**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**



**TRẦN LONG PHÚ**

**TRIỂN KHAI ỨNG DỤNG DI ĐỘNG ĐẶT XE TRỰC TUYẾN VỚI MERN STACK**

**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH KHOA HỌC MÁY TÍNH**

**TP. HỒ CHÍ MINH, 2020**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**



**TRẦN LONG PHÚ**

**TRIỂN KHAI ỨNG DỤNG DI ĐỘNG ĐẶT XE TRỰC TUYẾN**

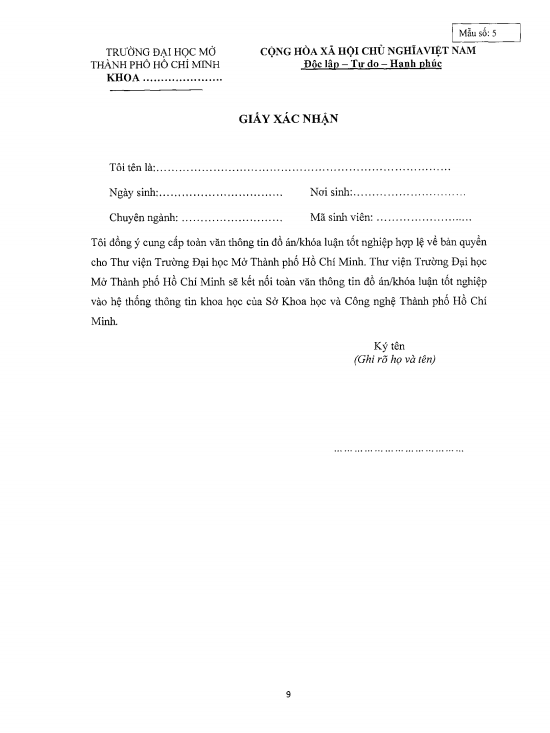
**Mã số sinh viên: 1751010108**

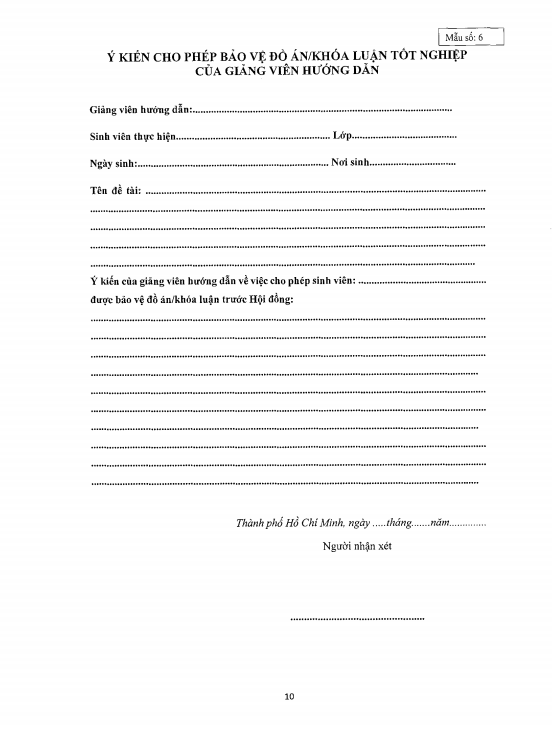
**KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH KHOA HỌC MÁY TÍNH**

**Giảng viên hướng dẫn: LÊ NGỌC HIẾU**

**TP. HỒ CHÍ MINH, 2020**





**MỤC LỤC**

[DANH MỤC HÌNH VẼ 10](#_Toc55092503)

[DANH MỤC BẢNG 11](#_Toc55092504)

[DANH MỤC TỪ VIẾT TẮT 12](#_Toc55092505)

[TÓM TẮT KHÓA LUẬN 13](#_Toc55092506)

[MỞ ĐẦU 14](#_Toc55092507)

[Chương 1. Giới thiệu tổng quan đề tài 15](#_Toc55092508)

[1.1. Lý do chọn đề tài 15](#_Toc55092509)

[1.2. Mục tiêu nghiên cứu: 15](#_Toc55092510)

[1.2.1. Về phía Mobile: 15](#_Toc55092511)

[1.2.2. Về phía Backend: 16](#_Toc55092512)

[Chương 2. GIỚI THIỆU VỀ REACT NATIVE 17](#_Toc55092513)

[2.1. Tổng quan về React Native ? 17](#_Toc55092514)

[2.1.1. Sơ lược về ReactJS 17](#_Toc55092515)

[2.1.2. Sự hình thành của React Native 17](#_Toc55092516)

[2.1.3. React Native khác với ReactJS điểm nào ? 18](#_Toc55092517)

[2.1.4. Các ưu và nhược điểm của React Native. 18](#_Toc55092518)

[2.1.5. Cách thức hoạt động của React Native ? 19](#_Toc55092519)

[2.1.6. Giới thiệu về công cụ Expo: 19](#_Toc55092520)

[2.2. Các kiến thức cơ bản của React Native 20](#_Toc55092521)

[2.2.1. Các kiến thức cơ bản trước khi nhập môn React Native 20](#_Toc55092522)

[2.2.2. Các khái niệm cơ bản của React Native 26](#_Toc55092523)

[Chương 3. GOOGLE MAP API 36](#_Toc55092524)

[3.1. Tổng quan về Google Map Api? 36](#_Toc55092525)

[3.1.1. Google Cloud là gì ? 36](#_Toc55092526)

[3.1.2. Google Map API là gì ? 36](#_Toc55092527)

[3.2. Cách đăng ký dịch vụ Google Map 38](#_Toc55092528)

[Chương 4. Xây dụng Back-end với ExpressJS 42](#_Toc55092529)

[4.1. Tổng quan về NodeJS 42](#_Toc55092530)

[4.1.1. NodeJS là gì ? 42](#_Toc55092531)

[4.1.2. Hướng dẫn cài đặt NodeJS 43](#_Toc55092532)

[4.2. Tổng quan về ExpressJS 44](#_Toc55092533)

[4.2.1. Giới thiệu về ExpressJS 44](#_Toc55092534)

[4.2.2. Cấu trúc của ExpressJS 44](#_Toc55092535)

[4.3. Tổng quan về MongoDB: 45](#_Toc55092536)

[4.3.1. MongoDB là gì ? 45](#_Toc55092537)

[4.3.2. Cách tạo cơ sở dữ liệu bằng MongoDB 45](#_Toc55092538)

**LỜI CẢM ƠN**

Để có thể hoàn thiện (đồ án/luận văn) tốt nghiệp này, cũng như các kết quả nghiên cứu của (đồ án/luận văn) này, em xin trân trọng cảm ơn thầy **TS. Lê Ngọc Hiếu** (Giảng viên hướng dẫn chính thức). Thầy đã tận tình và giúp đỡ cho em trong suốt quá trình nghiên cứu và thực hiện (đồ án/luận văn) này.

Ngoài ra, ---------------------------------------------------------------------------------------

Sau cùng, em xin gửi lời cảm ơn sâu sắc các bạn sinh viên cùng lớp học tập đã ủng hộ, động viên và giúp đỡ em vượt qua khó khăn trong suốt quá trình thực hiện (đồ án/luận văn).

TP.HCM, ngày -- tháng -- năm 20—

*Sinh viên thực hiện* (đồ án/luận văn)

**-----------------**

**LỜI CAM ĐOAN**

Tôi cam đoan rằng (đồ án/ luận văn): ***“-----------------------------”*** là đề tài do tôi thực hiện.

Ngoại trừ những tài liệu tham khảo được trích dẫn trong đồ án: này, Tôi cam đoan rằng toàn phần hay những phần nhỏ của đồ án này chưa từng được công bố hoặc được sử dụng để nhận bằng cấp ở những nơi khác.

Không có sản phẩm/nghiên cứu nào của người khác được sử dụng trong (đồ án/luận văn) này mà không được trích dẫn theo đúng quy định.

Đồ án này chưa bao giờ được nộp để nhận bất kỳ bằng cấp nào tại các trường đại học hoặc cơ sở đào tạo khác.

TP.HCM, ngày -- tháng -- năm 20--

*Sinh viên thực hiện* (đồ án/luận văn)

*-----------------------------------*

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

# DANH MỤC HÌNH VẼ

[Hình 1.1: Tên hình 1 3](#_Toc367742554)

# DANH MỤC BẢNG

[Bảng 1.1: Tên bảng 1 3](#_Toc367742567)

[Bảng 2.1: Tên bảng 1 4](#_Toc367742568)

# DANH MỤC TỪ VIẾT TẮT

RN React Native

JSX Javascript Extension

ES6 ECMA Script 6

CLI Command Line Input

IDE Integrated Development Environment

XML Extension HTML

HOC High Order Component

# TÓM TẮT KHÓA LUẬN

# MỞ ĐẦU

# Giới thiệu tổng quan đề tài

## Lý do chọn đề tài

Vào năm 2007, khi mà cố CEO của Apple, Steve Jobs, đã cho ra mắt chiếc smartphone đầu tiên trên thế giới mang tên Iphone 2G, chiếc điện thoại thổi bùng lên một cuộc cách mạng công nghệ toàn cầu. Và hơn 1 thập kỷ sau, cuộc cách mạng ấy đã thay đổi một số thứ trong cuộc sống của con người chúng ta: những tờ báo giấy được thay thế bẳng báo điện tử, Ví cũng được thay thế bằng ví điện tử, việc đi chợ hằng ngày cũng được thay thế bằng các ứng dụng shopping online,….

Ngành giao thông - vận tải cũng không phải là ngoại lệ. Không những thế, đây còn là ngành nghề thay đổi rõ ràng nhất. Ở Việt Nam, khi mà chưa có smartphone, việc đặt taxi hoặc là xe ôm phải tốn rất nhiền thời gian để tìm kiếm tài xế, cũng như là chi phí cũng rất là mắc. Thế nhưng bây giờ, những công việc ấy lại hết sức đơn giản, chỉ cần 1 chiếc smartphone có internet, thì sẽ có tài xế tới tận nhà để rước. Do đó, đề tài này sẽ xây dựng và triển khai một ứng dụng giống với cách ứng dụng đặt xe có sẵn như Grab, Gojek, Be,… để có thể hiểu được là cách thức hoạt động của các loại hình công nghệ này.

## Mục tiêu nghiên cứu:

### Về phía Mobile:

#### React Native:

Đây là một Framework đa nền tảng do Facebook phát triển. Nó cho phép các nhà phát triển ứng dụng di động vừa có thể thiết kế trên IOS và Android với sự trợ giúp của ngôn ngữ Javascript. Việc này rất có lợi vì chỉ cần 1 team là có thể tạo ra ứng dụng cho cả 2 nền tảng, thay vì phải phân ra làm 2 team, vừa nhanh, vừa tiết kiệm chi phí.

#### Google Maps API:

Là các API về bản đồ do chính google thiết kế. Các API này bao gồm:

* Google Map: Bản đồ đường đi trên thế giới
* Google Location API: chỉ định vị trí thông qua trục tung và trục hoành
* Google Direction API: Vẽ các đường đi hông qua 2 điểm đầu và điểm đến thông qua JSON.

Các API này sẽ giúp các lập trình viên tiết kiệm được rất nhiều thời gian vì chỉ việc gọi nhựng gì có sẵn của Google mà không cần phải tạo ra lại một ứng dụng map mới. Tuy nhiên, trong 3 API trên thì chỉ có Google Map là được Google cung cấp miễn phí, còn lại là phải trả phí.

### Về phía Backend:

#### NodeJS

Là nền tảng chạy trên môi trường V8 Javasciprt Runtime – Một trình thông dịch Javascript mạnh mẽ.

#### ExpressJS

Đây là một framework chạy trên nền tảng NodeJS.

#### MongoDB

Đây là một cơ sở dữ liệu mã nguồn mở NoSQL. Loại cơ sở dữ liệu này sẽ khắc phục được một số nhược điểm của các cơ sở dữ liệu truyền thống về tốc độ, tính năng, khả năng mở rộng.

#### Socket.io

Là một công cụ hỗ trợ gửi nhận dữ liệu real-time như là tin nhắn, game online,… Đây là 1 công cụ hữu dụng với ứng dụng đặt xe này, giúp cho tài xế và khách hàng có thể giao tiếp với nhau.

# GIỚI THIỆU VỀ REACT NATIVE

## Tổng quan về React Native ?

### Sơ lược về ReactJS

React (hay còn được gọi là ReactJS) là một thư viện do chính Facebook xây dựng và phát triển từ năm 2011. Từ đó đến nay, React đã có được một cộng đồng phát triển vững chắc không chỉ bên trong nội bộ của Facebook mà cả 1 cộng đồng rộng lớn bên ngoài và thư viện này luôn được sử dụng rộng rãi khắp nơi trên toàn thế giới, bên cạnh với AngularJS của Google và VueJS, một thư viện do 1 thành viên của Google phát triển.

### Sự hình thành của React Native

Quay trở lại với năm 2015, khi mà React Native vẫn chưa ra đời, các công ty phát triển ứng dụng di động thường phải chia ra làm 2 team để phát triển nên dự án: team Android và team IOS. Tất nhiên, để có thể duy trì được cả 2 team đó thì phải trả một mức phí khá cao và nếu đặt trường hợp bạn là một nhà phát triển Android hoặc là IOS, việc học lại 1 công nghệ nào khác cũng sẽ khá tốn thời gian và dễ sinh ra chán nản. Hiểu rõ được điều đó, Facebook đã phát triển ra một framework dựa trên công nghệ ReactJS trên, cho phép các lập trình viên vừa có thể tạo ứng dụng trên IOS, vừa có thể tạo ứng dụng trên Android, hoặc là tạo 1 ứng dụng PWA trên desktop.

Đây quả là một cuộc cách mạng mới trong ngành phát triển ứng dụng di động khi mà các công ty có thể tiết kiệm 1 khoảng lớn chi phí thay vì phải chia ra 2 team như mọi khi, họ chỉ cần 1 team duy nhất và các lập trình viên bên phía ReactJS cũng có thể qua và học React Native trong 1 thời gian ngắn vì cách viết trên 2 nền tảng này khá là tương đồng với nhau.

### React Native khác với ReactJS điểm nào ?

React Native được phát triển dựa trên ReactJS nên sẽ có một số điểm khá là tương đồng với nhau. Tuy nhiên, nó vẫn sẽ có 1 số điểm khác nhau so với người đàn anh của mình:

#### Cốt lõi

ReactJS là một thư viện được sinh ra để phát triển web, còn React Native được dùng để phát triển ứng dụng di động vừa trên Android, vừa trên IOS.

#### HTML – Điểm khác biệt lớn nhất

React Native sẽ không sử dụng HTML để hiển thị. Bởi vì như vậy cũng không khác gì việc nhúng một trang web vào 1 webview và tốc độ xử lý sẽ khá là chậm.

Thay vào đó, Facebook sẽ cung cấp cho các nhà phát triển các component như là <View><View/> sẽ thay thế <div></div>, <Text></Text> sẽ thay thế cho <span></span> hay <p></p>.

Và tất nhiên, do không sử dụng HTML, React Native cũng sẽ không sử dụng CSS để trang trí cho các component riêng của mình. Chúng ta có thể sử dụng Component Stylesheet bên trong thư viện React Native.

### Các ưu và nhược điểm của React Native.

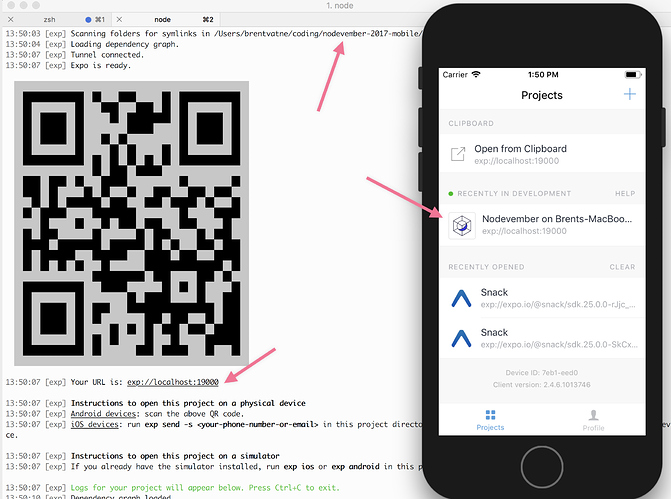
|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| * Tiết kiệm chi phí * Kiến thức dễ tiếp cận * Cộng đồng hỗ trợ mạnh mẽ * Hot reloading * Dễ bảo trì | * Bảo mật kém do được viết bằng Javascript * Công cụ debug khó sử dụng * Các Animation khó sử dụng * Các tập tin thường rất là nặng |

### Cách thức hoạt động của React Native ?

### Giới thiệu về công cụ Expo:

Đây là 1 công cụ hỗ trợ code React Native vô cùng tuyệt vời. Khác với react-native thuần, ta thường phải tốn nhiều thời gian để config environment JAVA\_HOME hoặc là vào Xcode để mở máy để debug. Nhưng với Expo, ta chỉ cần khởi động câu lệnh Expo start, 1 trình duyệt sẽ mở lên với port mặc định là 19000 cho tới 19002 sẽ được mở ra, cho phép chúng ta có thể mở ứng dụng trên Android 1 cách dễ dàng. Ngoài ra, Expo còn có sử dụng mã QR cho phép ta có thể dùng điện thoại để quét mã QR đó để khởi động chương trình, miễn là máy đang chạy Expo và điện thoại cùng chung 1 đường mạng. Đây là 1 điều hết sức tuyệt vời bởi vì không phải ai cũng có thể sắm cho mình được 1 chiếc máy MAC có Xcode để có thể debug trên điện thoại IOS, ta có thể dùng mã QR đó để quét, việc còn lại thì expo sẽ lo hết và ta sẽ có thể tạo ra app cho IOS mà không cần Xcode. Ngoài ra, Expo còn có thể hỗ trợ các developer upload ứng dụng của mình lên Play Store hay App Store mà không cần Xcode hay Android Studio.

Tuy nhiên, Expo vẫn có 1 số điểm yếu nhất định. Thứ nhất, cộng đồng hỗ trợ cho Expo vẫn chưa đủ mạnh, nên đôi lúc sẽ có 1 số thư viện bên ngoài không được hỗ trợ. Vấn đề này sẽ dẫn đến vấn đề thứ 2 là ta sẽ bị ràng buộc vào trong công cụ của họ. Đặt trường hợp nếu như ta vẫn muốn xài các thư viện mà expo không hỗ trợ hay là các thư viện yêu cầu , điều đó đồng nghĩa với việc là ta sẽ phải eject Expo ra và chạy React-Native thuần túy. Điều này sẽ dẫn đến việc tốn rất nhiều thời gian để config lại mọi thứ trong môi trường cho project React Native của bạn, hoặc là tệ hơn là sẽ phải tạo lại 1 project trắng và copy code cũ qua.



* **Cách cài đặt Expo:**

Như đã nói ở trên, cài đặt Expo rất là đơn giản, việc đầu tiên là ta chỉ cần cài đặt 1 CLI dành cho Expo thông qua câu lệnh npm install exp –global. Sau đó ta dùng câu lệnh expo init <tên project>. Thế là ta đã có 1 project React Native. Dùng lệnh cd <tên project> và sau đó nhập expo start, thế là ta đã có thể tự động khởi động ứng dụng react native bên trong điện thoại ảo. Còn nếu muốn dùng QR qua điện thoại thật, ta có thể tải App Expo Client trên Play Store hoặc App store và sau đó quét QR.

## Các kiến thức cơ bản của React Native

### Các kiến thức cơ bản trước khi nhập môn React Native

#### Javascript ES6 là gì ?

Đây là kiến thức vô cũng căn bản và thiết yếu để có thể bước chân vào được React Native hoặc là Nodejs sau này. Đây là 1 bộ nguyên tác và cũng là một chuẩn mực chung để viết framework, trong đó có cả React Native.

Dưới đấy là 1 số các tính năng của ES6:

* **Block-scoped:**

Có thể hiểu nôm na thì đây là 1 khối và các câu lệnh chỉ có tác dụng ở trong khuông khổ của 1 khối đó. Ví dụ:

var a = 19;

{

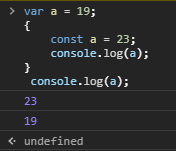
    const a = 23;

    console.log(a);

}

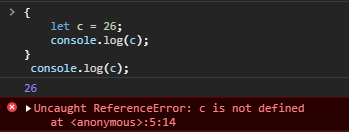
 console.log(a);

Và đây là kết quả sau khi complie:



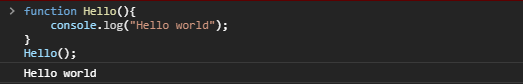
* Các câu lệnh thay đổi giá trị chỉ có tác dụng ở bên trong block, khi ra ngoài thì các giá trị đó sẽ trở về như cũ.

Điều này rất là hữu ích trong việc tiết kiệm bộ nhớ khi lập trình vì ta có thể khai báo 1 biến ảo ở bên trong đó để thực thi 1 số tính toán và sau đó nó sẽ tự động biến mất khi ra khỏi khối đó.



* **Arrow function**

Đối với cách viết function bình thường của Javascript :



Thì nó khá là dài dòng. Đối với ES6 cách viết này đã được rút gọn đi một phần và được gọi là Arrow function:



Với cách viết này, ta có thể thấy nó đơn giản hơn nhiều so với cách viết function thông thường và nó còn khá là tiện dụng với 1 số trường hợp như:

* Câu lệnh đơn:



Cách viết này tuân thử nguyên tắc là nếu chỉ có một câu lệnh thì ta có thể bỏ {}

* Một tham số:



Trong trường hợp chỉ có 1 tham số thì ta có thể bỏ dấu ngoặc

*Lưu ý:*

* *Khi viết Arrow Function, ta phải viết đúng với ràng buộc => như :*
* *()=>{}*
* *()=>{*

*//Câu lệnh*

*}*

* *()=>*

*{ //Câu lệnh}*

* *Còn các kiểu viết sau đây là sai:*

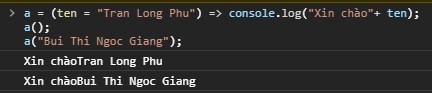
*()*

*=>{//Câu lệnh}*

* **Default Parameter:**

Đây là một tham số mặc định của một function khi mà ta gọi function đó lên mà không khai báo một tham số (parameter) nào thì nó sẽ mặc định gọi ra tham số đó ra cho người dùng:

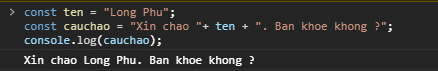
Ví dụ:

**

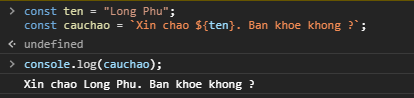
Như có thể thấy ở trên, khi không truyền một tham số nào hết thì nó sẽ gọi tham số mặc định do lập trình viên tự định nghĩa, còn nếu ta truyền vào thì nó sẽ dùng tham số do người dùng nhập vào.

* **Template Literals:**

Ở ngôn ngữ Javascipt bình thường, nếu muốn chèn một giá trị từ bên ngoài vào, ta chỉ có thể cộng chuỗi như thông thường. Ví dụ như:



Đối với cách viết trên, ta có thể thấy rằng nó khá là phức tạp. Vì thế, ở ES6 đã có 1 cách viết mới thay có cách viết trên bằng cách dùng dấu `` thay vì dùng “” hoặc ‘’ và thay vì cộng chuỗi, ta có thể bỏ ${bien}. Ví dụ như sau.



Và như đã thấy, kết quả cũng ra giống như trên nhưng cách viết sẽ đơn giản hơn và nhìn trực quan hơn so với cách viết thông thường.

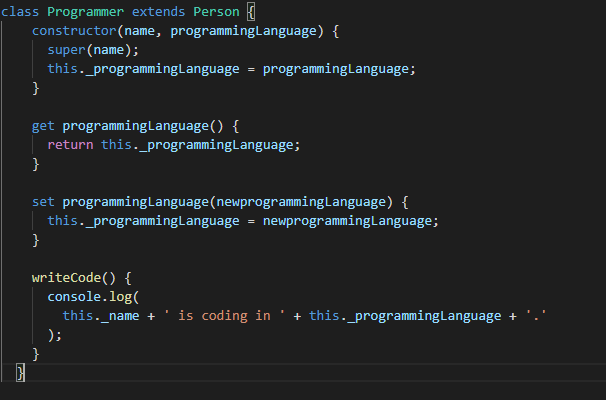
* **Classes**

Đây có lẽ là 1 trong những ưu điểm lớn của ES6 vì cách viết class khá là giống với các ngôn ngữ lập trình hướng đối tượng khác như C#, Java bao gồm có các phương thức khởi tạo, getter, setter, extends

Ví dụ:



Cách viết class của Javascipt ES6



Cách viết extends của Javascript 6

#### JSX là gì ?

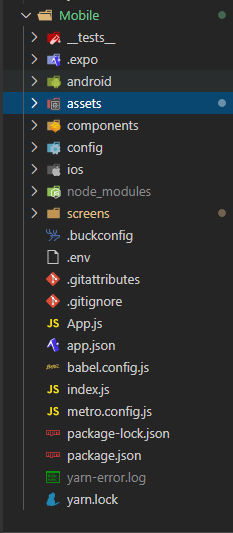
JSX là viết tắt của Javascipt Extension (Còn được gọi là Javascript mở rộng). Đây là một cú pháp không thể thiếu trong việc phát triển các trang web từ ReactJS hoặc là các ứng dụng từ React Native. Và sau đây là một ví dụ của JSX:

const hello = <Text>Xin chào</Text>;

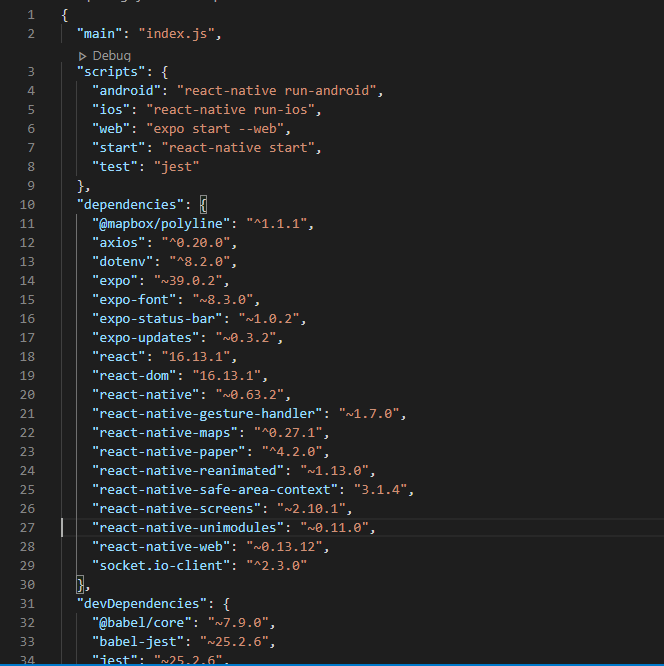
Ta có thể thấy cú pháp trên giống với 1 chuỗi bình thường trong Javascript, nhưng nó lại không phải là chuỗi vì Value của nó không có nháy đơn hoặc là nháy kép, nó cũng giống với 1 cú pháp XML (Phần mở rộng của HTML) nhưng nó lại không phải là XML. Đó là JSX và cú pháp này được sử dụng như chính tên gọi của nó là để mở rộng Javascript. Chính nhờ sự tiện ích này, Facebook đã sử dụng để viết các UI Component.

### Các khái niệm cơ bản của React Native

#### Cấu trúc thư mục của React Natuve

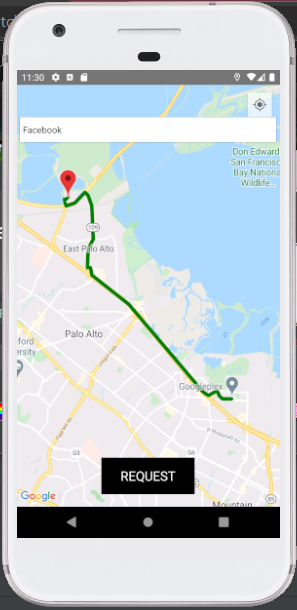


Hình trên đã cho thấy rõ cấu trúc cơ bản của 1 project React Native với các mục chính như:

* Android: nơi đây sẽ chứa các code dành cho môi trường Android và ta có thể vào đây để config 1 số thư viện nhất định như là Google Map,…
* IOS: Tương tự như với Android nhưng nếu muốn config, ta cần phải có 1 IDE riêng là Xcode và nếu muốn chạy được IDE đó, ta cần phải có 1 số máy chạy được MacOS như là MacBook, iMac, Mac mini hoặc là ta có thể Hackintosh.
* Node\_modules: chứa toàn bộ các thư viện bên trong React Native và ta có thể cài đặt thêm một số thư viện ở ngoài thông qua các câu lệnh như npm install –save tên-thư-viện nếu bạn đã cài đặt NPM CLI hoặc là yarn add tên-thư-viện nếu bạn cài đặt Yarn CLI.
* Package.json: đây là 1 tập tin sẽ lưu lại các config mà ta đã cài. Thông thường thì khi push 1 file tập tin lên Github hoặc là Gitlab, ta không thể thêm file Node\_modules vì nó rất nặng và tốn khá nhiều thời gian để push lên thì chính tập tin này sẽ giải quyết được vấn đề đó. Người dùng chỉ việc clone projec về như bình thường, sau đó là gõ npm install vào terminal thì lập tức các file đã được ghi vào trong package.json sẽ được tải về và sau đó ta có thể dùng ứng dụng như bình thường.
* **Index.js**

Đây là file mặc định sẽ được gọi đầu tiên và thường sẽ dùng file này để gọi AppRegistry tức là 1 component sẽ đăng ký 1 Component UI mặc định và đặt tên cho App

#### Components là gì ?

Có thể nói đây là 1 định nghĩa cơ bản nhất đối với ReactJS và React Native nói riêng, cũng như là các framework, library khác như Angular, Vue nói chung. Điều căn bản nhất của component là chia nhỏ UI thành nhiều phần, ví dụ trong hình ảnh sau:

Ta có thể thấy tất cả những gì hiển thị ở trên đều là Component, từ ô Input, nút Request cho tới Bản đồ và đường vẽ hiển thị lộ trình đi, và nếu muốn hiển thị toàn bộ các component trên thì sẽ phải gọi 1 component cha để bao bọc toàn bộ các Component trên.

Đây là 1 lợi thế lớn trong React Native vì ta có thể tái sử dụng và tăng khả năng mở rộng nó. Vì thế, trong cộng đồng React Native, có rất nhiều người đã tinh chỉnh lại các component gốc do Facebook tạo ra ví dụ như <TextInput> cho đẹp hơn và publish ra cho cộng đồng. Có thể kể đến 1 số ví dụ như thư viện React Native Material UI, Native Base, React Native Elements, … hoặc nhà phát triển có thể tự mình sáng tạo các UI riêng cho mình.

#### State và Props ?

Ở trong React Native, ta có thể dùng State và Props để điều khiển component.

* **Props là gì ?**

Props có thể được hiểu như là 1 đối tượng và được truyền vào bên trong component, component sẽ nhận được giá trị props từ component cha và không được quyền sửa lại ở component con, sau đó thì sẽ xuất ra dưới dạng react element. Cách viết truyền giá trị props vào từ component cha giống như là truyền atrribute vào trong 1 thẻ ở HTML. Ví dụ:

export class ChaoHoi extends Component {

    render(){

        return(

            <Text>{this.props.chaoHoiText}</Text>

        )

    }

}

ChaoHoi.proptypes = {

    chaoHoiText: PropTypes.string.isRequired

}

export class MainComponent extends Component{

    render(){

        return(

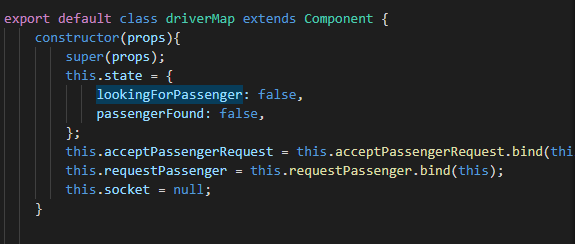
            <ChaoHoi chaoHoiText="Em đẹp lắm" />

        )

    }

}

* **State là gì ?**
* State thì giống như 1 kho dữ liệu dành cho component. Nó được sử dụng chủ yếu để cập nhật component khi user thực hiện một số thao tác như là giữ, ấn nút, gõ phím, …
* Khi khởi tạo 1 component, ta thường sẽ extends 1 lớp cơ sở bên trong thư viện React là component và bên trong lớp đó, sẽ có thuộc tính state và hàm tạo mặc định sẽ gán cho nó có giá trị là Null. Vì thế, khi mới tạo 1 component, ta có thể ghi đè hàm tạo và thay đổi giá trị bên trong state.

Ví dụ:

* Trong trường hợp nếu ta muốn cập nhật 1 state nào đó, chúng ta không thể gán trực tiếp giá trị của nó như this.state={lookingForPassenger: True}, mà ta sẽ xài 1 phương thức đó là setState. Với ví dụ trên, ta có thể thay đổi giá trị của nó như sau: this.setState({lookingForPassenger: True}).
* Bất cứ khi nào ta thay đổi 1 state của 1 component nào đó thì React sẽ tự động render lại component đó cho chúng ta.

#### Stylesheet là gì ?

Trước hết, hãy nhắc 1 chút về style ở trong các UI Component. Đây là 1 props của các UI Component cho phép chúng ta tùy biến lại hình dáng, kích thước của 1 component đó. Ví dụ như: <View style={width 23, height: 33}/>.

Tuy nhiên, hãy đặt trường hợp như ta cần tạo ra 2 hoặc nhiều component giống nhau về kích thước, thì việc viết lại nhiều lần sẽ tốn nhiều thời gian và code sẽ bị lặp lại. Qua đó, ta có thể sử dụng 1 biến Stylesheet để tạo ra style giống nhau của các component đó.

Ví dụ:

import React,{Component} from "react";

import {

    View,StyleSheet,Text

    } from "react-native";

    export default class testThu extends Component{

        render(){

            return(

                <View>

                    <View style={styles.forView}>

                        <Text>Hello world</Text>

                    </View>

                    <View style={styles.forView}></View>

                </View>

            )

        }

    }

    const styles = StyleSheet.create({

        forView:{

            width:30,

            height:30,

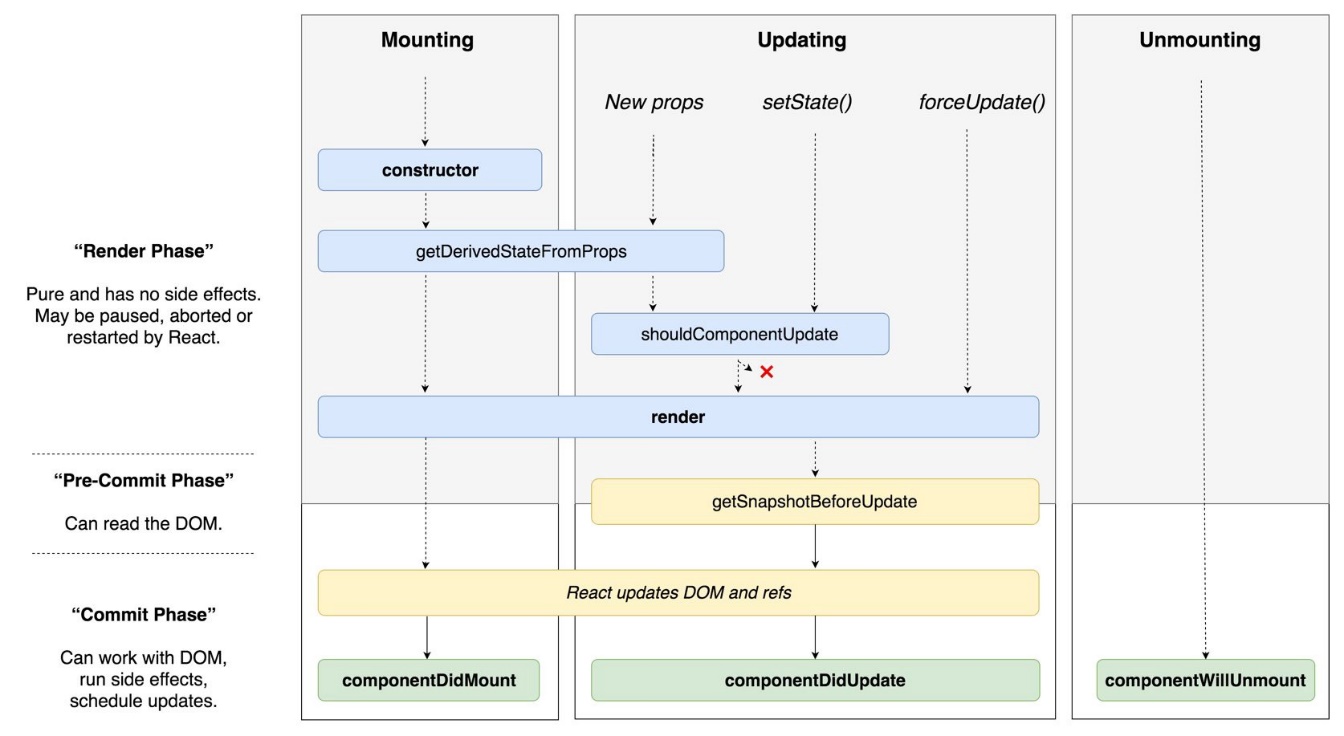
        }

    });

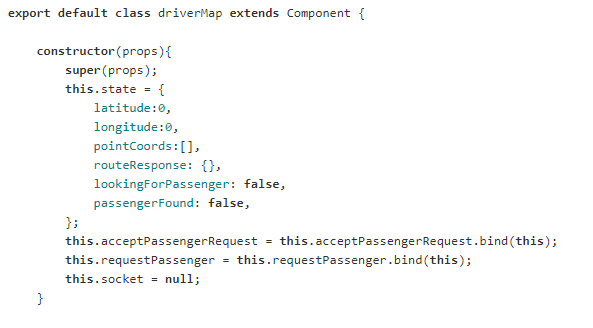
#### Vòng đời của 1 component trong React Native

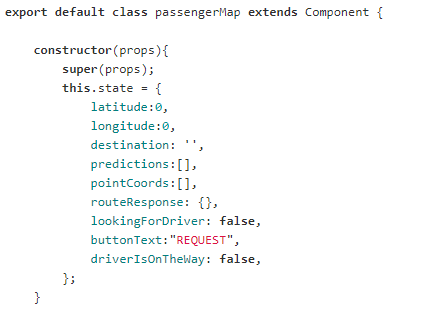
Mỗi component trong ReactJS hoặc là React Native, chúng đều phải trải qua 1 vòng đời từ lúc khởi tạo component cho tới khi hủy component đó. Vòng đời của từng component sẽ được chia làm 3 phần chính:

* **Mount (Gắn):**
* Giai đoạn này sẽ diễn tả quá trình khởi tạo các giá trị của component như là props, state và sẽ render lên các component đó lên trên màn hình. Ở trong giai đoạn này, React Native cũng sẽ cung cấp cho ta 1 số các hàm như:
* **Constructor**: Hàm này sẽ được dùng để khởi tạo các state, kế thừa các props của components cha hoặc là bind các hàm của lớp.
* **componentWillMount**(): method này sẽ được gọi sau khi thực hiện ở hàm Contructor. Tuy nhiên, method này hiện đang dần bị loại bỏ và được khuyên là không nên dùng nữa.
* **render**(): Hàm này sẽ thực hiện việc vẽ lên các component lên thông qua React element (Được viết bằng JSX). Tại đây thì ta có thể đọc các element và thực hiện 1 số thao tác. Tuy nhiên, chúng ta không nên thay đổi các props và states ở đây mà thực hiện ở 1 hàm khác sau khi render.
* **componentDidMount()**: method này được sử dụng khi component được render ở bên máy client. Đây cũng có thể là nơi để thực hiện 1 số hàm như update state, gọi API, …
* **Updating** (Cập nhật):
* Sau khi được render lần đầu, các method của updating sẽ được gọi với các lần render sau. Có 5 method trong Updating:
* **componentWillReceiveProps()**: hàm này sẽ được gọi khi 1 component sau khi qua giai đoạn mounting nhận được các props mới. Tuy nhiên, hàm này cũng đang dần bị loại bỏ trong tương lai sắp tới,
* **shouldComponentUpdate()**: đây là 1 hàm đặc biệt sẽ quyết định được là các hàm còn lại trong giai đoạn Updating có được thực hiện hay không. Mặc định thì hàm này sẽ trả về true, còn nếu không thì các hàm sao sẽ không được thực thi.
* **componentWillUpdate(nextProps, nextState):** hàm nãy sẽ được gọi trước khi hàm render của giai đoạn Updating được gọi, tuy có các state và props mới nhưng ta không thể họi hàm setState ở đây, mục tiêu của hàm này là tương tác với những thứ bên ngoài React như là API,…
* **render():** hàm này sẽ tương tự với hàm render() ở Mounting
* **componentDidUpdate(prevProps, prevState):** sẽ tự động truyền vào 2 tham chiếu là prevProps và prevState, ta có thể dùng chúng để so sánh với các state hiện tại để tiến hành request API nếu có thay đổi.
* **Unmounting (Gỡ):**
* Đây là giai đoạn cuối cùng của 1 component khi nó chuẩn bị bị hủy như là xóa dữ liệu hoặc là tắt kết nối tới API,…. Và giai đoạn này chỉ có 1 hàm duy nhất là componentWillUnmount() để thực hiện các thao tác trên.

**Dưới đây là bảng tóm tắt các method trong life circle của 1 component:**

#### Higher Order Component là gì ?

Hãy nhìn vào 2 đoạn code ví dụ dưới đây:



Ta có thể dễ dàng nhận ra được rằng nó sẽ có 1 số điểm khá là giống nhau như là thuộc tính latitude, longitude, destination bên trong state của 2 ví dụ trên và việc code 2 cùng những đoạn code đó đến tận 2 lần có thể gây nên lãng phí tài nguyên cũng như là nhàm chán, vì thế Higher Order Component được sinh ra để giải quyết vấn đề trên. Vậy Higher Order Component là gì ?

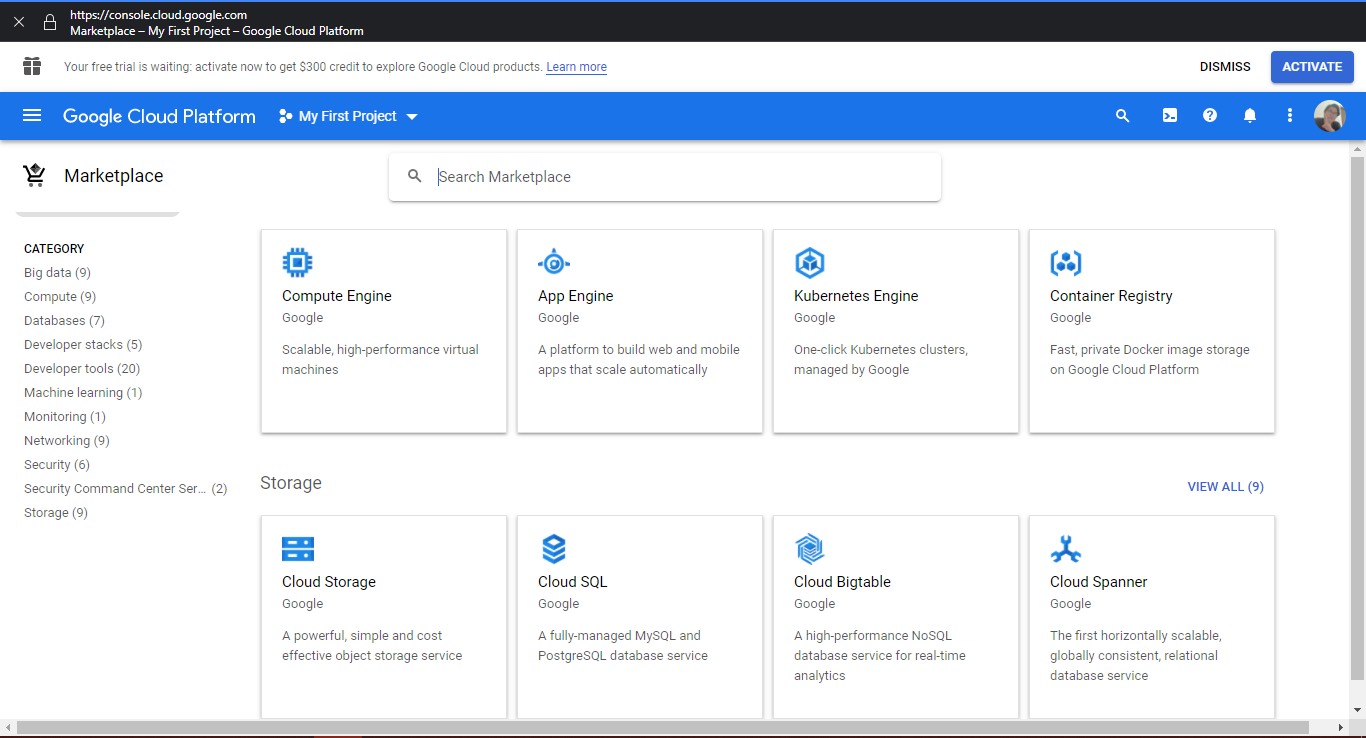
Higher Order Component (còn được gọi tắt là HOC) là 1 pattern cho phép ta có thể chia sẻ các tài nguyên chung với nhau như là function, state giữa các component. Điều này giúp cho các lập trình viên có thể tiết kiệm 1 khoảng thời gian để viết code, cũng như là tiết kiệm được tài nguyên cho máy khi không phải lặp lại các đoạn code cũ.

# GOOGLE MAP API

## Tổng quan về Google Map Api?

### Google Cloud là gì ?

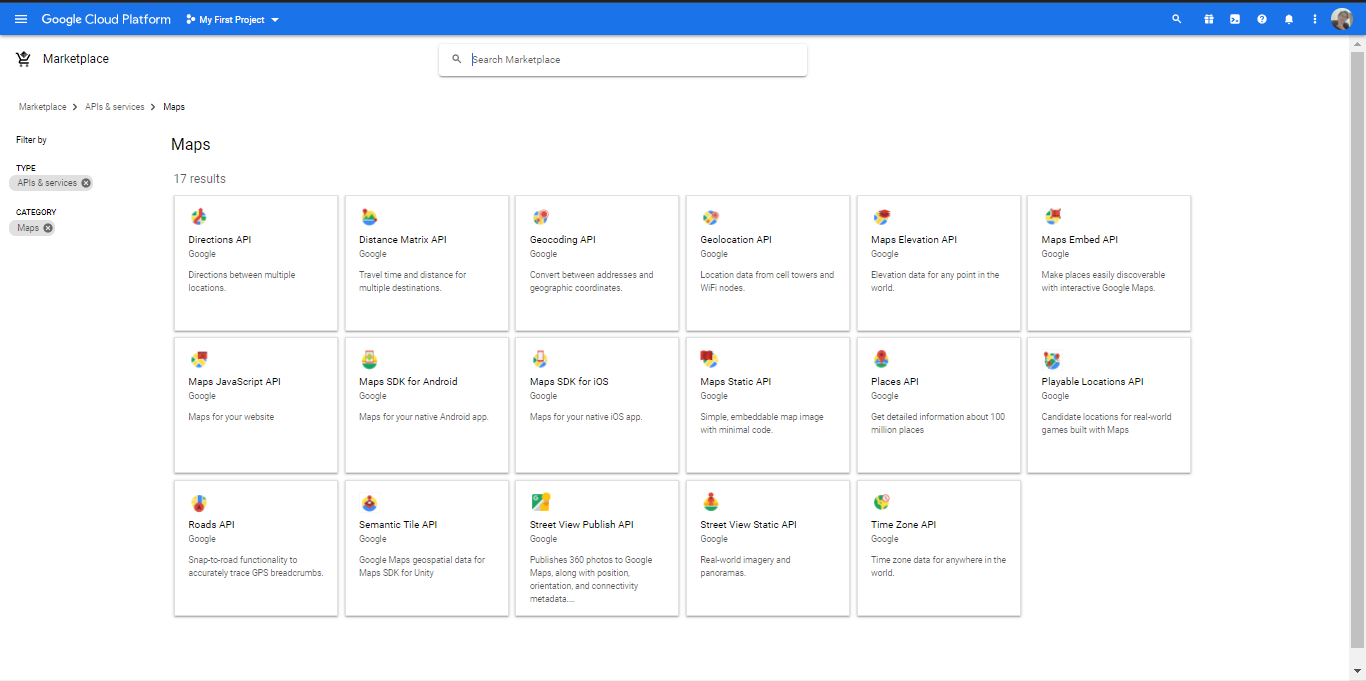
Đây là một bộ dịch vụ điện toán đám mây chạy cũng cơ sở hạ tầng với các sản phẩm người dùng cuối của Google như Youtube, Google Drive, Google Search,…. Google Cloud cung cấp rất nhiều dịch ụ đám mây như là Điện toán, Cơ sở dữ liệu, Máy ảo,… Và người dùng sẽ phải đăng ký bằng thẻ nếu muốn sử dụng các dịch vụ trên.



### Google Map API là gì ?

Đây là một trong những dịch vụ ở trong Google Cloud Platform. Dịch vụ này sẽ cung cấp cho người dùng cuối bản đồ trên thế giới, cũng như là một số các dịch vụ bên lề khác như là múi giờ, khoảng cách đường đi, điểm đến, …

Ở trong đồ án này, em sẽ sử dụng 3 dịch vụ chính bên trong Google Map API:

* **Google Direction API**: cung cấp cho người dùng khoảng cách của 2 điểm
* **Google Location**: Xác định vị trí mà người dùng muốn tới
* **Google Map for Android**: Kích hoạt View của Google map cho điện thoại android

Ở 3 dịch vụ được sử dụng ở trên, duy chỉ có **Google Map for Android** là hoàn toàn miễn phí, 2 dịch vụ còn lại sẽ hoàn toàn tính phí dựa theo số lần truy cập vào. Dưới đây là bảng giá cho 2 dịch vụ tính phí trên:

**Bảng giá cho dịch vụ Google Direction mỗi tháng**

(Giá cho mỗi lần Query)

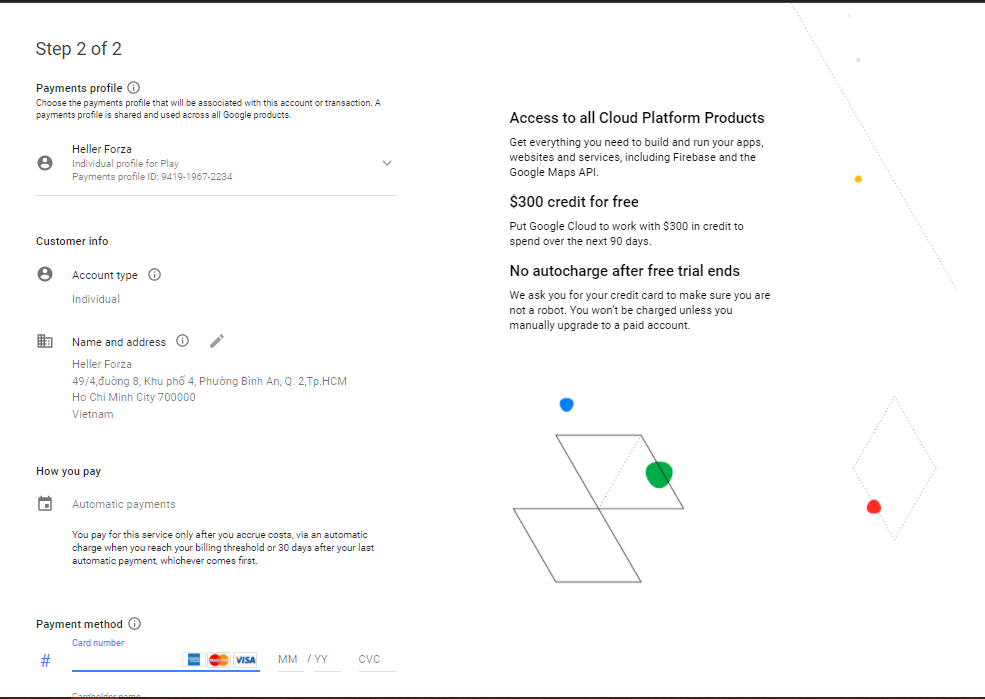
|  |  |  |
| --- | --- | --- |
| **Từ 0 đến 100,000 lần** | **Từ 100,001 cho đến 500,000 lần** | **Trên 500,000 lần** |
| 0.01 Đô cho 1 lần (10.00 Đô cho 1000 lần) | 0.008 Đô cho 1 lần (8.00 Đô cho 1000 lần) | Liên hệ để báo giá |

**Bảng giá cho dịch vụ Google Location mỗi tháng**

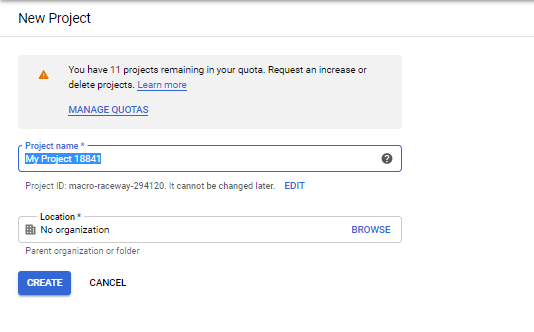
(Giá cho mỗi lần Query)

|  |  |  |
| --- | --- | --- |
| **Từ 0 đến 100,000 lần** | **Từ 100,001 cho đến 500,000 lần** | **Trên 500,000 lần** |
| **0.017 Đô cho 1 lần (17.00 Đô cho 1000 lần)** | **0.0136 Đô cho 1 lần (13.60 Đô cho 1000 lần)** | **Liên hệ để báo giá** |

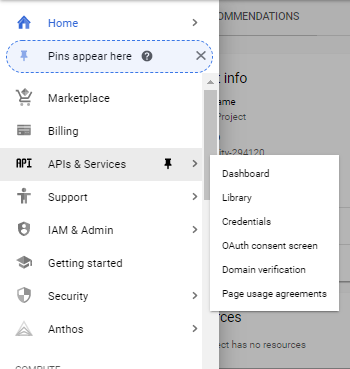
## Cách đăng ký dịch vụ Google Map

Để đăng ký các dịch vụ trên, trước tiên người dùng phải cung cấp thẻ Visa hoặc là thẻ MasterCard cho Google ở dịch vụ Google Billing. Google có cung cấp 1 gói 350 Đô với thời gian sử dụng là 12 tháng cho người dùng xài thử và sẽ không tự động trừ tiền sau khi hết thời gian sử dụng miễn phí trên.

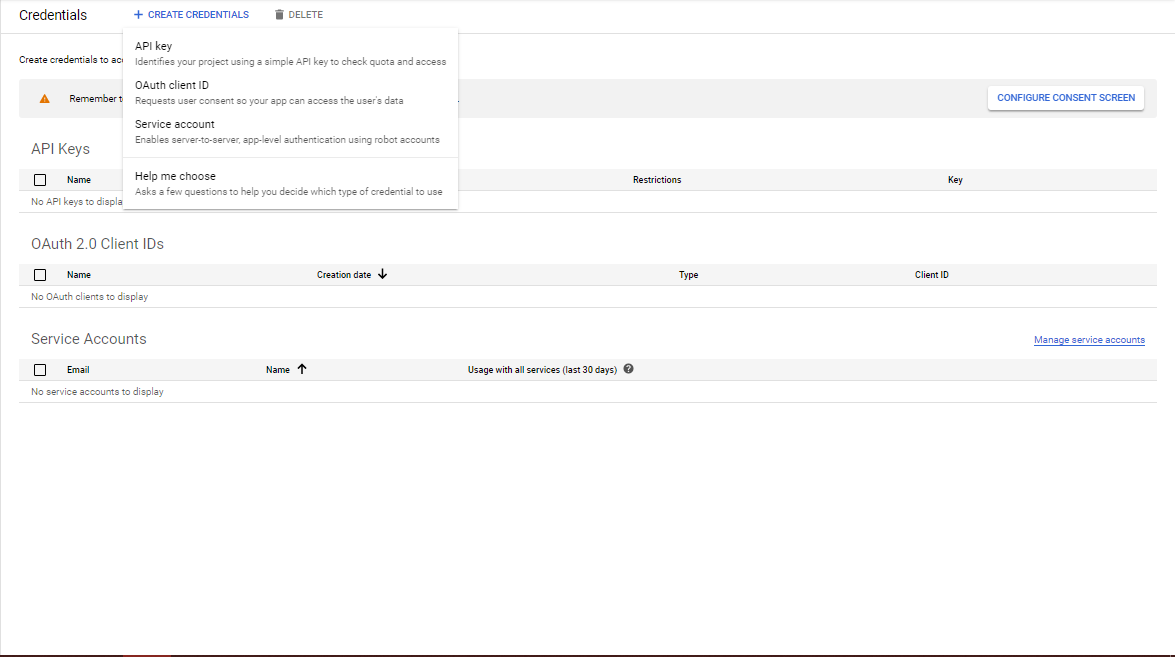
Sau khi kích hoạt tài khoản sử dụng miễn phí thành công, ta tiến hành khởi tạo 1 project mình mong muốn. Tiến hành đặt tên cho project và bấm create.

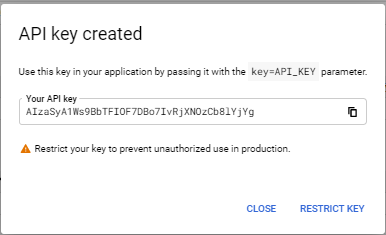


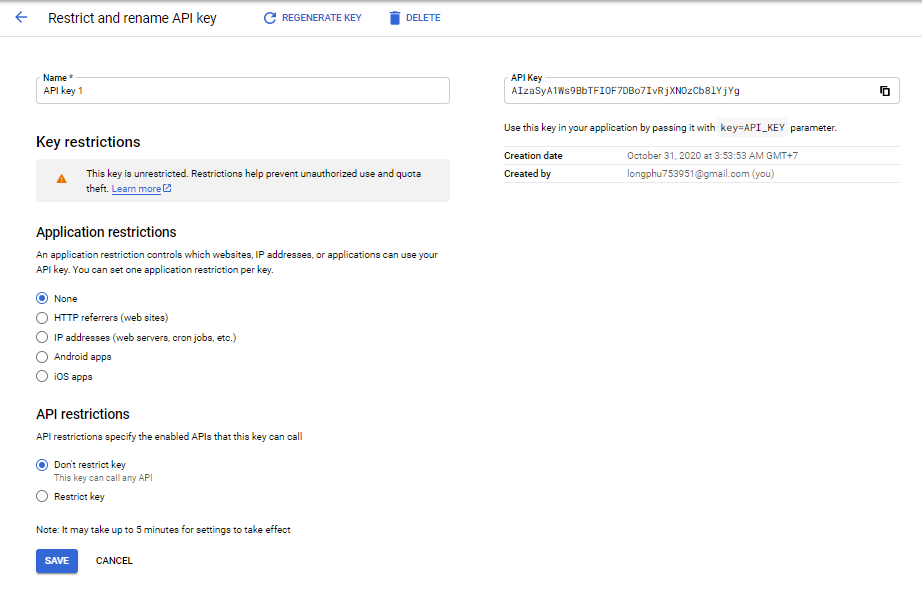
Khi project đã được tạo thành công, ta chọn mục APIs and Services và sau đó chọn mục Credentials.

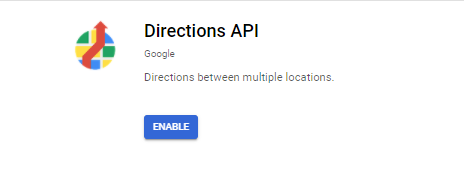


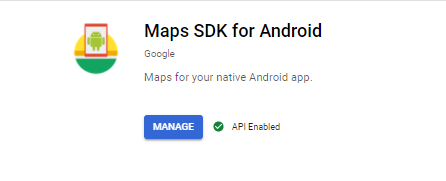
Bấm vào mục Create Credentials và chọn API Key

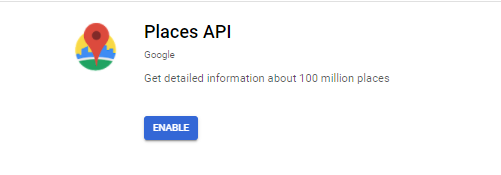


Sau đó, hệ thống của Google sẽ tạo ra 1 key và key này hoàn toàn là bí mật, không được để lộ ra cho người khác sử dụng. Ta có thể chọn Restrict key để giới hạn ứng dụng nào có thể sử dụng dịch vụ từ API key này.



Khi đã tạo Key thành công, ta vào mục Marketplace và chọn API mà mình muốn.





# Xây dụng Back-end với ExpressJS

## Tổng quan về NodeJS

### NodeJS là gì ?

Đây là 1 platform được xây dựng trên V8 Javasciprt Runtime, môi trường mã nguồn mở mạnh mẽ được phát triển cho các trình duyệt web chạy với nhân là Chrominum như là Google Chrome, Edge Chrominum. NodeJS được thiết kế để xây dựng các ứng dụng mạng một cách nhanh chóng và có khả năng mở rộng cao.

Dưới đây là một ví dụ “Hello World” được xây dựng trên NodeJS:

const http = require('http');

const hostname = '127.0.0.1';

const port = 4040;

const server = http.createServer((req, res) => {

  res.statusCode = 200;

  res.setHeader('Content-Type', 'text/plain');

  res.end('Hello World');

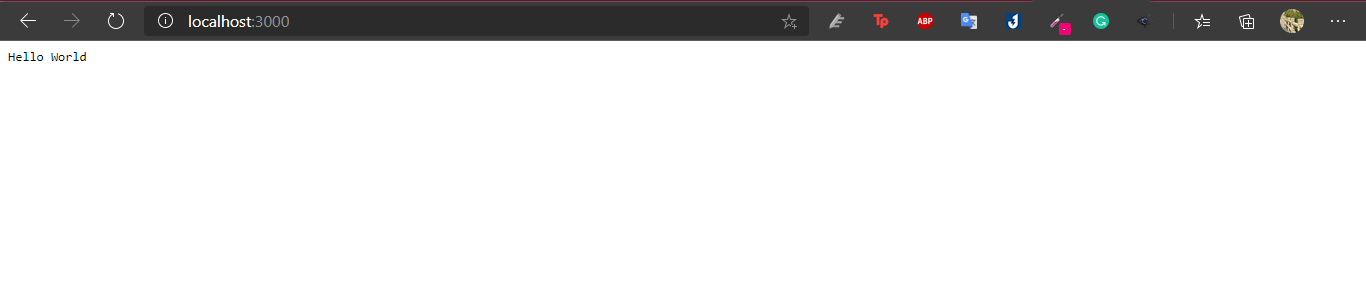
});

server.listen(port, hostname, () => {

  console.log(`Server running at http://${hostname}:${port}/`);

});

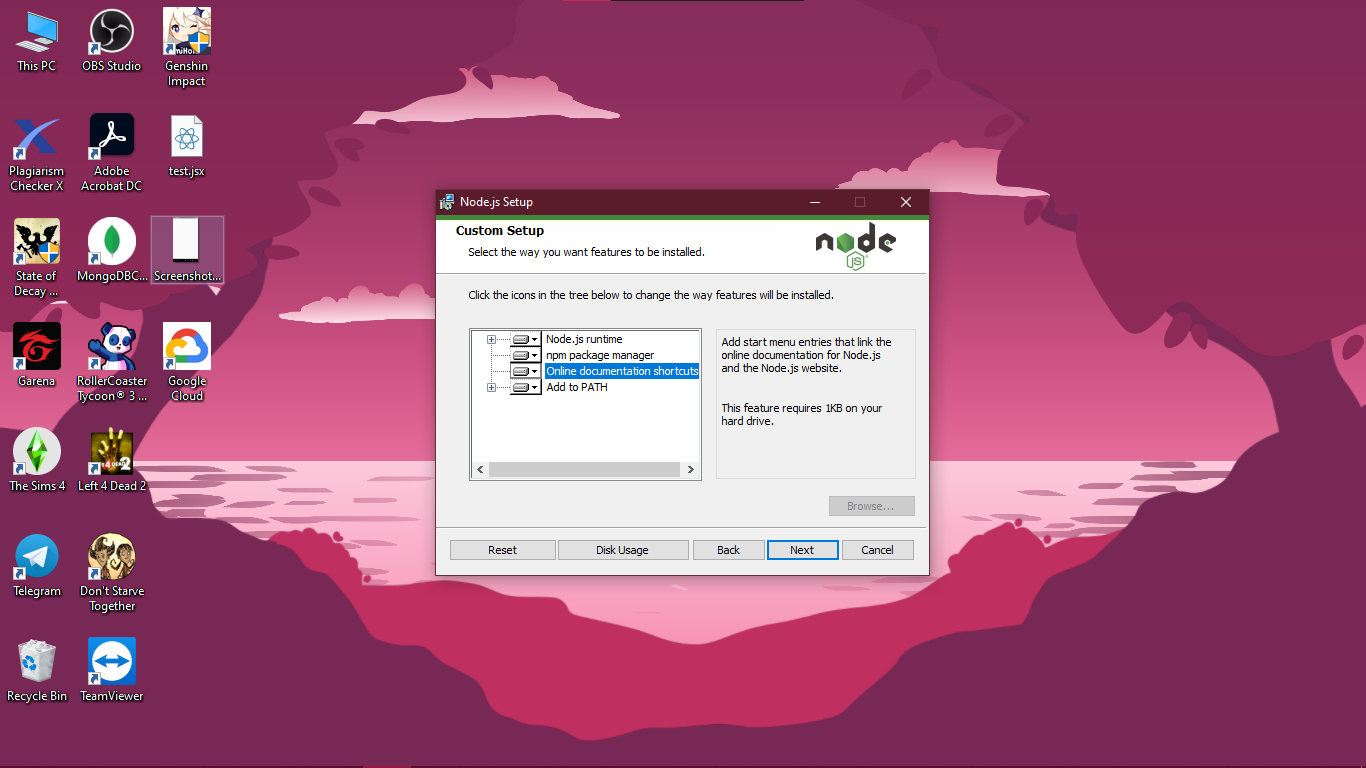
Và dưới đây là kết quả:



Qua ứng dụng Hello World trên, ta có thể thấy được sự tiện lợi và tốc độ khá là nhanh chóng của NodeJS. Tuy nhiên, nếu muốn có được một ứng dụng có tính ổn định cao và xử lý các logic phức tạp thì NodeJS sẽ không phải là ý tưởng hay. Các ứng dụng yêu cầu tốc độ cao như các ứng dụng Websocket (Chat server, Game Online), ứng dụng REST API, Cloud Services,…

### Hướng dẫn cài đặt NodeJS

Việc cài đặt NodeJS hết sức là đơn giản, việc duy nhất ta cần làm là lên trang chủ của NodeJS: <https://nodejs.org/en/download>, chọn phiên bản phù hợp đối với hệ điều hành.



Ở đây NodeJS sẽ hỗ trợ ta trong việc cài đặt CLI npm để có thể tiến hành cài đặt các thư viện từ bên thứ ba vào ứng dụng mà ta mong muốn. Sau đó là bấm Finish là NodeJS sẽ được tiến hành cài đặt vào trong máy. Để đảm bảo là NodeJS đã được cài đặt thành công chưa thì người dùng có thể vào Command Line và gõ lệnh node -v, nếu được trả về số phiên bản của NodeJS vừa cài là thành công.

## Tổng quan về ExpressJS

### Giới thiệu về ExpressJS

Giống như với các framework nổi tiếng khác như là ASP.Net, Laravel, Spring boot,… ExpressJS là một framework dùng để phát triển các ứng dụng web hoặc là Web API dựa trên nền tảng của NodeJS. Tuy đây là 1 framework nhỏ nhưng hết sức linh hoạt và không làm giảm tốc độ của NodeJS mà ngược lại còn cung cấp các tính năng, giúp cho việc lập trình trở nên tốt hơn.

### ExpressJS. Giới thiệu về ExpressJS | by Vmnguyen | MediumCấu trúc của ExpressJS

**Root:** sẽ là nơi lưu trữ các thông tin của project như là package.json, .gitignore và 1 file index.js dùng để khởi tạo project lên.

**Routes:** Là một bộ định tuyến giúp định ra các url và hành động (controller) kèm theo với url đó.

**Controller**: Nhận các request từ phía server là trả lại response. Controller sẽ tương tác với Models và View(Đối với MVC).

**Model:** Chứa các schema tương ứng với cơ sở dữ liệu ở bên phía server.

## Tổng quan về MongoDB:

### MongoDB là gì ?

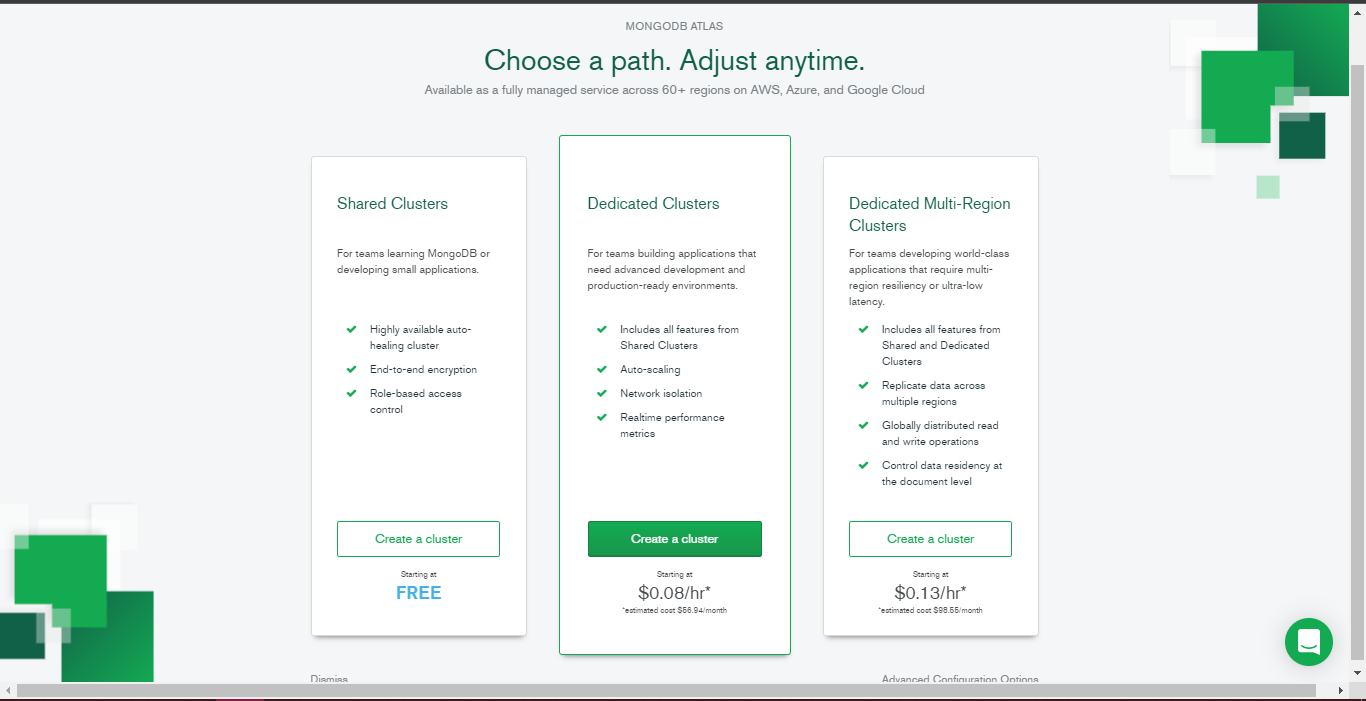
Đây là 1 cơ sở dữ liệu thường sẽ được dùng kèm với NodeJS và Framework ExpressJS trên. Khác với các cơ sở dữ liệu thường như SQL Server, MySQL, đây là database hướng tài liệu, tức là loại database này sẽ không cấu trúc bảng như các loại database mà sử dụng dưới dạng tài liệu như JSON. Điều này sẽ cho MongoDB tốc độ lưu trữ dữ liệu và query cực nhanh và hết sức linh hoạt.

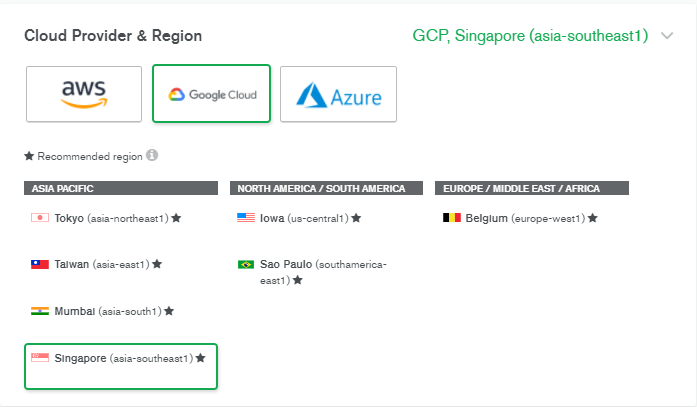
### Cách tạo cơ sở dữ liệu bằng MongoDB

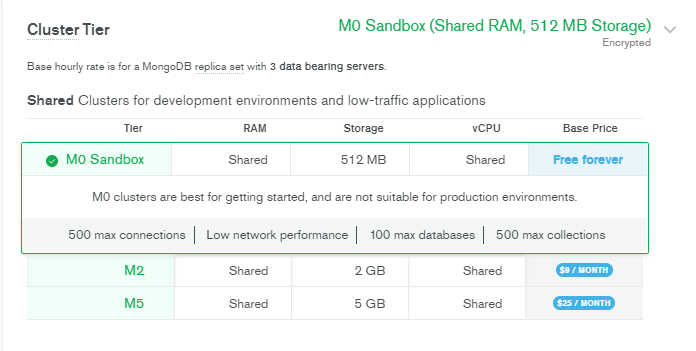
#### Tạo dữ liệu trên Atlas

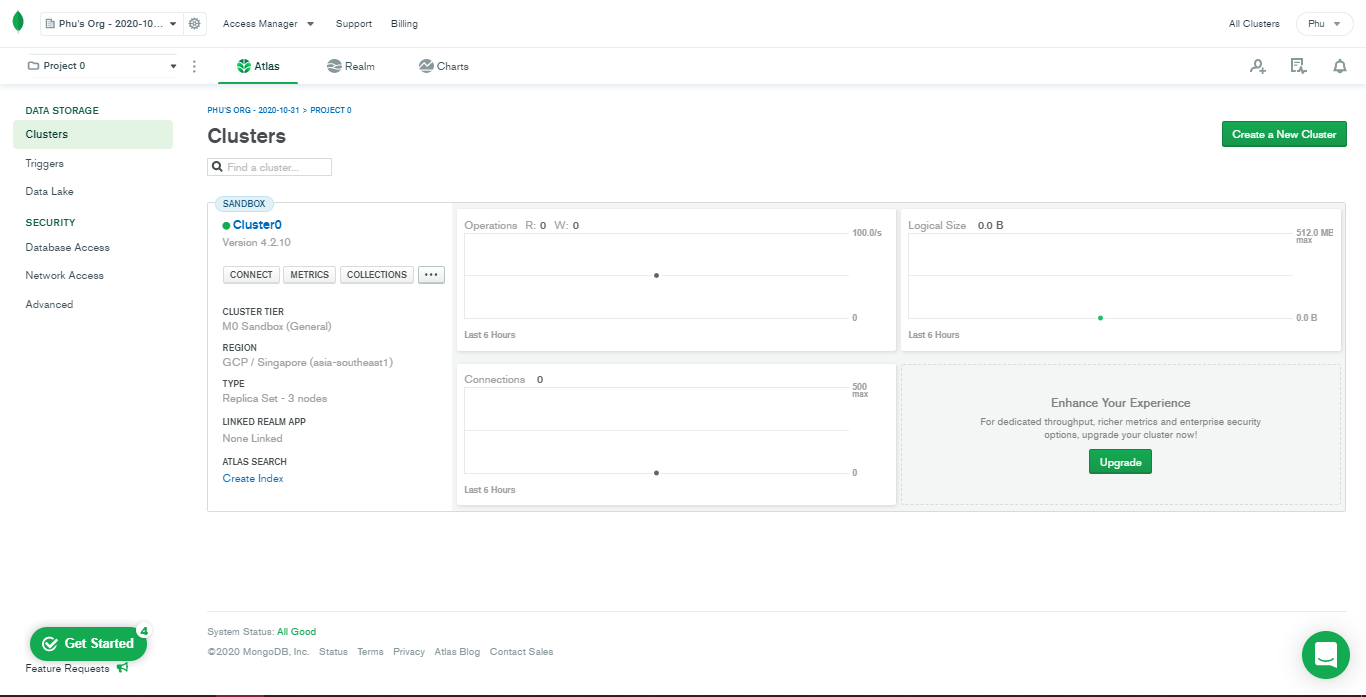
Đây là một dữ liệu đám mây chạy trên các AWS của Amazon, Azure của Microsoft và Google Cloud Platform của Google.

Việc tạo cơ sở dữ liệu trên đây khá là đơn giản. Việc trước tiên ta cần làm là vào trang web: <https://account.mongodb.com/account/login> để tạo tài khoản và sau đó đăng nhập.

Khi tạo tài khoản thành công, họ sẽ đưa ra ba gói cho người dùng, tùy vào mục đích sử dụng để đăng ký gói.

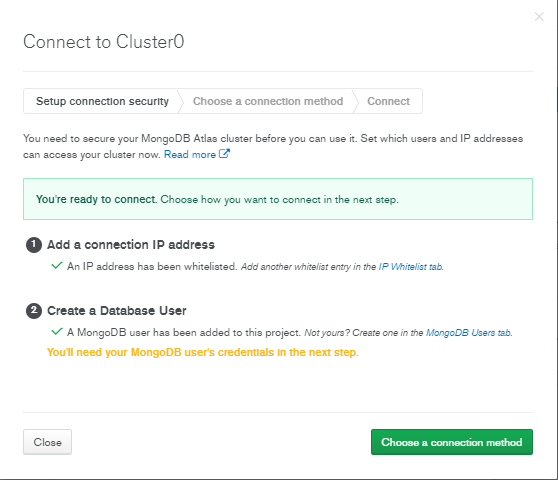
Sau khi chọn gói thành công, MongoDB sẽ yêu cầu người dùng lựa 1 trong 3 nền tảng: Azure, AWS hoặc là Google Cloud Platform. Mỗi nền tảng thì sẽ đặt server ở tùy vị trí do công ty quản lý 3 nền tảng kia muốn. Lời khuyên là chọn những nơi ở gần vị trí địa lý của người dùng để không bị ảnh hưởng tới tốc độ kết nối.

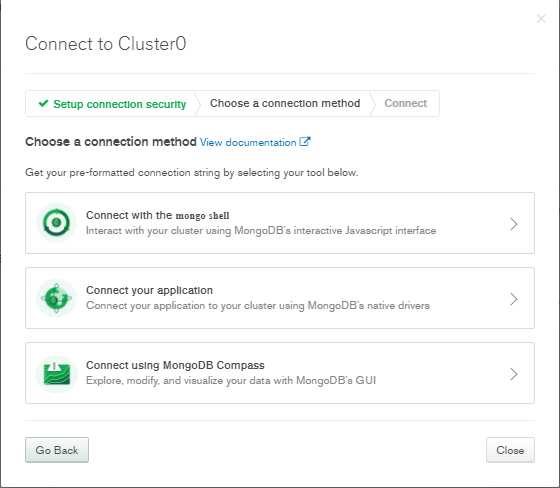
Sau khi chọn nền tảng thành công thì sẽ tiếp tục chọn Cluster. Sẽ có 3 gói như trong hình dưới đây, tùy vào mục đích của người dùng thì sẽ lựa gói thích hợp.

Sau khi chọn gói cluster xong, ta có thể chọn giữa việc có Backup dữ liệu hay không và sửa tên riêng cho Cluster của mình. Sau đó bấm vào nút Create Cluster. Thời gian tạo thường sẽ tốn từ 1-3 phút.

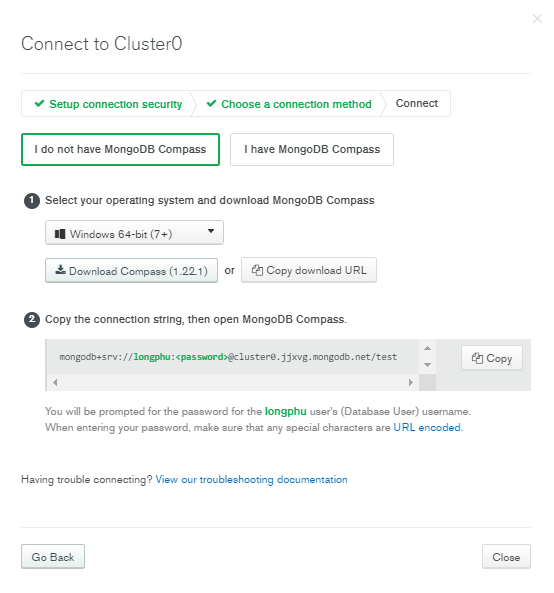
Bấm vào nút Collection và chọn nút Add My Own Data để tạo ra Database mới.

Sau khi Database được tạo, ta quay trở về trang chủ và chọn Connection. Ở đây ta sẽ tiến hành chọn IP được quyền truy cập tới Database này hoặc là có thể đăng nhập ở mọi nơi, tạo ra tài khoản để đăng nhập database trên.



Tới bước 2, ta sẽ được quyền chọn 1 trong 3 cách để kết nối tới Database trên. Nếu đã cài đặt MongoDB ở máy local thì ta có thể chọn mục Compass vì MongoDB cũng cung cấp cho người dùng một ứng dụng để sử dụng nội bộ bên trong máy. 

Tới bước cuối cùng, Atlas sẽ cung cấp 1 url để có thể truy cập vào cluster vừa tạo, việc duy nhất cần làm là nhập mật khẩu của tài khoản vừa tạo và chọn tên của database đã tạo ở Collection.



Tuy nhiên, với việc sửu dụng NodeJS thì ta phải thêm “?retryWrites=true&w=majority” vào url trên thì mới có thể truy cập vào được MongoDB trên. Ví dụ như:

“mongodb+srv://longphu:DayLaNoiNhapMatKhau@cluster0.ydr5k.mongodb.net/Ten\_Database\_Vua\_Tao?retryWrites=true&w=majority”

#### Kết nối với MongoDB qua NodeJS

Để kết nối với cơ sở dữ liệu trên, ta cần phải tải một thư viện tên là Mongoose để hỗ trợ trong việc giao tiếp giữa MongoDB và NodeJS. Việc ta cần làm là mở Command Line, đi tới project và nhập lệnh npm install mongoose.

Sau khi tải mongoose thành công, ta tiến hành vào trang index.js ngoài cùng