BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

ĐỀ TÀI: SỬ DỤNG REACT VÀ SPRING BOOT XÂY DỰNG WEBSITE HỖ TRỢ TUYỂN DỤNG VIỆC LÀM

MỞ ĐẦU

1. Tổng quan đề tài

Trong bối cảnh thị trường lao động ngày càng cạnh tranh và phát triển nhanh chóng, nhu cầu kết nối giữa nhà tuyển dụng và ứng viên trở nên cấp thiết hơn bao giờ hết. Các nền tảng tuyển dụng trực tuyến đóng vai trò quan trọng trong việc tối ưu hóa quy trình tìm kiếm việc làm và tuyển dụng nhân sự. Tuy nhiên, nhiều hệ thống hiện tại vẫn còn tồn tại những hạn chế về trải nghiệm người dùng, tính năng tìm kiếm, quản lý hồ sơ, hoặc khả năng mở rộng. Điều này đặt ra yêu cầu về việc xây dựng một website tuyển dụng hiện đại, hiệu quả, đáp ứng được cả nhu cầu của nhà tuyển dụng lẫn ứng viên.

Đề tài này tập trung vào việc xây dựng một website hỗ trợ tuyển dụng việc làm sử dụng các công nghệ tiên tiến như React cho giao diện người dùng (Frontend) và Spring Boot cho phần xử lý logic và dữ liệu (Backend). Mục tiêu là tạo ra một nền tảng trực tuyến mạnh mẽ, thân thiện với người dùng, cung cấp các tính năng toàn diện từ đăng tin tuyển dụng, tìm kiếm việc làm, nộp hồ sơ, quản lý ứng tuyển, đến các công cụ hỗ trợ giao tiếp giữa hai bên. Hệ thống sẽ được thiết kế để đảm bảo hiệu suất cao, bảo mật tốt và khả năng mở rộng trong tương lai.

2. Mục tiêu đề tài

Mục tiêu chính của đề tài là xây dựng thành công một website hỗ trợ tuyển dụng việc làm với các tính năng cốt lõi sau:

- **Đối với nhà tuyển dụng:** Cung cấp công cụ để đăng tin tuyển dụng một cách dễ dàng, quản lý các tin đã đăng, xem và quản lý hồ sơ ứng viên, cũng như theo dõi trạng thái ứng tuyển.
- **Đối với ứng viên:** Cung cấp khả năng tìm kiếm việc làm hiệu quả theo nhiều tiêu chí (ngành nghề, địa điểm, mức lương), nộp hồ sơ trực tuyến, quản lý hồ sơ cá nhân và theo dõi lịch sử ứng tuyển.
- **Đối với quản trị viên:** Cung cấp giao diện quản lý người dùng, tin tuyển dụng, danh mục ngành nghề, và các thiết lập hệ thống khác.

- **Về công nghệ:** Ứng dụng thành thạo React và Spring Boot để xây dựng một hệ thống có kiến trúc rõ ràng, dễ bảo trì và mở rộng.
- **Về trải nghiệm người dùng:** Đảm bảo giao diện trực quan, dễ sử dụng, tối ưu hóa tốc độ tải trang và tương thích trên nhiều thiết bị.

3. Đối tượng nghiên cứu và phạm vi nghiên cứu

a. Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài bao gồm:

- **Nhà tuyển dụng:** Các doanh nghiệp, tổ chức, hoặc cá nhân có nhu cầu đăng tin tuyển dụng và tìm kiếm ứng viên phù hợp.
- **Ứng viên:** Những người đang tìm kiếm việc làm, có nhu cầu nộp hồ sơ và theo dõi quá trình ứng tuyển.
- **Công nghệ:** Các framework và thư viện như ReactJS, Spring Boot, cơ sở dữ liệu MySQL, và các công cụ phát triển liên quan.

b. Phạm vi nghiên cứu

Đề tài tập trung nghiên cứu và xây dựng website hỗ trợ tuyển dụng việc làm trong phạm vi các tính năng cốt lõi đã nêu trong mục tiêu. Hệ thống sẽ được phát triển để phục vụ thị trường lao động tại Việt Nam, với khả năng mở rộng để hỗ trợ đa ngôn ngữ và đa tiền tệ trong tương lai.

4. Phương pháp nghiên cứu

Đề tài sử dụng kết hợp các phương pháp nghiên cứu sau:

- **Nghiên cứu tài liệu:** Thu thập và phân tích các tài liệu liên quan đến React, Spring Boot, kiến trúc phần mềm, cơ sở dữ liệu, và các hệ thống tuyển dụng trực tuyến hiện có.
- **Khảo sát và phân tích yêu cầu:** Tiến hành khảo sát thị trường, phân tích nhu cầu của nhà tuyển dụng và ứng viên để xác định các tính năng cần thiết cho hệ thống.
- **Phân tích và thiết kế hệ thống:** Áp dụng các phương pháp phân tích và thiết kế hướng đối tượng (UML) để xây dựng các mô hình như Use Case, Class Diagram, Sequence Diagram, và ERD.
- **Thực nghiệm và triển khai:** Xây dựng hệ thống theo các giai đoạn, kiểm thử, và triển khai trên môi trường thực tế để đánh giá hiệu quả.

5. Giải pháp công nghệ

Giải pháp công nghệ được lựa chọn cho đề tài này là sự kết hợp giữa ReactJS cho Frontend và Spring Boot cho Backend. Đây là hai công nghệ phổ biến, mạnh mẽ và có cộng đồng hỗ trợ lớn, phù hợp để xây dựng các ứng dụng web hiện đại, có khả năng mở rộng và hiệu suất cao. MySQL sẽ được sử dụng làm hệ quản trị cơ sở dữ liệu.

6. Cấu trúc đồ án

Đồ án được cấu trúc thành các chương chính như sau:

- **Mở đầu:** Giới thiệu tổng quan về đề tài, mục tiêu, đối tượng, phạm vi, phương pháp nghiên cứu, giải pháp công nghệ và cấu trúc đồ án.
- **Chương 1: Cơ sở lý thuyết:** Trình bày các kiến thức nền tảng về ReactJS, Spring Boot, và các công nghệ liên quan.
- **Chương 2: Phân tích thiết kế hệ thống:** Mô tả quá trình khảo sát yêu cầu, phân tích và thiết kế các thành phần của hệ thống, bao gồm các mô hình UML và thiết kế cơ sở dữ liệu.
- **Chương 3: Xây dựng chương trình:** Trình bày chi tiết quá trình cài đặt môi trường, triển khai các module, và phát triển giao diện người dùng.
- **Kết luận và hướng phát triển:** Tổng kết các kết quả đạt được, nêu ra những hạn chế và đề xuất các hướng phát triển tiếp theo cho đề tài.

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

Chương này trình bày các kiến thức lý thuyết nền tảng liên quan đến các công nghệ chính được sử dụng trong đề tài, bao gồm ReactJS cho phát triển giao diện người dùng (Frontend) và Spring Boot cho phát triển phần xử lý logic và dữ liệu (Backend). Ngoài ra, chương cũng đề cập đến hệ quản trị cơ sở dữ liệu MySQL và một số khái niệm quan trọng khác.

1.1. Giới thiệu về ReactJS

ReactJS là một thư viện JavaScript mã nguồn mở, được phát triển bởi Facebook, chuyên dùng để xây dựng giao diện người dùng (UI) động và tương tác cao. ReactJS nổi bật với kiến trúc dựa trên component, cho phép chia nhỏ giao diện thành các thành phần độc lập, có thể tái sử dụng, giúp quản lý mã nguồn dễ dàng và hiệu quả hơn.

1.1.1. Các khái niệm cốt lõi của ReactJS

- **Components (Thành phần):** Là những khối xây dựng cơ bản của một ứng dụng React. Mỗi component đại diện cho một phần của giao diện người dùng và có thể

chứa logic riêng. Components có thể được lồng vào nhau để tạo ra các giao diện phức tạp. Có hai loại component chính là Function Components và Class Components.
- **JSX (JavaScript XML):** Là một phần mở rộng cú pháp cho JavaScript, cho phép viết mã HTML-like trực tiếp trong mã JavaScript. JSX giúp việc mô tả cấu trúc giao diện trở nên trực quan và dễ đọc hơn.
- **Props (Properties):** Là cách để truyền dữ liệu từ component cha xuống component con. Props là bất biến (immutable), nghĩa là component con không thể thay đổi giá trị của props mà nó nhận được.
- **State (Trạng thái):** Là một đối tượng JavaScript dùng để lưu trữ dữ liệu động của một component. Khi state thay đổi, React sẽ tự động render lại component đó để phản ánh sự thay đổi trên giao diện người dùng. State chỉ có thể được quản lý và thay đổi bên trong chính component đó.
- **Virtual DOM (DOM ảo):** React sử dụng một bản sao ảo của DOM (Document Object Model) thực tế. Khi có sự thay đổi về state hoặc props, React sẽ cập nhật Virtual DOM trước, sau đó so sánh Virtual DOM mới với Virtual DOM cũ để xác định những thay đổi cần thiết. Cuối cùng, React chỉ cập nhật những phần thực sự thay đổi trên DOM thực tế, giúp tối ưu hóa hiệu suất render.
- **Lifecycle Methods (Phương thức vòng đời - đối với Class Components):** Là các phương thức được gọi tự động ở các giai đoạn khác nhau trong vòng đời của một component, từ khi được tạo ra, cập nhật, cho đến khi bị hủy bỏ. Các phương thức này cho phép thực hiện các hành động cụ thể tại từng giai đoạn, ví dụ như fetch dữ liệu từ API khi component được mount, hoặc dọn dẹp tài nguyên khi component unmount.
- **Hooks (Đối với Function Components):** Là các hàm đặc biệt cho phép sử dụng state và các tính năng khác của React trong Function Components mà không cần viết class. Một số Hooks phổ biến bao gồm `useState`, `useEffect`, `useContext`.

1.1.2. Ưu điểm của ReactJS

- **Hiệu suất cao:** Nhờ sử dụng Virtual DOM, React tối ưu hóa quá trình cập nhật giao diện, giúp ứng dụng chạy nhanh và mượt mà hơn.
- **Tái sử dụng component:** Kiến trúc component cho phép xây dựng các thành phần giao diện độc lập và có thể tái sử dụng ở nhiều nơi khác nhau, giúp tiết kiệm thời gian và công sức phát triển.
- **Cộng đồng lớn và hệ sinh thái phong phú:** React có một cộng đồng phát triển đông đảo và năng động, cùng với một hệ sinh thái thư viện và công cụ hỗ trợ đa dạng.

- **SEO-friendly (Thân thiện với SEO):** React có thể được render phía server (Server-Side Rendering - SSR) hoặc sử dụng các kỹ thuật pre-rendering, giúp cải thiện khả năng SEO của ứng dụng web.
- **Dễ học và sử dụng (đối với người đã có kiến thức JavaScript):** Với cú pháp rõ ràng và tài liệu hướng dẫn chi tiết, React tương đối dễ tiếp cận đối với các nhà phát triển đã quen thuộc với JavaScript.

1.2. Giới thiệu về Spring Boot

Spring Boot là một framework mã nguồn mở dựa trên Java, được phát triển bởi Pivotal Software (nay là một phần của VMware). Spring Boot được xây dựng trên nền tảng của Spring Framework, nhằm mục đích đơn giản hóa quá trình phát triển các ứng dụng Java, đặc biệt là các ứng dụng microservices và web applications.

1.2.1. Các tính năng chính của Spring Boot

- **Auto-configuration (Tự động cấu hình):** Spring Boot tự động cấu hình ứng dụng dựa trên các dependencies (thư viện phụ thuộc) có trong project. Điều này giúp giảm thiểu lượng mã cấu hình boilerplate mà nhà phát triển cần viết.
- **Standalone (Độc lập):** Các ứng dụng Spring Boot có thể được đóng gói thành các file JAR hoặc WAR thực thi độc lập, chứa sẵn một web server nhúng (ví dụ: Tomcat, Jetty, Undertow), giúp việc triển khai trở nên đơn giản hơn.
- **Opinionated (Có định hướng):** Spring Boot cung cấp các cấu hình mặc định được tối ưu hóa cho nhiều trường hợp sử dụng phổ biến, giúp nhà phát triển bắt đầu nhanh chóng mà không cần phải lo lắng về việc cấu hình chi tiết.
- **Spring Initializr:** Là một công cụ web cho phép dễ dàng tạo ra các project Spring Boot với các dependencies và cấu hình cơ bản đã được thiết lập sẵn.
- **Actuator:** Cung cấp các endpoint để giám sát và quản lý ứng dụng Spring Boot trong môi trường production, ví dụ như kiểm tra sức khỏe (health check), xem metrics, thông tin môi trường.
- **Hỗ trợ mạnh mẽ cho Microservices:** Spring Boot tích hợp tốt với các công cụ và thư viện hỗ trợ xây dựng kiến trúc microservices như Spring Cloud.

1.2.2. Ưu điểm của Spring Boot

- **Phát triển nhanh chóng:** Nhờ tính năng tự động cấu hình và các starter dependencies, Spring Boot giúp giảm thiểu thời gian thiết lập và cấu hình ban đầu, cho phép nhà phát triển tập trung vào việc xây dựng logic nghiệp vụ.
- **Dễ dàng triển khai:** Khả năng đóng gói ứng dụng thành file JAR/WAR độc lập với web server nhúng giúp đơn giản hóa quá trình triển khai.
- **Cộng đồng lớn và tài liệu phong phú:** Spring Boot được hỗ trợ bởi một cộng đồng lớn và có rất nhiều tài liệu, hướng dẫn, và ví dụ minh họa.

- **Tích hợp tốt với hệ sinh thái Spring:** Spring Boot tận dụng được sức mạnh của Spring Framework và các project con khác trong hệ sinh thái Spring, cung cấp một giải pháp toàn diện cho phát triển ứng dụng Java.
- **Phù hợp cho Microservices:** Các tính năng của Spring Boot rất phù hợp để xây dựng các ứng dụng theo kiến trúc microservices, giúp tăng tính linh hoạt và khả năng mở rộng của hệ thống.

1.3. Giới thiệu về MySQL

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System - RDBMS) mã nguồn mở phổ biến. Nó được biết đến với tốc độ, độ tin cậy và tính dễ sử dụng. MySQL được sử dụng rộng rãi trong các ứng dụng web, từ các website nhỏ đến các hệ thống doanh nghiệp lớn.

1.3.1. Các đặc điểm chính của MySQL

- **Mã nguồn mở:** MySQL là phần mềm mã nguồn mở, cho phép người dùng tự do sử dụng, sửa đổi và phân phối.
- **Hiệu suất cao:** MySQL được tối ưu hóa để xử lý các truy vấn dữ liệu nhanh chóng và hiệu quả.
- **Độ tin cậy:** MySQL cung cấp các tính năng đảm bảo tính toàn vẹn và nhất quán của dữ liệu, cũng như khả năng sao lưu và phục hồi.
- **Khả năng mở rộng:** MySQL có thể được cấu hình để hoạt động trên các cụm máy chủ (clusters), giúp tăng khả năng chịu tải và tính sẵn sàng cao.
- **Bảo mật:** MySQL cung cấp các cơ chế kiểm soát truy cập và mã hóa dữ liệu để bảo vệ thông tin nhạy cảm.
- **Hỗ trợ đa nền tảng:** MySQL có thể chạy trên nhiều hệ điều hành khác nhau như Linux, Windows, macOS.

1.4. Kiến trúc MVC (Model-View-Controller)

Kiến trúc MVC là một mẫu thiết kế phần mềm phổ biến, được sử dụng để tách biệt các thành phần của ứng dụng thành ba phần chính: Model, View, và Controller.

- **Model (Mô hình):** Đại diện cho dữ liệu và logic nghiệp vụ của ứng dụng. Model chịu trách nhiệm quản lý dữ liệu, thực hiện các thao tác xử lý dữ liệu và tương tác với cơ sở dữ liệu.
- **View (Giao diện):** Chịu trách nhiệm hiển thị dữ liệu cho người dùng. View nhận dữ liệu từ Controller và trình bày nó dưới dạng giao diện người dùng. View không chứa logic nghiệp vụ.

- **Controller (Bộ điều khiển):** Đóng vai trò trung gian giữa Model và View. Controller nhận yêu cầu từ người dùng (thông qua View), xử lý yêu cầu đó bằng cách tương tác với Model, và sau đó cập nhật View để hiển thị kết quả.

Việc áp dụng kiến trúc MVC giúp tăng tính module hóa, dễ bảo trì, và khả năng tái sử dụng mã nguồn của ứng dụng.

CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

Chương này tập trung vào quá trình phân tích và thiết kế hệ thống website hỗ trợ tuyển dụng việc làm, từ việc khảo sát yêu cầu người dùng đến việc xây dựng các mô hình thiết kế chi tiết. Mục tiêu là định hình cấu trúc và chức năng của hệ thống trước khi tiến hành xây dựng chương trình.

2.1. Khảo sát yêu cầu

Để xây dựng một hệ thống tuyển dụng hiệu quả, việc khảo sát và phân tích yêu cầu là bước vô cùng quan trọng. Quá trình này giúp xác định rõ ràng các tính năng cần thiết, đối tượng người dùng, và các ràng buộc của hệ thống.

2.1.1. Nghiên cứu thị trường và nhu cầu người dùng

Thị trường tuyển dụng trực tuyến tại Việt Nam đang phát triển mạnh mẽ với sự tham gia của nhiều nền tảng lớn như VietnamWorks, TopCV, CareerBuilder, JobStreet, v.v. Mỗi nền tảng đều có những ưu điểm và nhược điểm riêng. Việc nghiên cứu các nền tảng này giúp chúng ta học hỏi kinh nghiệm, xác định các tính năng phổ biến và những điểm còn thiếu sót để cải thiện trong hệ thống của mình.

Nhu cầu của người dùng có thể được chia thành hai nhóm chính:

- **Nhà tuyển dụng:**
    - Đăng tin tuyển dụng nhanh chóng, dễ dàng, có thể tùy chỉnh các thông tin chi tiết (vị trí, mức lương, yêu cầu, mô tả công việc).
    - Quản lý hiệu quả các tin tuyển dụng đã đăng (chỉnh sửa, ẩn/hiện, xóa).
    - Tiếp cận và quản lý hồ sơ ứng viên (xem, tải xuống, phân loại, đánh giá).
    - Theo dõi trạng thái ứng tuyển của các ứng viên.
    - Công cụ lọc và tìm kiếm ứng viên theo tiêu chí.
    - Khả năng giao tiếp trực tiếp với ứng viên (chat, email).

- **Ứng viên:**
    - Tìm kiếm việc làm linh hoạt theo nhiều tiêu chí (từ khóa, ngành nghề, địa điểm, mức lương, kinh nghiệm).
    - Nộp hồ sơ trực tuyến dễ dàng, nhanh chóng.
    - Quản lý hồ sơ cá nhân (tạo, chỉnh sửa, cập nhật CV).
    - Theo dõi trạng thái ứng tuyển của các công việc đã nộp.
    - Nhận thông báo về các công việc phù hợp hoặc cập nhật trạng thái ứng tuyển.
    - Khả năng tạo hồ sơ nổi bật để thu hút nhà tuyển dụng.

2.1.2. Phân tích các hệ thống tương tự

Việc phân tích các hệ thống tuyển dụng hiện có giúp chúng ta hiểu rõ hơn về các luồng nghiệp vụ, các tính năng đã được triển khai, và những thách thức mà các hệ thống này đang gặp phải. Từ đó, chúng ta có thể rút ra những bài học kinh nghiệm quý báu để áp dụng vào thiết kế hệ thống của mình, tránh lặp lại những sai lầm và tối ưu hóa các tính năng cần thiết.

2.2. Phân tích thiết kế hệ thống

Sau khi khảo sát yêu cầu, chúng ta tiến hành phân tích và thiết kế các thành phần của hệ thống. Phần này sẽ trình bày các mô hình thiết kế chính như ERD, Use Case, và các chức năng cốt lõi.

2.2.1. Liệt kê Actor và Use Case

Actors (Tác nhân):

- **Khách vãng lai (Guest):** Người dùng chưa đăng nhập vào hệ thống.
- **Ứng viên (Job Seeker):** Người dùng đã đăng ký tài khoản và tìm kiếm việc làm.
- **Nhà tuyển dụng (Recruiter):** Người dùng đã đăng ký tài khoản và có nhu cầu đăng tin tuyển dụng.
- **Quản trị viên (Admin):** Người dùng có quyền quản lý toàn bộ hệ thống.

Use Cases (Trường hợp sử dụng):

- **Khách vãng lai:**
    - Tìm kiếm việc làm.
    - Xem chi tiết tin tuyển dụng.
    - Đăng ký tài khoản.
    - Đăng nhập.

- **Ứng viên:**
    - Tìm kiếm việc làm.
    - Xem chi tiết tin tuyển dụng.
    - Nộp hồ sơ.
    - Quản lý hồ sơ cá nhân (tạo/chỉnh sửa CV).
    - Theo dõi trạng thái ứng tuyển.
    - Cập nhật thông tin cá nhân.
    - Đổi mật khẩu.
    - Đăng xuất.
- **Nhà tuyển dụng:**
    - Đăng tin tuyển dụng.
    - Quản lý tin tuyển dụng (chỉnh sửa, ẩn/hiện, xóa).
    - Xem danh sách ứng viên đã nộp hồ sơ.
    - Quản lý hồ sơ ứng viên (xem, đánh giá, thay đổi trạng thái).
    - Cập nhật thông tin công ty.
    - Đổi mật khẩu.
    - Đăng xuất.
- **Quản trị viên:**
    - Quản lý người dùng (ứng viên, nhà tuyển dụng, admin).
    - Quản lý tin tuyển dụng (duyệt, ẩn/hiện, xóa).
    - Quản lý danh mục (ngành nghề, địa điểm, cấp bậc).
    - Xem báo cáo thống kê hệ thống.
    - Đổi mật khẩu.
    - Đăng xuất.

2.2.2. Sơ đồ Use Case tổng quát

(Sơ đồ Use Case sẽ được vẽ và chèn vào đây. Hiện tại, tôi sẽ mô tả bằng văn bản.)

Sơ đồ Use Case tổng quát sẽ thể hiện mối quan hệ giữa các Actor và các Use Case chính của hệ thống. Các Use Case chung như Đăng nhập, Đăng ký sẽ được chia sẻ giữa các Actor phù hợp.

2.2.3. Thiết kế ERD (Entity-Relationship Diagram)

ERD mô tả cấu trúc cơ sở dữ liệu của hệ thống, bao gồm các thực thể (Entity), thuộc tính (Attribute) và mối quan hệ (Relationship) giữa chúng. Dưới đây là các thực thể chính dự kiến cho hệ thống tuyển dụng việc làm:

- **User:** Lưu trữ thông tin chung của người dùng (id, email, password, role, ...).
- **JobSeekerProfile:** Thông tin chi tiết của ứng viên (user_id, full_name, phone, address, education, experience, skills, cv_url, ...).
- **RecruiterProfile:** Thông tin chi tiết của nhà tuyển dụng/công ty (user_id, company_name, company_address, company_description, industry, ...).
- **JobPosting:** Thông tin về tin tuyển dụng (id, recruiter_id, title, description, requirements, benefits, location, salary_min, salary_max, employment_type, industry_id, created_at, status, ...).
- **Application:** Thông tin về việc ứng viên nộp hồ sơ vào tin tuyển dụng (id, job_seeker_id, job_posting_id, applied_at, status, cover_letter, ...).
- **Industry:** Danh mục ngành nghề (id, name).
- **Location:** Danh mục địa điểm (id, name).
- **Skill:** Danh mục kỹ năng (id, name).
- **UserSkill:** Mối quan hệ giữa User và Skill (user_id, skill_id).
- **Notification:** Thông báo cho người dùng (id, user_id, message, type, is_read, created_at).

(Sơ đồ ERD sẽ được vẽ và chèn vào đây. Hiện tại, tôi sẽ mô tả bằng văn bản.)

2.2.4. Các chức năng chính

- **Quản lý tài khoản người dùng:** Đăng ký, đăng nhập, đổi mật khẩu, cập nhật thông tin cá nhân/công ty.
- **Tìm kiếm và xem việc làm:** Cho phép ứng viên tìm kiếm việc làm theo từ khóa, ngành nghề, địa điểm, mức lương, loại hình công việc. Hiển thị chi tiết thông tin từng tin tuyển dụng.
- **Nộp hồ sơ ứng tuyển:** Ứng viên có thể nộp hồ sơ trực tuyến, đính kèm CV và thư xin việc.
- **Quản lý tin tuyển dụng (Nhà tuyển dụng):** Tạo mới, chỉnh sửa, xóa, ẩn/hiện tin tuyển dụng. Xem danh sách ứng viên đã nộp hồ sơ cho từng tin.
- **Quản lý hồ sơ ứng viên (Nhà tuyển dụng):** Xem, tải xuống CV, thay đổi trạng thái ứng tuyển (đã xem, phỏng vấn, từ chối, chấp nhận).
- **Quản lý hồ sơ cá nhân (Ứng viên):** Tạo, chỉnh sửa, cập nhật CV và thông tin cá nhân.
- **Theo dõi trạng thái ứng tuyển (Ứng viên):** Xem lịch sử các công việc đã ứng tuyển và trạng thái hiện tại của từng hồ sơ.

- **Hệ thống thông báo:** Gửi thông báo cho ứng viên về trạng thái hồ sơ, công việc mới phù hợp; gửi thông báo cho nhà tuyển dụng về hồ sơ mới.
- **Quản lý hệ thống (Quản trị viên):** Quản lý danh sách người dùng, tin tuyển dụng, danh mục, xem thống kê.

2.2.5. Phác thảo giao diện (Wireframes)

Phần này sẽ trình bày các phác thảo giao diện người dùng cho các trang chính của website, bao gồm trang chủ, trang tìm kiếm việc làm, trang chi tiết tin tuyển dụng, trang quản lý hồ sơ ứng viên, trang đăng tin tuyển dụng, v.v. Các phác thảo này giúp hình dung được bố cục và luồng tương tác của người dùng với hệ thống.

(Các hình ảnh phác thảo giao diện sẽ được chèn vào đây.)

CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH

Chương này mô tả chi tiết quá trình xây dựng website hỗ trợ tuyển dụng việc làm, bao gồm việc thiết lập môi trường phát triển, triển khai các module Backend với Spring Boot, phát triển giao diện Frontend với ReactJS, và tích hợp các thành phần lại với nhau. Mục tiêu là biến các thiết kế từ Chương 2 thành một ứng dụng hoạt động thực tế.

3.1. Công nghệ và Công cụ sử dụng

Để xây dựng hệ thống, chúng ta sử dụng các công nghệ và công cụ chính sau:

- **Backend:**
    - **Ngôn ngữ lập trình:** Java 17 (hoặc phiên bản mới nhất).
    - **Framework:** Spring Boot 3.x.
    - **Thư viện hỗ trợ:** Spring Data JPA (cho tương tác cơ sở dữ liệu), Spring Security (cho bảo mật và xác thực), Lombok (giảm boilerplate code), Jackson (xử lý JSON).
    - **Web Server nhúng:** Tomcat (mặc định của Spring Boot).
- **Frontend:**
    - **Thư viện JavaScript:** ReactJS 18.x.
    - **Ngôn ngữ:** JavaScript (ES6+) và JSX.
    - **Quản lý trạng thái:** React Context API hoặc Redux (tùy chọn).
    - **Routing:** React Router DOM.
    - **UI Framework/Library:** Ant Design, Material-UI, hoặc Tailwind CSS (tùy chọn cho giao diện).
    - **HTTP Client:** Axios (để gọi API từ Backend).

- **Cơ sở dữ liệu:**
    - **Hệ quản trị CSDL:** MySQL 8.x.
    - **Công cụ quản lý CSDL:** MySQL Workbench, DBeaver.
- **Công cụ phát triển và quản lý mã nguồn:**
    - **IDE:** IntelliJ IDEA (cho Backend), Visual Studio Code (cho Frontend).
    - **Hệ thống quản lý phiên bản:** Git.
    - **Nền tảng lưu trữ mã nguồn:** GitHub/GitLab/Bitbucket.
    - **Công cụ kiểm thử API:** Postman.
    - **Công cụ quản lý gói (Frontend):** npm hoặc Yarn.

3.2. Thiết lập môi trường phát triển

3.2.1. Môi trường Backend (Spring Boot)

1. **Cài đặt Java Development Kit (JDK):** Đảm bảo JDK 17 (hoặc phiên bản tương thích) đã được cài đặt và cấu hình biến môi trường `JAVA_HOME`.
2. **Cài đặt IntelliJ IDEA:** Đây là IDE được khuyến nghị cho phát triển Spring Boot.
3. **Tạo dự án Spring Boot:** Sử dụng Spring Initializr (start.spring.io) để tạo một dự án mới với các dependencies cần thiết như Spring Web, Spring Data JPA, MySQL Driver, Spring Security, Lombok.
4. **Cấu hình cơ sở dữ liệu:** Trong file `application.properties` (hoặc `application.yml`), cấu hình thông tin kết nối đến MySQL (URL, username, password) và các thuộc tính JPA/Hibernate.

3.2.2. Môi trường Frontend (ReactJS)

1. **Cài đặt Node.js và npm/Yarn:** Đảm bảo Node.js (bao gồm npm) hoặc Yarn đã được cài đặt trên hệ thống.
2. **Cài đặt Visual Studio Code:** IDE phổ biến cho phát triển Frontend.
3. **Tạo dự án React:** Sử dụng `create-react-app` hoặc Vite để khởi tạo một dự án React mới. `bash npx create-react-app job-portal-frontend # hoặc với Vite npm create vite@latest job-portal-frontend -- --template react`
4. **Cài đặt các thư viện cần thiết:** Cài đặt `axios` để gọi API, `react-router-dom` cho định tuyến, và các thư viện UI nếu có. `bash cd job-portal-frontend npm install axios react-router-dom`

3.3. Phát triển Backend với Spring Boot

Phần Backend sẽ chịu trách nhiệm xử lý logic nghiệp vụ, tương tác với cơ sở dữ liệu, và cung cấp các API cho Frontend. Kiến trúc sẽ tuân theo mô hình MVC (Model-View-Controller) hoặc một biến thể của nó (ví dụ: Controller-Service-Repository).

3.3.1. Thiết kế cơ sở dữ liệu và Entity

Dựa trên ERD đã thiết kế ở Chương 2, chúng ta sẽ tạo các bảng trong MySQL và định nghĩa các Entity Java tương ứng sử dụng Spring Data JPA. Mỗi Entity sẽ ánh xạ tới một bảng trong cơ sở dữ liệu.

Ví dụ về Entity `User`:

```
@Entity
@Table(name = "users")
@Data
@NoArgsConstructor
@AllArgsConstructor
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String email;
    private String password;
    private String role; // e.g., "JOB_SEEKER", "RECRUITER",
"ADMIN"
    // other fields
}
```

3.3.2. Xây dựng Repository Layer

Repository Layer (lớp kho lưu trữ) sử dụng Spring Data JPA để tương tác với cơ sở dữ liệu. Chúng ta sẽ định nghĩa các interface kế thừa từ `JpaRepository` để thực hiện các thao tác CRUD (Create, Read, Update, Delete) cơ bản và các truy vấn tùy chỉnh.

```
public interface UserRepository extends JpaRepository<User,
Long> {
    Optional<User> findByEmail(String email);
}
```

3.3.3. Xây dựng Service Layer

Service Layer (lớp dịch vụ) chứa logic nghiệp vụ chính của ứng dụng. Các Service sẽ sử dụng các Repository để thao tác với dữ liệu và thực hiện các quy tắc nghiệp vụ.

```
@Service
public class UserService {
    @Autowired
    private UserRepository userRepository;
    @Autowired
    private PasswordEncoder passwordEncoder;

    public User registerNewUser(User user) {

user.setPassword(passwordEncoder.encode(user.getPassword()));
        return userRepository.save(user);
    }

    public Optional<User> findByEmail(String email) {
        return userRepository.findByEmail(email);
    }
    // other business logic methods
}
```

3.3.4. Xây dựng Controller Layer

Controller Layer (lớp điều khiển) là nơi tiếp nhận các yêu cầu HTTP từ Frontend và trả về các phản hồi. Các Controller sẽ gọi các phương thức từ Service Layer để xử lý yêu cầu.

```
@RestController
@RequestMapping("/api/auth")
public class AuthController {
    @Autowired
    private UserService userService;

    @PostMapping("/register")
    public ResponseEntity<String> registerUser(@RequestBody
User user) {
        userService.registerNewUser(user);
        return ResponseEntity.ok("User registered
successfully!");
    }
    // other API endpoints
}
```

3.3.5. Cấu hình bảo mật với Spring Security

Spring Security sẽ được sử dụng để xử lý xác thực (Authentication) và ủy quyền (Authorization). Chúng ta sẽ cấu hình chuỗi lọc bảo mật, quản lý người dùng, và mã hóa mật khẩu.

3.4. Phát triển Frontend với ReactJS

Phần Frontend sẽ tập trung vào việc xây dựng giao diện người dùng tương tác, gọi các API từ Backend, và quản lý trạng thái của ứng dụng.

3.4.1. Cấu trúc dự án React

Dự án React sẽ được tổ chức theo cấu trúc module, với các thư mục cho components, pages, services, contexts (hoặc redux), assets, v.v.

```
src/
├── components/       // Các thành phần UI nhỏ, có thể tái sử
dụng
├── pages/            // Các trang chính của ứng dụng
├── services/         // Các hàm gọi API Backend
├── contexts/         // Quản lý trạng thái toàn cục (nếu dùng
Context API)
├── assets/           // Hình ảnh, CSS, fonts
├── App.js            // Component gốc
├── index.js          // Điểm khởi đầu của ứng dụng
└── ...
```

3.4.2. Xây dựng các Components và Pages

Chúng ta sẽ phát triển các React Components cho từng phần của giao diện (ví dụ: Header, Footer, JobCard, UserProfileForm) và kết hợp chúng thành các Pages (ví dụ: HomePage, JobListingPage, JobDetailPage, LoginPage, RegisterPage, Dashboard).

Ví dụ về một Component `JobCard`:

```jsx
// components/JobCard.js
import React from 'react';
import { Link } from 'react-router-dom';

const JobCard = ({ job }) => {
  return (
    <div className="job-card">
      <h3><Link to={`/jobs/${job.id}`}>{job.title}</Link></h3>
      <p>Công ty: {job.companyName}</p>
      <p>Địa điểm: {job.location}</p>
      <p>Mức lương: {job.salaryMin} - {job.salaryMax}</p>
      <p>Ngành nghề: {job.industry}</p>
      <button>Ứng tuyển ngay</button>
    </div>
  );
};
```

```
export default JobCard;
```

3.4.3. Tích hợp API với Axios

Các Components và Pages sẽ sử dụng thư viện Axios để gửi các yêu cầu HTTP (GET, POST, PUT, DELETE) đến các API do Backend cung cấp và xử lý dữ liệu trả về.

```
// services/jobService.js
import axios from 'axios';

const API_URL = 'http://localhost:8080/api/jobs'; // Thay đổi
khi deploy

export const getAllJobs = async () => {
  try {
    const response = await axios.get(API_URL);
    return response.data;
  } catch (error) {
    console.error('Error fetching jobs:', error);
    throw error;
  }
};

export const getJobById = async (id) => {
  try {
    const response = await axios.get(`${API_URL}/${id}`);
    return response.data;
  } catch (error) {
    console.error(`Error fetching job ${id}:`, error);
    throw error;
  }
};

// other job related API calls
```

3.4.4. Quản lý trạng thái và Định tuyến

React Context API hoặc Redux sẽ được sử dụng để quản lý trạng thái toàn cục của ứng dụng (ví dụ: thông tin người dùng đã đăng nhập, trạng thái loading). React Router DOM sẽ được dùng để quản lý định tuyến (routing) giữa các trang khác nhau của ứng dụng.

3.5. Triển khai và Vận hành

Sau khi phát triển xong, hệ thống cần được triển khai để người dùng có thể truy cập.

3.5.1. Cài đặt môi trường Production

- **Backend:** Đóng gói ứng dụng Spring Boot thành file JAR thực thi. Cài đặt Java Runtime Environment (JRE) trên server.
- **Frontend:** Build ứng dụng React để tạo ra các file tĩnh (HTML, CSS, JavaScript) đã được tối ưu hóa cho production.

3.5.2. Triển khai trên Server

- **Backend:** Chạy file JAR của Spring Boot trên server. Đảm bảo các cổng cần thiết (ví dụ: 8080) được mở và cấu hình tường lửa phù hợp.
- **Frontend:** Các file tĩnh của React có thể được phục vụ bởi một web server như Nginx hoặc Apache, hoặc được tích hợp trực tiếp vào Spring Boot nếu muốn triển khai cùng một gói.
- **Cơ sở dữ liệu:** Đảm bảo MySQL server đang chạy và có thể truy cập được từ ứng dụng Backend.

3.5.3. Giám sát và Bảo trì

Sau khi triển khai, cần có các công cụ giám sát để theo dõi hiệu suất, lỗi, và các vấn đề bảo mật của hệ thống. Việc bảo trì định kỳ, cập nhật phiên bản thư viện và framework cũng rất quan trọng để đảm bảo hệ thống hoạt động ổn định và an toàn.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết luận

Đề tài "Sử Dụng React và Spring Boot Xây Dựng Website Hỗ Trợ Tuyển Dụng Việc Làm" đã được hoàn thành với mục tiêu xây dựng một nền tảng tuyển dụng trực tuyến hiệu quả, kết nối nhà tuyển dụng và ứng viên. Hệ thống đã triển khai thành công các chức năng cốt lõi, bao gồm:

- **Đối với ứng viên:** Khả năng tìm kiếm việc làm theo nhiều tiêu chí, xem chi tiết tin tuyển dụng, nộp hồ sơ trực tuyến, quản lý hồ sơ cá nhân và theo dõi trạng thái ứng tuyển.
- **Đối với nhà tuyển dụng:** Khả năng đăng tin tuyển dụng, quản lý tin đã đăng, xem và quản lý hồ sơ ứng viên.
- **Đối với quản trị viên:** Khả năng quản lý người dùng, tin tuyển dụng và các danh mục hệ thống.

Việc áp dụng React cho Frontend và Spring Boot cho Backend đã chứng minh được hiệu quả trong việc xây dựng một ứng dụng web hiện đại, có khả năng mở rộng và dễ bảo trì.

Giao diện người dùng được thiết kế trực quan, thân thiện, mang lại trải nghiệm tốt cho người dùng. Kiến trúc phân tán giữa Frontend và Backend giúp hệ thống linh hoạt và dễ dàng phát triển độc lập từng phần.

2. Hướng phát triển

Mặc dù hệ thống đã hoàn thành các chức năng cơ bản, vẫn còn nhiều tiềm năng để phát triển và mở rộng trong tương lai nhằm nâng cao trải nghiệm người dùng và cung cấp nhiều giá trị hơn:

- **Tích hợp AI/Machine Learning:**
    - **Gợi ý việc làm thông minh:** Sử dụng thuật toán học máy để phân tích hành vi và sở thích của ứng viên, từ đó gợi ý các công việc phù hợp nhất.
    - **Phân tích CV tự động:** Tự động trích xuất thông tin từ CV, đánh giá mức độ phù hợp của ứng viên với tin tuyển dụng.
    - **Chatbot hỗ trợ:** Triển khai chatbot để hỗ trợ người dùng giải đáp thắc mắc, hướng dẫn sử dụng hệ thống.
- **Tính năng giao tiếp nâng cao:**
    - **Chat trực tuyến:** Cho phép nhà tuyển dụng và ứng viên trò chuyện trực tiếp trên nền tảng.
    - **Lên lịch phỏng vấn:** Tích hợp công cụ lên lịch phỏng vấn trực tuyến hoặc trực tiếp.
- **Mở rộng tính năng cho nhà tuyển dụng:**
    - **Quản lý chiến dịch tuyển dụng:** Cho phép nhà tuyển dụng tạo và quản lý các chiến dịch tuyển dụng lớn.
    - **Báo cáo và thống kê chuyên sâu:** Cung cấp các báo cáo chi tiết về hiệu quả tin tuyển dụng, số lượng ứng viên, nguồn ứng viên.
- **Cải thiện trải nghiệm ứng viên:**
    - **Tạo CV trực tuyến:** Cung cấp công cụ để ứng viên tạo CV chuyên nghiệp trực tiếp trên hệ thống.
    - **Đánh giá công ty:** Cho phép ứng viên đánh giá các công ty đã ứng tuyển hoặc làm việc.
- **Hỗ trợ đa ngôn ngữ và đa tiền tệ:** Mở rộng hệ thống để phục vụ thị trường quốc tế, hỗ trợ nhiều ngôn ngữ và các loại tiền tệ khác nhau.

- **Tích hợp thanh toán:** Đối với các tính năng cao cấp (ví dụ: đăng tin tuyển dụng nổi bật), tích hợp cổng thanh toán trực tuyến.
- **Phát triển ứng dụng di động:** Xây dựng ứng dụng di động (iOS/Android) để tăng cường khả năng tiếp cận và tiện lợi cho người dùng.

Những hướng phát triển này sẽ giúp hệ thống trở nên toàn diện và cạnh tranh hơn trên thị trường tuyển dụng trực tuyến.