**TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH**

**KHOA ĐIỆN – ĐIỆN TỬ**

**BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG**

****

**NGUYỄN MINH THÁI**

**LUẬN VĂN TỐT NGHIỆP**

**XÂY DỰNG HỆ THỐNG WEBSERVER**

**CHO HỆ THỐNG TỰ ĐỘNG**

**NGÀNH KỸ THUẬT VÀ ĐIỀU KHIỂN & TỰ ĐỘNG HÓA**

**TP. HỒ CHÍ MINH, 2024**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH**

**KHOA ĐIỆN – ĐIỆN TỬ**

**BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG**

**NGUYỄN MINH THÁI – 1910526**

**LUẬN VĂN TỐT NGHIỆP**

**XÂY DỰNG HỆ THỐNG WEBSERVER**

**CHO HỆ THỐNG TỰ ĐỘNG**

**BUIDING WEBSERVER APPLICATION**

**FOR AUTOMATION SYSTEM**

**NGÀNH KỸ THUẬT VÀ ĐIỀU KHIỂN & TỰ ĐỘNG HÓA**

**GIẢNG VIÊN HƯỚNG DẪN**

**TS. NGUYỄN TRỌNG TÀI**

**TP. HỒ CHÍ MINH, 2024**

ĐỒ ÁN ĐƯỢC HOÀN THÀNH TẠI

TRƯỜNG ĐẠI HỌC BÁCH KHOA – ĐHQG -HCM

Cán bộ hướng dẫn khóa luận tốt nghiệp :

(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Cán bộ nhận xét 1 :

(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Cán bộ nhận xét 2 :

(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Khóa luận tố nghiệp được bảo vệ tại Trường Đại học Bách Khoa, ĐHQG TP. HCM

Ngày ....... tháng .......năm ..........

Thành phần Hội đồng đánh giá khóa luận tốt nghiệp gồm :

(Ghi rõ họ, tên, học hàm, học vị của Hội đồng chấm bảo vệ khóa luận tốt nghiệp)



Xác nhận của Chủ tịch Hội đồng đánh giá khóa luận tốt nghiệp và Chủ nhiệm Bộ môn sau khi luận văn đã được sửa chữa (nếu có).

|  |  |
| --- | --- |
| **CHỦ TỊCH HỘI ĐỒNG** | **CHỦ NHIỆM BỘ MÔN** |

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM |
| **KHOA ĐIỆN – ĐIỆN TỬ** | Độc lập – Tự do – Hạnh phúc |
| **BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG** | *TP. HCM, ngày .... tháng..... năm .........* |

**NHẬN XÉT LUẬN VĂN TỐT NGHIỆP**

**CỦA CÁN BỘ HƯỚNG DẪN**

**Tên luận văn:**

**XÂY DỰNG HỆ THỐNG WEBSERVER**

**CHO HỆ THỐNG TỰ ĐỘNG**

**BUIDING WEBSERVER APPLICATION**

**FOR AUTOMATION SYSTEM**

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện** |  | **Cán bộ hướng dẫn** |
| Nguyễn Minh Thái | 1910526 | TS. Nguyễn Trọng Tài |

**Đánh giá Luận văn :**

1. Về cuốn báo cáo

|  |  |  |  |
| --- | --- | --- | --- |
| Số trang | \_\_\_\_\_\_\_\_\_\_ | Số chương | \_\_\_\_\_\_\_\_\_\_ |
| Số bảng số liệu | \_\_\_\_\_\_\_\_\_\_ | Số hình vẽ | \_\_\_\_\_\_\_\_\_\_ |
| Số tài liệu tham khảo | \_\_\_\_\_\_\_\_\_\_ | Sản phẩm | \_\_\_\_\_\_\_\_\_\_ |

Một số nhận xét về hình thức của cuốn báo cáo:

1. Về nội dung luận văn

1. Về tính ứng dụng

1. Về thái độ làm việc của sinh viên

**Đánh giá chung:** Đồ án đạt/không đạt yêu cầu của một đồ án tốt nghiệp cử nhân, xếp loại Giỏi/Khá/Trung bình

**Điểm từng sinh viên:**

Nguyễn Minh Thái: **........../10**

|  |  |
| --- | --- |
|  | **Cán bộ hướng dẫn** |
|  | (Ký tên và ghi rõ họ tên) |

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM |
| **KHOA ĐIỆN – ĐIỆN TỬ** | Độc lập – Tự do – Hạnh phúc |
| **BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG** | *TP. HCM, ngày .... tháng..... năm .........* |

**NHẬN XÉT LUẬN VĂN TỐT NGHIỆP**

**CỦA CÁN BỘ PHẢN BIỆN**

**Tên luận văn:**

**XÂY DỰNG HỆ THỐNG WEBSERVER**

**CHO HỆ THỐNG TỰ ĐỘNG**

**BUIDING WEBSERVER APPLICATION**

**FOR AUTOMATION SYSTEM**

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện** |  | **Cán bộ hướng dẫn** |
| Nguyễn Minh Thái | 1910526 | TS. Nguyễn Trọng Tài |

**Đánh giá Luận văn:**

1. Về cuốn báo cáo

|  |  |  |  |
| --- | --- | --- | --- |
| Số trang | \_\_\_\_\_\_\_\_\_\_ | Số chương | \_\_\_\_\_\_\_\_\_\_ |
| Số bảng số liệu | \_\_\_\_\_\_\_\_\_\_ | Số hình vẽ | \_\_\_\_\_\_\_\_\_\_ |
| Số tài liệu tham khảo | \_\_\_\_\_\_\_\_\_\_ | Sản phẩm | \_\_\_\_\_\_\_\_\_\_ |

Một số nhận xét về hình thức của cuốn báo cáo:

1. Về nội dung luận văn

1. Về tính ứng dụng

1. Về thái độ làm việc của sinh viên

**Đánh giá chung:** Đồ án đạt/không đạt yêu cầu của một đồ án tốt nghiệp cử nhân, xếp loại Giỏi/Khá/Trung bình

**Điểm từng sinh viên:**

Nguyễn Minh Thái: **........../10**

|  |  |
| --- | --- |
|  | **Cán bộ hướng dẫn** |
|  | (Ký tên và ghi rõ họ tên) |

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH | CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM |
| **KHOA ĐIỆN – ĐIỆN TỬ** | Độc lập – Tự do – Hạnh phúc |
| **BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG** | *TP. HCM, ngày .... tháng..... năm .........* |

**ĐỀ CƯƠNG CHI TIẾT**

|  |  |  |
| --- | --- | --- |
| **TÊN LUẬN VĂN:** **XÂY DỰNG HỆ THỐNG WEBSERVER CHO HỆ THỐNG TỰ ĐỘNG** | | |
| **Thời gian thực hiện: Từ ngày 4/1/2024 đến ngày 19/5/2024** | | |
| **Cán bộ hướng dẫn** | **TS. Nguyễn Trọng Tài** | |
| **Sinh viên thực hiện** | **Nguyễn Minh Thái – 1910526** | |
| **Nội dung đề tài:**   1. **Tổng quan**   Với sự phát triển của khoa học kỹ thuật và sự ra đời của công nghệ 4.0 thì việc giám sát và điểu khiển PLC thông qua internet là phổ biến và cần thiết, các thuật toán hỗ trợ kết nối web với PLC tạo điểu kiện cho các hãng PLC phát triển các chức năng public dữ liệu lên mạng viễn thông.  Webserver chạy trên nền tảng Nodejs chạy trên máy tính Server, thông qua địa chỉ IP  Public và cổng được định sẵn thì người dùng có thể truy cập dữ liệu của PLC gửi lên Webserver theo thời gian thực.   1. **Đối tượng**  * Mục tiêu: Mô phỏng quá trình, điểu khiển và giám sát hệ thống qua Internet * Phạm vi: Hệ thống trộn trong các nhà máy * Đối tượng: Hệ thống được điểu khiển bởi SIMATIC S7-1200 Controller của Siemens * Phương thức thực hiện: Lập trình mô phỏng hệ thống trên PLC S7-1200; Thiết kế Webserver để giao tiếp và giao diện SCADA nhằm điều khiển và giám sát dữ liệu | | |
| **Kế hoạch thực hiện:**   * Tìm hiểu mô hình Client-Server, MVC, giao thức Websocket * Tìm hiểu về ứng dụng web được xây dựng trên nền tảng Nodejs * Xây dựng một hệ thống mô phỏng trên TIA Portal, cách thức kết nối TIA tới Server * Xây dựng Server để giao tiếp tới PLC S7-1200 * Xây dựng Database để nhận dữ liệu từ Server vể * Xây dựng giao diện SCADA giám sát và thu thập dữ liệu lấy từ PLC S7-1200 * Tạo chức năng Report dữ liệu từ Server * Tạo chức năng đồ thị theo dõi dữ liệu theo thời gian * Xây dựng Database người dùng và phân quyền * Đóng gói ứng dụng * Public Server lên Internet | | |
| **TP.HCM, ngày....tháng.....năm.........** | | |
| **Xác nhận của Cán bộ hướng dẫn**  (Ký tên và ghi rõ họ và tên) | | **Sinh viên**  (Ký tên và ghi rõ họ và tên) |

**DANH SÁCH HỘI ĐỒNG BẢO VỆ LUẬN VĂN**

Hội đồng chấm đồ án tốt nghiệp, thành lập theo Quyết định số

Ngày của Hiệu trường Trường Đại học Bách Khoa TP.HCM.

1. ........................... - Chủ tịch
2. ........................... - Thư ký
3. ........................... - Ủy viên
4. ........................... - Ủy viên
5. ........................... - Ủy viên

**LỜI CẢM ƠN**

Kiến thức là vô cùng rộng lớn và đa dạng, vì vậy việc nắm bắt kiến thức là một hành trình gian khổ và vất vả. Trên con đường đó không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù nhiều hay ít, dù trực tiếp hay gián tiếp. Trong suốt quá trình học tập tại trường Đại học Bách Khoa TP.HCM, chúng em đã nhận được sự giúp đỡ tận tình của các thầy cô giáo, điều đó thật đáng quý và trân trọng.

Với lòng biết ơn sâu sắc nhất, em xin gửi lời cảm ơn đến quý thầy cô ở Khoa Điện – Điện tử đã truyền đạt vốn kiến thức quý báu cho chúng em trong suốt thời gian học tập tại trường. Riêng với luận văn tốt nghiệp này chúng em xin bày tỏ lòng biết ơn chân thành tới Thầy TS. Nguyễn Trọng Tài – Bộ môn Điều khiển và Tự động hóa là giáo viên hướng dẫn em đã tận tình chỉ bảo cũng như tạo điều kiện thuận lợi giúp chúng em trong suốt quá trình làm luận văn.

Xin được chân thành cảm ơn các bạn cùng khóa đã luôn giúp đỡ, sát cánh trong suốt quá trình làm luận văn, những lúc khó khăn cùng nhau bước qua quãng đời sinh viên tươi đẹp này. Sau cùng, chúng em xin kính chúc các thầy, các cô thật dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau.

**MỤC LỤC**

[TRANSMISSION CONTROL PROTOCOL 16](#_Toc164549254)

[TÓM TẮT LUẬN VĂN BẰNG TIẾNG VIỆT 17](#_Toc164549255)

[ABSTRACT 18](#_Toc164549256)

[Chương 1. GIỚI THIỆU ĐỀ TÀI 19](#_Toc164549257)

[1.1. Lý do chọn đề tài 19](#_Toc164549258)

[1.2. Mục tiêu 19](#_Toc164549259)

[1.3. Đối tượng và phạm vi 20](#_Toc164549260)

[1.4. Nhiệm vụ 20](#_Toc164549261)

[Chương 2. TỔNG QUAN HỆ THỐNG 21](#_Toc164549262)

[2.1. Kiến trúc hệ thống 21](#_Toc164549263)

[2.2. Sơ đồ kết nối 21](#_Toc164549264)

[Chương 3. CƠ SỞ LÝ THUYẾT 23](#_Toc164549265)

[3.1. PLC 23](#_Toc164549266)

[3.1.1. PLC là gì? 23](#_Toc164549267)

[3.1.2. Cấu tạo và phân loại PLC 23](#_Toc164549268)

[3.1.3. PLC hoạt động như thế nào? 24](#_Toc164549269)

[3.1.4. Ưu và nhược điểm của PLC 25](#_Toc164549270)

[3.1.5. PLC được lập trình như thế nào 25](#_Toc164549271)

[3.1.6. Ứng dụng của PLC 26](#_Toc164549272)

[3.2. Giao thức RFC1006 (ISO on TCP) 26](#_Toc164549273)

[3.3. Giao thức SocketIO 27](#_Toc164549274)

[3.3.1. Khái niệm 27](#_Toc164549275)

[3.3.2. Lợi thế khi sử dụng SocketIO 30](#_Toc164549276)

[3.4. Nền tảng Nodejs 32](#_Toc164549277)

[3.4.1. Tổng quan 33](#_Toc164549278)

[3.4.2. Ứng dụng viết bằng Nodejs 33](#_Toc164549279)

[3.5. Giới thiệu về MySQL Server 34](#_Toc164549280)

[3.5.1. SQL là gì? 34](#_Toc164549281)

[3.5.2. MySQL Server là gì? 35](#_Toc164549282)

[3.5.3. MySQL hoạt động như thế nào? 36](#_Toc164549283)

[3.6. Tổng quan về mô hình MVC 37](#_Toc164549284)

[3.6.1. Khái niệm 37](#_Toc164549285)

[3.6.2. Cấu trúc mô hình MVC 37](#_Toc164549286)

[3.6.3. Luồng đi trong mô hình MVC 38](#_Toc164549287)

[3.6.4. Ưu và nhược điểm của MVC 39](#_Toc164549288)

[3.6.5. Ứng dụng 39](#_Toc164549289)

[Chương 4. PHÂN TÍCH VÀ TRIỂN KHAI 40](#_Toc164549290)

[4.1. Lập trình PLC 40](#_Toc164549291)

[4.1.1. Tổng quát hệ thống trộn 40](#_Toc164549292)

[4.1.2. Nguyên lý vận hành hệ thống 41](#_Toc164549293)

[4.1.3. Thiết lập cài đặt cho thiết bị 42](#_Toc164549294)

[4.1.4. Chương trình PLC 44](#_Toc164549295)

[4.1.5. Khối trạng thái thiết bị (FC\_EQUIPMENT\_STATUS[FC3]) 50](#_Toc164549296)

[4.1.6. Các khối dữ liệu (Data[DB2], DB\_Alarm[DB4], DB\_Webserver[DB6]) 51](#_Toc164549297)

[4.2. Lập trình Webserver 53](#_Toc164549298)

[4.2.1. Giới thiệu 53](#_Toc164549299)

[4.2.2. Xây dựng Model MVC 56](#_Toc164549300)

[4.2.3. Xây dựng Server 57](#_Toc164549301)

[4.2.4. Xây dựng Database 59](#_Toc164549302)

[4.2.5. Kết nối Server với PLC 61](#_Toc164549303)

[4.3. Thiết kế giao diện người dùng(UI) 64](#_Toc164549304)

[4.3.1. Giới thiệu 64](#_Toc164549305)

[4.3.2. Thiết kế giao diện người dung UI 66](#_Toc164549306)

[Chương 5. KẾT QUẢ THỰC HIỆN VÀ ĐÁNH GIÁ 81](#_Toc164549307)

[5.1. Kết quả thức hiện 81](#_Toc164549308)

[5.2. Đánh giá 93](#_Toc164549309)

[Chương 6. KẾT LUẬN 94](#_Toc164549310)

[TÀI LIỆU THAM KHẢO 95](#_Toc164549311)

**DANH MỤC HÌNH ẢNH**

[Hình 2.1. Kiến trúc hệ thống phân tầng 21](#_Toc164548279)

[Hình 2.2. Sơ đồ kết nối 21](#_Toc164548280)

[Hình 3.1. Các loại PLC 23](#_Toc164548281)

[Hình 3.2. Cấu trúc PLC 24](#_Toc164548282)

[Hình 3.3. Mô hình Client - Server 27](#_Toc164548283)

[Hình 3.4. Giao thức Websocket 28](#_Toc164548284)

[Hình 3.5. Giao thức HTTP-long polling 28](#_Toc164548285)

[Hình 3.6. Những trình duyệt hỗ trợ được thử nghiệm ở Sauce Labs 31](#_Toc164548286)

[Hình 3.7. Room 31](#_Toc164548287)

[Hình 3.8. Đa luồng 32](#_Toc164548288)

[Hình 3.9. Nodejs Platform 32](#_Toc164548289)

[Hình 3.10. Những công ty lớn sử dụng Nodejs 34](#_Toc164548290)

[Hình 3.10. MySQL Server 35](#_Toc164548291)

[Hình 3.11. Minh Họa Database 36](#_Toc164548292)

[Hình 3.12. Mô hình Client-Server 36](#_Toc164548293)

[Hình 3.13. Cấu trúc của MCV 37](#_Toc164548294)

[Hình 4.1. Hệ thống trộn 40](#_Toc164548295)

[Hình 4.2. Cho phép truy cập dữ liệu trong PLC 42](#_Toc164548296)

[Hình 4.3. Cho phép truy cập PUT/GET 42](#_Toc164548297)

[Hình 4.4. System and Clock memory 43](#_Toc164548298)

[Hình 4.5. IP Address cho PLC1 43](#_Toc164548299)

[Hình 4.6. IP Address cho PLC2 44](#_Toc164548300)

[Hình 4.7. Các khối trong chương trình PLC 44](#_Toc164548301)

[Hình 4.8. 2 Mode hoạt động 45](#_Toc164548302)

[Hình 4.9. Gửi trạng thái thiết bị qua webserver 46](#_Toc164548303)

[Hình 4.10. Start/Stop hệ thống 46](#_Toc164548304)

[Hình 4.11. Chương trình cho chế độ Auto 47](#_Toc164548305)

[Hình 4.14. Chương trình chế độ Manual 49](#_Toc164548306)

[Hình 4.15. Thiết bị được điểu khiển bằng tay 49](#_Toc164548307)

[Hình 4.16. Các trạng thái của thiết bị 50](#_Toc164548308)

[Hình 4.17. Dữ liệu giao tiếp với Webserver 52](#_Toc164548309)

[Hình 4.16. Dữ liệu trạng thái Alarm 52](#_Toc164548310)

[Hình 4.17. Dữ liệu cho từng giai đoạn của chế độ Auto 52](#_Toc164548311)

[Hình 4.18. Sơ đồ cấu trúc giao tiếp API 53](#_Toc164548312)

[Hình 4.19. Sơ đồ cách thức giao tiếp giữa Server – Client 55](#_Toc164548313)

[Hình 4.20. Áp dụng Model MVC 56](#_Toc164548314)

[Hình 4.21. LVTN\_Database 59](#_Toc164548315)

[Hình 4.22. Bảng dữ liệu tạo trên MySQL 59](#_Toc164548316)

[Hình 4.23. Cấu trúc phân tầng trong lập trình giao diện 64](#_Toc164548317)

[Hình 4.24. Database người dùng 70](#_Toc164548318)

[Hình 5.1 Giao diện chính của SCADA khi đăng nhập vào tài khoản tương ứng 81](#_Toc164548319)

[Hình 5.2. Giao diện khi chưa đăng nhập 82](#_Toc164548320)

[Hình 5.3. Trang đăng nhập 83](#_Toc164548321)

[Hình 5.4. Tài khoản đăng nhập và kiểm tra thời điểm đăng nhập 83](#_Toc164548322)

[Hình 5.5. Trang Home 84](#_Toc164548323)

[Hình 5.6. Trang Main 84](#_Toc164548324)

[Hình 5.7. Trang Database 85](#_Toc164548325)

[Hình 5.8a. Trend 1 86](#_Toc164548326)

[Hình 5.8b. Trend 2 86](#_Toc164548327)

[Hình 5.8c. Trend 3 87](#_Toc164548328)

[Hình 5.9. Mô phỏng lỗi và khi kết thúc lỗi 87](#_Toc164548329)

[Hình 5.11a. Có lỗi 88](#_Toc164548330)

[Hình 5.11b. Kết thúc lỗi 88](#_Toc164548331)

[Hình 5.12 Trang About 89](#_Toc164548332)

[Hình 5.13. Dữ liệu report khối lượng 1 90](#_Toc164548333)

[Hình 5.14. Dữ liệu report khối lượng tổng 90](#_Toc164548334)

[Hình 5.15. Dữ liệu report Alarm 90](#_Toc164548335)

[Hình 5.16. Ứng dụng SCADA trên điện thoại 91](#_Toc164548336)

[Hình 5.17. Giao diện chờ của ứng dụng SCADA trên điện thoại 92](#_Toc164548337)

[Hình 5.18. Giao diện trên điện thoại 92](#_Toc164548339)

**DANH MỤC VIẾT TẮT**

|  |  |
| --- | --- |
| **PLC** | PROGRAMMABLE LOGIC CONTROLLER |
| **API** | APPLICATION PROGRAMMING INTERFACE |
| **HTTP** | HYPERTEXT TRANSFER PROTOCOL |
| **TCP** | TRANSMISSION CONTROL PROTOCOL |
| **DB** | DATA BLOCK |
| **FB** | FUNCTION BLOCK |
| **OB** | ORGANIZATION BLOCK |
| **UI** | USER INTERFACE |
| **SCADA** | SUPERVISORY CONTROL AND DATA ACQUISITION |
| **MVC** | MODEL-VIEW-CONTROLLER |
| **JSON** | JAVASCRIPT OBJECT NOTATION |
| **URL** | UNIFORM RESOURCE LOCATOR |
| **HTML** | HYPERTEXT MARKUP LANGUAGE |
| **CSS** | CASCADING STYLE SHEETS |
| **SQL** | STRUCTURED QUERY LANGUAGE |

TÓM TẮT LUẬN VĂN BẰNG TIẾNG VIỆT

Đề tài đồ án em chọn chủ yếu nghiên cứu vể mô hình tự động hóa cơ bản bao gồm các chức năng điều khiển bằng PLC , giám sát và thu thập dữ liệu thông qua database và hiển thị lên web app. Mô hình em thực hiện là 2 hệ thống trộn được điều khiển bằng 2 con cpu riêng biệt. Các trạng thái và dữ liệu thu thập từ hệ thống sẽ gửi về Server và từ Server gửi về database. MySQL Server chứa database thu thập dữ liệu và dữ chúng trong ổ cứng laptop. Web app được xây dựng trên nền tảng Nodejs và mô hình MVC để hiển thị dữ liệu, đồng thời có thể theo dõi và quan sát dữ liệu thông qua đường dẫn của web trên Internet. Cuối cùng là một giao diện được xây dựng trên mọi trình duyệt để mọi thiết bị di động đều có thể thích ứng.

Trong luận văn, em sử dụng giao thức SocketIO để giao tiếp giữa máy khách (Client) và máy chủ (Server) và cho phép truyền tải dữ liệu từ máy chủ đến máy khách và ngược lại từ đó xây dựng hệ thống giám sát và thu thập dữ liệu trên Web App và có thể tích hợp vào hệ thống công nghiệp.

Sau khi nghiên cứu và hoàn thành luận văn, em đã hoàn chỉnh được mô hình và các ứng dụng client. Mô hình hoạt động đúng với yêu cầu đặt ra ban đầu, và có tính khả thi khi đặt vào các hệ thống công nghiệp với quy mô dữ liệu lớn hơn.

ABSTRACT

The project topic I chose is mainly to research the basic automation model including PLC control functions, monitoring and data collection through the database and displaying it on the web app. The model I implemented is 2 mixing systems controlled by 2 separate CPUs. Statuses and data collected from the system will be sent to the Server and from the Server sent to the database. MySQL Server contains a database that collects data and stores it on the laptop hard drive. Web app is built on Nodejs platform and MVC model to display data, and can track and observe data through web links on the Internet. Finally, an interface built across all browsers so that any mobile device can adapt.

In the thesis, I use the SocketIO protocol to communicate between the client and the server and allow data transmission from the server to the client and vice versa, thereby building a monitoring and control system. Collect data on Web App and can integrate into industrial systems.

After researching and completing the thesis, I have completed the model and client applications. The model works in accordance with the initial requirements, and is feasible when placed in industrial systems with larger data scales.

Chương 1. GIỚI THIỆU ĐỀ TÀI

* 1. Lý do chọn đề tài

Theo sự phát triển của công nghệ thì chúng ta bước vào thời đại của việc điều khiển và giám sát mọi thứ từ nền tảng mạng internet. Và trong công nghiệp cũng vậy, nhu cầu của việc điều khiển và giám sát các loại máy móc một cách tiên dụng từ xa ở bất cứ đâu trở nên cấp thiết.

Dựa trên nhu cầu đó nhiều biện pháp điều khiển và giám sát từ xa đã ra đời. Tuy nhiên, khi giám sát từ xa thì cũng đi kèm với nhiều vấn đề như là biện pháp kết nối, chuẩn thông tin truyền, độ bảo mật, chi phí tiết kiệm, độ khó của việc thiết lập, sự nhanh chóng tiện lợi, khả năng tùy chỉnh lớn để có thể xử lý dữ liệu,…

Webserver chạy trên nền tảng Nodejs chạy trên máy tính Server, thông qua địa chỉ IP

Public và cổng được định sẵn thì người dùng có thể truy cập dữ liệu của PLC gửi lên Webserver theo thời gian thực. Webserver có thể chuyển đổi thành ứng dụng di động cài đặt trên các hệ điều hành android hoặc IOS, cho phép người dùng truy cập từ xa một cách linh hoạt và trực quan nhất.

* 1. Mục tiêu

Xây dựng Webserver dựa trên nền tảng Nodejs thực hiện kết nối với PLC, từ đó đưa dữ liệu theo thời gian thực lên internet.

Tổng quan:

|  |  |  |
| --- | --- | --- |
| **Thuyết minh** | **Sơ đồ nguyên lý** | |
| * PLC: Điểu khiển hệ thống, giám sát hệ thống, thu thập thông tin truyền và nhận dữ liệu của hệ thống bồn nước * API: Đây chính là Webserver, là một ứng dụng có thức năng lấy dữ liệu từ PLC, đồng thời nhận những tín hiệu điểu khiển hoặc dữ liệu cài đặt từ Web Browser đưa xuống để ghi vào PLC * Web Browser: là các ứng dụng trình duyệt web | PLC  API  Web Browser |  |

* 1. Đối tượng và phạm vi
* Đối tượng: Hệ thống trộn được điều khiển bởi SIMATIC S7-1200 Controller của Siemens
* Phạm vi: Hệ thống trộn trong các nhà máy
  1. Nhiệm vụ

Tìm hiểu và xây dựng kiến trúc hệ thống

* Tìm hiểu giao thức SocketIO, thư viện giao tiếp với các dòng PLC Siemens
* Xây dựng Web API và cách xây dựng từ MVC model
* Viết chương trình trên PLC, mở chức năng hỗ trợ webserver trên PLC

Xây dựng 2 hệ thống trộn với các biến cơ bản

Xây dựng code giao tiếp với Web API

* Tạo Web API kết nối và giao tiếp tới PLC
* Tạo Database để lưu dữ liệu từ Web API về
* Thiết lập giao diện cho Web và các API:
  + Viết controller quản lý dữ liệu và hành động trên giao diện. Sử dụng ngôn ngữ Javascript trên nền tảng Nodejs để xây dựng Model
  + Tích hợp API đã viết trước đó vào và tạo thành chương trình giữa Controller với API
  + Thiết kế SCADA trên máy tính Server. Tính năng: thu thập dữ liệu và điều khiển giám sát, cảnh báo, vẽ đồ thị, bảo mật và quản lý người dùng.

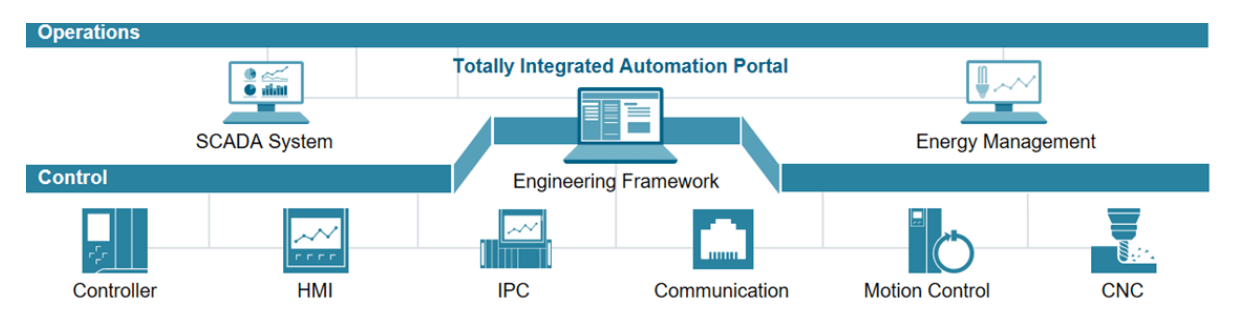
Chạy thử, kiểm tra, đánh giá hệ thống:

* Kiểm tra chức năng của các thành phần hệ thống
* Đánh giá chất lượng điểu khiển

Chương 2. TỔNG QUAN HỆ THỐNG

* 1. Kiến trúc hệ thống

Bước đầu tiên trong quá trình thiết kế và xây dựng hệ thống tự động hóa là phải lên được kiến trúc hệ thống. Kiến trúc hệ thống sẽ bao gồm nhiều cấp chức năng, mỗi cấp sẽ có các thiết bị chuyên dụng để thực hiện vai trò của từng cấp. Ngoài ra, kiến trúc hệ thống còn phải nêu được phương pháp liên kết và truyền thông giữa các cấp chức năng với nhau để đảm bảo hệ thống hoạt động liền mạch, trơn tru và đạt hiệu quả. Kiến trúc hệ thống có vai trò quan trọng, vì nó giúp người thiết kế, người vận hành và người giám sát dễ dàng hình dung được cấu tạo và hoạt động của hệ thống, từ đó tăng hiệu quả thiết kế, kiểm tra và giám sát hệ thống.



Hình 2.1. Kiến trúc hệ thống phân tầng

IP:50.50.50.10



1

* 1. Sơ đồ kết nối

|  |  |
| --- | --- |
| 1 | Trạm trộn 1 |
| 2 | Trạm trộn 2 |
| 3 | Database |
| 4 | PC Server |
| 5 | Các thiết bị di động |
| 6 | PLC S71200 |

IP:50.50.50.100

5

4

3

Socket.IO

Socket.IO

internet



profinet

IP:50.50.50.20



2

profinet

6

6

IP:50.50.50.1

IP:50.50.50.1

Hình 2.2. Sơ đồ kết nối

Hệ thống này bao gồm các cấp cơ bản sau:

**Cấp điều khiển:** Bộ điều khiển (Controller) là thiết bị thực hiện thuật toán điểu khiển thông qua chương trình được kỹ sư lập trình sẵn, đồng thời thu thập dữ liệu từ quá trình. Đối với luận văn này sinh viên sử dụng bộ điều khiển là PLC S7 – 1200 của hãng Siemens:

* Thiết bị sử dụng: Simatic S7-1200 Controller CPU 1214 C DC/DC/DC
* Phần mềm lập trình: TIA Portal V17

**Cấp giám sát:** Web API là là một ứng dụng có thức năng lấy dữ liệu từ PLC, đồng thời nhận những tín hiệu điểu khiển hoặc dữ liệu cài đặt từ Web Browser đưa xuống để ghi vào PLC. SCADA là hệ thống giám sát và thu thập dữ liệu nhằm hỗ trợ con người trong quá trình giám sát và điểu khiển từ xa một hệ thống tự động hóa hoặc một nhà máy.

Trong luận văn này, sinh viên xây dựng Web API và SCADA bằng nền tảng Nodejs, đây là một open-source và framework đa nền tảng (cross-platform) cho việc xây dựng những ứng dụng hiện tại như Web Apps,IoT và Backend cho Mobile.

* Nền tảng phần mềm: Nodejs module + Visual Studio Code
* Giao thức truyền thông: Socket.IO

Chương 3. CƠ SỞ LÝ THUYẾT

* 1. PLC
     1. PLC là gì?

PLC là từ viết tắt của Programmable Logic Controller (Bộ điều khiển lập trình được). Đây là một loại máy tính được thiết kế để tự động hóa các quy trình công nghiệp cụ thể, chức năng máy hoặc cả một dây chuyền sản xuất. PLC nhận thông tin từ các cảm biến hoặc thiết bị đầu vào được kết nối, xử lý dữ liệu và kích hoạt đầu ra dựa trên các tham số được lập trình sẵn.

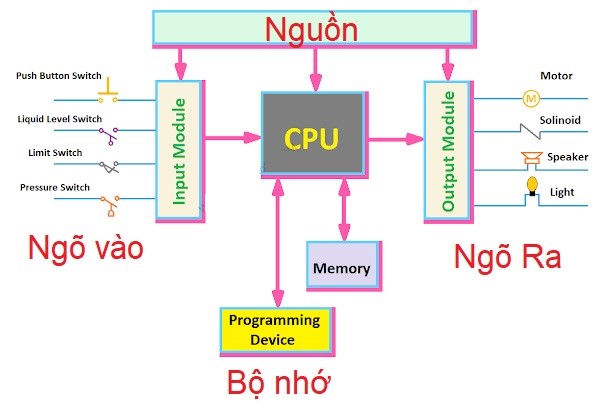


Hình 3.1. Các loại PLC

* + 1. Cấu tạo và phân loại PLC

Thông thường, hệ thống PLC có các bộ phận chính sau:

* Bộ nhớ chương trình: RAM, ROM, ngoài ra có thể sử dụng vùng nhớ ngoài – EPROM.
* Bộ xử lý trung tâm CPU.
* Module input/output. Thông thường module I/O được tích hợp trên PLC, khi có nhu cầu mở rộng I/O có thể lắp module I/O.



Hình 3.2. Cấu trúc PLC

Ngoài ra, PLC còn có các bộ phận khác:

* Cổng kết nối PLC và máy tính: RS232, RS422, RS485 thực hiện đổ chương trình và giám sát chương trình.
* Cổng truyền thông: PLC thường tích hợp cổng truyền thông Modbus RTU. Tùy hãng và dòng sản phẩm, PLC có thể được tích hợp thêm các chuẩn truyền thông khác như Profibus, Profinet, CANopen, EtherCAT…
  + 1. PLC hoạt động như thế nào?

Một số tính năng chính của PLC bao gồm:

* I/O: Bộ xử lý trung tâm (CPU) của PLC lưu trữ và xử lý dữ liệu chương trình, nhưng các mô-đun đầu vào/đầu ra kết nối PLC với phần còn lại của máy. Các mô-đun này cung cấp thông tin cho CPU và kích hoạt kết quả cụ thể. I/O có thể là tương tự hoặc kỹ thuật số; các thiết bị đầu vào có thể bao gồm cảm biến, công tắc và đồng hồ đo, trong khi đầu ra có thể bao gồm rơ-le, đèn, van và ổ đĩa. Người dùng có thể phối hợp I/O của PLC để có cấu hình phù hợp với ứng dụng của họ.
* Giao tiếp: Ngoài các thiết bị đầu vào và đầu ra, PLC cũng có thể cần kết nối với các loại hệ thống khác; ví dụ, người dùng có thể muốn xuất dữ liệu ứng dụng được ghi lại bởi PLC đến một hệ thống giám sát và thu thập dữ liệu (SCADA), theo dõi nhiều thiết bị được kết nối. PLC cung cấp nhiều cổng và giao thức truyền thông để đảm bảo rằng PLC có thể giao tiếp với các hệ thống khác.

HMI: Để tương tác với PLC theo thời gian thực, người dùng cần một HMI (Giao diện Người - Máy). Các giao diện điều khiển này có thể là màn hình hiển thị đơn giản, với màn hình đọc văn bản và bàn phím hoặc các bảng điều khiển cảm ứng lớn giống như điện tử tiêu dùng, nhưng dù sao đi nữa, chúng cho phép người dùng xem xét và nhập thông tin vào PLC theo thời gian thực.

* + 1. Ưu và nhược điểm của PLC

Ưu điểm: Bộ điều khiển PLC chống nhiễu tốt, đáng tin cậy trong môi trường công nghiệp. Đáp ứng các giải thuật phức tạp, độ chính xác cao. Thiết kế gọn nhẹ, lắp đặt dễ dàng. Thay thế hoàn toàn mạch điều khiển relay thông thường, dễ dàng đáp ứng mọi yêu cầu điều khiển. Hỗ trợ các chuẩn mạng truyền thông công nghiệp, tạo sự kết nối và trao đổi dữ liệu giữa các thiết bị trong và ngoài nhà máy đáp ứng tiêu chuẩn công nghiệp 4.0.

Nhược điểm: Giá thành cao: Chi phí sản phẩm cao hơn so với chi phí mạch relay thông thường. Tuy nhiên, hiện nay thị trường VN đã có mặt rất nhiều hãng PLC của Đức, Nhật Bản, Mỹ, Trung Quốc… dẫn đến giá thành cạnh tranh hơn so với trước. Chi phí phần mềm lập trình: Chi phí mua licence phần mềm lập trình tùy thuộc vào hãng sản xuất. Hiện nay có 2 dạng: hãng sản xuất cho phép sử dụng miễn phí và hãng sản xuất yêu cầu mua bản quyền. Yêu cầu người sử dụng có kiến thức về lập trình PLC: Để thiết bị PLC đáp ứng tốt trong điều khiển, người sử dụng cần có kiến thức căn bản về lập trình PLC.

* + 1. PLC được lập trình như thế nào

Chương trình PLC thường được viết trên máy tính và sau đó được tải xuống bộ điều khiển. Phần mềm lập trình PLC phổ biến cung cấp lập trình bằng Ladder Logic hoặc "C". Ladder Logic là ngôn ngữ lập trình truyền thống cho PLC và nó mô phỏng sơ đồ mạch với các "rung" logic được đọc từ trái sang phải. Mỗi rung đại diện cho một hành động cụ thể được điều khiển bởi PLC, bắt đầu với một đầu vào hoặc chuỗi đầu vào (liên hệ) dẫn đến một đầu ra (cuộn). Do tính chất trực quan của nó, Ladder Logic có thể dễ dàng triển khai hơn nhiều ngôn ngữ lập trình khác. Lập trình "C" là một sự đổi mới gần đây hơn cho PLC. Một số nhà sản xuất PLC cũng cung cấp phần mềm lập trình điều khiển.

Một số cách thực điều khiển chính của PLC là điều khiển logic, điều khiển đáp ứng, mạng truyền thông.

Điểu khiển logic:

* Điểu khiển tự động, bán tự động quy trình máy.
* Hộ trợ bộ đếm (Counter) và bộ định thời gian (Timer).

Điều khiển đáp ứng:

* Giải thuật điểu khiển PID, Logic
* Điểu khiển biến tần
* Điểu khiển Servo, động cơ bước
* Điểu khiển biến tần
* Điểu khiển đáp ứng nhiệt độ, áp suất, lưu lượng,...

Mạng truyển thông:

* Kết nối nhiều bộ điều khiển PLC.
* Kết nối bộ điểu khiển PLC với SCADA
  + 1. Ứng dụng của PLC

Trong ngành công nghiệp hiện đại, bộ lập trình PLC đóng vai trò quan trọng, đặc biệt là công nghiệp sản xuất. Với nhiều ưu điểm vượt trội về giá thành lẫn hiệu suất, **PLC**được ứng dụng rộng rãi, phổ biến trong các **tủ bảng điện tự động hóa** của các hệ thống như: cấp nước, nhà máy xử lý nước thải, sản xuất chế biến, đóng gói, giám sát dây chuyền sản xuất, công nghiệp nặng...

Cụ thể PLC là dùng trong công nghệ điều khiển cánh tay Robot để gắp phôi từ băng tải bỏ qua bàn gia công của máy CNC, hay điều khiển Robot đưa đổ vật liệu vào băng tải, hoặc thực hiện các việc đóng hộp, dán tem nhãn, hệ thống báo động…

Ngoài ra, Bộ lập trình PLC còn được dùng trong các ứng dụng giám sát quá trình trong nhà máy mạ, các dây chuyền lắp ráp linh kiện điện tử trong các nhà máy, dây chuyền kiểm tra chất lượng sản phẩm… bằng các công tắc hành trình hoặc các cảm biến.

* 1. Giao thức RFC1006 (ISO on TCP)

Là giao thức kết hợp hai phương thức truyền dữ liệu Ethernet riêng biệt:

* **Định dạng dữ liệu theo định hướng gói tập tin:** Điểu này để cập đến giao thức truyền tải ISO, phân chia dữ liệu thành các gói và đảm bảo việc phân phối đáng tin cậy
* **Định tuyến địa chỉ dễ dàng:** Đây là điểm mạnh của TCP/IP, cung cấp một hệ thống địa chỉ được thiết lập tốt để định tuyến dữ liệu qua mạng.

Lợi thế:

* Đảm bảo dữ liệu truyền đi chính xác và theo thứ tự nhất định
* Đơn giản hóa địa chỉ giúp dễ dàng định vị các thiết bị thông qua mạng

ISO-on-TCP tận dụng hai giao thức cốt lõi:

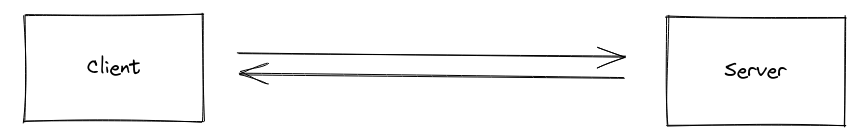
* Giao thức truyền tải ISO (ISO/IEC 8073) - Xác định việc trao đổi dữ liệu đáng tin cậy bằng tính năng kiểm tra và sắp xếp lỗi.
* TCP (Giao thức điều khiển truyền) - Cung cấp dịch vụ hướng kết nối trong bộ TCP/IP, đảm bảo phân phối dữ liệu đáng tin cậy giữa các ứng dụng.

Mặc dù ban đầu được thiết kế để sử dụng rộng rãi hơn, ISO-on-TCP đã tìm thấy ứng dụng cụ thể trong bộ điều khiển logic khả trình (PLC) SIMATIC S7 của Siemens. Các PLC này có thể tận dụng ISO-on-TCP để liên lạc, cho phép chúng trao đổi dữ liệu với các thiết bị hoặc chương trình khác qua mạng TCP/IP.

Trong đồ án này, sinh viên sử dụng thư viện nodes7 có hỗ trợ giao thức này để kết nối và giao tiếp với PLC S71200

* 1. Giao thức SocketIO
     1. Khái niệm

**SocketIO** là một giao thức truyền thông được xây dựng dựa trên HTTP và WebSocket, cung cấp các tính năng bổ sung như kết nối lại tự động, thông báo dựa trên sự kiện, v.v. Nó cho phép giao tiếp 2 chiều,có độ trễ thấp theo thời gian thực giữa máy khách(Client:Trình duyệt Web) và máy chủ (Server).

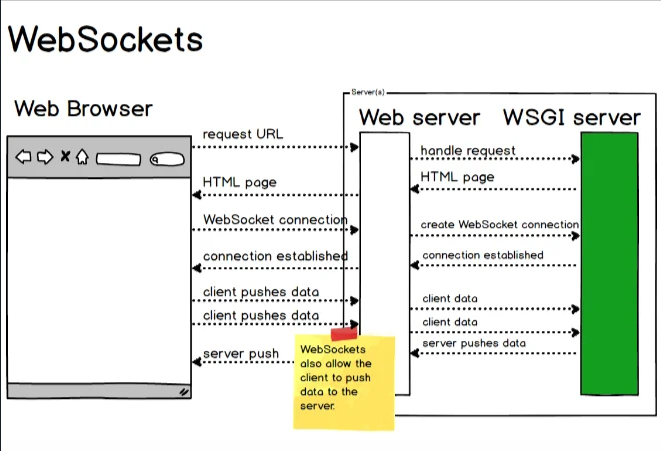


Hình 3.3. Mô hình Client - Server

Socket.IO sử dụng Engine.IO để thiết lập kết nối và trao đổi dữ liệu tạo điều kiện cho việc triển khai máy chủ của khách hàng. Engine.IO được sử dụng để triển khai máy chủ và Engine.IO-client được sử dụng cho máy khách.

Ngoài ra, Socket.IO còn dựa trên nền tảng Engine.IO để thiết lập kết nối cấp thấp hơn bằng cách sử dụng:

* **Websocket**: là giao thức hỗ trợ giao tiếp hai chiều giữa client và server để tạo một kết nối trao đổi dữ liệu. Giao thức này không sử dụng HTTP mà thực hiện nó qua TCP.



Hình 3.4. Giao thức Websocket

* **HTTP long-polling** : Là một phương thức liên lạc phổ biến khác trong đó client mở kết nối với server trong một khoảng thời gian nhất định. Nếu server không có bất kỳ thông tin nào, nó sẽ giữ yêu cầu mở cho đến khi có thông tin hoặc hết thời hạn được chỉ định (hết thời gian chờ)..Nếu WebSockets không được trình duyệt hoặc máy chủ hỗ trợ, Socket.IO sẽ quay trở lại HTTP – long polling.



Hình 3.5. Giao thức HTTP-long polling

Dù được xây dựng dựa trên Engine.IO nhưng SocketIO được bổ sung thêm các đặc tính sau:

* **HTTP long-polling fallback**: Một trong những tính năng nổi bật của Socket.IO là cơ chế dự phòng tuyệt vời của nó. Trong các trường hợp không hỗ trợ kết nối WebSocket, chẳng hạn như trong các trình duyệt cũ hơn hoặc một số môi trường mạng nhất định, Socket.IO tự động trở về sử dụng các kỹ thuật như HTTP long-polling.

Điều này liên quan đến việc thực hiện các yêu cầu HTTP lặp đi lặp lại tới máy chủ, giữ kết nối mở cho đến khi có dữ liệu mới. Việc này đảm bảo rằng chức năng thời gian thực vẫn còn nguyên vẹn ngay cả trong điều kiện kém thuận lợi hơn khi mà trước đây chưa có nhiều trình duyệt hỗ trợ cho WebSocket.

Ngay cả khi hầu hết các trình duyệt hiện nay đều hỗ trợ WebSocket (hơn 97%), đây vẫn là một tính năng tuyệt vời vì chúng tôi vẫn nhận được báo cáo từ người dùng rằng không thể thiết lập kết nối WebSocket vì họ sử dụng một số proxy bị định cấu hình sai.

* **Automatic reconnection**:(Trong trường hợp ngắt kết nối, Socket.IO có thể tự động thiết lập lại kết nối) Mạng vốn không đáng tin cậy và kết nối có thể bị mất vì nhiều lý do - có thể là trục trặc nhất thời hoặc mất kết nối Internet tạm thời. Socket.IO nổi bật lên trong những tình huống như vậy bằng cách cung cấp tính năng tự động kết nối lại.

Trong một số điều kiện cụ thể, kết nối WebSocket giữa server và client có thể bị gián đoạn mà cả hai bên đều không biết về trạng thái liên kết bị hỏng. Khi mất kết nối, Socket.IO sẽ liên tục cố gắng thiết lập lại kết nối, đảm bảo rằng người dùng gặp phải sự gián đoạn tối thiểu.

Đó là lý do tại sao Socket.IO bao gồm cơ chế heartbeat, kiểm tra định kỳ trạng thái kết nối. Và khi client cuối cùng bị ngắt kết nối, nó sẽ tự động kết nối lại với độ trễ chờ tăng dần theo theo cấp số nhân, để không làm server bị quá tải.

* **Package buffering**:(Các gói dữ liệu được đệm trong khi kết nối được thiết lập lại, đảm bảo tin nhắn không bị mất) Socket.IO được trang bị cơ chế tích hợp để lưu vào bộ đệm các tin nhắn. Điều này có nghĩa là ngay cả khi máy khách tạm thời mất kết nối, Các gói được tự động lưu vào bộ đệm khi client bị ngắt kết nối và sẽ được gửi tự động khi kết nối được thiết lập lại.

Tính năng này đảm bảo rằng không có dữ liệu nào bị mất do sự cố mạng tạm thời.Mặc dù hữu ích trong hầu hết các trường hợp (khi độ trễ kết nối lại ngắn), nhưng nó có thể dẫn đến một lượng lớn các sự kiện khi kết nối được khôi phục.

* **Acknowledgements**: (Máy khách có thể nhận được thông báo xác nhận từ máy chủ cho biết việc nhận dữ liệu thành công). Tín hiệu xác nhận là một khía cạnh thiết yếu của giao tiếp thời gian thực. Chúng cho phép máy chủ xác nhận rằng tin nhắn đã được khách hàng nhận hoặc ngược lại.

Socket.IO tạo điều kiện thuận lợi cho việc này bằng cách cho phép các nhà phát triển gắn các hàm gọi lại vào các sự kiện được phát ra. Điều này trao quyền cho các nhà phát triển xây dựng các hệ thống mạnh mẽ trong đó tính toàn vẹn và độ tin cậy của dữ liệu là tối quan trọng.

* **Broadcasting**: (Tin nhắn có thể được gửi đến tất cả các máy khách được kết nối hoặc đến các nhóm cụ thể) Socket.IO giúp dễ dàng truyền phát tin nhắn tới nhiều khách hàng cùng một lúc. Điều này rất có giá trị trong các tình huống mà nhiều người dùng cần được cập nhật theo thời gian thực, chẳng hạn như trong ứng dụng trò chuyện hoặc bảng điều khiển trực tiếp.

Việc phát sóng đảm bảo rằng các thông điệp được gửi đến tất cả các bên liên quan một cách hiệu quả. Từ phía server, bạn có thể gửi tin nhắn / sự kiện đến tất cả các client hoặc một nhóm client được chỉ định

* **Namespaces**: (Điều này cho phép tổ chức các kênh liên lạc ở phía máy chủ). Trong các ứng dụng phức tạp, khi có nhiều kênh liên lạc giữa máy khách và máy chủ Socket.IO hỗ trợ khái niệm ghép kênh. Cho phép tạo nhiều không gian tên độc lập trên một kết nối. Điều này cho phép các nhà phát triển tổ chức và quản lý các loại giao tiếp khác nhau trong một ứng dụng cách liền mạch.

Khái niệm namespace cho phép bạn phân chia logic của ứng dụng chỉ trong 1 kết nối. Điều này hữu ích cho trường hợp bạn phân quyền admin cho những người dùng được uỷ quyền.

* + 1. Lợi thế khi sử dụng SocketIO
* **Giao tiếp thời gian thực**

Socket.IO cho phép giao tiếp thời gian thực giữa client và server, giúp các ứng dụng cập nhật dữ liệu ngay khi có sự thay đổi mà không cần phải tải lại trang.

* **Hỗ trợ nhiều trình duyệt**



Hình 3.6. Những trình duyệt hỗ trợ được thử nghiệm ở Sauce Labs

* **Thiết lập kết nối dễ dàng**
* **Fallback options**

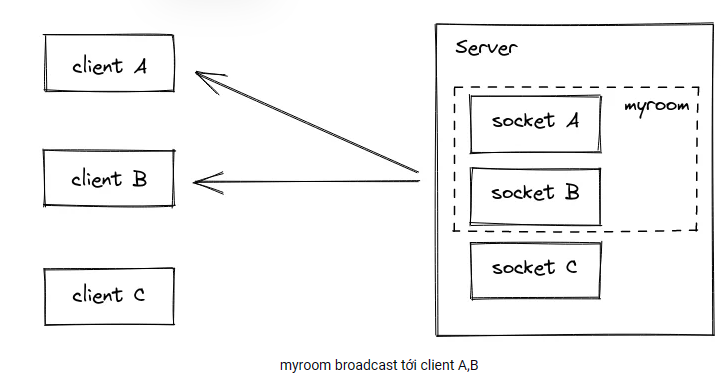
Socket.IO có khả năng tự động chuyển đổi về các phương thức truyền thông thay thế (polling, long-polling) nếu không được hỗ trợ WebSocket, đảm bảo tính ổn định trên môi trường không hỗ trợ WebSocket. Khắc phục được hạn chế của WebSocket.

* **Giao tiếp event-based**

Socket.IO sử dụng mô hình gửi và nhận các sự kiện (event-based communication), giúp tổ chức và xử lý dữ liệu dễ dàng.

* **Room**

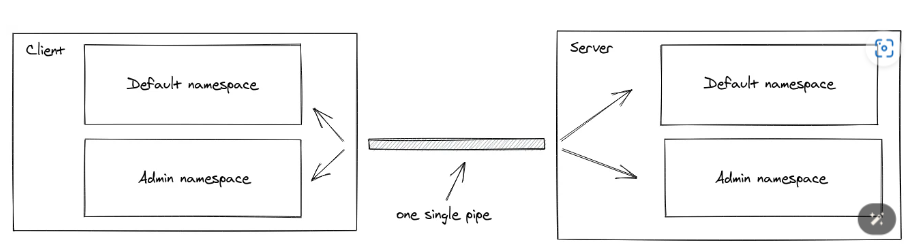
**Room:** là một kênh bất kì mà socket có thể join và leave. Được dùng để phát sóngcác sự kiệntới tập hợp client chỉ định.



Hình 3.7. Room

* **Hỗ trợ đa luồng**

Để phân tách ứng dụng (dựa theo module hay quyền hạn người dùng), Socket.IO cho phép bạn tạo nhiều namespace, những namespace này sẽ hoạt động như các kênh truyền thông riêng biệt nhưng chia sẻ cùng một kết nối cơ bản.

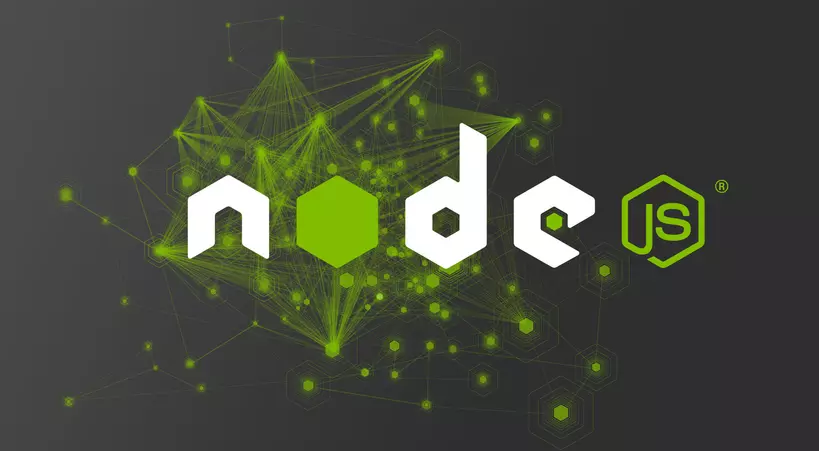


Hình 3.8. Đa luồng

* **Cộng đồng hỗ trợ**

[GitHub - socketio/socket.io: Realtime application framework (Node.JS server)](https://github.com/socketio/socket.io)

* 1. Nền tảng Nodejs

****

Hình 3.9. Nodejs Platform

* + 1. Tổng quan

**Nodejs** là một nền tảng (Platform) phát triển độc lập được xây dựng ở trên Javascript Runtime của Chrome mà chúng ta có thể xây dựng được các ứng dụng mạng một cách nhanh chóng và dễ dàng mở rộng.

Nodejs được xây dựng và phát triển từ năm 2009, bảo trợ bởi công ty Joyent, trụ sở tại California, Hoa Kỳ. Dù sao thì chúng ta cũng nên biết qua một chút chút lịch sử của thứ mà chúng ta đang học một chút chứ nhỉ? =))

Phần Core bên dưới của Nodejs được viết hầu hết bằng C++ nên cho tốc độ xử lý và hiệu năng khá cao.

Nodejs tạo ra được các ứng dụng có tốc độ xử lý nhanh, realtime thời gian thực. Nodejs áp dụng cho các sản phẩm có lượng truy cập lớn, cần mở rộng nhanh, cần đổi mới công nghệ, hoặc tạo ra các dự án Startup nhanh nhất có thể.

* + 1. Ứng dụng viết bằng Nodejs
* **Websocket server**: Các máy chủ web socket như là Online Chat, Game Server…
* **Fast File Upload Client**: là các chương trình upload file tốc độ cao.
* **Ad Server**: Các máy chủ quảng cáo.
* **Cloud Services**: Các dịch vụ đám mây.
* **RESTful API**: đây là những ứng dụng mà được sử dụng cho các ứng dụng khác thông qua API.
* **Any Real-time Data Application**: bất kỳ một ứng dụng nào có yêu cầu về tốc độ thời gian thực. Micro Services: Ý tưởng của micro services là chia nhỏ một ứng dụng lớn thành các dịch vụ nhỏ và kết nối chúng lại với nhau. Nodejs có thể làm tốt điều này.



Hình 3.10. Những công ty lớn sử dụng Nodejs

* 1. Giới thiệu về MySQL Server
     1. SQL là gì?

SQL là ngôn ngữ phi thủ tục, không yêu cầu cách thức truy cập cơ sở dữ liệu như thế nào. Tất cả các thông báo của SQL rất dễ dàng sử dụng và ít mắc lỗi.

SQL cung cấp các tập lệnh phong phú cho các công việc hỏi đáp dữ liệu như: Chèn, xóa và cập nhật các hàng trong 1 quan hệ. Tạp, thêm, xóa và sửa đổi các đối tượng trong của cơ sở dữ liệu. Điều khiển việc truy cập tới cơ sở dữ liệu và các đối tượng của cơ sở dữ liệu để đảm bảo tính bảo mật, tính nhất quán và sự ràng buộc của cơ sở dữ liệu.

Đối tượng của MySQL là các bảng dữ liệu với các cột và các hàng. Cột được gọi là trường dữ liệu và hàng là bản ghi của bảng. Cột dữ liệu và kiểu dữ liệu xác định tạo nên cấu trúc của bảng. Khi bảng được tổ chức thành một hệ thống cho một mục đích sử dụng cụ thể vào công việc nào đó sẽ trở thành một cơ sở dữ liệu.

* + 1. MySQL Server là gì?

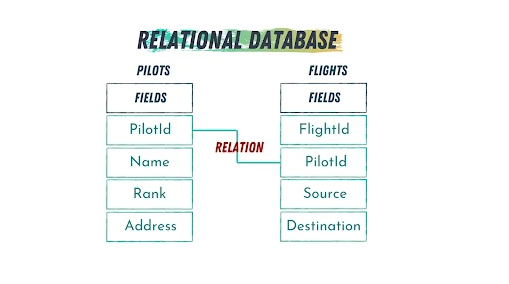
****

Hình 3.10. MySQL Server

MySQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System (RDBMS) ) sử dụng câu lệnh SQL (Transact-SQL)để trao đổi dữ liệu giữa máy Client và máy cài MySQL Server. Một RDBMS bao gồm databases, database engine và các ứng dụng dùng để quản lý dữ liệu và các bộ phận khác nhau trong RDBMS. MySQL được phát triển và tiếp thị bởi Oracle.

* + 1. MySQL hoạt động như thế nào?
       1. Cơ sở dữ liệu quan hệ (relation Database) trong MySQL

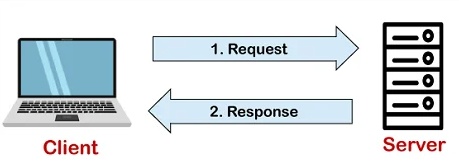
Database là tập hợp dữ liệu theo cùng một cấu trúc. Dữ liệu được đặt trong một bộ dữ liệu chung, dataset, được tổ chức sắp xếp giống như một bảng tính vậy. Mỗi “bảng” này có liên hệ với nhau theo cách nào đó.

****

Hình 3.11. Minh Họa Database

* + - 1. Mô hình Client – Server trong MySQL

MySQL là hệ thống quản lý (management system) được xây dựng theo mô hình **client-server.** Nói một cách dễ hiểu, **server** là nơi lưu trữ dữ liệu và **client** là công cụ mà bạn sẽ cần sử dụng để truy cập và truy xuất dữ liệu

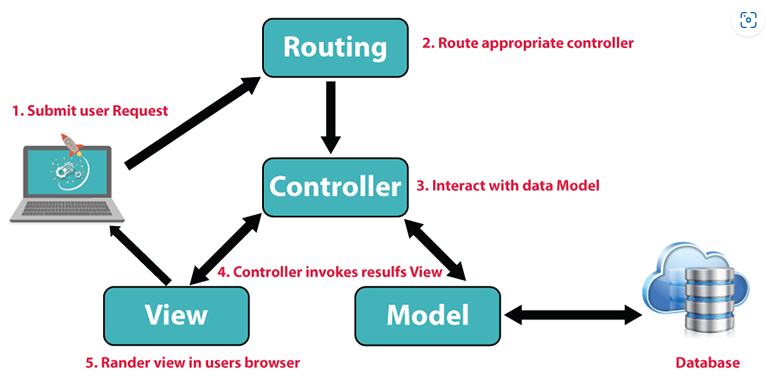
****

Hình 3.12. Mô hình Client-Server

* 1. Tổng quan về mô hình MVC
     1. Khái niệm

Mô hình Model-View-Controller (MVC) là một mẫu kiến ​​trúc phân tách một ứng dụng thành ba thành phần logic chính Model, View và Controller. Do đó viết tắt MVC. Mỗi thành phần kiến ​​trúc được xây dựng để xử lý khía cạnh phát triển cụ thể của một ứng dụng. MVC tách lớp logic nghiệp vụ và lớp hiển thị ra riêng biệt. Ngày nay, kiến ​​trúc MVC đã trở nên phổ biến để thiết kế các ứng dụng web cũng như ứng dụng di động.

* + 1. Cấu trúc mô hình MVC

****

Hình 3.13. Cấu trúc của MCV

Nền tảng MVC bao gồm các thành phần dưới đây:

**Models:** Các đối tượng Models là một phần của ứng dụng, các đối tượng này thiết lập logic của phần dữ liệu của ứng dụng. Thông thường, các đối tượng model lấy và lưu trạng thái của model trong CSDL. Ví dụ như, một đối tượng Product (sản phẩm) sẽ lấy dữ liệu từ CSDL, thao tác trên dữ liệu và sẽ cập nhật dữ liệu trở lại vào bảng Products ở MySQL.

Trong các ứng dụng nhỏ, model thường là chỉ là một khái niệm nhằm phân biệt hơn là được cài đặt thực thụ, ví dụ, nếu ứng dụng chỉ đọc dữ liệu từ CSDL và gửi chúng đến view, ứng dụng không cần phải có tầng model và các lớp liên quan. Trong trường hợp này, dữ liệu được lấy như là một đối tượng model (hơn là tầng model).

**Views:** Views là các thành phần dùng để hiển thị giao diện người dùng (UI). Thông thường, view được tạo dựa vào thông tin dữ liệu model. Ví dụ như, view dùng để cập nhật bảng Products sẽ hiển thị các hộp văn bản, drop-down list, và các checkbox dựa trên trạng thái hiện tại của một đối tượng Product.

**Controllers:** Controller là các thành phần dùng để quản lý tương tác người dùng, làm việc với model và chọn view để hiển thị giao diện người dùng. Trong một ứng dụng MVC, view chỉ được dùng để hiển thị thông tin, controller chịu trách nhiệm quản lý và đáp trả nội dung người dùng nhập và tương tác với người dùng. Ví dụ, controller sẽ quản lý các dữ liệu người dùng gửi lên (query-string values) và gửi các giá trị đó đến model, model sẽ lấy dữ liệu từ CSDL nhờ vào các giá trị này.

Mẫu MVC giúp bạn tạo được các ứng dụng mà chúng phân tách rạch ròi các khía cạnh của ứng dụng (logic về nhập liệu, logic xử lý tác vụ và logic về giao diện). Mẫu MVC chỉ ra mỗi loại logic kế trên nên được thiết lập ở đâu trên ứng dụng. Logic giao diện (UI logic) thuộc về views. Logic nhập liệu (input logic) thuộc về controller. Và logic tác vụ (Business logic – là logic xử lý thông tin, mục đích chính của ứng dụng) thuộc về model. Sự phân chia này giúp bạn giảm bớt được sự phức tạp của ứng dụng và chỉ tập trung vào mỗi khía cạnh cần được cài đặt ở mỗi thời điểm.

* + 1. Luồng đi trong mô hình MVC

Khi người dùng thực hiện thao tác trên ứng dụng hoặc website thì từ máy Client sẽ gửi yêu cầu đến server (máy chủ). Lúc này, controller sẽ tiến hành tiếp nhận và xử lý yêu cầu. Một vài trường hợp cần truy xuất dữ liệu thì controller sẽ kết nối với Model để hỗ trợ database.

Sau khi Controller xử lý xong các yêu cầu thì kết quả sẽ được chuyển về View. Lúc này View sẽ tiến hành tạo các mã HTML để trả về giao hiện của trình duyệt kết quả theo yêu cầu của người dùng.

* + 1. Ưu và nhược điểm của MVC
       1. Ưu điểm
* **Kiểm tra dễ dàng**: Các thành phần độc lập giúp người lập trình dễ kiểm soát và khắc phục các vấn đề, lỗi phát sinh trước khi hoàn thiện sản phẩm đến người dùng.
* **Chức năng control**: Khi kết hợp với các loại ngôn ngữ lập trình thông dụng như CSS<HTML, Javascript thì mô hình MVC là sự hỗ trợ đóng vai trò tối ưu bộ control trên nền tảng ngôn ngữ lập trình.
* **View và size**: MVC giúp tối ưu diện tích băng thông khi sử dụng tránh trường hợp khi nhiều yêu cầu được thực hiện cùng lúc sẽ tạo ra nhiều tệp với dung lượng lớn ảnh hưởng trực tiếp đến đường truyền mạng.
* **Chức năng Soc (Separation of Concern):** Cho phép phân loại các thành Model, View, Database,… để dễ quản lý và kiểm soát hơn.
* **Tính kết hợp:** Người lập trình có thể kết hợp mô hình MVC trên nhiều nền tảng website/ ứng dụng khác nhau giúp tiện lợi hơn khi viết code và giảm tải dung lượng.
* **Kết cấu khá đơn giản:** Phù hợp cho nhiều đối tượng sử dụng khi có nhu cầu lập trình website hoặc các loại ứng dụng.
  + - 1. Nhược điểm

Đối với mô hình MVC có tính phân tách cao giữa các thành phần nên phù hợp để ứng dụng trong các dự án lớn. Nếu ứng dụng MVC trong các dự án nhỏ sẽ dễ gặp tình trạng cồng kềnh, tốn nguồn lực khi phát triển dự án. Đồng thời, thời gian trung chuyển dữ liệu cũng là điều cần cân nhắc khi thực hiện dự án nhỏ.

* + 1. Ứng dụng

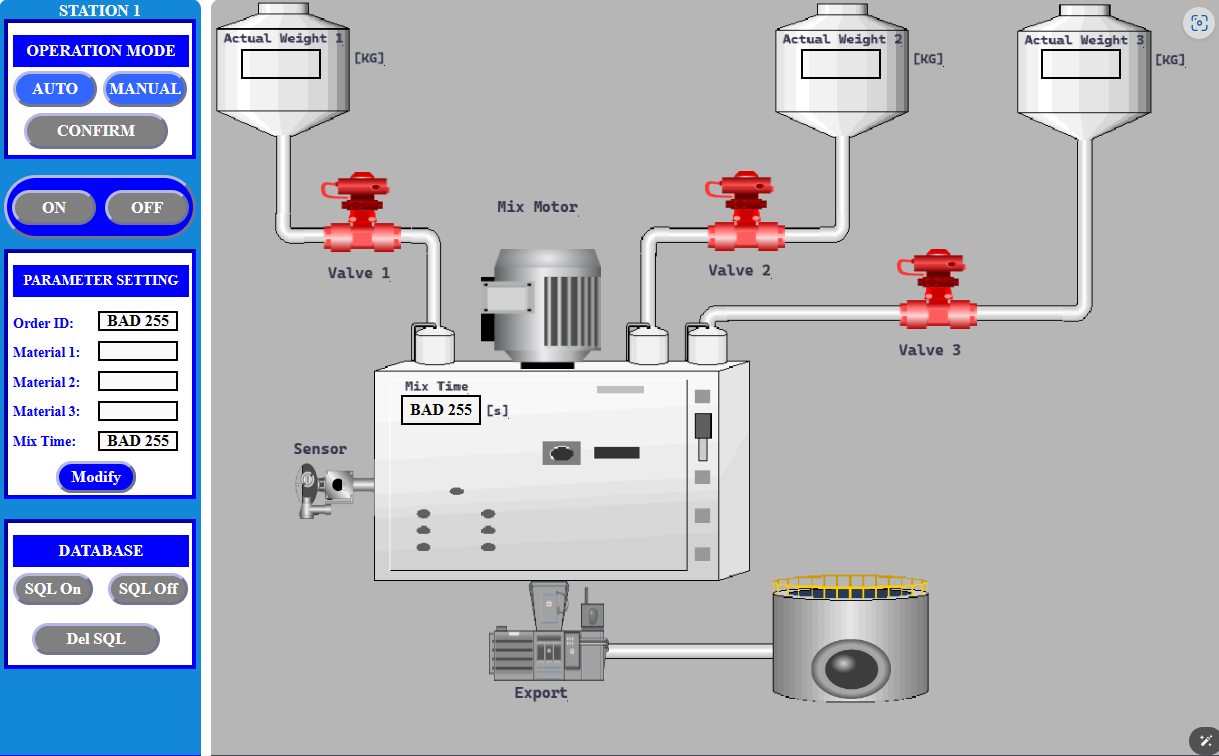
Nó được ứng dụng trong nhiều loại ngôn ngữ lập trình khác nhau, điển hình là PHP MVC và ASP.NET MVC.

Ngoài ra, mô hình MVC còn cho phép phát triển back - end và front - end trên cùng một hệ thống mà không có bất kỳ một sự can thiệp, chỉnh sửa hay chia sẻ tập tin khi một trong hai bên vẫn đang làm việc.

Chương 4. PHÂN TÍCH VÀ TRIỂN KHAI

* 1. Lập trình PLC
     1. Tổng quát hệ thống trộn

Mô hình bao gồm 2 hệ thống trộn riêng biệt đơn giản dùng để khảo sát hoạt động của các thiết bị, quan sát các thông số, trạng thái thay đổi và điều khiển chúng thông qua Internet. Hệ thống mô phỏng lập trình điểu khiển trên 2 con CPU trong phòng TN. Các giá trị được mô phỏng và lập trình bằng TIA Portal.



Hình 4.1. Hệ thống trộn

* + 1. Nguyên lý vận hành hệ thống

Cả 2 trạm trộn:

Hệ thống trộn nguyên liệu lấy từ 3 bể chứa, khối lượng xả xuống từ 3 bể xuống bồn trộn được điều khiển bằng van tương ứng với mỗi bể chứa, người thao tác có thể nhập trọng lượng xả nguyên liệu đối với mỗi mẻ, sau khi xả nguyên liệu xuống bồn trộn thì động cơ sẽ trộn theo thời gian cài đặt, trộn xong thời gian cài đặt thì bắt đầu xả nguyên liệu qua động cơ xả , khi xả xong cảm biến cạn tác động thì động cơ xả dừng và kết thúc 1 mẻ.

Hệ thống còn có thêm chức năng quản lý được dữ liệu được kết nối tới Database

Mọi dữ liệu kể cả thông số cài đặt, trạng thái , khối lượng, thời gian sẽ được lưu vào Database khi bấm nút SQL on; dừng gửi dữ liệu nếu bấm nút SQL off hoặc có thể xóa toàn bộ dữ liệu nếu bấm SQL delete.

**Chế độ tự động:**

* Nhập thông số cài đặt, sau khi nhập xong nhấn nút xác nhận để vận hành
* Hệ thống tiến hành xả nguyên liệu qua 3 van, khi trọng lượng đạt giá trị trọng lượng cài đặt thì tiến hành đóng van đồng thời bắt đầu trộn, động cơ trộn chạy theo thời gian cài đặt, khi thời gian thực tế bằng thời gian cài đặt thì dừng trộn đồng thời động cơ xả chạy.
* Sau khi xả nguyên liệu nếu cảm biến cạn tác động tức là đã xả xong lúc này động cơ xả dừng, kết thúc mẻ trộn, đèn thông báo sản phẩm đã hoàn thành
* Lặp lại từ bước đầu khi người thao tác muốn trộn một mẻ mới.

**Chế độ bằng tay:**

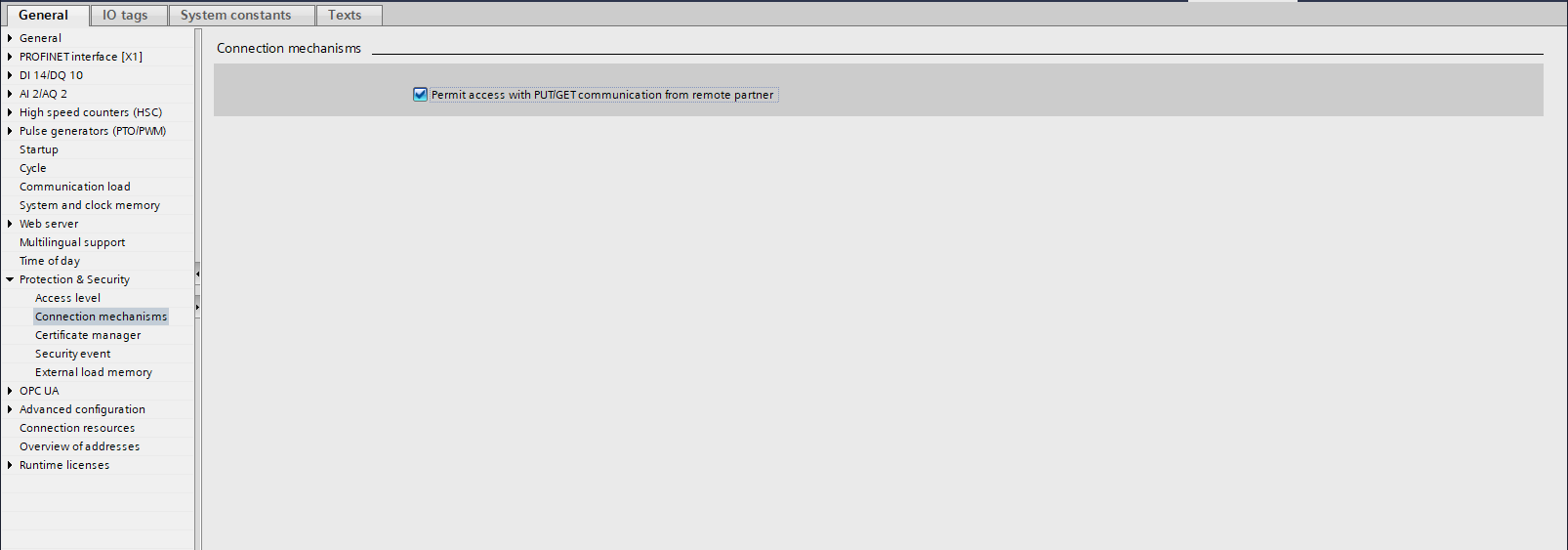
* Các thiết bị trong hệ thống được điều khiển bằng tay qua nút nhấn

**Trường hợp có 1 trong các thiết bị lỗi:** Toàn bộ hệ thống dừng không hoạt động cho đến khi hệ thống được sửa chữa

* + 1. Thiết lập cài đặt cho thiết bị

****

Hình 4.2. Cho phép truy cập dữ liệu trong PLC

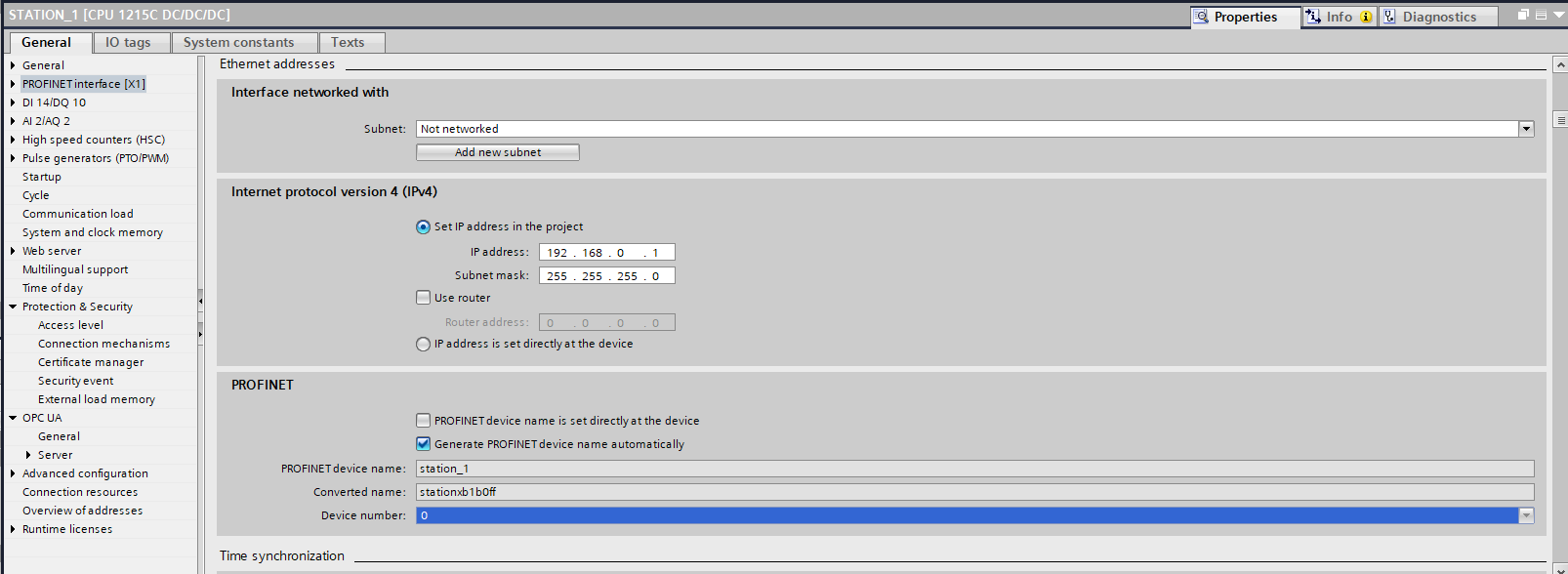


Hình 4.3. Cho phép truy cập PUT/GET

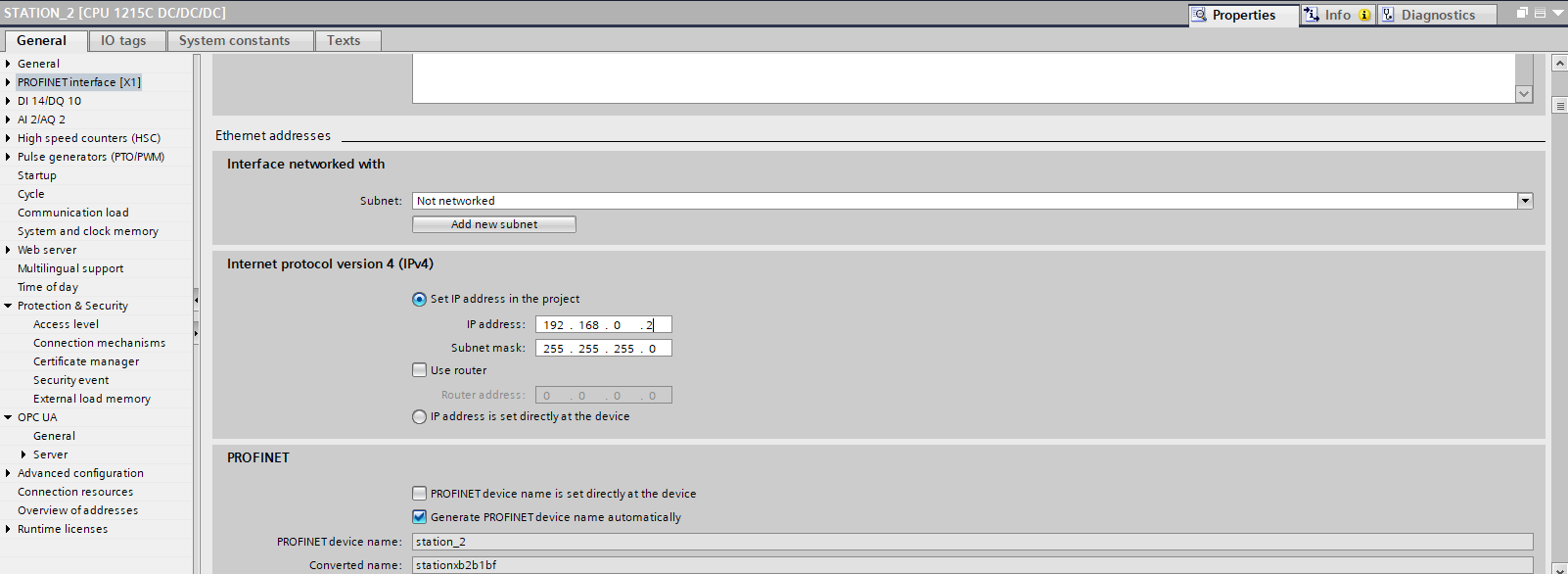
Graphical user interface, application

Description automatically generated

Hình 4.4. System and Clock memory



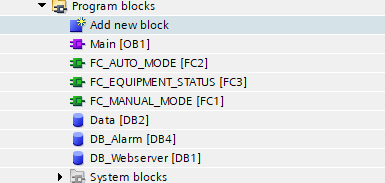
Hình 4.5. IP Address cho PLC1



Hình 4.6. IP Address cho PLC2

* + 1. Chương trình PLC

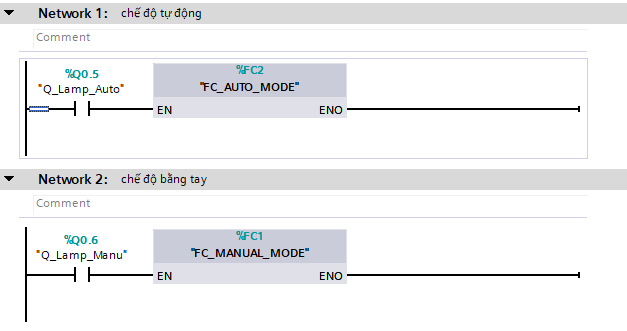
Trong quá trình viết chương trình điều khiển em đã viết ra các khối chương trình như hình bên dưới

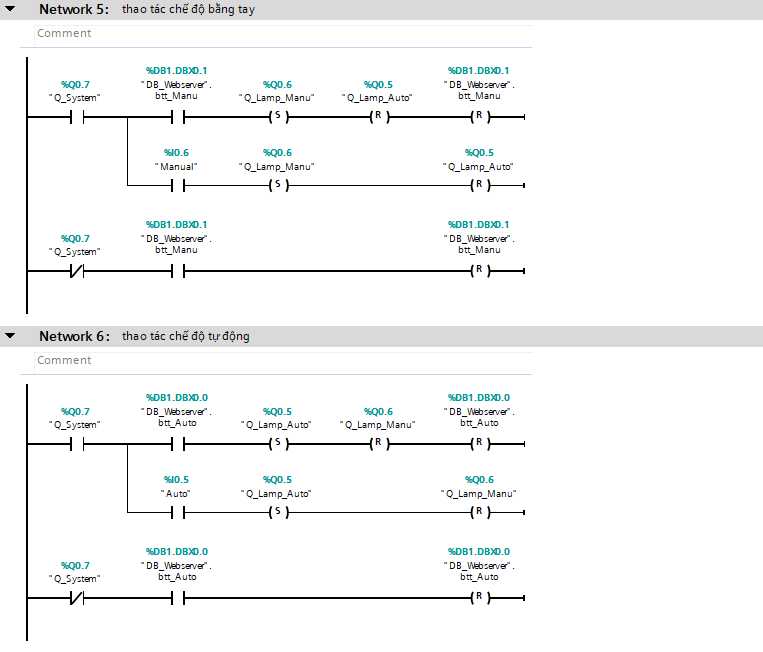


Hình 4.7. Các khối trong chương trình PLC

* + - 1. Khối Main [OB1]

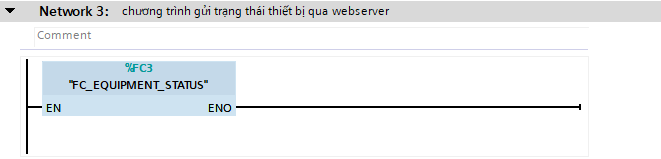
Khối Main nơi chương trình hoạt động xuyên suốt và là trung tâm của việc viết chương trình điều khiển PLC. 2 Networks đầu tiên : chương trình cho phép hoạt động ở 2 chế độ Auto và Manual

****

****

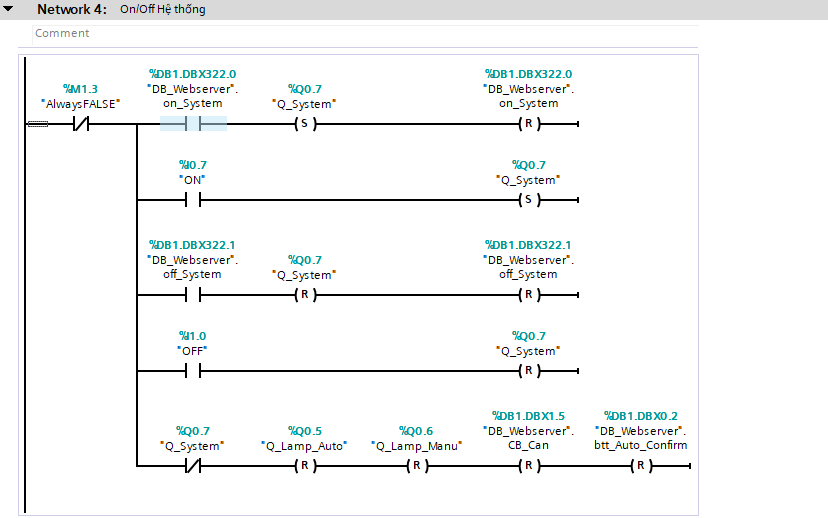
Hình 4.8. 2 Mode hoạt động

Network 3: Gửi trạng thái thiết bị qua webserver



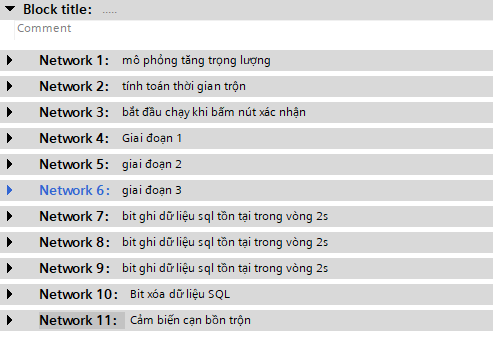
Hình 4.9. Gửi trạng thái thiết bị qua webserver

Network 4: Start/Stop hệ thống



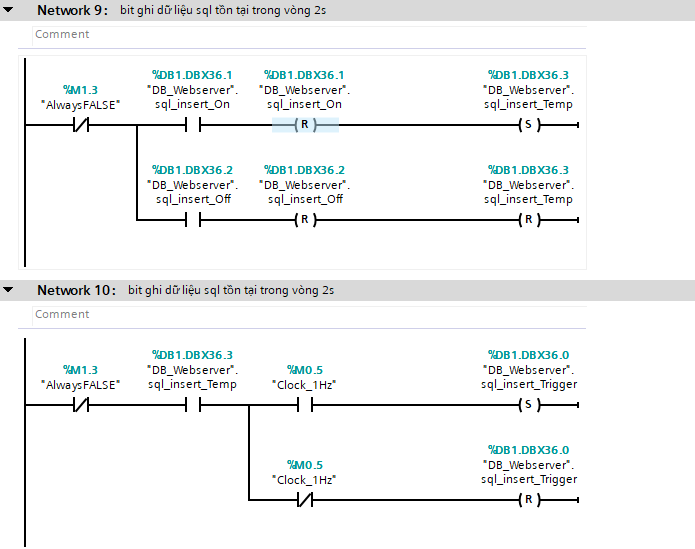
Hình 4.10. Start/Stop hệ thống

* + - 1. Khối chức năng chế độ Auto (FC\_AUTO\_MODE[FC2])



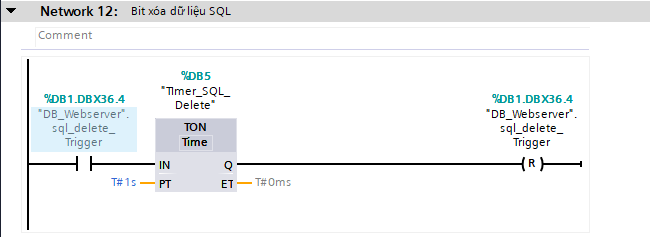
Hình 4.11. Chương trình cho chế độ Auto

Trong đó 2 Network 8,9 có chức năng như một trigger gửi lên webserver kích hoạt chức năng ghi dữ liệu từ PLC lên server rồi ghi vào Database mỗi lần kích hoạt



Hình 4.12. Chương trình cho tính năng gửi dữ liệu lên Database

Network 12: Trigger kích hoạt xóa dữ liệu trên Database



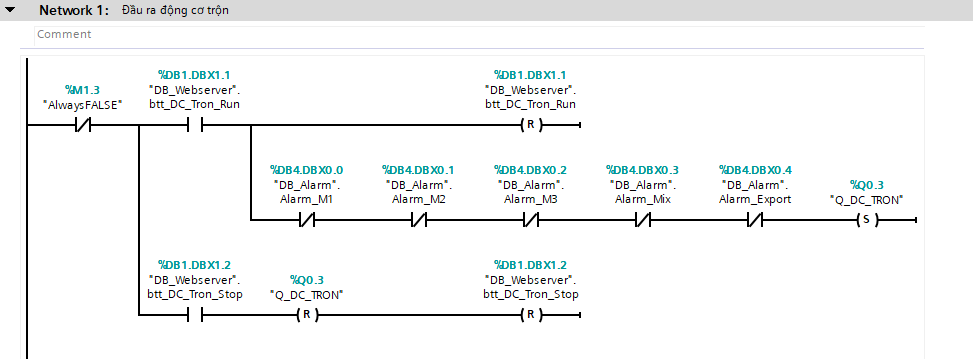
Hình 4.13. Chương trình cho tính năng xóa dữ liệu trên Database

* + - 1. Khối chức năng chế độ Manual (FC\_MANUAL\_MODE[FC1])

****

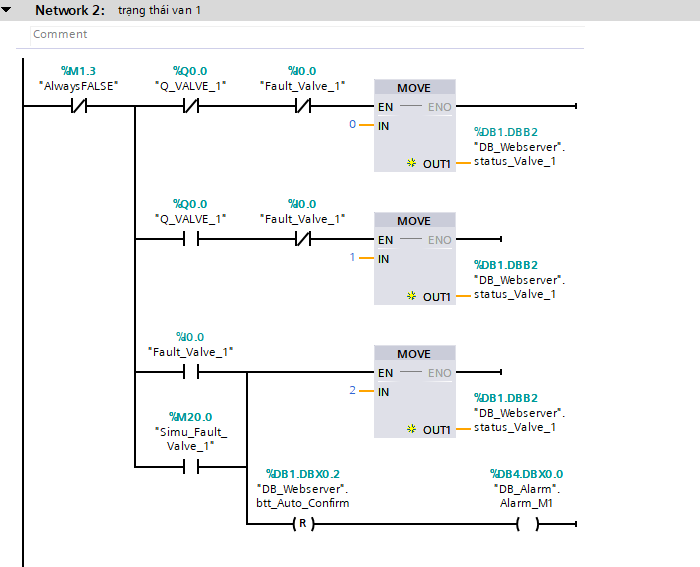
Hình 4.14. Chương trình chế độ Manual

Các thiết bị được điều khiển bằng các 2 nút On/Off tương ứng với mỗi thiết bị



Hình 4.15. Thiết bị được điểu khiển bằng tay

* + 1. Khối trạng thái thiết bị (FC\_EQUIPMENT\_STATUS[FC3])

****

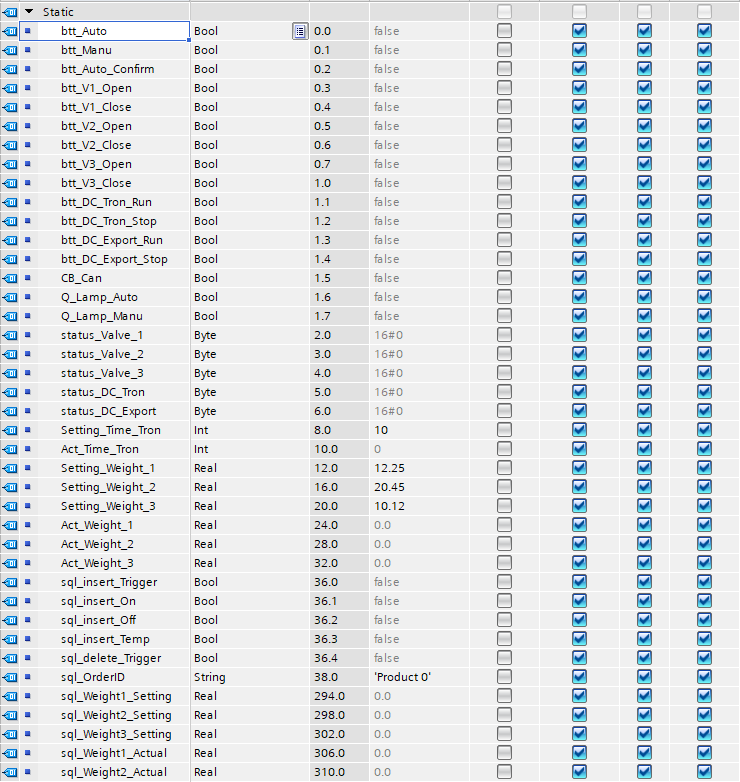
Hình 4.16. Các trạng thái của thiết bị

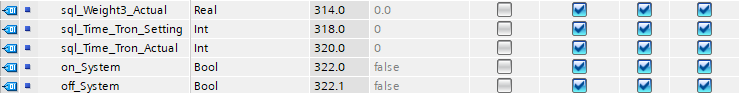
3 trạng thái của đèn tương ứng như sau:

|  |  |
| --- | --- |
| STT | Trạng thái |
| 0 | Dừng |
| 1 | Chạy |
| 2 | Lỗi |

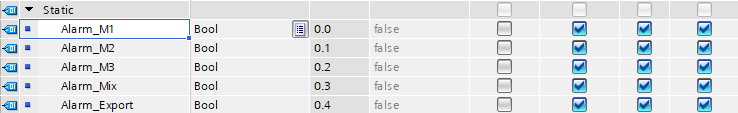
* + 1. Các khối dữ liệu (Data[DB2], DB\_Alarm[DB4], DB\_Webserver[DB6])

Đây là các khối sẽ giao tiếp trực tiếp với Webserver , mọi trạng thái, tín hiệu, dữ liệu sẽ từ đây gửi lên hoặc nhận về từ Server

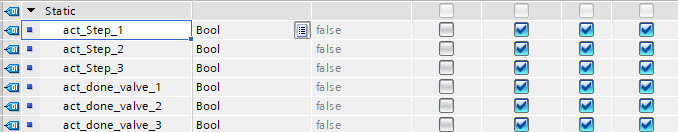




Hình 4.17. Dữ liệu giao tiếp với Webserver



Hình 4.16. Dữ liệu trạng thái Alarm



Hình 4.17. Dữ liệu cho từng giai đoạn của chế độ Auto

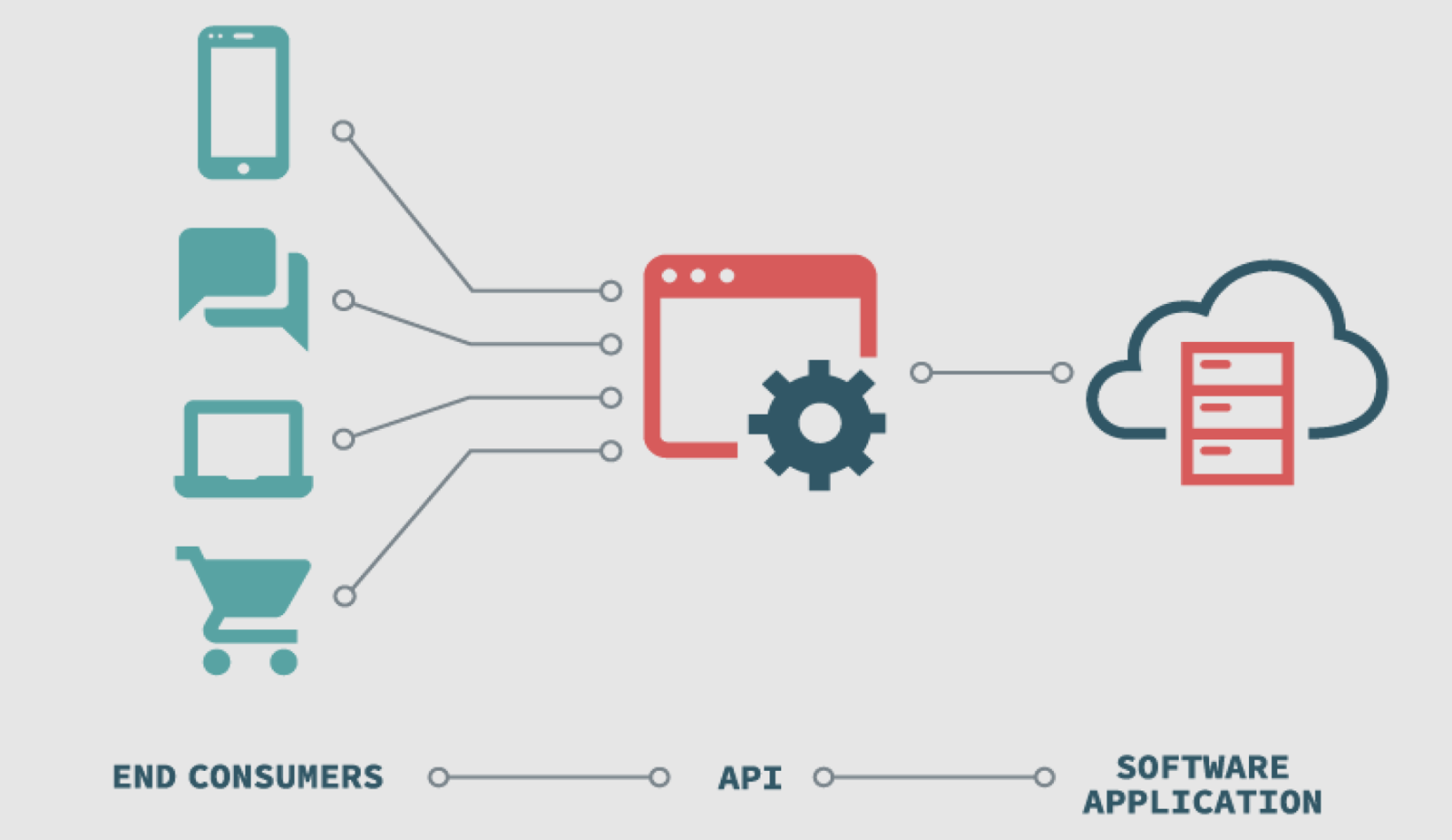
* 1. Lập trình Webserver
     1. Giới thiệu

API là các phương thức, giao thức kết nối với các thư viện và ứng dụng khác. Nó

là viết tắt của Application Programming Interface – giao diện lập trình ứng dụng. API

cung cấp khả năng cung cấp khả năng truy xuất đến một tập các hàm hay dùng. Và từ

đó có thể trao đổi dữ liệu giữa các ứng dụng.

****

Hình 4.18. Sơ đồ cấu trúc giao tiếp API

JSON là viết tắt của JavaScript Object Notation, là một kiểu định dạng dữ liệu tuân theo một quy luật nhất định mà hầu hết các ngôn ngữ lập trình hiện nay đều có thể đọc được. JSON là một tiêu chuẩn mở để trao đổi dữ liệu trên web.

Ví dụ:

{

“name” : “LVTN”,

“year” : “2023”

}

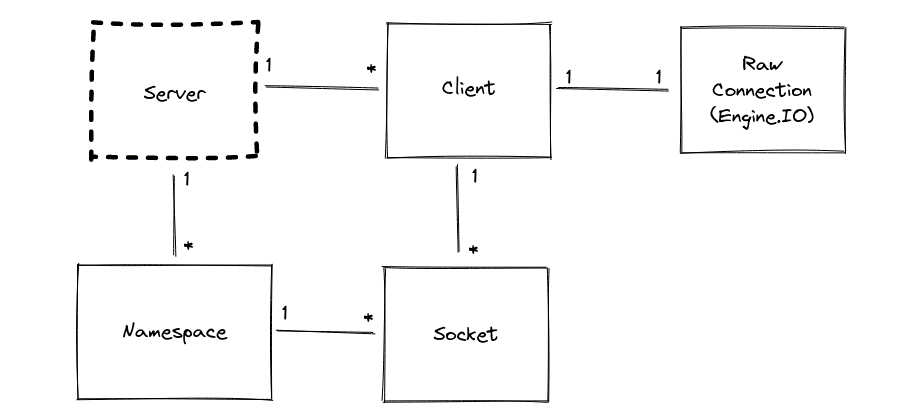
Ta có thể thấy cú pháp của JSON có 2 phần đó là key và value:

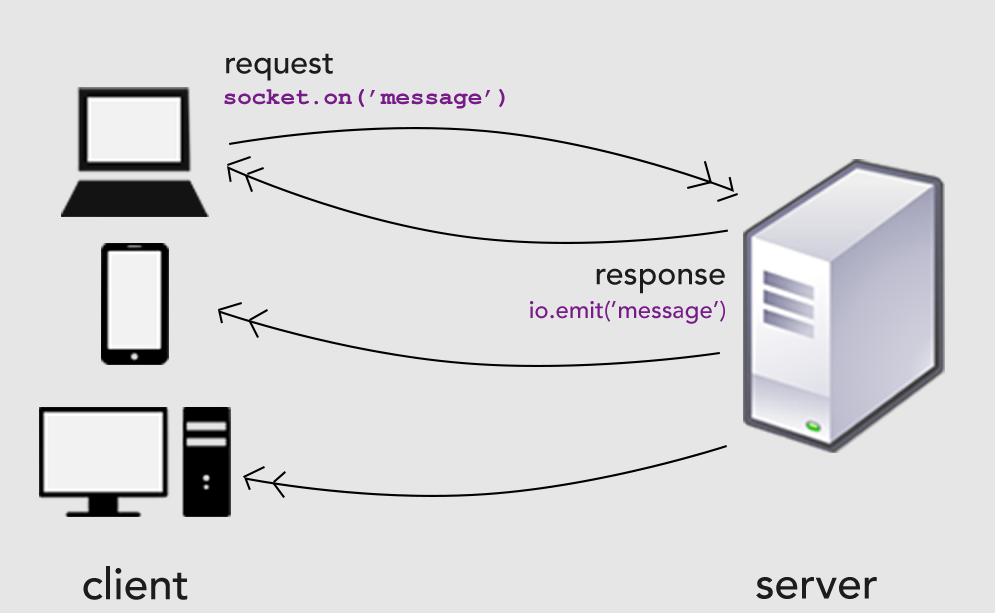
* Chuỗi JSON được bao lại bởi dấu ngoặc nhọn {}
* Các key, value của JSON bắt buộc phải đặt trong dấu nháy kép {“”}, nếu bạn

đặt nó trong dấu nháy đơn thì đây không phải là một chuỗi JSON đúng chuẩn.

SocketIO sử dụng hai API chính là Server API và Client API để cho phép giao tiếp hai chiều giữa server và client. API của SocketIO dựa trên sự kiện và cung cấp nhiều chức năng để thao tác với kết nối, dữ liệu và nhóm.

Phương thức chính giao tiếp dưới dạng bức điện:



****

Hình 4.19. Sơ đồ cách thức giao tiếp giữa Server – Client

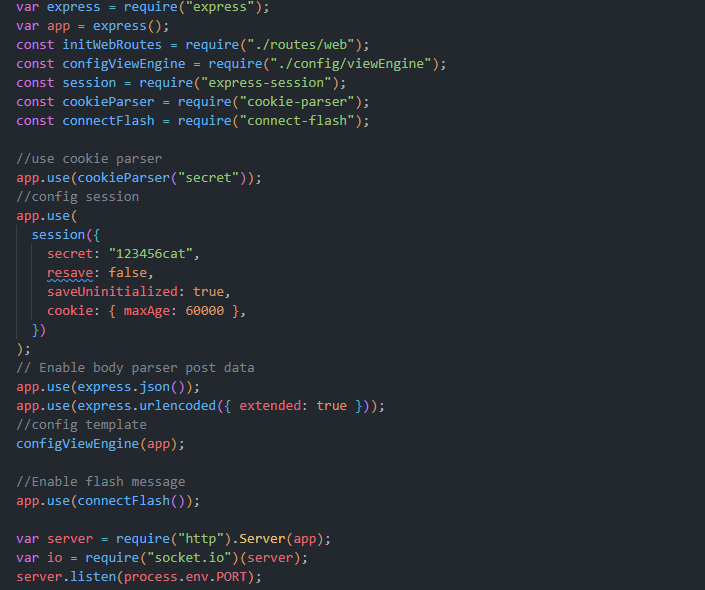
* + 1. Xây dựng Model MVC

****

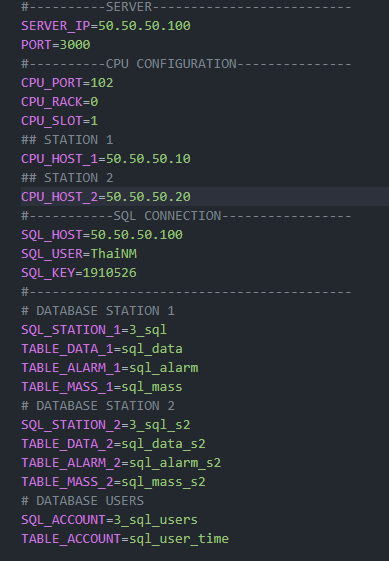
Hình 4.20. Áp dụng Model MVC

Model MVC:

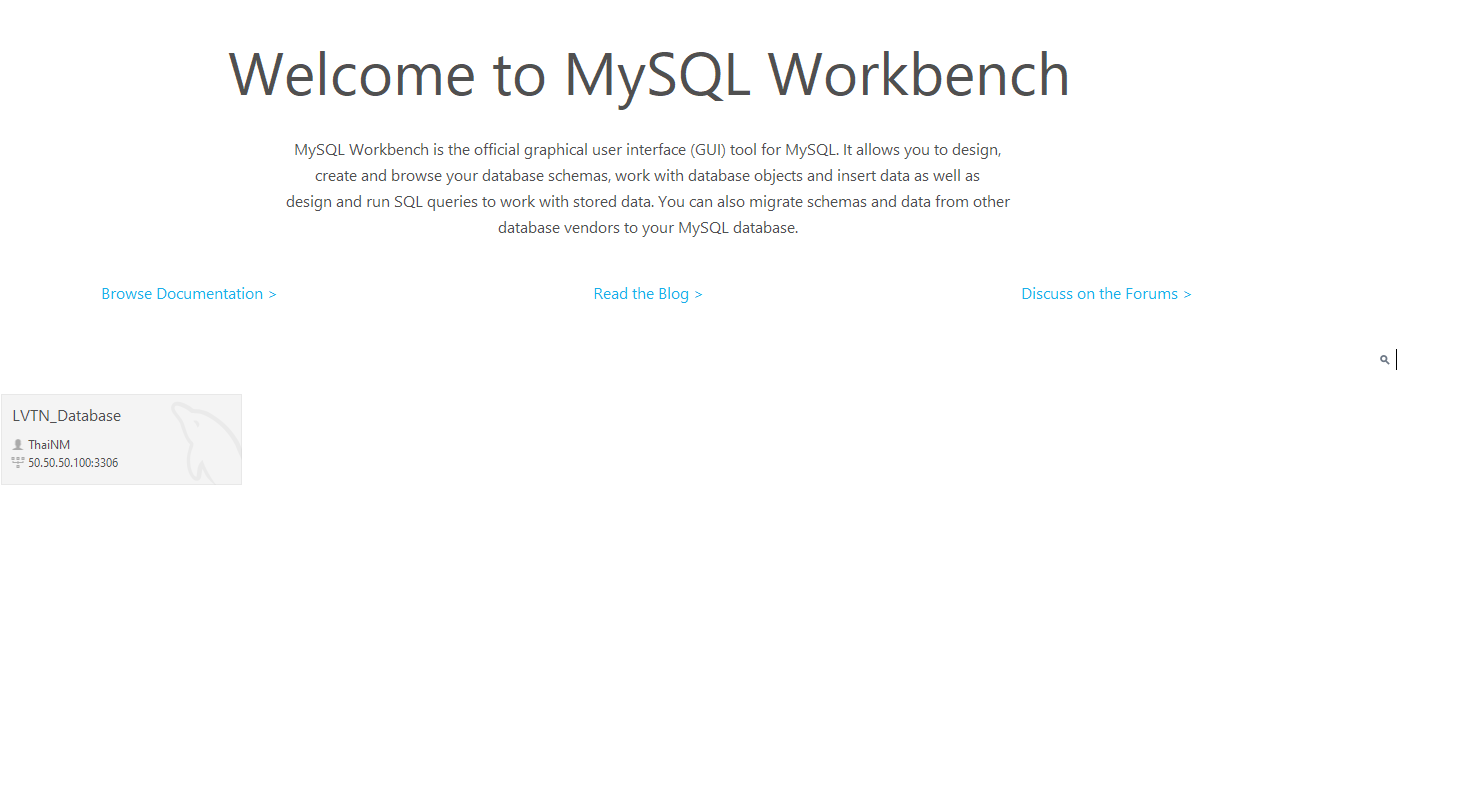
* **Model(Config):** thiết lập logic của phần dữ liệu của ứng dụng
* **Controller, Routes:** các thành phần dùng để quản lý tương tác người dùng, làm việc với model và chọn view để hiển thị giao diện người dùng
* **Views:** các thành phần dùng để hiển thị giao diện người dùng.
  + 1. Xây dựng Server
       1. Thiết lập Server API

****

* + - 1. Thiết lập môi trường kết nối

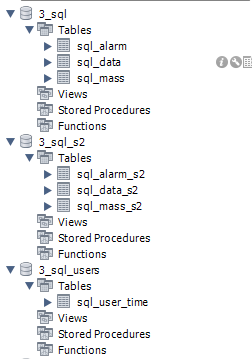
****

* + 1. Xây dựng Database
       1. Tạo Database trong MySQL

****

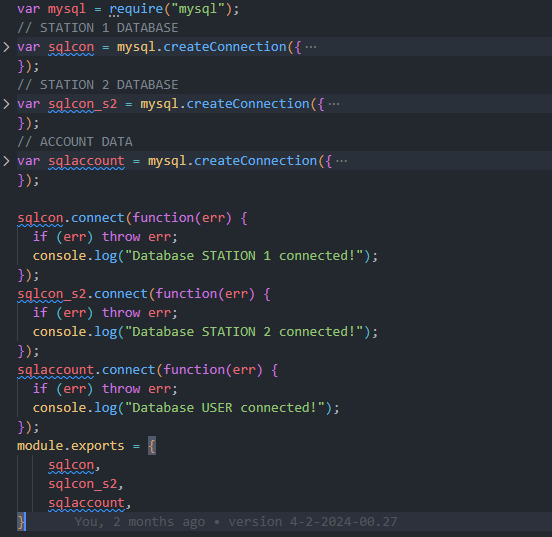
Hình 4.21. LVTN\_Database

* + - 1. Tạo bảng cho Database

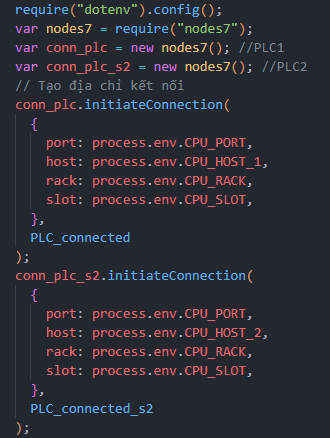
****

Hình 4.22. Bảng dữ liệu tạo trên MySQL

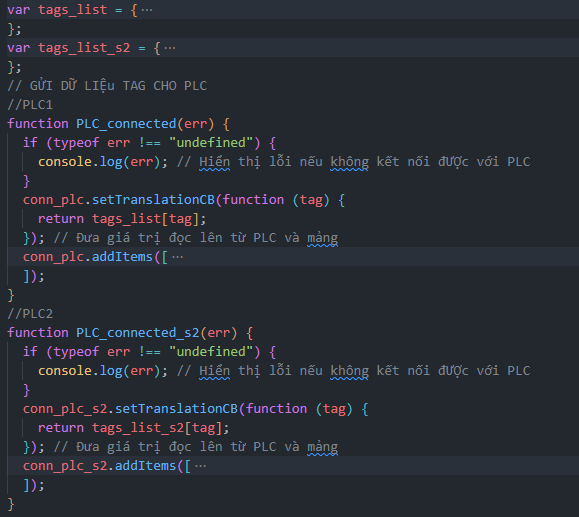
* + - 1. Thiết lập kết nối Server tới MySQL

****

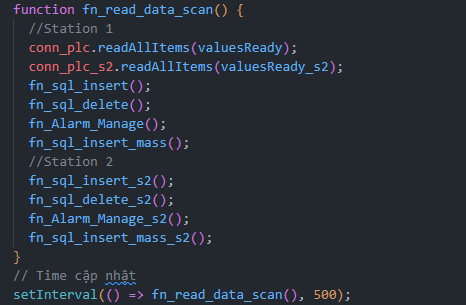
* + 1. Kết nối Server với PLC
       1. Chương trình kết nối

****

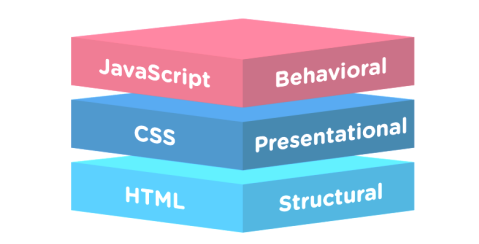
* + - 1. Khai báo tag trên Server tương ứng với kiểu dữ liệu trên PLC



* + - 1. Tạo chức năng scan và cập nhật giá trị theo thời gian

****

* 1. Thiết kế giao diện người dùng(UI)
     1. Giới thiệu
        1. HTML,CSS, Javascript



Hình 4.23. Cấu trúc phân tầng trong lập trình giao diện

**HTML**:

HTML là viết tắt của cụm từ Hypertext Markup Language (tạm dịch là Ngôn ngữ đánh dấu siêu văn bản). HTML được sử dụng để tạo và cấu trúc các thành phần

trong web hoặc ứng dụng, phân chia các đoạn văn, heading, titles, blockquotes…

và HTML không phải là ngôn ngữ lập trình.

HTML bao gồm một loạt các mã ngắn được quản trị viên trang web nhập vào

một tệp văn bản, gọi là các tag. Một HTML document khi hoàn thiện sẽ được lưu

dưới dạng đuôi .html hay .htm. Các files này sau đó sẽ được các trình duyệt web

như Google Chrome, Microsoft Edge đọc hiểu nội dung và chuyển chúng sang

dạng văn bản hiển thị để người dung có thể xem được nó.

Các ưu điểm của HTML:

* Cách sử dụng dễ dàng.
* Dễ dàng tích hợp với nhiều loại ngôn ngữ như PHP, Nodejs, …
* Mã nguồn mở, miễn phí.
* Các markup trong HTML ngắn gọn, tính đồng nhất cao.
* Hoạt động mượt mà trên phần lớn các trình duyệt phổ biến hiện nay

**CSS:**

CSS là chữ viết tắt của Cascading Style Sheets, nó là một ngôn ngữ được sử

dụng để tìm và định dạng lại các phần tử được tạo ra bởi các ngôn ngữ đánh dấu

(HTML). Nói ngắn gọn hơn là ngôn ngữ tạo phong cách cho trang web. Nếu HTML

đóng vai trò định dạng các phần tử trên website như việc tạo ra các đoạn văn bản,

các tiêu đề, bảng, … thì CSS sẽ giúp chúng ta có thể them style vào các phần tử

HTML đó như đổi bố cục, màu sắc trang, đổi màu chữ, font chữ, ...

Phương thức hoạt động của CSS là nó sẽ tìm dựa vào các vùng chọn, vùng chọn

có thể là tên một thẻ HTML, tên một ID, class hay nhiều kiểu khác. Sau đó là nó

sẽ áp dụng các thuộc tính cần thay đổi lên vùng chọn đó. Mối tương quan giữa

HTML và CSS rất mật thiết. HTML là ngôn ngữ markup (nền tảng của site) và

CSS định hình phong cách (tất cả những gì tạo nên giao diện website), chúng không

thể tách rời.

**Javascript:**

Javascript là một ngôn ngữ lập trình kịch bản dựa vào đối tượng phát triển có

sẵn hoặc tự định nghĩa. Nhiệm vụ của Javascript là xử lý những đối tượng HTML

trên trình duyệt. Nó có thể can thiệp với các hành động như thêm, xóa, sửa các

thuộc tính CSS và các thẻ HTML một cách dễ dàng. Hay nói cách khác, Javascript

là một ngôn ngữ lập trình nên trình duyệt ở phía client. Nếu việc xây dựng một

trang web giống như thiết kế một robot, thì HTML chính là khung sườn của robot,

CSS là trang trí hình dạng của robot và Javascript giúp tạo nên các chuyển động,

các hiệu ứng cho robot. Hiện nay, Javascript đang ngày càng trở nên phổ biến trong

thế giới lập trình.

* + 1. Thiết kế giao diện người dung UI
       1. Các chức năng chính

Trong luận văn này, ta đang xây dựng Client với mục đích giám sát nên chức

năng chính của Web bao gồm:

* Kết nối với các server với các loại mã hóa, bảo mật khác nhau.
* Khởi tạo kết nối tới các Client PLC bằng cách gửi thông tin Client tới Server.
* Theo dõi và quản lý các thông tin của Client.
* Giám sát hệ thống được xây dựng trên SIMATIC S7-1500 CPU.
* Vẽ biểu đồ khối lượng.
* Lưu thông tin Alarm cũng như Report.
  + - 1. Chức năng Phân quyền người dùng và điều hướng trang

Để có thể truy cập vào giao diện web, đầu tiên chúng ta cần đăng nhập tài khoản người dùng. Mục đích của việc này là để lưu lại các thông tin về người dùng trong quá trình sử dụng hệ thống. Thông tin người dùng được khai báo và lưu vào trong MySQL Server, từ đó người dùng có thể đăng nhập vào Web bất cứ lúc nào khi đã có tài khoản.

Để khởi tạo các thông tin khi đăng kí hoặc đăng nhập, ta cần tạo một mô hình

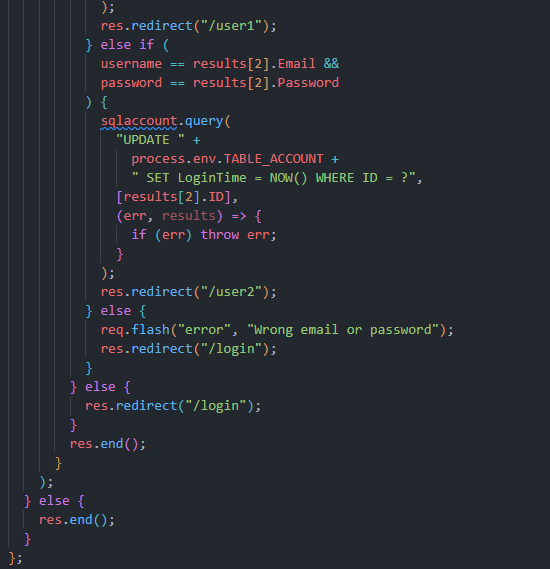
trong ASP.NET CORE MVC, từ đó kết nối các biến trong mô hình này vào MySQL

Server và tạo View trên giao diện người dùng để sử dụng.



**Xử lý đăng nhập – Xác thực người dùng :**

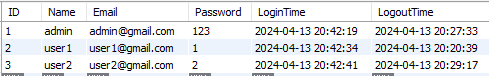
****

****

**Xử lý đăng xuất**



Dữ liệu thông tin người dùng trong Database:



Hình 4.24. Database người dùng

* + - 1. Các chức năng giám sát khác

**Trang quản lý Database dữ liệu khối lượng**

* Bên Server

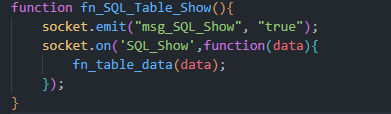


Đọc dữ liệu từ MySQL ghi vô bảng trên trình duyệt



Đọc dữ liệu từ MySQL ghi vô bảng trên trình duyệt

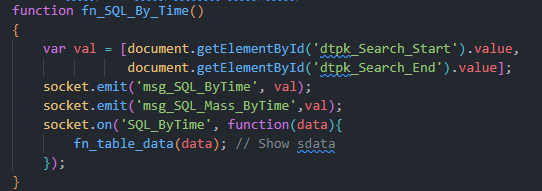
* Bên Browser(Client)



Lắng nghe bức điện từ Server và nhận giá trị để hiển thị lên trình duyệt



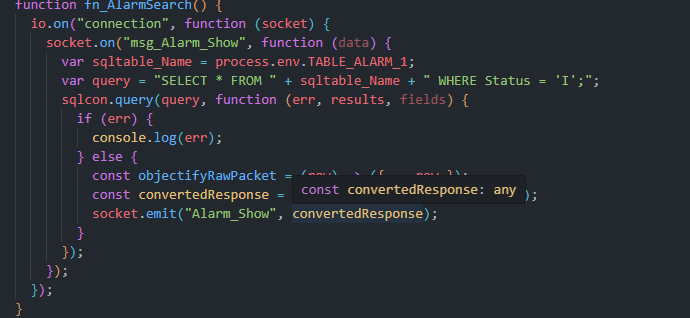
Lắng nghe bức điện từ Server và nhận giá trị để hiển thị lên trình duyệt



Lắng nghe bức điện từ Server và nhận giá trị để hiển thị lên trình duyệt

**Trang Alarm**

* Bên Server

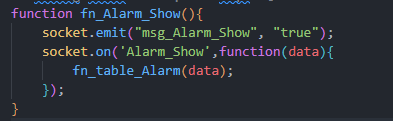
****

Đọc dữ liệu từ MySQL ghi vô bảng trên trình duyệt

****

Đọc dữ liệu từ MySQL ghi vô bảng trên trình duyệt

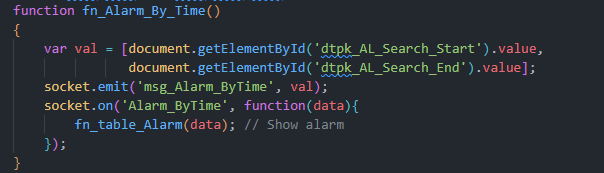
* **Bên Browser(Client)**

****

Lắng nghe bức điện từ Server và nhận giá trị để hiển thị lên trình duyệt

****

Lắng nghe bức điện từ Server và nhận giá trị để hiển thị lên trình duyệt

****

Lắng nghe bức điện từ Server và nhận giá trị để hiển thị lên trình duyệt

**Trang Trend**

Lắng nghe giá trị thu từ Server và vẽ trend lên Client:



Lấy dữ liệu thu từ Server để vẽ trend





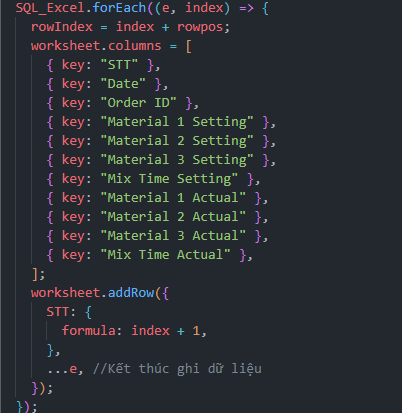
Tạo khoảng thời gian hiển thị trên màn hình

****

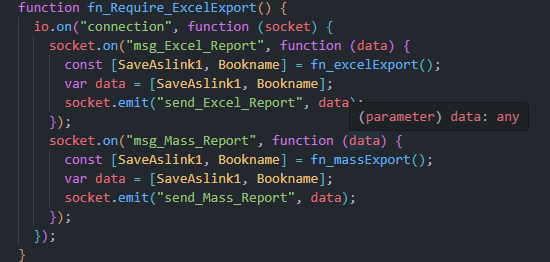
Tải ảnh đồ thị từ Browser

**Chức năng xuất báo cáo**

* **Server:**

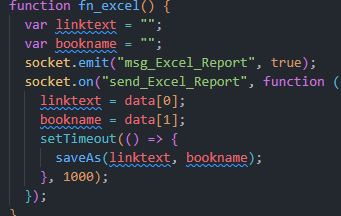
****

Lấy dữ liệu từ Database ghi vào Excel



Bức điện từ Server để tiến hành xuất file report

* **Bên Browser(Client)**



Chức Năng tạo file report

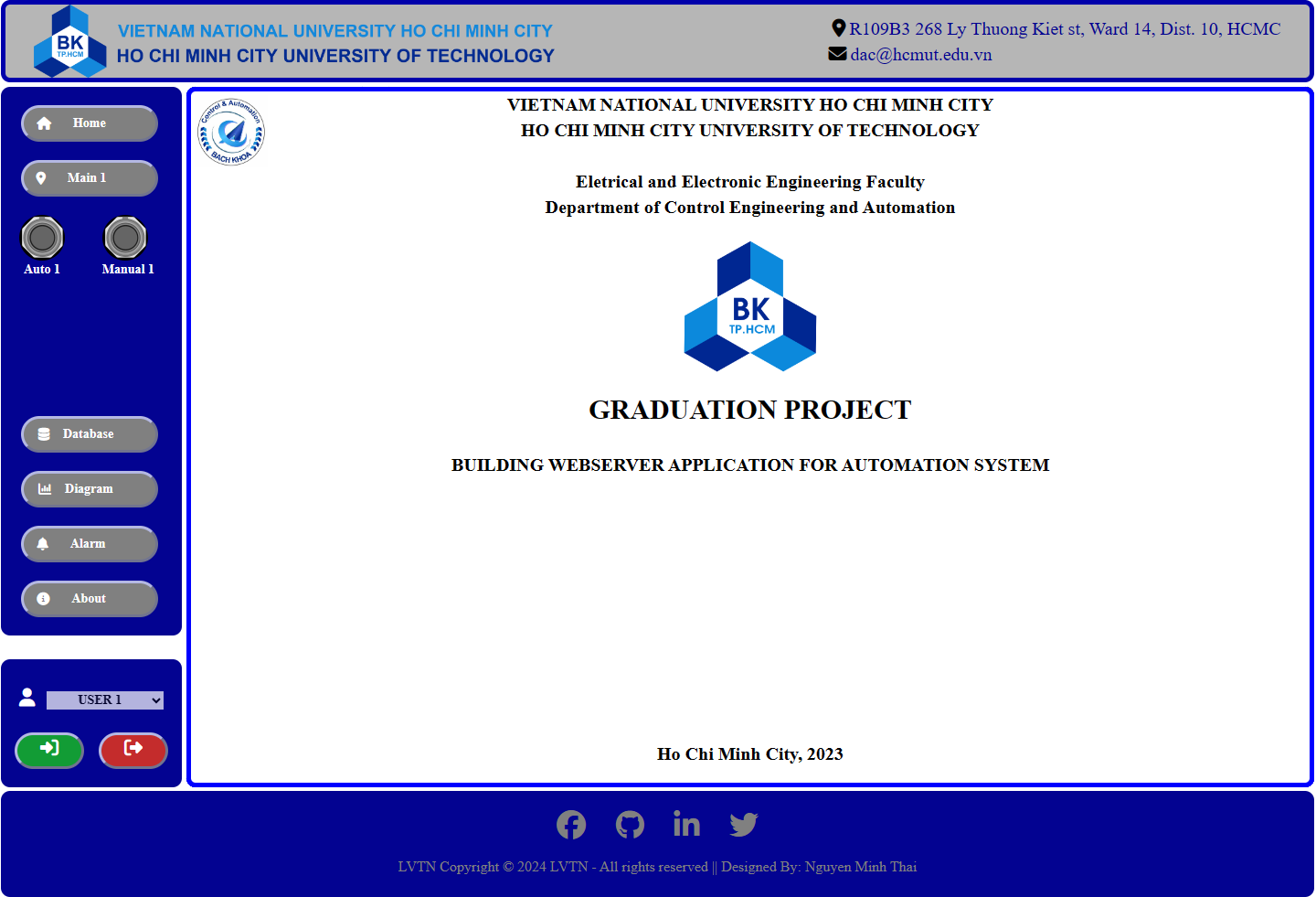
Chương 5. KẾT QUẢ THỰC HIỆN VÀ ĐÁNH GIÁ

* 1. Kết quả thức hiện

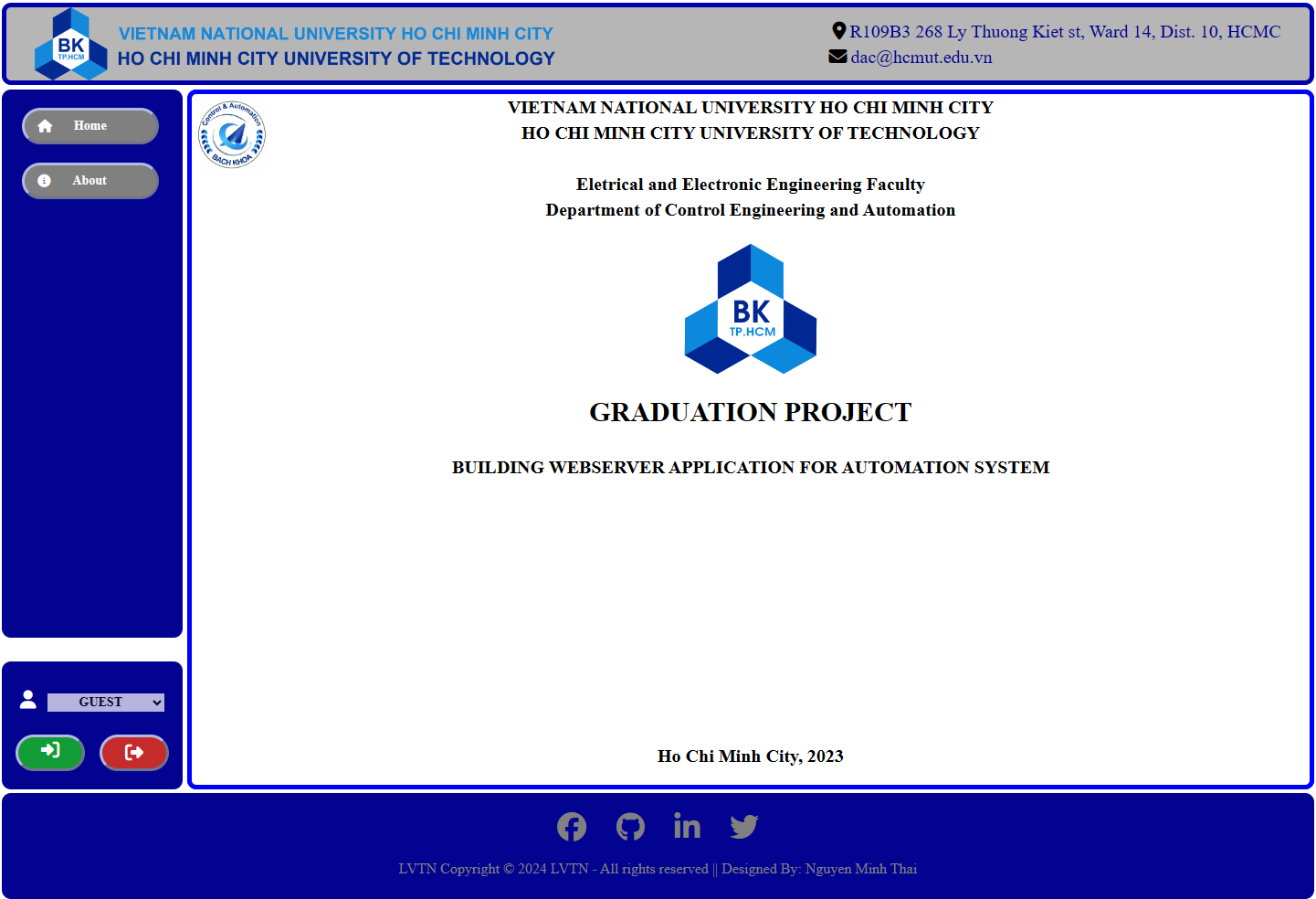
Giao diện SCADA chia làm 3 vùng chính

* Thanh điều hướng: Gồm các nút nhấn để hiển thị các trang thể hiện chức năng SCADA
* Vùng nội dung: điều khiển và giám sát toàn bộ quá trình, chứa các Faceplate của các thiết bị giúp điều khiển độc lập các thiết bị một cách nhanh chóng

Có 6 trang tương ứng: Giới thiệu (Home), Hệ thống trộn (Main), Cơ sở dữ liệu (Database), Đồ thị (Diagram), Thông tin thêm (About)



Hình 5.1 Giao diện chính của SCADA khi đăng nhập vào tài khoản tương ứng

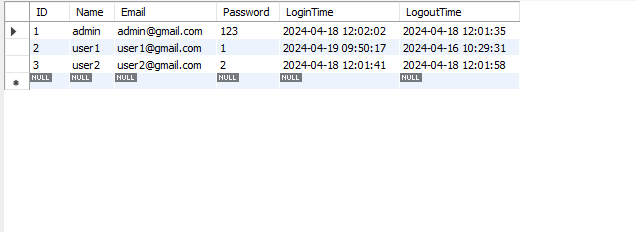


Hình 5.2. Giao diện khi chưa đăng nhập

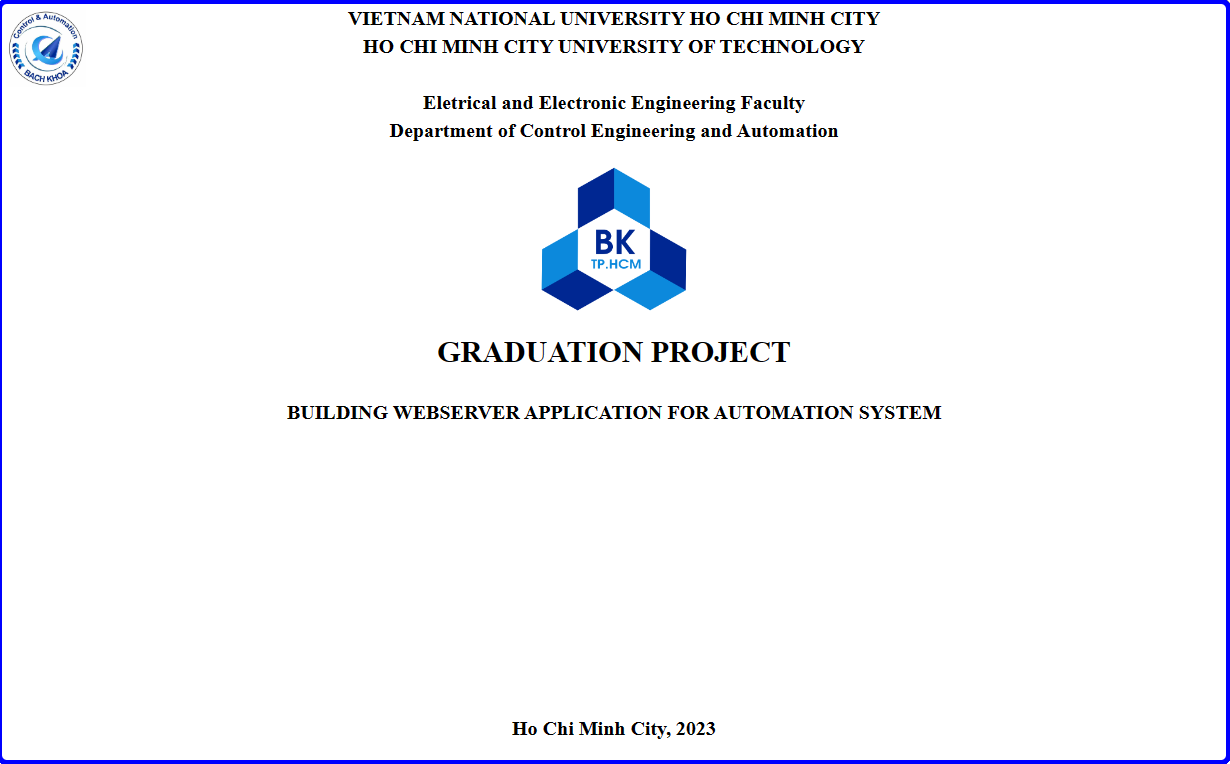
Ngoài ra, SCADA còn có giao diện cho phép người dùng đăng nhập vào trang Web



Hình 5.3. Trang đăng nhập

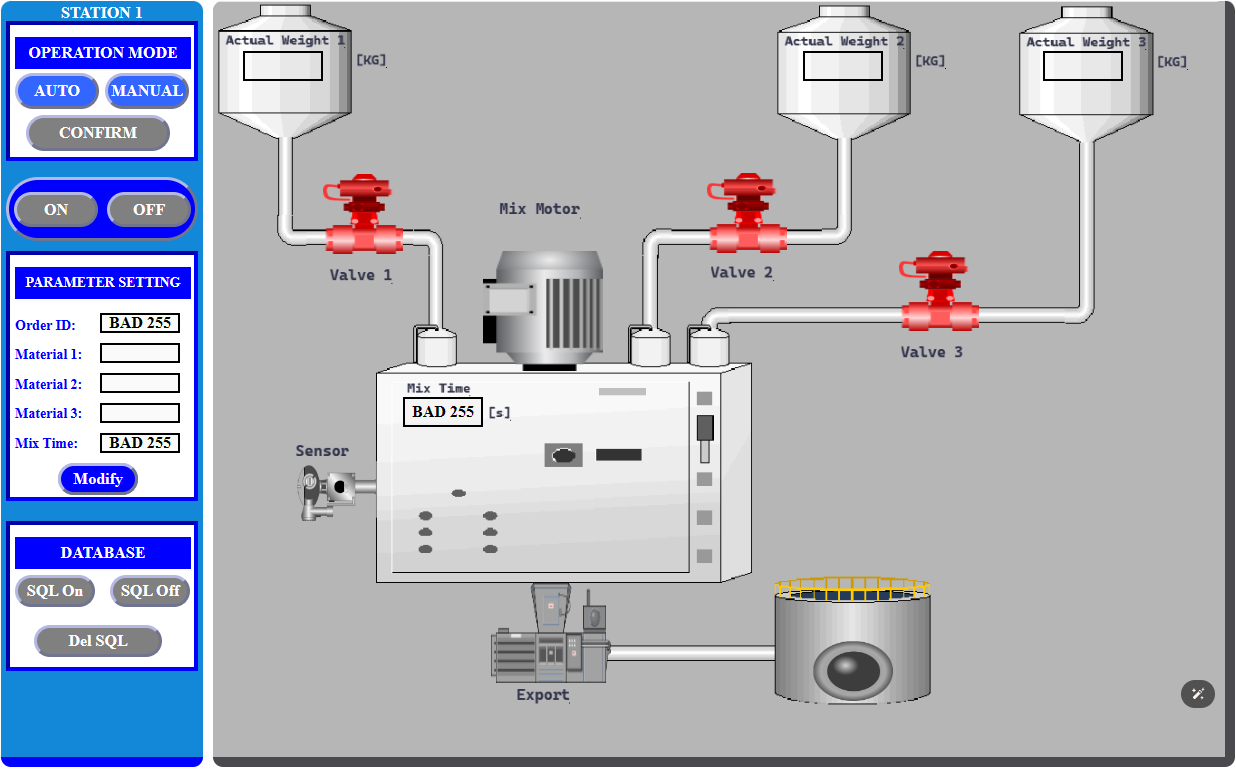


Hình 5.4. Tài khoản đăng nhập và kiểm tra thời điểm đăng nhập

**Trang Home: **

Hình 5.5. Trang Home

**Trang Main:**

****

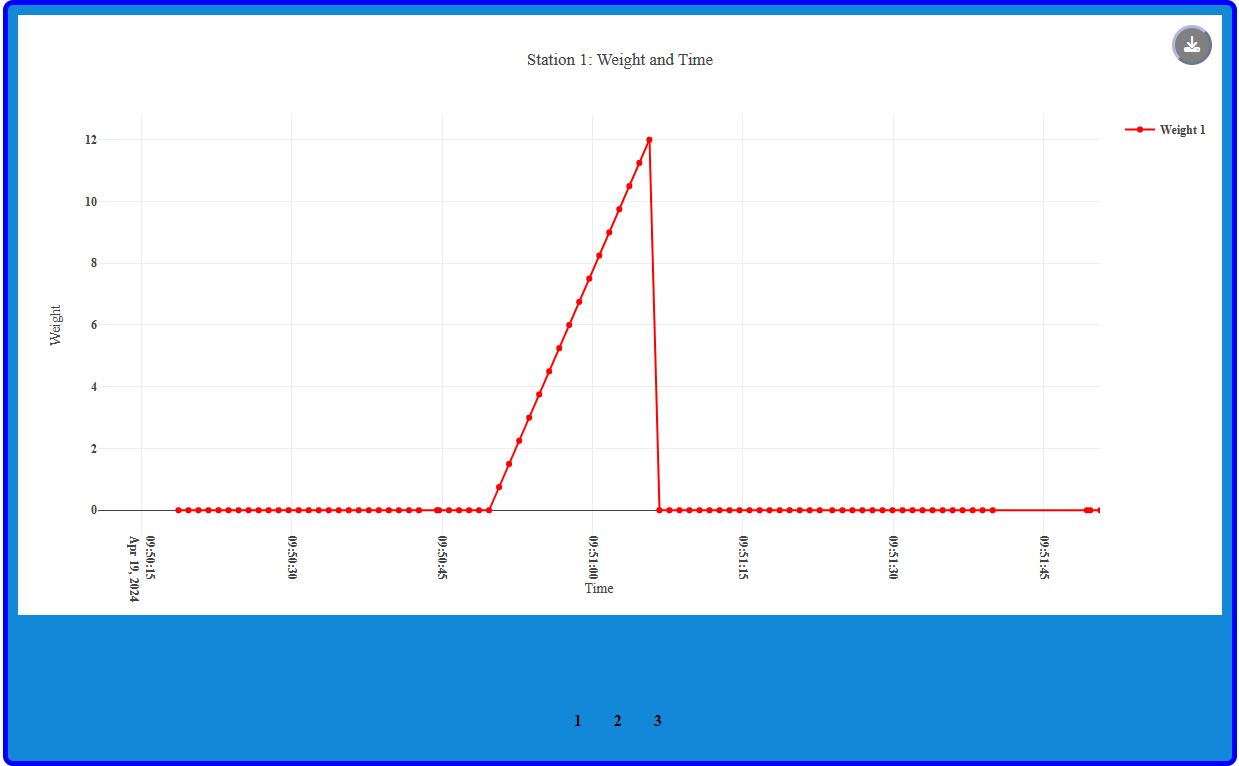
Hình 5.6. Trang Main

**Trang Database:**

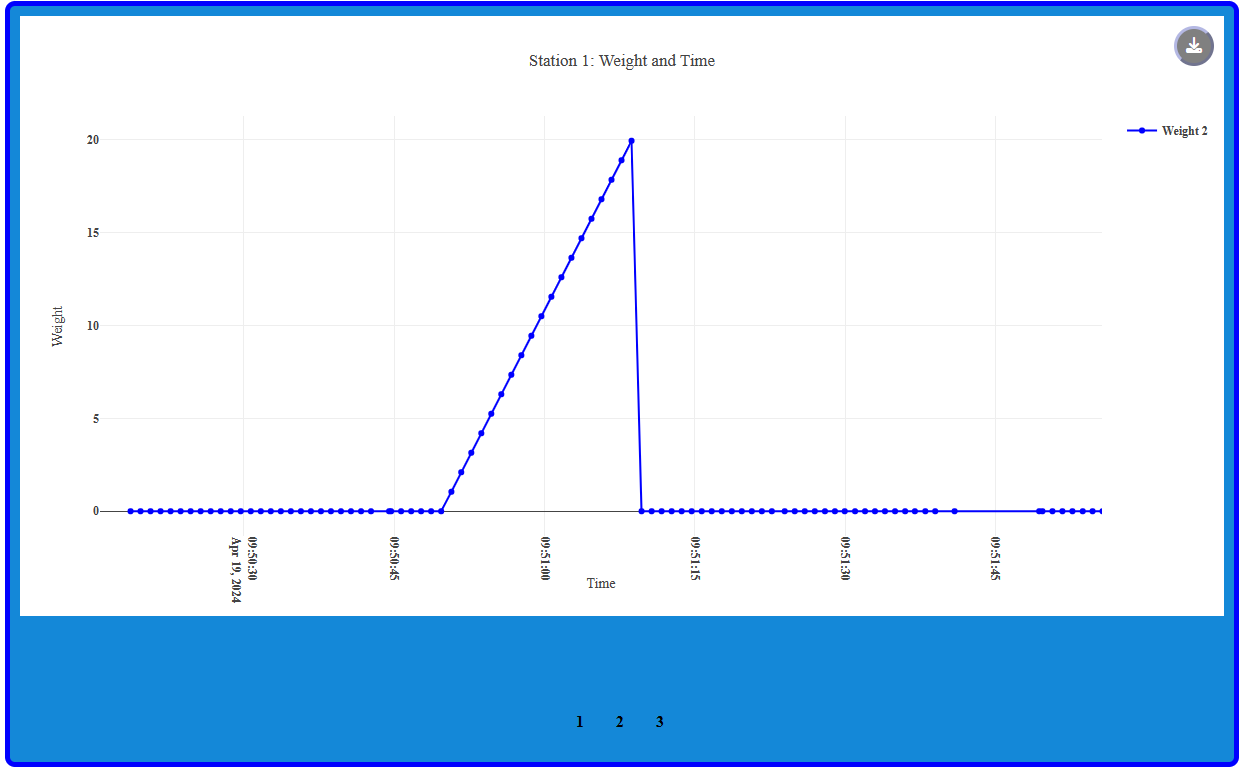
****

Hình 5.7. Trang Database

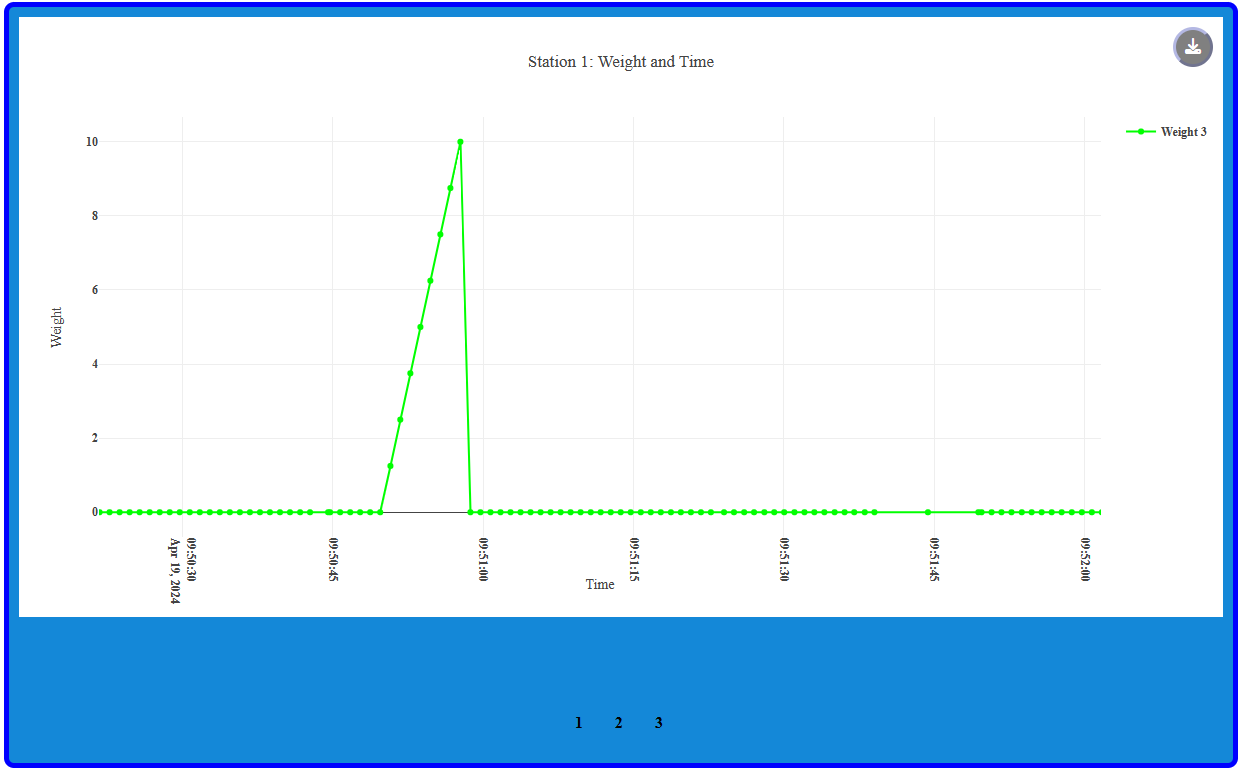
**Trang Diagram:**

****

Hình 5.8a. Trend 1

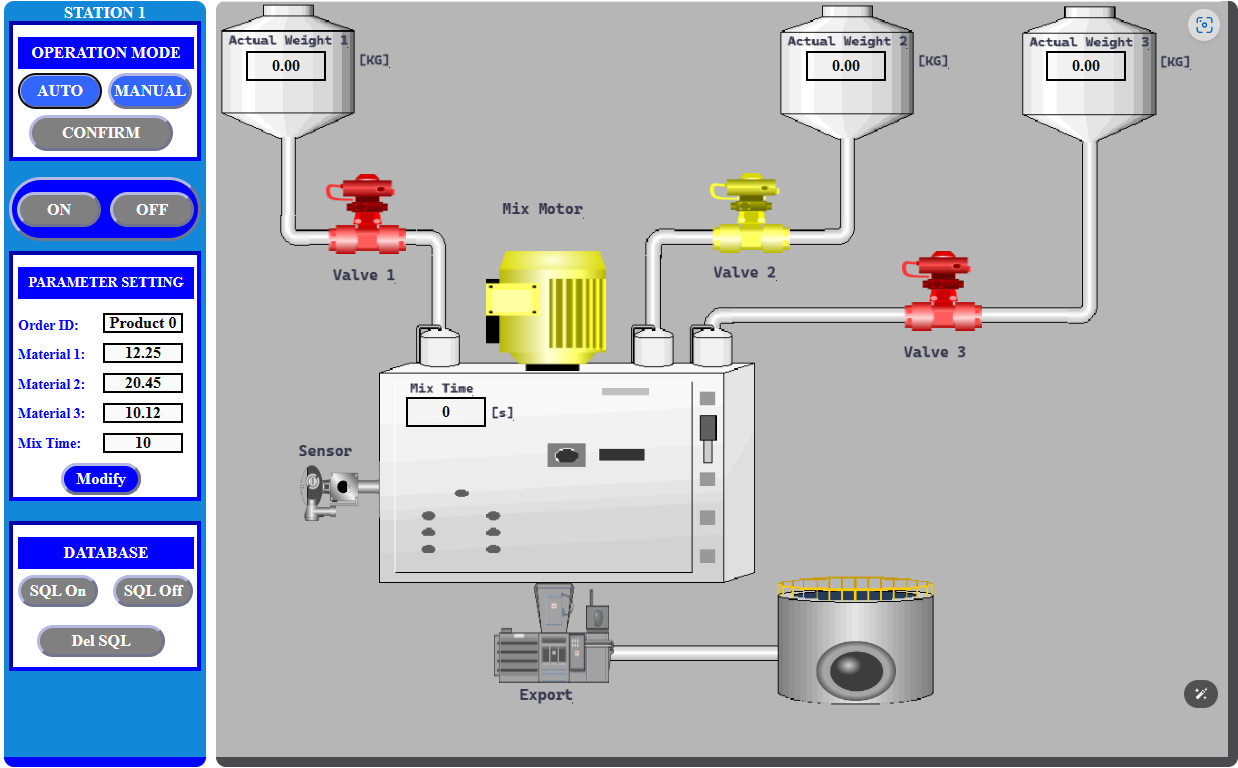
****

Hình 5.8b. Trend 2

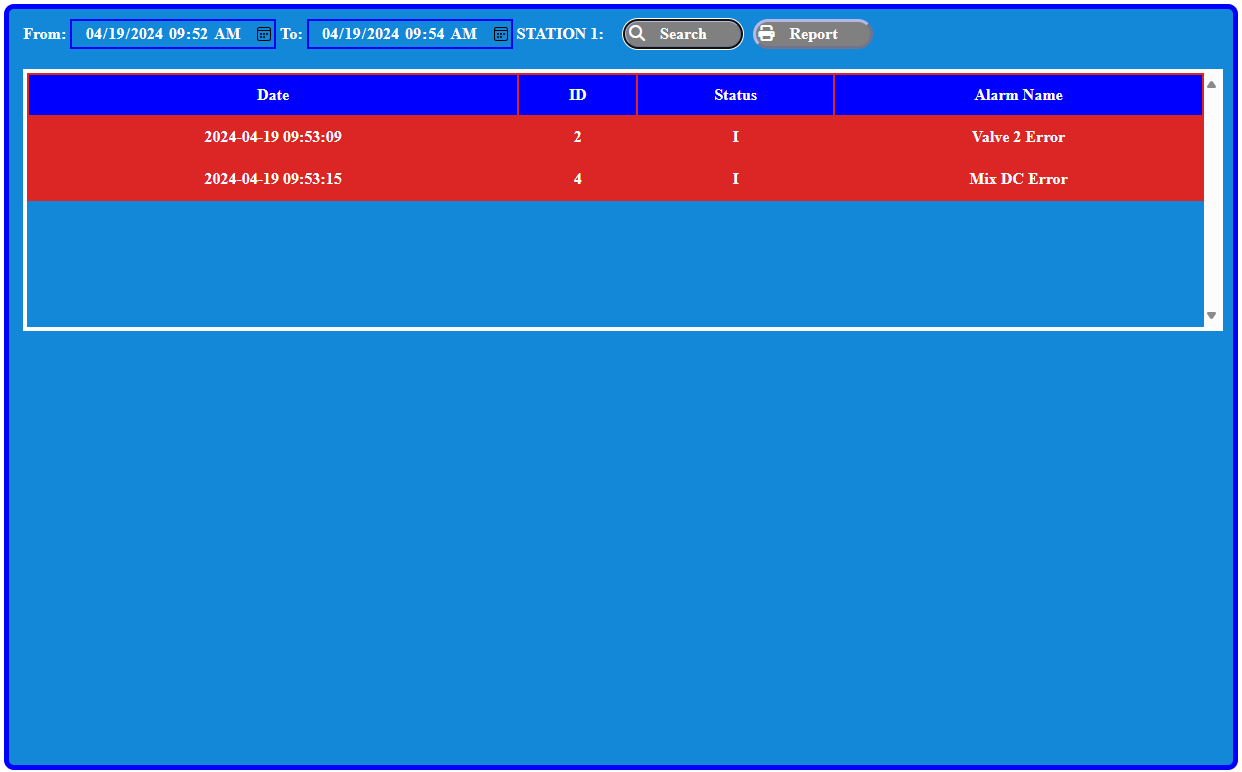
****

Hình 5.8c. Trend 3

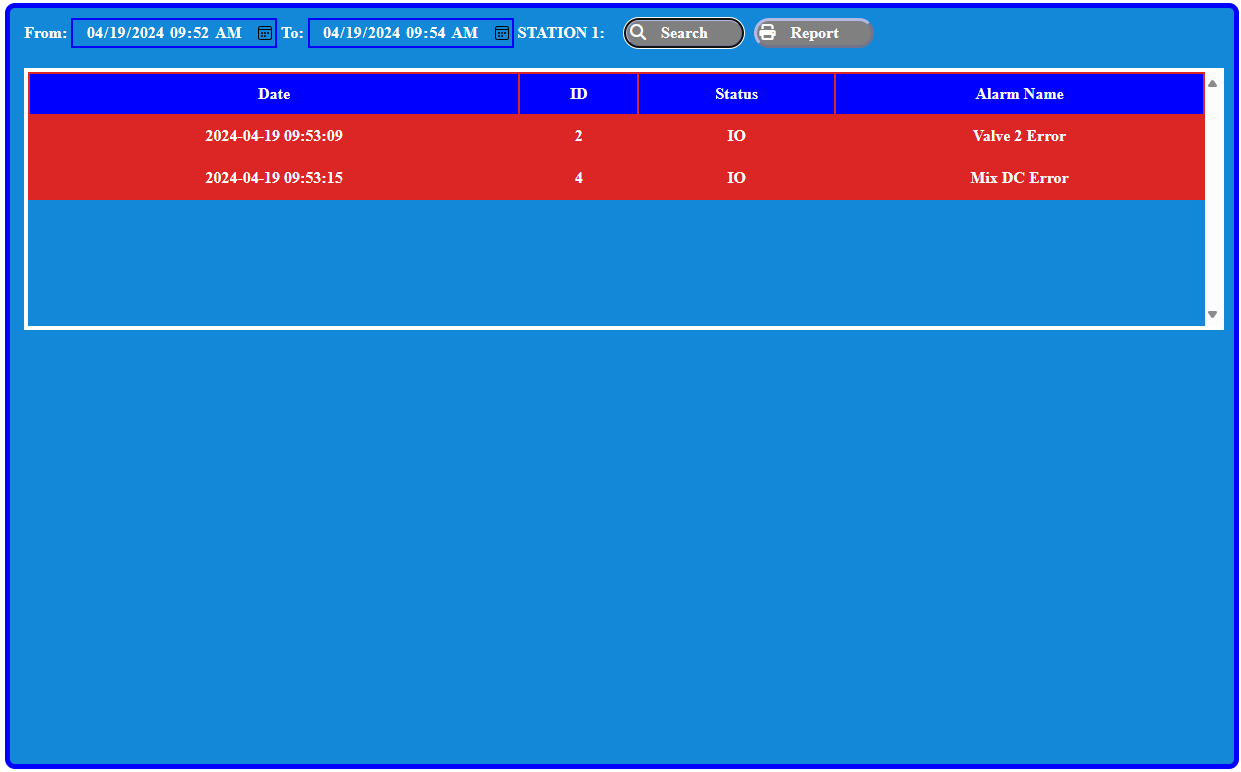
**Trang Alarm:**

****

Hình 5.9. Mô phỏng lỗi và khi kết thúc lỗi

****

Hình 5.11a. Có lỗi



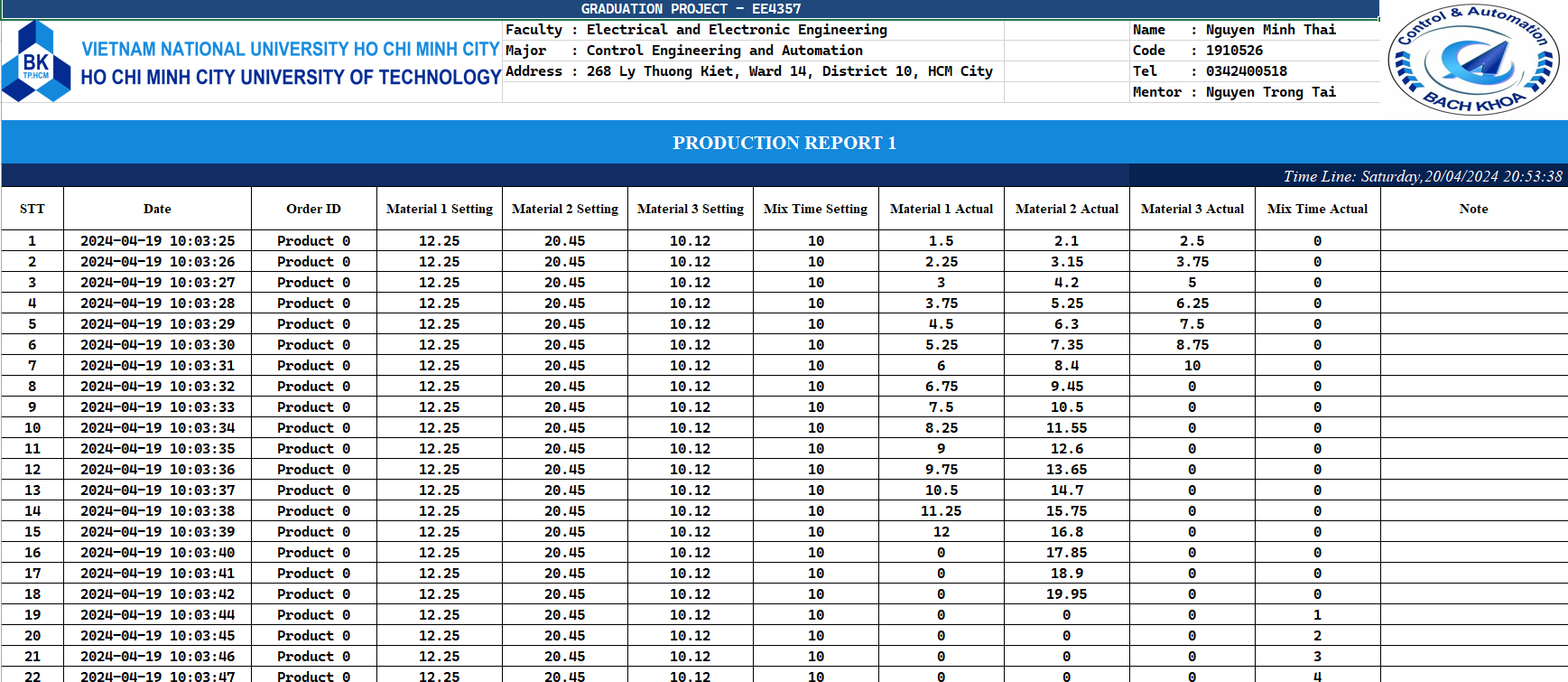
Hình 5.11b. Kết thúc lỗi

**Trang About:**

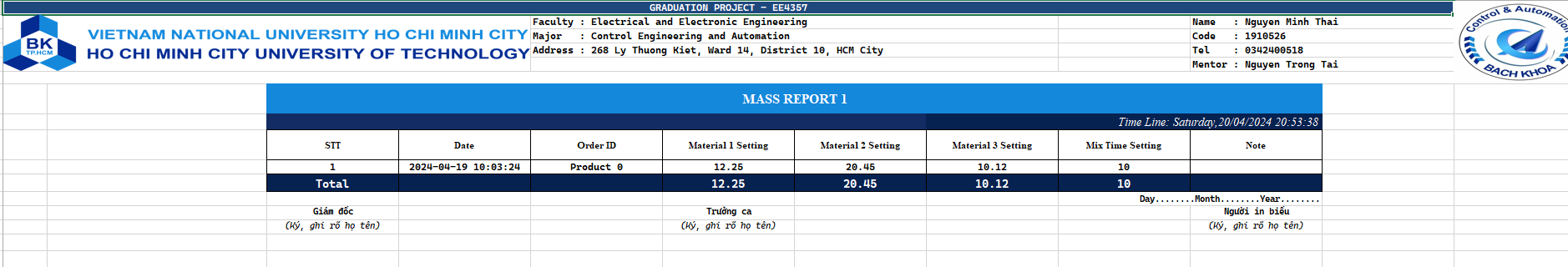
****

Hình 5.12 Trang About

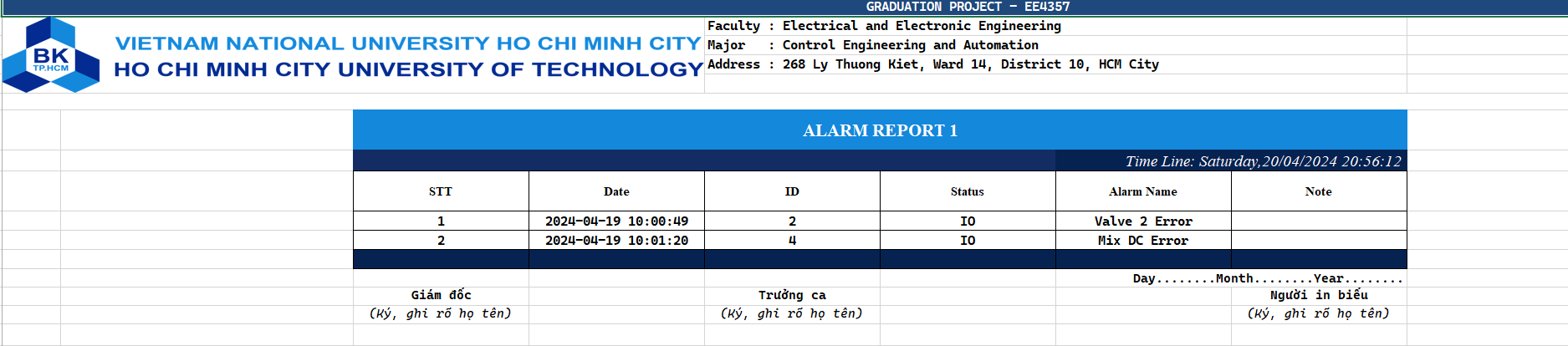
**Dữ liệu xuất ra Excel:**

****

Hình 5.13. Dữ liệu report khối lượng 1

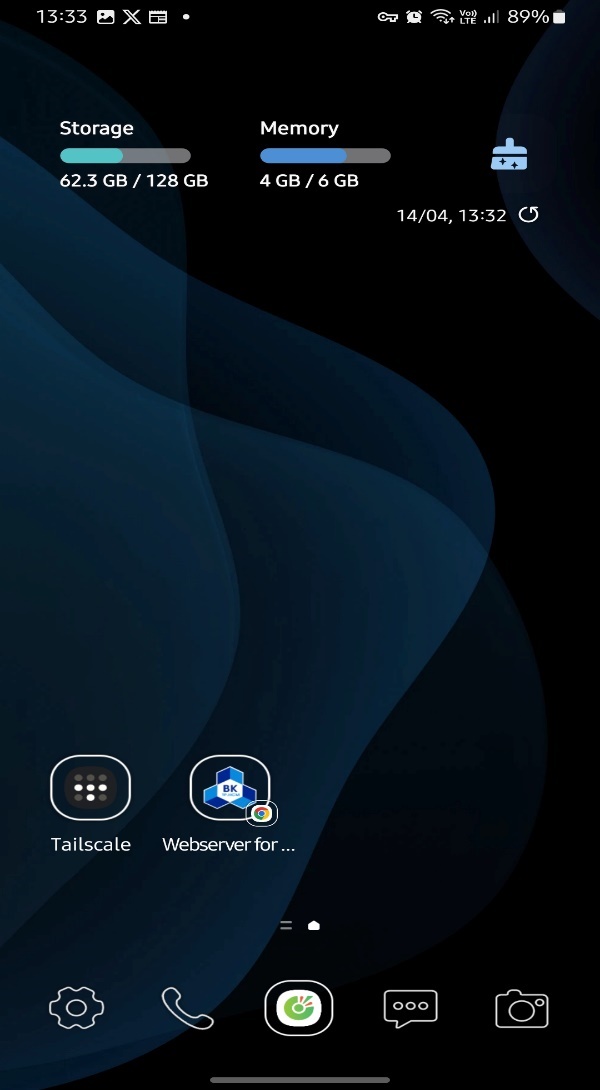


Hình 5.14. Dữ liệu report khối lượng tổng



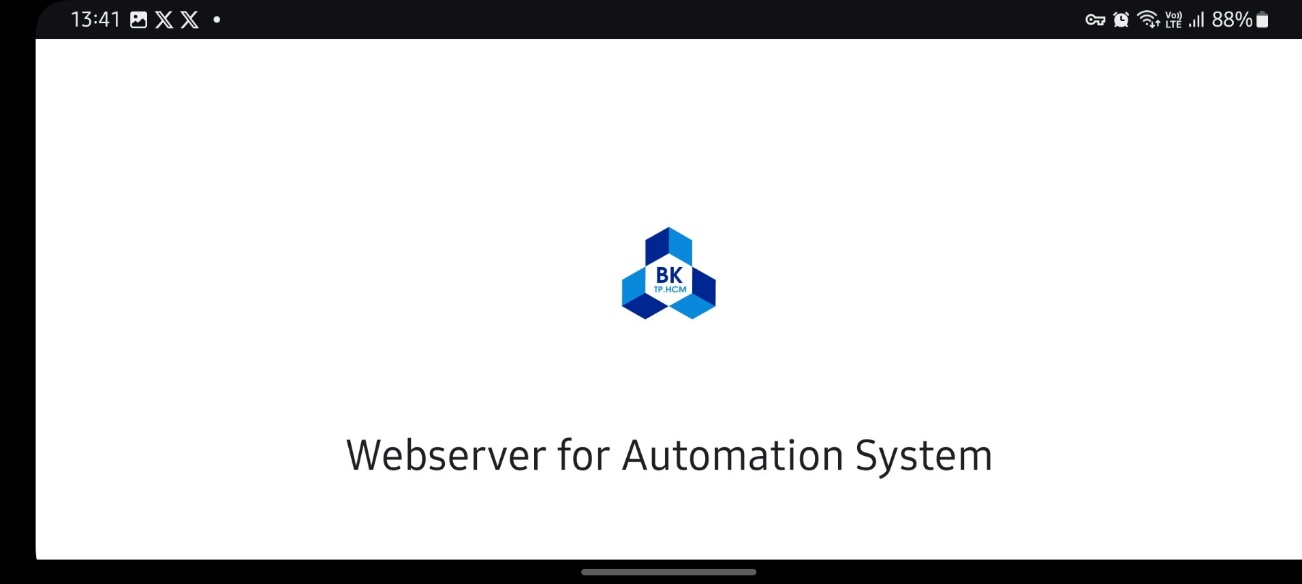
Hình 5.15. Dữ liệu report Alarm

**Ứng dụng trên điện thoại:**



Hình 5.16. Ứng dụng SCADA trên điện thoại

**Giao diện ở diện thoại:**



Hình 5.17. Giao diện chờ của ứng dụng SCADA trên điện thoại



Hình 5.18. Giao diện trên điện thoại

* 1. Đánh giá

Luận văn đã thực hiện được một số mục tiêu và nhiệm vụ đề ra

* Đã xây dựng được hệ thống điều khiển tự động với mô phỏng cấp trường, cấp điều khiển và cấp giám sát

Về chức năng:

* PLC: Giả lập mô hình hệ thống hoàn chỉnh, có chức năng điều khiển, xử lý sự kiện hệ thống, cung cấp dữ liệu cho quá trình giám sát
* SCADA: Thu thập, điều khiển và giám sát quá trình, cảnh báo (Alarm), vẽ đồ thị(Trend), xuất báo cáo (Report), bảo mật và quản lý người dùng (Security & User management)

Về kỹ thuật thực hiện: Thực hiện được khả năng lập trình hướng đối tượng ở PLC và lập trình SCADA dựa trên nền tảng Nodejs, từ đó khả năng mở rộng hệ thống và hoạt động được ở đa nền tảng.

Chương 6. KẾT LUẬN

* 1. **Những hạn chế**
* Mô phỏng hệ thống chưa đi sâu vào chi tiết cho từng thành phần của hệ thống
* Chưa ứng dụng thêm nhiều bộ điều khiển để xử lí hệ thống theo yêu cầu của người giám sát
* Hệ thống SCADA chưa đi sâu vào từng chức năng.
* Chưa phát triển hệ thống Webserver để giám sát từ xa (hệ thống chỉ hoạt động trong mạng nội bộ)
  1. **Hướng phát triển của luận văn**
* Nghiên cứu thêm từng phần của hệ thống, và phát triển các thành phần mô phỏng của cá nhân
* Phát triển mạnh hơn các tính năng trên SCADA.
* Tối ưu hoá khả năng ổn định của ứng dụng điện thoại
* Tiếp tục phát triển ứng dụng Web trên nền tảng Nodejs với các chức năng bảo mật, đồng thời phát triển dự án lên cộng đồng để có thể truy cập vào hệ thống ở bất kì nơi nào.

TÀI LIỆU THAM KHẢO

1. Nguyễn Đức Hoàng, Bài giảng SCADA, TPHCM
2. Tìm hiểu giao thức SocketIO. Truy cập từ:  
   [Introduction | Socket.IO](https://socket.io/docs/v4/)
3. Server API: [Server Installation | Socket.IO](https://socket.io/docs/v4/server-installation/)
4. Client API: [Client Installation | Socket.IO](https://socket.io/docs/v4/client-installation/)
5. MVC Model: [Model-View-Controller(MVC) architecture for Node applications - GeeksforGeeks](https://www.geeksforgeeks.org/model-view-controllermvc-architecture-for-node-applications/)
6. Create Website with Nodejs using Express and MySQL

[Express 4.x - API Reference (expressjs.com)](https://expressjs.com/en/4x/api.html)

[Install Express.js - javaTpoint](https://www.javatpoint.com/install-expressjs)

[GitHub - mysqljs/mysql: A pure node.js JavaScript Client implementing the MySQL protocol.](https://github.com/mysqljs/mysql)

1. Create report with exceljs:

[Exporting data in excel file in Node.js | by Amir Mustafa | Geek Culture | Medium](https://medium.com/geekculture/exporting-data-in-excel-file-in-node-js-f1b298997d47)

[exceljs - npm (npmjs.com)](https://www.npmjs.com/package/exceljs#fills)

1. Library for communicating with SIMENS PLC:

[nodes7 - npm (npmjs.com)](https://www.npmjs.com/package/nodes7)

1. https://stackoverflow.com/