04-2022	

FIGURE 35: MANAGE RECRUITMENT PAGE

Next is the feature of submitting CV for users, including the following steps:

- Step 1: Select the job post employee want to apply for, a detailed description page will be displayed to the employee



Thực Tập Sinh Kỹ Thuật - Làm việc tại Hai Bà Trưng

TẬP ĐOÀN AN PHÁT HOLDINGS

$ 2.500.000 VND / Tháng

Hà Nội

Nộp Đơn

Mô tả công việc

- Thực tập về việc giao, lắp, cài đặt thiết bị;

- Thực tập về lắp ráp bộ máy, cài đặt, kiểm tra, khắc phục lỗi của thiết bị máy tính;

- Thực tập các công việc kỹ thuật khác theo sự phân công của Trưởng bộ phận.

- Thời gian thực tập:Cam kết thực tập từ 3 đến 6 tháng

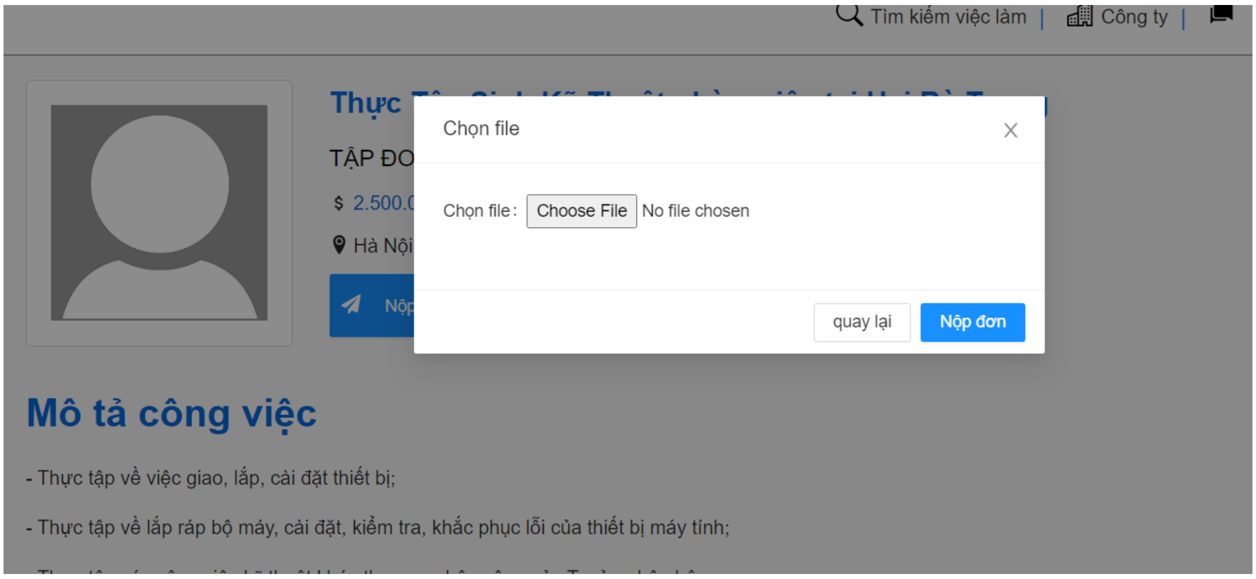
- Thời gian làm việc :Ca 1 : từ 8h30 đến 17h30, Ca2 : từ 10h đến 19h theo sự sắp xếp của trưởng bộ phận

Quyền lợi được hưởng

- Mức phụ cấp thực tập: 2.5000.000 đ + Phu cấp (xăng xe, gửi xe …)

FIGURE 36: SUBMIT CV STEP 1

- Step 2: Next, the employee clicks on the submit button, a modal will appear for them to choose the file.



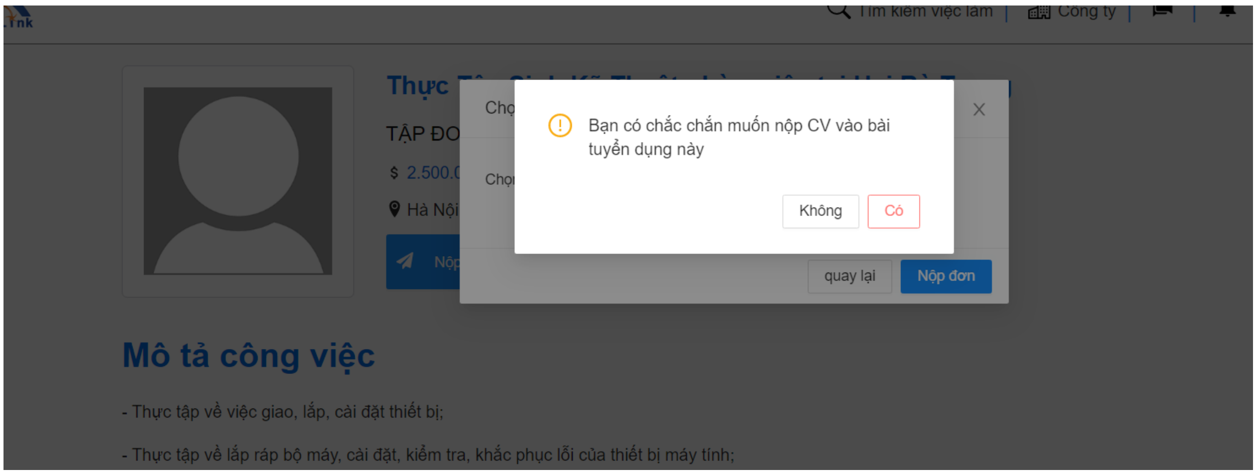
Mô tả công việc

- Thực tập về việc giao, lắp, cài đặt thiết bị;

- Thực tập về lắp ráp bộ máy, cài đặt, kiểm tra, khắc phục lỗi của thiết bị máy tính;

FIGURE 37: SUBMIT CV STEP 2

- Step 3: Next, the employee will select his CV file in pdf format and click the submit button. After clicking the apply button, the system will confirm that they are sure and will submit their CV to the company to apply for this job.


Tìm kiếm việc làm | Công ty

Thực Tập Sinh Kỹ Thuật

TẬP ĐO

$ 2.500.0

Hà Nội

Bạn có chắc chắn muốn nộp CV vào bài tuyển dụng này

Không Có

quay lại Nộp đơn

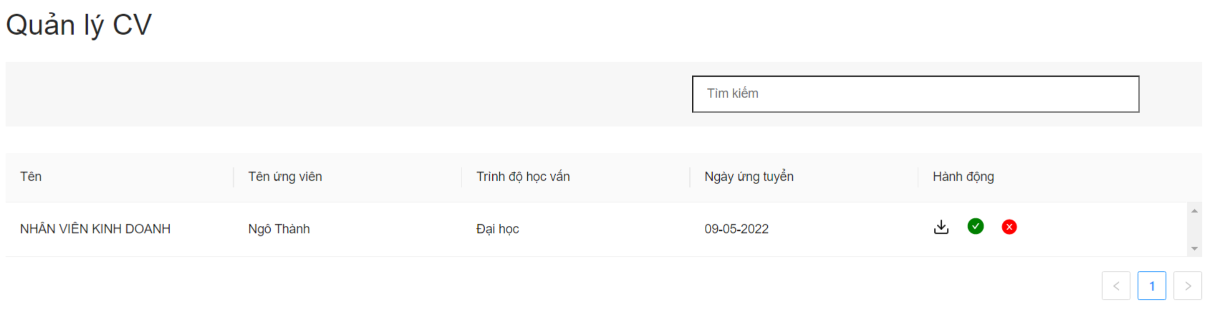
Mô tả công việc

- Thực tập về việc giao, lắp, cài đặt thiết bị;

- Thực tập về lắp ráp bộ máy, cài đặt, kiểm tra, khắc phục lỗi của thiết bị máy tính;

FIGURE 38: SUBMIT CV STEP 3

Next is the company's CV management feature, they can see all the applications of employees. They can optionally download the CV, accept or decline the job recruitment.


Quản lý CV

Tìm kiếm

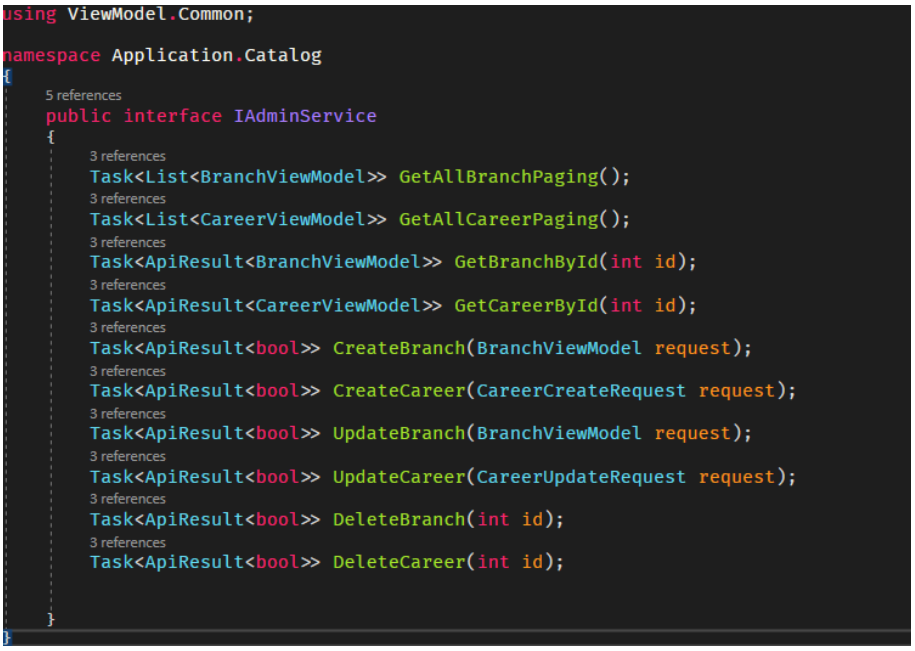
Tên	Tên ứng viên	Trình độ học vấn	Ngày ứng tuyển	Hành động
NHÂN VIÊN KINH DOANH	Ngô Thành	Đại học	09-05-2022	⤓ ✓ ✕

FIGURE 39: MANAGE CV

10. Testing

10.1. Explain the process testing

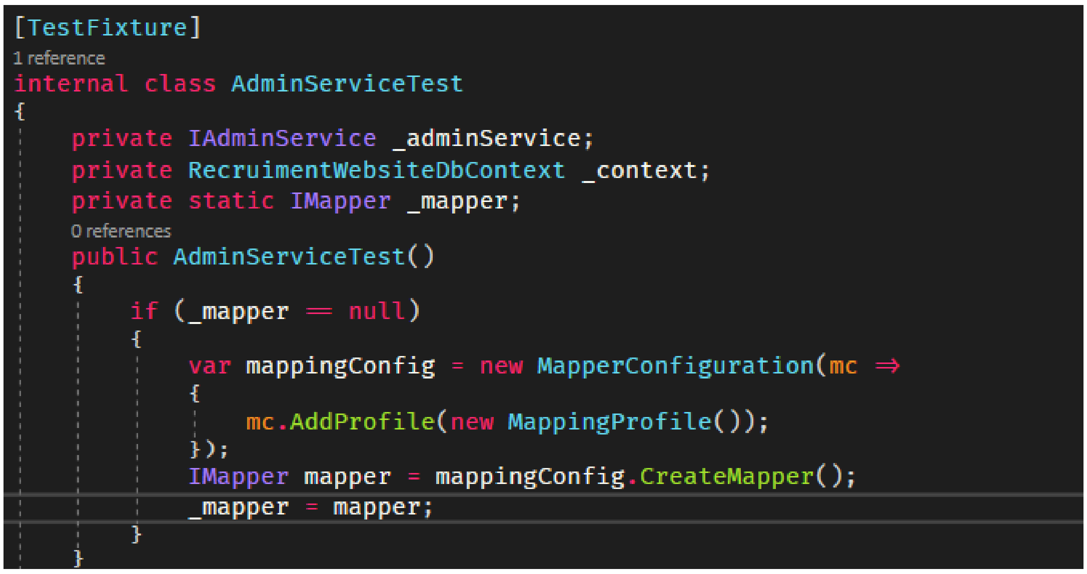
The Api part of this project I use Repository Design Pattern to write, so I decided to use unit test for the testing part of this project. In this section, I will explain the process of testing functions in IAdminService as an example. Below is all the function that I need to test in this section



```
using ViewModel.Common;

namespace Application.Catalog
{
    5 references
    public interface IAdminService
    {
        3 references
        Task<List<BranchViewModel>> GetAllBranchPaging();
        3 references
        Task<List<CareerViewModel>> GetAllCareerPaging();
        3 references
        Task<ApiResult<BranchViewModel>> GetBranchById(int id);
        3 references
        Task<ApiResult<CareerViewModel>> GetCareerById(int id);
        3 references
        Task<ApiResult<bool>> CreateBranch(BranchViewModel request);
        3 references
        Task<ApiResult<bool>> CreateCareer(CareerCreateRequest request);
        3 references
        Task<ApiResult<bool>> UpdateBranch(BranchViewModel request);
        3 references
        Task<ApiResult<bool>> UpdateCareer(CareerUpdateRequest request);
        3 references
        Task<ApiResult<bool>> DeleteBranch(int id);
        3 references
        Task<ApiResult<bool>> DeleteCareer(int id);

    }
}
```

FIGURE 40: IADMINSERVICE INTERFACE

Step 1: Create Class Test

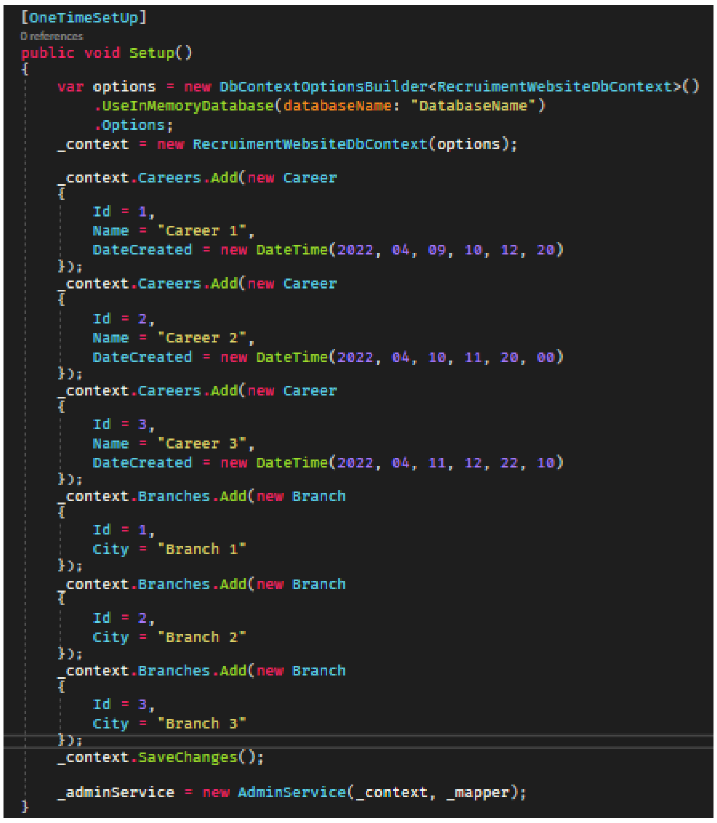


```
[TestFixture]
1 reference
internal class AdminServiceTest
{
    private IAdminService _adminService;
    private RecruimentWebsiteDbContext _context;
    private static IMapper _mapper;
    0 references
    public AdminServiceTest()
    {
        if (_mapper == null)
        {
            var mappingConfig = new MapperConfiguration(mc =>
            {
                mc.AddProfile(new MappingProfile());
            });
            IMapper mapper = mappingConfig.CreateMapper();
            _mapper = mapper;
        }
    }
```

FIGURE 41: CREATE CLASS TEST

First, I will create the AdminServiceTest class as shown above, then I will declare the necessary variables to serve the testing process. In this part, I created a mapper object and assigned it to the _mapper variable in the constructor as shown above.

Step 2: Create a dummy database



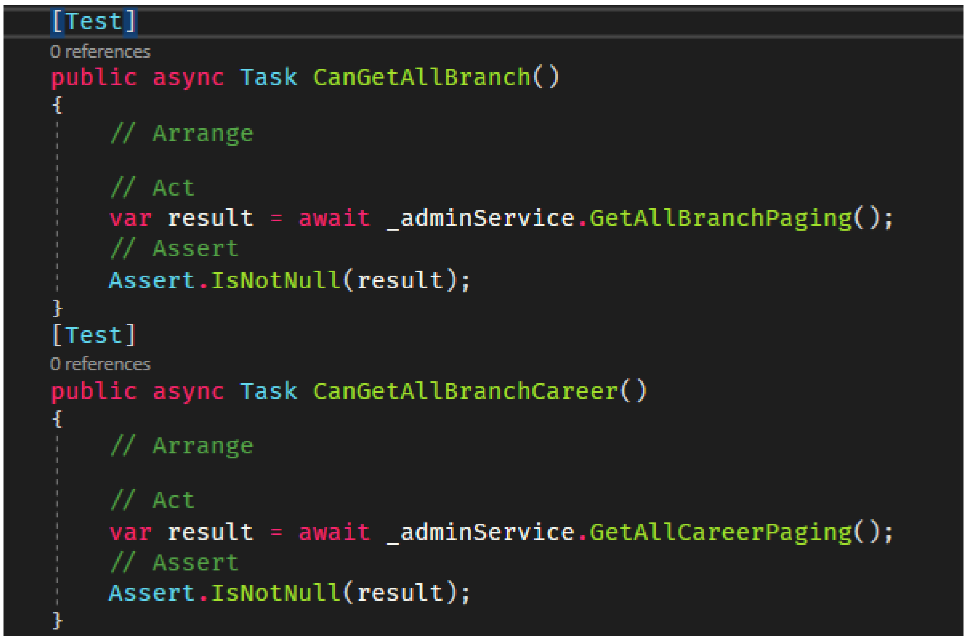
```
[OneTimeSetUp]
0 references
public void Setup()
{
    var options = new DbContextOptionsBuilder<RecruimentWebsiteDbContext>()
        .UseInMemoryDatabase(databaseName: "DatabaseName")
        .Options;
    _context = new RecruimentWebsiteDbContext(options);

    _context.Careers.Add(new Career
    {
        Id = 1,
        Name = "Career 1",
        DateCreated = new DateTime(2022, 04, 09, 10, 12, 20)
    });
    _context.Careers.Add(new Career
    {
        Id = 2,
        Name = "Career 2",
        DateCreated = new DateTime(2022, 04, 10, 11, 20, 00)
    });
    _context.Careers.Add(new Career
    {
        Id = 3,
        Name = "Career 3",
        DateCreated = new DateTime(2022, 04, 11, 12, 22, 10)
    });
    _context.Branches.Add(new Branch
    {
        Id = 1,
        City = "Branch 1"
    });
    _context.Branches.Add(new Branch
    {
        Id = 2,
        City = "Branch 2"
    });
    _context.Branches.Add(new Branch
    {
        Id = 3,
        City = "Branch 3"
    });
    _context.SaveChanges();

    _adminService = new AdminService(_context, _mapper);
}
```

FIGURE 42: CREATE A DUMMY DATABASE

In this part I will create a Setup class declared OneTimeSetUp so that each time test data created in this class will be generated and when the test is done, the data will disappear. So, in this part I have created a dummy Database assigned to the _context variable and an AdminService object assigned to the adminService variable.

Step 3: Test function

- **CanGetAll Function**

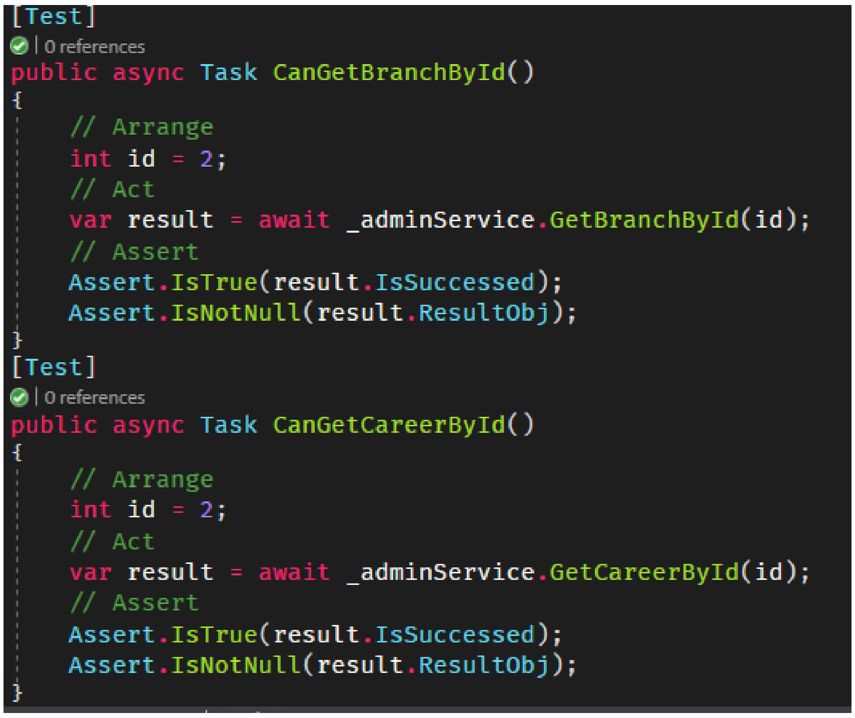


```
[Test]
0 references
public async Task CanGetAllBranch()
{
    // Arrange

    // Act
    var result = await _adminService.GetAllBranchPaging();
    // Assert
    Assert.IsNotNull(result);
}
[Test]
0 references
public async Task CanGetAllBranchCareer()
{
    // Arrange

    // Act
    var result = await _adminService.GetAllCareerPaging();
    // Assert
    Assert.IsNotNull(result);
}
```

FIGURE 43: CANGETALL FUNCTION

Function GetAll is the function used to get all the data in the Database so I will check if the returned result is null or not. If the test result is not null then the function is working correctly.

- **CanGetById Function**

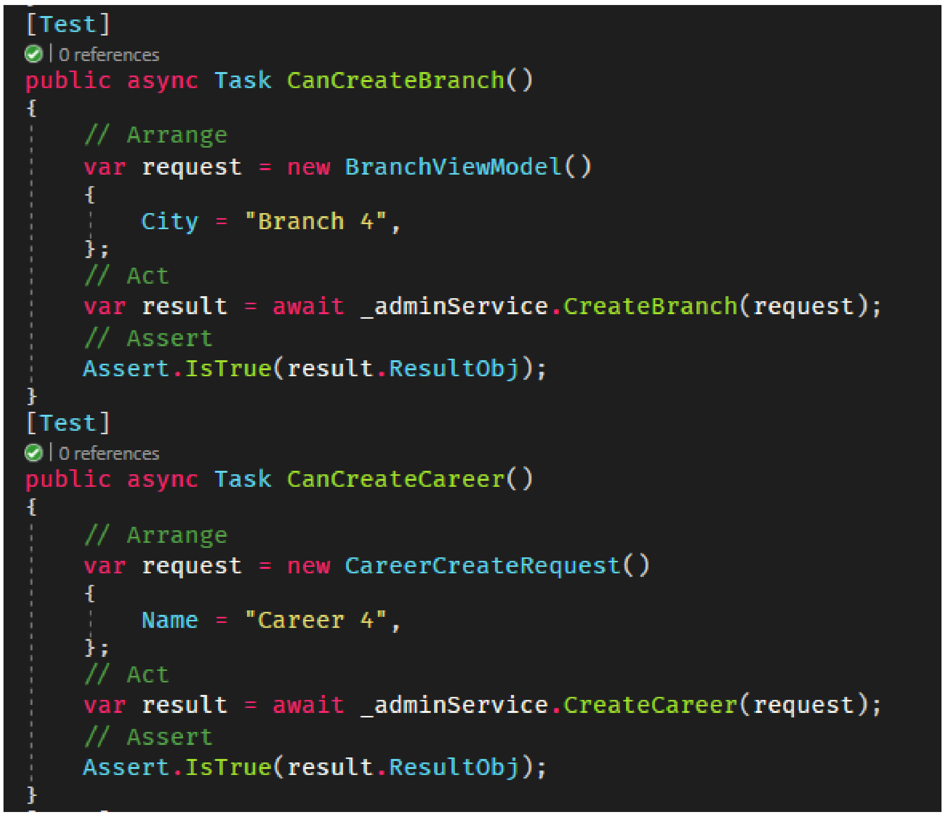


```
[Test]
0 references
public async Task CanGetBranchById()
{
    // Arrange
    int id = 2;
    // Act
    var result = await _adminService.GetBranchById(id);
    // Assert
    Assert.IsTrue(result.IsSuccessed);
    Assert.IsNotNull(result.ResultObj);
}
[Test]
0 references
public async Task CanGetCareerById()
{
    // Arrange
    int id = 2;
    // Act
    var result = await _adminService.GetCareerById(id);
    // Assert
    Assert.IsTrue(result.IsSuccessed);
    Assert.IsNotNull(result.ResultObj);
}
```

FIGURE 44: CANGETBYID FUNCTION

function GetById is a function used to get data by id in the Database, so the Arrange part I will create an id variable containing the id that I need to search and then proceed. In this part, I will check if the returned result is successful and the data is null. If both Asserts are correct, then the function is working correctly

- **CanCreate Function**

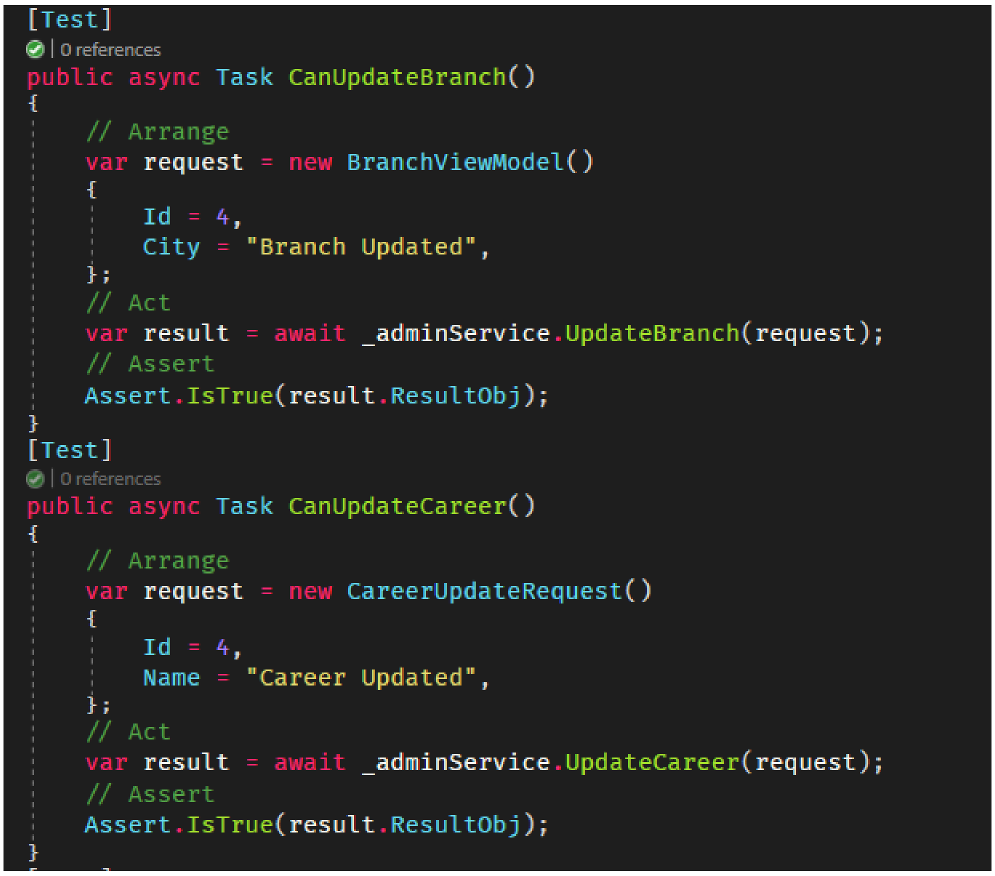


```
[Test]
0 references
public async Task CanCreateBranch()
{
    // Arrange
    var request = new BranchViewModel()
    {
        City = "Branch 4",
    };
    // Act
    var result = await _adminService.CreateBranch(request);
    // Assert
    Assert.IsTrue(result.ResultObj);
}
[Test]
0 references
public async Task CanCreateCareer()
{
    // Arrange
    var request = new CareerCreateRequest()
    {
        Name = "Career 4",
    };
    // Act
    var result = await _adminService.CreateCareer(request);
    // Assert
    Assert.IsTrue(result.ResultObj);
}
```

FIGURE 45: CANCREATE FUNCTION

The Create function is the function used to create new data and save it in the Database, so the Arrange part I will create a variable containing the data that I need to create and then proceed. In this part, the return result will be true if it succeeds or false if it fails so I want to check the result is true or not. If Assert is correct, then the function is working correctly

- **CanUpdate Function**

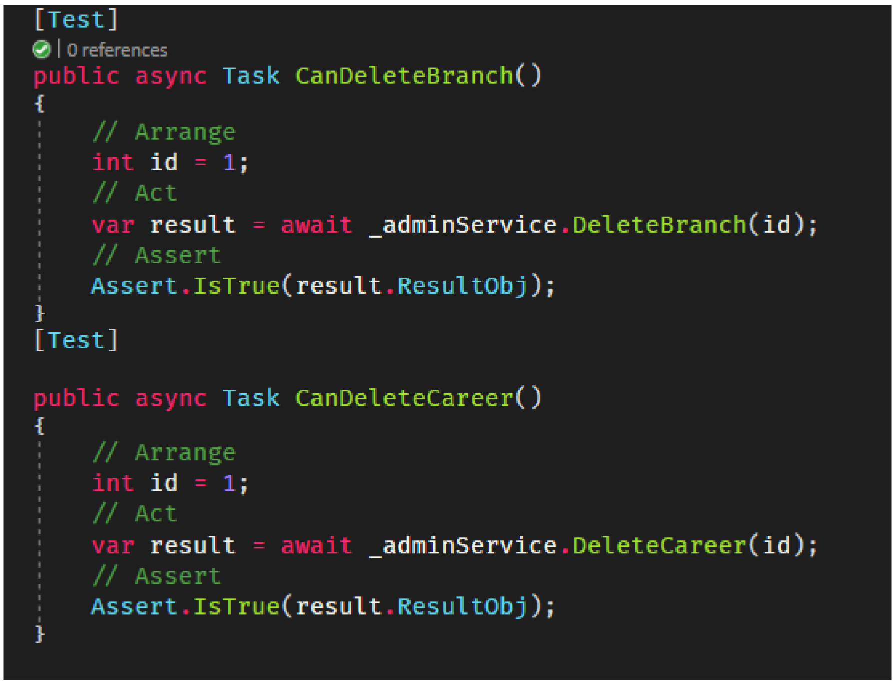


```
[Test]
0 references
public async Task CanUpdateBranch()
{
    // Arrange
    var request = new BranchViewModel()
    {
        Id = 4,
        City = "Branch Updated",
    };
    // Act
    var result = await _adminService.UpdateBranch(request);
    // Assert
    Assert.IsTrue(result.ResultObj);
}
[Test]
0 references
public async Task CanUpdateCareer()
{
    // Arrange
    var request = new CareerUpdateRequest()
    {
        Id = 4,
        Name = "Career Updated",
    };
    // Act
    var result = await _adminService.UpdateCareer(request);
    // Assert
    Assert.IsTrue(result.ResultObj);
}
```

FIGURE 46: CANUPDATE FUNCTION

Function Update is a function used to change data in the Database, so the Arrange part I will create a variable containing the data that I need to change and then proceed. Similar to the Create part, the return result will be true if it succeeds or false if it fails so I want to check the result is true or not. If Assert is correct, then the function is working correctly

- **CanDelete Function**



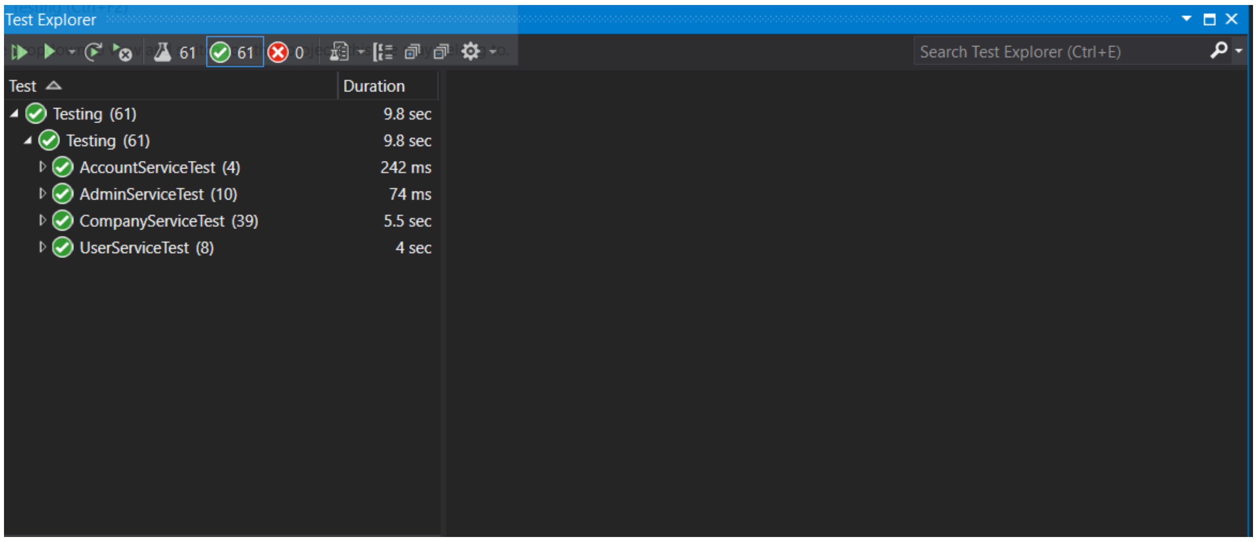
```
[Test]
0 references
public async Task CanDeleteBranch()
{
    // Arrange
    int id = 1;
    // Act
    var result = await _adminService.DeleteBranch(id);
    // Assert
    Assert.IsTrue(result.ResultObj);
}
[Test]

public async Task CanDeleteCareer()
{
    // Arrange
    int id = 1;
    // Act
    var result = await _adminService.DeleteCareer(id);
    // Assert
    Assert.IsTrue(result.ResultObj);
}
```

FIGURE 47: CANDELETE FUNCTION

function Update is a function used to delete data in the Database, so the Arrange part I will create a variable containing the id that I need to delete and then proceed. The return result of the previous section will also be true if it succeeds or false if it fails so I want to check if the result is true or not. If Assert is correct, then the function is working correctly.

10.2. Summary of result testing

In section 10.1 I explained my unit test execution process and I applied to all 61 functions and all was successful. The results are shown in the figure below:



```
Test                          Duration
Testing (61)                  9.8 sec
  Testing (61)                9.8 sec
    AccountServiceTest (4)    242 ms
    AdminServiceTest (10)     74 ms
    CompanyServiceTest (39)   5.5 sec
    UserServiceTest (8)       4 sec
```

FIGURE 48: SUMMARY OF RESULT TESTING

11. Evaluation

11.1. Summarised Key findings from the project

11.1.1. Technical

Discovery	Justification
Back-End (.NET Core Framework)	During project development, I learned how to construct APIs with the.NET Core Framework. In addition, I learnt how to use the automapper library and design patterns such as the Repository and Dependency Injection design patterns. In addition, instead of using Postman, I learnt how to use Swagger to define the structure of the API and test it.
Front-End (React JS)	This is my first time doing a project using React JS, so there are still quite a few shortcomings. However, after building applications using React, I believe this is a good framework. It not only allows me to code more effortlessly, but it also has a plethora of libraries that allow me to construct apps rapidly. Furthermore, it has a robust support network, so I can easily receive help with my problems.
Testing (Unit Test)	During project development, I learnt how to utilize unit testing to test the application's functionality. Instead of spending a long time evaluating each function, I now just need approximately 10 seconds to test all of the system functions in the Back-End portion.

TABLE 4: EVALUATION TECHNICAL

11.1.2. Business

Regarding the business part of the project, I've seen that the number of jobless people in Vietnam today is still fairly large, therefore I believe this project will be extremely suitable to everyone's job search people. I also considered using the benefits of similar applications on the market while limiting their drawbacks when applied to the project. In addition, I study and build a few additional unique features to provide people with the greatest product possible.

11.2. Recommendations for future development

Future development	Justification
Responsive	Currently, the program works fine on computers but not on other devices such as tablets or mobile phones. As a result, I will provide this functionality to help users with the greatest possible experience.
Create CV directly	In the future, I need to include a function that allows workers to build and submit CVs immediately from the application. This capability will assist employees in swiftly creating and submitting CVs to employers.

Interview	I want to create a video call function so that employers and workers may conduct interviews directly on the application rather than using Google Meet or Messenger.
Mobile App	People nowadays use smart phones a lot, thus I believe designing a mobile app would attract more consumers. So, I will create a mobile app for a future application.
Improved user interface	Because this is my first time building an app using ReactJS so I just created a simple interface. So, in the future, I will try to improve the app's user interface as much as possible.
Deployment	The project has only developed a local host application, in the future should deploy and host online.

TABLE 5: RECOMMENDATIONS FOR FUTURE DEVELOPMENT

11.3. Project Evaluation

After completing the project, the basic ASEP application achieved all of the objectives that were initially specified. ASEP relies on research on the strengths and drawbacks of existing market applications to improve and correct to provide users with the optimal application. The tool also allows users to talk between employees and employers, allowing them to readily exchange information and make recruiting easier. Furthermore, the program provides users with a variety of useful functions such as chat between employee and employer, commenting on job posts and watching announcements. The program still has certain shortcomings, such as an unappealing user interface. Furthermore, the program has not yet created a few capabilities such as assisting users in conveniently creating CVs, a direct video chat function for employee and employer to easily speak with each other, or the development of a mobile app to attract more people to use the app. In the future, I will try to repair the system's remaining flaws in order to provide users with the greatest application possible.

11.4. Personal Evaluation

The things I have achieved during the development of this project are as follows:

- On the positive side, I have enhanced my research and learning abilities, allowing me to find interesting topics that can be used to the project. I've also moved ahead of myself and can accomplish things I never believed I could do before, so I believe I'll be able to progress even farther in the future.

- In addition to the positives, I identified some of my limitations when developing ASEP apps. The first is that my knowledge is rather limited, so I have to learn while doing, so certain features don't function flawlessly or I even have to eliminate some features that I can't perform. Furthermore, I do not manage my work time properly; some jobs are postponed and are not done on time as intended.

- Aside the above two things, the knowledge I have learned is really the most important thing. I learned how to create a client-server application utilizing two languages: .NET Core and ReactJS. I also learnt how to rapidly and accurately fix faulty code. I also learnt how to do rapid and precise reference searches. I finally know where my flaws are, and I'll work on overcoming them in the future to improve myself.

11.5. Conclusion

After doing this project, I learnt a lot and gained a lot of useful experience. I learnt how to do everything from gathering user needs to establishing system requirements to project planning and development while working on ASEP. In addition, I must research and learn new technology. Furthermore, while working on this project, I learnt a lot of new things and got a lot of experience for myself. I'd want to thank Mr. Tran Trong Minh and Hoang Nhu Vinh for accompanying and assisting me much during the project's execution. Finally, I'd want to thank the University of Greenwich for providing a positive learning atmosphere and providing me with the basic knowledge I need to pursue a career in the IT business in the future.

12. References

Aggarwal, S., 2018. Modern Web-Development using ReactJS. *Proceeding of National Conference on Advances in Computing and Communications,* 5(1), pp. 133-137.

Cekindo Vietnam, 2021. *cekindo.* [Online]
Available at: https://www.cekindo.vn/blog/recruitment-in-vietnam-labour-market-and-hiring-process
[Accessed 24 December 2021].

Chong, T. T.-L., Li, X. & Yip, C., 2021. Economic and Political Studies. *The impact of COVID-19 on ASEAN,* Volume 9, pp. 166-185.

Das, K., n.d. *vietnam briefing.* [Online]
Available at: https://www.vietnam-briefing.com/news/labor-market-trends-vietnam.html/
[Accessed 29 June 2018].

DIDAOUI, N., 2018. *linkedin.* [Online]
Available at: https://www.linkedin.com/pulse/top-10-reasons-choose-sql-server-2019-nabil-didaoui
[Accessed 2 Oct 2018].

Huong T. T. Nguyen, T. T. N. V. A. T. D. L. H. N. G. T. V. H. L. T. N. H. T. N. a. H. T. L., 2020. *COVID-19 Employment Crisis in Vietnam: Global Issue, National Solutions,* s.l.: Front. Public Health.

Lijuan, G. S. a. W., 2013. Ethical perspectives on e-commerce: an empirical investigation. pp. 7-9.

Lukasz, P., 2020. *iteo.* [Online]
Available at: https://iteo.com/blog/post/why-you-should-consider-net-core-for-your-next-project/
[Accessed 3 Nov 2020].

Ngoc, N. M., June 2021. *Foreign direct investment net inflows in Vietnam from 2013 to 2020,* Vietnam: statista.

Nishith Desai Associates, 2005. LEGAL ISSUES IN eCOMMERCE. *Legal & Tax Counselling Worldwide*, pp. 4-12.

Roomi, M., 2020. *hitechwhizz.* [Online]
Available at: https://www.hitechwhizz.com/2020/11/5-advantages-and-disadvantages-drawbacks-benefits-of-client-server-network.html
[Accessed 12 November 2020].

Team, E., 2017. *techastral.* [Online]
Available at: https://www.techastral.com/job-search-vietnam/
[Accessed 5 May 2017].

Team, I. E., 2021. *indeed.* [Online]
Available at: https://www.indeed.com/career-advice/career-development/what-is-web-application
[Accessed 11 November 2021].

Tri minh law corporation, 2021. *luattriminh.* [Online]
Available at: https://luattriminh.vn/bi-cam-hoat-dong-thuong-mai-dien-tu-vi-cac-hanh-vi-sau-

day.html
[Accessed 2021].