

HỌC VIỆN NÔNG NGHIỆP VIỆT NAM

**KHOA CÔNG NGHỆ THÔNG TIN**

------- 🙞🕮🙜 -------



**BÁO CÁO THỰC TẬP CHUYÊN NGÀNH**

***­­­Đề tài:***

**PHÁT TRIỂN ỨNG DỤNG VỚI REACT**

**XÂY DỰNG ỨNG DỤNG QUẢN LÝ DƯỢC PHẨM**

Họ và tên sinh viên: **Phạm Ngọc Anh - 621189 – K62CNTTA**

**Vũ Minh Hiếu - 621130 – K62CNTTA**

Giảng viên hướng dẫn**: ThS.Nguyễn Văn Hoàng**

Bộ môn**: Khoa học máy tính**

Tên cơ sở thực tập**: ­­­­­­ Công ty Cổ Phần Đào tạo và Phát triển Công nghệ cao Việt Nam**

**HÀ NỘI – 2021**

**LỜI CẢM ƠN**

*Trong lời đầu tiên của báo cáo thực tập về đề tài “****phát triển ứng dụng với React, Xây dựng ứng dụng quản lý dược phẩm****” này, chúng em muốn gửi lời cảm ơn và biết ơn chân thành nhất của mình tời tất cả những người đã hỗ trợ, giúp đỡ chúng em về kiến thực và tinh thần trong lần đầu chúng em được thực tập thực hiện đề tài.*

*Trước hết chúng em xin chân thành cảm ơn thầy Nguyễn Văn Hoàng, thạc sĩ và là giảng viên khoa Công Nghệ Thông Tin, Học Viện Nông Nghiệp Việt Nam, người đã trực tiếp hướng dẫn, nhận xét và giúp đỡ chúng em trong suốt quá trình thực hiện đề tài thực tập.*

*Xin chân thành cảm ơn ban đào tạo Học Viện đã tạo điều kiện tốt nhất cho chúng em cũng như các bạn khác trong suốt thời gian học tập và làm đề tài.*

*Cuối cùng chúng em xin gửi lời cảm ơn đến đội ngũ các anh chị của công ty Cổ Phần Đào tạo và Phát triển Công nghệ cao Việt Nam đã giúp đỡ chúng em rất nhiều trong quá trình thực tập cũng như thực hiện đề tài. Do thời gian thực tập và hoàn thành đề tài có hạn, kiến thức vẫn còn nhiều hạn chế. Đề tài chúng em thực hiện chắc chắn không tránh khỏi những thiếu sót nhất định.*

*Em mong nhận được ý kiến đóng góp của các thầy cô để chúng em có thêm kinh nghiệm để tiếp tục trong các đợt thực tập và khóa luận sau này.*

*Em xin chân thành cảm ơn!*

**MỤC LỤC**

[**Phần I: Mở đầu** 5](#_Toc78045945)

[1.1. Tên đề tài 5](#_Toc78045946)

[1.2. Đặt vấn đề 5](#_Toc78045947)

[1.3. Mục đích, yêu cầu 5](#_Toc78045948)

[1.3.1. mục đích 5](#_Toc78045949)

[1.3.2. yêu cầu 5](#_Toc78045950)

[**Phần II: Tổng quan tình hình nghiên cứu trong và ngoài nước** 6](#_Toc78045951)

[2.1. Tình hình nghiên cứu trong nước 6](#_Toc78045952)

[2.2. Tình hình nghiên cứu ngoài nước 6](#_Toc78045953)

[2.3 Nêu tên đề tài và tính thời sự, tầm quan trọng của đề tài 7](#_Toc78045954)

[**Phần III: Nội dung và phương pháp nghiên cứu** 7](#_Toc78045955)

[3.1. Nội dung nghiên cứu 7](#_Toc78045956)

[3.2. Phương pháp nghiên cứu 8](#_Toc78045957)

[**Phần IV: Kết quả và thảo luận** 8](#_Toc78045958)

[4.1. Tìm hiểu về React 8](#_Toc78045959)

[4.1.1. Giới thiệu 8](#_Toc78045960)

[4.1.2. Đặc tính của React 8](#_Toc78045961)

[4.1.3. Components 10](#_Toc78045962)

[4.1.4. props và state 10](#_Toc78045963)

[4.1.5. Hướng xây dựng component 11](#_Toc78045964)

[4.1.6. Cài đặt dự án ReactJs 12](#_Toc78045965)

[4.2. Tìm hiểu các thư viện 13](#_Toc78045966)

[4.2.1. Ant design 13](#_Toc78045967)

[4.2.2. Recoil 15](#_Toc78045968)

[4.2.3. Visx 17](#_Toc78045969)

[4.3. Backend với Nodejs 20](#_Toc78045970)

[4.3.1. Nodejs 20](#_Toc78045971)

[4.3.2. ExpressJs 22](#_Toc78045972)

[4.3.3. MongoDB 23](#_Toc78045973)

[4.4. Xây dựng ứng dụng quản lý dược phẩm 25](#_Toc78045974)

[4.4.1. Các chức năng chính 25](#_Toc78045975)

[4.4.2. Giao diện khác của ứng dụng 31](#_Toc78045976)

[4.4.3. Database của mongoDB 32](#_Toc78045977)

[**Phần V: Kết luận và đề nghị** 33](#_Toc78045978)

[5.1. Kết luận 33](#_Toc78045979)

[5.2. Đề nghị 34](#_Toc78045980)

[**Phần VI. Tài liệu tham khảo:** 35](#_Toc78045981)

# **Phần I: Mở đầu**

## Tên đề tài

Phát triển ứng dụng web bằng react, để xây dựng một ứng dụng web quản lý dược phẩm

## Đặt vấn đề

Hiện nay React đang ngày càng phát triển và chiếm phần lớn thị trường của frontend, vì vầy việc sử dụng React để xây dựng ứng dụng web là cấp thiết hiện nay. Để ứng dụng điều này nhóm đã quyết định xây dựng một ứng dụng quản lý thuốc.

## Mục đích, yêu cầu

### 1.3.1. mục đích

Xây dựng một ứng dụng web với ReactJs về quản lý dược phẩm, với giao diện đẹp mắt dễ dàng sử dụng, đáp ứng đầy đủ các tính năng mà người quản trị cần có.

### 1.3.2. yêu cầu

* Tìm hiểu về ReactJs
* Xác định cấu trúc, các thành phần chính trong React
* Xây dựng một ứng dụng web quản lý dươc phẩm
* Quản lý các loại thuốc, thành phần, công thức
* Thêm các quy trình sản xuất
* Cập nhật, thêm, sửa, xóa
* BackEnd sử dụng Nodejs

# **Phần II: Tổng quan tình hình nghiên cứu trong và ngoài nước**

## 2.1. Tình hình nghiên cứu trong nước

Tại Việt Nam, vai trò của khoa học - công nghệ (KH&CN) ngày càng trở nên quan trọng. Những tiến bộ như vũ bão của KH&CN trong hai thập kỷ qua, đặc biệt là trong lĩnh vực công nghệ thông tin. Vì vậy việc nghiên cứu phát triển cũng như ứng dụng các thành tự mới về Kh&CN trong lĩnh vực Y tế cần được hết sức coi trọng.

Trong lĩnh vực dược, trang thiết bị y tế cần tiếp tục tiếp thu ứng dụng các công nghệ mới trong vào sản xuất thuốc và vật tư tiêu hao đáp ứng nhu cầu tiêu thụ trong tỉnh và mở rộng thị trường, nâng cao khả năng cạnh tranh của sản phẩm trên thị trường trong và ngoài nước; Nghiên cứu cải tiến các phương pháp sản xuất, bào chế thuốc để tăng năng suất và hiệu quả, hạ giá thành sản phẩm. Từng bước tiếp thu kỹ thuật, công nghệ tiến tới hình thành trung tâm sửa chữa, bảo dưỡng, kiểm chuẩn trang thiết bị y tế phục vụ nhu cầu của các cơ sở y tế trong và ngoài tỉnh.

Bên cạnh đó là sự phát triển không ngừng của react nói chung và ReactJs nói riêng. Từ khi mới xuất hiện tại Việt nam nó đã mang đến một làn gió mới cho cộng đồng frontend. Với các ưu điểm mà nó đem lại ReactJs được sử dụng vào để xây dựng các ứng dụng trên nền web.

## 2.2. Tình hình nghiên cứu ngoài nước

Ngày nay, ReactJS đã trở nên rất phổ biến bởi những tính năng linh hoạt và đơn giản với hàng ngàn developer và hơn 94,000 trang web đang sử dụng ReactJS.

Nhiều người ám chỉ rằng ReactJS sẽ là tương lai của việc phát triển web. Sự phổ biến của ReactJS một phần nhờ vào các ông trùm về công nghệ như Facebook, PayPal, Uber, Instagram và Airbnb, sử dụng để giải quyết các vấn đề liên quan đến UI. Từ đó kéo theo làn sóng rất nhiều người và công ty tiếp tục tin tưởng sử dụng ReactJS.

Tại các nước phát triển việc ứng dụng công nghệ thông tin trong lĩnh vực dược phẩm đã được hình thành từ rất lâu. Nó giúp thế giới quản lý về dược phẩm trở nên dễ dàng đem lại một phần không nhỏ giúp phát triên kinh tế rút, giảm bớt đi các nguồn nhân lực truyền thống. Bên cạnh đó vẫn còn các nước kém phát triển chưa đưa các ứng dụng của công nghệ thông tin vào lĩnh vữ nay. Đây là một vấn đề cấp thiết trong thời đại 4.0 khi công nghệ thông tin len lỏi vào từng ngóc ngách của cuộc sống.

## 2.3 Nêu tên đề tài và tính thời sự, tầm quan trọng của đề tài

Tên đề tài: phát tiển ứng dụng với React, xây dựng ứng dụng quản lý dược phẩm

Tính thời sự và quan trọng của đề tài: Dựa trên những dữ liệu chúng em tham khảo cả trong và ngoài nước như ở trên, chúng em đã đặt ra đề tài xây dựng ứng dụng web quản lý dược phẩm với mong muốn tạo ra một ứng dụng web giúp người dùng có được trải nghiệm tuyệt vời với đầy đủ các tính năng của một website với thời gian phát triển ngắn và tiết kiệm tối đa chi phí.

# **Phần III: Nội dung và phương pháp nghiên cứu**

## 3.1. Nội dung nghiên cứu

Tìm hiểm về React nói chung và ReactJs nói riêng

Xây dựng một ứng dụng cho phép quản lý các loại thuốc, các thành phần cấu tạo nên thuốc, tạo ra các quy trình sản xuất, tạo các kế hoạch sản xuất.

## 3.2. Phương pháp nghiên cứu

* Khảo sát thực tế tại các doanh nghiệp dược liệu
* Sử dụng các thông tin trong các trang web uy tín trên internet
* Tham khảo các lời góp ý từ giảng viên và cán bộ hướng dẫn tại cơ sỏ thực tập
* Sử dụng các kiến thức có sẵn về lập trình
* Đọc các document về các thư viện đã sử dụng

# **Phần IV: Kết quả và thảo luận**

## 4.1. Tìm hiểu về React

### 4.1.1. Giới thiệu

[React](https://reactjs.org/)Js là thư viện [JavaScript](https://www.hostinger.vn/huong-dan/javascript-la-gi/) phổ biến nhất để xây dựng giao diện người dùng (UI). Nó cho tốc độ phản hồi hiệu quả khi user nhập liệu bằng cách sử dụng phương pháp mới để render trang web.

### 4.1.2. Đặc tính của React

Tư tưởng của ReactJs là xây dựng các components có tính tái sử dụng, dễ dàng cho việc chia nhỏ vấn đề. Nó giúp chúng ta dễ dàng quản lý, mở rông hệ thống.

**Ưu điểm**

* Reactjs cực kì hiệu quả: Reactjs tạo ra cho chính nó DOM ảo – nơi mà các component thực sự tồn tại trên đó. Điều này sẽ giúp cải thiện hiệu suất rất nhiều. Reactjs cũng tính toán những thay đổi nào cần cập nhật len DOM và chỉ thực hiện chúng. Điều này giúp Reactjs tránh những thao tác cần trên DOM mà nhiều chi phí. Chúng ta có thể viết một ví dụ đơn giản về ReactJS như sau.
* Reactjs giúp việc viết các đoạn code JS dễ dàng hơn: Nó dung cú pháp đặc biệt là JSX (Javascript mở rộng) cho phép ta trộn giữa code HTML và Javascript. Ta có thể them vào các đoạn HTML vào trong hàm render mà không cần phải nối chuỗi. Đây là đặc tính thú vị của Reactjs. Nó sẽ chuyển đổi các đoạn HTML thành các hàm khởi tạo đối tượng HTML bằng bộ biến đổi JSX.
* Nó có nhiều công cụ phát triển: Khi bạn bắt đầu Reactjs, đừng quên cài đặt ứng dụng mở rộng của Chrome dành cho Reactjs. Nó giúp bạn debug code dễ dàng hơn. Sau khi bạn cài đặt ứng dụng này, bạn sẽ có cái nhìn trực tiếp vào virtual DOM như thể bạn đang xem cây DOM thông thường.
* Render tầng server: Một trong những vấn đề với các ứng dụng đơn trang là tối ưu SEO và thời gian tải trang. Nếu tất cả việc xây dựng và hiển thị trang đều thực hiện ở client, thì người dung sẽ phải chờ cho trang được khởi tạo và hiển thị lên. Điều này thực tế là chậm. Hoặc nếu giả sử người dung vô hiệu hóa Javascript thì sao? Reactjs là một thư viện component, nó có thể vừa render ở ngoài trình duyệt sử dụng DOM và cũng có thể render bằng các chuỗi HTML mà server trả về.
* Làm việc với vấn đề test giao diện: Nó cực kì dễ để viết các test case giao diện vì virtual DOM được cài đặt hoàn toàn bằng JS.
* Hiệu năng cao đối với các ứng dụng có dữ liệu thay đổi liên tục, dễ dàng cho bảo trì và sửa lỗi.

**Nhược điểm**

Nhược điểm chính của reactjs đó là:

* Reactjs chỉ phục vụ cho tầng View. React chỉ là View Library nó không phải là một MVC framework như những framework khác. Đây chỉ là thư viện của Facebook giúp render ra phần view. Vì thế React sẽ không có phần Model và Controller, mà phải kết hợp với các thư viện khác. React cũng sẽ không có 2-way binding hay là Ajax
* Tích hợp Reactjs vào các framework MVC truyền thống yêu cầu cần phải cấu hình lại.
* React khá nặng nếu so với các framework khác React có kích thước tương tương với Angular (Khoảng 35kb so với 39kb của Angular). Trong khi đó Angular là một framework hoàn chỉnh
* Khó tiếp cận cho người mới học Web

### 4.1.3. Components

Cho phép chúng ta chia nhỏ các thành phần UI độc lập mục đích để dễ quản lý và tái sử dụng nó. Ví dụ chúng ta có một list các post các post này được sử dụng ở nhiều page khác nhau và có cùng một style ở đây chúng ta sẽ tạo một component Post khi nào cần xử dụng chỉ cần gọi nó ra việc này giúp chũng ta quan lý code tốt hơn.

### 4.1.4. props và state

**Props:**

* Chúng ta sử dụng props để gửi dữ liệu đến component.
* Mọi component được coi là một hàm javascript thuần khiết (Pure Function).
* Trong ReactJS, props tương đương với các tham số của hàm javascript thuần khiết.
* Props là bất biến (không thể thay đổi được). Bởi vì điều này được phát triển trong khái niệm về các hàm thuần khiết. Trong các hàm thuần khiết, chúng ta không thể thay đổi dữ liệu của các tham số. Vì vậy, cũng không thể thay đổi dữ liệu của prop trong ReactJS.

**State:**

* State giống như một kho lưu trữ dữ liệu cho các component trong ReactJS. Nó chủ yếu được sử dụng để cập nhật component khi người dùng thực hiện một số hành động như nhấp vào nút, nhập một số văn bản, nhấn một số phím, v.v.
* React.Component là lớp cơ sở cho tất cả các lớp component cơ bản khác trong ReactJS. Bất cứ khi nào một lớp kế thừa lớp React.Component, hàm tạo (Constructor) của nó sẽ tự động gán thuộc tính state cho lớp với giá trị ban đầu được gán bằng null. Chúng ta có thể thay đổi nó bằng cách ghi đè hàm tạo (Constructor).
* Trong nhiều trường hợp chúng ta cần cập nhật state. Để làm điều đó, chúng ta phải sử dụng phương thức setState và chúng ta không thể gán trực tiếp như thế này this.state = {'key': 'value'}.

### 4.1.5. Hướng xây dựng component

- Xây dựng theo **class**: là một hướng xây dựng component gần với hướng đối tượng, khi sử dụng con trỏ **this** cho các hàm và biến bên trong nó. Tính kế thừa các component khác nhau và có các hàm chuyên dụng cho **lifecycle component**.

- Xây dựng theo hướng **function**: đây là một hướng xây dựng theo hàm, ưu điểm của nó là ngắn gọn hơn so với người anh class của mình nhưng chính bản thân nó lại không hỗ trợ các hàm **lifecycle component** và **state** như class. Chính vì vậy React đã phát triển **hooks**, là các hàm sinh ra để kết nối **state** và **lifecycle component** lại với nhau. Và trong dự án này chúng tôi sẽ tập trung xây dựng component theo hướng function.

### 4.1.6. Cài đặt dự án ReactJs

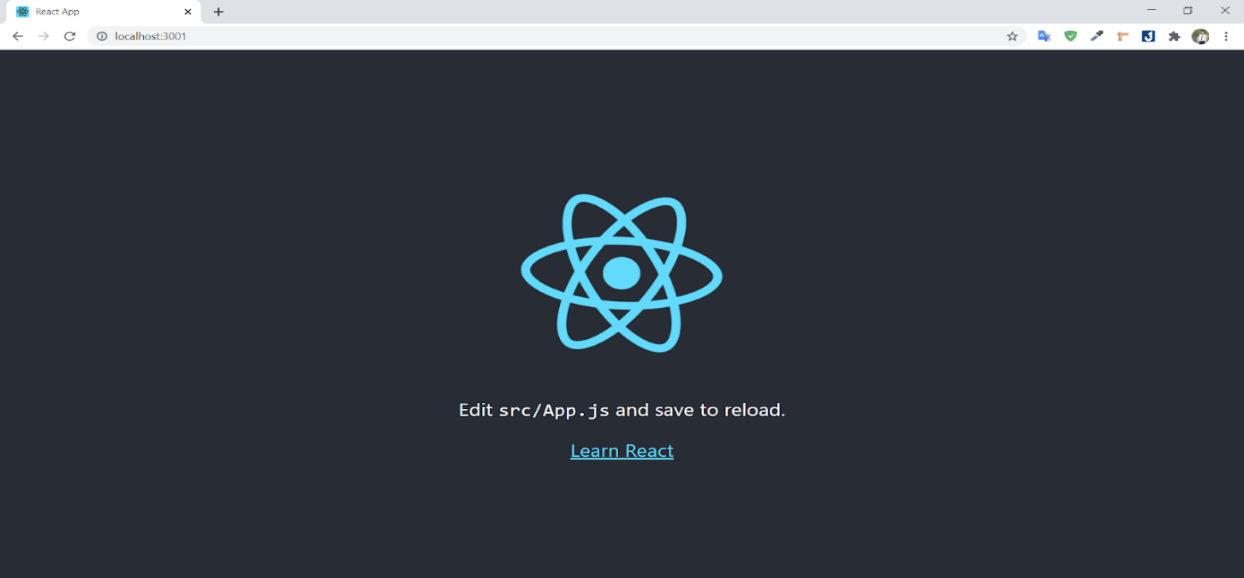
Mở cửa sổ lệnh trên CMD lần lượt chạy các lệnh:

npx create-react-app tên ứng dụng

cd my-app

npm start

Ta sẽ có giao diện như sau:



*Hình 1: giao diện bắt đầu khi tạo dự án với ReactJs*

## 4.2. Tìm hiểu các thư viện

Để chuẩn bị cho dự án, điều đầu tiên cần thực hiện là cài đặt các thư viện của react để giúp code được đơn giản và thuận tiện hơn:

### 4.2.1. Ant design

*Ant design* là tập hợp các components của React được xây dựng theo chuẩn thiết kế của Ant UED Team. Tương tự như chuẩn Material Design, Ant cung cấp hầu hết các component thông dụng trong ứng dụng web hiện đại, như Layout, Button, Icon, DatePicket, v.v…

Cài đặt ant design dung npm hoặc yarn

* npm install antd --save
* # or
* yarn add antd

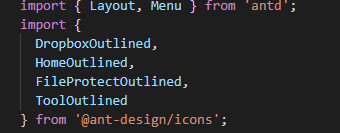
Có thể coi Ant Design cho React là tập hợp của hầu hết các thư viện về React. Nó đáp ứng được hầu hết các yêu cầu của project của bạn mà ban không phải cài thêm bất cứ thư viện nào nữa. Dưới đây là danh sách các component mà nó cung cấp:

* General: Button, Icon
* Layout: Grid, Layout
* Navigation: Affix, Breadcrumb, Dropdown, Menu, Pagination, Steps
* Data Entry: AutoComplete, Checkbox, Cascader, DatePicker, Form, InputNumber, Input, Mention, Rate, Radio, Switch, Slider, Select, TreeSelect, Transfer, TimePicker, Upload
* Data Display: Avatar, Badge, Collapse, Carousel, Card, Calendar, List, Popover, Tree, Tooltip, Timeline, Tag, Tabs, Table
* Feedback: Alert, Drawer, Modal, Message, Notification, Progress, Popconfirm, Spin, Skeleton
* Other: Anchor, BackTop, Divider, LocaleProvider

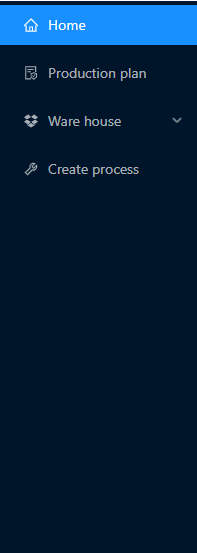
Cách sử dụng:

Bạn chỉ cần import nó vào và sử dụng như các component trong React.

Ví dụ viết thanh menu dọc bằng antd trong ứng dụng:



Giao diện sẽ như sau:



### 4.2.2. Recoil

**Recoil** chính là 1 thư viện dùng để quản lý state của **React**, bên cạnh 1 loạt thư viện hầm hố khác như: redux, mobX, flux, …

Cách cài đặt:

npm install recoil

hoặc

yarn add recoil

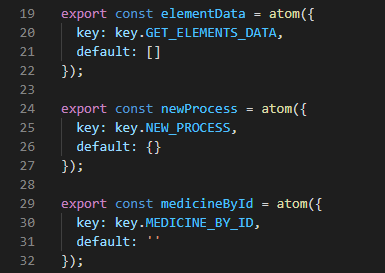
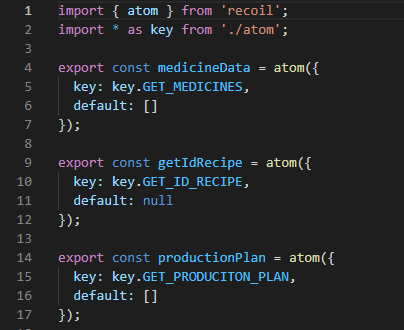
Recoil quản lý state qua:

* Atom: mẩu dữ liệu tương tự state, nhưng sẽ được dùng chung nếu các component subscribe nó.
* Selector: với sức mạnh của getter và setter giúp biến đổi các giá trị đầu ra của atom hoặc thực hiện xử lý bất đồng bộ như gọi API về từ một public resources chẳng hạn.
* Nó hỗ trợ các hàm để lấy ra hay thay đổi state chung giống như cách một state nội bộ hoạt động.

Các thành phần chính của Recoil:

* Atom

**Recoil** lưu trữ dữ liệu theo các đơn vị gọi là **atom**, **React component** có thể truy cập tới dữ liệu chứa trong **atom** và thay đổi giá trị của dữ liệu. **atom** yêu cầu 2 tham số là mã định danh duy nhất (**unique key**) và giá trị mặc định của dữ liệu.



* RecoilRoot

Để có thể sử dụng **recoil**, ứng dụng cần được bao bọc bởi một **component** có tên là **RecoilRoot**, chúng ta sẽ tìm hiểu thêm về **api** này hơn sau này, trước mắt cứ hiểu nó như **Provider** trong **redux**, và không cần truyền vào **store** nào cả.

<RecoilRoot>

<App />

</RecoilRoot>

* **Selectors**

Trả về dữ liệu đã được tính toán (giống với thư viện **reselect** trong hệ sinh thái **redux**). Nếu bạn chưa sử dụng **reselect**, chúng ta sẽ tìm hiểu khái niệm dữ liệu tính toán thông qua ví dụ phía sau đây.  
**Selectors** nhận vào hai tham số, mã định danh duy nhất và hàm **get** trả về dữ liệu được tính toán.

* Hooks

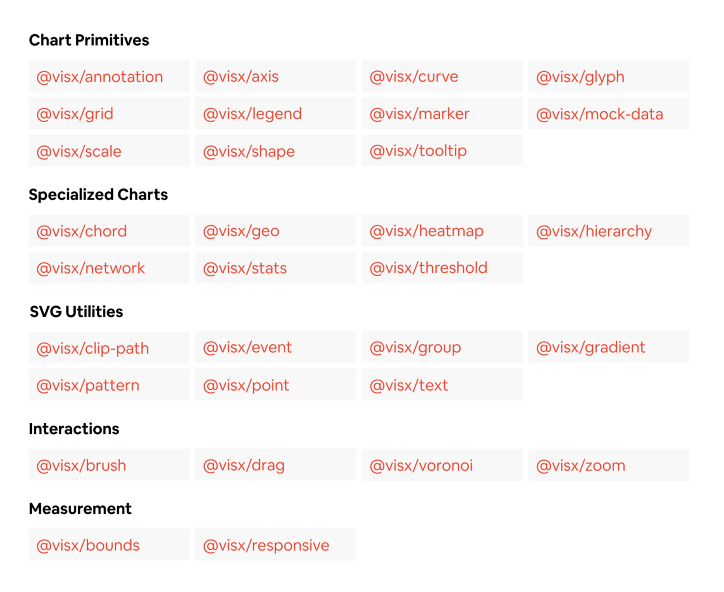
Component sẽ tương tác với dữ liệu thông qua các hooks được cung cấp bởi recoil, chúng ta sẽ tìm hiểu kĩ hơn ở phần sau.

### 4.2.3. Visx

Visx là một thư viện hỗ trợ React trong việc vẽ các biểu đồ kết hợp giữa D3 và React. Được sử dụng thông qua các gói và được viết bằng TypeScript một ngôn ngữ sinh ra để nâng cấp cho JavaScript.

**Ưu điểm của visx:**

* **Giảm kích thước gói.**visx được chia thành nhiều gói. Bắt đầu với quy mô nhỏ và chỉ sử dụng những gì bạn cần.
* **Không cố ý vào mục đích.**Mang đến giải pháp quản lý trạng thái, thư viện hoạt ảnh hoặc CSS-in-JS của riêng bạn. Tỷ lệ cược là tốt, ứng dụng React của bạn đã có ý kiến ​​về cách hoạt động, chủ đề hoặc kiểu được thực hiện. visx cẩn thận không thêm một cái khác và tích hợp với tất cả chúng.
* **Không phải là một thư viện biểu đồ.**Khi bạn bắt đầu sử dụng các nguyên tắc trực quan hóa, bạn sẽ kết thúc việc xây dựng thư viện biểu đồ của riêng mình được tối ưu hóa cho trường hợp sử dụng của bạn. Bạn đang kiểm soát.



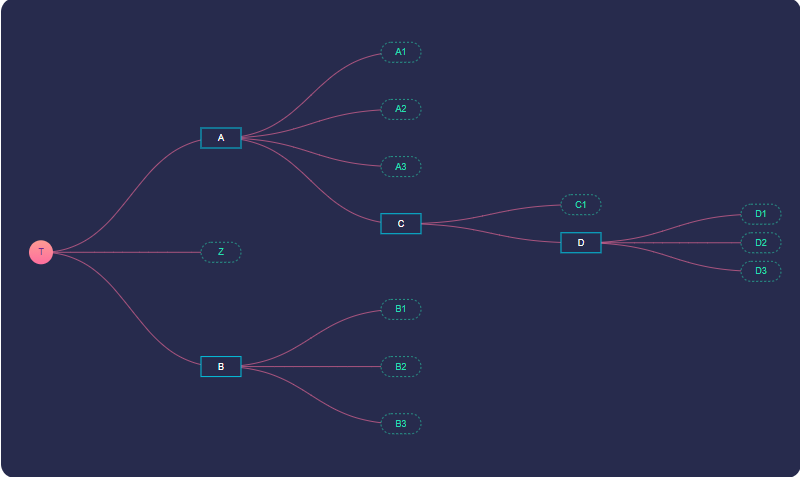
*Hình 1. visx là một bộ mô-đun gồm hơn 30 gói riêng biệt cung cấp các giải pháp trừu tượng cho các tác vụ phổ biến trong kỹ thuật trực quan hóa React.*

**Cách sử dụng visx trong D3**

Một số gói sử dụng D3 để tính toán bố cục và toán học, đồng thời phản ánh chức năng gói D3 bên dưới với API React khai báo - như @ visx / axis, @ visx / geo, @ visx / hierarchy và @ visx / shape. Các gói khác thay thế chức năng thao tác DOM của D3 và mang lại các nguyên tắc tương tác hạng nhất cho React - chẳng hạn như @ visx / brush, @ visx / drag và @ visx / zoom.

Ví dụ:

Sử dụng visx để lưu dữ các thành phần thuốc



Cần import các gói sau:

[@visx/gradient](https://airbnb.io/visx/docs/gradient) [@visx/group](https://airbnb.io/visx/docs/group) [@visx/hierarchy](https://airbnb.io/visx/docs/hierarchy) [@visx/responsive](https://airbnb.io/visx/docs/responsive) [@visx/shape](https://airbnb.io/visx/docs/shape)

import { Group } from '@visx/group';

import { hierarchy, Tree } from '@visx/hierarchy';

import { LinearGradient } from '@visx/gradient';

import { pointRadial } from 'd3-shape';

## 4.3. Backend với Nodejs

Hiện nay bên cạnh các cơ sở dữ liệu SQL dạng bảng đang rất phổ biến như MySQL, SQLServer,… thì cơ sở dữ liệu NoSQL dạng JSON cũng đang rất được ưa chuộm với những đặc điểm như không yêu cầu phép nối, không cần lược đồ cố định và dễ dàng mở rộng.Nó phù hợp với các cơ sở dữ liệu thời gian thực và dữ liệu khổng lồ đang được các ông lớn tin dùng như: FaceBook, Twitter,….

Và chúng tôi sẽ sử dụng Nodejs, Expressjs, MongoDB để xây dựng nó.

### 4.3.1. Nodejs

NodeJS là một nền tảng được xây dựng trên V8 JavaScript Engine – trình thông dịch thực thi mã JavaScript, giúp xây dựng các ứng dụng web một cách đơn giản và dễ dàng mở rộng. Hay có thể nói Nodejs là môi trường để thực thi javascript.

NodeJS được phát triển bởi Ryan Dahl vào năm 2009 và có thể chạy trên nhiều hệ điều hành khác nhau: OS X, Microsoft Windows, Linux.

**Đặc điểm cơ bản của Nodejs:**

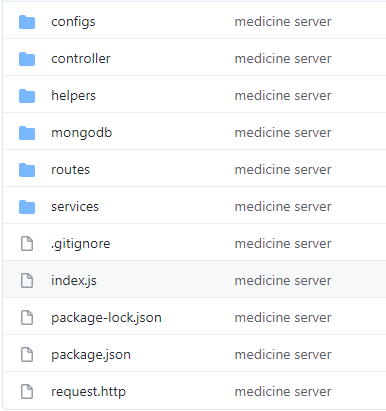
Nodejs sở hữu khá nhiều đặc điểm cơ bản, nổi trội và ưu việt. Hiểu thêm về các đặc điểm này sẽ giúp bạn có hiểu biết vững vàng hơn về Nodejs, đồng thời đưa ra quyết định sử dụng nền tảng này chính xác.

* **Không đồng bộ:** Đặc điểm đầu tiên của Nodejs là tính bất đồng bộ. Node.js không cần đợi [API](https://bizfly.vn/techblog/api-la-gi.html) trả dữ liệu về, vậy nên mọi APIs nằm trong thư viện Node.js đều không được đồng bộ, hiểu đơn giản là chúng không hề blocking (khóa). Server có cơ chế riêng để gửi thông báo và nhận phản hồi về các hoạt động của Node.js và API đã gọi.
* **Tốc độ nhanh:** Với phần core phía dưới lập trình gần như toàn bộ bằng ngôn ngữ C++, kết hợp với V8 Javascript Engine mà Google Chrome cung cấp, tốc độ vận hành, thực hiện code của thư viện Node.js diễn ra rất nhanh.
* **Đơn giản - Hiệu quả:** Tiến trình vận hành của Node.js đơn giản song lại mang đến hiệu năng cao nhờ ứng dụng mô hình single thread và các sự kiện lặp. Một loạt cơ chế sự kiện cho phép server trả về phản hồi bằng cách không block, đồng thời tăng hiệu quả sử dụng. Các luồng đơn cung cấp dịch vụ cho nhiều request hơn hẳn Server truyền thống.
* **Không đệm**: Nền tảng Node.js không có vùng đệm, tức không cung cấp khả năng lưu trữ dữ liệu buffer.

**NodeJS làm việc như thế nào?**

* Ý tưởng chính của Node js là sử dụng non-blocking, hướng sự vào ra dữ liệu thông qua các tác vụ thời gian thực một cách nhanh chóng.
* Node js có khả năng mở rộng nhanh chóng, khả năng xử lý một số lượng lớn các kết nối đồng thời bằng thông lượng cao.
* Nếu như các ứng dụng web truyền thống, các request tạo ra một luồng xử lý yêu cầu và chiếm RAM của hệ thống. Việc tài nguyên của hệ thống sẽ được sử dụng không hiệu quả.
* Chính vì lẽ đó giải pháp mà Node js đưa ra là sử dụng luồng đơn (Single-Threaded), kết hợp với non-blocking I/O để thực thi các request, cho phép hỗ trợ hàng chục ngàn kết nối đồng thời.

Soure tree của ứng dụng demo:



### 4.3.2. ExpressJs

Nó là một framework được xây dựng trên nền tảng của **Nodejs**. Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile. **Expressjs** hỗ trợ các method HTTP và midleware tạo ra API vô cùng mạnh mẽ và dễ sử dụng với các chức năng chính:

* Thiết lập các lớp trung gian để trả về các HTTP request.
* Define router cho phép sử dụng với các hành động khác nhau dựa trên phương thức HTTP và URL.
* Cho phép trả về các trang HTML dựa vào các tham số.

### 4.3.3. MongoDB

**MongoDB** đã trở thành một trong những NoSQL database nổi trội nhất bấy giờ, được dùng làm backend cho rất nhiều website như eBay, SourceForge và The New York Times.

**Các feature của MongoDB gồm có:**

* **Các ad hoc query:** hỗ trợ search bằng field, các phép search thông thường, regular expression searches, và range queries.
* **Indexing:** bất kì field nào trong BSON document cũng có thể được index.
* **Replication:** có ý nghĩa là “nhân bản”, là có một phiên bản giống hệt phiên bản đang tồn tại, đang sử dụng. Với cơ sở dữ liệu, nhu cầu lưu trữ lớn, đòi hỏi cơ sở dữ liệu toàn vẹn, không bị mất mát trước những sự cố ngoài dự đoán là rất cao. Vì vậy, người ta nghĩ ra khái niệm “nhân bản”, tạo một phiên bản cơ sở dữ liệu giống hệt cơ sở dữ liệu đang tồn tại, và lưu trữ ở một nơi khác, đề phòng có sự cố.
* **Aggregation:** Các Aggregation operation xử lý các bản ghi dữ liệu và trả về kết quả đã được tính toán. Các phép toán tập hợp nhóm các giá trị từ nhiều Document lại với nhau, và có thể thực hiện nhiều phép toán đa dạng trên dữ liệu đã được nhóm đó để trả về một kết quả duy nhất. Trong SQL, count (\*) và GROUP BY là tương đương với Aggregation trong MongoDB.
* **Lưu trữ file:** MongoDB được dùng như một hệ thống file tận dụng những function trên và hoạt động như một cách phân phối qua sharding.

Ví dụ cho Embedded Documents:

* \_id: ObjectId("5099803df3f4948bd2f98391"),
* name: becberin clorid,
* amout: 50,

Các kiểu dữ liệu trong MongoDB

MongoDB hỗ trợ các kiểu dữ liệu sau:

* **Chuỗi:** Đây là kiểu dữ liệu được sử dụng phổ biến nhất để lưu giữ dữ liệu. Chuỗi trong MongoDB phải là UTF-8 hợp lệ.
* **Số nguyên:** Kiểu dữ liệu này được sử dụng để lưu một giá trị số. Số nguyên có thể là 32bit hoặc 64bit phụ thuộc vào Server của bạn.
* **Boolean**: Kiểu dữ liệu này được sử dụng để lưu giữ một giá trị Boolean (true/false).
* **Double**: Kiểu dữ liệu này được sử dụng để lưu các giá trị số thực dấu chấm động.
* **Min/ Max keys:** Kiểu dữ liệu này được sử dụng để so sánh một giá trị với các phần tử BSON thấp nhất và cao nhất.
* **Mảng:** Kiểu dữ liệu này được sử dụng để lưu giữ các mảng hoặc danh sách hoặc nhiều giá trị vào trong một key.
* **Timestamp:** Đánh dấu thời điểm một Document được sửa đổi hoặc được thêm vào.
* **Object:** Kiểu dữ liệu này được sử dụng cho các Document được nhúng vào.
* **Null:** Kiểu dữ liệu này được sử dụng để lưu một giá trị Null.
* **Symbol:** Kiểu dữ liệu này được sử dụng giống như một chuỗi
* **Date:** Kiểu dữ liệu này được sử dụng để lưu giữ date và time hiện tại trong định dạng UNIX time.
* **Object ID:** Kiểu dữ liệu này được sử dụng để lưu giữ ID của Document.
* **Binary data:** Kiểu dữ liệu này được sử dụng để lưu giữ dữ liệu nhị phân.
* **Code:** Kiểu dữ liệu này được sử dụng để lưu giữ JavaScrip code vào trong Document.
* **Regular expression:** Kiểu dữ liệu này được sử dụng để lưu giữ Regular Expresion.

**Ưu điểm của mongodb**

* Dữ liệu lưu trữ phi cấu trúc, không có tính ràng buộc, toàn vẹn nên tính sẵn sàng cao, hiệu suất lớn và dễ dàng mở rộng lưu trữ.
* Dữ liệu được caching (ghi đệm) lên RAM, hạn chế truy cập vào ổ cứng nên tốc độ đọc và ghi cao.

## 4.4. Xây dựng ứng dụng quản lý dược phẩm

### 4.4.1. Các chức năng chính

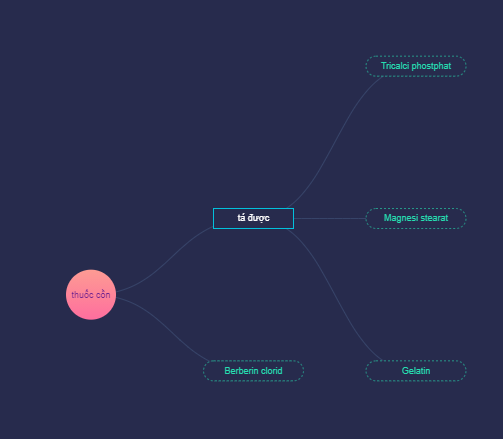
#### 4.4.1.1. Quản lý thành phần thuôc

Tại đây sẽ quản lý các thành phần tạo nên 1 viên thuộc với tên thuốc thành phần và sô lượng.

Mỗi viên thuốc có các thuộc tính sau:

* Tên
* Số lượng
* Id
* Các thành phần cấu tạo

Mỗi 1 đơn vị thuốc gồm các thành phần khác nhau tạo nên. Các thành phần con lại có các thành phần nhỏ hơn tạo nên. Mỗi thành phần có nồng độ khác nhau. Các thành phần được hiện thị dưới dạng cây như sau:



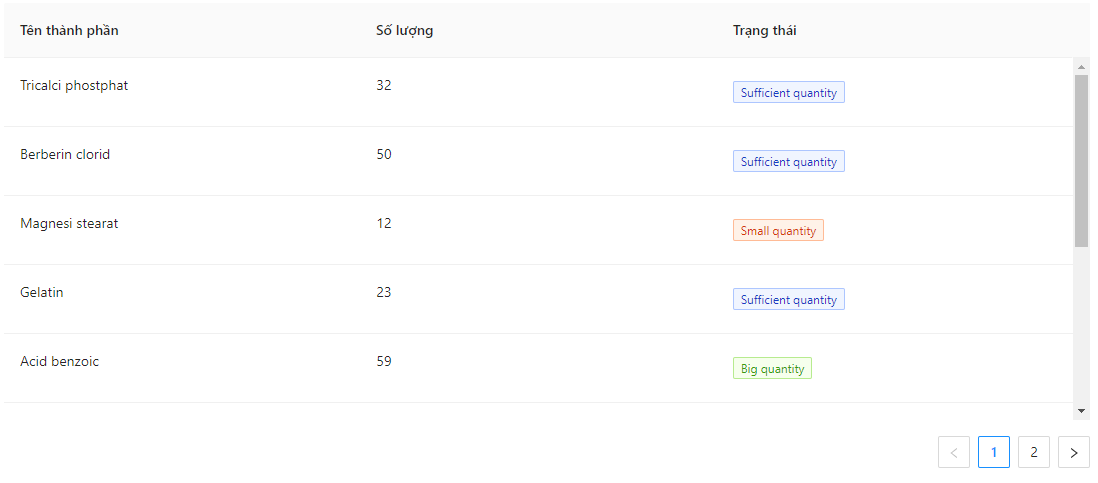
*Hình 2: thành phần thuốc dưới dạng cây*

#### 4.4.1.2. kho thành phần

Kho thành phần gồm các thuộc tính:

* tên thành phần
* số lượng
* trạng thái (còn, hết hay còn ít).

Kho thành phần là nơi lưu trữ các thành phần dùng để cấu tạo nên một loại thuốc với số lượng còn lại trong kho của nó. Và cảnh báo khi số lượng còn quá ít để dễ dàng kiểm soát nhập nguyên liệu hoặc sản xuẩt thêm.



*Hình 3: Kho các thành phần*

#### 4.4.1.3 Kho thuốc

Kho thuốc bao gồm các thuộc tính

* Số thứ tự
* Tên sản phẩm (tên thuốc)
* Số lượng thuốc
* Trạng thái
* Tạo quy trình sản xuất

Cũng giống như kho dược phẩm, ngoài việc dễ dàng quản lý số lượng, trạng thái, ta có thể tạo quy trính sản xuất với mỗi loại dược phẩm chính.



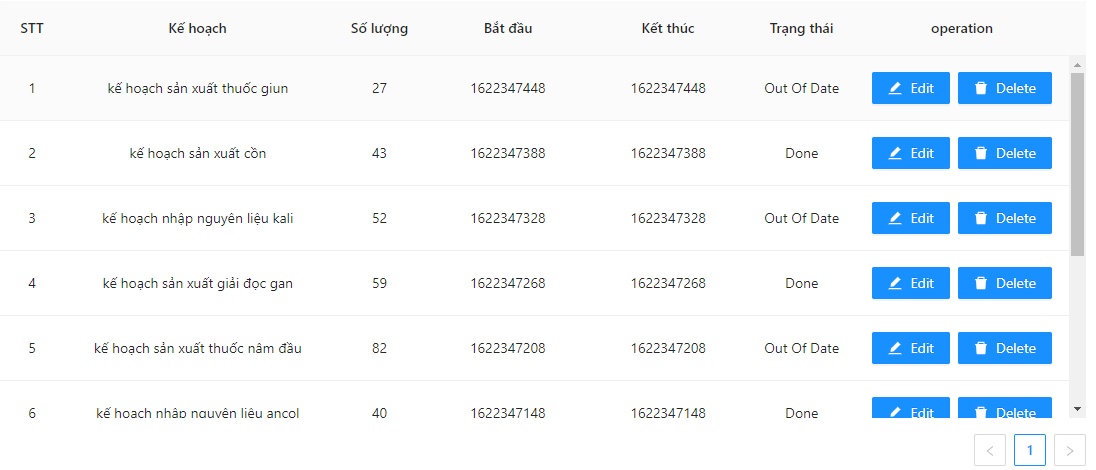
*Hình 4: Kho các dược phẩm*

#### 4.4.1.4. kế hoạch sản xuất

Trong phần kế hoạch sản xuất này. Chúng ta có thể quan sát các kế hoạch có sẵn và biết đc thời gian kết thúc và bắt đầu của kế hoạch cũng như trạng thái của nó. Sửa và xóa một kế hoạch khi chúng đã hoàn thành.

Kho thuốc bao gồm các thuộc

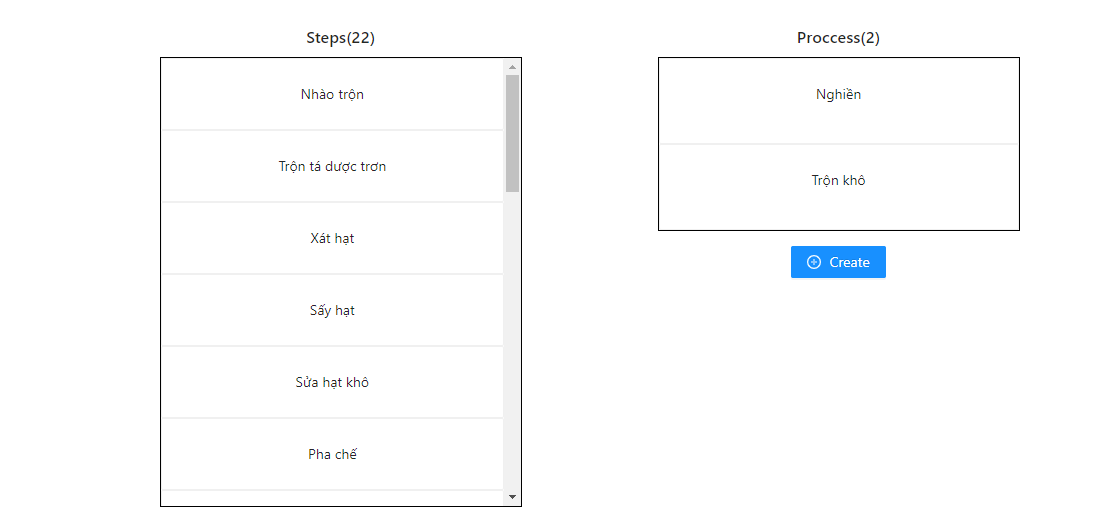
* Số thứ tự
* Tên kế hoạch
* Số lượng thuốc cần sản xuất
* Thời gian băt đầu và kết thúc
* Trạng thái (đã hoàn thành hay chưa)



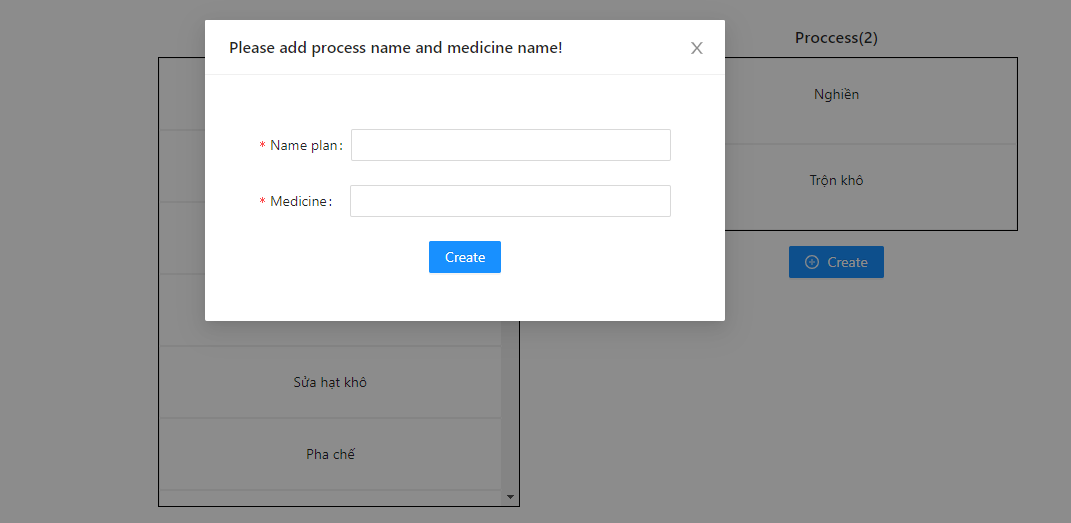
*Hình 5: Các kế hoạch sản xuất*

#### 4.4.1.5. Tạo quy trình sản xuất

Quy trình sản xuất được tạo ra bằng cách chọn và sắp xếp các step lại với nhau để tạo ra một quy trình hoàn chỉnh nhất. Và đặt tên cùng với dược phẩm đc tạo ra bới quy trình đó. Quy trình sau được thực hiện sau khi quy trình trước được thực hiện thành công, đây là điều kiện tiên quyết. Mỗi loại thuốc được thực hiện bằng các quy trình khác nhau.



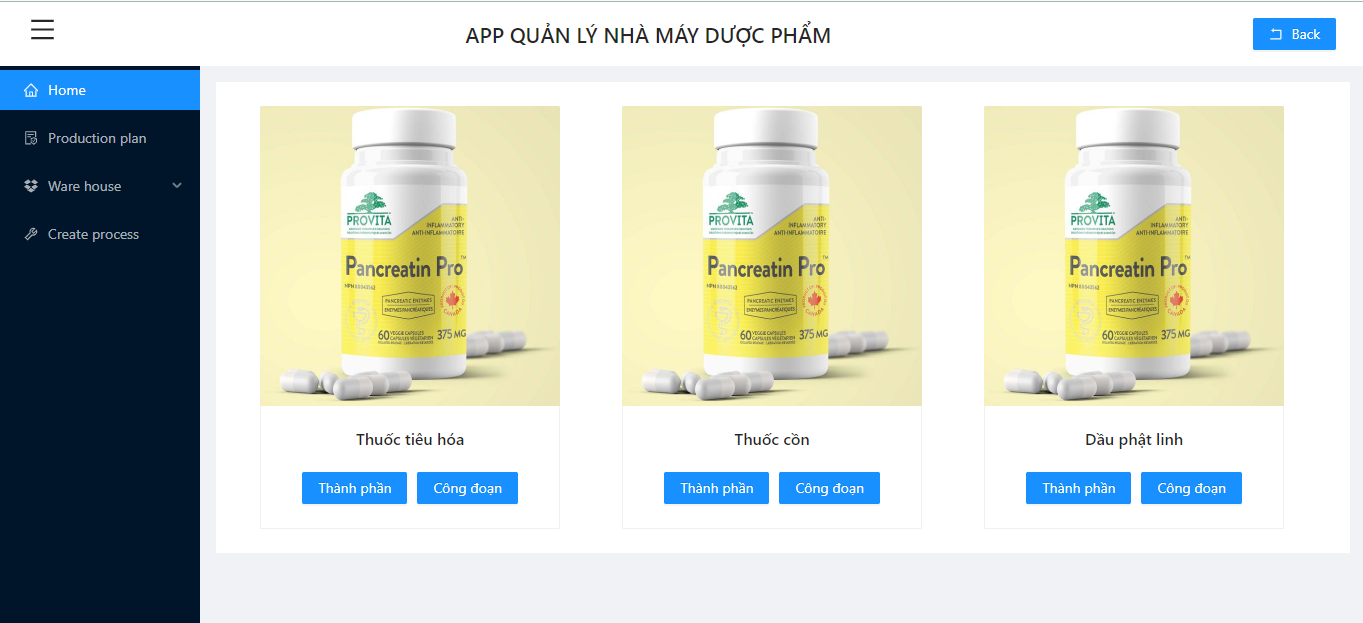
*Hình 6: Các step*



*Hình 7: Modal đặt tên*

#### 4.4.1.6. Quản lý chung

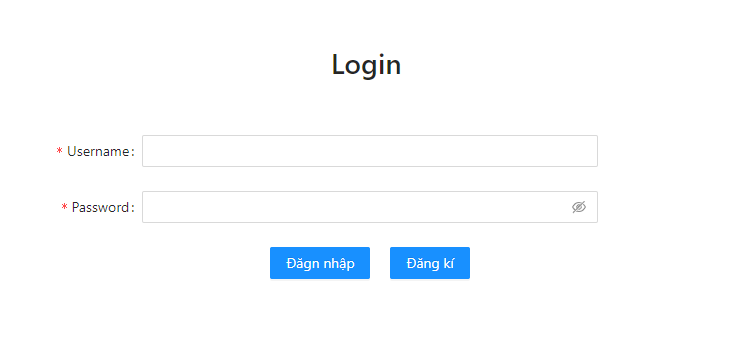
Quản lý chung cho ứng dụng, cho xem được các loại thuốc có trong danh mục thuốc của nhà máy.



*Hình 8: Quản lý chung dược phẩm*

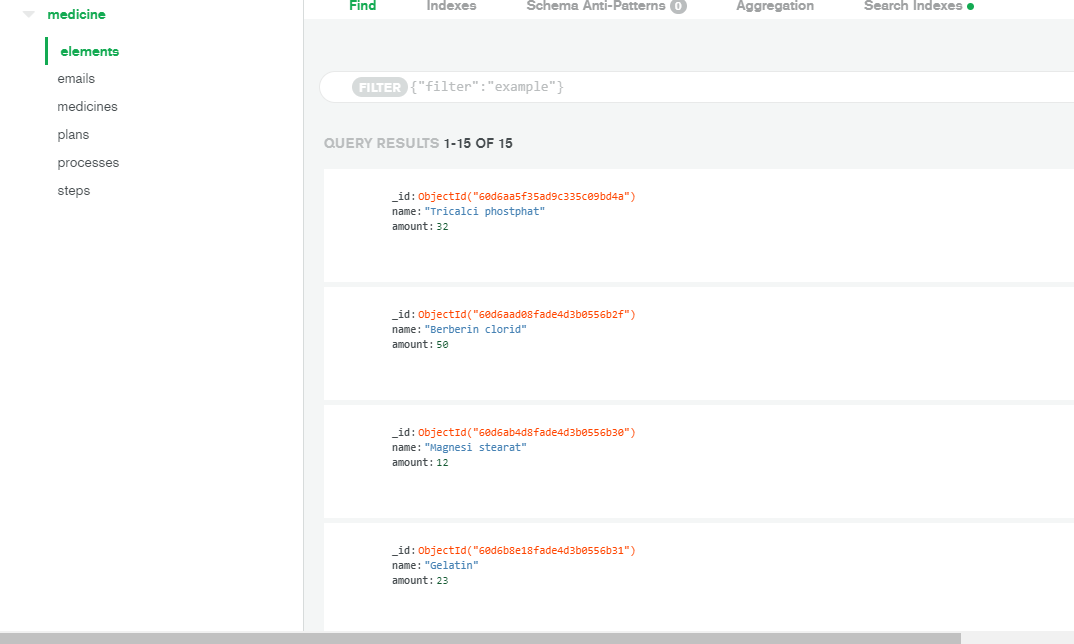
### 4.4.2. Giao diện khác của ứng dụng

Giao diện đăng nhập

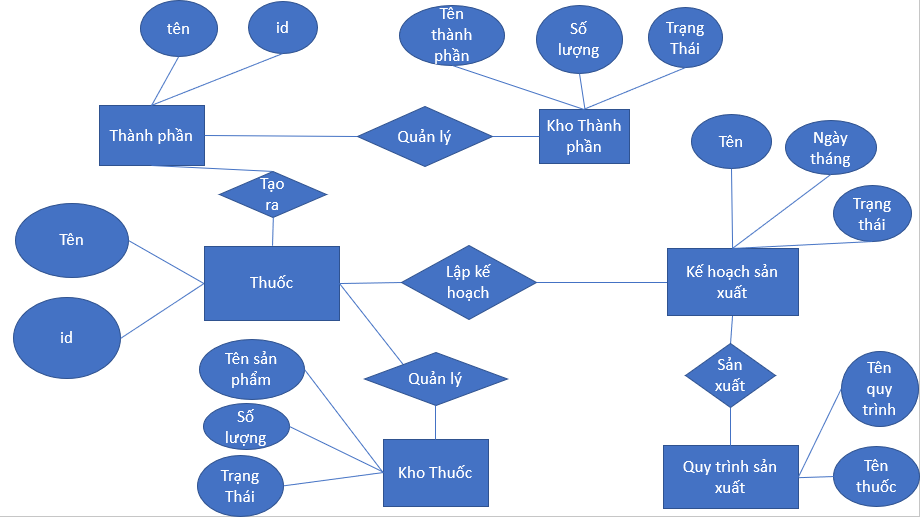


*Hình 9: Đăng nhập*

### 4.4.3. Database của mongoDB



*Hình 9: MongoDB*

**

*Hình 10: Mô hình thực thể liên kết*

# **Phần V: Kết luận và đề nghị**

## 5.1. Kết luận

Sau một thời gian thực tế tại công ty cũng như trong quá trình thực hiện đề tài thực tập, nhóm đã đạt được một số kết quả như sau:

- Kiến thức chuyên môn:

+ Nắm được các kiến thức về Front-end

+ Học được kiến thức về ReactJs

- Kinh nghiệm thực tế

+ Tiếp xúc và tham gia môi trường làm việc tại doanh nghiêp.

+ Vận dụng kiến thức đã học vào trong thực tế.

- Kỹ năng mềm

+ Tự tin hơn trong giao tiếp.

+ Cải thiện kỹ năng làm việc nhóm.

## 5.2. Đề nghị

Trong quá trình thực hiện đề tài nhóm đã cố gắng hết sức hoàn thiện chương trình demo nhưng do điều kiện thời gian có hạn và kinh nghiệm thực thế, kiến thức còn hạn hẹp nên chương trình chưa được hoàn thiện tốt nhất và vẫn còn nhiều thiếu sót. Vì vậy hướng phát triển tiếp theo dự kiến của nhóm là:

- Cải thiện thêm về phần giao diện và chức năng.

- Xây dựng thêm hệ thống quản lý một cách tối ưu, hoàn chỉnh hơn.

# **Phần VI. Tài liệu tham khảo:**

1. *”Document Reactjs”*: <https://reactjs.org>
2. *” Document Ant.design”:* <https://ant.design>
3. *” Document Recoil”:* <https://recoiljs.org>
4. *” Document Nodejs”:* <https://nodejs.org/en>
5. *” Document Expressjs”:* <https://expressjs.com>
6. *” Document MongoDB”:* <https://www.mongodb.com>
7. *” Document Visx”:* <https://airbnb.io/visx>
8. [Trey Huffine](https://medium.com/@treyhuffine?source=post_page-----1e7d07966d6c--------------------------------) (2019),” *A Recap of Frontend Development in 2019*”, [A Recap of Frontend Development in 2019 | by Trey Huffine | Level Up Coding (gitconnected.com)](https://levelup.gitconnected.com/a-recap-of-frontend-development-in-2019-1e7d07966d6c)
9. [DAISY](javascript:void(0)) (2021),”*Tìm hiểu về Node.js***”,** [Tìm hiểu về Node.js - Tự Học Lập Trình (tuhoclaptrinh.edu.vn)](https://tuhoclaptrinh.edu.vn/bai-viet/tim-hieu-ve-nodejs-428.html)