Trường Đại Học Sư Phạm Kỹ Thuật

Khoa Đào Tạo Chất Lượng Cao



**ĐỒ ÁN TỐT NGHIỆP**

**NGHIÊN CỨU SPRING MVC, RESTFUL WEBSERVICE, REACTJS VÀ PHÁT TRIỂN WEBSITE TÌM VIỆC LÀM**

TPHCM, Tháng 12 năm 2018

# **Nhận xét của giáo viên hướng dẫn**

.....................................................................................................................................

....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

....................................................................................................................................

....................................................................................................................................

.....................................................................................................................................

Ngày.........Tháng..........Năm.........

Giáo Viên Hướng Dẫn

(Ký, ghi rõ họ và tên)

# **LỜI CẢM ƠN**

Đầu tiên em xin gửi lời cảm ơn đến Thầy Lê Văn Vinh – người hướng dẫn tiểu luận chuyên ngành ở trường, thầy đã tận tình giảng dạy những kiến thức cơ bản giúp chúng em có được nền tảng vững chắc để hoàn thành được đề tài.

Em cũng xin chân thành cảm ơn các quý thầy cô trong khoa Công Nghệ Thông Tin đã giúp đỡ hỗ trợ kiến thức cũng như giải đáp thắc mắc của em. Cùng với đó, em xin được gửi cảm ơn đến các bạn cùng khóa đã cung cấp nhiều thông tin và kiến thức hữu ích giúp cho em hoàn thiện đề tài hơn.

Bài thu hoạch được thực hiện trong khoảng thời gian gần 2 tuần. Khoãng thời gian có hạn, cùng với kiến thức còn hạn chế và còn nhiều bỡ ngỡ khác do đó thiếu sót là điều không thể tránh khỏi nên em rất mong nhận được những ý kiến đóng góp quý báo của các quý Thầy Cô để kiến thức của em được hoàn thiện hơn sau này. Em xin chân thành cảm ơn.

Chúng em mong thầy thông cảm và góp ý, nhận xét để chúng em có thể khắc phục được những sai sót ở các bài sau.

Kính chúc các thầy dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho các thế hệ mai sau..

Ký tên

Nguyễn Xuân Tuấn

# **LỜI NÓI ĐẦU**

Công nghệ ngày càng phát triển, đặc biệt là trong kỷ nguyên 4.0 như ngày nay thì công nghệ là thứ không thể thiếu trong đời sống chứng ta. Với công nghệ ta có thể dùng thông tin để xử lý, rút ngắn thời gian các công việc của chứng ta, giúp ta làm được nhiều việc hơn trong thời gian ngắn hơn. Ngoài ra, ứng dụng của công nghệ để giúp chứng ta tập hợp và tìm kiếm thông tin một cách nhanh chóng. Điều đó được thể hiện đặc biệt trong việc tìm kiếm thông tin việc làm, tuyển dụng.

Vì thế, người thực hiện đề tài đã nghiên cứu và phát triển một website việc làm nhằm đáp ứng được nhu cầu tìm kiếm việc một cách nhanh chóng như hiện nay.

# **MỤC LỤC**

[**Nhận xét của giáo viên hướng dẫn** 2](#_Toc532472432)

[**LỜI CẢM ƠN** 3](#_Toc532472433)

[**LỜI NÓI ĐẦU** 4](#_Toc532472434)

[**MỤC LỤC** 4](#_Toc532472435)

[**DANH MỤC HÌNH** 4](#_Toc532472436)

[**DANH MỤC BẢNG** 5](#_Toc532472437)

[**PHẦN MỞ ĐẦU** 6](#_Toc532472438)

[**1.** **Đặt vấn đề** 6](#_Toc532472439)

[**2.** **Mục tiêu và nhiệm vụ của đề tài** 6](#_Toc532472440)

[**2.1** **Mục tiêu** 6](#_Toc532472441)

[**2.2** **Nhiệm vụ** 6](#_Toc532472442)

[**3.** **Phương pháp nghiên cứu** 6](#_Toc532472443)

[**4.** **Bố cục của báo cáo** 7](#_Toc532472444)

[**PHẦN NỘI DUNG** 7](#_Toc532472445)

[**CHƯƠNG 1: CÔNG NGHỆ SỬ DỤNG** 7](#_Toc532472446)

[**1.** **SPRING BOOT** 7](#_Toc532472447)

[1.1 Những kiến thức cần chuẩn bị 7](#_Toc532472448)

[**1.2** **Giới thiệu về Spring boot [3]** 8](#_Toc532472449)

[**1.3** **Ưu điểm của Spring boot** 9](#_Toc532472450)

[**2.** **REACT JS [4]** 10](#_Toc532472451)

[**2.1** **Đặt vấn đề** 10](#_Toc532472452)

[**2.2** **Giới thiệu ReactJs** 10](#_Toc532472453)

[**2.2.1** **Virtual Dom** 10](#_Toc532472454)

[**2.2.2** **JSX** 11](#_Toc532472455)

[**2.2.3** **Component** 12](#_Toc532472456)

[**2.2.4** **Props và State** 12](#_Toc532472457)

[**2.3** **Kết luận** 12](#_Toc532472458)

[**3.** **RESTFUL API** 12](#_Toc532472459)

[**3.1** **Giới thiệu** 12](#_Toc532472460)

[**3.2** **Restful, Restful API là gì [5]** 13](#_Toc532472461)

[**3.3** **Kiến trúc của Rest [6]** 13](#_Toc532472462)

[**3.4** **Ưu điểm** 14](#_Toc532472463)

[**CHƯƠNG 2: KHẢO SÁT HIỆN TRẠNG VÀ XÁC ĐỊNH YÊU CẦU** 14](#_Toc532472464)

[**1.** **Khảo sát nghiệp vụ** 14](#_Toc532472465)

[**2.** **Xác định yêu cầu** 16](#_Toc532472466)

[**CHƯƠNG 3: THIẾT KẾ TRANG WEB** 21](#_Toc532472467)

[**1.** **Lược đồ lớp** 21](#_Toc532472468)

[**2.** **Lược đồ Sequence** 22](#_Toc532472469)

[**3.** **Thiết kế CSDL** 32](#_Toc532472470)

[**4.** **Thiêt kế giao diện** 34](#_Toc532472471)

[**CHƯƠNG 4: CÀI ĐẶT VÀ KIỂM THỬ** 37](#_Toc532472472)

[**1.** **Cài Đặt** 37](#_Toc532472473)

[**2.** **Kiểm Thử** 38](#_Toc532472474)

[**PHẦN KẾT LUẬN** 39](#_Toc532472475)

[**1.** **Kết quả đạt được và hướng phát triển** 39](#_Toc532472476)

[**2.** **Hạn chế** 39](#_Toc532472477)

[**3.** **Hướng phát triển** 39](#_Toc532472478)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 39](#_Toc532472479)

# **DANH MỤC HÌNH**

Hình 1: Mô hình MVC

Hình 2: Spring Framework Runtime

Hình 3: Spring Features

Hình 4: Vitual Dom

Hình 5: React Component tree

Hình 6: Rest API Design

Hình 7: trang chủ website khảo sát

Hình 8: Trang danh sách công viêc website khảo sát

Hình 9: trang quản lý tài khoản website khảo sát

Hình 10: use case diagram

Hình 11: lược đồ lớp

Hình 12: sequence đăng nhập

Hình 13: sequence Đăng Ký Thành Viên

Hình 14: sequence Đăng Ký Tuyển Dụng

Hình 14: sequence Đăng Công Việc

Hình 15: Sequence Sửa Cộng Việc

Hình 17: sequence Sửa Hồ Sơ

Hình 18: sequence Tìm kiếm công việc theo tiêu chí (chuyên ngành, vị trí)

Hình 19: senquence Nhận công việc

Hình 20: sequence tìm kiếm ứng viên

Hình 21: Database Diagrams

Hình 21 : Giao diện Trang chủ

Hình 22 : Giao diện Trang chủ

Hình 23: Giao diện Trang chủ

Hình 24: Modal Đăng nhập

Hình 25: kiến trúc website

# **DANH MỤC BẢNG**

Bảng 1: đặc tả usecase đăng nhập

Bảng 2: đặc tả usecase đăng xuất

Bảng 3: đặc tả usecase đăng ký

Bảng 4: đặc tả usecase Tìm kiếm công việc

Bảng 5: đặc tả usecase Đăng công việc

Bảng 6: đặc tả usecase sửa công việc

Bảng 7: đặc tả usecase xóa công việc

Bảng 8: đặc tả usecase ứng tuyển công việc

Bảng 9: đặc tả usecase sửa hồ sơ

Bảng 10: đặc tả usecase xem lịch sử

Bảng 11: bảng đặc tả database diagram

# **PHẦN MỞ ĐẦU**

1. **Đặt vấn đề**

Ngày nay, xã hội ngày càng phát triển, dân số ngày càng tăng dẫn tới nhu cầu việc làm ngày càng lớn. Ngày trước khi muốn tìm việc ta phải đến tận công ty và hỏi thông tin tuyển dụng rất là phiền phức và tốn thời gian, ngoài ra còn gặp nhiều vấn đề nữa khi mà người tìm việc không biết công ty nào tuyển người để mà ứng tuyển.

Cùng lúc đó, theo khảo sát, có khoảng 60% các doanh nghiệp thiếu nhân lực, và trong nghành IT hiện nay thì con số đó là 87%. Và cũng như những người đi tìm việc làm, doanh nghiệp cũng rất khó khan trong việc tìm kiếm nhân lực.

Vì thế, để kết nối được doanh nghiệp và những người cần việc làm nên người thực hiện đề tài quyết định chọn đề tài website tìm việc làm có tên là JobFinder sử dụng công nghệ ReactJS để lấy ưu thế về tốc độ frontend, Java Spring cho một nền móng backend vững trãi và cuối cùng là Restful Webservice để việc thực hiện dự án lin động hơn

1. **Mục tiêu và nhiệm vụ của đề tài**
   1. **Mục tiêu**

Ứng dụng công nghệ mới như ReactJs, Restful, Spring boot vào quy trình phát triển một website toàn diện.

* 1. **Nhiệm vụ**

Một số nhiệm vụ chính mà nhóm em cần thực hiện để có thể hoàn thành môn học này:

* Nghiên cứu công nghệ mới: Spring boot, ReactJs, Restful Webservice
* Phát triển ứng dụng: Website tìm kiếm việc làm

1. **Phương pháp nghiên cứu**

Đối với client: Hiện thực front – end bằng ReactJS.

Đối với Server: Xây dựng các service và API thông qua Spring Boot.

Database: Tạo database ảo trên Spring Boot.

1. **Bố cục của báo cáo**

Ngoài các mục nói trên của báo cáo thì còn có các chương sau:

Chương 1: Các công nghệ sử dụng

Chương 2: Khảo sát hiện trạng và xác định yêu cầu

Khảo sát nghiệp vụ

Xác định yêu cầu

Chương 3: Thiết kế phần mềm

Thiết kế hệ thống

Thiết kế dữ liệu

Thiết kế giao diện

Chương 4: Cài đặt và kiểm thử

# **PHẦN NỘI DUNG**

# **CHƯƠNG 1: CÔNG NGHỆ SỬ DỤNG**

1. **SPRING BOOT**
   1. **Những kiến thức cần chuẩn bị** 
      1. **Sơ lược về ngôn ngữ lập trình Java [1]**

Java (phiên âm Tiếng Việt: "Gia-va") là một ngôn ngữ lập trình hướng đối tượng (OOP) và dựa trên các lớp (class). Khác với phần lớn ngôn ngữ lập trình thông thường, thay vì biên dịch mã nguồn thành mã máy hoặc thông dịch mã nguồn khi chạy, Java được thiết kế để biên dịch mã nguồn thành bytecode, bytecode sau đó sẽ được môi trường thực thi (runtime environment) chạy.

Có 6 mục tiêu chính trong việc xây dựng ngôn ngữ Java:

Đơn giản, hướng đối tượng và quen thuộc.

Mạnh mẽ và an toàn.

Kiến trúc trung lập và di động.

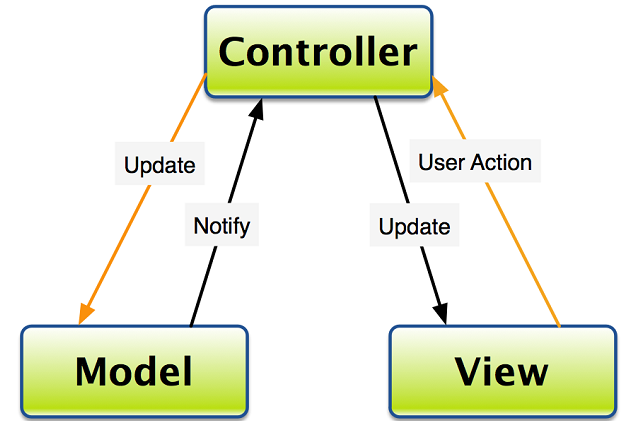
Thực thi với hiệu suất cao.

Dịch ra bytecode, phân luồng và năng động.

Dễ sử dụng cho người dùng Java

* + 1. **Sơ lược về mô hình MVC [2]**

MVC là viết tắt của Model – View – Control. Là một mô hình thiết kế hay kiến trúc phần mềm được sử dụng trong kỹ thuật phần mềm, nói một cách đơn giản đó là mô hình phân bổ sourcode thành 3 thành phần chính, mỗi thành phần có một nhiệm vụ riêng biệt và xử lý độc lập với các thành phần khác.



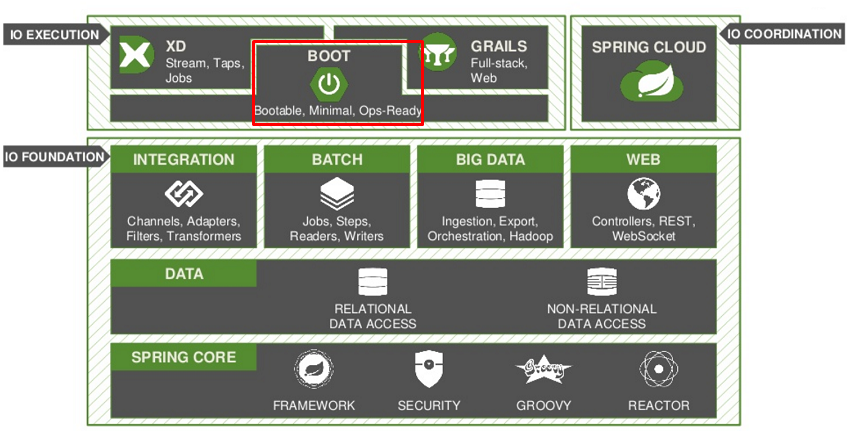
Hình 1: Mô hình MVC

Chú thích:

* Model: Đây là thành phần chứa tất cả các nghiệp vụ logic, phương thức xử lý, truy xuất database, đối tượng mô tả dữ liệu như các hàm, class xử lý.
* View: đảm nhận việc hiển thị thông tin trang, tương tác với người dùng, nơi chứa tất cả các đối tượng GUI như images, textbox. Nói dễ hiểu đó là tập hợp các file HTML và các form.
* Control: có nhiệm vụ điều hướng các yêu cầu từ người dùng và gọi đúng các phương thức xử lý chúng, ví dụ như nhận request từ các form và Url để thao tác trực tiếp với thành phần Model.
  1. **Giới thiệu về Spring boot [3]**
* Spring Boot là một module của Spring Framework, cung cấp tính năng RAD (Rapid Application Development) – Phát triển ứng dụng nhanh.
* Spring Boot được dùng để tạo các ứng dụng độc lập dựa trên Spring.
* Spring Boot không yêu cầu cấu hình XML
* Nó là một chuẩn cho cấu hình thiết kế phần mềm, tăng cao năng suất cho developer.
* Spring là một mã nguồn mở, được phát triển, chia sẻ và có cộng đồng người dùng rất lớn
* Spring Framework được xây dựng dựa trên 2 nguyên tắc design chính là: Dependency Injection và Aspect Oriented Programming.
* Những tính năng cốt lõi của Spring có thể được sử dụng để phát triển Java Desktop, ứng dụng mobile, Java Web. Mục tiêu chính của Spring là giúp phát triển các ứng dụng J2EE một cách dễ dàng hơn dựa trên mô hình sử dụng POJO.



Hình 2: Spring Framework Runtime



Hình 3: Spring Features

* 1. **Ưu điểm của Spring boot**
* Có các tính năng của Spring Framework.
* Tạo ứng dụng độc lập, có thể chạy bằng java -jar (cho cả java web)
* Nhúng trực tiếp các ứng dụng server (Tomcat, Jetty…) do đó không cần phải triển khai file WAR
* Cấu hình ít, tự động cậu hình bất kì khi nào có thể (Giảm thời gian viết code, tăng năng suất)
* Không yêu cầu XML config…
* Cung cấp nhiều plugin
* Chuẩn cho Microservices (Cloud support; giảm việc setup, config; các thư viện hỗ trợ…)

1. **REACT JS [4]**
   1. **Đặt vấn đề**

React.js là một thư viện Javascript đang nổi lên trong những năm gần đây với xu hướng Single Page Application. Trong khi những framework khác cố gắng hướng đến một mô hình MVC hoàn thiện thì React nổi bật với sự đơn giản và dễ dàng phối hợp với những thư viện Javascript khác. Nếu như AngularJS là một Framework cho phép nhúng code javasscript trong code html thông qua các attribute như ng-model, ng-repeat...thì với react là một library cho phép nhúng code html trong code javascript nhờ vào JSX, bạn có thể dễ dàng lồng các đoạn HTML vào trong JS.Tích hợp giữa javascript và HTML vào trong JSX làm cho các component dễ hiểu hơn

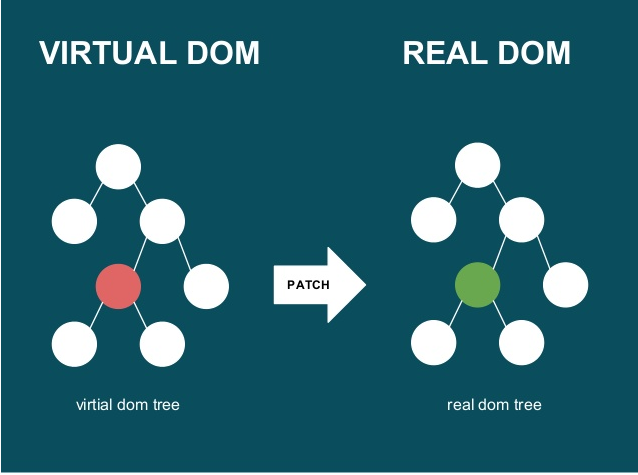
* 1. **Giới thiệu ReactJs**

React là một thư viện UI phát triển tại Facebook để hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái và có thể sử dụng lại được. React được sử dụng tại Facebook trong production, và www.instagram.com được viết hoàn toàn trên React.

Một trong những điểm hấp dẫn của React là thư viện này không chỉ hoạt động trên phía client, mà còn được render trên server và có thể kết nối với nhau. React so sánh sự thay đổi giữa các giá trị của lần render này với lần render trước và cập nhật ít thay đổi nhất trên DOM. Trươc khi đến cài đặt và cấu hình, chúng ta sẽ đi đến một số khái niệm cơ bản:

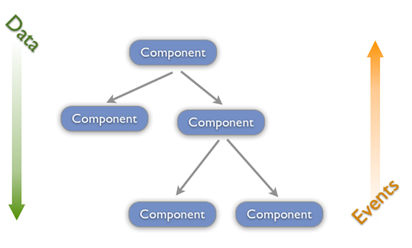
* + 1. **Virtual Dom**

công nghệ DOM ảo giúp tăng hiệu năng cho ứng dụng. Việc chỉ node gốc mới có trạng thái và khi nó thay đổi sẽ tái cấu trúc lại toàn bộ, đồng nghĩa với việc DOM tree cũng sẽ phải thay đổi một phần, điều này sẽ ảnh hưởng đến tốc độ xử lý. React JS sử dụng Virtual DOM (DOM ảo) để cải thiện vấn đề này.Virtual DOM là một object Javascript, mỗi object chứa đầy đủ thông tin cần thiết để tạo ra một DOM, khi dữ liệu thay đổi nó sẽ tính toán sự thay đổi giữa object và tree thật, điều này sẽ giúp tối ưu hoá việc re-render DOM tree thật.



Hình 4: Vitual Dom

React sử dụng cơ chế one-way data binding – luồng dữ liệu 1 chiều. Dữ liệu được truyền từ parent đến child thông qua props. Luồng dữ liệu đơn giản giúp chúng ta dễ dàng kiểm soát cũng như sửa lỗi.



Hình 5: React Component tree

Với các đặc điểm ở trên, React dùng để xây dựng các ứng dụng lớn mà dữ liệu của chúng thay đổi liên tục theo thời gian. Dữ liệu thay đổi thì hầu hết kèm theo sự thay đổi về giao diện. Ví dụ như Facebook: trên Newsfeed của bạn cùng lúc sẽ có các status khác nhau và mỗi status lại có số like, share, comment liên tục thay đổi. Khi đó React sẽ rất hữu ích để sử dụng.

* + 1. **JSX**

JSX là một dạng ngôn ngữ cho phép viết các mã HTML trong Javascript. Đặc điểm: Faster: Nhanh hơn. JSX thực hiện tối ưu hóa trong khi biên dịch sang mã Javacsript. Các mã này cho thời gian thực hiện nhanh hơn nhiều so với một mã tương đương viết trực tiếp bằng Javascript. Safer: an toàn hơn. Ngược với Javascript, JSX là kiểu statically-typed, nghĩa là nó được biên dịch trước khi chạy, giống như Java, C++. Vì thế các lỗi sẽ được phát hiện ngay trong quá trình biên dịch. Ngoài ra, nó cũng cung cấp tính năng gỡ lỗi khi biên dịch rất tốt. Easier: Dễ dàng hơn. JSX kế thừa dựa trên Javascript, vì vậy rất dễ dàng để cho các lập trình viên Javascripts có thể sử dụng

* + 1. **Component**

React được xây dựng xung quanh các component, chứ không dùng template như các framework khác. Trong React, chúng ta xây dựng trang web sử dụng những thành phần (component) nhỏ. Chúng ta có thể tái sử dụng một component ở nhiều nơi, với các trạng thái hoặc các thuộc tính khác nhau, trong một component lại có thể chứa thành phần khác. Mỗi component trong React có một trạng thái riêng, có thể thay đổi, và React sẽ thực hiện cập nhật component dựa trên những thay đổi của trạng thái. Mọi thứ React đều là component. Chúng giúp bảo trì mã code khi làm việc với các dự án lớn. Một react component đơn giản chỉ cần một method render. Có rất nhiều methods khả dụng khác, nhưng render là method chủ đạo.

* + 1. **Props và State**

Props: giúp các component tương tác với nhau, component nhận input gọi là props, và trả thuộc tính mô tả những gì component con sẽ render. Prop là bất biến. State: thể hiện trạng thái của ứng dụng, khi state thay đồi thì component đồng thời render lại để cập nhật UI.

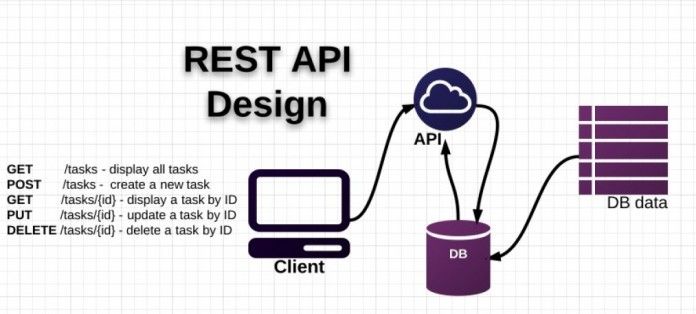
* 1. **Kết luận**

React là một thư viện rất thú vị và được phát triển dựa trên rất nhiều cấu trúc phức tạp. Tuy nhiên thư viện này lại rất dễ sử dụng và thêm vào trong nhiều application khác nhau. Các bạn có thể đọc thêm tại trang chủ của facebook: https://facebook.github.io/react/. Ngoài ra React còn có thư viện React Native được dùng để thiết kế native apps. Tiếp theo sang phần 2 chúng ta sẽ thực hành về cách cài đặt và cấu hình.

1. **RESTFUL API**
   1. **Giới thiệu**

Trong thế giới kết nối như bây giờ, một sản phẩm không thể đứng độc lập, và sản phẩm nào không có APIs, giống như máy tính không được kết nối Internet vậy.

Việc xây dựng một API là một trong những điều quan trọng nhất bạn có thể làm để nâng cao giá trị dịch vụ của bạn. Bởi việc có một API thì dịch vụ hay ứng dụng của bạn mới có khả năng trở thành một nền tảng mà từ đó các dịch vụ khác sẽ phát triển theo. Hãy cùng xem những công ty hiện tại như: Facebook, Twitter, Google, GitHub, Amazon, Netflix…, sẽ không có một công ty nào có thể lớn mạnh như ngày nay nếu như họ không cung cấp mở những dữ liệu của họ thông qua API.



Hình 6: Rest API Design

* 1. **Restful, Restful API là gì [5]**

REST là viết tắt của Representational State Transfer. Giải thích đơn giản, REST là một loạt hướng dẫn và dạng cấu trúc dùng cho việc chuyển đổi dữ liệu. Thông thường, REST hay được dùng cho ứng dụng web, nhưng cũng có thể làm việc được với dữ liệu phần mềm.

API là viết tắt của Application Programming Interface, phương thức kết nối với các thư viện và ứng dụng khác. Windows có nhiều API, và Twitter cũng có web API, tuy chúng thực hiện các chức năng khác nhau, với mục tiêu khác nhau.

RESTful API là những API đi theo cấu trúc REST.

* 1. **Kiến trúc của Rest [6]**
     1. **Client–server architecture**

Hệ thống hoạt động theo mô hình client-server, trong đó server là tập hợp các service nhỏ lắng nghe các request từ client. Với từng request khác nhau thì có thể một hoặc nhiều service xử lý.

* + 1. **Statelessness**

Stateless (phi trạng thái). Đơn giản server và client không lưu trạng thái của nhau -> mỗi request lên server thì client phải đóng gói thông tin đầy đủ để thằng server hiểu được. Điều này giúp hệ thống của bạn dễ phát triển,bảo trì, mở rộng vì không cần tốn công CRUD trạng thái của client . Hệ thống phát triển theo hướng này có ưu điểm nhưng cũng có khuyết điểm là gia tăng lượng thông tin cần truyền tải giữa client và server.

* + 1. **Cacheability**

Các response có thể lấy ra từ cache. Bằng cách cache các response , server giảm tải việc xử lý request, còn client cũng nhận được thông tin nhanh hơn. Ở đây ta đặt 1 thằng cache vào giữa : client- cache- server.

* + 1. **Layered system**

Phân lớp hệ thống : trong hệ thống REST chia tách các thành phần hệ thống theo từng lớp, mỗi lớp chỉ sử dụng lớp ở dưới nó và giao tiếp với lớp ở ngay trên nó mà thôi. Điều này giúp giảm độ phức tạp của hệ thống, giúp các thành phần tách biệt nhau từ đó dễ dàng mở rộng từng thành phần.

* + 1. **Uniform interface**

Chuẩn hóa các interface : Đây là một trong những đặc tính quan trọng của hệ thống REST. Bằng cách tạo ra các quy ước chuẩn để giao tiếp giữa các thành phần trong hệ thống, đơn giản hóa việc client có thể tương tác với server. Các quy ước này áp dụng cho toàn bộ các service giúp cho người sử dụng hệ thống của bạn dễ dụng hơn. Dễ hiểu hơn trên hệ thống đặt ra 1 chuẩn API để người dùng dù là mobile, web đều có thể kết nối vào được. Hệ thống REST có yếu điểm ở đây vì khi chuẩn hóa rồi ta không thế tối ưu từng kết nối.

* 1. **Ưu điểm**

- Giúp cho ứng dụng trở nên rõ ràng hơn.

- REST URL đại diện cho resource chứ không phải là hành động.

- Dữ liệu được trả về với nhiều định dạng khác nhau như: xml, html, json …

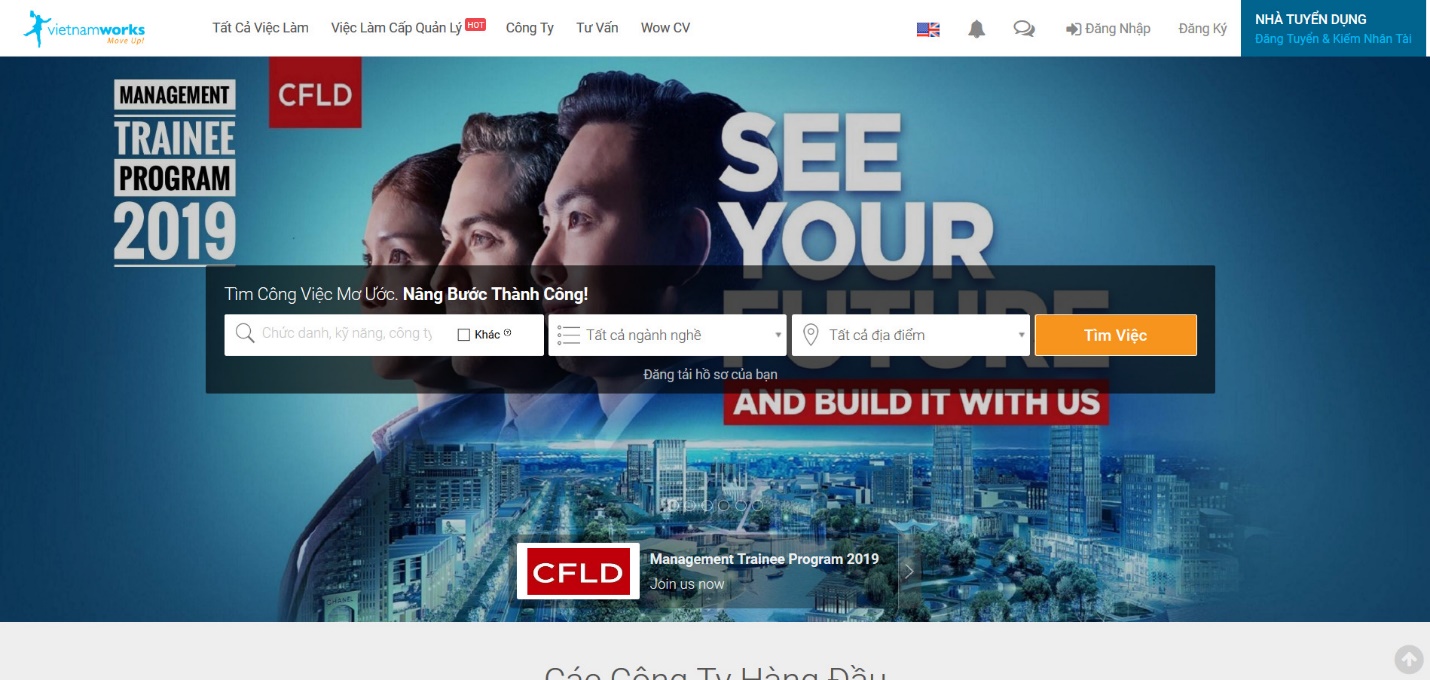
- Code đơn giản và ngắn gọn.

- REST chú trọng vào tài nguyên hệ thống.

# **CHƯƠNG 2: KHẢO SÁT HIỆN TRẠNG VÀ XÁC ĐỊNH YÊU CẦU**

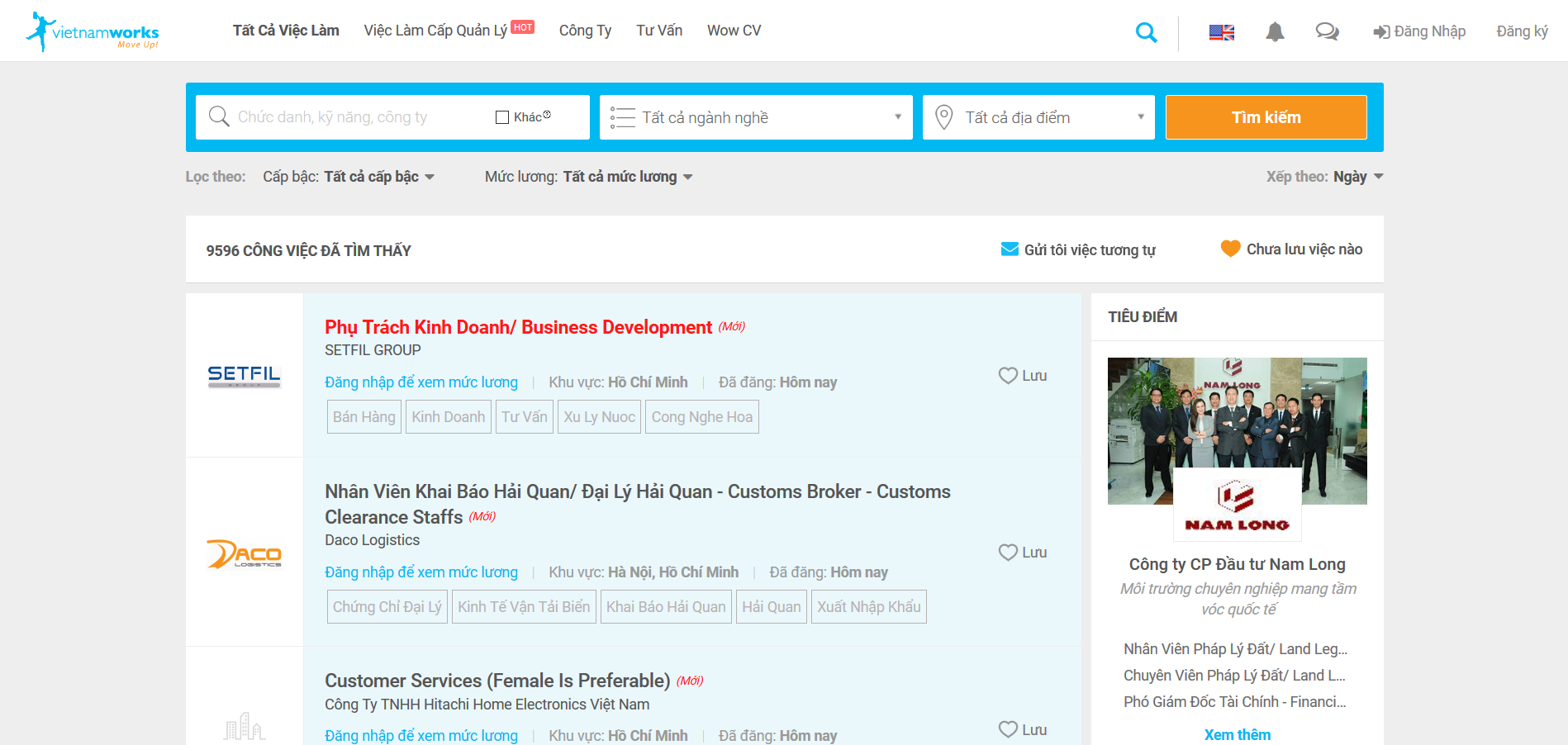
1. **Khảo sát nghiệp vụ**

Website được khảo sát nghiệp vụ sẽ là website vietnamwork.

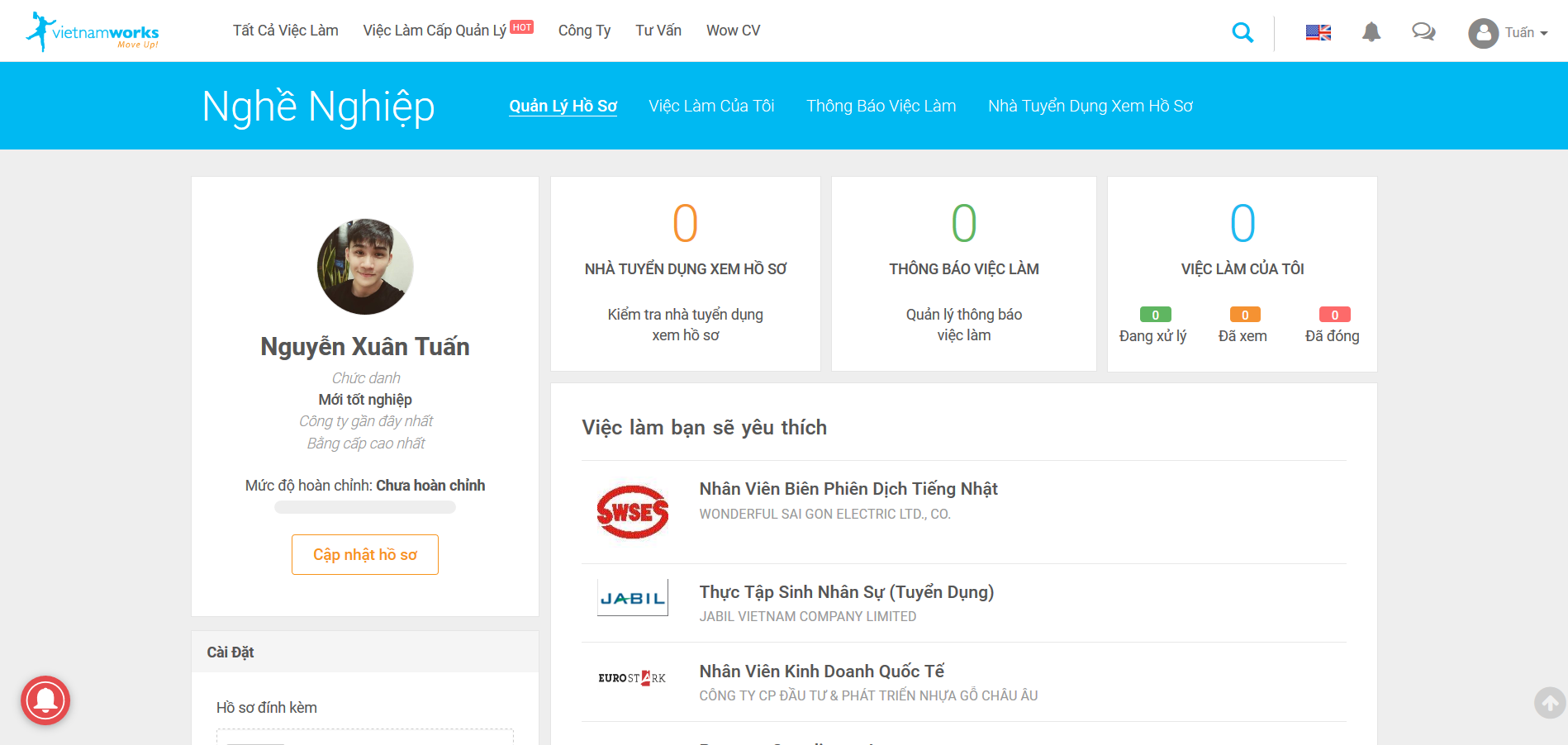
Hình 7: trang chủ website khảo sát

Website được khảo sát gồm 3 vai trò: admin, user, guest

Guest khi truy cập vào trang web sẽ có khả năng tìm việc làm (theo địa điểm, theo ngành nghề, … ) như hình dưới đây:

Hình 8: Trang danh sách công viêc website khảo sát

Sau khi đăng ký làm thành viên, ta có thể đăng nhập. Sau khi đăng nhập với tư cách user, Ta có thể quản lý thông tin hồ sơ của mình, 1 user có thể đồng thời là nhà tuyển dụng, hay là người tìm việc.

Hình 9: trang quản lý tài khoản website khảo sát

Khi đăng nhập với tư cách user ta có thể xem hồ sơ của mình và đi nhận những công việc được hiển thị trên web.

Khi đăng nhập với tư cách nhà tuyển dụng ta có thể xem hồ sơ của công ty mình và đăng công việc tìm kiếm người làm, xem những người đã apply vào công việc mình đăng

Cả 2 người có thể quản lý hợp đồng trong trang hồ sơ của mình.

Nhận xét trang web:

Ưu điểm

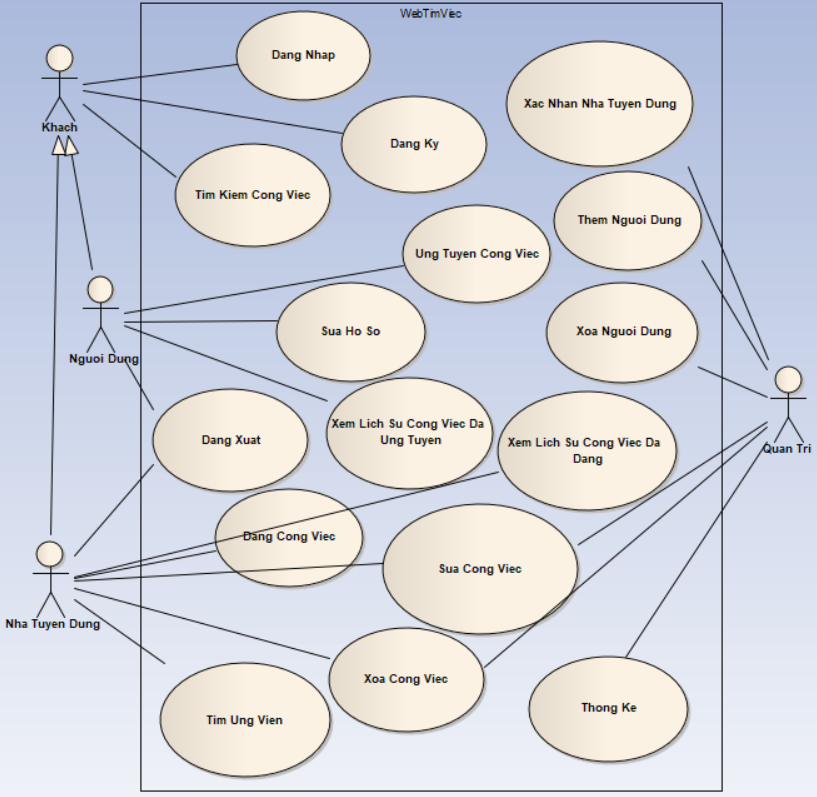
* Thiết kế giao diện web đẹp
* Chức năng tìm kiếm việc tiện lợi, tìm theo chuyên ngành, vị trí, …
* Có chức năng quản lý thông tin cá nhân của doanh nghiệp , người dùng
* Thủ tục nhận việc nhanh chóng
* Web responsive trên cả mobile và tablet
* Có chức năng tạo CV
* Đăng nhập được với tài khoản facebook hoặc google cực kỳ tiện lợi
* Có bộ phận tư vấn cho người dùng

Nhược điểm:

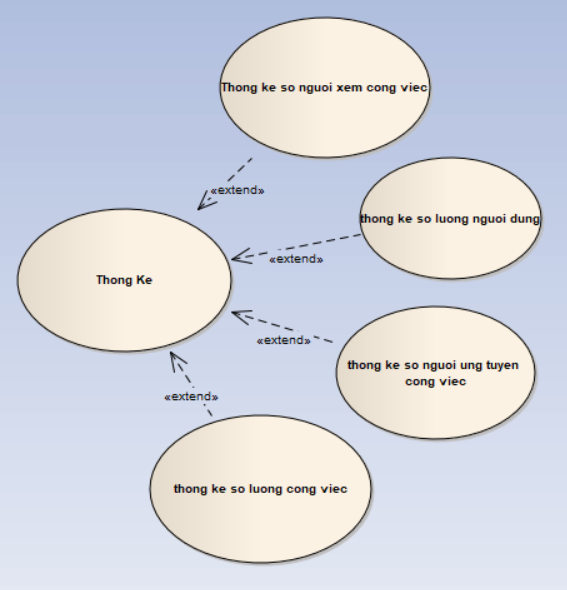
* Không có chức năng tìm kiếm công việc phù hợp dựa trên những kỹ năng người dùng.
* Trang web khá chậm, giảm trải nghiệm người dùng

1. **Xác định yêu cầu**

Lược đồ Usecase



Hình 10: use case diagram



Hình 11: Đặc tả chi tiết Usecase Thống Kê

|  |  |
| --- | --- |
| **Name** | **Đăng nhập** |
| **Brief description** | Người dùng đăng nhập |
| **Actor(s)** | admin, guest |
| **Pre-conditions** |  |
| **Post-conditions** | Tìm kiếm thành công sẽ hiện kết quả thành công, ngược lại sẽ thông báo thất bại |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng mở website.   1. Người dùng bấm vào đăng nhập 2. Người dùng chọn tư cách đăng nhập 3. Người dùng nhập email và mật khẩu 4. Nhấn đăng nhập |
| Alternative flow  (Thất bại) | Nếu thông tin đăng nhập không đúng sẽ thông báo thất bại và trở lại với tư cách guest |
| **Extension point** | Không có |

Bảng 1: đặc tả usecase đăng nhập

|  |  |
| --- | --- |
| **Name** | **Đăng xuất** |
| **Brief description** | Người dùng đăng xuất |
| **Actor(s)** | Admin, user, recruiter |
| **Pre-conditions** | Người dùng đã đăng nhập |
| **Post-conditions** | Đăng xuất thành công sẽ thông báo đăng xuất thành công và trở lại với tư cách guest |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng đã đăng nhập.   1. Người dùng bấm vào nút đăng xuất 2. Nhấn xác nhận để đăng xuất |
| Alternative flow  (Thất bại) | Nếu có lỗi kết nối trong quá trình xảy ra người dùng sẽ trở về tư cách guest kể cả khi chưa đăng xuất |
| **Extension point** | Không có |

Bảng 2: đặc tả usecase đăng xuất

|  |  |
| --- | --- |
| **Name** | **Đăng ký** |
| **Brief description** | Người dùng đăng ký thành viên |
| **Actor(s)** | guest |
| **Pre-conditions** |  |
| **Post-conditions** | Đăng ký thành công sẽ hiện kết quả thông báo thành công, ngược lại thông báo thất bại |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng mở website.   1. Người dùng bấm vào link đăng ký 2. Người chọn đăng ký với tư cách nào 3. Người dùng nhập thông tin đăng ký 4. Nhấn đăng ký để đăng ký thành viên |
| Alternative flow  (Thất bại) | Nếu thông tin đăng ký không hợp lệ sẽ trả về thông báo thất bại |
| **Extension point** | Không có |

Bảng 3: đặc tả usecase đăng ký

|  |  |
| --- | --- |
| **Name** | **Tìm kiếm công việc** |
| **Brief description** | Người dùng tìm kiếm công việc |
| **Actor(s)** | user, admin, guest |
| **Pre-conditions** |  |
| **Post-conditions** | Tìm kiếm thành công sẽ hiện kết quả tìm kiếm, ngược lại trả về danh sách trắng |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng mở website.   1. Người dùng bấm vào link danh sách công việc 2. Người dùng chọn các tiêu chí tùy theo cá nhân để lọc kết quả 3. Nhấn tìm kiếm để lọc kết quả |
| Alternative flow  (Thất bại) | Nếu không có kết quả phù hợp với tiêu chí người dùng đề ra thì sẽ trả về trang trắng |
| **Extension point** | Không có |

Bảng 4: đặc tả usecase Tìm kiếm công việc

|  |  |
| --- | --- |
| **Name** | **Đăng công việc** |
| **Brief description** | Nhà tuyển dụng đăng công việc |
| **Actor(s)** | recruiter |
| **Pre-conditions** | Đăng nhập với tư cách nhà tuyển dụng và tài khoản nhà tuyển dụng phải được admin xác nhận |
| **Post-conditions** | Đăng thành công sẽ có thông báo kết quả thành công, ngược lại thông báo thất bại |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng đăng nhập với tư cách recruiter   1. Người dùng bấm trang quản lý việc làm 2. Người dùng bấm thêm công việc 3. Người dùng nhập thông tin công việc 4. Nhấn Đăng để đăng công việc |
| Alternative flow  (Thất bại) | Nếu nhập thông tin không đúng yêu cầu thì hệ thống sẽ báo đăng việc thất bại |
| **Extension point** | Không có |

Bảng 5: đặc tả usecase Đăng công việc

|  |  |
| --- | --- |
| **Name** | **Sửa công việc** |
| **Brief description** | Nhà tuyển dụng sửacông việc |
| **Actor(s)** | recruiter |
| **Pre-conditions** | Đăng nhập với tư cách nhà tuyển dụng và tài khoản nhà tuyển dụng phải được admin xác nhận và nhà tuyển dụng đã đăng ít nhất 1 công việc |
| **Post-conditions** | Sửa thành công sẽ có thông báo kết quả thành công, ngược lại thông báo thất bại |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng đăng nhập với tư cách recruiter   1. Người dùng bấm trang quản lý việc làm 2. Người dùng bấm vào danh sách công việc 3. Người dùng bấm sửa 4. Người dùng sửa thông tin công việc 5. Nhấn Lưu để lưu sửa công việc |
| Alternative flow  (Thất bại) | Nếu thông tin sau khi sửa không đúng yêu cầu thì hệ thống sẽ báo sửa việc thất bại và thông tin việc đó sẽ trở về thời điểm trước khi sửa. |
| **Extension point** | Không có |

Bảng 6: đặc tả usecase sửa công việc

|  |  |
| --- | --- |
| **Name** | **Xóa công việc** |
| **Brief description** | Nhà tuyển dụng xóa công việc |
| **Actor(s)** | recruiter |
| **Pre-conditions** | Đăng nhập với tư cách nhà tuyển dụng và tài khoản nhà tuyển dụng phải được admin xác nhận và nhà tuyển dụng đã đăng ít nhất 1 công việc |
| **Post-conditions** | Xóa thành công sẽ có thông báo kết quả thành công, ngược lại thông báo thất bại |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng đăng nhập với tư cách recruiter   1. Người dùng bấm trang quản lý việc làm 2. Người dùng bấm vào danh sách công việc 3. Người dùng bấm xóa 4. Người dùng xác nhận muốn xóa |
| Alternative flow  (Thất bại) | Nếu có quá trình lỗi kết nối xảy ra thì xóa sẽ thất bại và trở về thời điểm chưa xóa |
| **Extension point** | Không có |

Bảng 7: đặc tả usecase xóa công việc

|  |  |
| --- | --- |
| **Name** | **ứng tuyển công việc** |
| **Brief description** | Người dùng ứng tuyển công việc |
| **Actor(s)** | User |
| **Pre-conditions** | Đăng nhập vơi tư cách user |
| **Post-conditions** | Nhận việc thành công sẽ thông báo thành công và công việc sẽ xuất hiện trong danh sách công việc đã ứng tuyển của user |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng đăng nhập với tư cách user và tìm được việc muốn nhận   1. Người dùng bấm vào nút apply 2. Người dùng điền thông tin được yêu cầu 3. Nhấn Xác nhận sẽ nhận công việc |
| Alternative flow  (Thất bại) | Nếu nhà tuyển dụng đã xóa công việc trong lúc user apply thì quá trình apply sẽ bị hủy bỏ và thông báo thất bại |
| **Extension point** | Không có |

Bảng 8: đặc tả usecase ứng tuyển công việc

|  |  |
| --- | --- |
| **Name** | **Sửa thông tin hồ sơ** |
| **Brief description** | Người dùng sửa thông tin hồ sơ |
| **Actor(s)** | User, recruiter |
| **Pre-conditions** | Đăng nhập vơi tư cách user hoặc recruiter |
| **Post-conditions** | Sửa thành công sẽ thông báo thành công và ngược lại sẽ thông báo thất bại |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng đăng nhập   1. Người dùng vào trang quản lý tài khoản 2. Người dùng bấm vào basic info 3. Người dùng nhập thông tin muốn sửa 4. Nhấn Lưu đẻ lưu thông tin sửa |
| Alternative flow  (Thất bại) | Nếu thông tin sau khi sửa không hợp lệ sẽ trả về thông báo thất bại và thông tin trở về thời điểm chưa sửa |
| **Extension point** | Không có |

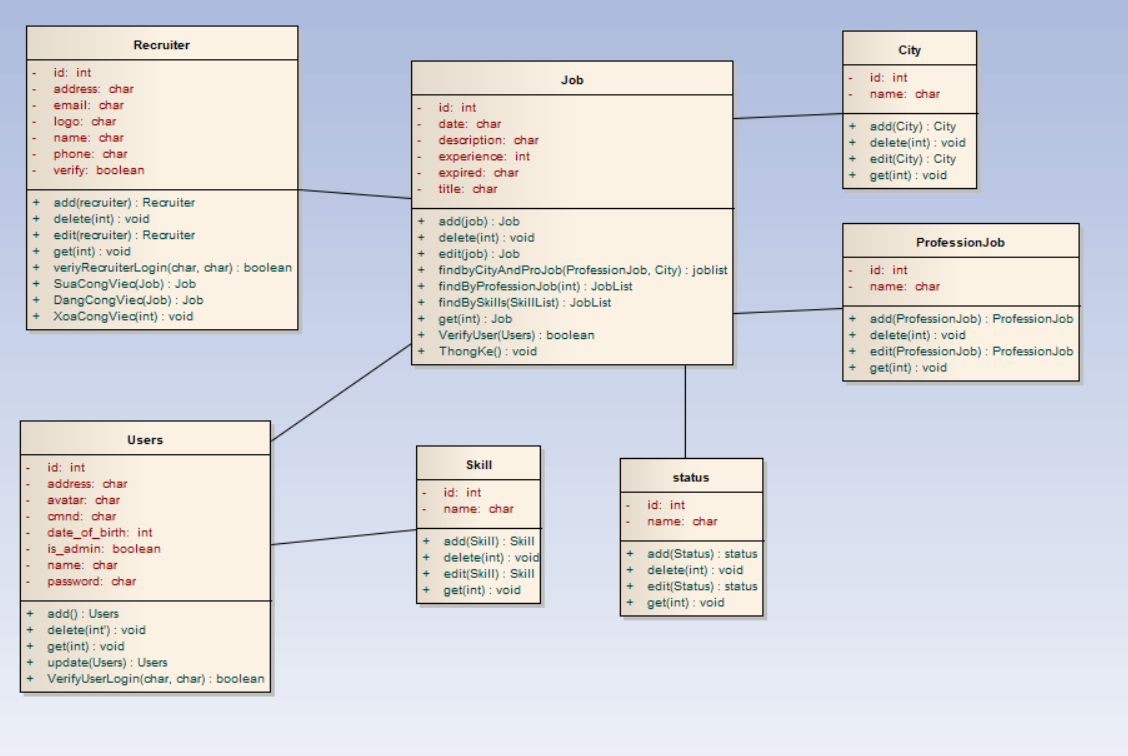
Bảng 9: đặc tả usecase sửa hồ sơ

|  |  |
| --- | --- |
| **Name** | **Xem lịch sử** |
| **Brief description** | Người dùng hoặc nhà tuyển dụng xem lịch sử những công việc đã đăng và ai đã ứng tuyển, người dùng xem lịch sử những công việc đã ứng tuyển |
| **Actor(s)** | User, Recruiter |
| **Pre-conditions** | Đăng nhập vơi tư cách user hoặc recruiter |
| **Post-conditions** | Xem lịch sử công việc |
| **Flow of events** |  |
| Basic flow  (Thành công) | Use case bắt đầu khi người dùng đăng nhập với tư cách user hoặc recruiter   1. Người dùng bấm trang quản lý tài khoản 2. Người dùng chọn mục lịch sử |
| Alternative flow  (Thất bại) | Nếu người dùng chưa ứng tuyển công việc nào hoặc nhà tuyển dụng chưa đăng công việc nào danh sách lịch sử sẽ trống |
| **Extension point** | Không có |

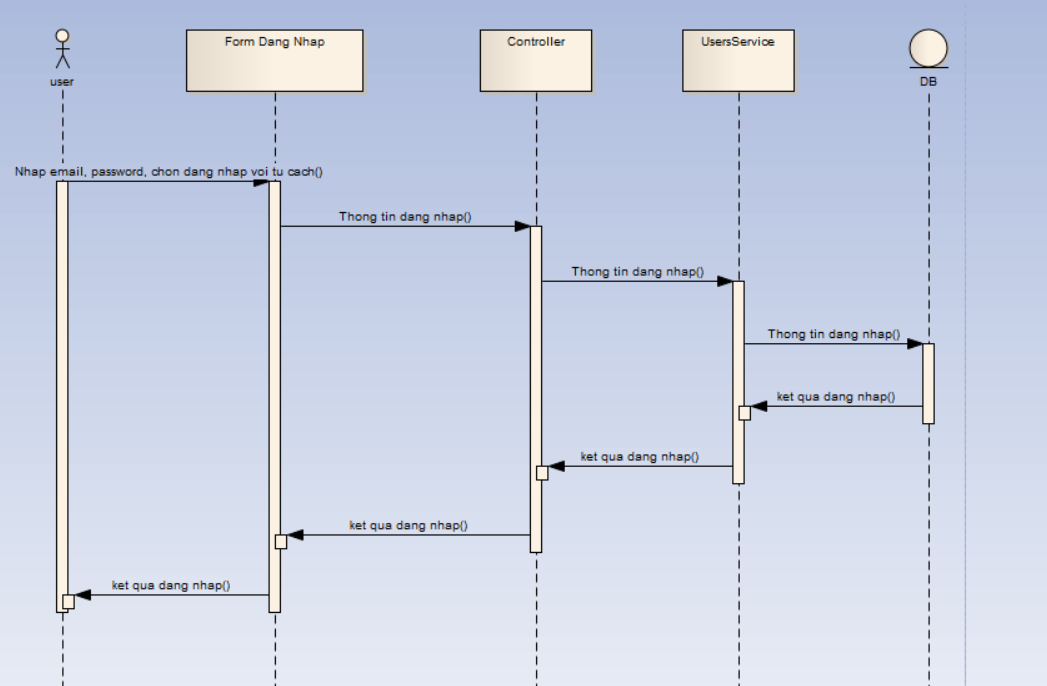
Bảng 10: đặc tả usecase xem lịch sử

# **CHƯƠNG 3: THIẾT KẾ TRANG WEB**

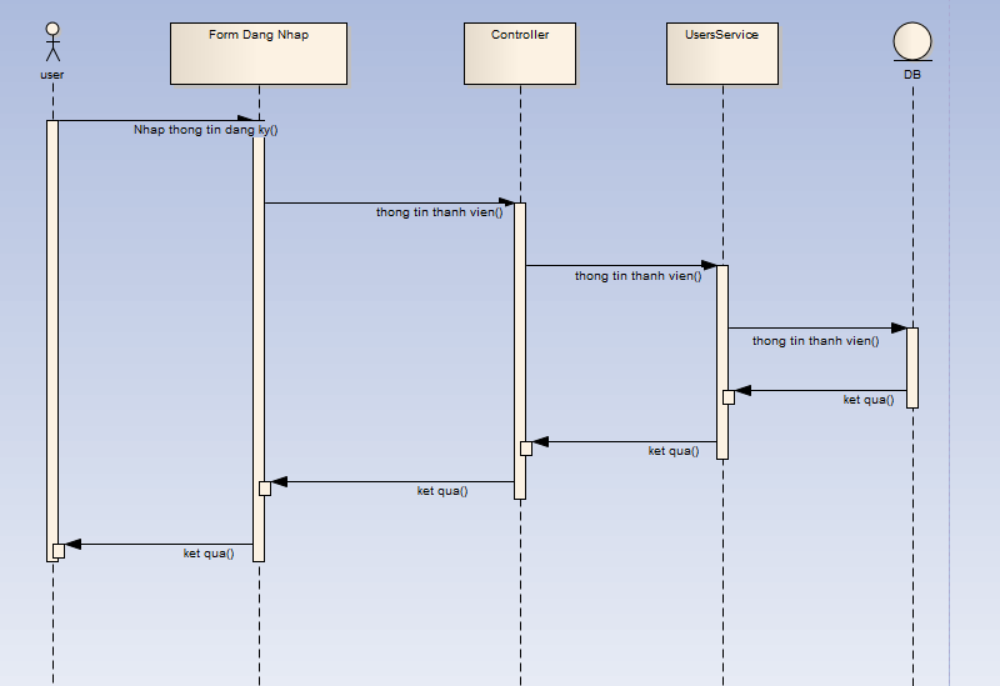
1. **Lược đồ lớp**

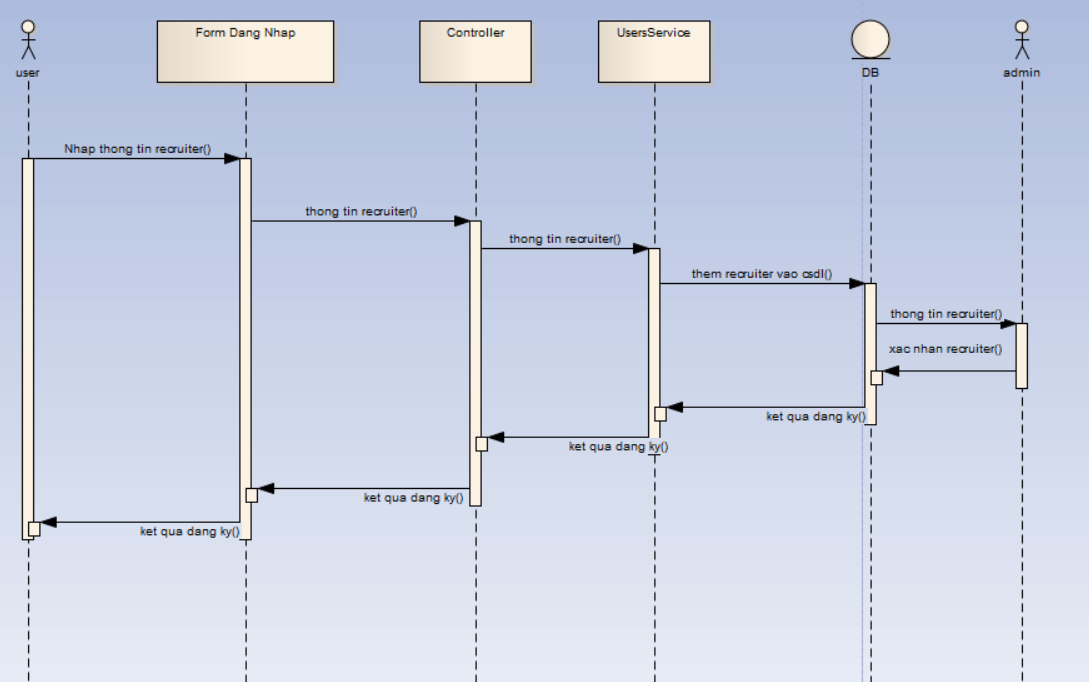
Hình 11: lược đồ lớp

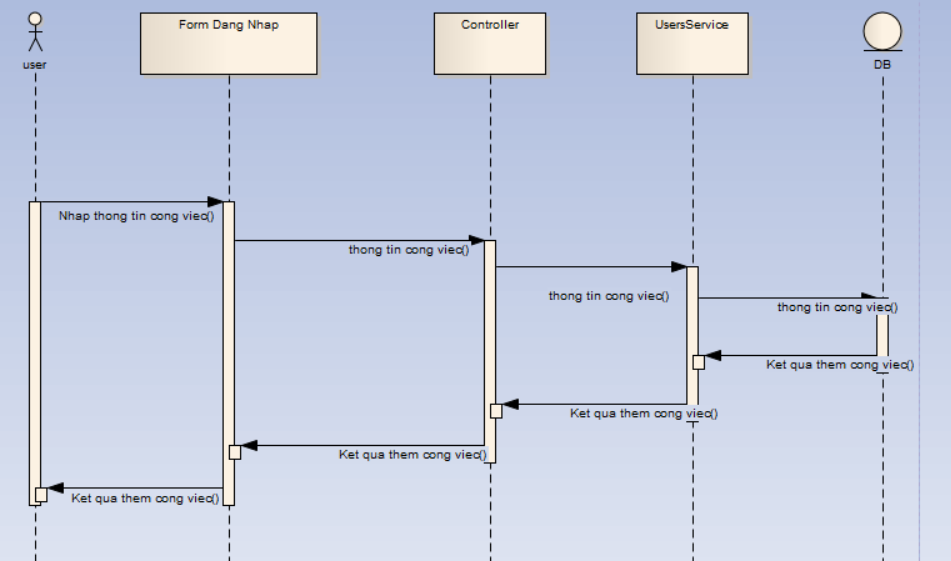
1. **Lược đồ Sequence**

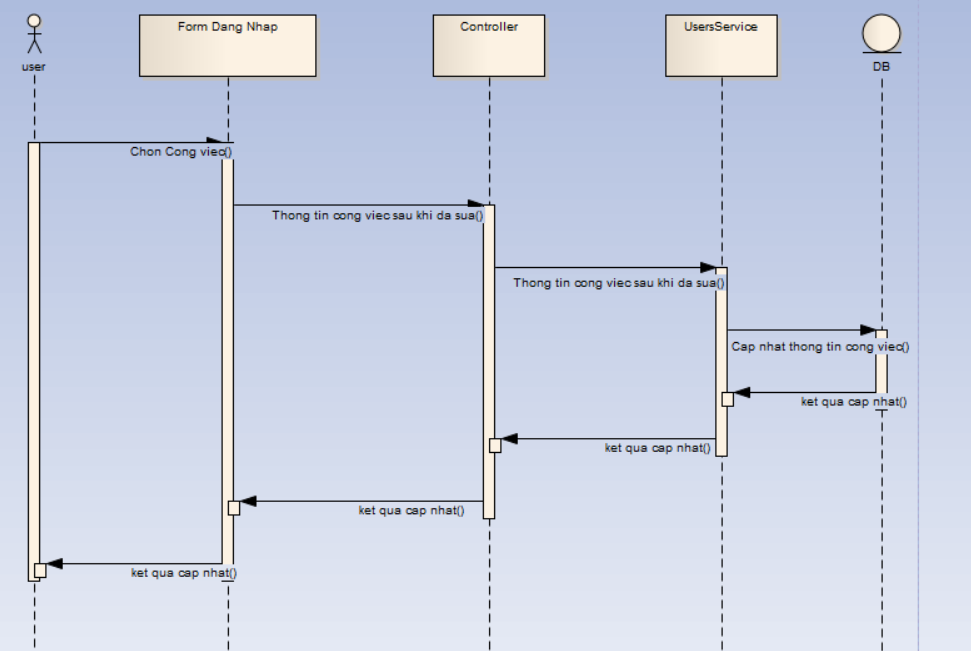


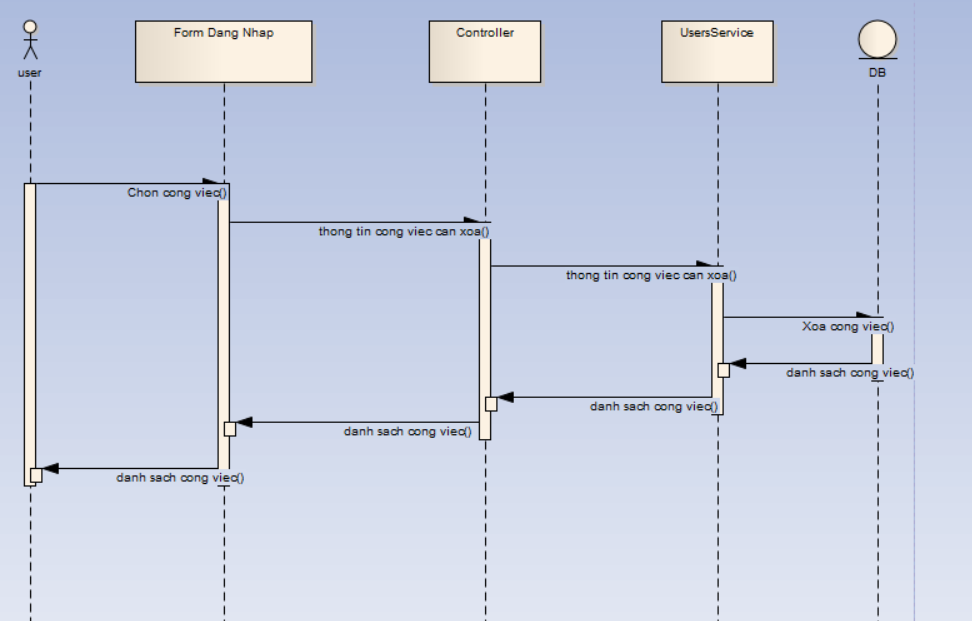
Hình 12: lược đồ đăng nhập

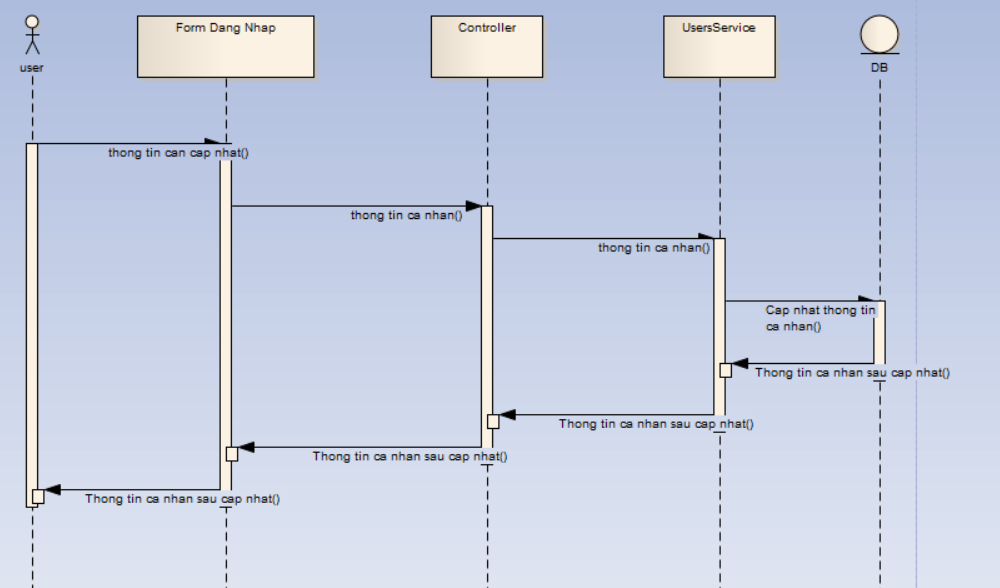
Hình 13: lược đồ Đăng Ký Thành Viên

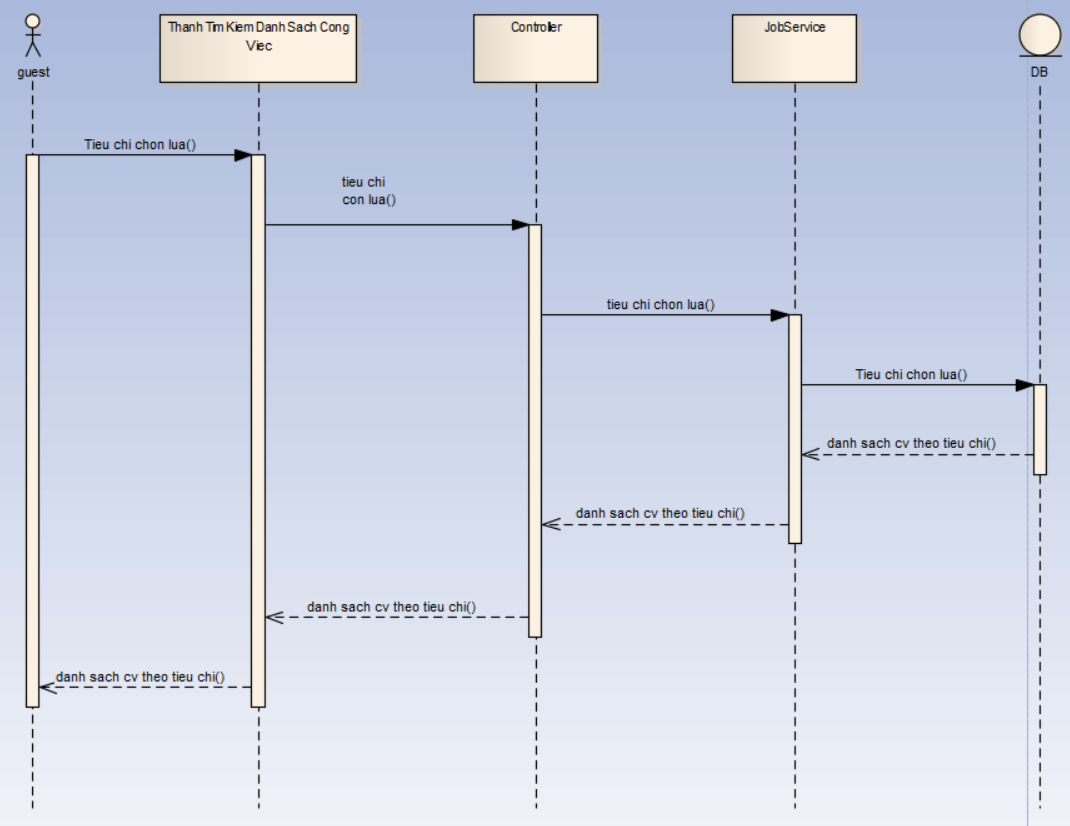
Hình 14: lược đồ Đăng Ký Tuyển Dụng

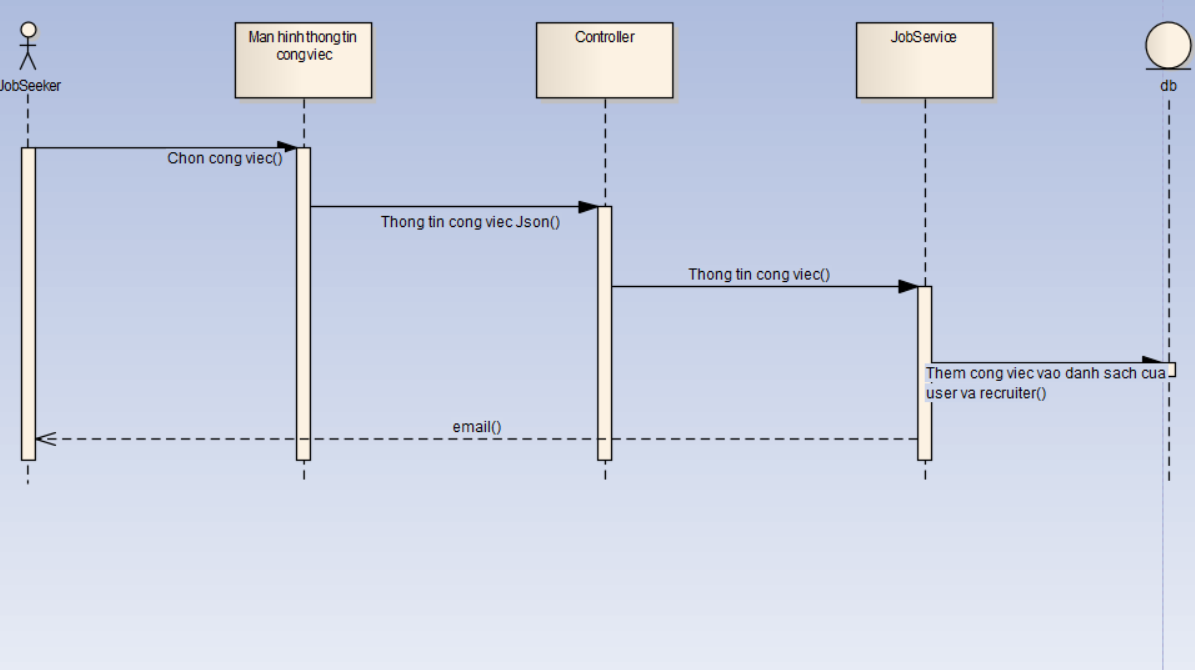
Hình 14: lược đồ Đăng Công Việc

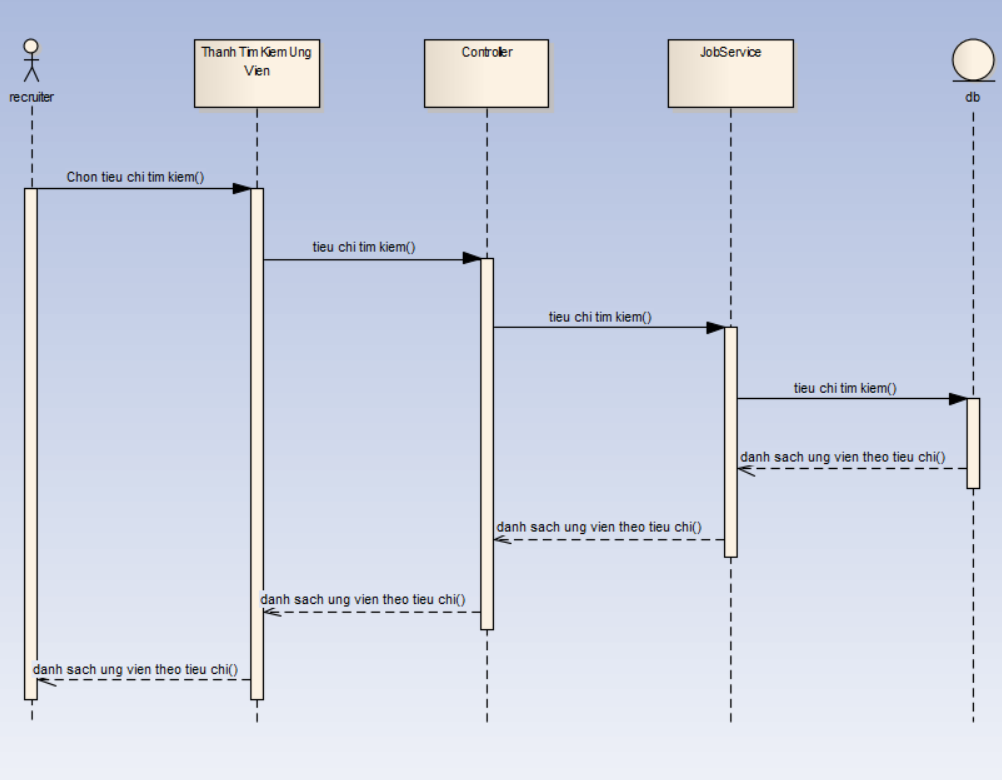
Hình 15: lược đồ Sửa Cộng Việc

Hình 16: lược đồ Xóa Công Việc

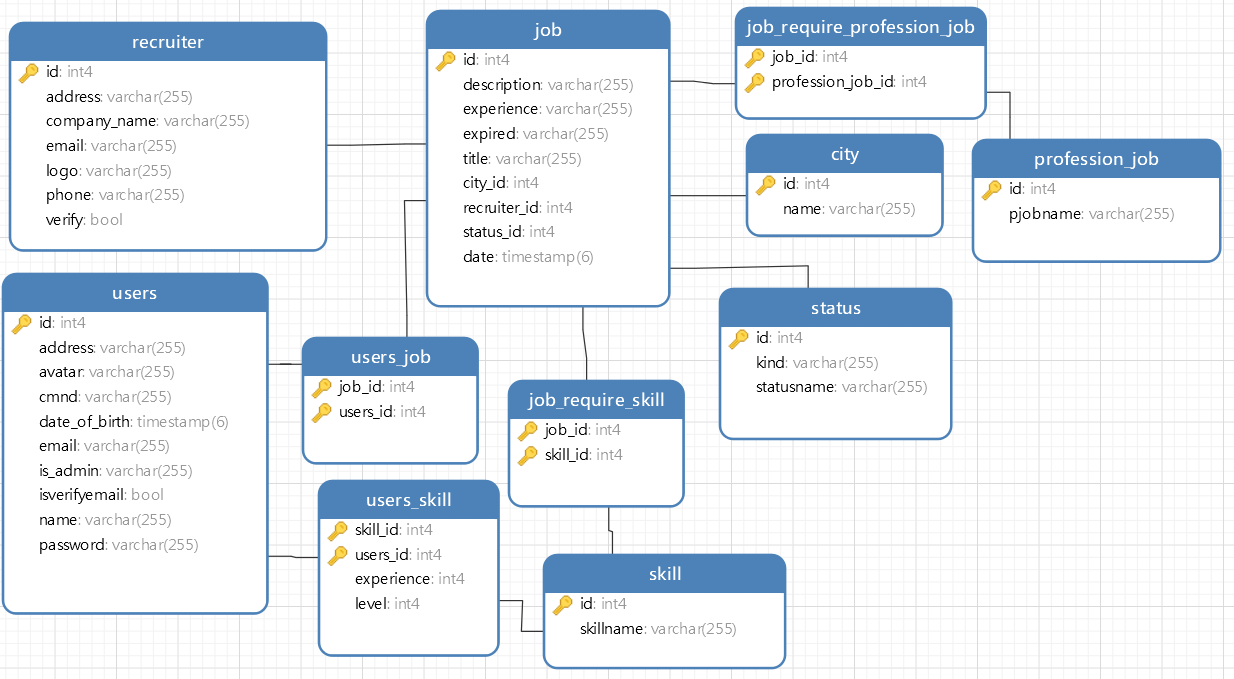
Hình 17: lược đồ Sửa Hồ Sơ

Hình 18: lược đồ Tìm kiếm công việc theo tiêu chí (chuyên ngành, vị trí)

Hình 19: lược đồ Nhận công việc

Hình 20: lược đồ tìm kiếm ứng viên

1. **Thiết kế CSDL**



Hình 21: Database Diagrams

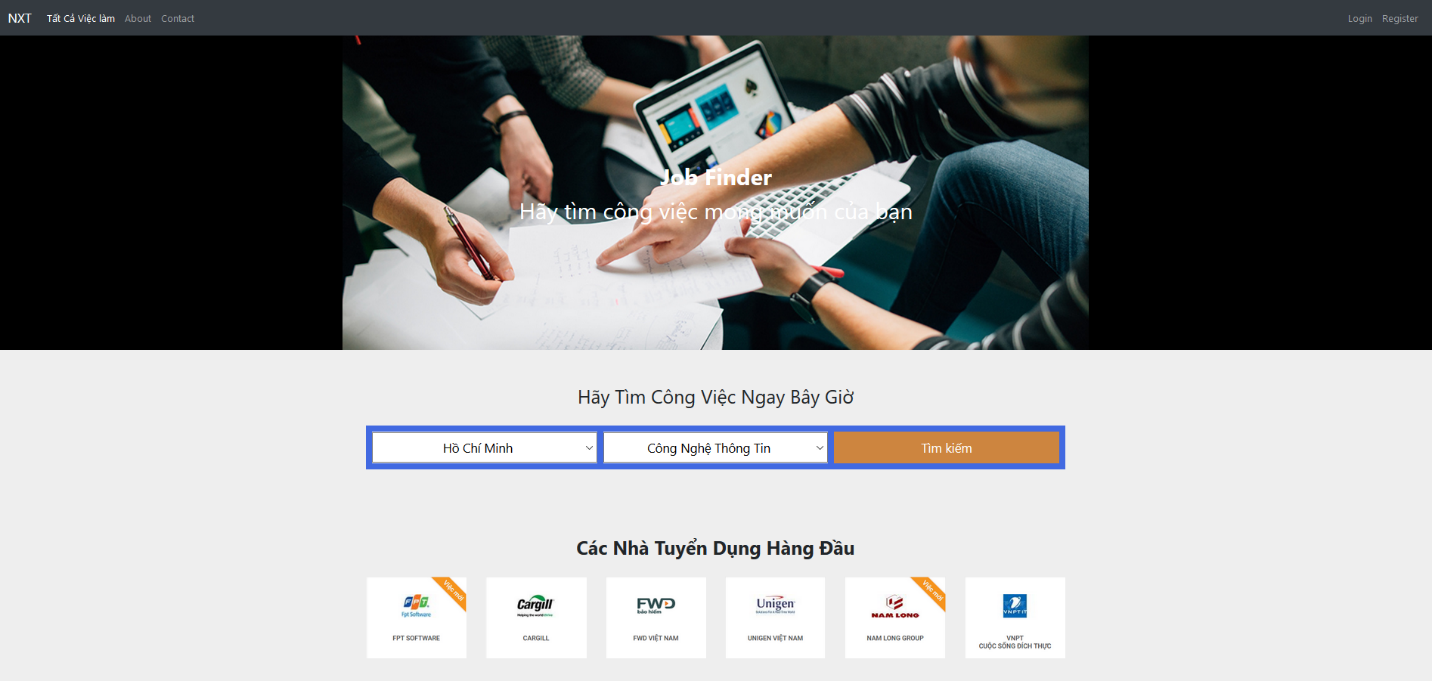
|  |  |  |
| --- | --- | --- |
| Tên Bảng | Cột | Ý nghĩa |
| Job | * id: int * description: varchar * title: varchar * experience: varchar * expired: varchar * date: timestamp | Chứa thông tin các công việc |
| Users | * id: int * address: varchar * cmnd: varchar * date\_of\_birth: timestamp * email: varchar * is\_admin: bool * isverifyemail: bool * name: varchar * password: varchar | Chứa thông tin các user |
| Recruiter | * id: int * company\_name: varchar * address: varchar * email: varchar * logo: varchar * phone: varchar * verify: bool | Chứa thông tin các nhà tuyển dụng |
| City | * id: int * name: varchar | Chứa thông tin vị trí thành phố |
| Status | * id: int * name: varchar | Chứa thông tin các trạng thái trong phần mềm |
| Professionjob | * id:int * name: varchar | Chứa thông tin các chuyên ngành. |
| Skill | * id: int * name: varchar | Chứa thông tin các kỹ năng |
| User\_job | * user\_id: int * job\_id: int | Chứa thông tin các user đã nhận việc nào, user đã nhận những việc nào |
| User\_skill | * user\_id: int * skill\_id: int | Chứa thông tin user có những kỹ năng nào, những user nào sở hữu kỹ năng đó |
| Job\_require\_skill | * job\_id: int * skill\_id: int | Chứa thông tin những kỹ năng mà công việc yêu cầu hay những công việc việc yêu cầu kỹ năng đó |
| Job\_require\_profession\_job | * job\_id: int * profession\_job\_id: int | Chứa thông tin những chuyên ngành liên quan tới công việc đó hay những công việc đó thuộc một chuyên ngành nào đó. |

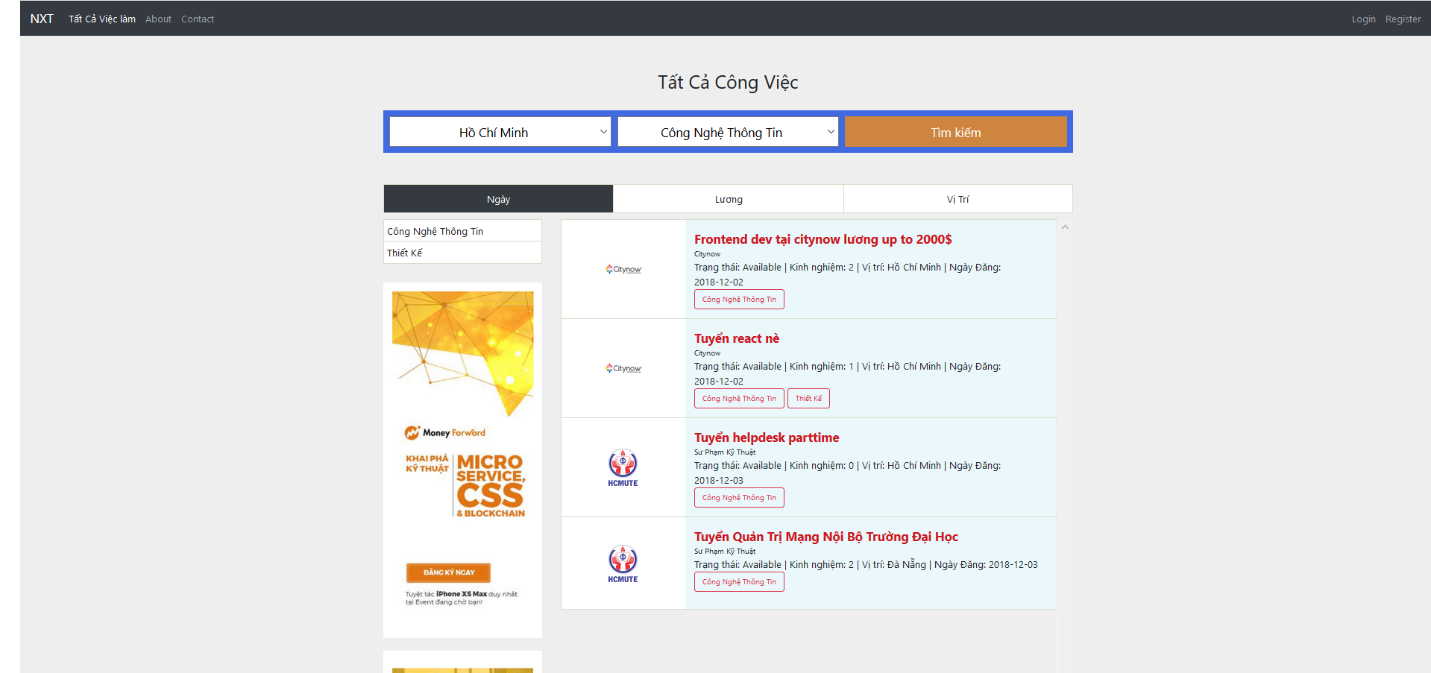
Bảng 11: bảng đặc tả database diagram

Ràng buộc toàn vẹn:

* Password cho user và recruiter phải có chữ Hoa và số
* Email phải có tên miền hợp lệ
* Một công việc phải có ít nhất một chuyên ngành
* Thông tin công việc phải có đầy đủ địa điểm, tiêu đề, mô tả công việc
* Một Recruiter phải có số điện thoại và địa chỉ
* Một Recruiter chỉ được phép đăng công việc khi đã được admin xác nhận tài khoản

1. **Thiêt kế giao diện**

Hình 21 : Giao diện Trang chủ

Hình 22 : Giao diện Trang chủ

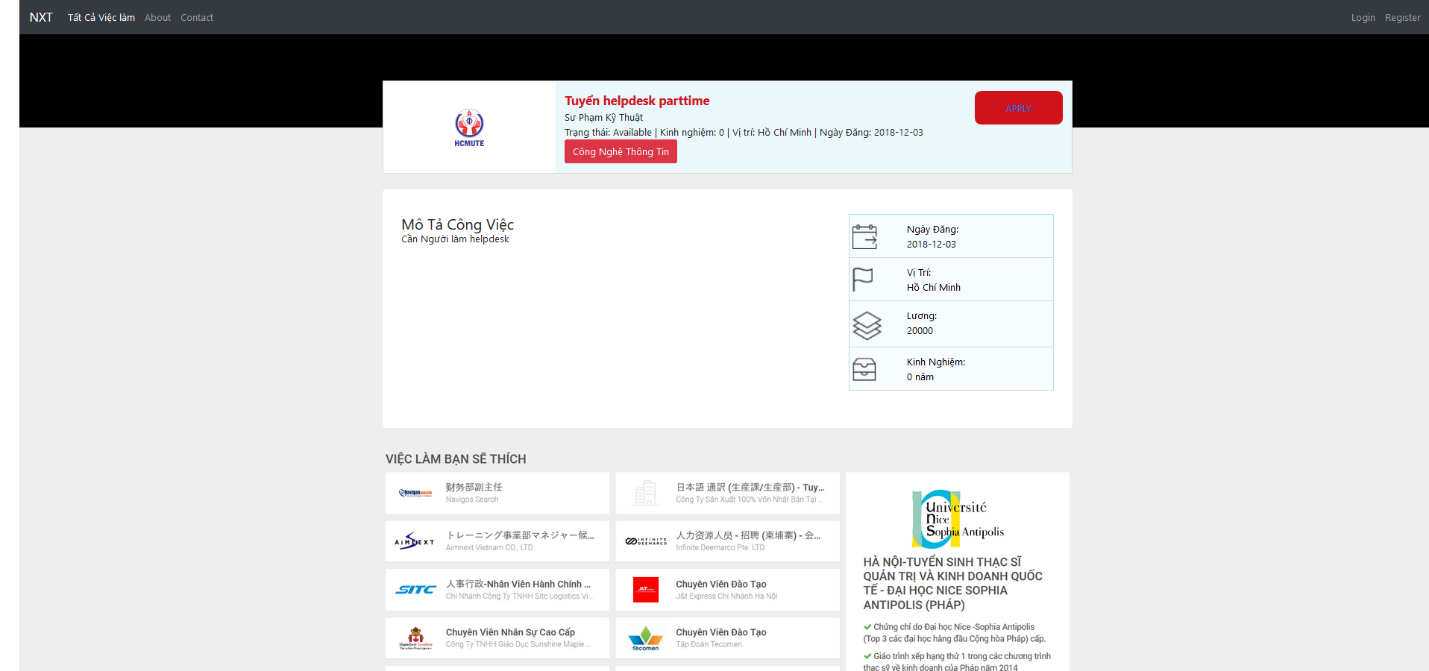
3

2

1

Mô tả chi tiết

* 1: Thanh tìm kiếm công việc
* 2: Bảng chuyên ngành
* 3: danh sách công việc

Hình 23: Giao diện Trang chủ

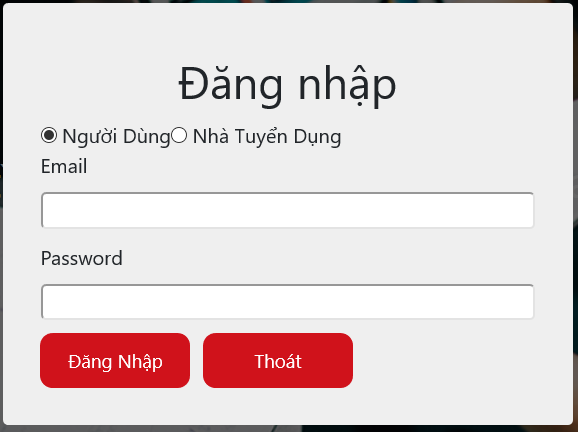
3

2

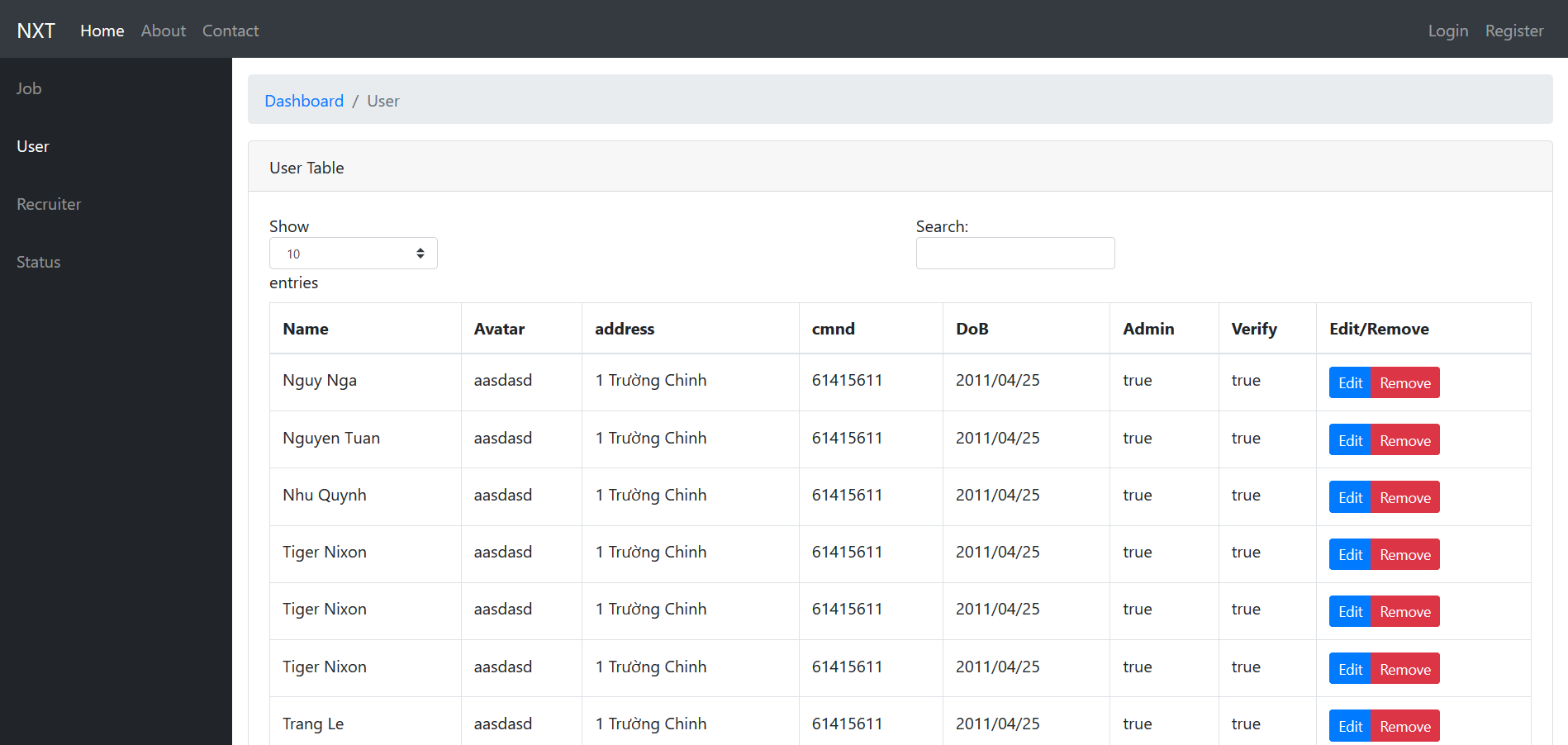
1

Mô tả chi tiết:

* 1: thông tin cơ bản công việc
* 2: thong tin mô tả công việc
* 3: Bảng thuộc tính công việc



Hình 24: Modal Đăng nhập



3

2

1

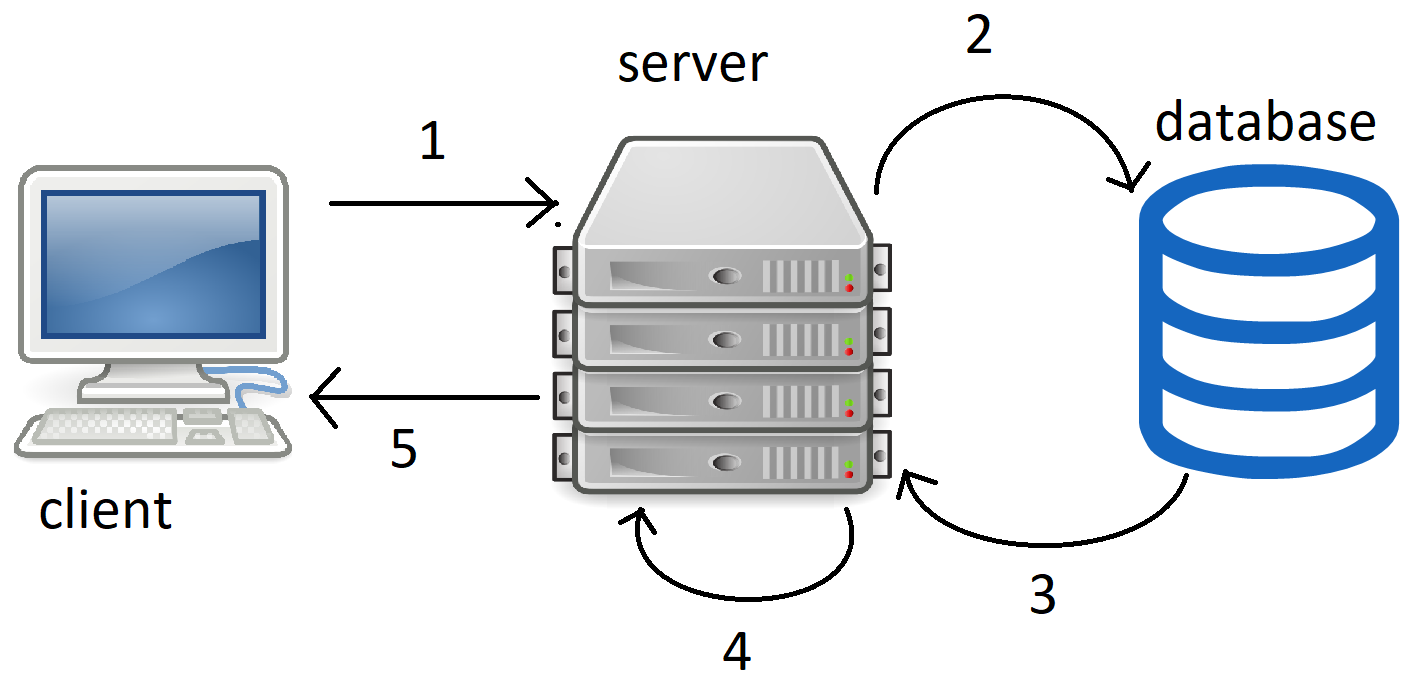
Hình 25: Trang quản trị

Mô tả chi tiết:

* 1: Các bảng dữ liệu
* 2: Khu vực dũ liệu
* 3: Khu vực các nút chỉnh sửa dữ liệu

# **CHƯƠNG 4: CÀI ĐẶT VÀ KIỂM THỬ**

1. **Cài Đặt**



Hình 25: kiến trúc website

Mô tả chi tiết:

1 client gửi request tới server

2 server yêu cầu dữ liệu từ database

3 database trả về dữ liệu cho server

4 server xử lý dữ liệu được trả về

5 server trả về dữ liệu cho client hiển thị lên cho người dùng

Hướng dẫn cài đặt:

* Cài đặt Java 7 trở lên kèm IntelliJ phiên bản trả phí để chạy server
* Cài đặt Node bản 6. trở lên
* Cài đặt Postgresql để lưu dữ liệu

Hướng dẫn chạy website:

B1: vào postgresql tạo CSDL có tên là webtimviec, thông tin truy cập vô postgres đặt là:

Username: postgres

Password: 123456

B1: dùng IntelliJ để mở thư mục backend, config là project spring boot và sau đó start

B2: Mở cmd và điều hướng vào thư mục frontend sau đó gõ lệnh npm start

1. **Kiểm Thử**

|  |  |  |
| --- | --- | --- |
| Tên Chức năng | Mô tả | Kết quả |
| Tìm kiếm công việc | Tìm kiếm tất cả công việc hoặc tìm công việc theo tiêu chí của người dùng | Pass |
| Đăng nhập | Người dùng đăng nhập vào website với tư cách người dùng hoặc nhà tuyển dụng | Pass |
| Đăng Xuất | Người dùng Đăng xuất website | Pass |
| Thêm Công Việc | Admin Thêm Công Việc | Pass |
| Sửa Công Việc | Admin Sửa Công Việc | Pass |
| Xóa Công Việc | Admin Xóa Công Việc | Pass |
| Thêm Người Dùng | Admin Thêm Người Dùng | Pass |
| Sửa Người Dùng | Admin Sửa Người Dùng | Pass |
| Xóa Người Dùng | Admin Xóa Người Dùng | Pass |
| Thêm Kỹ Năng | Admin Thêm Kỹ Năng | Pass |
| Sửa Kỹ Năng | Admin Sửa Kỹ Năng | Pass |
| Xóa Kỹ Năng | Admin Xóa Kỹ Năng | Pass |
| Thêm Địa Điểm | Admin Thêm Địa Điểm | Pass |
| Sửa Địa Điểm | Admin Sửa Địa Điểm | Pass |
| Xóa Địa Điểm | Admin Xóa Địa Điểm | Pass |

# **PHẦN KẾT LUẬN**

## **Kết quả đạt được và hướng phát triển**

* Về nghiên cứu công nghệ mới:
  + Nắm vững kiến thức về Springboot và cách cấu hình, xây dụng model, lâp trình theo hướng code first.
  + Nắm vững kiến thức cơ bản về ReactJs và các kiến thức liên quan như HTML, CSS, JavaScript.
  + Ứng dụng thành thạo mô hình MVC
  + Nắm cách thức giao tiếp giữa client và server bằng Restful.
* Về mặt ứng dụng web:
  + Thiết kế cơ sở dữ liệu tối ưu, chiếm ít bộ nhớ, gọn, nhẹ.
  + Tốc độ hiển thị dữ liệu nhanh chóng, thời gian loading thấp
  + Thiết kế website bằng HTML, CSS có giao diện bắt mắt
  + Các chức năng cơ bản của một website tìm việc làm đầy đủ, đặc biệt là chức năng tìm kiếm.

## **Hạn chế**

* Vì đề tài khá lớn và khoảng thời gian thực hiện đề tài khá ngắn cộng với việc thiếu nguồn nhân lực nên vẫn còn thiếu nhiều chức năng.

## **Hướng phát triển**

* Cá nhân hóa cho người dùng (hiển thị những công việc mà người dùng sẽ quan tâm, hoặc giới thiệu những ứng cử viên sáng giá cho nhà tuyển dụng).
* Ứng dụng bảng cấu trúc dạng cây và đệ quy để tìm kiếm công việc được chính xác hơn, đúng chuyên ngành hơn.
* Tích hợp AI để học hồ sơ người dùng và đưa ra công việc phù hợp nhất.

# **DANH MỤC TÀI LIỆU THAM KHẢO**

[1] <https://vi.wikipedia.org/wiki/Java_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)>

[2] <https://blogloi.com/mo-hinh-mvc-la-gi-uu-diem-cua-mvc-nhu-nao/>

[3] <https://stackjava.com/spring/spring-boot-la-gi-gioi-thieu-spring-boot-framework.html>

[4] <https://viblo.asia/p/gioi-thieu-ve-reactjs-phan-i-cac-khai-niem-co-ban-V3m5WzjblO7>

[5] <https://viblo.asia/p/tim-hieu-ve-restful-bWrZnLvQ5xw>

[6] <https://en.wikipedia.org/wiki/Representational_state_transfer>