TRƯỜNG ĐẠI HỌC KINH TẾ

**KHOA THỐNG KÊ – TIN HỌC**

–––––––––––––––––––––––––––––––



**BÁO CÁO THỰC TẬP NGHỀ NGHIỆP**

**NGÀNH HỆ THỐNG THÔNG TIN QUẢN LÝ**

**CHUYÊN NGÀNH QUẢN TRỊ HỆ THỐNG THÔNG TIN**

**TÌM HIỂU VỀ REACTJS VÀ**

**XÂY DỰNG WEBSITE NIGHTMOVIE**

Sinh viên thực hiện : Thái Đức Tân

Lớp : 45K21.1

Đơn vị thực tập : Công ty phần mềm Gear inc.

Cán bộ hướng dẫn : Nguyễn Thị Thanh Vân

Giảng viên hướng dẫn : ThS. Cao Thị Nhâm

**Đà Nẵng, 8/2022**

**NHẬN XÉT CỦA ĐƠN VỊ THỰC TẬP**

Họ và tên sinh viên: Thái Đức Tân

Lớp: 45K21.1 Khoa: Thống kê – tin học

Trường: Đại học Kinh tế - Đại học Đà Nẵng

Thực tập từ ngày: 06/ 06/ 2022 đến ngày: ........./ .……./ 2022

Tại: Công ty phần mềm Gear Inc

Địa chỉ: 52 Nguyễn Văn Linh, Đà Nẵng, Việt Nam

Sau quá trình thực tập tại đơn vị của sinh viên, chúng tôi có một số nhận xét, đánh giá như sau:

**1. Về thái độ, ý thức, đạo đức, kỷ luật**

**2. Kiến thức chuyên môn**

**3. Khả năng hòa nhập và thích nghi với công việc**

**4. Trách nhiệm, sáng tạo trong công việc**

**5. Các nhận xét khác**

**Đánh giá chung:**

**Điểm:**

……….., ngày .......tháng ......năm 2022

**Xác nhận của đơn vị thực tập**

# LỜI CẢM ƠN

# LỜI CAM ĐOAN

Quy định:

Soạn thảo trên trang A4 (trang dọc, lề trái: 3.5cm; trên, phải, dưới: 2.5 cm), Font Times New Roman, canh đều 2 bên, size 13, cách dòng 1.5, cách đoạn trên 6pt, cách đoạn dưới 3pt, hàng đâu tiên lùi vào 1.27 cm; hình và bảng soạn theo caption, chèn trích dẫn chéo (Cross-reference) cho bảng và hình; các danh mục hình, bảng, mục lục làm tự động; đánh số trang như file mẫu (bìa không có số trang, danh mục + mục lục số trang theo i, ii, iii…, nội dung chính theo 1,2,3…)

# MỤC LỤC

[NHẬN XÉT CỦA DOANH NGHIỆP THỰC TẬP ii](#_Toc98339732)

[LỜI CẢM ƠN iii](#_Toc98339733)

[LỜI CAM ĐOAN iv](#_Toc98339734)

[MỤC LỤC vi](#_Toc98339735)

[DANH MỤC HÌNH ẢNH viii](#_Toc98339736)

[DANH MỤC BẢNG BIỂU ix](#_Toc98339737)

[DANH MỤC CÁC TỪ VIẾT TẮT x](#_Toc98339738)

[LỜI MỞ ĐẦU 1](#_Toc98339739)

[CHƯƠNG 1. TỔNG QUAN … 2](#_Toc98339740)

[1.1. Mục 1.1 2](#_Toc98339741)

[1.1.1. Mục 1.1.1 2](#_Toc98339742)

[1.1.2. Mục 1.1.2 2](#_Toc98339743)

[1.2. Mục 1.2 2](#_Toc98339744)

[CHƯƠNG 2. LÝ THUYẾT 3](#_Toc98339745)

[2.1. Mục 2.1 3](#_Toc98339746)

[2.1.1. Mục 2.1.1 3](#_Toc98339747)

[2.1.2. Mục 2.1.2 3](#_Toc98339748)

[2.2. Mục 2.2 3](#_Toc98339749)

[CHƯƠNG 3. TRIỂN KHAI … 4](#_Toc98339750)

[3.1. Mục 3.1 4](#_Toc98339751)

[3.1.1. Mục 3.1.1 4](#_Toc98339752)

[3.1.2. Mục 3.1.2 4](#_Toc98339753)

[3.2. Mục 3.2 4](#_Toc98339754)

[CHƯƠNG 4. KẾT QUẢ 5](#_Toc98339755)

[4.1. Mục 4.1… 5](#_Toc98339756)

[4.2. Mục 4.2… 5](#_Toc98339757)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 6](#_Toc98339758)

[TÀI LIỆU THAM KHẢO 7](#_Toc98339759)

[PHỤ LỤC 8](#_Toc98339760)

*(Mục lục này chỉ là ví dụ)*

# DANH MỤC HÌNH ẢNH

[Hình 1.1 Ngôn ngữ lập trình Python 2](#_Toc98336120)

[Hình 4.1 Kiến trúc của mô hình 5](#_Toc98336121)

# DANH MỤC BẢNG BIỂU

[Bảng 4.1 Kiến trúc 5](#_Toc74235471)

# DANH MỤC CÁC TỪ VIẾT TẮT

**AI** : Artificial Intelligence

**…**

# LỜI MỞ ĐẦU

1. **Mục tiêu nghiên cứu của đề tài**

* Đề tài này nghiên cứu …

1. **Nhiệm vụ của đề tài**

* Nghiên cứu ...
* ...

1. **Phương pháp nghiên cứu**

* ...

1. **Đối tượng và phạm vi nghiên cứu**

* ...

1. **Kết cấu của đề tài**

Đề tài được tổ chức gồm phần mở đầu, x chương nội dung và phần kết luận...

* Mở đầu
* **Chương 1**:
* **Chương 2**:
* **Chương 3**:
* **Chương 4**:
* Kết luận và hướng phát triển

# CƠ SỞ LÝ THUYẾT

## Giới thiệu về React JS

React JS là thư viện Javascript được xây dựng bởi các kỹ sư của Facebook, đang được rất nhiều công ty nổi tiếng sử dụng để phát triển các sản phẩm của họ như Yahoo, Airbnb và tất nhiên là trong nội tại Facebook, Instagram. Nó phù hợp với các dự án lớn có tính mở rộng hơn là các dự án nhỏ.

*Hình 1.1: Ngôn ngữ React JS*

* Đặc tính của React JS

Tư tưởng React JS là xây dựng lên các components có tính tái sử dụng, dễ dàng cho việc chia nhỏ vấn đề, testing. Nó giúp chúng ta dễ dàng quản lý, mở rộng hệ thống, điều này nếu là Angular JS thì đòi hỏi cấu trúc, cách viết code phải thật tối ưu.

React JS luôn giữ các components ở trạng thái stateless (nhiều nhất có thể) khiến ta dễ dàng quản lý bởi nó chẳng khác gì một trang static HTML. Bản thân các components này không có trạng thái (state) nó nhận đầu vào từ bên ngoài và chỉ hiện thị ra dựa vào các đầu vào đó, điêu này lý giải tại sao nó lại mang tính tái sử dụng (reuse) và dễ dàng cho việc test.

* Thế mạnh của React JS

React JS là một framework hiển thị view chú ý đến hiệu năng (performance-minded). Rất nhiều đối thủ nặng ký về framework MVVM (Model-View-ViewModel) mất một thời gian lớn để hiển thị những lượng data lớn, như trong trường hợp những danh sách (list) và tương tự. Nhưng React đó không còn là vấn đề, vì nó chỉ hiển thị những gì thay đổi.

Một trong những điểm mạnh nữa của React JS là virtual DOM - thứ nằm ẩn bên trong mỗi view và là lí do khiến cho React đạt được hiệu năng tốt. Khi một view yêu cầu gọi, tất cả mọi thứ sẽ được đưa vào trong một bản sao ảo của DOM. Sau khi việc gọi hoàn thành, React tiến hành một phép so sánh giữa DOM ảo và DOM thật, và thực hiện những thay đổi được chỉ ra trong phép so sánh trên.

Ví dụ: nếu chúng ta đang xem một danh sách có 20 products được hiển thị bởi React, và chúng ta thay đổi product thứ 2, thì chỉ product đó được hiển thị lại, và 19 products còn lại vẫn giữ nguyên (không cần hiển thị lại hay reload lại trang). React đã dùng cái gọi là “DOM ảo” (“virtual DOM”) để tăng hiệu năng bằng cách xuất ra một hiển thị ảo, sau đó kiểm tra sự khác biệt giữa hiển thị ảo và những gì có trên DOM và tạo một bản vá (a patch).

## Một số khái niệm, kiến thức khi nghiên cứu React

### Khái niệm

* Component là gì?

React được xây dựng xung quanh các component, chứ không dùng template như các framework khác. Ta có thể tạo ra một component bằng cách extend Component như sau

*Ảnh có chứa văn bản

Mô tả được tạo tự độngHình 1.2: Component trong React Js*

Phương thức quan trọng nhất là render, phương thức này được trigger khi component đã sẵn sàng để được render lên trên page.

Trong hàm đó, bạn sẽ trả về một mô tả cho việc bạn muốn React render cái gì lên trên page. Như trong ví dụ ở trên, đơn giản mình muốn render 1 thẻ h1 chứa lời chào và tên.

Hàm render chính là mô tả cụ thể của UI tại bất cứ thời điểm nào. Vì thế nếu dữ liệu thay đổi, React sẽ take care việc update UI với dữ liệu tương ứng. Các bạn có thể hiểu đơn giản là, khi dữ liệu thay đổi, React sẽ tự động gọi hàm render để update lại UI.

* Multiple Components

Ảnh có chứa văn bản

Mô tả được tạo tự độngNếu muốn lồng nhiều component vào nhau, ta sẽ làm điều này trong lệnh return của phương thức render.

*Hình 1.3: Component lồng nhau*

* Prop

Props là viết tắt của properties. Chúng là các giá trị đơn hoặc các đối tượng có chứa 1 tập hợp các giá trị được chuyển đến Component. Nó sử dụng quy ước đặt tên tương tự như các thuộc tính của thẻ HTML.

Props là cách để các component giao tiếp với nhau. Props được truyền từ component cha.

Ảnh có chứa văn bản

Mô tả được tạo tự độngProps chỉ để đọc. Cho dù bạn khai báo component dưới dạng hàm hay class thì nó vẫn không bao giờ có thể sửa đổi props của chính nó.

*Hình 1.4: Phương thức Prop*

*Chú ý:* Khi một props được truyền vào component thì nó là bất biến tức là dữ liệu của nó không được thay đổi kiểu như một "pure" function.

* State

Một component trong React có 2 cách để lấy thông tin props và state. Không như props, state của 1 component không được truyền từ bên ngoài vào. Một component sẽ tự quyết định state của chính nó. Để tạo ra state cho component, ta sẽ phải khai báo nó trong hàm constructor.

*Ảnh có chứa văn bản

Mô tả được tạo tự độngHình 1.5: Phương thức State*

* Sự giống và khác nhau giữa Prop và State

*Giống:*

* + Props và state đều là plain JS objects
  + Props và state đều trigger render update khi thay đổi

*Khác:*

|  |  |  |
| --- | --- | --- |
|  | Prop | State |
| Can get initial value from parent Component? | Yes | Yes |
| Can be changed by parent Component? | Yes | No |
| Can set default values inside Component? | Yes | Yes |
| Can change inside Component? | No | Yes |
| Can change in child Components? | Yes | No |

### Kiến thức:

React JS là một framework đơn nhất, nhưng để xây dựng được ứng dụng hoàn chỉnh, chúng ta cần nhiều thứ hơn nữa. Đây là một số kiến thức khi muốn nghiên cứu React:

* Javascript:

Ngôn ngữ xây dựng nên React, yêu cầu mức độ trung bình: các khái niệm “object”, “prototype”, “callback” là bắt buộc, nếu bạn chưa biết gì, tôi nghĩ bạn nên đọc tài liệu tại đây và thực hành trong ít nhất 3 tuần trước khi quay lại để bắt đầu.

* Npm:

Công cụ quản lý package của Node.js và là phương pháp phổ biến nhất để các lập trình viên front-end và các nhà thiết kế chia sẻ mã nguồn Javascript. Nó bao gồm một hệ thống quản lý module gọi là CommonJS và cho phép bạn cài đặt bất cứ công cụ command-line nào được viết bằng Javascript.

Hầu hết các components, thư viện và công cụ tái sử dụng trong hệ sinh thái của React đều được triển khai như là thành các module CommonJS và có thể cài đặt thông qua npm.

* JavaScript “bundlers”

Hay còn gọi là các công cụ đóng gói JavaScript. Vì một vài lí do kĩ thuật mà module CommonJS (ví dụ, các thư viện trên npm) không thể sử dụng native trên trình duyệt. Bạn cần có một “công cụ đóng gói” Javascript để “đóng gói” các modules này thành các files .js mà có thể include được trên trang web qua thẻ <script/>.

* ES6:

Là phiên bản mới nhất của Javascript, vì còn quá mới nên chưa được hỗ trợ trên các trình duyệt, nhưng các công cụ đóng gói sẽ thực hiện phiên dịch với cấu hình đúng để có thể sử dụng được trên trình duyệt.

* Routing:

“Các ứng dụng single-page” trở nên khá rầm rộ ngày nay. Đây là những trang web mà chỉ thực hiện khởi tạo một lần, và khi người dùng ấn vào một đường dẫn hay nút ấn thì Javascript trên trang web sẽ thực thi mà trang web không cần phải reload lại. Việc quản lý địa chỉ trên thanh địa chỉ được thực hiện bởi router.

Router được sử dụng phổ biến nhất trong hệ sinh thái React là react-router. Nếu như bạn đang xây dựng một ứng dụng single-page, hãy sử dụng nó trừ khi bạn có lý do mà không thể sử dụng.

* Server Rendering:

Server rendering thường được gọi là “toàn bộ” hay “đồng bộ” JS. Điều đó có nghĩa là bạn có thể lấy bất cứ React components nào và render chúng thành HTML tĩnh trên server. Điều này cải thiện được tốc độ ban đầu vì người dùng không cần phải chờ JS thực hiện download để có thể thấy được UI ban đầu, và React có thể tái sử dụng HTML đã được render phía server, vì vậy mà không cần xử lý ở phía client.

Chỉ cần sử dụng server rendering khi ta thấy việc render ban đầu trở nên quá chậm hoặc muốn cải thiện thứ hạng tìm kiếm. Mặc dù Google bây giờ đã thực hiện index các nội dung render phía client, nhưng tại thời điểm tháng một 2016, mỗi khi được lấy ra đo lường thì thấy rõ thứ hạng bị ảnh hướng đi xuống, có khả năng là do vi phạm về tốc độ render phía client.

Việc render phía server vẫn cần nhiều công cụ để làm cho nó trở nên chính xác. Vì việc này mặc định được hỗ trợ trong các React components mà không cần quan tâm tới việc render phía server, chúng ta nên xây dựng ứng dụng trước và quan tâm tới việc render phía server sau. Ta sẽ không cần thiết phải viết lại tất cả các components để xử lý việc đó.

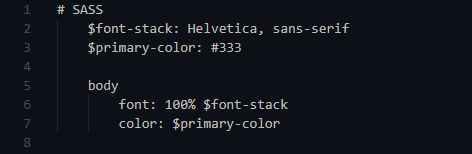
## Giới thiệu tổng quan về SCSS:

### SASS/SCSS là gì?

Theo Wekipekia, CSS là ngôn ngữ tạo phong cách cho trang web – Cascading Style Sheet language, được dùng để miêu tả cách trình bày các tài liệu viết bằng ngôn ngữ HTML và XHTML.

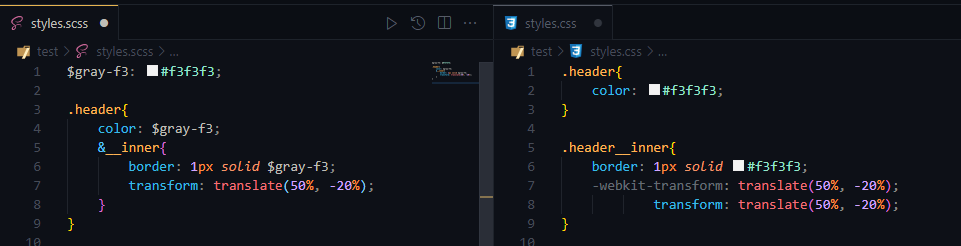
SASS/SCSS là một chương trình tiền xử lý CSS (CSS preprocessor), nó giúp chúng ta viết CSS theo cách của một ngôn ngữ lập trình, có cấu trúc rõ ràng, rành mạch, dễ phát triển và bảo trì code hơn. Ngoài ra nó có rất nhiều các thư viện hỗ trợ kèm theo giúp chúng ta viết code CSS một cách dễ dàng vào đơn giản hơn.

SASS là có phần mở rộng là *.sass* Sử dụng cách thụt đầu dòng để phân tách các khối, sử dụng xuống dòng để phân biệt rules (thực tế thì SASS không được sử dụng nhiều trong lập trình).



*Hình 1.6: Ví dụ về SASS*

SCSS phần mở rộng là *.scss* thực chất cũng giống với SASS chỉ là khác nhau một chút về cú pháp như một bên sử dụng dấu ngoặc nhọn {} (SCSS) một bên thì không (SASS), dấu chấm phải ở cuối ...

**

*Hình 1.7: Bên trái là SCSS, bên phải là CSS*

### Các tính năng cơ bản của SCSS:

##### Biến – Variables

Ảnh có chứa văn bản, màn hình, ảnh chụp màn hình, trong nhà

Mô tả được tạo tự độngSử dụng biến với SCSS vô cùng cơ bản, chỉ cần đặt tên cho biến - bắt đầu bằng $, gán cho nó 1 giá trị là xong. Sau đó, chúng ta có thể gọi nó ra để sử dụng bất cứ lúc nào. Và khi cần, chúng ta có thể thay đổi giá trị của biến trong quá trình sử dụng.

*Hình 1.8: So sánh về biến của CSS và SCSS*

##### Xếp chồng - Nested Rules

Ảnh có chứa văn bản

Mô tả được tạo tự độngNhư chúng ta đã biết thì để truy vấn các đối tượng HTML trong CSS ta sử dụng cú pháp selector. Nhưng khi trong một trang HTML mà có nhiều phân cấp cha con thì việc truy cập đến một đối tượng HTML thực sự rất rối, vì vậy SCSS cung cấp cho chúng ta một cách để truy cập dễ dàng hơn đó là Nested Rule.

*Hình 1.9: Ví dụ so sánh xếp chồng thẻ a và thẻ p từ HTML*

Khi sử dụng nested rules của SCSS thì chúng ta viết sẽ trở nên ngắn gọn hơn nhìn cũng gần như cấu trúc phân cấp của HTML mà khi biên dịch ra CSS để sử dụng thì kết quả hoàn toàn như nhau so với khi chúng ta viết CSS thông thường.

# LÝ THUYẾT

## Mục 2.1

Nội dung văn bản….

### Mục 2.1.1

### Mục 2.1.2

## Mục 2.2

# TRIỂN KHAI …

## Mục 3.1

### Mục 3.1.1

### Mục 3.1.2

## Mục 3.2

# KẾT QUẢ

## Mục 4.1…

## Mục 4.2…

Kết quả được xây dựng dựa trên ngôn ngữ lập trình thể hiện như (Bảng 4.1).

Bảng 4.1 Kiến trúc

|  |  |  |
| --- | --- | --- |
| Lớp | Ý nghĩa | Tham số |
|  |  |  |
|  |  |  |
|  | | |

Hình 4.1 Kiến trúc của mô hình

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Đề tài đã thực hiện được …

# TÀI LIỆU THAM KHẢO

* + - 1. Tên tác giả, Tên tác giả… *Tên công trình*. Nơi xuất bản, năm, trang
      2. Tên tác giả, Tên tác giả… *Tên công trình*. Nơi xuất bản, năm, trang

# PHỤ LỤC