**Báo cáo**

*Số 17*

**Họ Tên**: Nguyễn Thái Sơn

**Ngày**: 11-12-2019

Task name

* Nội dung tìm hiểu : Tìm hiểu chung về nodejs, ưu nhược điểm của nodejs
* Tiến độ (%) 100%
* Vấn đề gặp phải

**NodeJs là gì**

* Nodejs là một mã nguồn mở, một môi trường cho các máy chủ và ứng dụng mạng
* Nodejs sử dụng Google V8 Javascript engine để thực thi mã, và một tỷ lệ lớn các module cơ bản được viết bằng Javascript. Các ứng dụng nodejs được viết bằng Javascript
* Nodejs chứa một thư viện built-in cho phép các ứng dụng hoạt động như một webserver mà không cần phần mềm như Nginx, Apache HTTP Server hoặc IIS
* Nodejs cung cấp kiến trúc hướng sự kiện (event-driven) và non-blocking I/O API,tối ưu hóa thông lượng của ứng dụng và có khả năng tự mở rộng cao
* Mọi hàm trong nodejs là không đồng bộ, do đó các tác vụ đều được xử lý và thực thi ở chế độ nền

**Ứng dụng của NodeJs**

* Xây dựng websocket server (chat server)
* Hệ thống Notification (giống như Facebook hay Twitter)
* Ứng dụng Upload file trên client
* Các máy chủ quảng cáo
* Các ứng dụng dữ liệu thời gian thực khác

**Nhược điểm của NodeJS**

* Ứng dụng nặng tốn tài nguyên. Nếu bạn cần xử lý các ứng dụng tốn tài nguyên CPU như encoding video, convert file, decoding encrytion hoặc các ứng dụng tương tự như vậy thì không nên dụng NodeJS (vì Nodejs được viết bằng c++ và Javascript nên phải thông qua một trình biên dịch nữa nên sẽ lâu hơn)
* Nodejs và các ngôn ngữ khác như PHP, Ruby, Python, .Net thì việc cuối cùng cũng chỉ là phát triển các app web. Node mới sơ khai như các ngôn ngữ lập trình khác. Vậy nên đừng hi vọng Nodejs sẽ hơn PHP, Ruby, Python ở thời điểm này

**Ưu điểm của NodeJS**

* Đặc điểm nổi bật của Nodejs là nó nhận và xử lý nhiều kết nối chỉ với một single-thread. Điều này giúp hệ thống tốn ít RAM nhất và chạy nhanh nhất khi không phải tạo thread mới cho mỗi truy vấn giống PHP. Ngoài ra tận dụng ưu điểm non-blocking I/O của Javascript mà nodejs tận dụng tối đa tài nguyên của server mà không tạo ra độ trễ như PHP
* Json APIs với cơ chế event-driven, non-blocking I/O (input/output) và mô hình kết hợp với javascript là sự lựa chọn tuyệt vời cho các dịch vụ webs làm bằng Json
* Ứng dụng trên một trang (single page application) nếu bạn định viết 1 ứng dụng thể hiện trên 1 trang Nodejs rất phù hợp để làm. Với khả năng xử lý rất nhiều request/s đồng thời thời gian phản hồi nhanh. Các ứng dụng bạn định viết không muốn nó tải lại trang, gồm rất nhiều request từ người dùng cần sự hoạt động nhanh để thể hiện sự chuyên nghiệp thì nodejs sẽ là sự lựa chọn của bạn
* Shelling tools unix Nodejs sẽ tận dụng tối đa Unix để hoạt động. Nodejs có thể xử lý hàng nghìn quá trình và trả về 1 luồng khiến cho hiệu xuất hoạt động đạt mức tối đa nhất và tuyệt vời nhất
* Streamming Data (Luồng dữ liệu) các web thông thường gửi HTTP request và sẽ nhận phản hồi lại (luồng dữ liệu). Giả xử sẽ cần xử lý 1 luồng dữ liệu cực lớn, Nodejs sẽ xây dựng các Proxy phân vùng các luồng dữ liệu để đảm bảo tối đa hoạt động cho các luồng dữ liệu khác
* Ứng dụng web thời gian thực với sự ra đời của các ứng dụng di động & HTML5 nên Nodejs rất hiệu quả khi xây dựng những ứng dụng chat thời gian thực (real-time applications) như ứng dụng chat, các dịch vụ mạng xã hội Facebook

**Với những ưu – nhược điểm trên, khi nào chúng ta sẽ dụng nodejs**

* Không nên dùng nodejs khi:
  + Xây dựng các ứng dụng hao tốn tài nguyên: Không nên sử dụng Nodejs khi bạn dang muốn viết một chương trình convert video. Nodejs hay bị rơi vào tình trạng treo khi làm việc với file có dung lượng lớn
  + Một ứng dụng chỉ toàn CRUD: Nodejs không nhanh hơn PHP khi làm các task nặng tính I/O như vậy. Ngoài ra với sự ổn định lâu dài của các webserver script khác, các tác vụ CRUD của nó đã được tối ưu hóa. Còn Nodejs sẽ cho ra các API khá ngớ ngẩn
  + Khi bạn cần sự ổn định trong ứng dụng
  + Quan trọng hơn hết là bạn chưa hiểu hết về nodejs
* Nên dùng NodeJS khi nào
  + Nodejs thực sự tỏa sáng trong việc xây dựng RESTful API (json). Gần như không có ngôn ngữ nào xử lý Json dễ dàng hơn Javascript, chưa kể các API server thường không thực hiện những xử lý nặng nề nhưng lượng concurrent request thì rất cao. Mà Nodejs thì xử lý non-blocking. Chẳng còn gì thích hợp hơn Nodejs trong trường hợp này
  + Những ứng dụng đòi hỏi các giao thức kết nối khác chứ không phải chỉ có HTTP. Với việc hỗ trợ giao thức tcp, từ đó có thể xây dựng một giao thức custom nào đó một cách dễ dàng
  + Những ứng dụng thời gian thực : khỏi phải nói vì Nodejs dường như sinh ra để làm việc này
  + Những website stateful. Nodejs xử lý mọi request trên cùng một process giúp cho việc xây dựng các bộ nhớ đệm chưa bao giờ đơn giản thế: Hãy lưu nó vào một biến global, và thế là mọi request đều có thể truy cập đến bộ nhớ đệm đó. Caching sẽ không còn quá đau đầu như trước đây, và bạn có thể lưu cũng như chia sẻ trạng thái của một client với các client khác ngay trong ngôn ngữ, chứ bạn không cần thông qua bộ nhớ ngoài

**Tại sao lại là NodeJs**

* NodeJs sử dụng lập trình không đồng bộ.
* Một nhiệm vụ chung cho một máy chủ web có thể là mở một tệp trên máy chủ và trả lại nội dung cho máy khách
* Đây là cách PHP hoặc ASP sử lý một yêu cầu tệp:
  + Gửi tác vụ đến hệ thống tệp của máy tính
  + Chờ trong khi hệ thống tệp mở và tệp đóng
  + Trả về nội dung cho khách
  + Sẵn sàng để xử lý các yêu cầu tiếp theo
* Đây là cách NodeJs xử lý một yêu cầu tệp
  + Gửi tác vụ đến hệ thống tệp của máy tính
  + Sẵn sàng để xử lý các yêu cầu tiếp theo
  + Khi hệ thống tệp đã mở và đọc tệp, mày chủ sẽ trả lại nội dung cho máy khách
* NodeJs loại bỏ sự chờ đợi và chỉ cần tiếp tục với yêu cầu nào tiếp theo
* NodeJs chạy lập trình đơn luồng, không chặn lập trình không đồng bộ, rất hiệu quả về bộ nhớ

**NodeJs có thể làm gì?**

* NodeJs có thể tạo ra nội dung một trang động
* NodeJs có thể tạo, mở, đọc, viết, xóa và đóng tệp trên máy chủ
* NodeJs có thể thu thập dữ liệu biểu mẫu
* NodeJs có thể thêm, xóa, sửa, đổi dữ liệu trong cơ sở dữ liệu của bạn

**Tệp NodeJs là gì?**

* Các tệp NodeJs chứa các tác vụ sẽ được thực thi trong các sự kiện nhất định
* Một sự kiện điển hình là ai đó đang cố truy cập vào một cổng trên máy chủ
* Các tệp NodeJs phải được bắt đầu trên máy chủ trước khi có bất kỳ ảnh hưởng nào
* Các tệp NodeJs có phần mở rộng là .js

**Download NodeJs**

* Trang web chính thức của NodeJs có hướng dẫn cài đặt [https://nodejs.org](https://nodejs.org/)

**Bắt đầu**

* Khi đã tải xuống và cài đặt NodeJs trên máy tính cá nhân, hãy thử hiển thị “hello world” trên trình duyệt web. Tạo tệp myfirst.js với nội dung như sau:

var http = require('http');  
  
http.createServer(function (req, res) {  
  res.writeHead(200, {'Content-Type': 'text/html'});  
  res.end('Hello World!');  
}).listen(8080);

* Lưu lại và mở command line và gõ lệnh node myfirst.js
* Sau đó mở trình duyệt lên và truy cập đường dẫn [http://localhost:8080](http://localhost:8080/) sẽ thấy dòng chữ hello world