**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN 2**

-----🙞🙜🕮🙞🙜-----

****

**MÔN: IoT và ứng dụng**

**ĐỀ 2: XÂY DỰNG NODE-RED DASHBOARD SỬ DỤNG DỮ**

**LIỆU CẢM BIẾN OPENEEW**

***Thành viên nhóm 10:***

1. ***Phùng Đức Mạnh – N20DCCN040***
2. ***Đặng Đức Trọng – N20DCCN079***

***Lớp: D20CQCNPM01-N***

**Thành phố Hồ Chí Minh, 05/01/202****4**

Mục lục nội dung

[Chương 1: Tổng quan (3-5 trang): 4 đoạn 4](#_Toc155278608)

[1.1. Sự cấp thiết 4](#_Toc155278609)

[1.2. Nội dung liên quan đề tài 4](#_Toc155278610)

[1.3. Phân tích đề tài 4](#_Toc155278611)

[1.4. Kết luận: qua thực nghiệm làm được gì? 5](#_Toc155278612)

[Chương 2: Mô hình đề xuất cho ứng dụng 6](#_Toc155278613)

[2.1. Thuật toán liên quan đề tài 6](#_Toc155278614)

[Chương 3: Triển khai ứng dụng 6](#_Toc155278615)

[3.1. Môi trường thực nghiệm 6](#_Toc155278616)

[a. Node.js là gì? 6](#_Toc155278617)

[b. Node-RED 7](#_Toc155278618)

[c. OpenEEW là gì? 7](#_Toc155278619)

[d. Cài đặt Node.js 7](#_Toc155278620)

[e. Cài đặt Node-RED 8](#_Toc155278621)

[f. Phạm vi các biến (context) trong node-red 10](#_Toc155278622)

[g. Thêm các palette cần thiết 10](#_Toc155278623)

[h. Chức năng các node cơ bản trong Node-RED 13](#_Toc155278624)

[3.2. Triển khai ứng dụng: 21](#_Toc155278625)

[Chương 4: Kết luận, hướng phát triển 25](#_Toc155278626)

[4.1. Kết quả thực nghiệm triển khai được gì? 25](#_Toc155278627)

[4.2. Ứng dụng vô gì? 25](#_Toc155278628)

[4.3. Tồn tại hạn chế 25](#_Toc155278629)

[4.4. Hướng phát triển trong tương lai 25](#_Toc155278630)

[References 25](#_Toc155278631)

MỤC LỤC HÌNH ẢNH

[Hình 1. Tải node.js 8](#_Toc155278569)

[Hình 2. Kiểm tra phiên node và npm bằng cmd 8](#_Toc155278570)

[Hình 3. Tải node-red bằng cmd 8](#_Toc155278571)

[Hình 4. Chạy node-red để tạo webserver 9](#_Toc155278572)

[Hình 5. Địa chỉ ip của webserver 9](#_Toc155278573)

[Hình 6. Giao diện của node-RED 10](#_Toc155278574)

[Hình 7. Tải palette trên giao diện node-RED 11](#_Toc155278575)

[Hình 8. Tải node-red-dashboard từ palette 12](#_Toc155278576)

[Hình 9. Tải node-red-contrib-web-worldmap từ palette 12](#_Toc155278577)

[Hình 10. Tải node-red-node-twilio từ palette 13](#_Toc155278578)

[Hình 11. Tải node-red-node-ui-table từ palette 13](#_Toc155278579)

[Hình 12. Biểu tượng inject node 14](#_Toc155278580)

[Hình 13. Chỉnh thuộc tính của inject node 14](#_Toc155278581)

[Hình 14. Ví dụ inject node 15](#_Toc155278582)

[Hình 15. Giá trị thuộc tính Payload kiểu JSON 15](#_Toc155278583)

[Hình 16. Biểu tượng debug node 15](#_Toc155278584)

[Hình 17. Biểu tượng function node 16](#_Toc155278585)

[Hình 18. Ví dụ về funciton node 16](#_Toc155278586)

[Hình 19. Ví dụ về function node 16](#_Toc155278587)

[Hình 20. Nội dung hàm trong function node 17](#_Toc155278588)

[Hình 21. Ví dụ về function node 17](#_Toc155278589)

[Hình 22. Nội dung hàm trong function node 17](#_Toc155278590)

[Hình 23. Biểu tượng change node 18](#_Toc155278591)

[Hình 24. Điều chỉnh method SET trong change node 18](#_Toc155278592)

[Hình 25. Điều chỉnh value thuộc tính của các biến msg, flow, global 18](#_Toc155278593)

[Hình 26. Ví dụ về change node 19](file:///C:\Users\ADMIN\Downloads\formatIOTcuoiky-1.docx#_Toc155278594)

[Hình 27. Ví dụ về điều chỉnh thuộc tính trong change node 19](#_Toc155278595)

[Hình 28. Biểu tượng switch node 19](#_Toc155278596)

[Hình 29. Ví dụ về switch node 20](#_Toc155278597)

[Hình 30. Ví dụ về điều chỉnh thuộc tính trong switch node 20](#_Toc155278598)

[Hình 31. Biểu tượng link in node 20](#_Toc155278599)

[Hình 32. Biểu tượng http request node 20](#_Toc155278600)

[Hình 33. Ví dụ về http request node 21](#_Toc155278601)

[Hình 34. Ví dụ điều chỉnh thuộc tính của http request node 21](#_Toc155278602)

[Hình 35. Giao diện xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW 22](#_Toc155278603)

[Hình 36. Mô hình xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW 22](#_Toc155278604)

[Hình 37. Mã giả luồng chương trình xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW 23](#_Toc155278605)

[Hình 38. Giao diện luồng chương trình trong node-red xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW 24](#_Toc155278606)

[Hình 39. Hiển thị dữ liệu lấy được từ tập dataset OpenEEW 25](#_Toc155278607)

# Chương 1: Tổng quan (3-5 trang): 4 đoạn

## Sự cấp thiết

* Trong thời buổi mà sự phát triển của công nghệ bùng nổ như hiện nay việc áp dụng công nghệ vào đời sống rất phổ biến trải rộng trong nhiều lĩnh vực của đời sống. Đặc biệt là lĩnh vực IOT.
* Việc xây dựng các ứng dụng thường mất rất nhiều thời gian để phát triển và đòi hỏi nhà phát triển phải có kiến thức khá chuyên sâu về lập trình.
* Từ những lý do trên, node-red ra đời như là một vị cứu tinh. Node red là một công cụ lập trình để kết nối các thiết bị phần cứng, API và dịch vụ trực tuyến.
* Nó cung cấp trình chỉnh sửa dựa trên trình duyệt giúp đễ dàng kết nối các flow với nhau bằng cách sử dụng các node trong palette.
* Node-red dễ sử dụng chính vì vậy người lập trình có thể nhanh chóng tạo các ứng dụng bằng node-red.

## Nội dung liên quan đề tài

* Tên đề tài: Ứng dụng IBM cloud và Node Red vào mạch đo lường và điều khiển bằng máy tính - Mã số: TR:2020-30/KCN-SV - Chủ nhiệm đề tài: Bùi Nguyễn Thiên Phước - Đơn vị quản lý về chuyên môn: Khoa Công nghệ - Thời gian thực hiện: Từ tháng 10 năm 2020 đến tháng 1 năm 2021
* Tên đề tài: Nghiên cứu ứng dụng hệ thống giám sát môi trường nuôi tôm theo thời gian thực dựa trên công nghệ IoT – Tác giả: Đỗ Văn Cần\*, Bùi Văn Vũ, Lương Ngọc Toàn, Nguyễn Quốc Bảo, Nguyễn Văn Quang - Khoa Kỹ thuật và Công nghệ, Trường Đại học Quy Nhơn, Việt Nam
* Tên đề tài: "Nghiên cứu xây dựng ứng dụng trên nền tảng Node RED phục vụ nhà thông minh," – Tác giả: Lê Tùng Giang, TS. Ngô Đức Thiện. Học viện công nghệ Bưu chính Viễn thông, 2021.

## Phân tích đề tài

* Đề tài "Xây dựng Node-RED Dashboard sử dụng dữ liệu cảm biến OpenEEW" có thể được phân tích từ các góc độ khác nhau để hiểu rõ hơn về mục tiêu, phạm vi và ý nghĩa của nó:

1. **Mục tiêu của đề tài**:

* Cung cấp cảnh báo sớm động đất: Mục tiêu chính của việc xây dựng Dashboard là để sử dụng dữ liệu từ cảm biến OpenEEW để cung cấp cảnh báo sớm và chuẩn bị cho sự kiện động đất.
* Hiển thị thông tin đáng tin cậy và dễ đọc: Mục tiêu là tạo ra một giao diện người dùng trực quan và dễ sử dụng, cung cấp thông tin cảnh báo động đất một cách dễ hiểu và chính xác.

1. **Phạm vi của đề tài**:

* Thu thập và xử lý dữ liệu từ cảm biến OpenEEW: Bắt đầu từ việc thu thập dữ liệu từ cảm biến OpenEEW, sau đó xử lý và chuyển đổi dữ liệu để có thể sử dụng được trong Node-RED.
* Thiết kế và xây dựng Dashboard: Tạo ra giao diện người dùng thông qua Node-RED Dashboard để hiển thị thông tin động đất từ dữ liệu cảm biến OpenEEW một cách trực quan.

1. **Ý nghĩa và lợi ích**:

* Cảnh báo sớm và bảo vệ an toàn: Đề tài mang lại lợi ích lớn trong việc cung cấp thông tin cảnh báo sớm về động đất, giúp người dân và nhà chức trách chuẩn bị và phản ứng kịp thời.
* Trực quan hóa thông tin và nhận thức cộng đồng: Giao diện người dùng trực quan giúp cải thiện nhận thức của cộng đồng về rủi ro và an toàn trong trường hợp động đất.
* Hỗ trợ quyết định và phản ứng khẩn cấp: Dữ liệu trực quan và cảnh báo sớm có thể hỗ trợ quyết định và phản ứng nhanh chóng của những người liên quan khi có sự kiện động đất xảy ra.

1. **Đối tượng sử dụng và tiềm năng ứng dụng:**

* Cộng đồng và dân cư: Dashboard có thể được sử dụng bởi cộng đồng và dân cư để nhận thông tin cảnh báo và chuẩn bị cho sự kiện động đất.
* Nhà chức trách và nhóm cứu hộ: Cung cấp dữ liệu hữu ích để nhà chức trách và nhóm cứu hộ có thể phản ứng nhanh chóng và hiệu quả khi có sự kiện động đất xảy ra.

1. **Tiềm năng phát triển và mở rộng:**

* Tối ưu hóa và mở rộng Dashboard: Có thể phát triển thêm các tính năng như cập nhật liên tục, tích hợp các dịch vụ khác, hoặc tạo ra các biểu đồ và báo cáo chi tiết hơn.
* Ứng dụng cho các khu vực khác: Mở rộng áp dụng của đề tài này sang các khu vực địa lý khác có nguy cơ động đất.
* Qua việc phân tích này, ta có cái nhìn tổng quan về mục tiêu, phạm vi và tiềm năng của đề tài xây dựng Node-RED Dashboard sử dụng dữ liệu từ cảm biến OpenEEW, cũng như lợi ích mà nó có thể mang lại cho cộng đồng và người dùng cuối.

## Kết luận: qua thực nghiệm làm được gì?

* Việc xây dựng một Node-RED Dashboard sử dụng dữ liệu từ cảm biến OpenEEW (Earthquake Early Warning - Cảnh báo sớm động đất) có thể mang lại nhiều lợi ích quan trọng:

1. **Cảnh báo sớm và an toàn:** Sử dụng dữ liệu từ cảm biến OpenEEW trong Node-RED Dashboard có thể cung cấp thông tin cảnh báo sớm về các sự kiện động đất. Người dùng có thể nhận được thông tin cảnh báo kịp thời để chuẩn bị và đưa ra các biện pháp an toàn.
2. **Trực quan hóa dữ liệu:** Node-RED Dashboard cung cấp giao diện trực quan để hiển thị dữ liệu từ cảm biến OpenEEW dưới dạng biểu đồ, biểu đồ đường thời gian, bản đồ và các thành phần trực quan khác. Điều này giúp người dùng dễ dàng theo dõi và hiểu thông tin về hoạt động động đất.
3. **Thông tin chi tiết và đa dạng:** Dashboard có thể hiển thị các thông tin chi tiết về động đất, như biểu đồ cường độ, vị trí, thời gian xảy ra, và dữ liệu liên quan khác từ cảm biến OpenEEW. Điều này giúp cung cấp một cái nhìn tổng quan và đa dạng về các sự kiện động đất.
4. **Quản lý rủi ro và phản ứng khẩn cấp:** Node-RED Dashboard có thể hỗ trợ trong việc quản lý rủi ro và phản ứng khẩn cấp bằng cách cung cấp thông tin cảnh báo nhanh chóng cho cộng đồng, tổ chức cứu hộ và nhà chức trách.
5. **Nghiên cứu và phát triển:** Dữ liệu từ Node-RED Dashboard cũng có thể được sử dụng cho mục đích nghiên cứu về động đất, phát triển các mô hình dự báo, và cải thiện các hệ thống cảnh báo động đất trong tương lai.

* Quan trọng nhất, việc xây dựng Node-RED Dashboard sử dụng dữ liệu từ cảm biến OpenEEW giúp nâng cao sự nhận thức và sẵn sàng phản ứng của cộng đồng trước nguy cơ động đất, từ đó giúp giảm thiểu thiệt hại và bảo vệ an toàn cho người dân và tài sản.

# Chương 2: Mô hình đề xuất cho ứng dụng

## Thuật toán liên quan đề tài

* Phát hiện địa chấn bằng cách đo gia tốc từ các cảm biến gia tốc (accelerometer sensors) openeew, dựa trên ngưỡng cao hơn mức nhiễu thông thường của gia tốc kế (Gia tốc kế là một cảm biến điện tử đo lực gia tốc tác dụng lên một vật thể, nhằm xác định vị trí của vật thể đó trong không gian và theo dõi chuyển động của vật thể đó).
* Gal là đơn vị đo gia tốc 1 cm/s2. Hoạt động địa chấn khi vượt quá 3cm/s2
* Các luồng Node-RED này quan sát dữ liệu gia tốc kế openeew theo thời gian thực và tính toán xem cảm biến có thể gặp phải hoạt động địa chấn hay không bằng thuật toán sau. Mỗi giây, hàm này nhận được mảng x/y/z dữ liệu rung động dưới giây. Các mảng dữ liệu được truyền vào hàm bên trong msg.payload.traces[0] Hàm javascript lặp qua dữ liệu rung để tìm kiếm gia tốc vượt quá 3 cm/giây2

# Chương 3: Triển khai ứng dụng

## Môi trường thực nghiệm

* Ngôn ngữ dụng: javascript
* Công cụ: node-RED, node.js

### Node.js là gì?

* Là môi trường mã nguồn mở chạy mã javascript trên đa nền tảng.
* Node.js chạy trên công cụ Javascript V8, cốt lỗi của Google Chrome.
* Ứng dụng Node.js chạy trong một quy trình duy nhất mà không tạo chuỗi mới cho mọi yêu cầu. Node.js cung cấp một tập hợp các nguyên hàm I/O không đồng bộ trong thư viện tiêu chuẩn của nó để ngăn mã JavaScript chặn và nói chung, các thư viện trong Node.js được viết bằng cách sử dụng mô hình không chặn, khiến hành vi chặn trở thành ngoại lệ thay vì chuẩn mực.
* Khi Node.js thực hiện thao tác I/O, như đọc từ mạng, truy cập cơ sở dữ liệu hoặc hệ thống tệp, thay vì chặn luồng và lãng phí chu kỳ CPU chờ đợi, Node.js sẽ tiếp tục các hoạt động khi có phản hồi.
* Điều này cho phép Node.js xử lý hàng nghìn kết nối đồng thời với một máy chủ mà không gây ra gánh nặng quản lý luồng đồng thời, đây có thể là nguồn gây ra lỗi đáng kể.
* Node.js có một lợi thế duy nhất là hàng triệu nhà phát triển giao diện người dùng viết JavaScript cho trình duyệt hiện có thể viết mã phía máy chủ ngoài mã phía máy khách mà không cần phải học một ngôn ngữ hoàn toàn khác.

### Node-RED

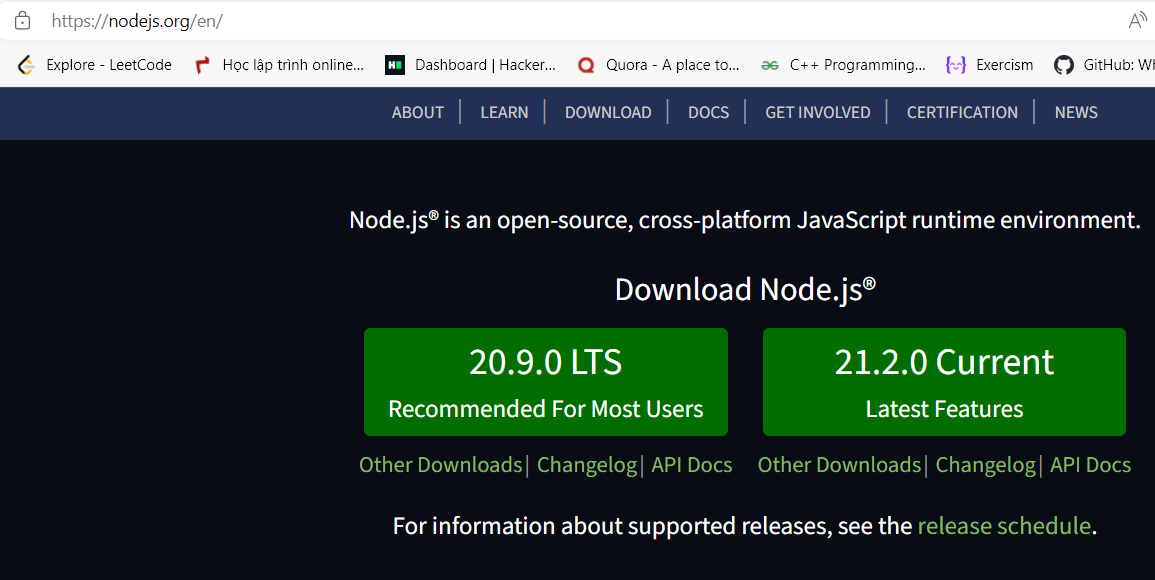
* Node-RED là một công cụ lập trình để kết nối các thiết bị phần cứng, API và dịch vụ trực tuyến theo những cách mới và thú vị.
* Nó cung cấp trình chỉnh sửa dựa trên trình duyệt giúp dễ dàng kết nối các luồng với nhau bằng cách sử dụng nhiều nút trong bảng màu có thể được triển khai vào thời gian chạy chỉ bằng một cú nhấp chuột.
* Được phát minh bởi J. Paul Morrison vào những năm 1970, lập trình dựa trên dòng chảy (flow) là một cách mô tả hành vi của ứng dụng như một mạng lưới các hộp đen hoặc “nút” (nodes) như chúng được gọi trong Node-RED. Mỗi node có một mục đích được xác định rõ ràng; nó được cung cấp một số dữ liệu, nó thực hiện điều gì đó với dữ liệu đó và sau đó truyền dữ liệu đó đi. Mạng chịu trách nhiệm về luồng dữ liệu giữa các node.
* Đây là một mô hình có khả năng trình bày trực quan rất tốt và làm cho nó dễ tiếp cận hơn với nhiều người dùng hơn. Nếu ai đó có thể chia nhỏ một vấn đề thành các bước riêng biệt, họ có thể nhìn vào một quy trình và hiểu được nó đang làm gì; mà không cần phải hiểu từng dòng mã riêng lẻ trong mỗi node
* Node-RED bao gồm thời gian chạy dựa trên Node.js mà bạn trỏ trình duyệt web vào để truy cập trình chỉnh sửa quy trình. Trong trình duyệt, bạn tạo ứng dụng của mình bằng cách kéo các nút từ bảng màu vào không gian làm việc và bắt đầu kết nối chúng lại với nhau. Chỉ với một cú nhấp chuột, ứng dụng sẽ được triển khai trở lại thời gian chạy nơi nó được chạy.
* Bảng màu của các nút có thể được mở rộng dễ dàng bằng cách cài đặt các node mới do cộng đồng tạo và các luồng bạn tạo có thể được chia sẻ dễ dàng dưới dạng tệp JSON..
* Tại sao nó được gọi là Node-RED? Cái tên này là một cách chơi chữ nhẹ nhàng của những từ nghe giống như 'Code Red'. Nó bị mắc kẹt và là một cải tiến lớn so với bất cứ thứ gì nó được gọi trong vài ngày đầu tiên. Phần 'Node' phản ánh cả mô hình lập trình luồng/nút cũng như thời gian chạy Node.JS cơ bản. Chúng tôi chưa bao giờ đi đến kết luận về ý nghĩa của phần 'RED'. “Rapid Event Developer” (“Nhà phát triển sự kiện nhanh") là một gợi ý, nhưng chúng tôi chưa bao giờ cảm thấy bị buộc phải chính thức hóa bất cứ điều gì. Chúng tôi gắn bó với 'Node-RED'.

### OpenEEW là gì?

* Công nghệ để phát hiện những dấu hiệu sớm nhất của trận động đất và đưa ra cảnh báo. Công nghệ này có thể giúp mọi người có thời gian quan trọng để tìm nơi trú ẩn hoặc thoát ra khỏi các tòa nhà trước khi có động đất.

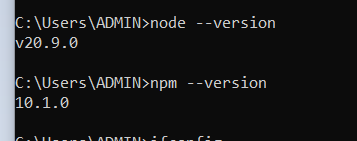
### Cài đặt Node.js

* Nếu chưa cài node.js thì ta nên tải package nodejs ở web dưới



Hình . Tải node.js

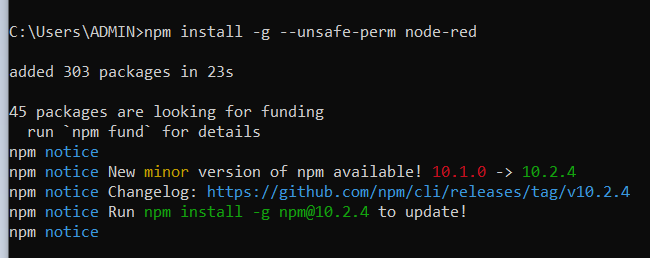
* Sau khi cài nodejs từ file .msi, kiểm tra version nodejs và npm bằng cmd:



Hình . Kiểm tra phiên node và npm bằng cmd

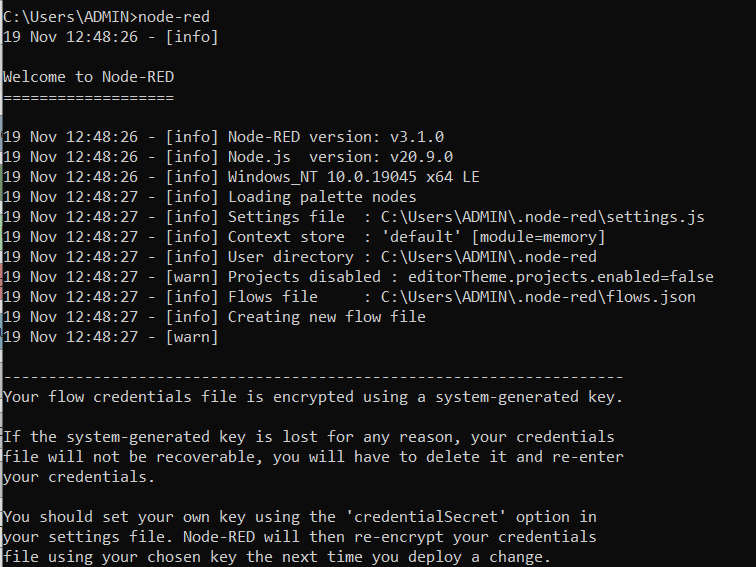
### Cài đặt Node-RED

* Sau khi cài thành công node.js, ta tiếp tục dùng cmd để tải node-red:

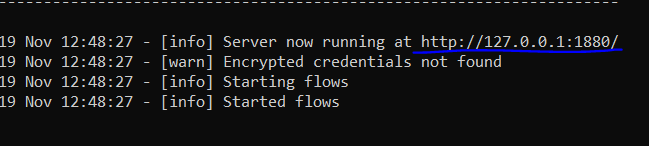


Hình . Tải node-red bằng cmd

* nhập node-red trên cmd để khởi động server node-red:

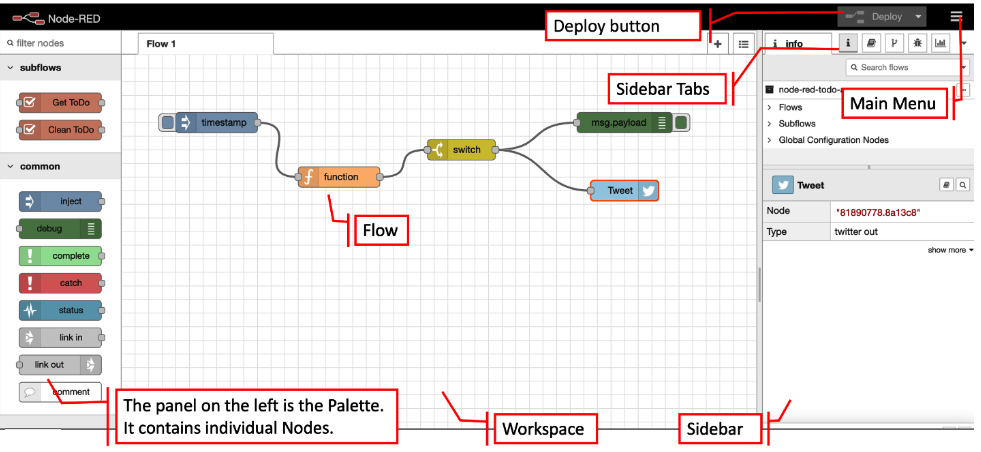


Hình . Chạy node-red để tạo webserver



Hình . Địa chỉ ip của webserver

* Copy địa chỉ ip trên và dán vào web browser để mở giao diện server node-red:
* Giao diện Flow 1 ở giữa màn hình sẽ chứa các node được kéo từ layout bên trái vào



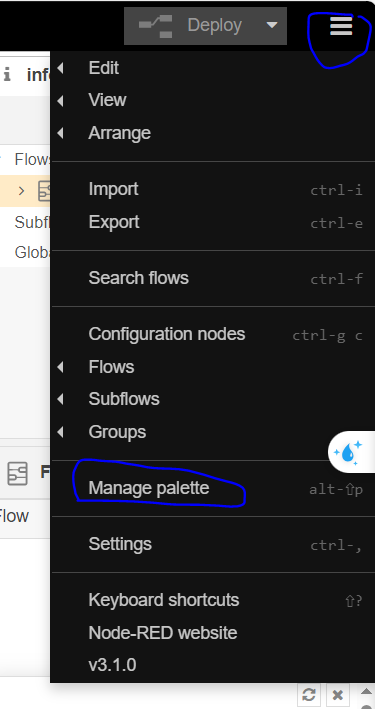
Hình . Giao diện của node-RED

### Phạm vi các biến (context) trong node-red

* Trong Node-RED, msg, flow, và global là các biến (được gọi là context) được sử dụng để lưu trữ và chia sẻ dữ liệu trong các phần khác nhau của ứng dụng Node-RED.
  + **msg**: msg là đối tượng chứa dữ liệu của mỗi thông điệp đi qua các nodes trong luồng. Mỗi thông điệp được đại diện bởi một đối tượng msg, và nó chứa dữ liệu gốc cũng như thông tin điều khiển (như msg.payload, msg.topic, và các trường thông tin khác). msg có thể được biến đổi, mở rộng hoặc truyền tiếp qua các nodes để thực hiện các tác vụ xử lý dữ liệu.
  + **flow**: flow là ngữ cảnh (context) ở mức luồng (flow-level context). Dữ liệu được lưu trữ trong flow có thể được truy cập và chia sẻ giữa tất cả các nodes trong cùng một luồng (flow). Điều này cho phép lưu trữ thông tin hoặc trạng thái ứng dụng mà có thể được truy cập từ bất kỳ node nào trong luồng đó.
  + **global**: global là ngữ cảnh ở mức toàn cầu (global-level context). Dữ liệu trong global có thể được truy cập và chia sẻ giữa tất cả các luồng (flows) và tất cả các nodes trong ứng dụng Node-RED. Điều này cho phép lưu trữ thông tin hoặc trạng thái mà có thể được truy cập từ mọi nơi trong ứng dụng.
* Cả ba ngữ cảnh này (msg, flow, global) đều cung cấp cách để lưu trữ dữ liệu tạm thời và chia sẻ nó giữa các nodes trong Node-RED, nhưng mỗi loại lại có phạm vi truy cập khác nhau (cục bộ cho từng thông điệp, cho mỗi luồng, hoặc toàn cầu). Điều này giúp bạn quản lý trạng thái ứng dụng, truyền dữ liệu và thực hiện các tác vụ xử lý dữ liệu linh hoạt trong quá trình xây dựng ứng dụng Node-RED của bạn.

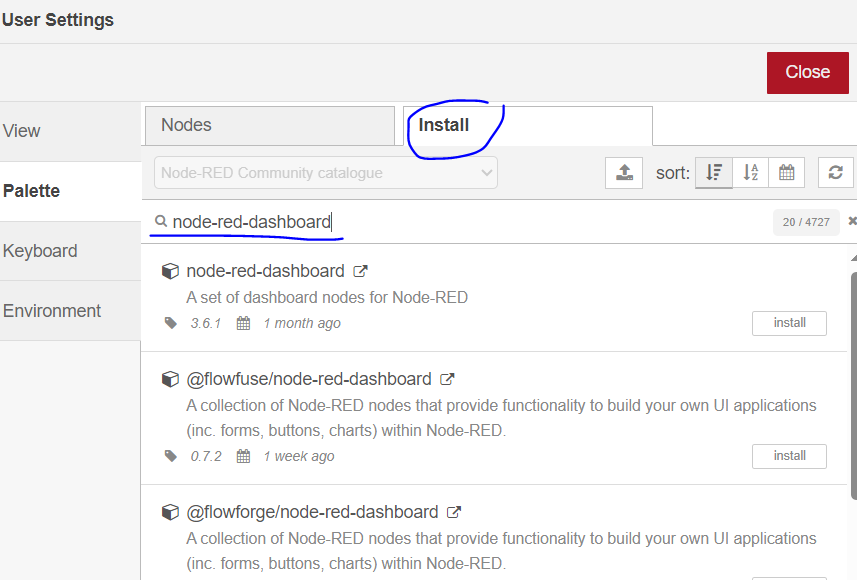
### Thêm các palette cần thiết

* [node-red-dashboard](https://flows.nodered.org/node/node-red-dashboard):
  + Cách cài đặt, từ trang node-red ở trên, vào Manage palette :



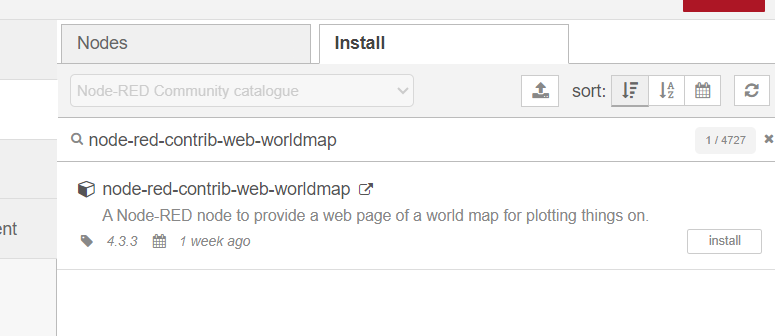
Hình . Tải palette trên giao diện node-RED

* + trong cửa sổ User setting được mở ra, chọn tab Install và tìm node-red-dashboard để tải thư viện này về



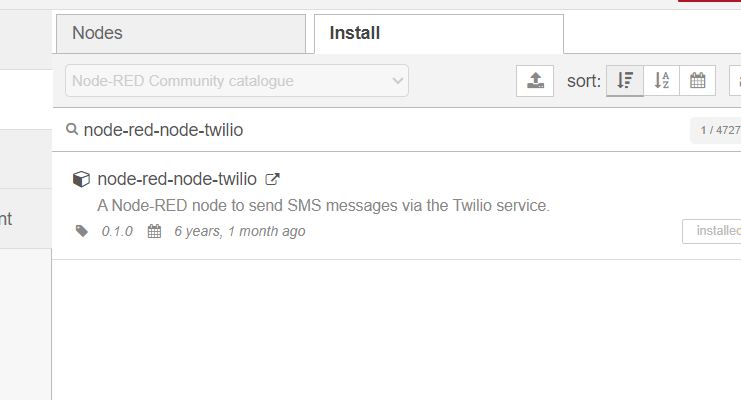
Hình . Tải node-red-dashboard từ palette

* [node-red-contrib-web-worldmap](https://flows.nodered.org/node/node-red-contrib-web-worldmap): map trên node-red, cách cài đặt tương tự vào Manage palette ở trên:



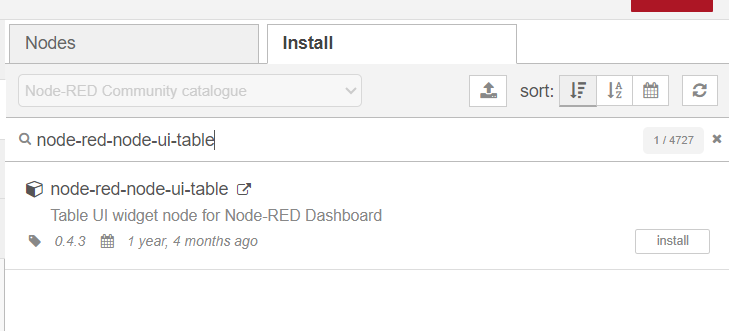
Hình . Tải node-red-contrib-web-worldmap từ palette

* [node-red-node-twilio](https://flows.nodered.org/node/node-red-node-twilio): gui sms den dien thoai bang service twilio cách cài đặt, vào Manage palette và tìm node-red-node-twilio:



Hình . Tải node-red-node-twilio từ palette

* [node-red-node-ui-table](https://flows.nodered.org/node/node-red-node-ui-table): tao table in node-red, cách cài đặt vào Manage palette và tìm



Hình . Tải node-red-node-ui-table từ palette

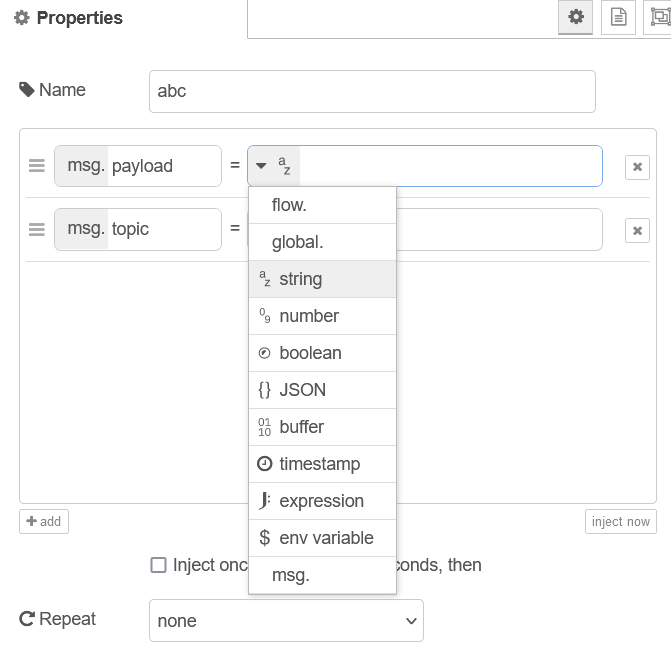
### Chức năng các node cơ bản trong Node-RED

* Inject node:



Hình . Biểu tượng inject node

* + Gửi một tin nhắn vào luồng có nhiều kiểu dữ liệu: strings, numbers, booleans, JavaScript objects, or flow/global context values, current time,…
  + có thể cấu hình 2 biến trong node inject



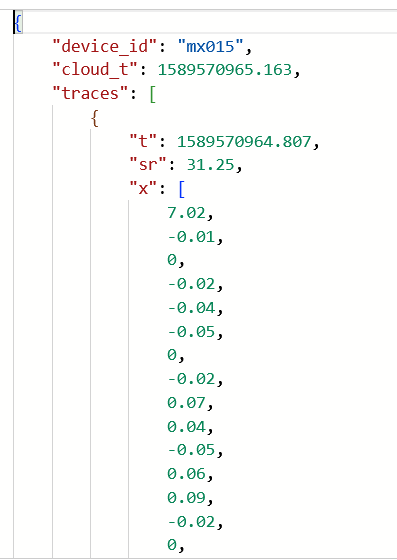
Hình . Chỉnh thuộc tính của inject node

* + thuộc tính msg.payload: có thể chọn kiểu dữ liệu để tiêm vào, ở dưới Repeat có thể thiết lập thời gian tự động tiêm sau khoảng thời gian.
  + Node Inject sử dụng trong project:
    - Trong cảnh báo động đất qua twillio



Hình . Ví dụ inject node

* + - Phần payload sẽ truyền vào kiểu JSON



Hình . Giá trị thuộc tính Payload kiểu JSON

* Debug node:



Hình . Biểu tượng debug node

* + Nút Gỡ lỗi khiến bất kỳ thông báo nào được hiển thị trong thanh bên Gỡ lỗi. Theo mặc định, nó chỉ hiển thị payload của tin nhắn nhưng có thể hiển thị toàn bộ đối tượng tin nhắn.
* Function node:

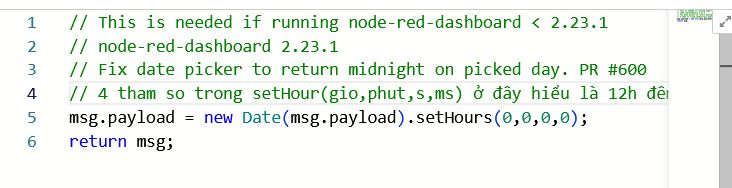


Hình . Biểu tượng function node

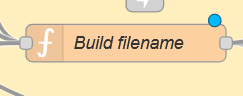
* + Một hàm JavaScript để chạy dựa trên các tin nhắn được nút nhận.
  + Các tin nhắn được truyền vào dưới dạng một đối tượng JavaScript có tên là msg.
  + Theo quy ước, nó sẽ có thuộc tính msg.payload chứa nội dung thư.
  + Hàm dự kiến ​​sẽ trả về một đối tượng tin nhắn (hoặc nhiều đối tượng tin nhắn), nhưng có thể chọn không trả về gì để tạm dừng luồng
  + Tab Khi bắt đầu chứa mã sẽ được chạy bất cứ khi nào nút được khởi động. Tab On Stop chứa mã sẽ được chạy khi nút dừng.
  + Function node sử dụng trong project:
    - Hàm trả về thời gian là 12h đêm



Hình . Ví dụ về funciton node



* + - Hàm build đường dẫn file tới tập dataset



Hình . Ví dụ về function node

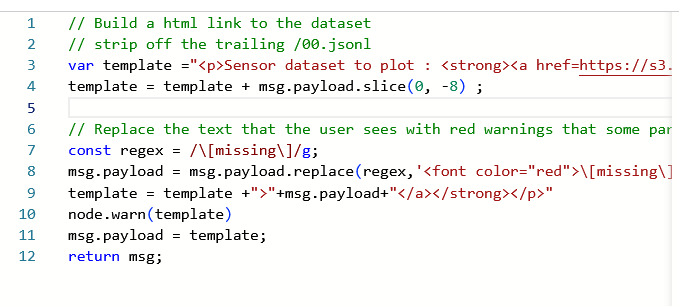


Hình . Nội dung hàm trong function node

* + - Hàm tạo ra một đường dẫn HTML chứa liên kết đến tập dữ liệu (dataset) từ msg.payload



Hình . Ví dụ về function node



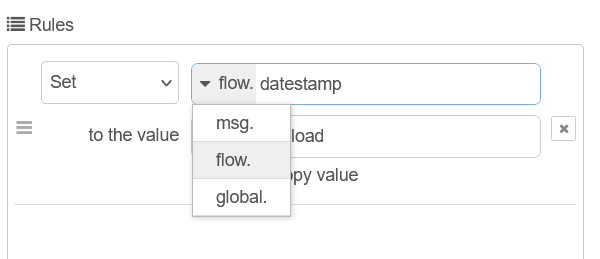
Hình . Nội dung hàm trong function node

* Change node:

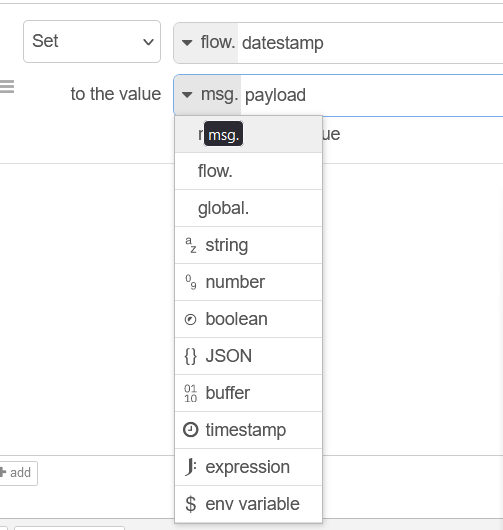


Hình . Biểu tượng change node

* + set, change, delete hoặc move các thuộc tính của tin nhắn (msg), bối cảnh luồng (flow) hoặc bối cảnh chung (global).



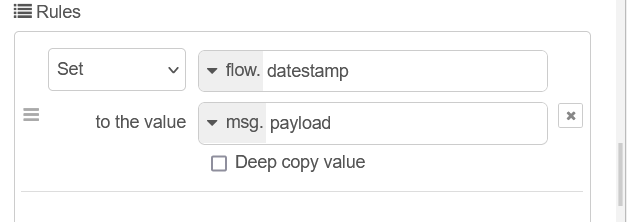
Hình . Điều chỉnh method SET trong change node



Hình . Điều chỉnh value thuộc tính của các biến msg, flow, global

* + Change node trong project:
    - Set giá trị của thuộc tính msg.payload khi lựa chọn ngày trên ui của node-red cho giá trị flow.datastamp (flow: biến phạm vi trong luồng)

Hình . Ví dụ về change node



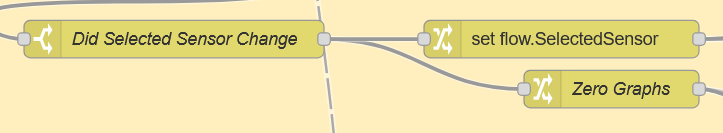
Hình . Ví dụ về điều chỉnh thuộc tính trong change node

* Switch node:



Hình . Biểu tượng switch node

* + Định tuyến các thông điệp dựa trên giá trị thuộc tính hoặc vị trí trình tự của chúng
  + Sẽ kiểm tra điều kiện của biến đầu vào (biến msg, flow, global,…). Nếu đúng điều kiện thì route đến node a, sai thì route đến node b.
  + Có 4 loại quy tắc:
    - Values: ==, !=, >, <, is null, is true, is between, …
    - Sequence: sẽ kiểm tra giá trị của trường cụ thể trong thông điệp msg có khớp với một chuỗi giá trị hoặc mẫu chuỗi xác định không.
    - chọn "JSONata Expression", có thể sử dụng các biểu thức JSONata để so sánh giá trị của một trường trong thông điệp msg với các điều kiện được xác định.
    - Otherwise: dùng quy tắc nếu các quy tắc trên không khớp
  + Switch node trong project:
    - Did selected sensor change (switch node) kiểm tra xem nếu device\_id của thuộc tính payload trong biến msg khác với thuộc tính SelectedSenssor của luồng hay không?



Hình . Ví dụ về switch node



Hình . Ví dụ về điều chỉnh thuộc tính trong switch node

* Link in node:



Hình . Biểu tượng link in node

* + là một node đặc biệt được sử dụng để tạo một liên kết hoặc URL truy cập trực tiếp vào một flow trong ứng dụng Node-RED. Node này cung cấp một URL mà người dùng có thể sử dụng để truy cập trực tiếp đến một phần của luồng hoặc một flow cụ thể mà không cần truy cập thông qua giao diện người dùng Node-RED.
* http request node:

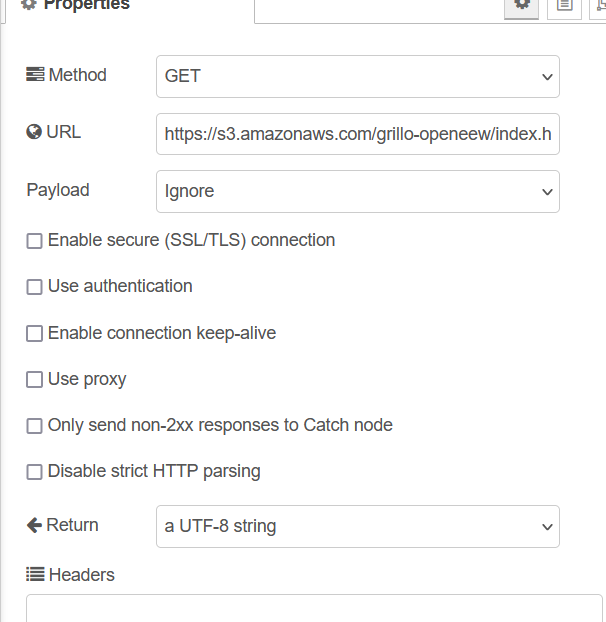


Hình . Biểu tượng http request node

* + Dùng gửi HTTP requests và nhận response
  + Input: url, method (‘GET’, ‘POST’,…), headers, cookies, payload (body of request),…
  + Output: payload, statusCode, headers, responseUrl (Trong trường hợp xảy ra bất kỳ chuyển hướng nào trong khi xử lý yêu cầu, thuộc tính này là url được chuyển hướng cuối cùng. Nếu không, url của yêu cầu ban đầu)
  + http request node trong project:
    - Gửi HTTP request đến cloud trên amazon để lấy dataset trong quá khứ



Hình . Ví dụ về http request node



Hình . Ví dụ điều chỉnh thuộc tính của http request node

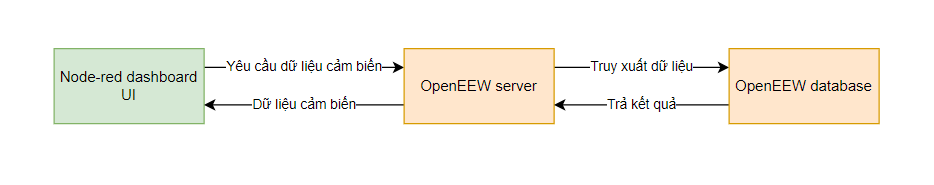
## Triển khai ứng dụng:

* **Biểu đồ hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW**: Luồng này hiển thị bảng điều khiển Node-RED cho phép người điều tra chọn một cảm biến thú vị và khoảng thời gian để nghiên cứu, sau đó truy vấn bộ dữ liệu OpenEEW. Sau đó, luồng sẽ vẽ dữ liệu cảm biến lịch sử trong một tập hợp biểu đồ.
* Form chọn thời gian, cảm biến và hiển thị dữ liệu từ cảm biến



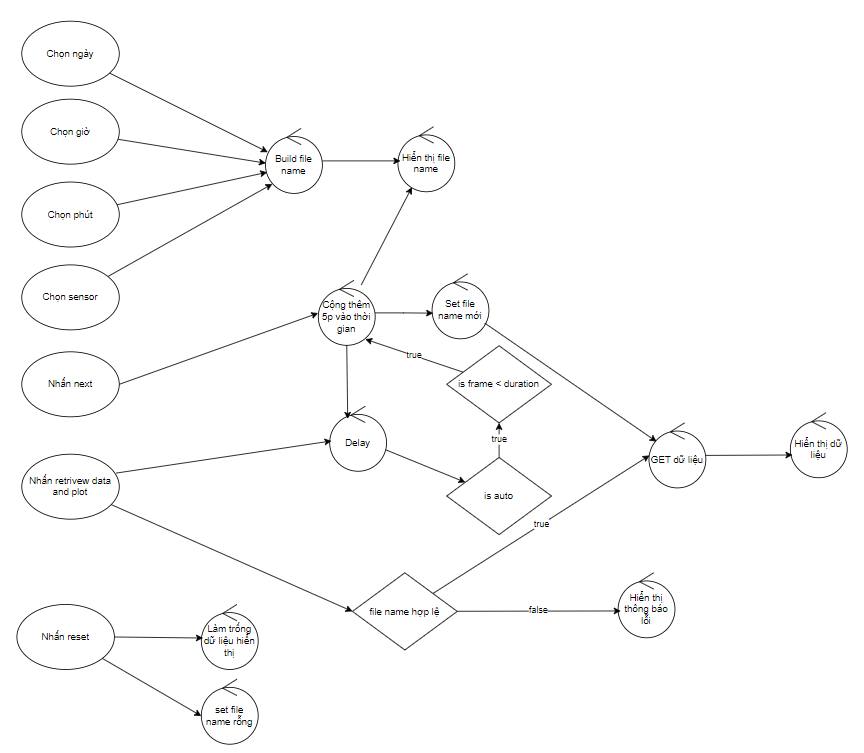
Hình . Giao diện xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW

* Mô hình ứng dụng: Mô hình 3 lớp



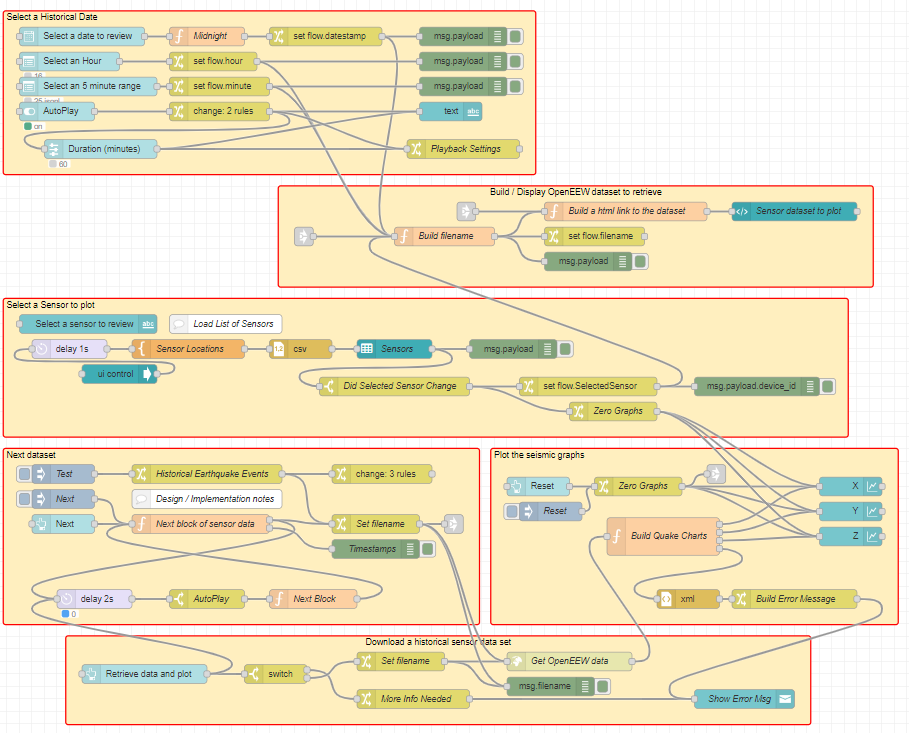
Hình . Mô hình xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW

* Triển khai ứng dụng trên node-red
  + Các chức năng của ứng dụng
    - Chọn thời gian (Ngày, giờ, phút).
    - Chọn sensor.
    - Chọn chức năng auto play và khoảng thời gian auto play.
    - Hiển thị đường link lấy dữ liệu.
    - Lấy dữ liệu và vẽ đồ thị.
    - Lấy dữ liệu và vẽ đồ thị cho khoảng thời gian 5p tiếp theo.
    - Reset link lấy dữ liệu và đồ thị.
    - Vẽ đồ thị
  + Mô tả luồng chương trình



Hình . Mã giả luồng chương trình xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW

* + - Build file name khi người dùng chọn thời gian và sensor rồi hiển thị.
    - Khi nhấn retrivew data and plot thì:
      * Kiểm tra file name:
        + File name không hợp lệ: thông báo lỗi.
        + File name hợp lệ: Get dữ liệu và hiển thị
      * Kiểm tra và thực hiện auto play nếu hợp lệ gọi vào chức năng next.
    - Khi nhấn next:
      * Cộng 5p vào file name.
      * Hiển thị file name mới.
      * Set file name mới.
      * Get data.
    - Khi nhấn reset:
      * Làm mới file name.
      * Clear dữ liệu hiển thị.
  + Các biến toàn cục trong flow
    - Flow.datestamp: ngày được chọn.
    - Flow.hour: giờ được chọn.
    - Flow.minute: phút được chọn.
    - Flow.duration: khoảng thời gian tiếp theo để auto play.
    - Flow.frame: thời gian trong khoảng thời gian auto play.
    - Flow.filename: file name để lấy dữ liệu, được thiết lập từ thời gian và sensor được chọn.
    - Flow.SelectedSensor: sensor id được chọn.
  + Triển khai trên node-red



Hình . Giao diện luồng chương trình trong node-red xem hoạt động địa chấn lịch sử từ bộ dữ liệu OpenEEW

* + Kết quả triển khai



Hình . Hiển thị dữ liệu lấy được từ tập dataset OpenEEW

# Chương 4: Kết luận, hướng phát triển

## Kết quả thực nghiệm triển khai được gì?

* Qua việc nghiên cứu đã triển khải được ứng dụng đọc và hiển thị dữ liệu từ các cảm biến động đất trên hệ thông OpenEEW.
* Người sử dụng có thể dễ dàng quan sát dữ liệu động đất mà các cảm biến động đất thu được dưới dạng biểu đồ.

## Tồn tại hạn chế

* Trong quá trình thực nghiệm đã gặp không ít những khó khăn cũng như những sáng kiến, ý tưởng từ đó nhận thấy được những tồn tại hạn chế của ứng dụng như sau: Ứng dụng chỉ đọc được các dữ liệu cũ, có sẵn trên OpenEEW mà không thể đọc được dữ liệu thời gian thực do đó không thể phát hiện và cảnh báo nếu như cảm biến ghi nhận được sự bất thường trong rung chuyển của mặt đất.

## Hướng phát triển trong tương lai

* Từ việc nhận thấy tồn tại hạn chế ở trên trong tương lai ứng dụng cần được phát triển để có thể thu thập được các dữ liệu thời gian thực từ các cảm biến gia tốc từ đó xây dựng hệ thống cảnh báo thời gian thực.

# References

|  |  |
| --- | --- |
| [1] | T. G. Lê and A. T. N. Đ. Thiện, "Nghiên cứu xây dựng ứng dụng trên nền tảng Node RED phục vụ nhà thông minh," Học viện công nghệ Bưu chính Viễn thông, 2021. |
| [2] | V. H. N. N. T. H. Nguyễn Thế Cường, "Ứng dụng mạng cảm biến xây dựng hệ thống quan trắc tự động liên tục môi trường lao động tại VNPT Thanh Hóa," *Tạp chí khoa học Đại học Hồng Đức,* 2019. |
| [3] | S. Pasquali, Mastering Node.js, Packt Publishing Ltd. Livery Place 35 Livery Street Birmingham B3 2PB, UK., 2013. |
| [4] | D. Guttman, Fullstack Node.js The Complete Guide to Building Production Apps with Node.js, N. Murray, Ed., 2019. |
| [5] | T. Hagino, Practical Node-RED Programming Learn powerful visual programming techniques and best practices for the web and IoT, Packt Publishing Ltd. Livery Place 35 Livery Street Birmingham B3 2PB, UK., 2021. |