

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO

NGHÀNH CÔNG NGHỆ KỸ THUẬT ĐIỆN TỬ-VIỄN THÔNG



HCMUTE

ĐỒ ÁN TỐT NGHIỆP

**THIẾT KẾ VÀ THI CÔNG HỆ THỐNG GIÁM
SÁT VÀ CHẨN ĐOÁN TRÊN XE CƠ GIỚI**

SVTH 1: NGUYỄN VŨ HUY HOÀNG

MSSV: 19161110

SVTH 2: ĐỖ THANH HOÀNG VĨ

MSSV: 19161199

Khóa : 2019

GVHD: ThS. TRƯƠNG QUANG PHÚC

Tp. Hồ Chí Minh, tháng 06 năm 2023

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO
NGHÀNH CÔNG NGHỆ KỸ THUẬT ĐIỆN TỬ-VIỄN THÔNG



HCMUTE

ĐỒ ÁN TỐT NGHIỆP
**THIẾT KẾ VÀ THI CÔNG HỆ THỐNG GIÁM
SÁT VÀ CHẨN ĐOÁN TRÊN XE CƠ GIỚI**

SVTH 1: NGUYỄN VŨ HUY HOÀNG

MSSV: 19161110

SVTH 2: ĐỖ THANH HOÀNG VĨ

MSSV: 19161199

Khóa : 2019

GVHD: ThS. TRƯƠNG QUANG PHÚC

Tp. Hồ Chí Minh, tháng 06 năm 2023

LỜI CẢM ƠN

Trước tiên, nhóm sinh viên thực hiện đề tài xin gửi lòng biết ơn sâu sắc nhất đến các bậc phụ huynh đã giúp đỡ, tạo mọi điều kiện tốt nhất về mặt tinh thần cũng như vật chất để nhóm có thể theo đuổi và phát triển bản thân trên con đường học tập trong suốt thời gian trong môi trường Đại học thời gian qua.

Bên cạnh đó, nhóm xin chân thành cảm ơn quý Thầy/Cô của khoa Đào Tạo Chất Lượng Cao đặc biệt là thầy Nguyễn Ngô Lâm và thầy Trương Quang Phúc là giáo viên hướng dẫn đã trao đổi và chia sẻ những kinh nghiệm quý giá có thể giúp nhóm hoàn thành đồ án tốt nghiệp này.

Nhóm thực hiện báo cáo cũng xin trân trọng cảm ơn anh Trần Đắc Trịnh, anh Hồ Đức Huy, anh Bùi Khánh Văn, chị Hồ Trần Thuận Thảo và tất cả các anh chị trong tập thể phòng ban EEU32 thuộc công ty TNHH Bosch Global Software Technologies đã đồng hành và giúp đỡ trong suốt quá trình thực hiện đề tài tại công ty.

Quãng thời gian thực hiện đồ án, được cộng tác với các anh chị đã có nhiều năm kinh nghiệm có thể nói đây là một bước đệm rất quan trọng cho tương lai sau này đối với các sinh viên ngành kỹ thuật. Việc được tiếp xúc với công việc thực tế, đổi chiếu và áp dụng những kiến thức được học ở trường đã giúp nhóm trưởng thành hơn rất nhiều. Không chỉ giúp hình dung rõ ràng được công việc của một kỹ sư thực thụ mà còn giúp nhóm có thể chuẩn bị hành trang thật chỉnh chu trên con đường sau này.

Ngoài ra nhóm xin cảm ơn các anh/chị khóa trước cũng như các bạn trong lớp 19161CLVT2A đặc biệt là bạn Trần Lam Nhật Vy và Trần Mạc Gia Phong đã nhiệt tình đóng góp chia sẻ ý kiến, giúp đỡ cho đề tài này.

Nhóm thực hiện báo cáo kính chúc Thầy/Cô đang công tác tại trường Đại Học Sư Phạm Kỹ Thuật TPHCM, cùng toàn thể các anh chị trong công ty TNHH Bosch Global Software lời chúc sức khỏe. Cuối cùng, một lần nữa xin cảm ơn tới các bậc phụ huynh, gia đình đã tạo điều kiện nền tảng tốt nhất cho nhóm để có thể hoàn thành đề tài.

LỜI CAM ĐOAN

Nhóm thực hiện đồ án tốt nghiệp với đề tài “THIẾT KẾ VÀ THI CÔNG HỆ THỐNG GIÁM SÁT VÀ CHẨN ĐOÁN TRÊN XE CƠ GIỚI” cam đoan không sao chép nội dung và kết quả của các công trình khác. Các nội dung tham khảo đã được trích dẫn đầy đủ ở phần tài liệu tham khảo.

Đại diện nhóm

Đỗ Thanh Hoàng Vũ - Nguyễn Vưu Huy Hoàng

TÓM TẮT

Đề tài “THIẾT KẾ VÀ THI CÔNG HỆ THỐNG GIÁM SÁT VÀ CHẨN ĐOÁN TRÊN XE CƠ GIỚI” với mục tiêu hướng tới là sử dụng ECU thực tế được hỗ trợ từ công ty Bosch Global Software Việt Nam để có thể đọc các giá trị được gửi từ ECU thông qua chuẩn truyền thông CAN và kết hợp chuẩn giao thức J1939.

Từ việc có thể đọc được các giá trị đó như nhiệt độ động cơ, tốc độ xe, mức nhiên liệu và quãng đường đi được thì sẽ sử dụng một Gateway giúp trung chuyển các giá trị thông qua các chuẩn giao thức TCP/IP như MQTT và HTTP để gửi lên một Cloud giúp lưu trữ và gửi về cho các ứng dụng người dùng.

Đối với các giao diện người dùng như web, ứng dụng di động và UI trên một màn hình gắn trực tiếp. Người dùng có thể đăng nhập và kết nối vào ứng dụng di động cũng như web để có thể theo dõi quan sát giám sát xe từ xa. UI trên màn hình có tích hợp cảm ứng người dùng có thể tương tác và xem các thông số tính năng trên hệ thống. Bên cạnh đó, sẽ đánh giá toàn bộ hệ thống và kiểm tra hiệu năng của hệ thống cũng như xây dựng mô hình hoàn chỉnh có thể chạy thực tế và đảm bảo kết quả hoạt động ổn định.

Cuối cùng, kết quả thực hiện nghiên cứu hệ thống của nhóm đã hoàn thành được mô hình có đầy đủ các chức năng từ yêu cầu người dùng. Đi kèm với đó là hướng phát triển áp dụng mô hình vào thực tế nhất từ đó có thể phát triển thêm về các tính năng đáp ứng các yêu cầu của người dùng.

MỤC LỤC

LỜI CẢM ƠN.....	i
LỜI CAM ĐOAN	ii
TÓM TẮT	iii
MỤC LỤC	iv
DANH MỤC CÁC TỪ VIẾT TẮT	vii
DANH MỤC BẢNG BIỂU	viii
DANH MỤC HÌNH ẢNH	ix
CHƯƠNG 1: TỔNG QUAN	1
1.1 GIỚI THIỆU	1
1.2 MỤC TIÊU ĐỀ TÀI	2
1.3 TÌNH HÌNH NGHIÊN CỨU	3
1.3.1 Tình hình nghiên cứu ngoài nước:.....	3
1.3.2 Tình hình nghiên cứu trong nước:	4
1.4 PHƯƠNG PHÁP NGHIÊN CỨU	4
1.5 BỐ CỤC CỦA ĐỀ TÀI	5
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	6
2.1 TỔNG QUAN VỀ HỆ THỐNG IOTS, NHÚNG, GIAO THÚC CAN VÀ CHUẨN GIAO THÚC J1939 TRÊN XE CƠ GIỚI.....	6
2.2 GIỚI THIỆU VỀ CHUẨN GIAO THÚC J1939.....	8
2.3 GIỚI THIỆU FRAMEWORK SỬ DỤNG TRONG HỆ THỐNG	12
2.3.1 Flutter.....	12
2.3.2 NodeJS	14
2.3.3 ReactJS	15
2.4 GIỚI THIỆU VỀ CƠ SỞ DỮ LIỆU.....	17
2.4.1 Các chứng năng chính của phpMyAdmin	18
2.5 PHẦN MỀM NODE RED.....	20
2.5.1 Các tính năng của node red.	20
2.5.2 Vùng làm việc của node-red.....	21
2.5.3 Các thiết lập một node trong node-red.	21
2.6 RASPBERRY PI 4B	23

2.7 MODULE WAVESHARE CAN HAT RS485.....	26
2.8 MODULE NEO-6M GPS.....	28
CHƯƠNG 3: THIẾT KẾ VÀ THI CÔNG	30
3.1 YÊU CẦU HỆ THỐNG	30
3.2 ĐẶC TẢ HỆ THỐNG	30
3.2.1 Chức năng của hệ thống	30
3.2.2 Mô hình tổng thể hệ thống.....	31
3.2.3 Sơ đồ khái niệm	32
3.2.4 Hoạt động của hệ thống	33
3.3 THIẾT KẾ PHẦN CỨNG	35
3.3.1 Khối xử lý trung tâm	36
3.3.2 Khối đọc CAN.....	37
3.3.3 Khối đọc định vị	38
3.3.4 Khối hiển thị	39
3.3.5 Khối nguồn	40
3.3.6 Sơ đồ nguyên lý	42
3.4 THIẾT KẾ PHẦN MỀM	42
3.4.1 Lưu đồ giải thuật trên central gateway	42
3.4.2 Lưu đồ giải thuật trên giao diện người dùng	44
CHƯƠNG 4: KẾT QUẢ, ĐÁNH GIÁ HỆ THỐNG	60
4.1 KẾT QUẢ PHẦN CỨNG	60
4.2 GIAO DIỆN HIỂN THỊ TRÊN ỨNG DỤNG ĐIỆN THOẠI ANDROID	60
4.2.1 Hiển thị giao diện đăng nhập.....	61
4.2.2 Giao diện hiển thị các thông số	69
4.2.3 Giao diện bản đồ	69
4.2.4 Giao diện thông tin của software.....	71
4.3 GIAO DIỆN HIỂN THỊ TRÊN WEB	73
4.3.1 Giao diện đăng nhập	73
4.3.2 Giao diện chính.....	74
4.4 GIAO DIỆN HIỂN THỊ GUI TRÊN MÀN HÌNH 7INCH.....	79
4.4.1 Giao diện trang Home	81

4.4.2 Giao diện trang Controller.....	82
4.4.3 Map	91
4.5 ĐÁNH GIÁ HỆ THỐNG	92
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	93
5.1 KẾT LUẬN	93
5.2 HƯỚNG PHÁT TRIỂN	94
TÀI LIỆU THAM KHẢO	95

DANH MỤC CÁC TỪ VIẾT TẮT

Viết tắt	Mô tả
ECU	Engine Control Unit
CAN	Controller Area Network
GPS	Global Positioning System
PGN	Parameter Group Number
UI	User Interface
GUI	Graphical User Interface
OBD	On-Board Diagnostics
SAE	Society of Automotive Engineers
IoTs	Internet of Things
PDU	Protocol Data Unit
CRC	Cyclic Redundancy Check
VIN	Vehicle Identical Number
MQTT	Message Queuing Telemetry Transport
HTTP	Hyper Text Transfer Protocol
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver – Transmitter

DANH MỤC BẢNG BIỂU

Bảng 2.1: So sánh giữa SQL và NoSQL.....	17
Bảng 2.2: Thông số kỹ thuật của module	25
Bảng 2.3: Bảng thông tin các chân của Raspberry pi4	26
Bảng 2.4: Thông số kỹ thuật của module	27
Bảng 2.5: Sơ đồ chân của module RS485 CAN HAT	28
Bảng 2.6: Ký hiệu chân module NEO-6M	28
Bảng 2.7: Thông số của module GPS-NEO-6M.....	29
Bảng 3.1: Sơ đồ kết nối module Waveshare RS485 CAN HAT với Raspberry	37
Bảng 3.2: Sơ đồ kết nối module NEO-6M với Raspberry	39
Bảng 3.3: Sơ đồ kết nối giữa màn hình 7Inch và Raspberry	40
Bảng 3.4: Các thông số về dòng điện, điện áp và công suất tiêu thụ của ECU	40
Bảng 3.5: Các thông số về dòng điện, điện áp, công suất tiêu thụ của các linh kiện khác	41

DANH MỤC HÌNH ẢNH

Hình 2.1: Tổng quan hệ thống nhúng trên xe ô tô	6
Hình 2.2: Giao thức CAN	7
Hình 2.3: Giao thức J1939	8
Hình 2.4: Một khung truyền cơ bản CAN, J1939	9
Hình 2.5: Chi tiết khung truyền Identifier trong J1939	10
Hình 2.6: Chi tiết khung truyền CAN, J1939	11
Hình 2.7: Framework Flutter ứng dụng lập trình di động	12
Hình 2.8: Ngôn ngữ lập trình Dart trong Flutter	13
Hình 2.9: Nền tảng Platform NodeJS	14
Hình 2.10: ExpressJS framework phổ biến trong NodeJS	15
Hình 2.11: ReactJS framework	16
Hình 2.12: Biểu tượng của cơ sở dữ liệu phpMyAdmin	17
Hình 2.13: Trang chủ của phpMyAdmin	18
Hình 2.14: Bảng chứa thông tin dữ liệu	19
Hình 2.15: Logo phần mềm node-red	20
Hình 2.16: Các vùng làm việc chính của node-red	21
Hình 2.17: Định dạng cơ bản của một node	22
Hình 2.18: Node chứa hàm của người lập trình	22
Hình 2.19: Hình minh họa kết quả thực thi một flow trên node-red	23
Hình 2.20: Sơ đồ kết nối raspberry	24
Hình 2.21: Sơ đồ thứ tự các chân của raspberry pi4	25
Hình 2.22: Module RS485 CAN HAT	26
Hình 2.23: Kết nối module RS485 CAN HAT và Raspberry pi4	27
Hình 2.24: Mô hình hoạt động của module CAN HAT RS485	27
Hình 2.25: Module NEO-6M	28
Hình 3.1: Mô hình tổng quát của hệ thống	31
Hình 3.2: Sơ đồ khối tổng thể hệ thống	32
Hình 3.3: Sơ đồ hoạt động của hệ thống	33
Hình 3.4: Sơ đồ khối chi tiết tại central gateway	35
Hình 3.5: Sơ đồ nguyên lý Raspberry Pi 4	36
Hình 3.6: Sơ đồ nguyên lý kết nối module CAN HAT với Raspberry	38
Hình 3.7: Sơ đồ nguyên lý kết nối module CAN HAT với Raspberry và ECU	38
Hình 3.8: Sơ đồ nguyên lý kết nối module NEO-6M với Raspberry	39
Hình 3.9: Sơ đồ nguyên lý kết nối màn hình 7Inch với Raspberry	40
Hình 3.10: Sơ đồ nguyên lý kết nối phần cứng	42
Hình 3.11: Lưu đồ giải thuật xử lý CAN Frame	43
Hình 3.12: Lưu đồ giải thuật truyền gửi dữ liệu	44
Hình 3.13: Lưu đồ giải thuật đăng nhập tài khoản	45

Hình 3.14: Lưu đồ người dùng sử dụng xem thông số của xe	46
Hình 3.15: Lưu đồ người dùng sử dụng xem bản đồ.....	47
Hình 3.16: Lưu đồ người dùng sử dụng xem thông tin của xe.....	48
Hình 3.17: Lưu đồ người dùng sử dụng xem thông số trên giao diện Web	49
Hình 3.18: Lưu đồ hàm chính của chương trình hiển thị GUI	51
Hình 3.19: Lưu đồ hàm con hiển thị tốc độ của xe	52
Hình 3.20: Lưu đồ chương trình con hiển thị nhiệt độ động cơ	53
Hình 3.21: Lưu đồ hàm con hiển thị trạng thái bảo trì của phương tiện	54
Hình 3.22: Lưu đồ hàm con hiển thị mức nhiên liệu của động cơ	55
Hình 3.23: Lưu đồ hàm con hiển thị vị trí của phương tiện	56
Hình 3.24: Lưu đồ hàm con trích xuất dữ liệu	57
Hình 3.25: Lưu đồ hàm con xóa dữ liệu tự động.....	58
Hình 4.1: Kết quả phần cứng hoàn chỉnh	60
Hình 4.2: Giao diện đăng nhập	61
Hình 4.3: Các thông tin cần cho đăng nhập.....	62
Hình 4.4: Giao diện hiển thị các thông số	69
Hình 4.5: Giao diện và tính năng thông số tốc độ xe	70
Hình 4.6: Giao diện và cảnh báo thông số nhiệt độ động cơ.....	69
Hình 4.7: Giao diện và cảnh báo thông số mức nhiên liệu.....	69
Hình 4.8: Giao diện và cảnh báo thông số quãng đường đi được.....	69
Hình 4.9: Tổng quan trang giao diện bản đồ	69
Hình 4.10: Khi ấn “Search Gas Station”	69
Hình 4.11: Khi ấn chỉ đường tới cây xăng.....	70
Hình 4.12: Giao diện hiển thị phiên bản của Software.....	71
Hình 4.13: Chức năng đăng xuất	72
Hình 4.14: Giao diện đăng nhập	73
Hình 4.15: Khi đăng nhập sai mật khẩu	74
Hình 4.16: Giao diện chính.....	74
Hình 4.17: Nhiệt độ động cơ	75
Hình 4.18: Cảnh báo khi nhiệt độ động cơ cao	76
Hình 4.19: Mức nhiên liệu.....	76
Hình 4.20: Cảnh báo khi nhiên liệu gần hết	77
Hình 4.21: Tốc độ hiện tại của xe.....	77
Hình 4.22: Cảnh báo tốc độ xe cao	78
Hình 4.23: Thông số quãng đường xe đã đi được	78
Hình 4.24: Thông báo tới lúc xe cần bảo trì, bảo dưỡng	79
Hình 4.25: Giao diện lựa chọn các tab của GUI.....	80
Hình 4.26: Giao diện tổng thể GUI.	81
Hình 4.27: Các thành phần trong trang Controller.	82
Hình 4.28: Thông số tốc độ xe	83

Hình 4.29: Thông số nhiệt độ động cơ	84
Hình 4.30: Thông số mức nhiên liệu động cơ	85
Hình 4.31: Thông số quãng đường đi được	86
Hình 4.32: Chức năng lưu trữ các thông số	87
Hình 4.33: Trích xuất dữ liệu lưu trữ	88
Hình 4.34: Hiển thị các giá trị lưu trữ theo ngày	89
Hình 4.35: Cảnh báo nhiệt độ động cơ vượt quá ngưỡng cho phép	90
Hình 4.36: Cảnh báo tốc độ của xe vượt quá ngưỡng cho phép	90
Hình 4.37: Cảnh báo nhiên liệu thấp quá ngưỡng cho phép	91
Hình 4.38: Giao diện bản đồ	91

CHƯƠNG 1: TỔNG QUAN

1.1 GIỚI THIỆU

Việt Nam được biết đến là nước đang phát triển, với ngành nông nghiệp giữ vai trò cốt lõi trong nền kinh tế, các mặt hàng nông nghiệp của ta ngày càng được ưa chuộng và đánh giá cao trên thị trường quốc tế. Để giữ vững vai trò ấy bên cạnh việc nâng cao hiệu suất và chất lượng nông sản ta cũng cần phải cải tiến và phát triển máy móc thiết bị phục vụ cho nền nông nghiệp. Ngày nay, sự phát triển của công nghệ trong các lĩnh vực điện tử, truyền thông nói chung đặc biệt là công nghệ IoTs (Internet of Things) đã và đang dần được áp dụng vào trong máy móc và các loại xe phục vụ cho việc sản xuất để có thể giải quyết được những vấn đề về nâng cao chất lượng sản phẩm trong ngành nông nghiệp. Theo báo cáo cho thấy việc ứng dụng công nghệ IoTs vào sản xuất nông nghiệp đã đóng góp trên 30% giá trị gia tăng, ngoài ra mức độ nông sản bị tổn thất giảm đáng kể cũng như mức độ cơ giới hóa ở khâu làm đất của các loại cây hàng năm đạt 94% và khâu thu hoạch đạt 50%. Từ đó, có thể thấy việc đưa các công nghệ IoTs vào nuôi, trồng, canh tác đang là giải pháp tối ưu phần nào giải quyết được các yêu cầu về cải thiện và nâng cao chất lượng và hiệu suất nông sản.

Hiện nay, các công nghệ IoTs nói chung và hệ thống nhúng nói riêng đã được ứng dụng nhiều trong nông nghiệp bởi tính thiết thực cũng như đáp ứng được các nhu cầu cần thiết cho việc điều khiển và quản lý các phương tiện sản xuất từ xa. Đối với ngành ô tô (Automotive) việc ứng dụng các công nghệ IoTs để cải tiến các hệ thống tiện ích trên xe nông nghiệp như máy kéo, máy xúc, máy cày đang thu hút rất nhiều sự quan tâm. Việc điều khiển và quản lý một hệ thống trên xe cơ giới nói chung và xe nông nghiệp nói riêng đều thông qua một bộ điều khiển điện tử được biết đến là ECU (Electronic control unit). Đây được xem như là “bộ não” của một chiếc xe, nó có thể giao tiếp với các thiết bị khác thông qua một giao thức mạng nối tiếp là CAN (Controller area network). Nhờ vào giao thức này, các ECU có thể kết nối rất nhiều thiết bị trên một mạng duy nhất với tốc độ cao. Khác với các dòng xe ô tô khách, đối với các loại xe cơ giới, hiệp hội các kỹ sư trong ngành ô tô –

SAE (Society of Automotive Engineers) đã nghiên cứu ra chuẩn giao thức J1939 chuyên dùng cho các dòng xe này. Chuẩn giao thức J1939 được biết đến là một giao thức tiêu chuẩn trên các xe nông nghiệp vì có thể dùng cho các hệ thống chẩn đoán cũng như liên lạc giữa các thành phần xe. Đây là giao thức cấp cao được phát triển dựa trên giao thức CAN, chịu trách nhiệm cho việc liên lạc, giao tiếp thông tin của các ECU trên tất cả các loại xe cơ giới.

Việc ứng dụng một bộ điều khiển điện tử ECU có thể kết nối Internet và giúp người dùng theo dõi, giám sát cũng như chẩn đoán các sự cố trên xe cơ giới phục vụ cho việc sản xuất từ đó nâng cao năng suất và quản lý được các hoạt động của xe, giảm thiểu thiệt hại cũng như gia tăng mức độ cơ giới hóa ở khâu làm đất và khâu thu hoạch của các loại nông sản đang là một xu hướng cần được phát triển trong thực tế. Vì thế hiện nay nhiều tập đoàn trong ngành công nghiệp ô tô đều đã tiến đến cải tiến và phát triển dựa trên xu hướng này nên nhóm chúng tôi quyết định nghiên cứu lý thuyết liên quan đến đề tài từ đó thiết kế và thi công hệ thống giám sát, chẩn đoán dành cho xe nông nghiệp dựa trên chuẩn giao thức J1939.

1.2 MỤC TIÊU ĐỀ TÀI

Đối với đề tài nhóm chúng tôi đã đề ra 4 mục tiêu chính:

Thứ nhất: thiết kế hệ thống có thể xử lý, chuyển đổi các dữ liệu CAN được gửi từ ECU theo chuẩn J1939 và có thể xác định vị trí của phương tiện thông qua GPS.

Thứ hai: thiết lập và cấu hình Server có thể truyền các dữ liệu lên Database và giao tiếp với giao diện người dùng.

Thứ ba: thiết kế giao diện hiển thị các thông số xe như nhiệt độ động cơ, tốc độ xe, quãng đường đã đi được, mức nhiên liệu lên trên Web và ứng dụng điện thoại, đồng thời sử dụng một màn hình GUI (graphical user interface) hiển thị để người dùng có thể trực tiếp xem các thông số trên xe.

Thứ tư: tiến hành thiết lập kiểm thử hệ thống và đảm bảo độ trễ truyền dữ liệu của hệ thống không quá 5s.

1.3 TÌNH HÌNH NGHIÊN CỨU

Trong phần tình hình nghiên cứu chúng tôi muốn trình bày về các công trình đã được nghiên cứu có thể hỗ trợ cho việc xây dựng và phát triển đề tài từ đó đưa ra được phương án thực hiện tốt, tối ưu nhất cho đề tài. Góp phần cho việc có thể hoàn thành hoàn chỉnh cũng như chỉnh chu nhất cho toàn bộ hệ thống.

1.3.1 Tình hình nghiên cứu ngoài nước:

Việc ứng dụng các công nghệ IoTs vào một hệ thống trong ngành ô tô đang là một xu hướng trên thế giới cho nên hiện nay có rất nhiều bài báo có nghiên cứu về việc ứng dụng đó, tuy nhiên nhóm thực hiện đã tìm kiếm và nghiên cứu được hai bài viết từ quốc tế cảm thấy phù hợp với đề tài cũng như là đóng góp nhiều nhất cho quá trình xây dựng và thực hiện dự án.

Đầu tiên trong bài viết [1] đã đề cập liên quan đến các nội dung nghiên cứu về chuẩn truyền thông qua J1939, cụ thể là kích thước và nội dung truyền nhận dữ liệu của một khung dữ liệu theo chuẩn J1939, cũng như cách thức chẩn đoán lỗi, chẩn đoán sự cố thông qua chuẩn truyền thông trên. Dựa trên bài viết [2] ta có thể hình dung được cách thức hoạt động và truyền nhận dữ liệu của chuẩn truyền thông J1939, từ đó có thể lựa chọn các khung dữ liệu chứa các thông số cần thiết.

Tiếp đến trong bài viết [2] nhóm tác giả đã xây dựng hệ thống IoTs hoàn chỉnh với bộ vi điều khiển Raspberry dùng hai chuẩn giao tiếp OBDII và giao thức truyền thông MQTT để tiến hành theo dõi và chẩn đoán động cơ trên xe ô tô, dựa vào dữ liệu được truyền từ khung dữ liệu OBDII, hệ thống có thể đưa ra các chẩn đoán về tình hình hoạt động của xe, thông giao thức truyền thông MQTT để gửi và nhận dữ liệu với hệ thống Internet. Bài viết tuy nghiên cứu dựa trên chuẩn OBDII tuy nhiên nhìn rộng ở mức độ xây dựng hệ thống rất phù hợp để áp dụng vào đề tài.

Thông qua hai bài viết [1] và [2] ta có thể hình dung ra cách vận hành của một hệ thống IoTs cũng như cách thức hoạt động, chẩn đoán và truyền nhận dữ liệu trong động cơ của xe ô tô khi áp dụng chuẩn giao thức J1939.

1.3.2 Tình hình nghiên cứu trong nước:

Trong phạm vi nghiên cứu trong nước chúng tôi đã chọn hai bài viết [3] và [4] đây là hai công trình được nghiên cứu song song, cùng thực hiện dưới sự tài trợ của công ty Bosch, có thể coi hai công trình này là hai công trình nền tảng để phát triển đề tài của chúng tôi.

Bài viết [3] và [4] là các dự án chẩn đoán và theo dõi các thông số của xe máy cày thông qua chuẩn giao tiếp J1939, trong hai công trình nghiên cứu trên nhóm tác giả đã trình bày về các thông số quan trọng dùng để chẩn đoán hoạt động của động cơ xe máy cày (tốc độ động cơ, nhiệt độ động cơ, mức độ hao hụt nhiên liệu) và mô phỏng kết quả chẩn đoán thông qua vi điều khiển Arduino Uno R3. Hệ thống tương đối hoàn chỉnh, tuy nhiên việc áp dụng vi điều khiển Arduino Uno R3 vẫn chưa đáp ứng được yêu cầu về tốc độ xử lý và truyền dẫn dữ liệu trong hệ thống, ta có thể phát triển hệ thống mới nhằm khắc phục được các nhược điểm này.

Dựa vào hai bài viết [3] và [4] giúp ta hình dung cụ thể các yếu tố, những yêu cầu đặc thù, cần được chú ý trong việc theo dõi và chẩn đoán trên xe máy kéo ở riêng thị trường Việt Nam.

1.4 PHƯƠNG PHÁP NGHIÊN CỨU

Để thực hiện đề tài này, đầu tiên nhóm chúng tôi sẽ tìm hiểu, thu nhập, phân tích lý thuyết, cách vận hành và hoạt động của một ECU. Nghiên cứu và phân tích sâu vào các yêu cầu của người dùng từ đó đưa ra được các thông số thích hợp với các yêu cầu đó. Để tiến đến thiết kế chi tiết một hệ thống thì nhóm chúng tôi sẽ thiết lập các khối theo từng thành phần và phân chia công việc dựa vào các khối thành phần trên. Sau đó sẽ tổng hợp các khối với nhau thông qua các chuẩn giao tiếp để có thiết kế hệ thống một cách hoàn chỉnh. Cuối cùng nhóm sẽ kiểm tra hoạt động của toàn hệ thống dựa trên các kết quả thực tế.

1.5 BỘ CỤC CỦA ĐỀ TÀI

Chương 1: Tổng quan:

Giới thiệu khái quát và tổng quan về tình hình nghiên cứu cũng như các phương pháp nghiên cứu của đề tài.

Chương 2: Cơ sở lý thuyết:

Giới thiệu mô tả về cách hoạt động đặc điểm của các Framework như Flutter, NodeJS, Node-red, ReactJS, MySql và các chuẩn giao thức CAN, J1939.

Chương 3: Thiết kế và thi công:

Trình bày các sơ đồ khối hệ thống, nguyên lý hoạt động của các khối thành phần hệ thống.

Chương 4: Kết quả, đánh giá hệ thống:

Đánh giá các chức năng, trình bày kết quả cuối cùng đạt được từ phần cứng và phần mềm.

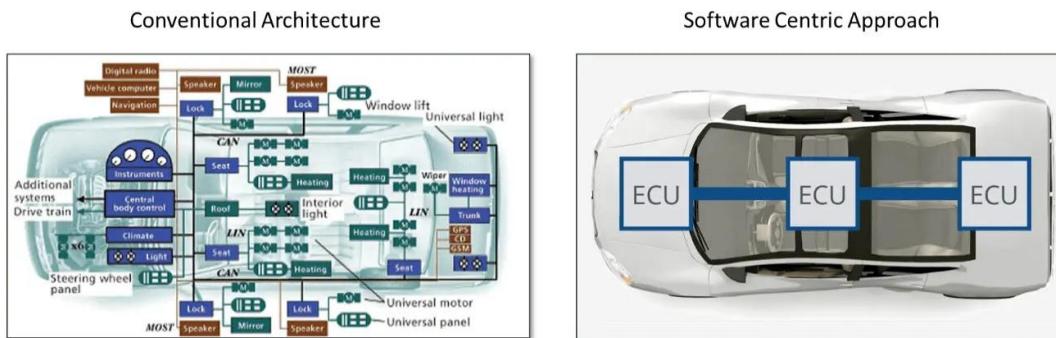
Chương 5: Kết luận và hướng phát triển:

Kết luận đề tài đối với những kết quả đạt được, nêu mặt còn hạn chế từ đó đề xuất hướng phát triển cho hệ thống.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 TỔNG QUAN VỀ HỆ THỐNG IOTS, NHÚNG, GIAO THỨC CAN VÀ CHUẨN GIAO THỨC J1939 TRÊN XE CƠ GIỚI

Hiện nay với sự phát triển của các chuẩn truyền thông không dây điển hình nhất chính là Internet thì việc truyền gửi dữ liệu từ một nơi rất xa và nhờ những tính năng truyền gửi như thế nên IoTs hay là kết nối vạn vật thông qua Internet đang thu hút cũng như là bùng nổ được ứng dụng vào các thiết bị công nghệ. Mục đích của IoTs là giúp các thiết bị có thể thu thập dữ liệu và truyền tải thông tin một cách tự động và thông minh. Điều này có thể giúp cho việc giám sát và điều khiển các quá trình trong nhà máy, hệ thống điều khiển, chiếu sáng, an ninh và nhiều ứng dụng khác. Đối với ngành ô tô (Automotive) cũng vậy thì việc ứng dụng thêm một thiết bị có thể truyền gửi các thông số của xe lên các ứng dụng di động hay website từ đó người dùng có thể theo dõi, quan sát được trình trạng xe từ rất xa để có thể dễ dàng quản lý hơn.



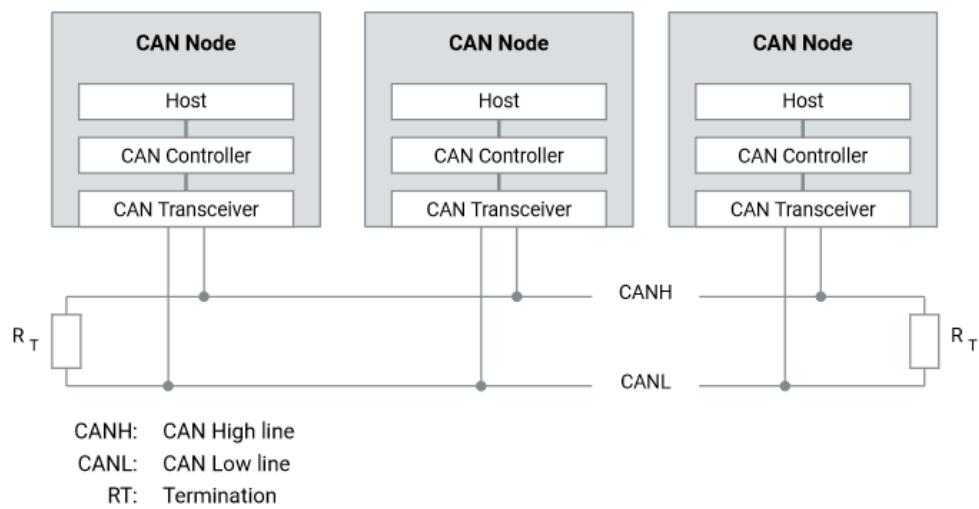
Hình 2.1: Tổng quan hệ thống nhúng trên xe ô tô

Tương tự như IoTs thì trong ngành ô tô hệ thống nhúng là một phần quan trọng của các bộ phận điện tử trên xe ô tô. Chúng được thiết kế để điều khiển và giám sát các hoạt động của xe. Các hệ thống này có thể được lắp đặt trực tiếp trên các bộ phận của xe ô tô hoặc trên một bo mạch điện tử riêng biệt, được kết nối với nhau để phối hợp giúp xe hoạt động hiệu quả hơn và đáp ứng các yêu cầu của người lái và hành khách hướng tới mục đích là tối ưu hóa hiệu suất và giảm thiểu sự cố hệ thống. Các thành phần của hệ thống nhúng gồm các vi điều khiển, ram, các thiết bị ngoại vi cũng như là các cảm biến và actuator.

Một ví dụ điển hình về hệ thống nhúng trên xe ô tô là giao thức CAN (Controller Area Network). Đây là một giao thức truyền thông phổ biến trong hệ thống điện tử trên xe ô tô, được sử dụng để kết nối liên kết các bộ phận điện tử và truyền tín hiệu giữa chúng một cách đáng tin cậy và nhanh chóng.

CAN Communication

CAN Network



Hình 2.2: Giao thức CAN

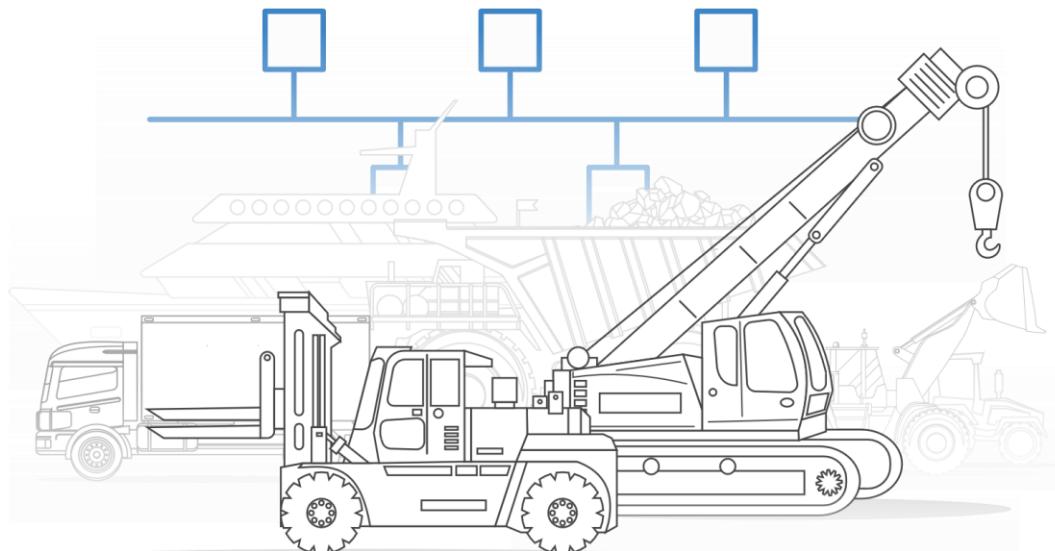
Giao thức này được phát triển bởi Bosch vào những năm 1980, là một công nghệ tiên tiến trong việc kết nối các thiết bị điện tử trên xe ô tô. Giao thức CAN sử dụng một mạng lưới các nút kết nối với nhau để có thể truyền tín hiệu. Các nút này có thể truyền dữ liệu đến các nút khác trên mạng bằng cách sử dụng một giao thức truyền thông đặc biệt bằng cách sử dụng 2 dây CAN HIGH và CAN LOW, ngoài ra còn sử dụng 2 điện trở termination để tránh cho việc conflict trong mạng.

Giao thức CAN được đánh giá cao về khả năng xử lý lỗi trong quá trình truyền dữ liệu. Khi dữ liệu bị mất hoặc bị hư hỏng trong quá trình truyền tải, giao thức này cho phép các nút khác trên mạng phát hiện và sửa chữa lỗi đó. Điều này giúp đảm bảo tính tin cậy và độ ổn định cao cho các hệ thống điện tử trên xe ô tô. Không chỉ áp dụng rộng rãi trong

ngành ô tô, giao thức CAN còn được ứng dụng vào các hệ thống điện tử công nghiệp, thiết bị y tế, và các phương tiện bay. Đối với định dạng về một frame của CAN sẽ được giới thiệu chung với giao thức J1939 dưới đây do giao thức này cũng được phát triển dựa trên CAN.

2.2 GIỚI THIỆU VỀ CHUẨN GIAO THỨC J1939

J1939 là một chuẩn giao thức truyền thông phổ biến trong ngành công nghiệp ô tô và các phương tiện di chuyển khác, được phát triển bởi Hiệp hội Kỹ sư ô tô Hoa Kỳ (SAE International). Chuẩn này giúp các nhà sản xuất xe ô tô khác nhau kết nối các bộ phận điện tử và truyền tín hiệu giữa chúng một cách tương thích, đồng thời giúp giảm chi phí phát triển sản xuất các hệ thống điện tử.

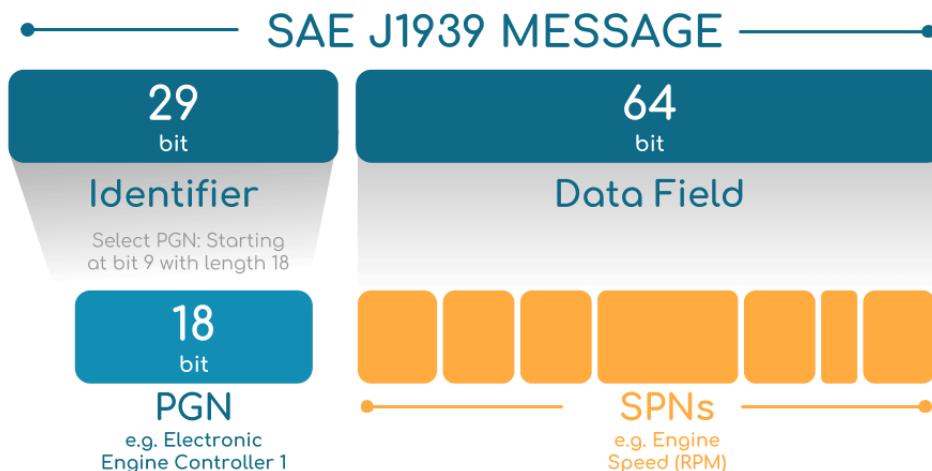


Hình 2.3: Giao thức J1939

J1939 đóng vai trò quan trọng trong việc truyền tải các thông tin về động cơ, hộp số, phanh, hệ thống đánh lái, hệ thống giám sát và các thông tin khác giữa các nút trên mạng. Các thông tin này được đóng gói trong các tin nhắn có định dạng chuẩn và được truyền tải trên các kênh truyền thông riêng biệt. Mỗi tin nhắn được định danh bằng một địa chỉ ID riêng, giúp các nút khác trên mạng có thể xác định và phân tích nội dung của tin

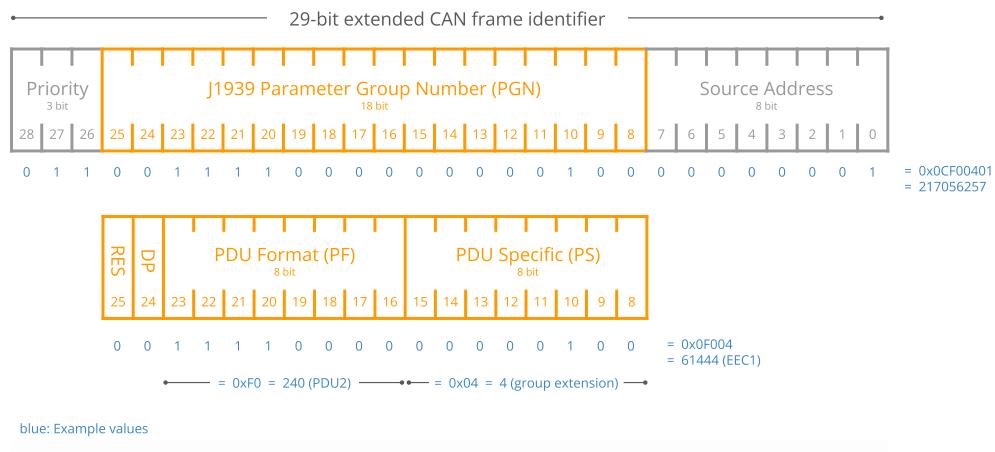
nhắn. J1939 được sử dụng rộng rãi trong ngành công nghiệp ô tô và các phương tiện di chuyển khác như xe tải, xe buýt, máy công trình, tàu thủy và máy bay. Đóng vai trò quan trọng trong việc giúp các hệ thống điện tử trên các phương tiện này hoạt động một cách hiệu quả và đảm bảo tính an toàn cho người sử dụng.

J1939 sử dụng giao thức truyền thông CAN (Controller Area Network) để truyền thông giữa các thiết bị trên mạng. Vì vậy, các khung tin truyền thông (message frames) của J1939 cũng tuân theo định dạng khung tin truyền thông CAN.



Hình 2.4: Một khung truyền cơ bản CAN, J1939

Với khung truyền của J1939 và CAN về cơ bản sẽ bao gồm hai phần chính, một sẽ là Identifier (ID) để có thể nhận diện cũng như biết được giá trị của khung đó đang là gì, tiếp đến sẽ là Data Field đây là vùng sẽ chứa các giá trị từ đó có thể chuyển đổi qua một số vật lý nhất định cho người dùng có thể đọc được. Hiện nay về khung truyền có thể mở rộng lên đến 29 bit đối với vùng Identifier, có thể biết đây là khung Extended ID đối với CAN và J1939.

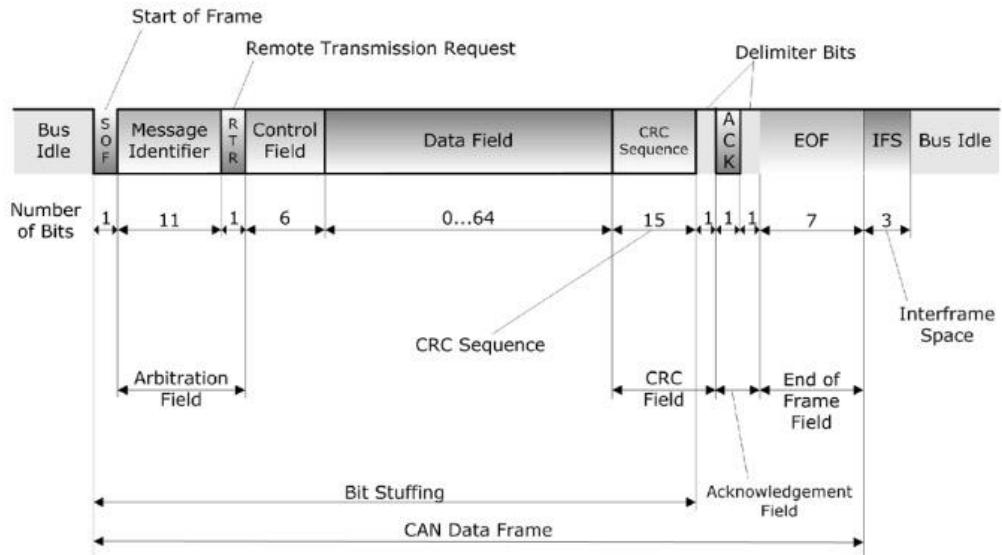


Hình 2.5: Chi tiết khung truyền Identifier trong J1939

Một khung tin truyền thông CAN của J1939 bao gồm 4 phần chính: khung tiêu đề (header) hay là Arbitration Field, trường dữ liệu (data field), khung điều khiển (control field) và khung kết thúc (end of frame).

Khung tiêu đề (header): Đây là phần đầu tiên của khung tin truyền thông sẽ bao gồm các phần về Identifier của khung truyền, bao gồm các trường sau:

- + Priority: Xác định mức độ ưu tiên của khung tin truyền thông. Trong J1939, giá trị priority được sử dụng để xác định ưu tiên truyền thông giữa các thông điệp khác nhau. Các thông điệp cần truyền thông quan trọng hơn sẽ có giá trị priority cao hơn.
- + PDU format (Protocol Data Unit format): Xác định định dạng PDU được sử dụng trong khung tin truyền thông.
- + Source address: Xác định địa chỉ của thiết bị gửi thông điệp.



Hình 2.6: Chi tiết khung truyền CAN, J1939

Khung điều khiển (Control field): Đây là phần đặc biệt của J1939, bao gồm các trường sau:

- + Data page (trang dữ liệu): Xác định trang dữ liệu được sử dụng trong trường dữ liệu.
- + Number of packets (số lượng gói tin): Xác định số lượng gói tin được sử dụng để chuyển đổi trường dữ liệu.
- + Packet sequence number (số thứ tự gói tin): Xác định số thứ tự của gói tin trong chuỗi gói tin.
- + End of message (kết thúc tin nhắn): Xác định kết thúc tin nhắn.

Trường dữ liệu (Data field): Đây là phần chứa thông tin cần truyền thông. Các trường dữ liệu của J1939 có thể bao gồm các thông số như tốc độ động cơ, nhiên liệu còn lại, áp suất lốp, nhiệt độ nước làm mát.

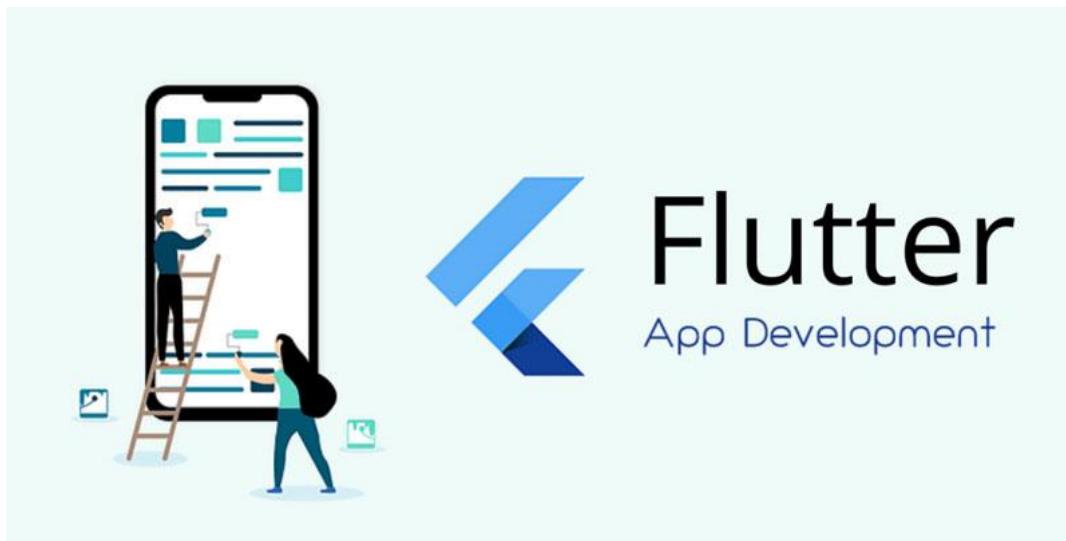
Khung CRC (Cyclic Redundancy Check): đây là khung để kiểm tra lỗi.

Khung kết thúc (End of frame): Đây là phần kết thúc của khung tin truyền thông, bao gồm một bit CRC để kiểm tra lỗi và các bit tạm dừng (interframe space) để tạo khoảng thời gian giữa các khung tin truyền thông.

2.3 GIỚI THIỆU FRAMEWORK SỬ DỤNG TRONG HỆ THỐNG

2.3.1 Flutter

Flutter là một mobile UI framework được Google phát triển nhằm giải quyết các vấn đề thường gặp trong lập trình ứng dụng cho di động là Fast Development cũng như Native Development. Lý do Flutter được sử dụng nhiều trong việc lập trình di động là do nền tảng này sẽ cho phép bạn lập trình, tạo ra một ứng dụng di động dựa vào một codebase trên Android và iOS với một ngôn ngữ lập trình cũng như là một nền mã duy nhất.



Hình 2.7: Framework Flutter ứng dụng lập trình di động

Đối với Flutter việc sử dụng kết hợp Fast Development, Native Development, Expressive và Flexible UI đã thu hút cũng như tạo sự tiện lợi đối với người dùng nên framework này đã và đang được áp dụng nhiều vào việc thiết kế ứng dụng điện thoại:

+ Fast Development: Là tính năng cho phép người lập trình áp dụng Hot Reload hoạt động trong milliseconds để hiển thị giao diện, cũng như fix bug, lỗi tiết kiệm thời gian hơn. Các widget đa dạng có thể customize để xây dựng giao diện nhanh chóng.

- + Native Development: Tính năng này sẽ giúp các widget có trong Flutter kEEP nỐI với nhau dù có sự khác biệt trong các nền tảng như navigation, icons, font từ đó sẽ cho hiệu năng tối ưu và tối nhất cho cả nền tảng iOS và Android.
- + Expressive và Flexible UI: Đối với Flutter việc kết hợp các thành phần khác nhau để xây dựng nên một giao diện vô cùng đẹp mắt là chuyện dễ dàng thông qua sử dụng các Smooth scrolling, Material Design Cupertino và các APIs.



Hình 2.8: Ngôn ngữ lập trình Dart trong Flutter

Flutter sử dụng ngôn ngữ lập trình Dart đây là ngôn ngữ do chính Google tạo ra. Dart được tạo ra nhằm tối ưu hóa việc xây dựng một UI, có thể hiểu rằng đây là ngôn ngữ được Google sinh ra nhằm để phục vụ trong Flutter với nhiều điểm mạnh mà ngôn ngữ này đem lại. Flutter được phát triển vào năm 2018, tập trung chủ yếu hỗ trợ phát triển ứng dụng di động, nhưng cho đến nay Google đã hỗ trợ và kết hợp với các cộng đồng nguồn mở như Reddit, Stack Overflow, Github cũng như phát triển đa nền tảng giúp người dùng có thể dễ dàng tiếp cận, Flutter đã có thể phát triển áp dụng trên cả sáu nền tảng: iOS, Android, MacOS, web, Windows và Linux.

2.3.2 NodeJS

NodeJS được biết tới là một nền tảng xây dựng trên Node runtime hay Javascript runtime sẽ gồm các thứ cần phải có và hoàn thành trong một chương trình chạy bằng ngôn ngữ Javascript.

Nền tảng Javascript runtime được biết đến là “V8 Javascript engine” được viết bằng c++ và Javascript, có thể thấy tên V8 dựa trên tên của một loại động cơ được gắn trên xe nhằm biểu thị cho việc xử lý mạnh mẽ, tốc độ xử lý nhanh, realtime.



Hình 2.9: Nền tảng Platform NodeJS

NodeJS được ứng dụng nhiều trong việc phát triển các ứng dụng chat online, các trang phát video trực tiếp, các trang web khác nhau có thể chạy trên nhiều nền tảng cũng như hệ điều hành khác nhau.

Dựa vào Hình 2.9 có thể thấy được các ứng dụng nên được viết bằng NodeJS:

- + Websocket Server: Tạo một máy chủ Server vận chuyển các dữ liệu.
- + Rest API: Dựa vào API có thể cho các ứng dụng khác kết nối sử dụng.
- + Real-time data: các ứng dụng yêu cầu về thời gian thực thì NodeJS có thể đáp ứng dễ dàng.

+ Cloud Services: Tạo một máy chủ cloud.

Ngoài ra trong NodeJS sẽ có rất nhiều các framework nhỏ hỗ trợ cho các công việc, yêu cầu khác nhau đa dạng về hình thức. Trong NodeJS sử dụng một thư viện npm (Node Package Manager) đây là thư viện được tạo bởi cộng đồng những người lập trình nhằm giải quyết các vấn đề về lỗi, bug fix, cũng như tạo các package có sẵn giúp bạn dễ dàng lập trình và UI đẹp mắt.



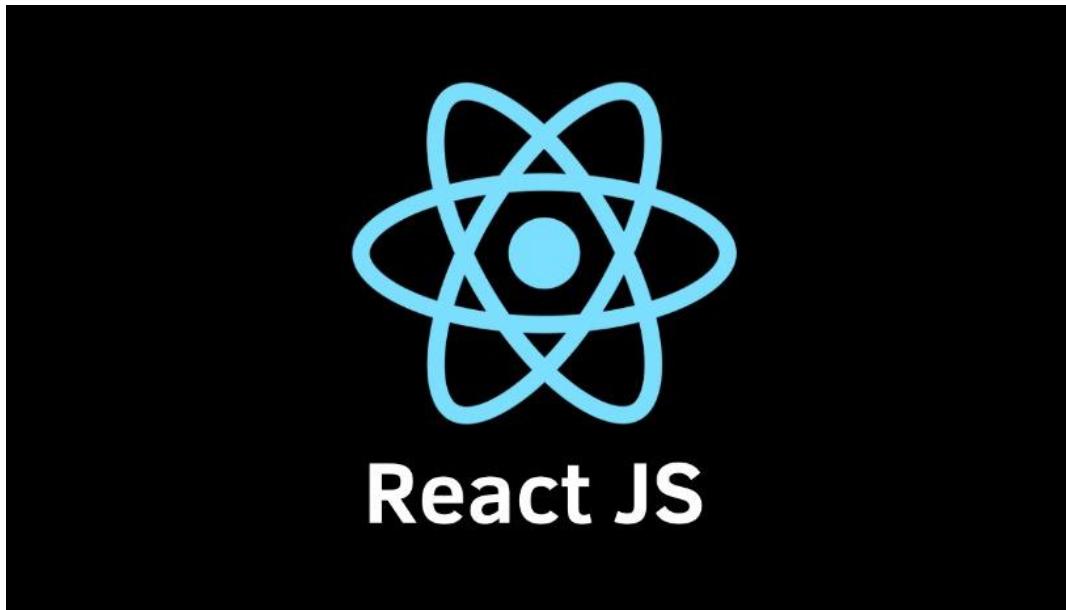
Hình 2.10: ExpressJS framework phổ biến trong NodeJS

Một framework được sử dụng rộng rãi trong NodeJS nhất chính là ExpressJS đây là một framework động giúp cung cấp sự linh hoạt cho người dùng có thể thỏa thích phát triển giao diện của ứng dụng. Framework này sẽ hỗ trợ các plugin và các tiện ích mở rộng khác nhau làm tăng thêm tính đa dạng trong NodeJS, với hơn 14 công cụ mẫu và hiệu suất tuyệt vời nên ExpressJS đang là framework được ứng dụng và sử dụng nhiều hiện nay.

2.3.3 ReactJS

ReactJS là một thư viện với mã nguồn mở sử dụng ngôn ngữ Javascript, được Facebook sáng tạo và phát triển năm 2013, cho đến nay việc sử dụng ReactJS vẫn đang rất rộng rãi và ứng dụng nhiều đối với các người dùng phát triển Javascript front-end. Việc sử dụng ReactJS sẽ đem lại hiệu quả cao với tốc độ load nhanh, khả năng mở rộng cao và dễ

thực hiện. Người dùng có thể kết hợp và chuyển đổi các giao diện người dùng từ những cách phức tạp trở nên đơn giản thông qua các tính năng có trong ReactJS dễ dàng hơn.



Hình 2.11: ReactJS framework

Đối với việc lập trình trong một ứng dụng người dùng thì việc hướng tới UI và xử lý tương tác của người dùng là điều quan trọng nhất. Hiểu nôm na là UI sẽ là phần mà người dùng có thể xử lý tương tác, sau khi người dùng tương tác sẽ dẫn đến việc xử lý từ đó sẽ là xử lý tương tác đó.

ReactJS hỗ trợ xây dựng giao diện nhanh, phát triển cải thiện hiệu suất web cũng như là hạn chế các lỗi trong quá trình lập trình. ReactJS có thể phù hợp đa dạng các thể loại website từ đó khiến việc khởi tạo website dễ dàng hơn. Có thể sử dụng lại các Component flexible đáp ứng được các yêu cầu của nhiều dự án khác nhau. Tương tự với các tính năng nổi bật của Flutter về Hot Reload trong ReactJS cũng sẽ có tích hợp tính năng này nhằm giúp người dùng lập trình nhanh và tốt hơn.

Đối với ReactJS cũng có thể sử dụng thư viện được cộng đồng lập trình cùng nhau phát triển npm như NodeJS nên việc có thể sử dụng ReactJS giao tiếp với các thư viện, package cũng như các framework khác.

2.4 GIỚI THIỆU VỀ CƠ SỞ DỮ LIỆU

Cơ sở dữ liệu là một hệ thống điện tử với mục đích chính là lưu trữ các loại dữ liệu có thể là hình ảnh, số liệu, ký tự, và có thể được truy suất để có thể sử dụng. Có hai dạng lưu trữ dữ liệu cơ bản là SQL và NoSQL.

Bảng 2.1: So sánh giữa SQL và NoSQL

Đối số	SQL	NoSQL
Ngôn ngữ lập trình	Ngôn ngữ SQL (Structured Query Language.)	Không sử dụng ngôn ngữ SQL
Kiểu thiết kế	Dạng bảng, với các trường dữ liệu được định dạng rõ ràng.	Dạng object, với các trường thông tin có thể lồng vào nhau.
Đối tượng sử dụng	Thích hợp với dạng lưu trữ không phân cấp	Thích hợp với dạng lưu trữ có phân cấp.

Cơ sở dữ liệu phpMyAdmin



Hình 2.12: Biểu tượng của cơ sở dữ liệu phpMyAdmin

PhpMyAdmin là một phần mềm mã nguồn mở dành cho việc lưu trữ dữ liệu dùng cho lập trình web dựa trên ngôn ngữ PHP. Giúp các nhà phát triển phần mềm theo cách dễ dàng hơn trong việc lưu trữ cũng như trực quan hóa dữ liệu của mình. phpMyAdmin hỗ

trợ các cơ sở dữ liệu dạng NoSQL, được lưu và đồng bộ dữ liệu trên cloud, dữ liệu sẽ được đồng bộ trên tất cả hệ thống người dùng.

2.4.1 Các chứng năng chính của phpMyAdmin.

1. Tạo cơ sở dữ liệu và bảng dữ liệu:

phpMyAdmin cho phép chúng ta tạo cơ sở dữ liệu và các bảng dữ liệu, tùy thuộc từng loại yêu cầu và dịch vụ thì phpMyAdmin sẽ cung cấp một khoảng dung lượng lưu trữ tương ứng.

2. Cho phép truy cập từ nhiều thiết bị:

phpMyAdmin cho phép truy cập từ nhiều thiết bị khác nhu cung nhu từ đa dạng các loại ngôn ngữ lập trình phổ biến hiện nay, tuy nhiên phpMyAdmin có các cơ chế xác minh quyền truy cập của các người dùng, tối ưu về mặt bảo mật thông tin cũng như tính toàn vẹn dữ liệu cho người dùng, sau khi người dùng đã được xác minh, các thiết bị truy cập có thể truy vấn và sử dụng các dữ liệu đã được lưu trữ từ trước.

3. Tự động lưu trữ khi không có mạng:

Khi hoạt động ở trạng thái không có kết nối internet (offline) dữ liệu sẽ được lưu vào bộ nhớ cache của dữ liệu (mặc định ở host localhost:/port80)

4. Không gian làm việc của phpMyAdmin:

The screenshot shows the phpMyAdmin interface with the following elements highlighted:

- Left sidebar:** Shows the database structure for 'sql12613711'. A specific table, 'Engine_fuel_information', is selected and highlighted with a blue border. Number 1 is placed near this selection.
- Central workspace:** Displays a list of tables under the 'Bộ lọc' (Filter) section. The table 'Engine_fuel_information' is listed here. Number 2 is placed near the table name.
- Table list area:** Shows a detailed list of tables with columns for 'Hạng' (Status), 'Kiểu' (Type), 'Bảng mã đối chiếu' (Checklist table), 'Kích thước' (Size), and 'Tổng chi phí' (Total cost). Number 3 is placed near the table names.
- Bottom navigation:** Includes buttons for 'Theo dõi bảng' (Follow table) and 'Lưu mục đã chọn' (Save selected items).
- Bottom right:** A 'Thực hiện' (Execute) button.
- Bottom center:** A 'Tạo bảng' (Create table) dialog box is open, showing fields for 'Tên:' (Name:) and 'Số cột:' (Number of columns: 4). Number 5 is placed near this dialog.

Hình 2.13: Trang chủ của phpMyAdmin

PhpMyAdmin cho phép ta làm việc với một không gian làm việc tiện lợi, giúp người dùng có cái nhìn trực quan về cơ sở dữ liệu của mình. Không gian làm việc của phpMyAdmin gồm các phần chính:

1. Nơi lưu trữ các cơ sở dữ liệu hiện đang có của người dùng, thông tin gồm tên cơ sở dữ liệu, các bảng bên trong cơ sở dữ liệu ấy.
2. Chứa tên bảng Khu vực thể hiện chi tiết thông tin của một cơ sở dữ liệu cụ thể hiện ra khi ta chọn vào một cơ sở dữ liệu cụ thể.
3. Vùng gồm các thao tác mà ta có thể thực hiện với các bảng bao gồm các hành động: Thêm, xóa, sửa, hiển thị thông tin bảng trống, người dùng có thể điều chỉnh bảng tại đây mà không cần thông qua các thao tác với bảng
4. Nơi chứa thông tin máy chủ hiện tại đang lưu dữ liệu.
5. Khu vực cho phép người dùng tạo bảng mới

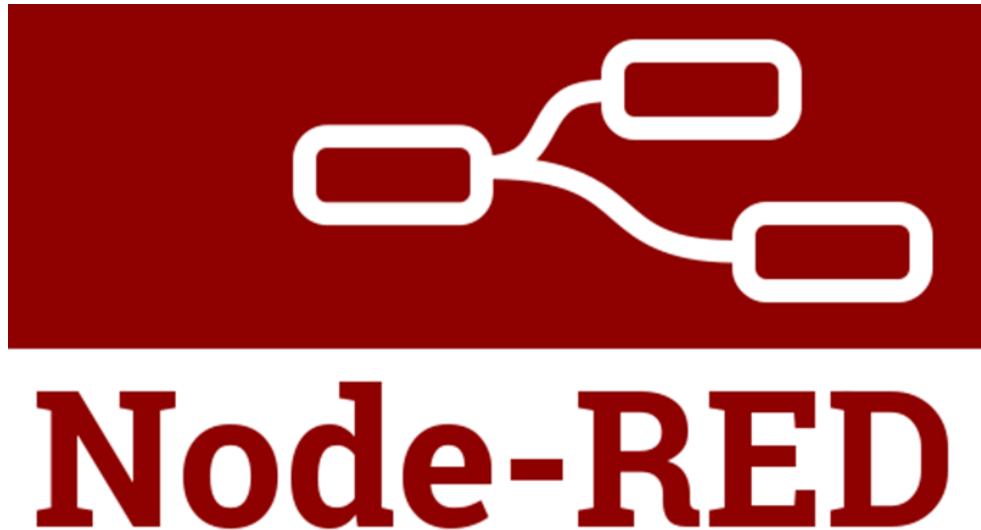
	Sửa	Chép	Xóa bỏ	ID	Fuel	Date	Time
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	1	24	22/4/2023	16:19:37
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	2	38	22/4/2023	16:38:41
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	3	50	22/4/2023	16:39:8
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	4	49	22/4/2023	16:39:13
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	5	68	22/4/2023	16:39:18
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	6	46	22/4/2023	16:39:23
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	7	99	22/4/2023	16:39:28
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	8	94	22/4/2023	16:39:33
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	9	23	22/4/2023	16:39:38
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	10	6	22/4/2023	16:39:43
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	11	37	22/4/2023	16:39:48
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	12	21	22/4/2023	16:39:53
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	13	55	22/4/2023	16:39:58
<input type="checkbox"/>	Sửa	Chép	Xóa bỏ	14	8	22/4/2023	16:40:3

Hình 2.14: Bảng chứa thông tin dữ liệu

Khi nhấn chọn bảng cụ thể trong trường hợp người dùng muốn xem dữ liệu đã lưu trữ trong bảng, không gian lưu trữ dữ liệu sẽ hiện ra, không gian lưu trữ dữ liệu gồm hai vùng chính:

1. Khu vực thêm, sửa, xóa các dữ liệu trong bảng, thao tác trực tiếp với từng hàng dữ liệu cụ thể, đây chính là cách giúp người dùng hình dung hóa tốt nhất về hệ thống.
2. Khu vực hiển thị dữ liệu, dữ liệu sẽ được hiển thị thành từng hàng với mỗi lần cập nhật là một hàng riêng biệt.

2.5 PHẦN MỀM NODE RED.



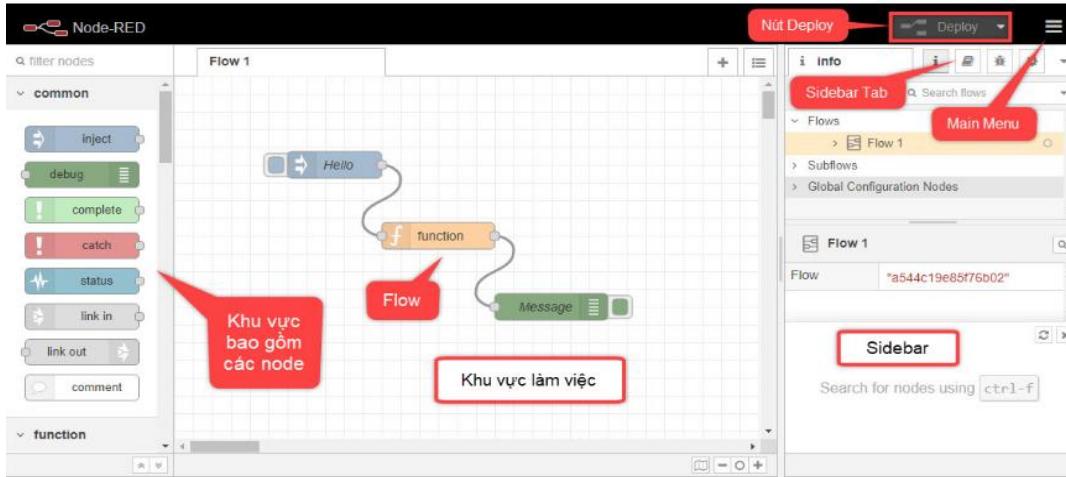
Hình 2.15: Logo phần mềm node-red

Node red là một công cụ phần mềm kéo thả hỗ trợ kết nối phần cứng với các API và webserver của người dùng. Node red được thiết kế dựa trên ngôn ngữ lập trình NodeJS, Node red cung cấp một trình soạn thảo với các thao tác kéo thả các Node người dùng có thể thiết kế và kết nối phần cứng với phần mềm một cách dễ dàng.

2.5.1 Các tính năng của node red.

1. Hỗ trợ môi trường thời gian thực, quản lý các luồng hoạt động.
2. Thiết kế tính năng các Node đa dạng, đáp ứng nhiều nhu cầu sử dụng của người dùng.
3. Được hỗ trợ hầu hết các thiết bị phần cứng Raspberry Pi, Beagle Bone Black, Arduino, các thiết bị Android, v.v.
4. Có thể chạy trên môi trường đám mây IBM Cloud, AWS, Microsoft Azure, v.v.

2.5.2 Vùng làm việc của node-red



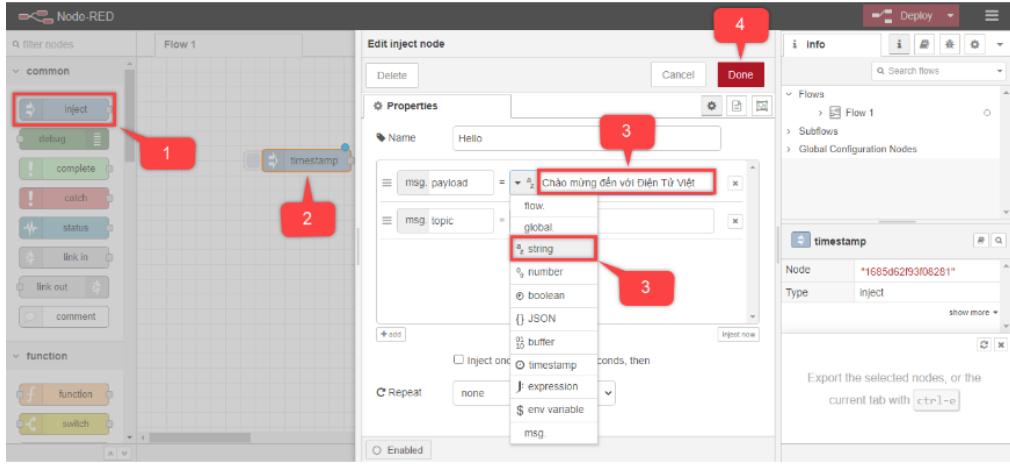
Hình 2.16: Các vùng làm việc chính của node-red

Node-red cung cấp một không gian làm việc được trực quan hóa cao với mức độ tương tác được tối ưu. Không gian làm việc của node-red gồm các phần chính:

1. Tiêu đề ở trên cùng, chứa nút Deploy và menu chính.
2. Bảng màu (palette) bên trái, nơi lưu trữ các node được sử dụng trong quá trình thiết kế, các node này có thể tải thêm trên thư viện của node-red.
3. Khu vực làm việc chính (workspace) ở giữa, nơi các flow được tạo ra, khu vực này có thể tự điều chỉnh kích thước tùy theo nhu cầu sử dụng của người dùng.
4. Thanh sidebar bên phải, nơi chứa các thông tin của chung của toàn bộ flow, thường được để ở chế độ xem các lỗi phát sinh trong quá trình thiết kế.

2.5.3 Các thiết lập một node trong node-red.

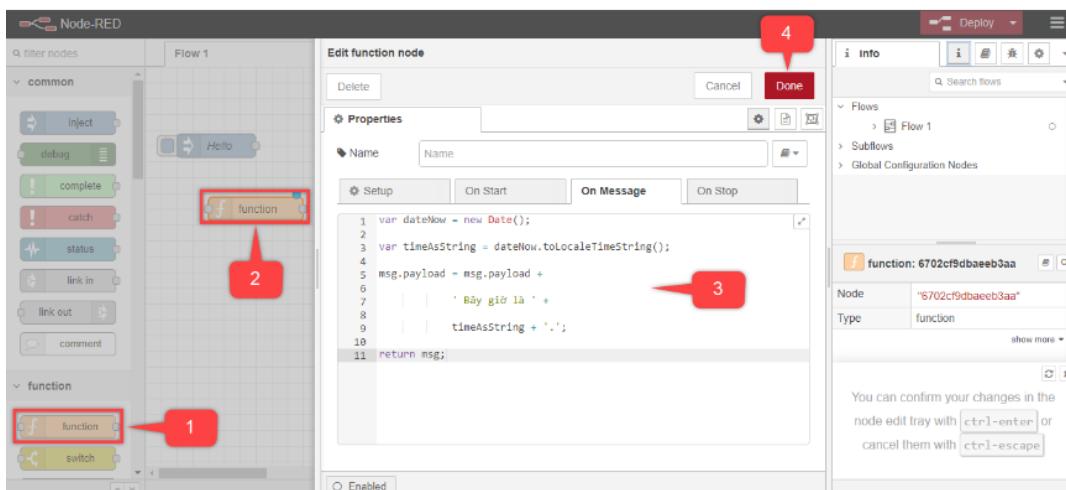
Node là một đơn vị lập trình trong node-red, với mỗi node thực hiện một chức năng cụ thể đã được định trước, các node có thể có ngõ vào và ngõ ra, cũng có thể chỉ có ngõ vào hoặc ngõ ra tùy theo nhu cầu sử dụng của mỗi người dùng, do đó mỗi node đều có cách thiết lập các thông số riêng.



Hình 2.17: Định dạng cơ bản của một node

Ta có thể thấy trong Hình 2.17 là định dạng một node của node-red gồm có các thành phần chính:

1. Ký hiệu, biểu tượng của node nằm trên thanh công cụ palette bên trái vùng làm việc.
2. Node được triển khai khi người dùng kéo node vào khu vực làm việc (workspace), tại đây khi nhấn chọn node bảng thiết lập của node sẽ hiện lên
3. Các thành phần có thể có trong bảng thiết lập của một node, trong hình minh họa trên người hướng dẫn đang định dạng node chứa một thông điệp dạng “String”
4. Sau khi hoàn thành tất cả các thao tác của thiết lập, người dùng chọn “Done” để lưu các thông tin vừa thiết lập trong node.

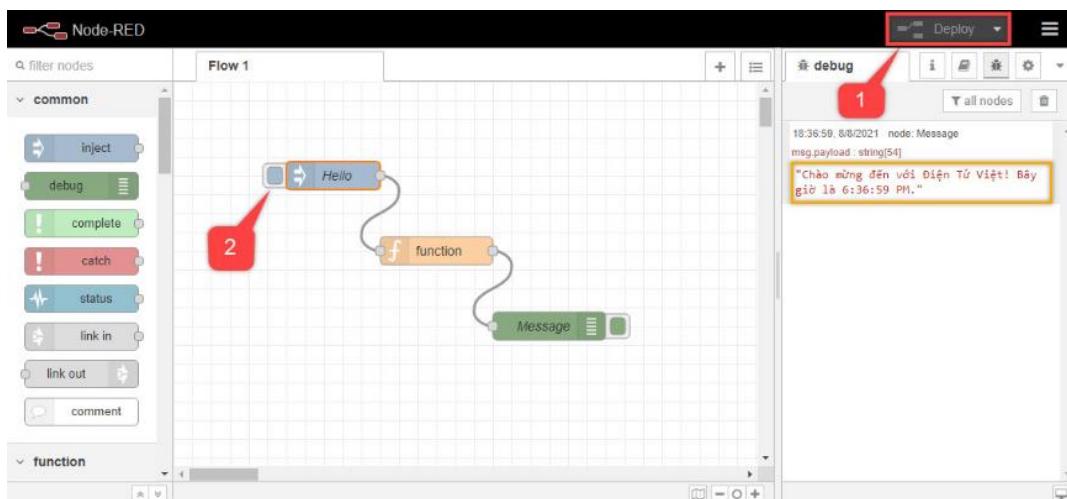


Hình 2.18: Node chứa hàm của người lập trình.

Ngoài lựa chọn thiết lập cứng, node-red còn cung cấp cho người dùng các node chứa các hàm, đây là nơi người dùng có thể tùy chỉnh chức năng của chương trình theo các điều kiện cụ thể của riêng mình.

1. Ký hiệu, biểu tượng của node nằm trên thanh công cụ palette bên trái vùng làm việc.
2. Node được triển khai khi người dùng kéo node vào khu vực làm việc (workspace), tại đây khi nhấn chọn node bảng thiết lập của node sẽ hiện lên
3. Khu vực chứa toàn bộ nội dung code của người dùng, tất cả các code đều dựa trên ngôn ngữ NodeJS.
4. Sau khi hoàn thành tất cả các thao tác của thiết lập, người dùng chọn “Done” để lưu các thông tin vừa thiết lập trong node.

Sau khi người dùng xây dựng xong một hoạt động cụ thể có thể chọn lưu và chạy thử chương trình.



Hình 2.19: Hình minh họa kết quả thực thi một flow trên node-red

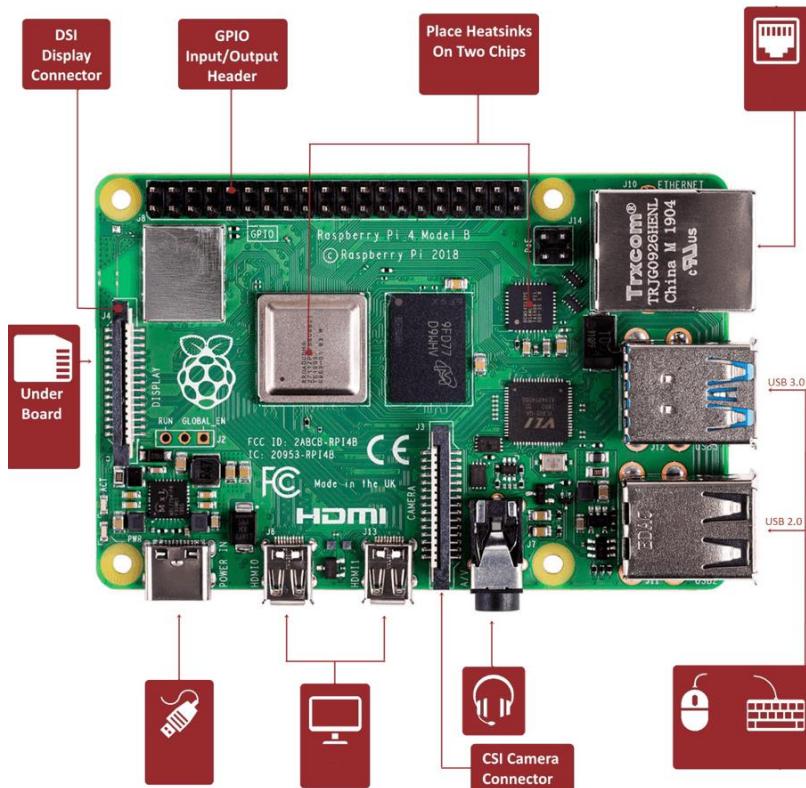
Sau khi chương trình được thực thi kết quả hiển thị sẽ nằm bên khung sidebar bên phải.

2.6 RASPBERRY PI 4B

Raspberry là một máy tính thu nhỏ với kích thước nhỏ tương đương với một điện thoại. Raspberry Pi 4 Model B là phiên bản mới nhất của dòng series Raspberry Pi giá rẻ.

Pi không giống như thiết bị thông thường nó không có vỏ bạn có thể tìm thấy bên trong PC hoặc máy tính xách tay, nhưng nhỏ hơn nhiều.

Bản chất của Raspberry Pi 4 là một máy tính hoàn chỉnh với CPU, RAM, Bl, các cổng kết nối khác. Để có thể vận hành raspberry pi4 ta cần cài đặt cho nó một hệ điều hành thông qua thẻ nhớ, hệ điều hành được sử dụng là Linux, Raspberry pi4 nói riêng và raspberry nói chung rất hữu ích trong các tác vụ như xây dựng một máy tính nhỏ để làm việc hoặc điều khiển các thiết bị trong gia đình và đặc biệt thích hợp trong các trường hợp xây dựng các hệ thống IoT.



Hình 2.20: Sơ đồ kết nối raspberry

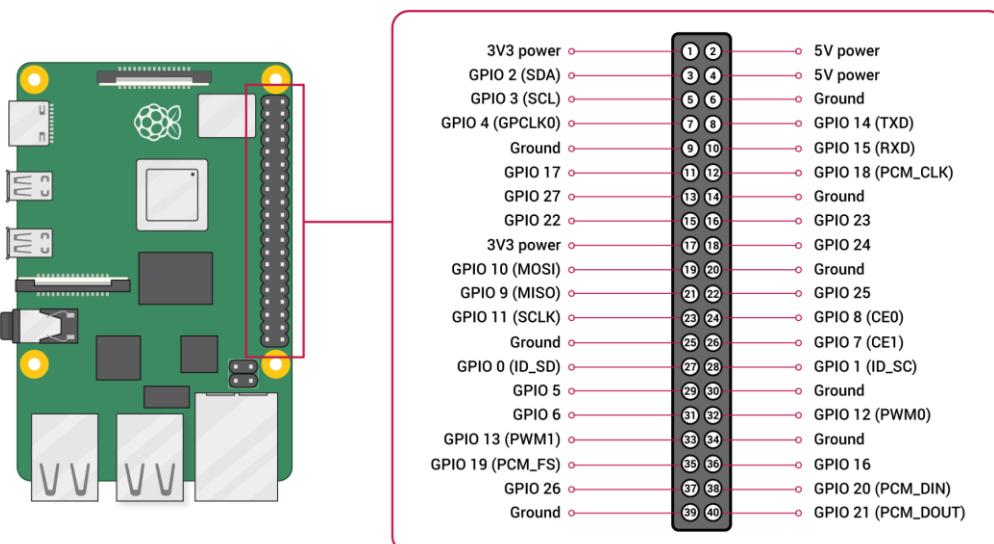
Trên Hình 2.20 Biểu thị toàn bộ các kết nối ngoại vi có thể kết nối với raspberry pi4 gồm: CPU, RAM, chân IO, khe cắm thẻ nhớ, đầu nối cáp mạng, dây nối nguồn, khe cắm CSI camera, hai cổng micro-HDMI để kết nối với màn hình, raspberry pi4 cho phép có thể

sử dụng song song hai màn hình, đầu cắm các thiết bị âm thanh, hai cổng USB và hai cổng USB 2B.

Để có thể kết nối nhiều thiết bị điện tử raspberry pi4 cung cấp cho người dùng 30 chân IO phục vụ cho việc tự thiết kế của người dùng.

Bảng 2.2: Thông số kỹ thuật của module

Nguồn tiêu chuẩn	5V
Dòng hoạt động bình thường	1.1A
Bộ nhớ	1GB, 2GB, 4GB hoặc 8GB
Nhiệt độ hoạt động	0 ~ 50°C
Bộ xử lý	SoC lõi Cortex-A72 (ARM v8) 64 bit @ 1.5GHz
Công suất tiêu thụ	5.5mW
Kích thước	39*25.5mm



Hình 2.21: Sơ đồ thứ tự các chân của raspberry pi4

Bảng 2.3: Bảng thông tin các chân của Raspberry pi4

STT	Loại chân	GPIO
1	PWM	GPIO12, GPIO13, GPIO18, GPIO19
2	SPI	SPI0: GPIO9 (MISO), GPIO10 (MOSI), GPIO11 (SCLK), GPIO8 (CE0), GPIO7 (CE1)
		SPI1: GPIO19 (MISO), GPIO20 (MOSI), GPIO21 (SCLK), GPIO18 (CE0), GPIO17 (CE1), GPIO16 (CE2)
3	I2C	Data: GPIO2 Clock: GPIO3 EEPROM Data: GPIO0 EEPROM Clock: GPIO1
4	UART	TX: GPIO14 RX: GPIO15
5	VCC	GPIO2, GPIO4
6	GND	GPIO14, GPIO5, GPIO30, GPIO34, GPIO39, GPIO25, GPIO9.

2.7 MODULE WAVESHARE CAN HAT RS485

Module RS485 CAN HAT được thiết kế nhằm mục đích cho phép Pi giao tiếp ổn định với các thiết bị khác trong khoảng cách xa thông qua các chức năng RS485/CAN. Module RS485 CAN HAT được phát triển từ module MCP2515 tích hợp thêm chuẩn giao tiếp SPI để giao tiếp với raspberry, từ đó giúp raspberry có thể nhận được dữ liệu từ CAN Bus.

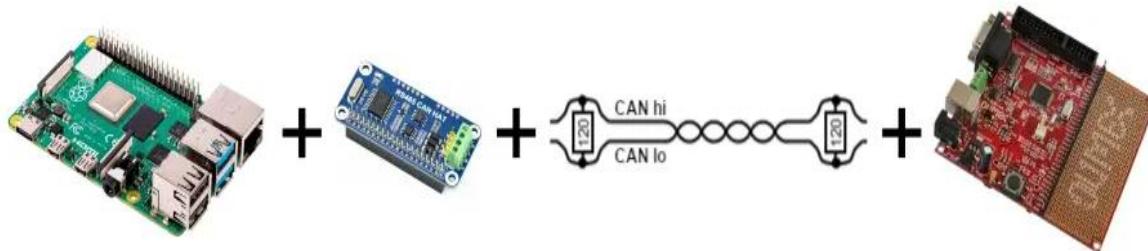


Hình 2.22: Module RS485 CAN HAT

Module được thiết kế nhằm đồng bộ hóa việc sử dụng với Raspberry vì vậy được thiết kế với 30 chân tín hiệu để người dùng dễ dàng kết nối với Raspberry.



Hình 2.23: Kết nối module RS485 CAN HAT và Raspberry pi4



Hình 2.24: Mô hình hoạt động của module CAN HAT RS485

Như đã trình bày chức năng chính của module CAN HAT RS485 được dùng để giúp raspberry có thể truyền nhận dữ liệu thông qua CAN

Bảng 2.4: Thông số kỹ thuật của module

Nguồn tiêu chuẩn	3.3V
Dòng hoạt động bình thường	10mA
Chuẩn giao tiếp	SPI
Công suất tiêu thụ	33mW
Kích thước	39*25.5mm

Bảng 2.5: Sơ đồ chân của module RS485 CAN HAT

STT	Tên chân	Chức năng
1	3V3	Cáp nguồn 3V3
2	GND	Nối đất
3	SCK	Tạo xung clock
4	MOSI	Ngõ vào liệu chuẩn SPI
5	MISO	Ngõ ra dữ liệu chuẩn SPI
6	CS	Chọn tín hiệu
7	INT	Ngắt
8	RX	Nhận tín hiệu chuẩn UART
9	TX	Truyền tín hiệu chuẩn UART
10	RSE	Điều khiển tín hiệu TX/RX (mặc định mức cao TX và mức thấp RX)

2.8 MODULE NEO-6M GPS

Module NEO-6M GPS có chức năng định vị trí theo hệ thống GPS của Hoa Kỳ, module cho kết quả chính xác cao và thời gian phản hồi tương đối ngắn, module thích hợp sử dụng trong các ứng dụng điện tử phổ thông.



Hình 2.25: Module NEO-6M

Module Neo-6M có 5 chân giao tiếp vi ngoại vi gồm: VCC, GND, TX, RX, PPS chỉ tiết được trình bày ở bảng...

Bảng 2.6: Ký hiệu chân module NEO-6M

STT	Tên chân	Chức năng
1	VCC	Cấp nguồn 5V
2	TX	Gửi tín hiệu UART
3	RX	Nhận tín hiệu UART
4	GND	Nối đất

Bảng 2.7: Thông số của module GPS-NEO-6M

Nguồn tiêu chuẩn	3.3-5.5V
Dòng hoạt động bình thường	50 mA
Chuẩn giao tiếp	UART/TTL
Tốc độ truyền nhận dữ liệu	Gồm nhiều mức khác nhau 1200, 2400, 4800, 19200, 38400, 9600 (mặc định), 57600, 115200
Kích thước	39*25.5mm

CHƯƠNG 3: THIẾT KẾ VÀ THI CÔNG

3.1 YÊU CẦU HỆ THỐNG

Hệ thống được thiết kế với những yêu cầu sau:

- Hệ thống có khả năng chuyển đổi các giá trị CAN thành số vật lý mà người dùng có thể đọc được thông qua chuẩn giao thức J1939.
- Thiết kế thành phần hệ thống Cloud có thể thu thập, lưu trữ, truyền dữ liệu và cung cấp API cho các thành phần khác trong hệ thống có thể giao tiếp với nhau.
- Thiết kế Front-end với hai giao diện là web và ứng dụng trên điện thoại Android cho người dùng có thể quan sát các thông số của xe về tốc độ xe, nhiệt độ động cơ, quãng đường xe đã đi được, mức nhiên liệu. Đảm bảo cập nhật các thông số liên tục và độ trễ khi truyền ít hơn 5 giây.

3.2 ĐẶC TẢ HỆ THỐNG

3.2.1 Chức năng của hệ thống

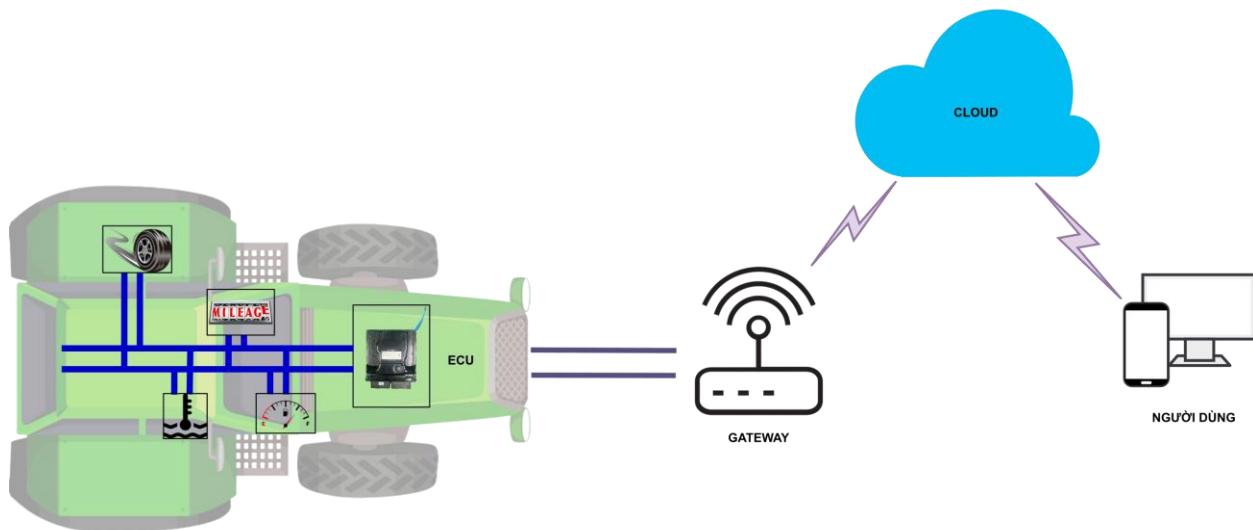
Dựa vào những yêu cầu hệ thống thì nhóm chúng tôi đã thiết kế và xây dựng hệ thống gồm những chức năng như sau:

- Hệ thống có chức năng đăng nhập, đăng xuất dựa trên tài khoản được cấp dành cho người dùng.
- Hệ thống có thể thu thập các thông số của xe như nhiệt độ động cơ, tốc độ xe, quãng đường đã đi được, mức nhiên liệu còn lại.
- Hệ thống có khả năng cảnh báo cho người dùng khi các thông số được thu thập vượt quá ngưỡng cho phép.
- Hệ thống hiển thị các thông số của xe hiển thị trên web và ứng dụng trên điện thoại Android.
- Hệ thống có định vị GPS giúp người dùng có thể quan sát vị trí hiện tại của phương tiện.

- Hệ thống hiển thị các thông số trên một màn hình riêng phòng trường hợp người dùng mắt mèo không thể vào web và ứng dụng điện thoại. Ngoài ra, người dùng có thể quan sát và theo dõi lại các dữ liệu liên tục như nhiệt độ động cơ, tốc độ xe, quãng đường đã đi được, mức nhiên liệu còn lại dựa trên bảng thống kê dữ liệu.

3.2.2 Mô hình tổng thể hệ thống

Mô hình hoạt động tổng quát của hệ thống giám sát và chẩn đoán trên xe:



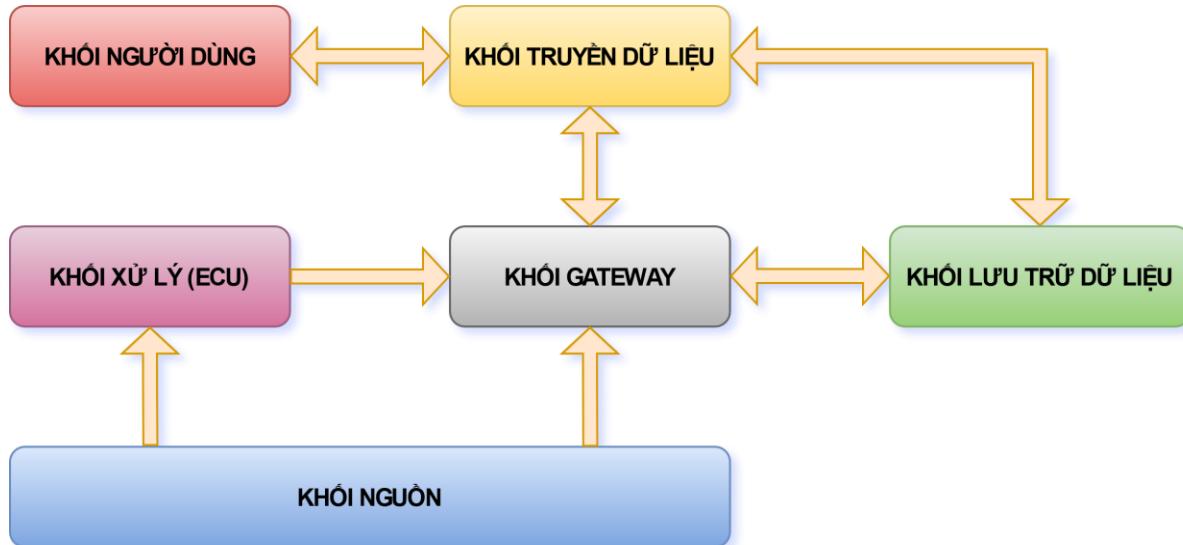
Hình 3.1: Mô hình tổng quát của hệ thống

Dựa trên những yêu cầu hệ thống thì nhóm đã thiết kế được mô hình tổng thể của toàn bộ hệ thống như sau:

- + **ECU:** thu thập và điều khiển các thông số được lấy từ xe như nhiệt độ động cơ, tốc độ xe, quãng đường đã đi được, mức nhiên liệu dưới dạng CAN cũng như là chuyển đổi và xử lý các thông số về theo chuẩn giao thức J1939.
- + **Gateway:** trung tâm thu thập, xử lý và chuyển đổi tất cả các dữ liệu thu được từ ECU và truyền các dữ liệu đó lên Cloud.
- + **Cloud:** xử lý, truyền và lưu trữ các dữ liệu nhận được từ Gateway, trung chuyển các dữ liệu đó về cho Người Dùng

+ **Người dùng**: cung cấp giao diện hệ thống, hiển thị các giá trị thông số thu thập từ ECU được gửi từ Cloud nhờ vào Gateway từ đó người dùng có thể theo dõi các thông số của xe.

3.2.3 Sơ đồ khái niệm hệ thống



Hình 3.2: Sơ đồ khái niệm tổng thể hệ thống

Khối nguồn: cấp nguồn từ máy cấp nguồn cho khối xử lý (ECU) và khối central gateway hoạt động.

Khối xử lý (ECU): sử dụng một thiết bị điều khiển điện tử (ECU) thực tế được hỗ trợ từ công ty Bosch dùng để đọc và chuyển đổi các thông số của xe thành giao thức CAN và truyền cho khối central gateway xử lý.

Khối central gateway: sử dụng vi xử lý để thực hiện thu thập các thông số từ khối xử lý (ECU) thông qua CAN và chuyển đổi thành một số vật lý như tốc độ xe, nhiệt độ động cơ, mức nhiên liệu hiện tại, quãng đường đã đi được dựa trên chuẩn giao thức J1939 cũng như là nhận và truyền các thông số tới khối lưu trữ dữ liệu và khối truyền dữ liệu.

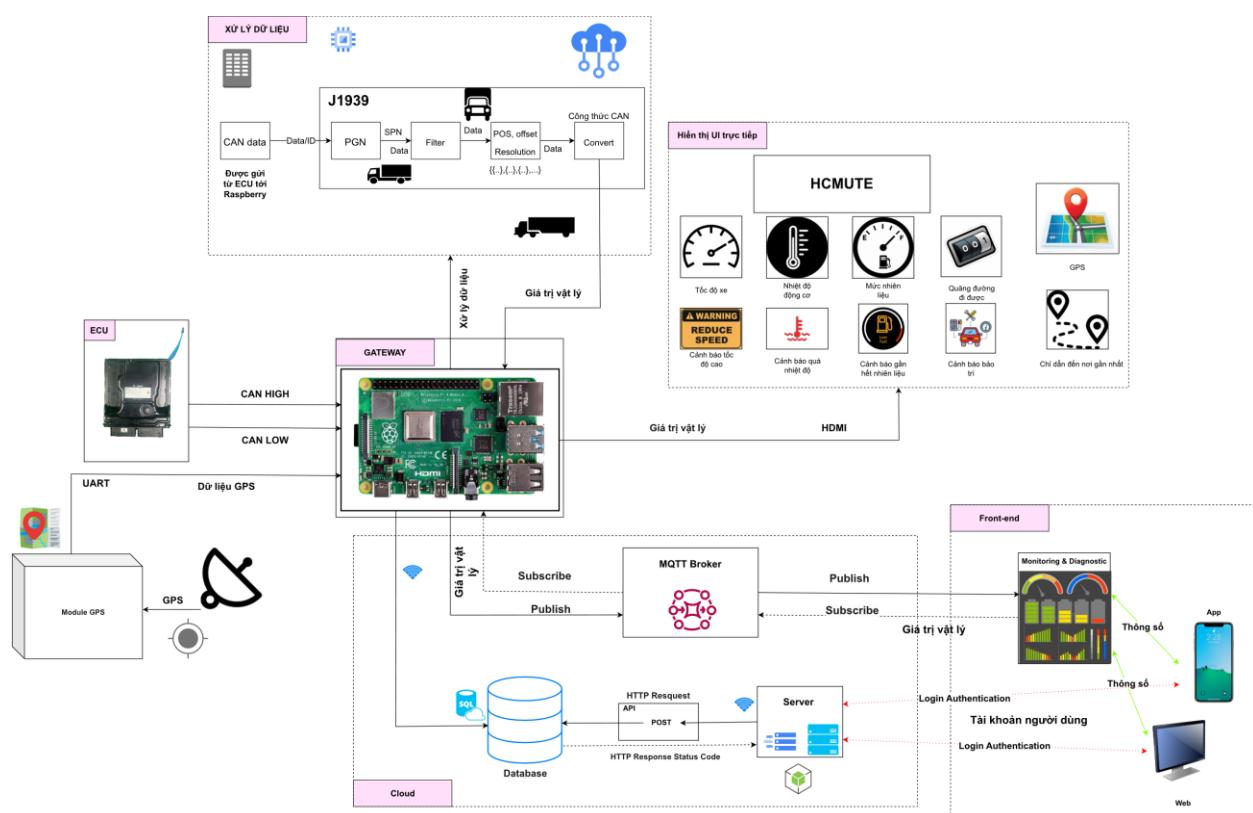
Khối truyền dữ liệu: đây là khối dùng để truyền, nhận và cũng là nơi giao tiếp trung gian giữa các khối người dùng, khối central gateway, khối lưu trữ dữ liệu. Trong khối này sẽ được bao gồm hai loại giao thức truyền, sử dụng HTTP với tính bảo mật thông qua

server để giao tiếp khói người dùng với khói lưu trữ dữ liệu nên phù hợp cho việc đăng nhập và sử dụng MQTT thông qua một Broker để truyền các thông số của xe tới khói người dùng vì độ trễ khi truyền không cao.

Khối lưu trữ dữ liệu: có khả năng lưu trữ thông số được gửi từ khói central gateway và khói truyền dữ liệu.

Khối người dùng: giao diện hiển thị các thông số của xe dưới dạng số vật lý như tốc độ xe, nhiệt độ động cơ, mức nhiên liệu hiện tại, quãng đường đã đi, vị trí của xe, thông tin về phần mềm đang sử dụng và có chức năng đăng nhập để đảm bảo tính bảo mật cho người dùng, kết nối thông qua khói truyền dữ liệu.

3.2.4 Hoạt động của hệ thống



Hình 3.3: Sơ đồ hoạt động của hệ thống

Trong Hình 3.2 sẽ mô tả về luồng hoạt động của dữ liệu, cách di chuyển từ ECU cho đến yêu cầu về hiển thị các thông số giá trị vật lý trên giao diện web và ứng dụng trên di động Android.

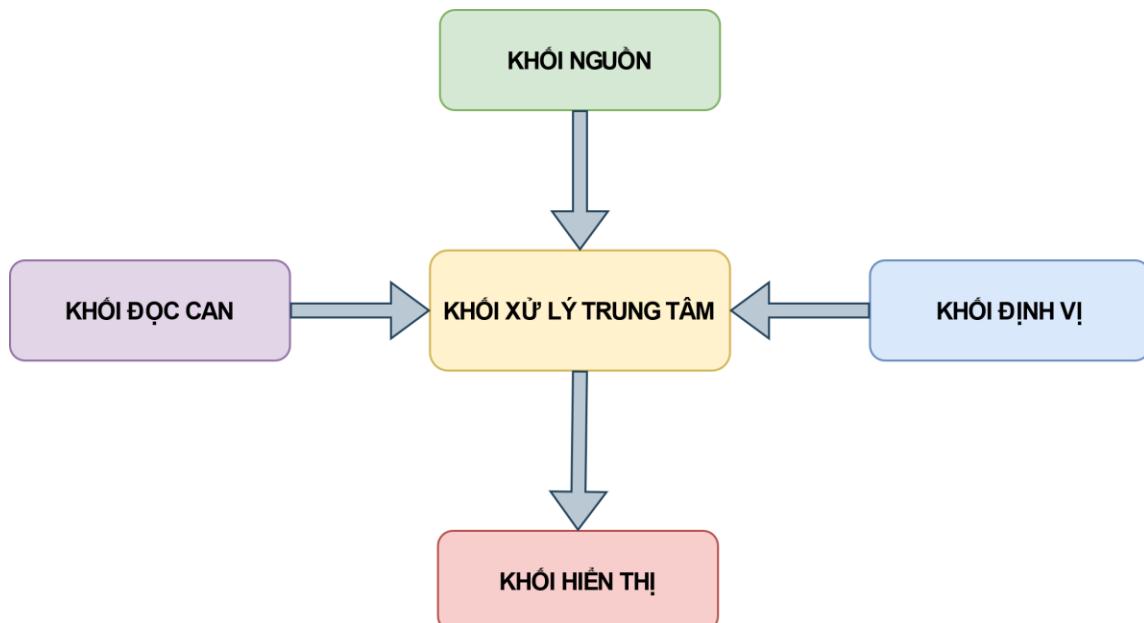
- Đầu tiên ở ECU sẽ sử dụng các công cụ được hỗ trợ bởi công ty Bosch để có thể cấu hình các biến CAN mà mình muốn dùng với các ID được chọn lựa theo chuẩn giao thức J1939. Ngoài ra còn gắn thêm một Module Neo-6M GPS vào Raspberry Pi 4 để có thể cung cấp vị trí hiện tại của xe.
- Tiếp đến sau khi các dữ liệu CAN được gửi từ ECU thì trên Raspberry thông qua chuẩn CANBUS với hai dây CAN HIGH và CAN LOW sau đó sẽ sử dụng các thuật toán cũng như code để chuyển đổi các giá trị đó thành các giá trị vật lý mà người dùng có thể đọc được như tốc độ xe, nhiệt độ động cơ, mức nhiên liệu, quãng đường đi được.
- Ở khói xử lý dữ liệu thì trên Raspberry sẽ thực hiện lọc, đọc các khung CAN dựa vào PGN hay được biết là ID của CAN thông qua chuẩn J1939 và dò vị trí, resolution, offset sau đó áp dụng công thức CAN để chuyển sang giá trị vật lý.
- Sử dụng một màn hình 7inch gắn trực tiếp trên Raspberry Pi để hiển thị các giá trị thông số của xe cũng như là bản đồ chỉ đường, các số liệu thống kê dạng bảng để người dùng có thể quan sát trực tiếp mà không cần dùng wifi kết nối để xem.
- Giá trị vật lý sau khi được xử lý sẽ được Raspberry gửi lên Database để có thể lưu trữ dữ liệu và gửi lên một MQTT Broker để có thể truyền các dữ liệu theo chuẩn MQTT về web và ứng dụng di động. Sử dụng một Server kết nối với Database thông qua HTTP để dựa vào các thông tin tài khoản trên Database từ đó hỗ trợ cho việc đăng nhập trên web và ứng dụng điện thoại.
- Cuối cùng là hai giao diện web và ứng dụng điện thoại Android, để có thể xem được các thông số của xe thì cần phải qua bước đăng nhập đều có ở cả hai giao diện thông qua các API để kết nối tới Server hoạt động và dựa trên những HTTP status code mà từ đó sẽ đưa ra các quyết định về đăng nhập, nếu đăng nhập thành công sẽ hiển thị các thông số của xe thông qua chuẩn truyền MQTT, bản đồ hiện vị trí hiện tại,

chỉ dẫn đường đi, thông tin cơ bản của xe. Dựa vào các thông số của xe mà từ đó trên giao diện sẽ đưa ra các thông báo tương ứng như cảnh báo quá nhiệt độ, tốc độ cao, gần hết nhiên liệu từ đó người dùng có thể xem và xử lý.

3.3 THIẾT KẾ PHẦN CỨNG

Khối gateway được thiết kế để sử dụng cho việc thu thập và xử lý các dữ liệu CAN nhận được từ khối xử lý (ECU), gửi các giá trị đã chuyển đổi mà người dùng có thể đọc được lên khối lưu trữ dữ liệu và khối truyền dữ liệu.

Dựa vào đó nhóm chúng tôi sẽ thiết kế sơ đồ khái chi tiết tại central gateway gồm:



Hình 3.4: Sơ đồ khái chi tiết tại central gateway

Khối đọc CAN: sử dụng module RS485 CAN HAT kết nối với ECU thông qua hai dây CAN HIGH và CAN LOW để đọc dữ liệu CAN và truyền các khung CAN đó qua khối xử lý trung tâm là Raspberry Pi thông qua chuẩn giao tiếp SPI.

Khối xử lý trung tâm: sử dụng một máy tính thu nhỏ Raspberry Pi để đảm nhiệm cho việc quản lý và xử lý mọi hoạt động của hệ thống từ việc chuyển đổi các dữ liệu cho đến khi truyền gửi các dữ liệu đó lên Cloud.

Khối định vị: dùng module định vị GPS NEO-6M với kích thước nhỏ gọn và độ nhạy cao thích hợp cho việc giám sát, định vị phương tiện.

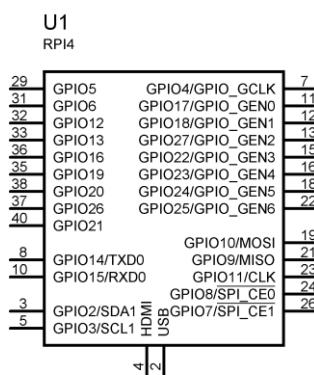
Khối hiển thị: sử dụng màn hình Waveshare 7 Inch kết nối HDMI với Raspberry Pi, có tích hợp cảm ứng trên màn hình, dùng để hiển thị trực tiếp các thông số nhiệt độ động cơ, tốc độ xe, mức nhiên liệu, quãng đường đã đi trên xe.

Khối nguồn: dùng máy phát nguồn để cấp trực tiếp cho ECU hoạt động và nguồn chuyên dụng của Raspberry để hoạt động.

Từ những thiết kế sơ đồ khái chi tiết trên nhóm sẽ dựa vào đó và tiến tới việc thiết kế các khái đó dưới dạng sơ đồ nguyên lý từng phần và cuối cùng sẽ là sơ đồ nguyên lý tổng thể của toàn bộ hệ thống.

3.3.1 Khối xử lý trung tâm

Nhóm chúng tôi lựa chọn Raspberry Pi 4 để làm khái xử lý vì đây được xem như là một chiếc máy tính thu nhỏ có thể chạy được hệ điều hành và cài đặt nhiều ứng dụng khác nhau. Do khi ECU xử lý và truyền dữ liệu với tốc độ rất cao với một Cycle là khoảng 10ms thì khi sử dụng các loại vi điều khiển như các series của Arduino, PIC,... khó có thể đọc được các dữ liệu đó đúng với thời gian mà ECU truyền vào, ngoài ra để có thể đáp ứng được yêu cầu của hệ thống cần cập nhật các thông số liên tục và độ trễ khi truyền ít hơn 5 giây thì Raspberry Pi 4 sẽ là phần cứng lý tưởng nhất cho toàn bộ hệ thống.



Hình 3.5: Sơ đồ nguyên lý Raspberry Pi 4

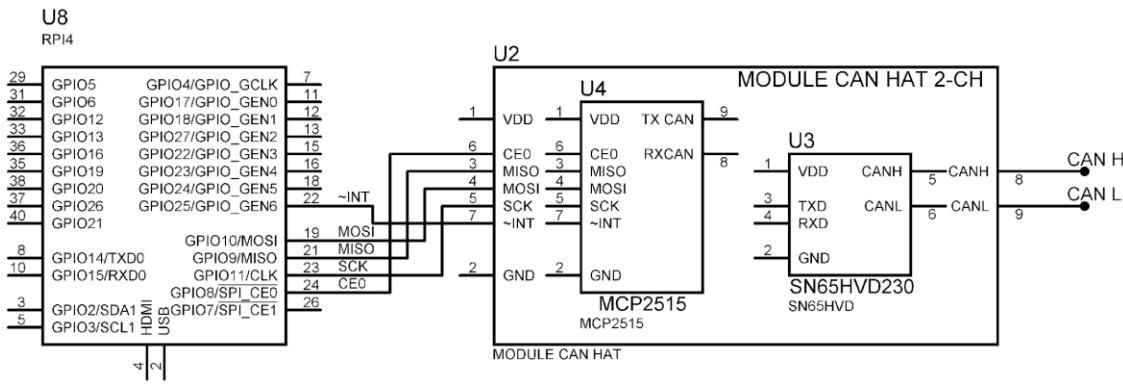
Raspberry Pi 4 được sử dụng CPU quad-core 64-bit ARM Cortex-A72 có thể xem đây là dòng chip mạnh nhất đối với chiếc máy tính thu nhỏ này, với phiên bản RAM 2GB từ đó có thể xử lý nhiều tác vụ bởi vì trên hệ thống thì Raspberry không chỉ đọc dữ liệu CAN mà còn dùng chuyển đổi dữ liệu theo chuẩn J1939 và kết nối wifi hỗ trợ truyền gửi dữ liệu lên Database và Broker.

3.3.2 Khối đọc CAN

Sử dụng module Waveshare RS485 CAN HAT vì đây là loại chuyên dùng đọc các chuẩn CAN và RS485, được sử dụng phổ biến với giá thành không cao, có thể tìm thấy ở các cửa hàng linh kiện gần khu vực xung quanh trường nên đây là sự lựa chọn tốt nhất cho nhóm chúng tôi thực hiện với đề tài này, đặc biệt được thiết kế để sử dụng chuyên dụng cho Raspberry Pi, sử dụng vi điều khiển CAN kết nối với một CAN Transceiver và giao tiếp với khối xử lý trung tâm cũng như là với ECU của Bosch. Sử dụng nguồn cấp 3.3V trực tiếp từ Raspberry.

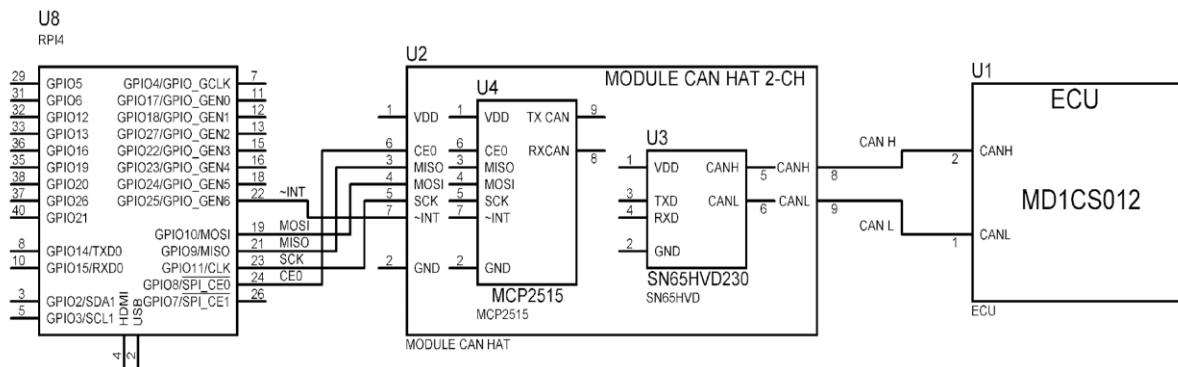
Bảng 3.1: Sơ đồ kết nối module Waveshare RS485 CAN HAT với Raspberry

Module Waveshare RS485 CAN HAT	Raspberry Pi 4
MOSI	GPIO10
MISO	GPIO9
SCK	GPIO11
~INT	GPIO25
3.3V	3.3V
GND	GND



Hình 3.6: Sơ đồ nguyên lý kết nối module CAN HAT với Raspberry

Ngoài ra về việc kết nối giữa ECU với khối xử lý trung tâm Raspberry Pi 4 thì cũng sẽ thông qua module CAN này để hỗ trợ cho Raspberry việc đọc các CAN Frame thông qua hai dây CAN HIGH và CAN LOW. Hình 3.6 dưới đây sẽ mô tả thêm về cách kết nối module CAN HAT với ECU MD1CS012 do công ty Bosch hỗ trợ cho hệ thống.



Hình 3.7: Sơ đồ nguyên lý kết nối module CAN HAT với Raspberry và ECU

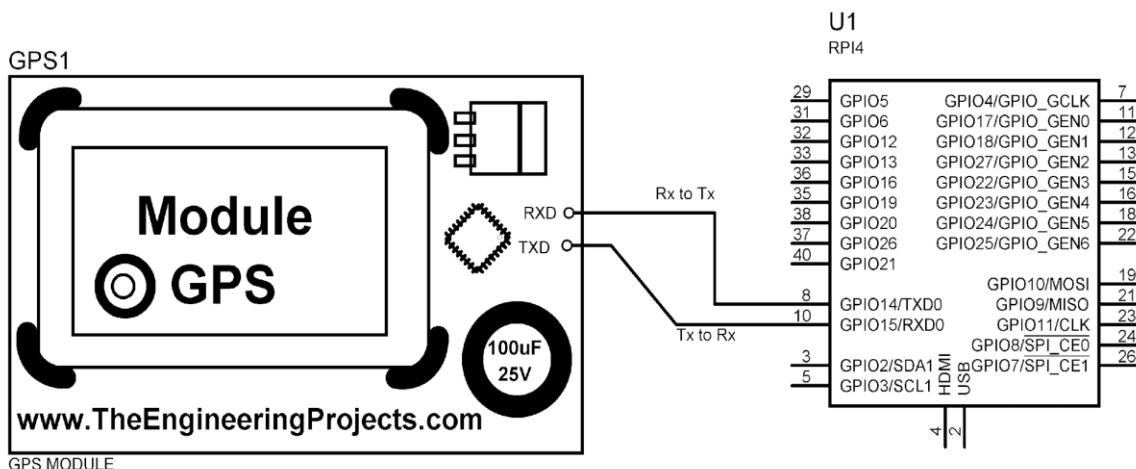
3.3.3 Khối đọc định vị

Sử dụng module NEO-6M GPS đây được xem là loại module GPS phổ biến nhất hiện nay với giá thành rẻ và dễ sử dụng có thể định vị toàn cầu. Với chức năng GPS sẽ sử dụng hệ thống vệ tinh của Mỹ sẽ cung cấp tọa độ vị trí chính xác và nhanh. Sử dụng chuẩn

giao tiếp UART/TTL để kết nối với khíu xử lí, module này sè cung cấp giá trị tọa độ víi sai sò nhò chèn lêch khòng cao trong khoàng bán k&inh 5m.

Bảng 3.2: So đồ kết nối module NEO-6M với Raspberry

Module NEO-6M GPS	Raspberry Pi 4
Tx	GPIO14
Rx	GPIO15
5V	5V
GND	GND



Hình 3.8: Sơ đồ nguyên lý kết nối module NEO-6M với Raspberry

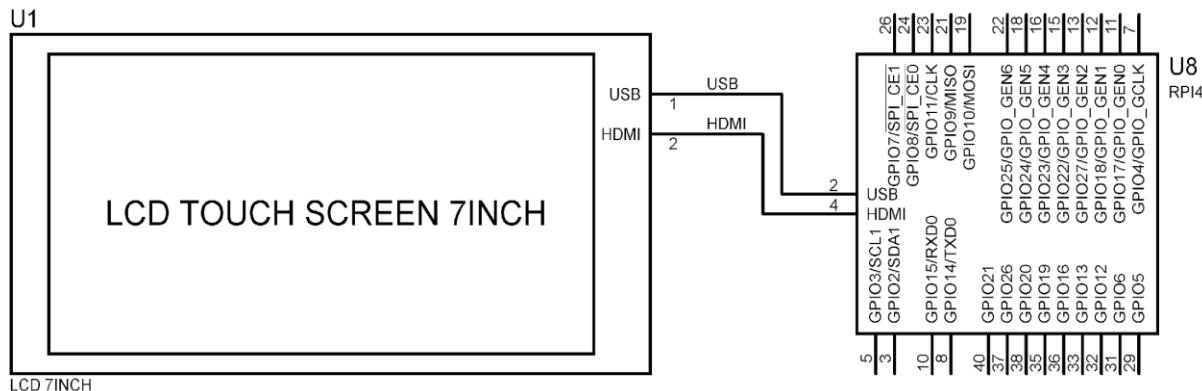
3.3.4 Khối hiển thi

Đây là khối sẽ hiển thị các giá trị thông số cũng như là bảng thống kê trực tiếp được kết nối với khối xử lý trung tâm, nhóm chúng tôi lựa chọn một màn hình hiển thị tương thích với Raspberry, thì hiện nay trên thị trường linh kiện có rất nhiều loại màn hình chúng tôi chọn màn hình với kích thước 7inch nhằm tạo điều kiện thuận lợi nhất cho người dùng có thể quan sát đầy đủ các chi tiết, thông số, giao diện và có tích hợp cảm ứng chạm trên

màn hình từ đó sẽ dễ dàng thao tác hơn. Màn hình bao gồm 2 dây là HDMI để truyền tín hiệu lên màn hình cũng như là cảm ứng từ Raspberry và dây còn lại là USB thì sẽ là dây cấp nguồn cho màn hình.

Bảng 3.3: Sơ đồ kết nối giữa màn hình 7Inch và Raspberry

Màn hình 7Inch	Raspberry Pi 4
HDMI	HDMI
USB	USB



Hình 3.9: Sơ đồ nguyên lý kết nối màn hình 7Inch với Raspberry

3.3.5 Khối nguồn

Về sử dụng nguồn sẽ được chia ra sử dụng 2 nguồn khác nhau:

+ Nguồn sử dụng từ một máy phát nguồn đa năng để có thể cấp nguồn cho ECU MD1CS012 vì đây là một linh kiện được sử dụng thực tế trong xe nên về giá thành rất cao nên nhóm phải sử dụng nguồn đa năng này để đảm bảo an toàn nhất cho thiết bị.

Bảng 3.4: Các thông số về dòng điện, điện áp và công suất tiêu thụ của ECU

Tên linh kiện	Điện áp hoạt động	Dòng điện tiêu thụ	Công suất tiêu thụ trung bình
ECU MD1CS012	12V	~0.4A khi hoạt động ~0.1A khi không hoạt động	~4.8W khi hoạt động 1.2W khi không hoạt động

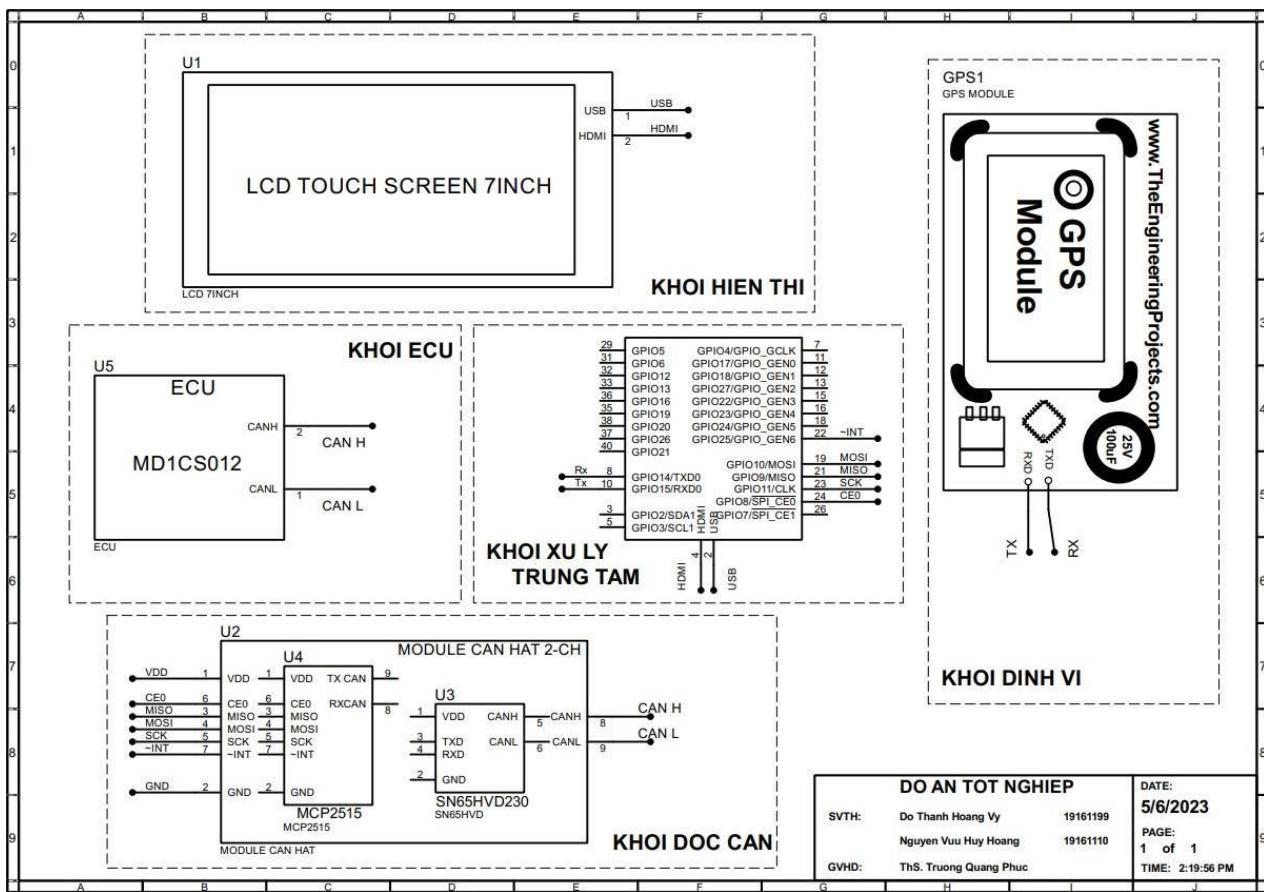
+ Tiếp đến là khi sử dụng Raspberry sẽ được đi kèm với nguồn Adapter mức điện áp ra là 5.1V, đây là khối nguồn chuẩn của Raspberry nên không cần dùng thêm các nguồn ngoài khác.

Bảng 3.5: Các thông số về dòng điện, điện áp, công suất tiêu thụ của các linh kiện khác

Tên linh kiện	Điện áp hoạt động	Dòng điện tiêu thụ	Công suất tiêu thụ trung bình
Raspberry Pi 4	5V	1.1A	5.5W
Module RS485 CAN HAT	3.3V	10mA	33mW
Module NEO-6M GPS	5V	24mA	120mW
Màn hình cảm ứng 7inch	5V	~2A	~10W

+ Dựa trên thông số dòng và áp của các linh kiện trên thì công suất tiêu thụ trung bình của toàn hệ thống là 5.653W. Ngoài ra, dùng nguồn Adapter của kit Raspberry với điện áp 5.1V DC và dòng tiêu thụ là 3A để hoạt động có thể cung cấp công suất tiêu thụ trung bình lên tới hơn 15W, hai linh kiện khác có điện áp hoạt động là 3.3V, 5V thì sẽ sử dụng nguồn từ các chân GPIO 5V, 3.3V và GND trên kit Raspberry.

3.3.6 Sơ đồ nguyên lý



Hình 3.10: Sơ đồ nguyên lý kết nối phần cứng

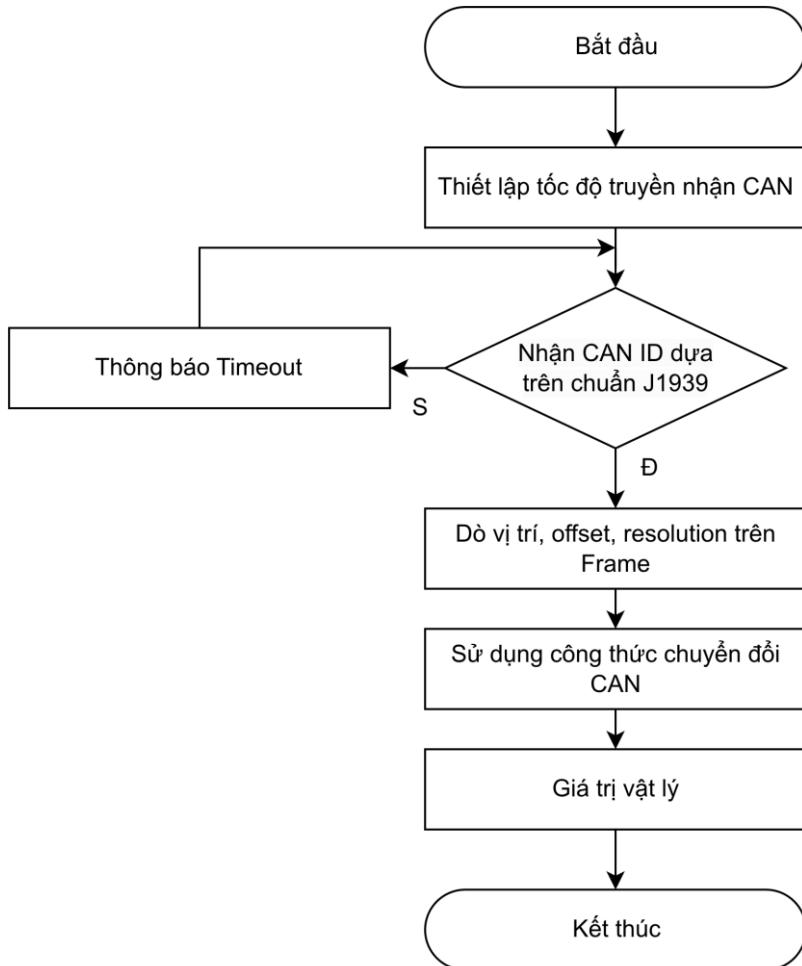
3.4 THIẾT KẾ PHẦN MỀM

Đối với việc thiết kế phần mềm thì các giải thuật và lưu đồ sẽ được thiết kế theo thứ tự đi từ central gateway, cách xử lý CAN Frame, đọc dữ liệu CAN cho đến các giao diện người như web và ứng dụng trên điện thoại Android.

3.4.1 Lưu đồ giải thuật trên central gateway

Lưu đồ giải thuật trên central gateway chúng tôi sẽ chia ra thành hai phần khác nhau ứng với các bước từ xử lý dữ liệu cho đến truyền dữ liệu lên Database hay là MQTT Broker.

- **Lưu đồ giải thuật xử lý CAN Frame:**



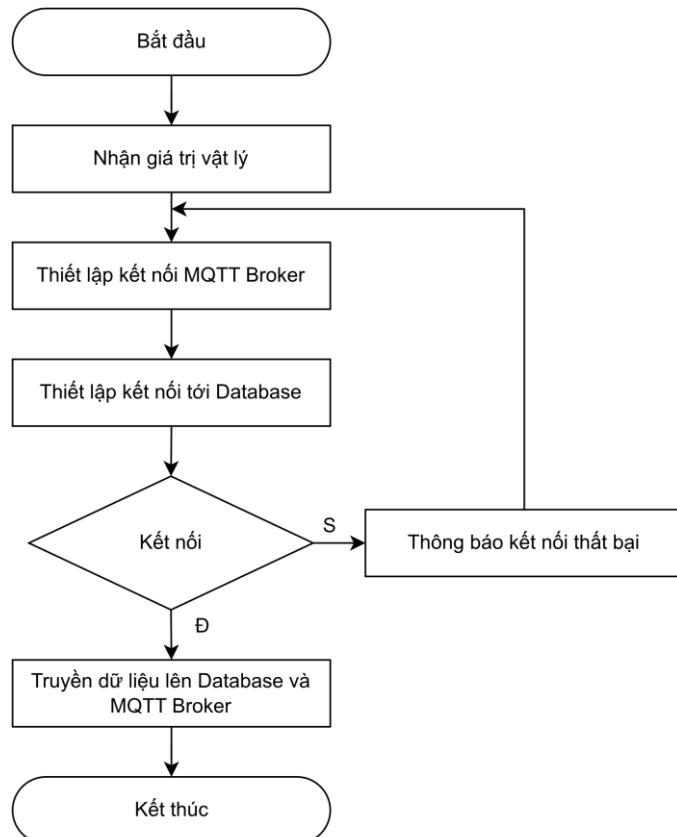
Hình 3.11: Lưu đồ giải thuật xử lý CAN Frame

Thiết lập tốc độ truyền nhận CAN với BaudRate là 500000, sử dụng thư viện Python-Can để thiết lập chọn lọc các dữ liệu dựa trên những CAN ID được lựa chọn dựa theo tiêu chuẩn J1939, khi không có dữ liệu đúng với CAN ID (PGN) được lựa chọn sẽ xuất hiện thông báo Timeout. Nếu hiển thị đúng các CAN ID thì dựa vào chuẩn J1939 với vị trí tương ứng và offset cũng như resolution của các giá trị hiển thị trên Frame, từ đó áp dụng công thức chuyển đổi giá trị của CAN để có thể chuyển đổi sang giá trị vật lý. Tất cả đều được thiết lập và vận hành bằng code Python được thực hiện trên Raspberry có thể theo dõi thông qua Terminal để quan sát được các giá trị đã chuyển đổi.

Công thức chuyển đổi giá trị vật lý như nhiệt độ động cơ, tốc độ xe, quãng đường đi được, mức nhiên liệu từ các giá trị CAN:

$$\text{Physical value} = \text{CAN value} * \text{resolution} + \text{offset}$$

➤ **Lưu đồ giải thuật truyền gửi dữ liệu:**



Hình 3.12: Lưu đồ giải thuật truyền gửi dữ liệu

Khi đã có các giá trị vật lý, sẽ tiến tới thiết lập kết nối với MQTT Broker thông qua các Host, Port, tương tự với Database kết nối thông qua Host, User, Password. Sử dụng code Python để tiến hành cấu hình các kết nối và khi kết nối không thành công thì sẽ có thông báo kết nối thất bại, kết nối lại đến khi nào đã kết nối được tới MQTT Broker và Database thì sẽ bắt đầu cập nhật và truyền gửi dữ liệu lên.

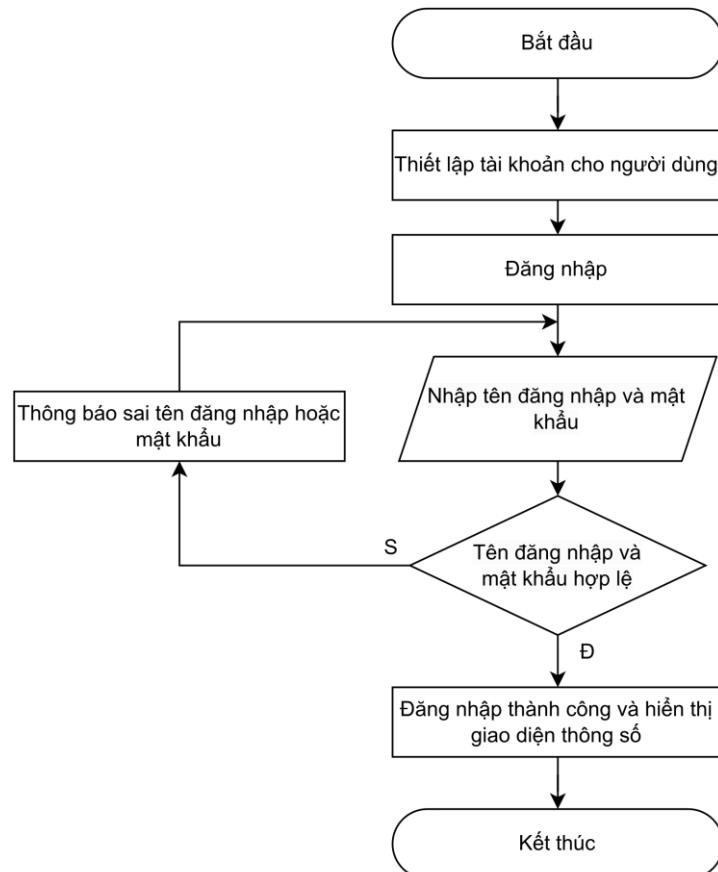
3.4.2 Lưu đồ giải thuật trên giao diện người dùng

Với lưu đồ giải thuật trên giao diện người dùng sẽ bao gồm 4 phần lớn khác nhau đầu tiên sẽ là lưu đồ giải thuật đăng nhập cho người dùng tiếp đến là lưu đồ giải thuật trên

ứng dụng điện thoại tiếp theo sẽ là lưu đồ giải thuật trên web và cuối cùng là lưu đồ giải thuật trên GUI của màn hình 7inch.

➤ Lưu đồ giải thuật đăng nhập tài khoản cho người dùng

Ở đây nhóm chúng tôi sẽ thiết lập lưu đồ giải thuật đăng nhập tài khoản cho người dùng ở web và ứng dụng điện thoại là cùng một lưu đồ vì khi sử dụng chức năng này thì cả hai đều sẽ có phần đăng nhập nên về mặt lý thuyết thì đều chung một cách vận hành, hoạt động.



Hình 3.13: Lưu đồ giải thuật đăng nhập tài khoản

Khi người dùng sử dụng web và ứng dụng trên điện thoại thì bước đầu tiên để có thể xem được các thông số trong giao diện là phải đăng nhập tài khoản, tài khoản sẽ được cấp giá định như từ nhà cung cấp xe cung cấp. Nếu tên đăng nhập và mật khẩu không hợp lệ thì trên giao diện người dùng sẽ hiện thông báo không hợp lệ có thể sai tên đăng nhập

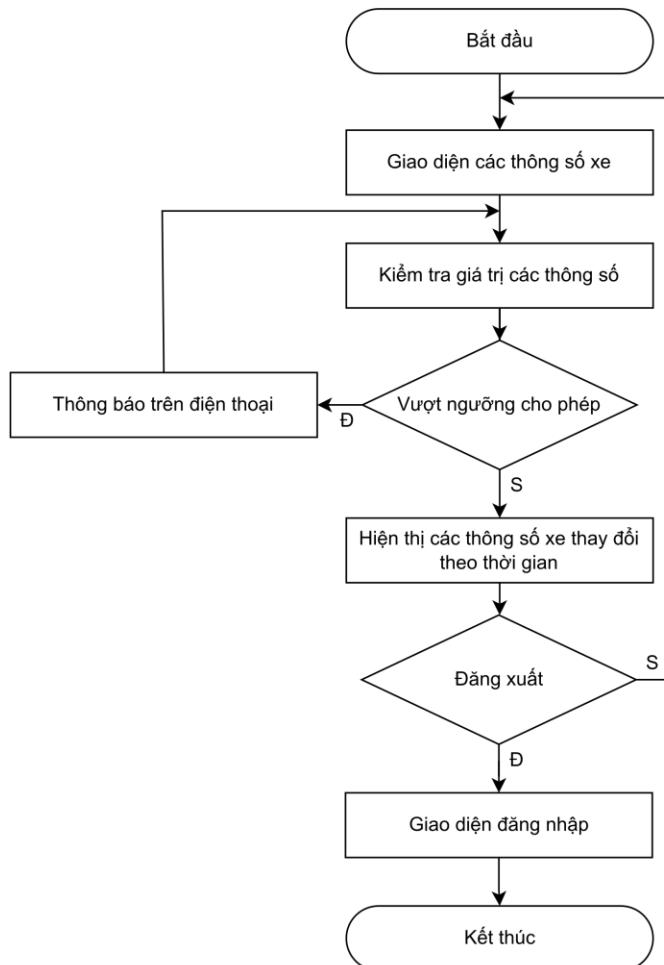
hay mật khẩu. Nếu người dùng đăng nhập thành công thì giao diện trên web và ứng dụng điện thoại sẽ hiển thị các trang thông số, từ đó người dùng có thể chọn các tính năng khác.

➤ **Lưu đồ người dùng sử dụng xem thông số trên giao diện ứng dụng điện thoại**

Với ứng dụng trên điện thoại sẽ bao gồm ba trang khác nhau và ứng với mỗi trang sẽ có một công dụng khác nhau. Tương ứng sẽ có 3 lưu đồ thể hiện cho từng tính năng có trong các trang:

- **Lưu đồ trang hiển thị giao diện các thông số của xe:**

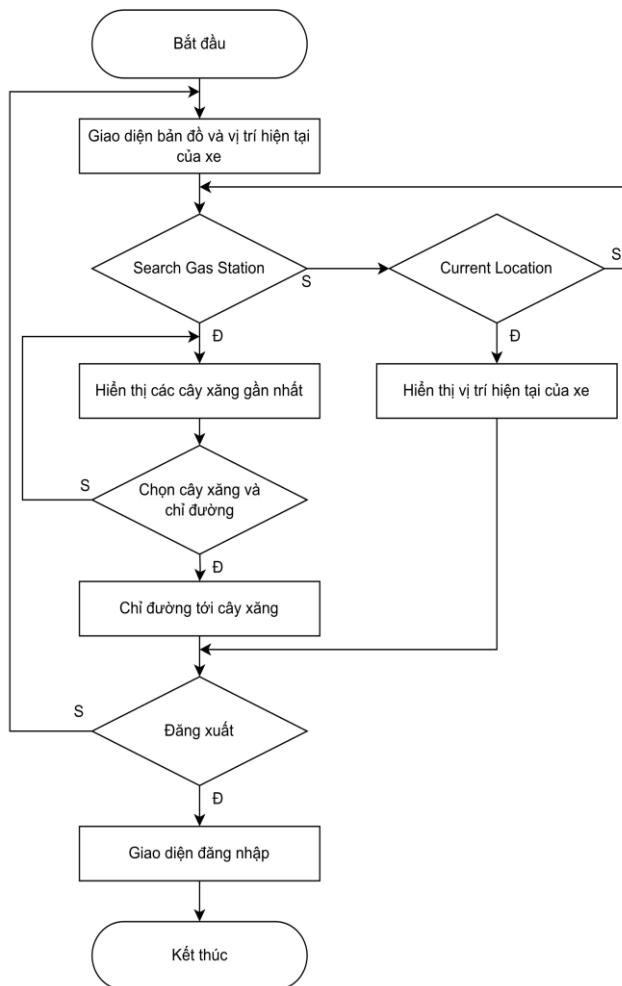
Đầu tiên sẽ là trang sẽ hiển thị các thông số của xe đã được thu thập thông qua MQTT Broker được gửi từ Raspberry Pi 4.



Hình 3.14: Lưu đồ người dùng sử dụng xem thông số của xe

Sau khi người dùng đã đăng nhập thành công thì hiển thị giao diện thông số, tại đó người dùng có thể chuyển đổi giữa các trang, hiện nay trên ứng dụng điện thoại sẽ có 3 trang thì trang đầu tiên cũng là trang chính sẽ hiển thị giao diện các thông số của xe như nhiệt độ động cơ, tốc độ xe, quãng đường đã đi được, mức nhiên liệu còn lại và được thiết lập với các ngưỡng cho phép khác nhau, khi một thông số vượt qua ngưỡng sẽ gửi thông báo cảnh báo về trên điện thoại, chẳng hạn như quá nhiệt độ hoặc cảnh báo tốc độ xe đang chạy rất nhanh,...

- **Lưu đồ hiển thị giao diện bản đồ:**

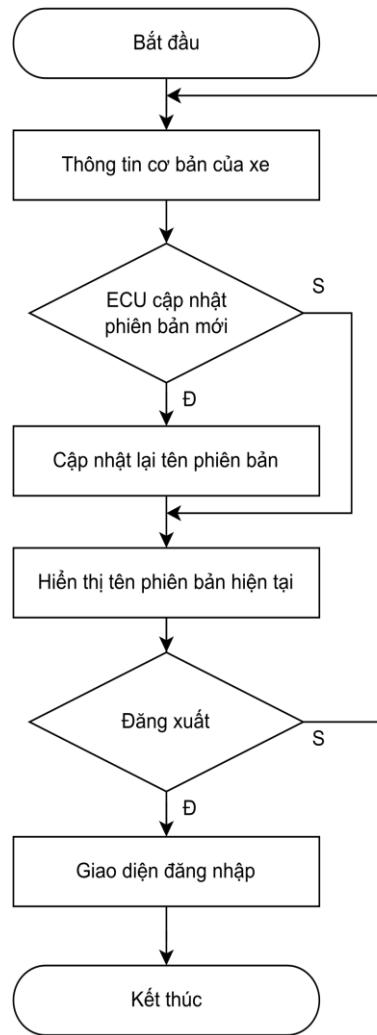


Hình 3.15: Lưu đồ người dùng sử dụng xem bản đồ

Tiếp đến là trang thứ hai sẽ là giao diện hiển thị bản đồ, trên bản đồ người dùng có thể quan sát được vị trí hiện tại của xe. Giao diện sẽ có 2 nút chức năng đó là “Search Gas

Station” và “Current Location”, khi người dùng nhấn chọn “Search Gas Station” thì trên giao diện sẽ hiển thị các cây xăng gần nhất trong vòng tròn với một bán kính nhất định, khi ấn vào đó sẽ có hiển thị thông tin cây xăng cũng như là có thể chỉ đường tới cây xăng đó khi nhấn thêm một lần nữa. Đối với việc ấn vào “Current Location” thì sẽ hiển thị vị trí hiện tại của người dùng và khi camera của bản đồ ở bất cứ đâu khi ấn vào sẽ được chuyển camera đến vị trí hiện tại của người dùng. Cuối cùng là đăng xuất nếu người dùng muốn.

- **Lưu đồ hiển thị giao diện thông tin của xe:**



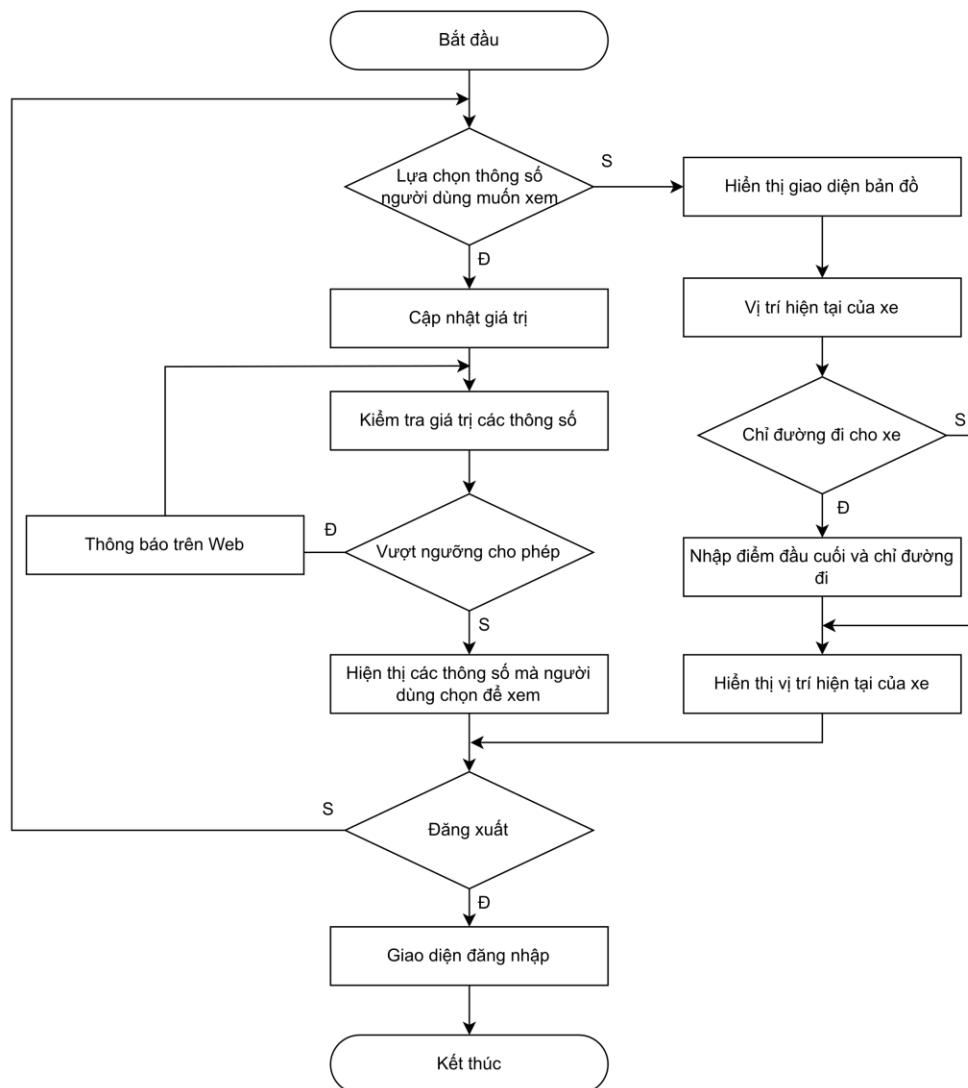
Hình 3.16: Lưu đồ người dùng sử dụng xem thông tin của xe

Với trang cuối sẽ là hiển thị các thông tin về tên ECU, tên phiên bản, tên thiết bị. Khi ECU được cập nhật thông qua OTA thì sẽ được cập nhật một phiên bản mới và khi đó

ứng dụng trên điện thoại sẽ cập nhật các giá trị đó vào trong các bảng thông tin về phiên bản, để từ đó có thể biết được phiên bản ECU hiện tại đang được dùng.

Cuối cùng ở mỗi trang thì đều sẽ tích hợp một nút nhấn đăng xuất, khi người dùng muốn đăng xuất khỏi ứng dụng điện thoại thì sẽ trở về giao diện đăng nhập. Nếu người dùng không đăng xuất ứng dụng thì dựa vào tên đăng nhập và mật khẩu trước đó thì trên ứng dụng sẽ lưu trữ, khi người dùng thoát khỏi ứng dụng này và vào lại thì vẫn sẽ chuyển thẳng vào trang hiển thị giao diện thông số mà không cần đăng nhập lại.

➤ Lưu đồ người dùng sử dụng xem thông số trên giao diện Web



Hình 3.17: Lưu đồ người dùng sử dụng xem thông số trên giao diện Web

Với web thì giao diện hiển thị và cơ chế hoạt động sẽ tương tự như App, sau khi đã đăng nhập thành công thì trên web sẽ hiển thị trang chính và trên trang chính sẽ có thanh công cụ để có thể lựa chọn các tính năng để xem thông số xe hay bản đồ hoặc đăng xuất, thì sẽ tùy thuộc vào người dùng mà các trang sẽ chuyển đổi khác nhau.

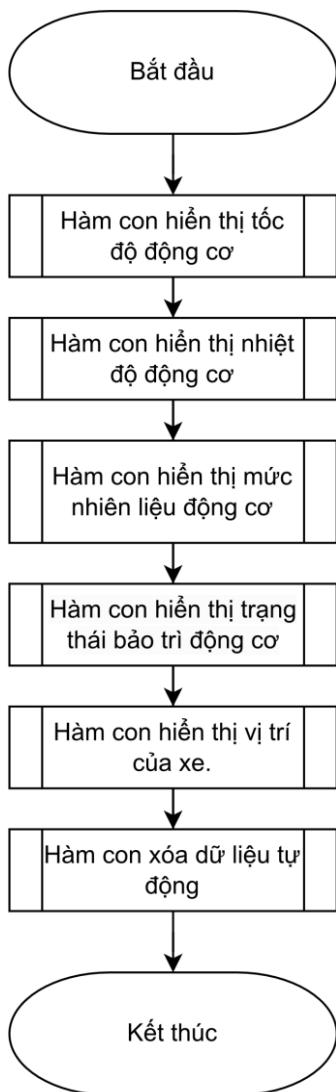
Trên web sẽ không hiển thị hết một lượt các giá trị thông số xe như ứng dụng trên điện thoại mà ở đây sẽ chỉ hiển thị từng giá trị thông số cho người dùng có thể quan sát dễ dàng các thông số hơn, khi có thông số của xe mà vượt ngưỡng cho phép thì sẽ hiện thông báo trên trang Web đó để cảnh báo cho người dùng. Tích hợp một bản đồ vào web vừa hiển thị vị trí hiện tại của xe cũng như là sẽ giúp người dùng có thể chỉ dẫn đường đi khi nhập vị trí hiện tại và vị trí mà người dùng muốn đến. Khi không muốn dùng web thì sẽ có một nút đăng xuất giúp người dùng thoát được web.

➤ **Lưu đồ người dùng sử dụng xem GUI, thông số trực tiếp trên màn hình 7inch**

Lưu đồ giải thuật sẽ bao gồm một chương trình chính và trong chương trình chính đó sẽ bao gồm các lưu đồ hàm con khác nhau, hiện trên lưu đồ chính sẽ bao gồm 6 lưu đồ hàm con: hiển thị tốc độ động cơ, hiển thị nhiệt độ động cơ, hiển thị mức nhiên liệu động cơ, hiển thị trạng thái bảo trì động cơ, hiển thị vị trí của xe, hiển thị dữ liệu trích xuất, xóa dữ liệu tự động

● **Lưu đồ chương trình chính.**

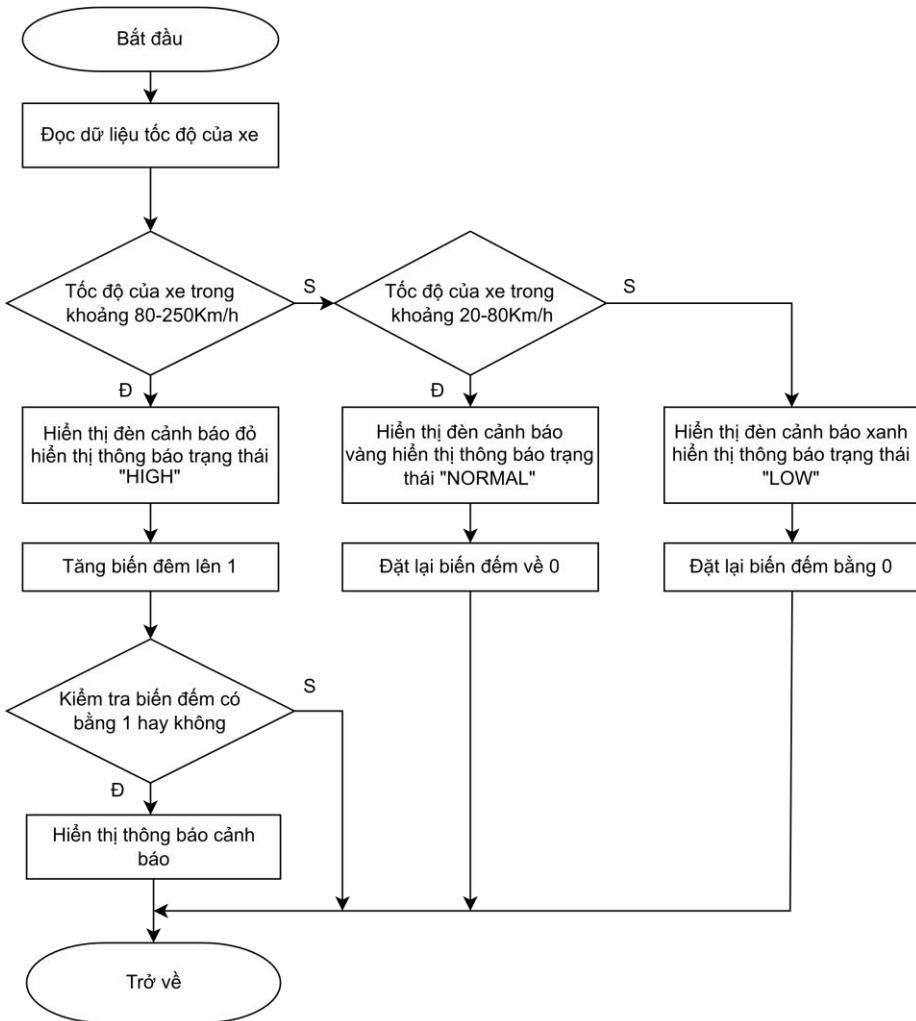
Chương trình hiển thị trên GUI được thực hiện bằng cách tuần tự gọi và thực hiện các hàm con: hiển thị tốc độ động cơ, hiển thị nhiệt độ động cơ, hiển thị mức nhiên liệu động cơ, hiển thị trạng thái bảo trì động cơ, hiển thị vị trí của xe, hiển thị dữ liệu trích xuất, xóa dữ liệu tự động. Chương trình chính của GUI sẽ thực hiện lặp lại liên tục từ khi người dùng khởi động phương tiện cho đến khi người dùng dừng hoạt động của phương tiện



Hình 3.18: Lưu đồ hàm chính của chương trình hiển thị GUI

- **Lưu đồ hàm con hiển thị tốc độ của xe**

Hàm con hiển thị tốc độ xe được thiết kế với mục đích hiển thị thông số tốc độ của xe lên màn hình của GUI, đồng thời dựa trên giá trị ấy đưa ra các cảnh báo phù hợp trong quá trình người dùng sử dụng phương tiện.



Hình 3.19: Lưu đồ hàm con hiển thị tốc độ của xe

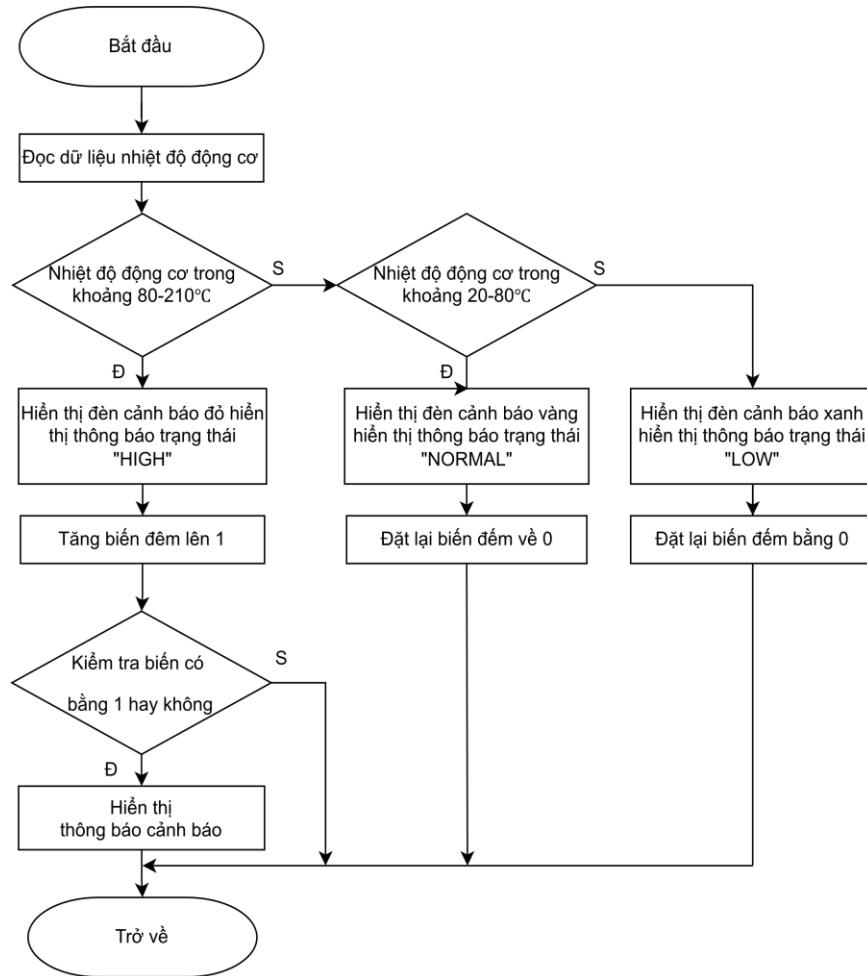
Khi hàm con này được gọi, chương trình sẽ thực hiện tuần tự các hoạt động.

Đọc dữ liệu từ các tệp tin lưu trữ dữ liệu tốc độ của xe, tiến hành kiểm tra tốc độ xe có nằm trong khoảng 80-250Km/h hay không, nếu đúng tiến hành bật đèn trạng thái màu đỏ và chú thích trạng thái “HIGH” sau đó tăng biến đêm lên một, kiểm tra xem biến đêm có bằng một hay không, nếu có hiển thị cảnh báo nếu không quay về chương trình chính.

Trong trường hợp tốc độ không nằm trong khoảng 80-250Km/h kiểm tra tốc độ xe có nằm trong khoảng 20-80Km/h hay không, nếu đúng tiến hành bật đèn trạng thái màu vàng và chú thích trạng thái “NORMAL” sau đó đặt lại biến đêm bằng 0, trả về chương trình chính.

Trong trường hợp tốc độ không nằm trong khoảng 80-250Km/h và cũng không nằm trong khoảng 20-80Km/h, tiến hành bật đèn trạng thái màu xanh và chú thích trạng thái “LOW” sau đó đặt lại biến đếm bằng 0, trở về chương trình chính.

- **Lưu đồ hàm con hiển thị nhiệt độ động cơ**



Hình 3.20: Lưu đồ chương trình con hiển thị nhiệt độ động cơ

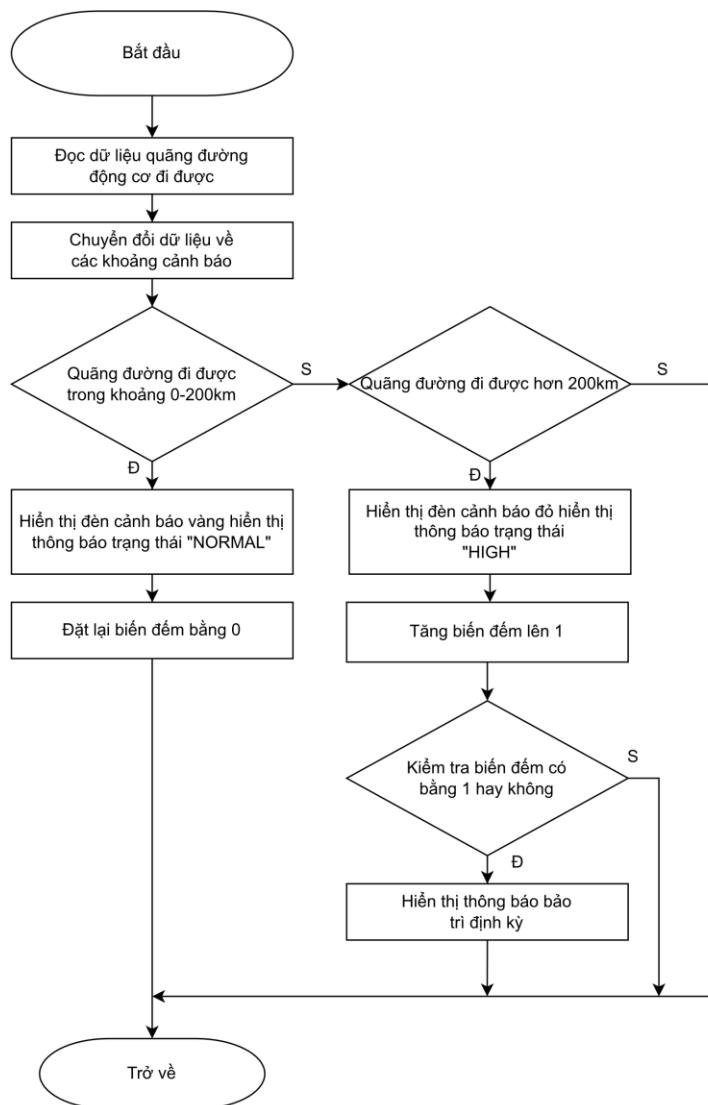
Khi hàm con này được gọi, chương trình sẽ thực hiện tuần tự các hoạt động.

Đọc dữ liệu từ các tệp tin lưu trữ dữ liệu nhiệt độ động cơ, tiến hành kiểm tra nhiệt độ động cơ có nằm trong khoảng 80-210°C hay không, nếu đúng tiến hành bật đèn trạng thái màu đỏ và chú thích trạng thái “HIGH” sau đó tăng biến đếm lên một, kiểm tra xem biến đếm có bằng một hay không, nếu có hiển thị cảnh báo nếu không quay về chương trình chính.

Trong trường hợp tốc độ không nằm trong khoảng 80-210°C kiểm tra tốc độ động cơ có nằm trong khoảng 20-80°C hay không, nếu đúng tiến hành bật đèn trạng thái màu vàng và chú thích trạng thái “NORMAL” sau đó đặt lại biến đếm bằng 0, trở về chương trình chính.

Trong trường hợp tốc độ không nằm trong khoảng 80-210°C và cũng không nằm trong khoảng 20-80°C, tiến hành bật đèn trạng thái màu xanh và chú thích trạng thái “LOW” sau đó đặt lại biến đếm bằng 0, trở về chương trình chính.

● Lưu đồ hàm con hiển thị trạng thái bảo trì động cơ



Hình 3.21: Lưu đồ hàm con hiển thị trạng thái bảo trì của phương tiện

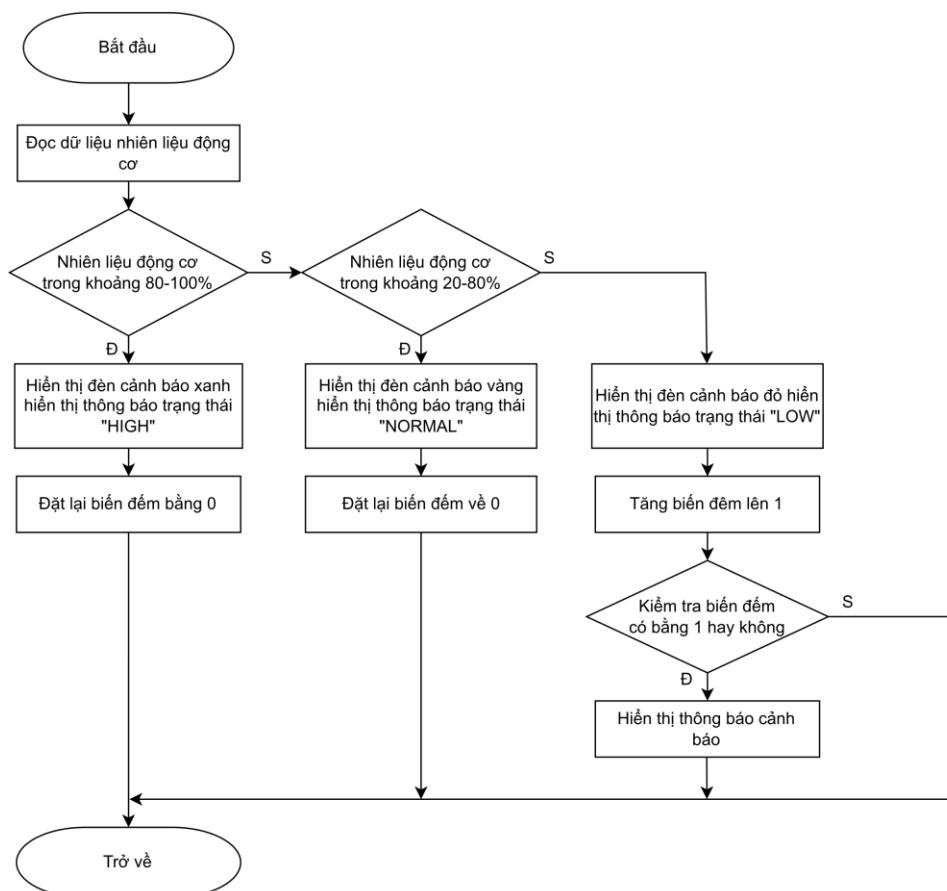
Khi hàm con này được gọi, chương trình sẽ thực hiện tuần tự các hoạt động.

Đọc dữ liệu từ các tệp tin lưu trữ quãng đường mà xe đã đi được, tiến hành chuyển đổi quãng đường xe đã đi sang số Km phù hợp với các khoảng so sánh.

Tiến hành kiểm tra số Km có nằm trong khoảng 0-200Km hay không, nếu đúng tiến hành bật đèn trạng thái màu vàng và chú thích trạng thái “NORMAL” sau đó đặt lại biến đếm bằng 0, trở về chương trình chính.

Trong trường hợp số Km hơn 200Km tiến hành bật đèn trạng thái màu đỏ và chú thích trạng thái “HIGH” sau đó tăng biến đếm lên một, kiểm tra xem biến đếm có bằng một hay không, nếu có hiển thị cảnh báo bảo trì nếu không quay về chương trình chính.

• Lưu đồ hàm con hiển thị mức nhiên liệu động cơ



Hình 3.22: Lưu đồ hàm con hiển thị mức nhiên liệu của động cơ

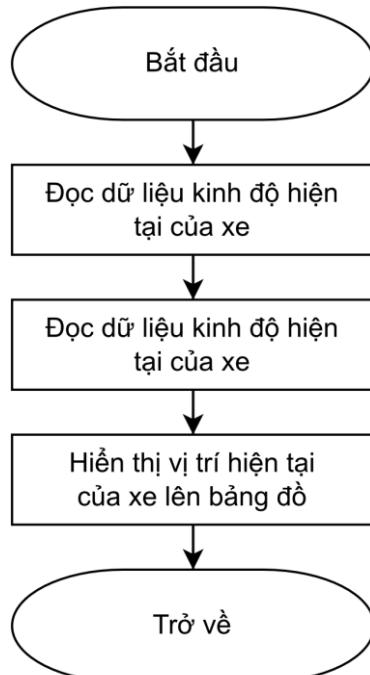
Khi hàm con này được gọi, chương trình sẽ thực hiện tuần tự các hoạt động.

Đọc dữ liệu từ các tệp tin lưu trữ dữ liệu nhiên liệu động cơ, tiến hành kiểm tra nhiên liệu động cơ có nằm trong khoảng 80-100% hay không, nếu đúng tiến hành bật đèn trạng thái màu xanh và chú thích trạng thái “HIGH”, sau đó đặt lại biến đếm bằng 0 và trở về chương trình chính.

Trong trường hợp mức nhiên liệu không nằm trong khoảng 80-100% kiểm tra tốc độ động cơ có nằm trong khoảng 20-80% hay không, nếu đúng tiến hành bật đèn trạng thái màu vàng và chú thích trạng thái “NORMAL” sau đó đặt lại biến đếm bằng 0, trở về chương trình chính.

Trong trường hợp mức nhiên liệu không nằm trong khoảng 80-100% và cũng không nằm trong khoảng 20-80%, tiến hành bật đèn trạng thái màu đỏ và chú thích trạng thái “LOW”, sau đó tăng biến đếm lên một, kiểm tra xem biến đếm có bằng một hay không, nếu có hiển thị cảnh báo nếu không quay về chương trình chính.

- **Lưu đồ hàm con hiển thị vị trí của xe.**

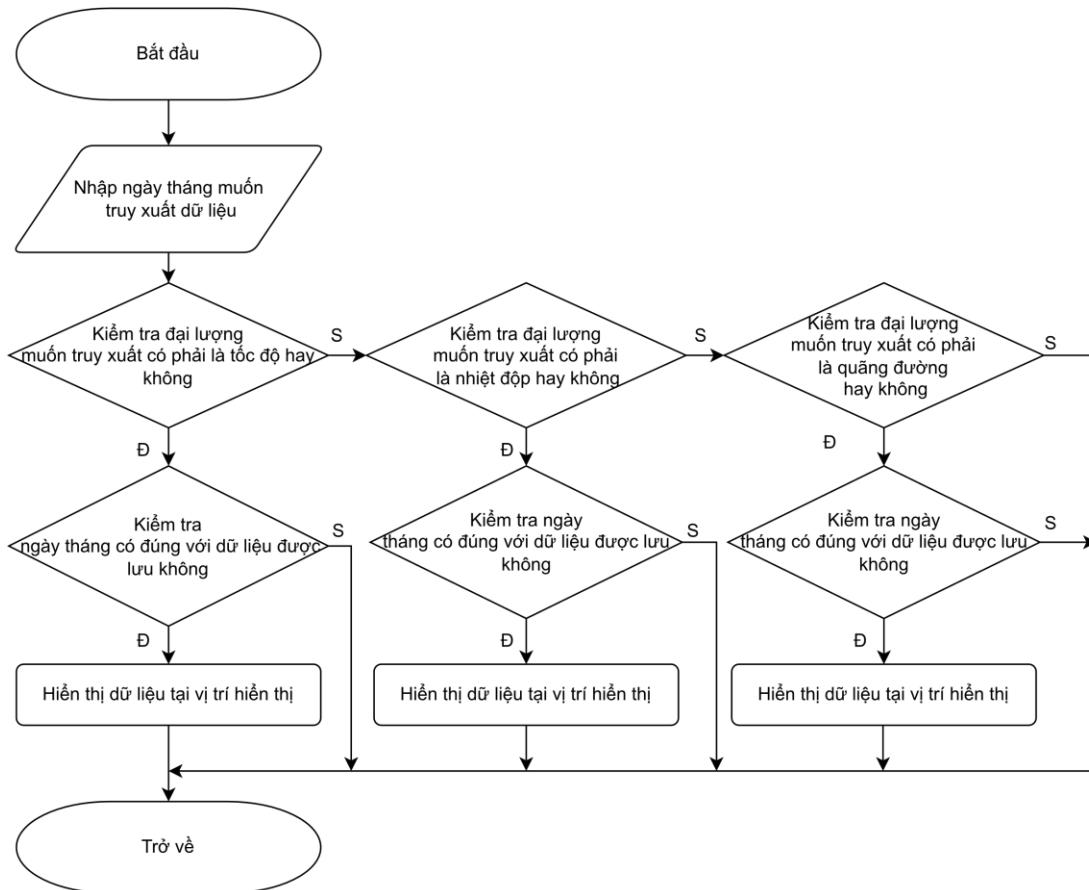


Hình 3.23: Lưu đồ hàm con hiển thị vị trí của phương tiện

Khi hàm con này được gọi, chương trình sẽ thực hiện tuần tự các hoạt động.

Đọc dữ liệu kinh độ và vĩ độ hiện tại của phương tiện được lưu trữ trong các tệp lưu trữ, tiến hành đưa dữ liệu vào bảng đồ và hiển thị vị trí hiện tại của phương tiện.

- **Lưu đồ hàm con trích xuất dữ liệu.**



Hình 3.24: Lưu đồ hàm con trích xuất dữ liệu

Khi hàm con này được gọi, chương trình sẽ thực hiện tuần tự các hoạt động.

Đọc dữ liệu thời gian muốn trích xuất được người dùng nhập từ bàn phím.

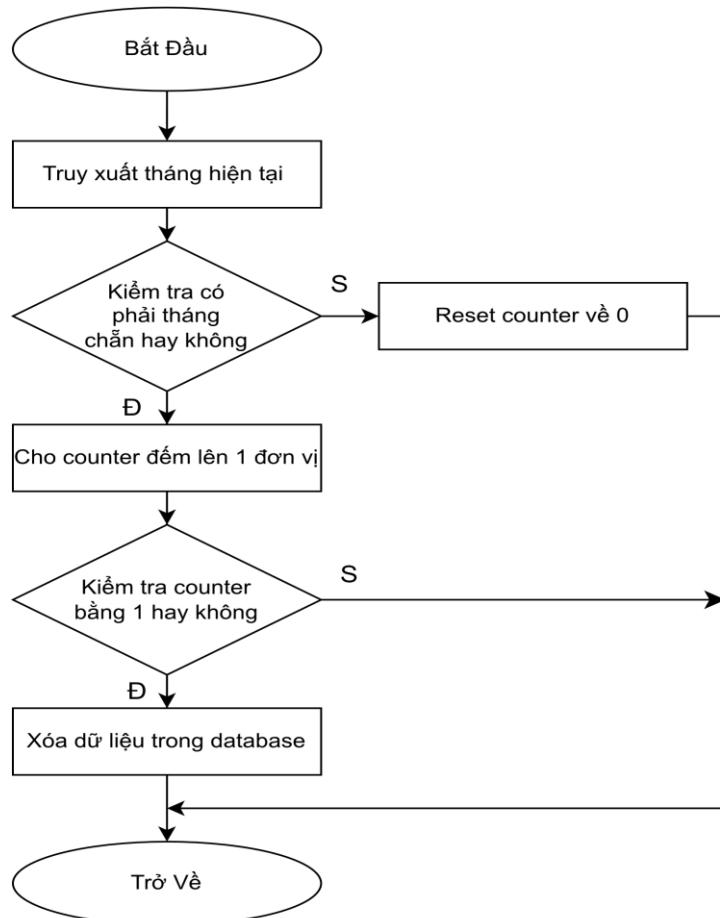
Kiểm tra xem người dùng có muốn trích xuất tốc độ động cơ hay không, nếu có kiểm tra thời gian muốn trích xuất có phù hợp với dữ liệu được lưu không, nếu đúng hiển thị bảng dữ liệu nếu không đúng thì quay về chương trình chính.

Kiểm tra xem người dùng có muốn trích xuất nhiệt độ động cơ hay không, nếu có kiểm tra thời gian muốn trích xuất có phù hợp với dữ liệu được lưu không, nếu đúng hiển thị bảng dữ liệu nếu không đúng thì quay về chương trình chính.

Kiểm tra xem người dùng có muốn trích quãng đường đi được hay không, nếu có kiểm tra thời gian muốn trích xuất có phù hợp với dữ liệu được lưu không, nếu đúng hiển thị bảng dữ liệu nếu không đúng thì quay về chương trình chính.

Nếu người dùng không chọn một trong ba lựa chọn trên thì trở về chương trình chính.

- **Lưu đồ hàm con xóa dữ liệu tự động.**



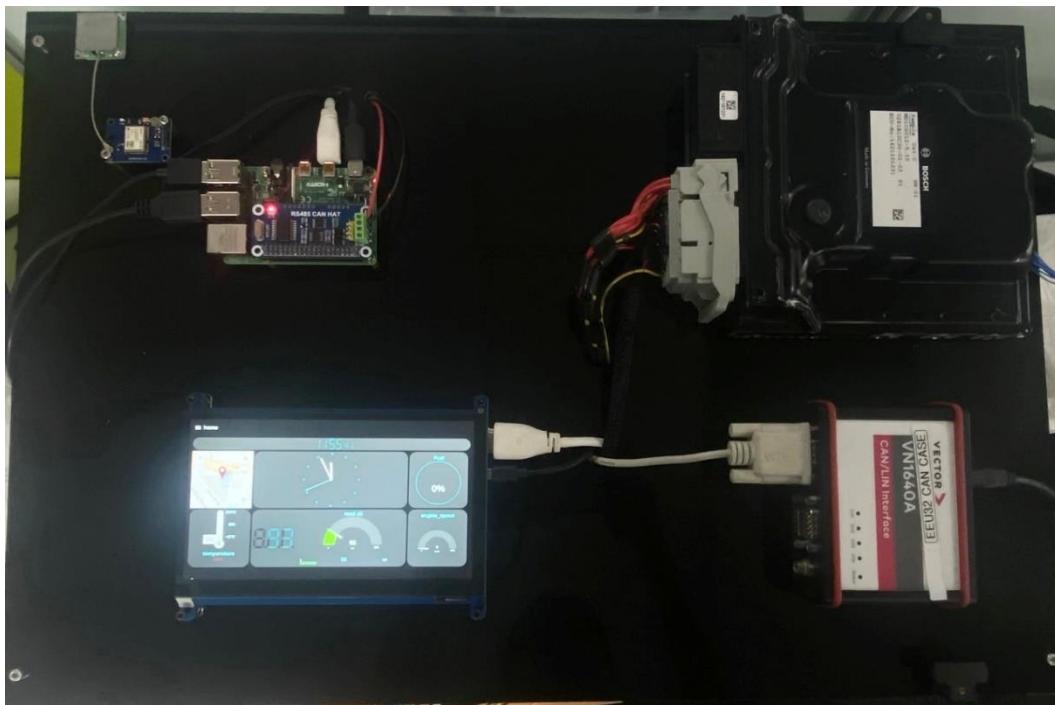
Hình 3.25: Lưu đồ hàm con xóa dữ liệu tự động

Đọc dữ liệu tháng hiện tại từ hàm con kiểm tra thời gian hiện tại, kiểm tra tháng có phải tháng chẵn hay không, nếu đúng cho biến đếm lên một đơn vị, kiểm tra biến đếm nếu biến đếm bằng 1 tiến hành xóa dữ liệu trên cơ sở dữ liệu, nếu biến đếm không bằng một trở về chương trình chính. Trong trường hợp tháng hiện tại không phải là tháng chẵn tiến hành đưa biến đếm về 0 và quay lại chương trình chính.

CHƯƠNG 4: KẾT QUẢ, ĐÁNH GIÁ HỆ THỐNG

4.1 KẾT QUẢ PHẦN CỨNG

Với việc đã lựa chọn được các linh kiện phần cứng để phù hợp với yêu cầu được đề ra từ trước khi thiết kế, nhóm sẽ đóng gói, bố trí hợp lý, đi dây lại các linh kiện để có thể ra được một sản phẩm gọn gàng, nhỏ gọn, tiện lợi.



Hình 4.1: Kết quả phần cứng hoàn chỉnh

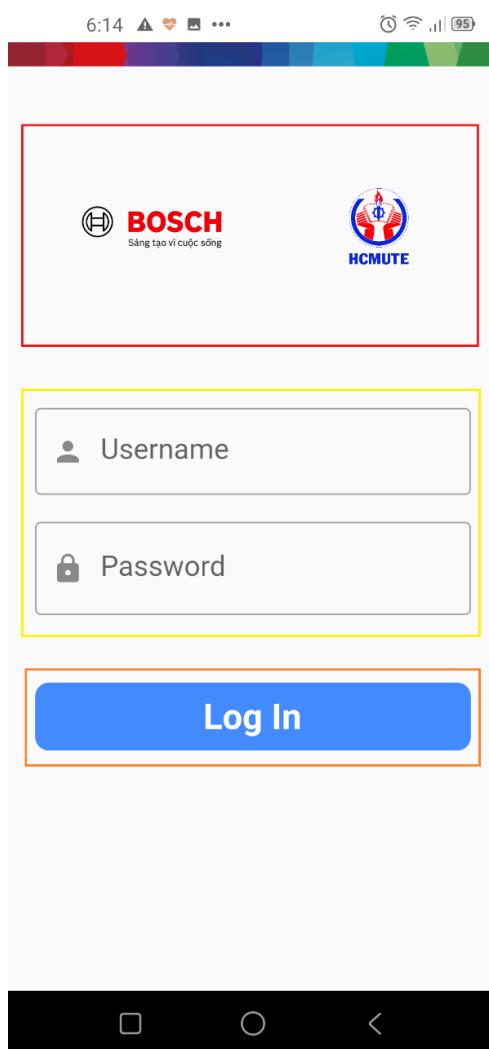
Đối với phần cứng thì nhóm đã thiết kế và đi dây dựa trên mô hình được giới thiệu cho khách hàng ở công ty Bosch. Với thiết kế theo một dạng sa bàn đặt các thiết bị phần cứng lên trên một bàn mà người dùng có thể quan sát được các thiết bị hoạt động cũng như có thể kiểm tra, theo dõi các linh kiện như ECU và cục Vector thuộc công ty Bosch giúp truyền gửi CAN Frame.

Đây là kết quả mô hình phần cứng hoàn chỉnh đã được nhóm đóng khung cũng như sắp xếp, đi dây hợp lý. Ngoài ra nhóm đã cố gắng thu gọn mô hình về nhỏ nhất so với mô hình thực tế được gắn trên xe từ ECU cho đến màn hình hiển thị trên xe.

4.2 GIAO DIỆN HIỂN THỊ TRÊN ỨNG DỤNG ĐIỆN THOẠI ANDROID

4.2.1 Hiển thị giao diện đăng nhập

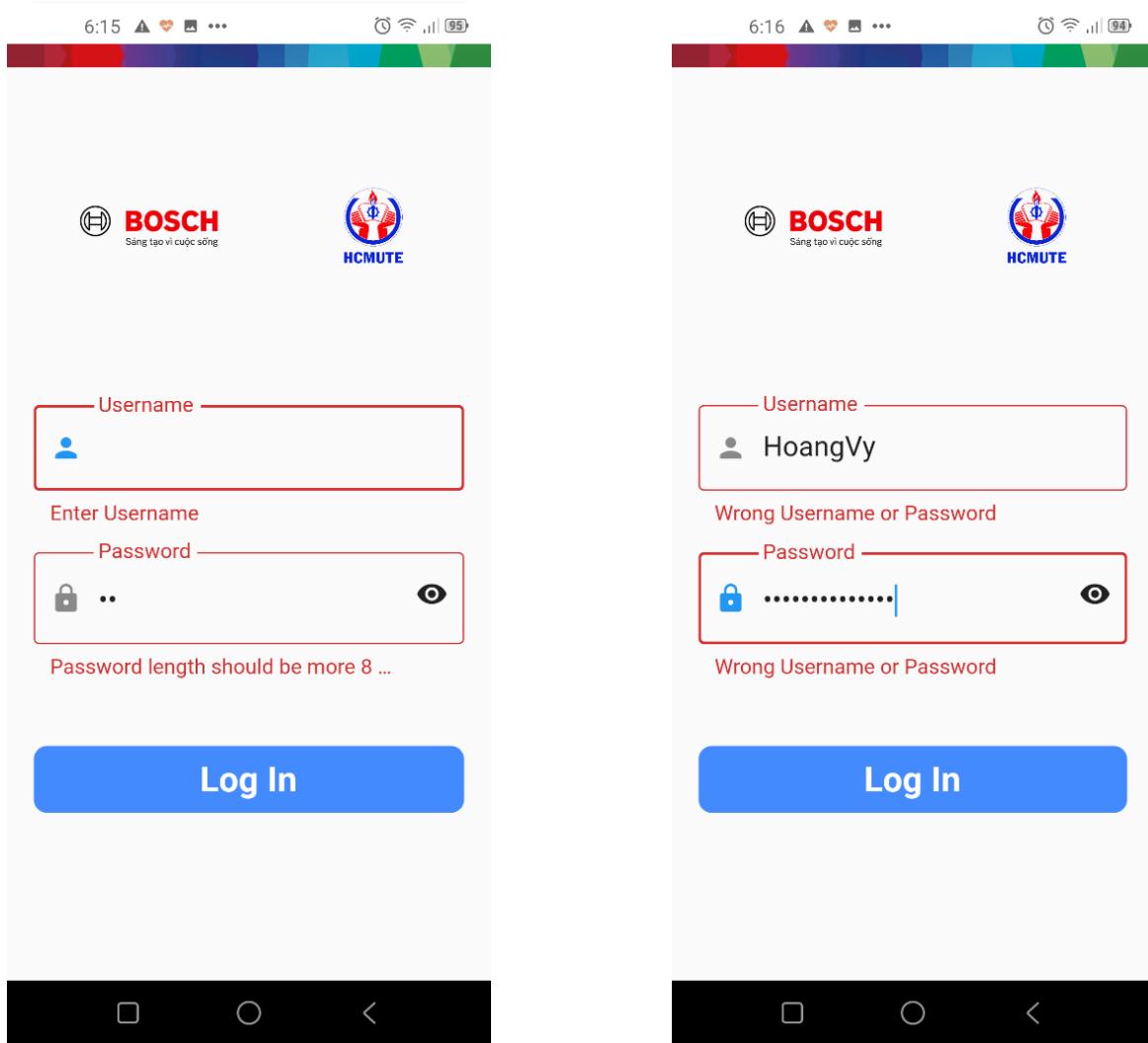
Giao diện đăng nhập sẽ giúp người dùng đảm bảo tính bảo mật khi sử dụng ứng dụng trên di động, đảm bảo an toàn thông tin cho người dùng cũng như hạn chế tối đa đối với việc người ngoài có thể sử dụng và theo dõi vào thông tin của xe.



Hình 4.2: Giao diện đăng nhập

Ở giao diện này sẽ được kết nối với Server thông qua API nhằm mục đích xác minh đúng với các tài khoản ở trên Database đang chứa hay không nếu được thì sẽ hiển thị tiếp theo đến trang chính còn ngược lại thì sẽ hiển thị sai mật khẩu hoặc tên đăng nhập. Ở phần

trên thì nhóm đã sử dụng cả hai logo một là logo của trường UTE và logo của công ty Bosch đã hỗ trợ các thiết bị, công cụ để nhóm có thể làm được đề tài này. Hai khung dưới sẽ là nơi nhập Username và Password mà giả định rằng chủ xe đã được cấp bởi nhà cung cấp xe, tiếp đến là nút Log In để có thể đăng nhập vào trang tiếp theo.



(a) Một số quy tắc khi đăng nhập

(b) Khi nhập sai Username hay Password

Hình 4.3: Các thông tin cần cho đăng nhập

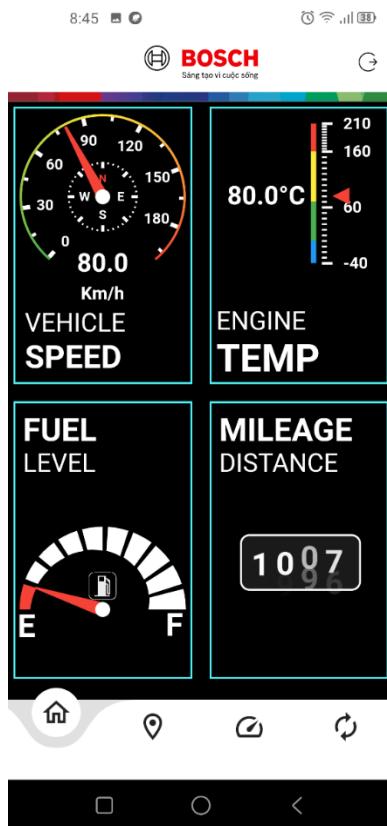
Với Hình 4.3 (a) thì người dùng bắt buộc phải nhập cả Username và Password không được để trống cũng như là mật khẩu phải thỏa ràng buộc như sau đó là phải hơn 8 ký tự.

Ngoài ra còn một nút ở dưới khung đăng nhập có biểu tượng hình con mắt để tiện lợi cho người dùng trong việc cần xem mật khẩu đã đúng chưa từ đó có thể sửa lại.

Sau khi đã nhập thỏa được các ràng buộc về mật khẩu thì có thể ấn nút Log In màu xanh ở dưới để có thể chuyển sang trang tiếp theo. Khi nhấn đăng nhập thì hệ thống sẽ kiểm tra các dữ liệu có trùng khớp với các thông tin trên Database không, nếu không thì trên hệ thống giao diện sẽ xuất hiện dòng thông báo rằng đã sai tên người dùng hoặc mật khẩu cần phải nhập lại cho chính xác.

4.2.2 Giao diện hiển thị các thông số

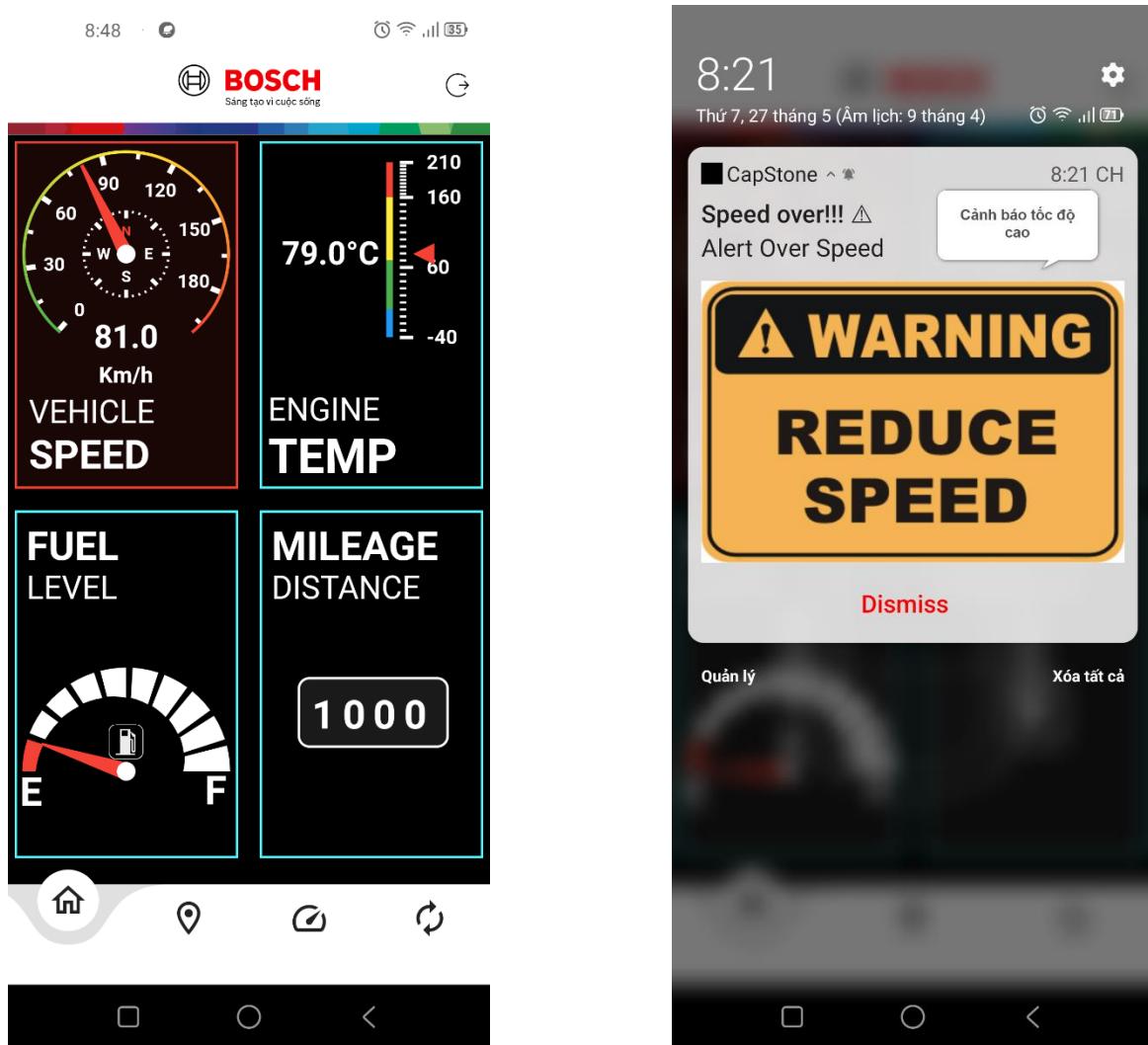
Đối với việc đăng nhập thành công thì ứng dụng sẽ trực tiếp chuyển sang trang giao diện hiển thị các thông số. Đặc biệt khi người dùng đã đăng nhập vào được ứng dụng điện thoại thì lần đăng nhập sau sẽ không cần đăng nhập lại và chuyển thẳng vào trang giao diện hiển thị các thông số.



Hình 4.4: Giao diện hiển thị các thông số

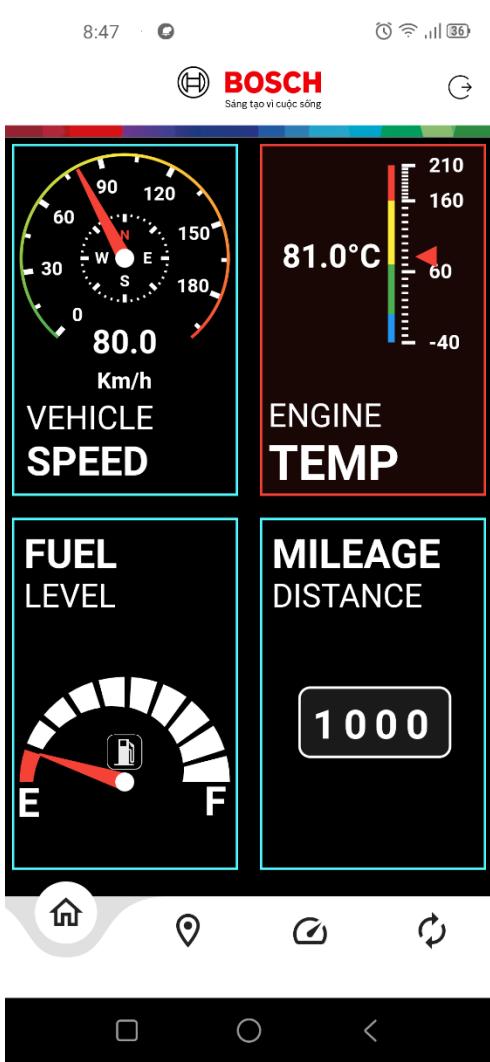
Dựa vào Hình 4.4 trên, giao diện sẽ hiển thị các thông số như tốc độ xe, nhiệt độ động cơ, mức nhiên liệu và quãng đường đi được tương ứng với 4 ô khác nhau đi theo thứ tự trên xuống và từ trái qua phải, ngoài ra còn có cách hiển thị các thông số dựa trên các hoạt ảnh động giúp tạo ấn tượng đối với người dùng.

Có thể thấy được thì ô đầu tiên sẽ là một đồng hồ hiển thị tốc độ hiện tại của xe, ứng dụng sẽ nhận các giá trị thông qua chuẩn truyền MQTT được gửi thông qua một MQTT Broker nên hầu như độ trễ sẽ rất thấp.

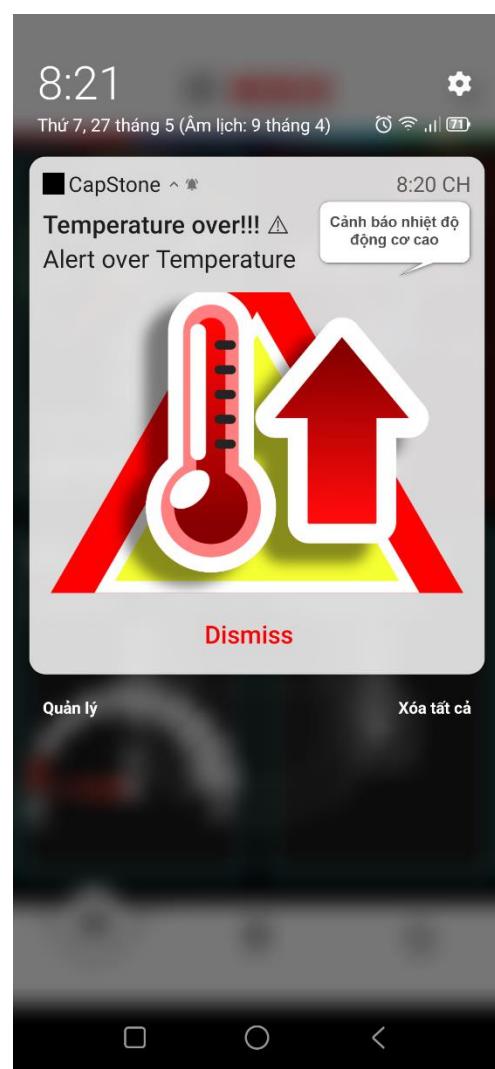


Hình 4.5: Giao diện và tính năng thông số tốc độ xe

Khi tốc độ của xe đã đạt tới một ngưỡng cao nhất định với giá trị được thiết lập đó là trên 80 Km/h thì trên khung của ô sẽ đổi sang màu đỏ để cảnh báo về việc tốc độ cao. Ngoài ra trên ứng dụng sẽ gửi một thông báo về trên điện thoại nhằm cảnh báo, thông báo cho người dùng biết được rằng tốc độ xe đang đi rất cao, để hướng tới mục đích lái xe đúng tốc độ, đảm bảo an toàn cho người dùng. Tương tự với các thông số còn lại, khác biệt mỗi thông báo sẽ có những mục đích cảnh báo khác nhau để phù hợp với việc chẩn đoán đối với các thông số đó.



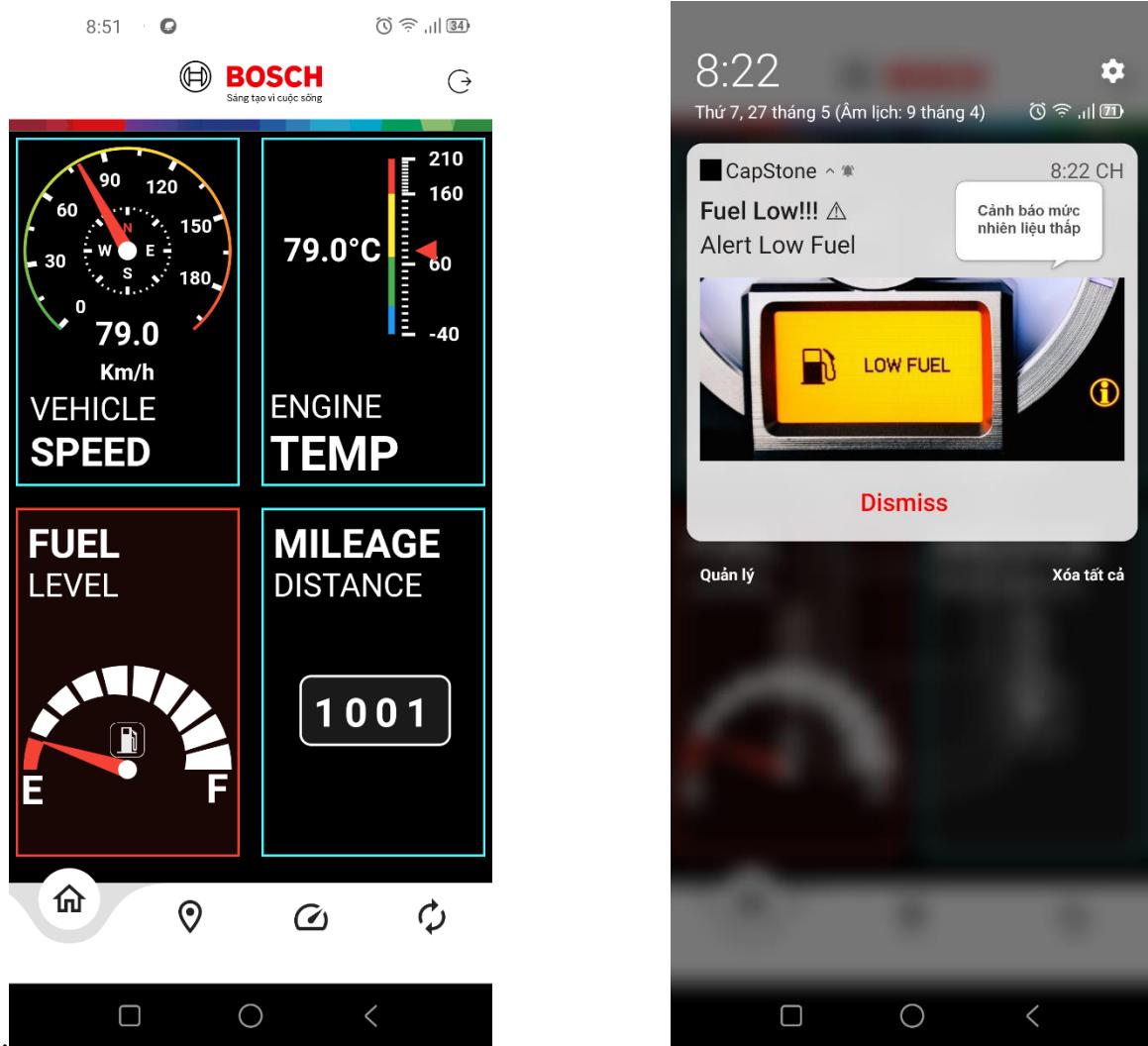
(a) Giao diện hiển thị khi nhiệt độ động cơ cao



(b) Thông báo khi nhiệt độ động cơ cao

Hình 4.6: Giao diện và cảnh báo thông số nhiệt độ động cơ

Nhiệt độ động cơ sẽ có giá trị nằm trong khoảng -40 đến 210 độ C và ngưỡng cảnh báo được thiết lập khi nhiệt độ động cơ vượt quá 80 độ C thì sẽ thay đổi khung ô trên giao diện cũng như là gửi các thông báo về cho điện thoại để nhằm cảnh báo về nhiệt độ cao của động cơ trong xe



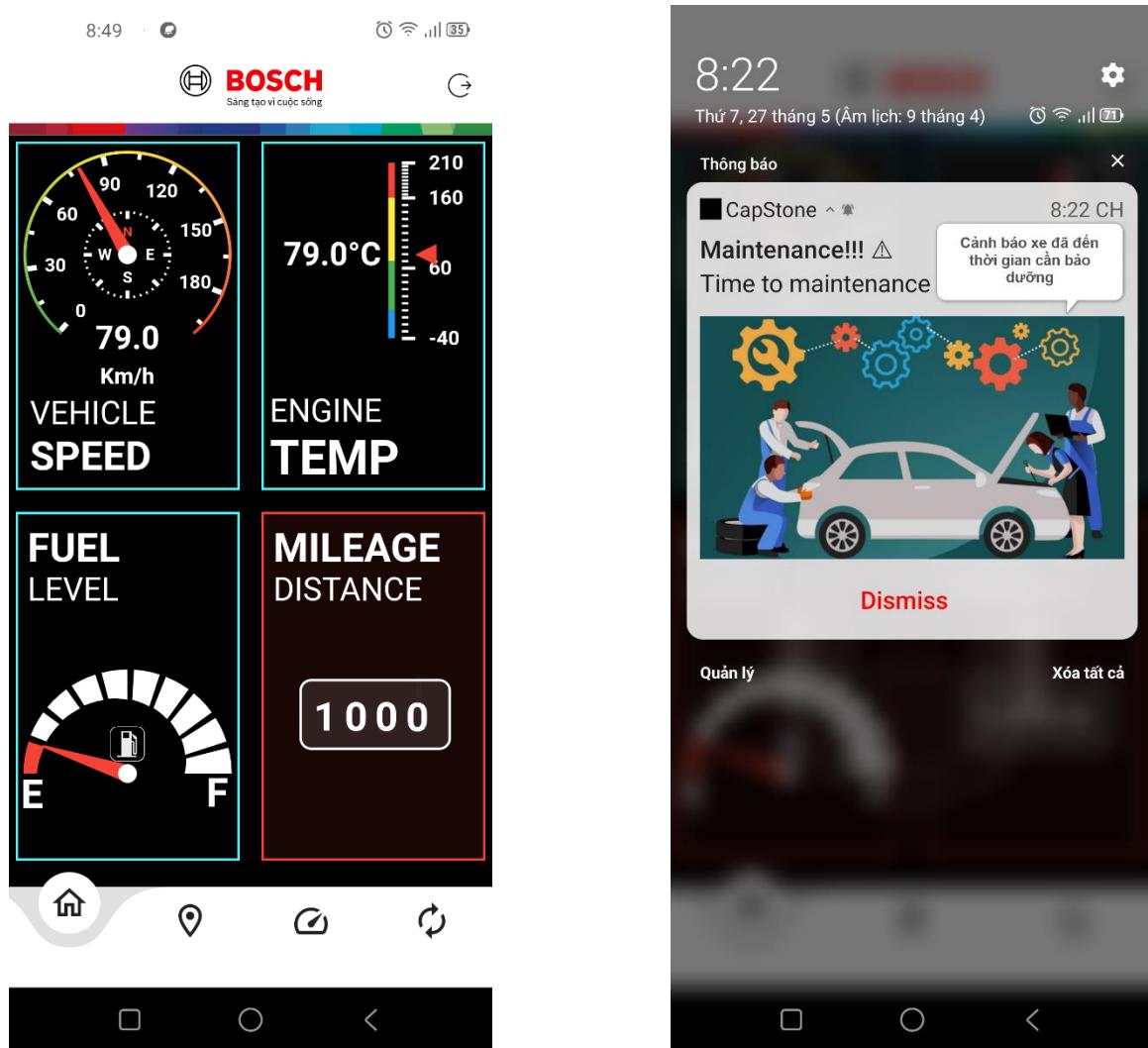
(a) Giao diện hiển thị khi gần hết nhiên liệu

(b) Thông báo khi gần hết nhiên liệu

Hình 4.7: Giao diện và cảnh báo thông số mức nhiên liệu

Với mức nhiên liệu dựa theo chuẩn J1939 đã đưa ra thì trung bình giá trị tiêu thụ nhiên liệu là 3,212 L/H và chuyển đổi để ra được giá trị phần trăm từ đó có thể xác định

được mức tiêu thụ nhiên liệu dễ dàng hơn, trên ứng dụng điện thoại sẽ gửi các thông báo cũng như là đổi màu của khung như trên Hình 4.7b khi mức tiêu thụ thấp hơn 20%. Để có thể cảnh báo cho người dùng về việc nạp liệu nhiên liệu tránh khi đang lưu thông sẽ hết nhiên liệu.



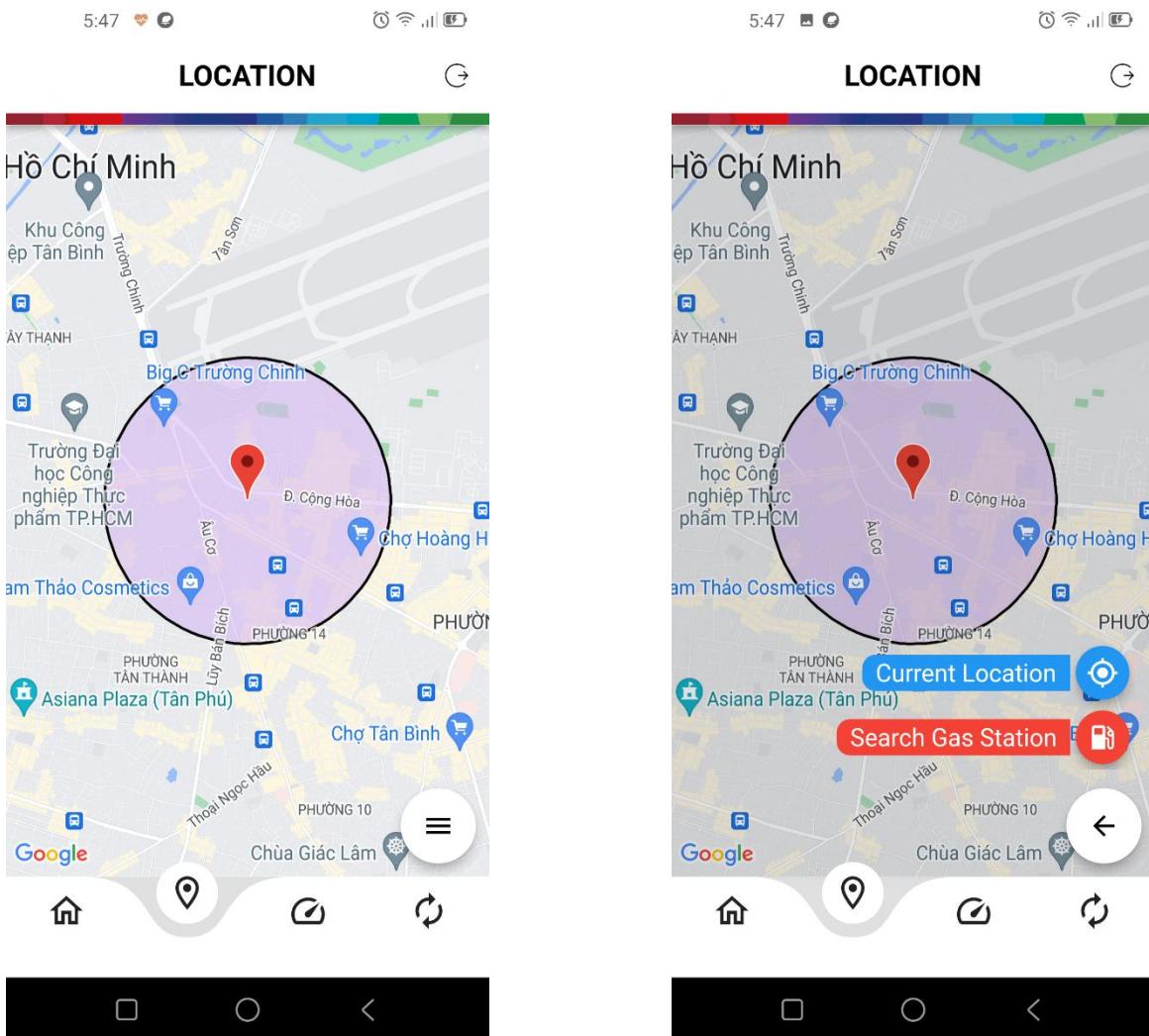
Hình 4.8: Giao diện và cảnh báo thông số quãng đường đi được

Về quãng đường xe đã đi được thì hiện tại trên ECU sẽ có mặc định một giá trị nhất định được lưu lại trong ECU nên chỉ cần chuyển đổi lại về theo chuẩn của J1939, khi xe

đã đi được hơn 200Km thì sẽ cảnh báo cũng như là thay đổi màu khung của ô với mục đích thông báo cho người dùng đã đến lúc cần phải đi bảo dưỡng cho xe.

4.2.3 Giao diện bản đồ

Ứng dụng điện thoại được tích hợp một bản đồ của Google nhằm hướng dẫn chỉ đường cũng như là hiển thị vị trí hiện tại của xe.



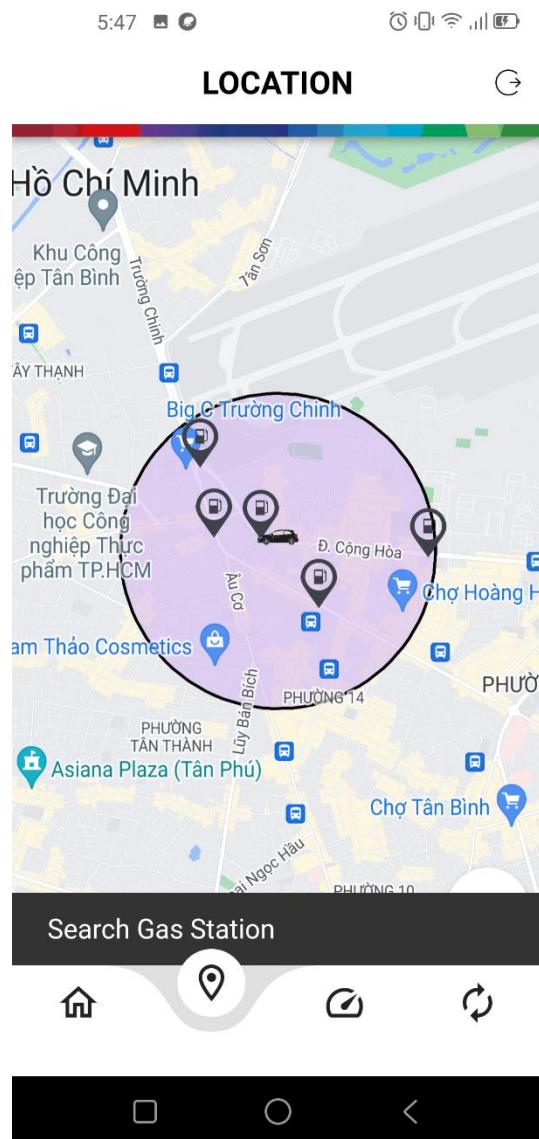
(a) Giao diện hiển thị bản đồ

(b) Giao diện hiển thị các nút chức năng

Hình 4.9: Tổng quan trang giao diện bản đồ

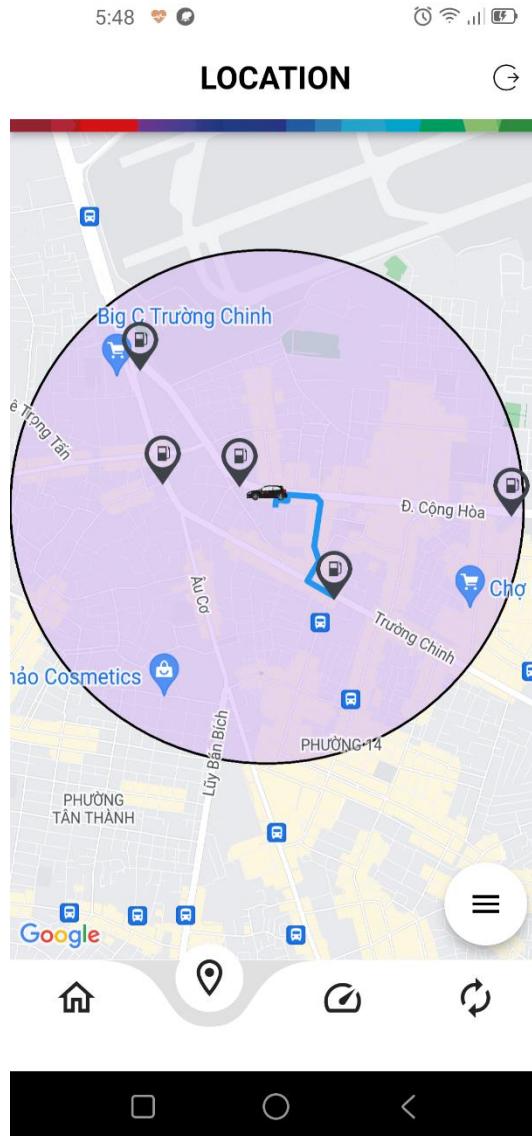
Với Marker màu đỏ sẽ là nơi hiện tại của phương tiện được xác định thông qua module NEO6M GPS, với hai nút nhấn “Current Location” và “Search Gas Station” sẽ cung cấp các chức năng khác nhau.

Khi ấn nút “Current Location” thì sẽ hiện lên vị trí hiện tại của phương tiện sẽ được cập nhật dựa trên các giá trị được gửi từ module GPS, khi người dùng xem bất kỳ ở đâu trên bản đồ thì khi ấn camera của bản đồ sẽ tự zoom về vị trí hiện tại của xe.



Hình 4.10: Khi ấn “Search Gas Station”

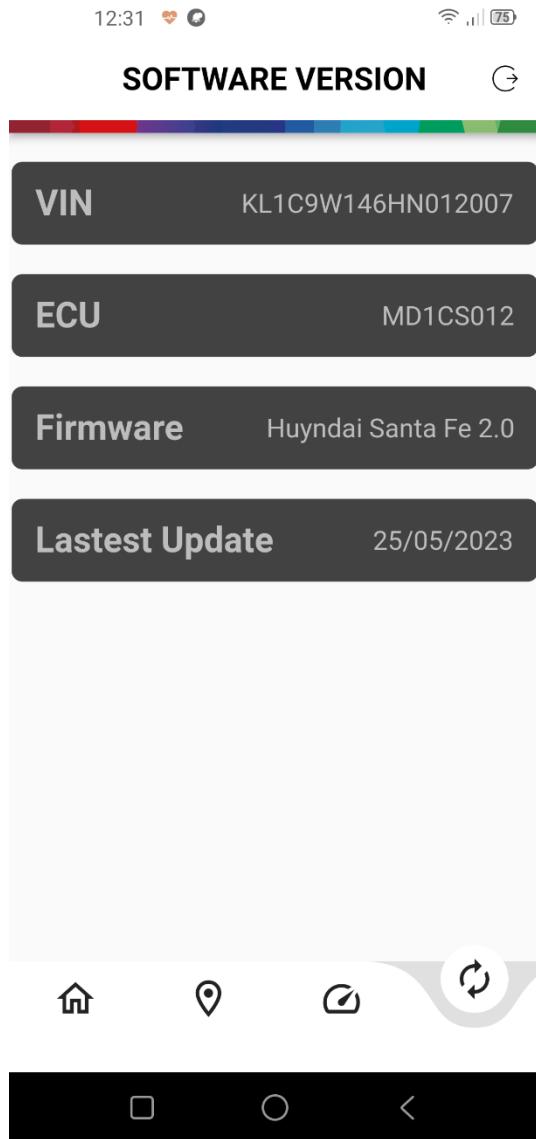
Khi ấn nút “Search Gas Station” sẽ hiển thị các Icon trạm xăng nằm bên trong vòng tròn màu tím như Hình 4.10, tùy thuộc vào người dùng có thể ấn vào từng Icon sẽ hiển thị các thông tin về của cây xăng như địa chỉ, tên cây xăng.



Hình 4.11: Khi ấn chỉ đường tới cây xăng

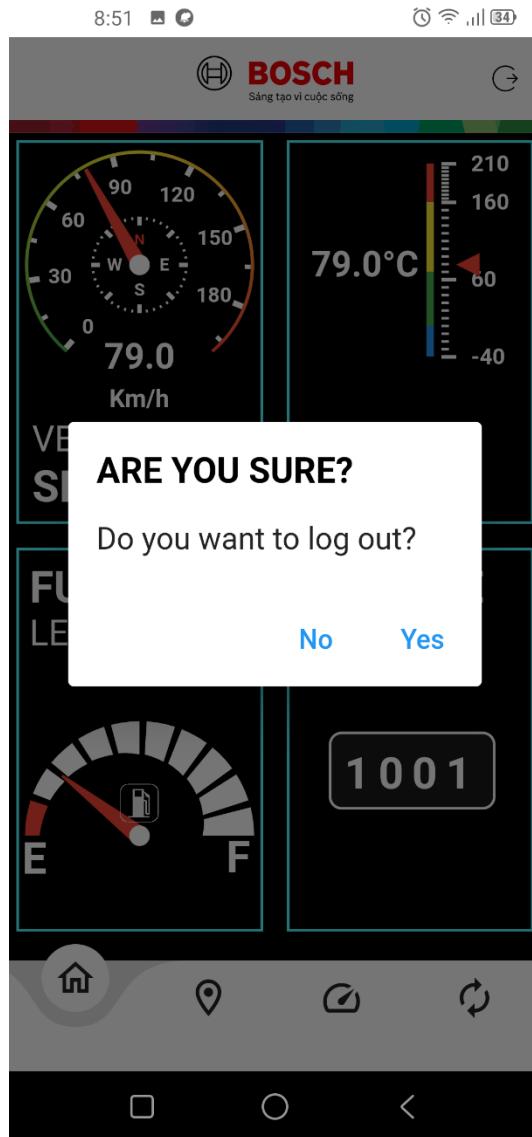
Sau khi người dùng ấn chọn cây xăng mà mình muốn đến thì trên ứng dụng sẽ chỉ đường đến cây xăng đó. Về mục đích của chức năng này thực tế khi kết hợp với giao diện thông số mức nhiên liệu ở trên khi mức nhiên liệu gần hết thì sẽ nhảy tới trang bản đồ để nhằm hướng dẫn, chỉ đường người dùng tới cây xăng gần nhất.

4.2.4 Giao diện thông tin của software



Hình 4.12: Giao diện hiển thị phiên bản của Software

Giao diện này sẽ hiển thị các thông tin chi tiết về các phiên bản mà ECU hiện tại đang sử dụng, với số VIN (Vehicle Identify Number) được nhà sản xuất thiết lập để nhằm mục đích cho việc quản lý cũng như thuận lợi cho việc Update Over The Air cho ECU. Tiếp đến sẽ là tên ECU được định theo tiêu chuẩn của Autosar, tùy thuộc vào dòng xe mà tên ECU sẽ được đặt theo cũng như là Firmware sử dụng cho dòng xe nào. Cuối cùng sẽ là hiển thị ngày tháng năm mà ECU được cập nhật phần mềm với phiên bản mới nhất hiện tại.



Hình 4.13: Chức năng đăng xuất

Như đã đề cập từ trước thì khi người dùng đăng nhập vào ứng dụng điện thoại thì lần vào lại ứng dụng tiếp theo sẽ không cần đăng nhập lại nữa, nhưng khi mà người dùng nhấn đăng xuất như Hình 4.13 trên thì lúc này ứng dụng sẽ không còn lưu tài khoản của người dùng nữa và trở lại về trang đăng nhập, lần vào lại sau sẽ phải đăng nhập lại.

4.3 GIAO DIỆN HIỂN THỊ TRÊN WEB

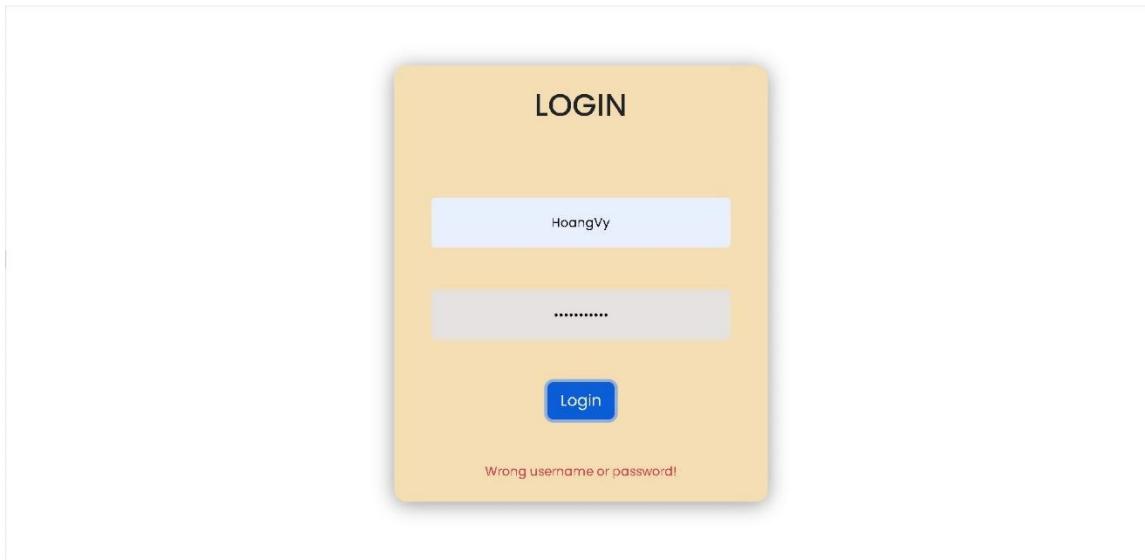
4.3.1 Giao diện đăng nhập



Hình 4.14: Giao diện đăng nhập

Giống như với tính năng đăng nhập có trên ứng dụng điện thoại thì ở web cũng sẽ có tính năng đăng nhập để người dùng có thể đăng nhập và sử dụng các tính năng có trong web. Cách vận hành của giao diện này cũng sẽ giống với cách vận hành hoạt động của ứng dụng trên điện thoại, chỉ khác cách bố trí cũng như màu sắc, sắp xếp các khối. Về tài khoản người dùng thì cũng sẽ được cấp bởi nhà cung cấp, vừa có thể dùng tài khoản đó đăng nhập vào ứng dụng trên điện thoại và cả trên web này. Để có thể gửi các thông tin và kiểm tra xem có đúng với các tài khoản được định sẵn trên database hay không, sẽ sử dụng giao thức HTTP, truyền gửi một cách nhanh chóng cũng như đảm bảo về tính bảo mật, an toàn.

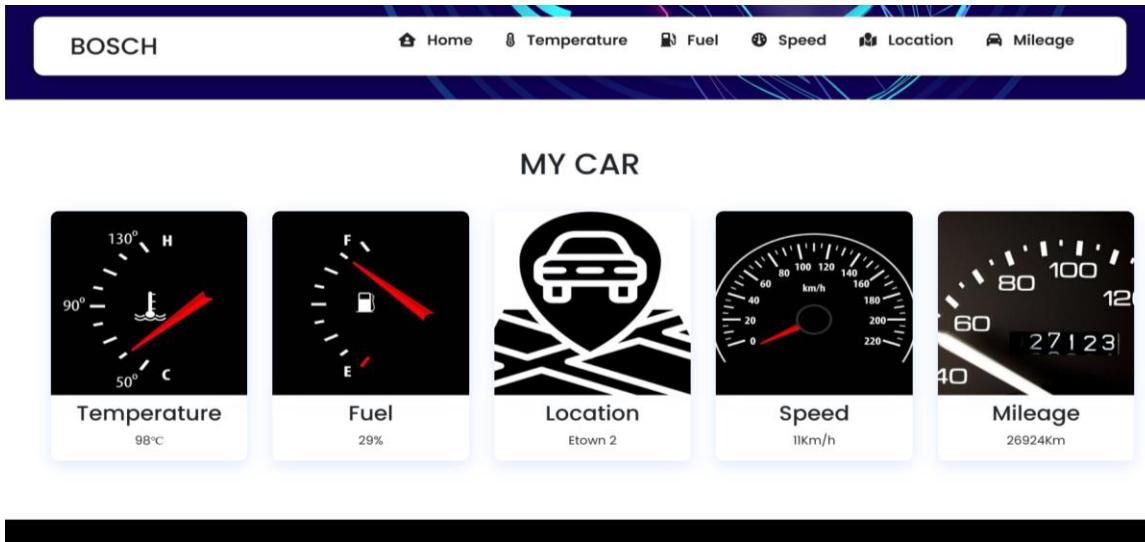
Cách để có thể nhận diện khi mình đăng nhập có đúng tài khoản hay mật khẩu không thì trên giao diện sẽ hiển thị. Chỉ khi đăng nhập sai thì trên giao diện sẽ hiện dòng chữ “Wrong username or password” như Hình 4.15 thì người dùng cần phải nhập đúng với tài khoản, ngược lại khi đăng nhập thành công thì sẽ chuyển thẳng vào trang sau.



Hình 4.15: Khi đăng nhập sai mật khẩu

4.3.2 Giao diện chính

Sau khi người dùng đã đăng nhập thành công dựa trên các giá trị trạng thái được gửi về từ database thì web sẽ chuyển sang hiển thị vào trang chính.

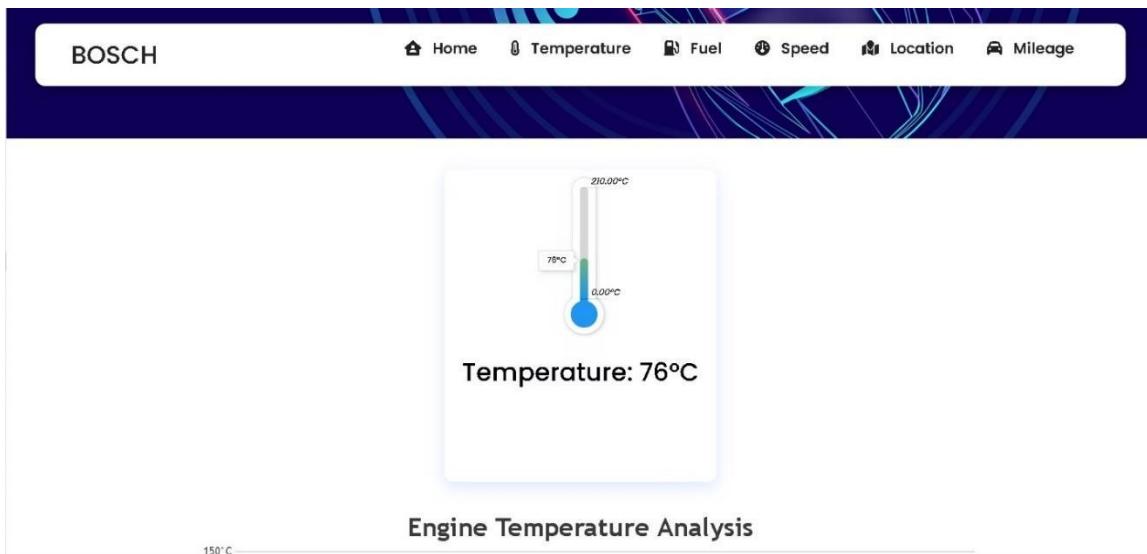


Hình 4.16: Giao diện chính

Ở giao diện Hình 4.16 sẽ hiển thị hết toàn bộ các thông số của xe như tốc độ xe, nhiệt độ động cơ, mức nhiên liệu và quãng đường đi được, đây cũng là trang chính nhằm giúp cho người dùng xem tổng quát về các giá trị thông số mà nhóm lựa chọn. Ngoài ra thì

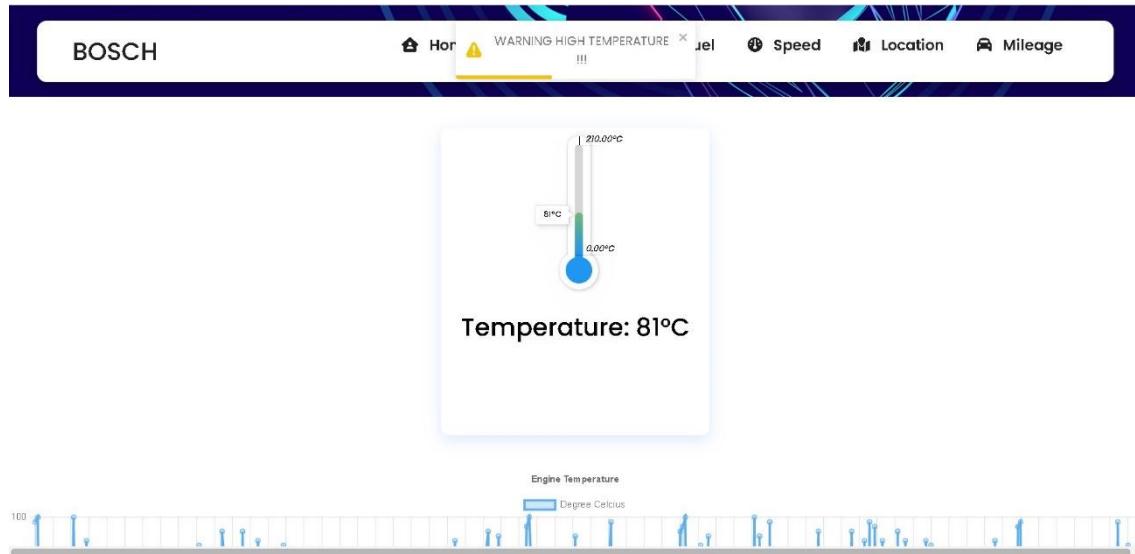
có thể ấn vào từng thông số để có thể xem chi tiết các thông số giao diện hiển thị như gauge, nhiệt kế cũng như có thể nhận được các thông báo chẩn đoán, cảnh báo đến cho người dùng.

- **Nhiệt độ động cơ**



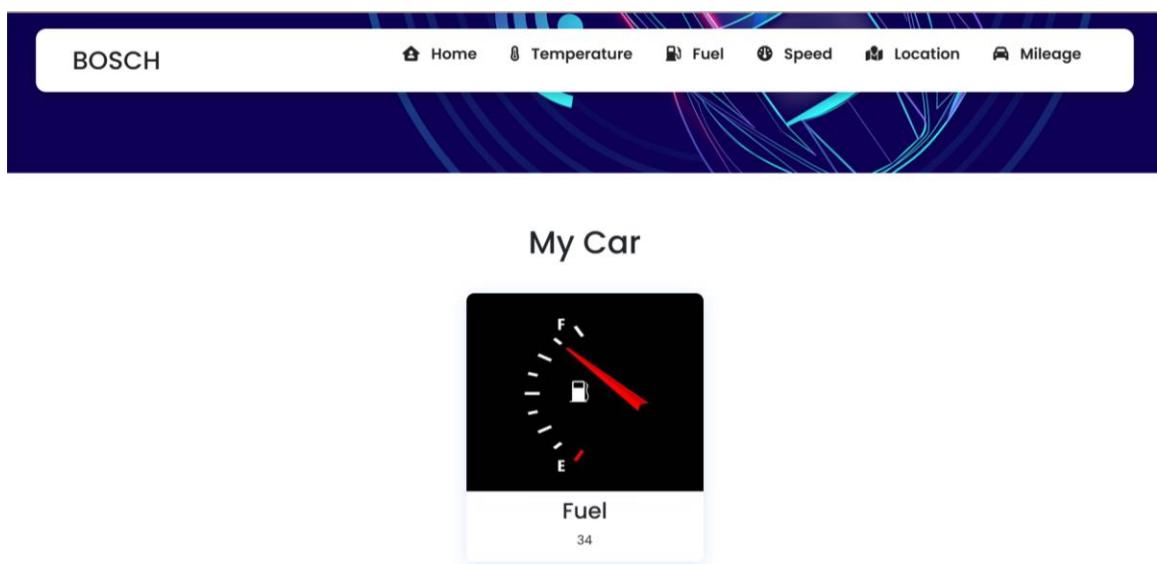
Hình 4.17: Nhiệt độ động cơ

Ở giao diện chính người dùng sẽ có thể ấn vào từng thông số muốn xem riêng biệt, khi lựa chọn thông số là nhiệt độ động cơ thì trên web sẽ hướng tới một giao diện tập trung hiển thị nhiệt độ động cơ cũng như là nếu có nhiệt độ cao hay quá nhiệt thì sẽ hiển thị toast để thông báo cho người dùng biết, quá nhiệt chỉ khi nhiệt độ động cơ lớn hơn 80°C. Về mục đích sẽ là báo hiệu cho người dùng về mức nhiệt độ của động cơ từ đó có thể xử lý kịp thời như đem đi sửa đúng lúc.



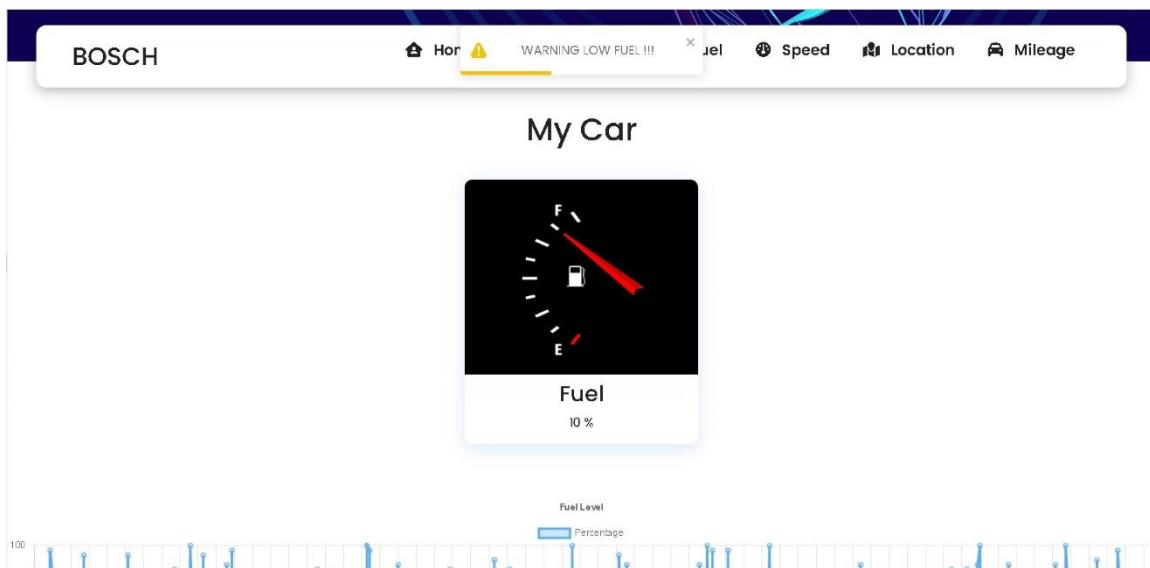
Hình 4.18: Cảnh báo khi nhiệt độ động cơ cao

- **Mức nhiên liệu**



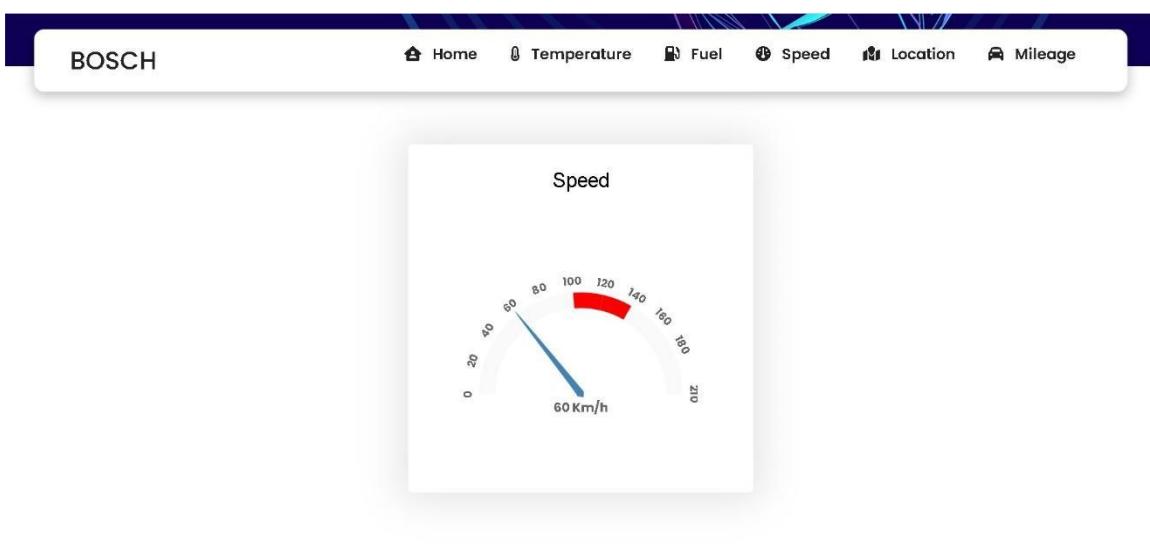
Hình 4.19: Mức nhiên liệu

Với mức nhiên liệu sẽ được hiển thị, khi người dùng ấn vào ô hiển thị mức nhiên liệu sẽ có giao diện như Hình 4.19 bên trên, thì với mức nhiên liệu xuống thấp hơn 20% thì trên web sẽ hiển thị toast thông báo, cảnh báo cho người dùng biết xe sắp hết nhiên liệu để có thể đi đổ xăng hay nạp nhiên liệu lại sớm nhất, để tránh cho việc bị hết nhiên liệu trong lúc sử dụng xe.



Hình 4.20: Cảnh báo khi nhiên liệu gần hết

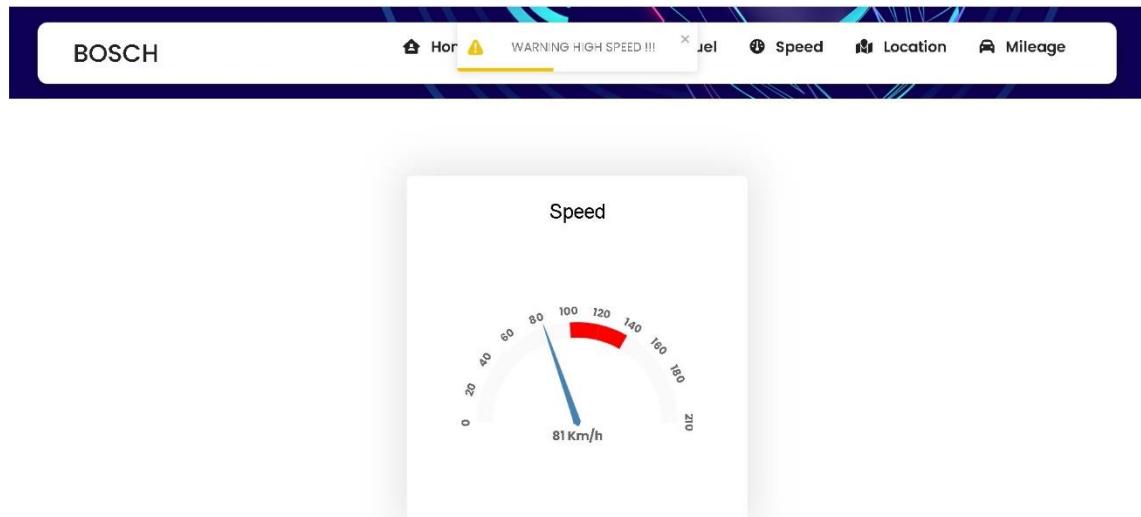
- **Tốc độ xe**



Hình 4.21: Tốc độ hiện tại của xe

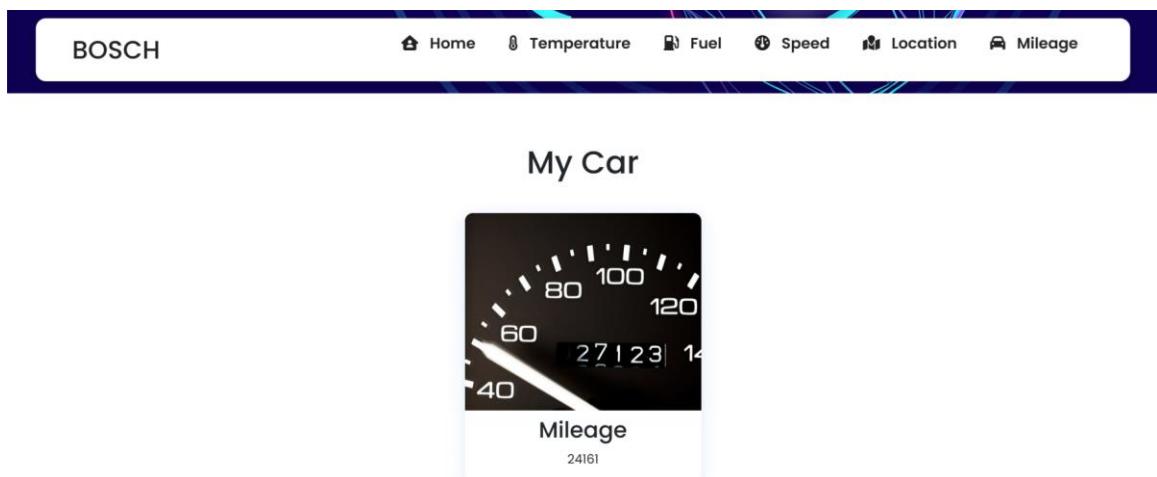
Tương tự với các thông số khác thì giao diện hiển thị tốc độ hiện tại của xe với Gauge như Hình 4.21 các giá trị thông số sẽ được cập nhật và nhận liên tục thông qua chuẩn giao thức MQTT dựa trên một MQTT Broker. Khi tốc độ xe đang chạy rất cao ở đây được set là trên 80km/h thì trên web sẽ hiển thị một cảnh báo về tốc độ cao để người dùng

có thể xem và giảm tốc độ hạn chế gây tai nạn cũng như là tránh những việc vi phạm luật giao thông.



Hình 4.22: Cảnh báo tốc độ xe cao

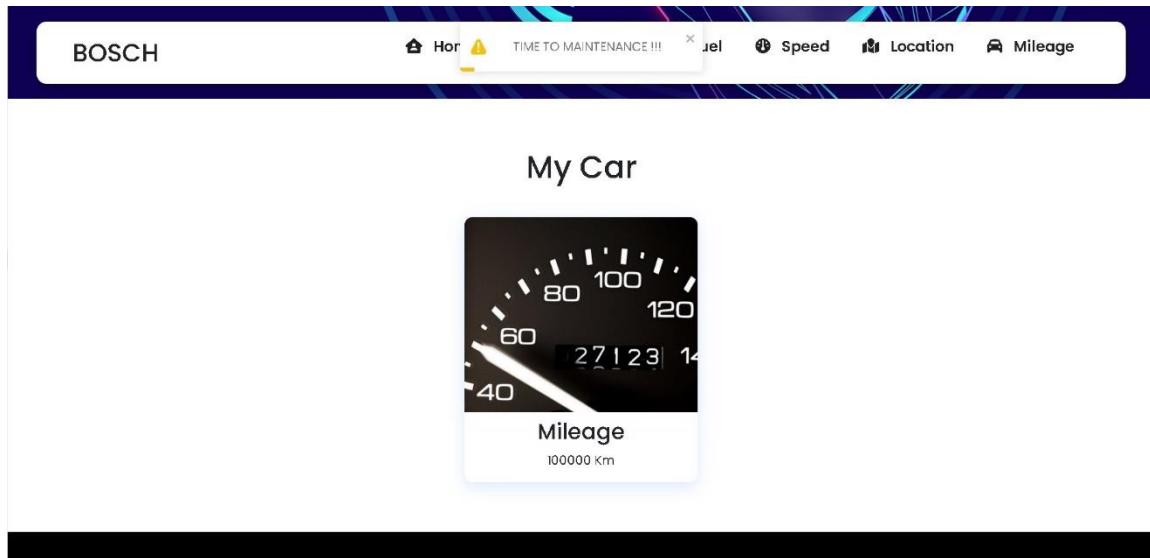
- **Quãng đường xe đã đi được**



Hình 4.23: Thông số quãng đường xe đã đi được

Giao diện này sẽ hiển thị giá trị quãng đường mà xe đã đi được, đây được xem như là giá trị của đồng hồ “Công-tơ-mét” trên xe. Dựa vào các giá trị có được từ ECU gửi ra thì khi mà xe di chuyển hơn 200km thì sẽ có một cảnh báo đã tới thời điểm xe nên được đi

bảo trì, nhằm có thể hạn chế việc hư hỏng của xe cũng như là kiểm tra và xử lý kịp thời nếu như bị hư hỏng. Giúp người dùng có thể tiện lợi và giám sát được tình trạng hiện tại của xe.



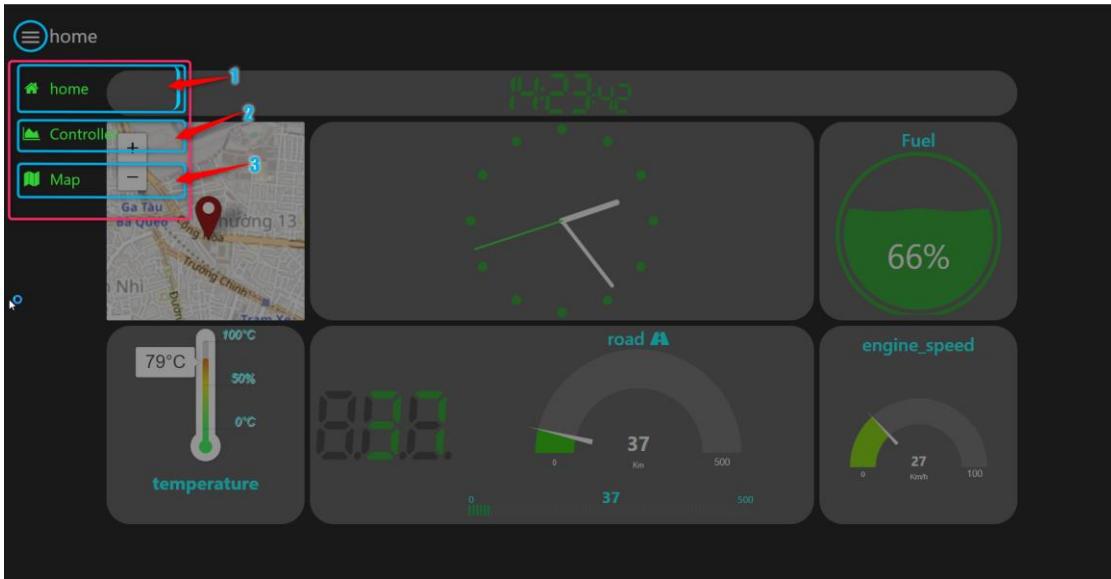
Hình 4.24: Thông báo tới lúc xe cần bảo trì, bảo dưỡng

4.4 GIAO DIỆN HIỂN THỊ GUI TRÊN MÀN HÌNH 7INCH

Giao diện GUI được thiết kế nhằm để người sử dụng có thể theo dõi động cơ trực tiếp khi đang di chuyển mà không cần thông qua web và app, đồng thời hạn chế được các tác động về đường truyền mạng internet có thể ảnh hưởng đến độ chính xác của hệ thống.

Giao diện được thiết kế trên các tiêu chí:

- +Hiển thị đúng, đủ các thông số đặc trưng của động cơ.
- +Thiết kế đơn giản, dễ tương tác với người dùng.

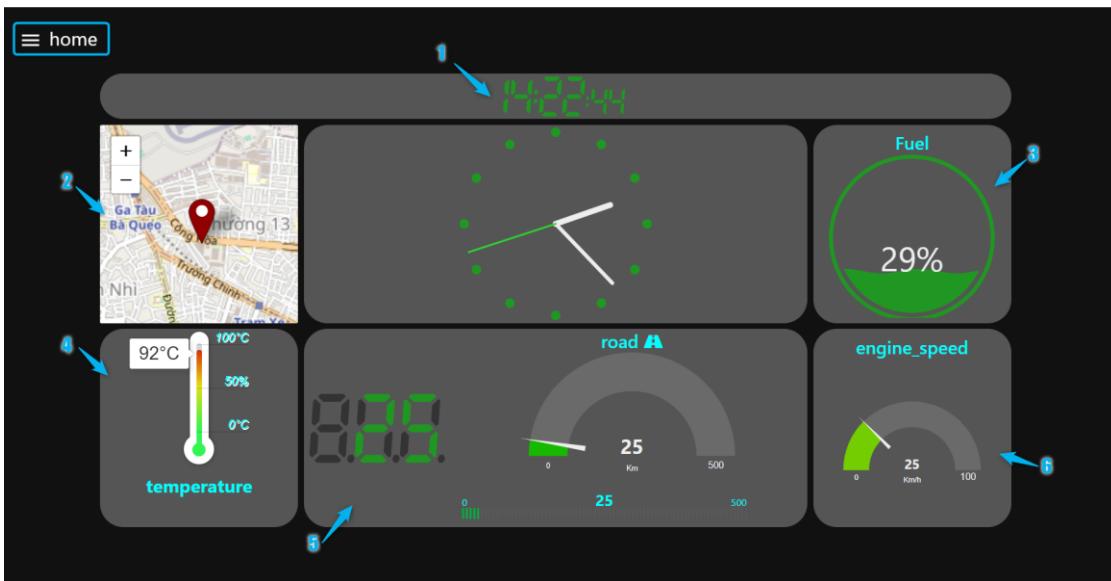


Hình 4.25: Giao diện lựa chọn các tab của GUI.

Giao diện GUI cần thể hiện nhiều thông tin nhưng vẫn phải đảm bảo tính thẩm mỹ cũng như mức độ tương tác đơn giản, vì thế chúng tôi đã chia tác thông tin hiển thị ra làm ba trang chính, người dùng có thể lựa chọn hiển thị các trang áy thông qua nút chọn trang (góc trên bên trái màn hình), người dùng có thể lựa chọn các trang bao gồm:

1. Home, trang này thể hiện tổng thể các thông số động cơ.
2. Controller, ở trang này người dùng có thể chọn xem chi tiết từng thông số mà mình muốn theo dõi.
3. Map, được cấu hình như một bản đồ giúp người dùng định hình được vị trí của phương tiện và các con đường xung quanh.

4.4.1 Giao diện trang Home



Hình 4.26: Giao diện tổng thể GUI.

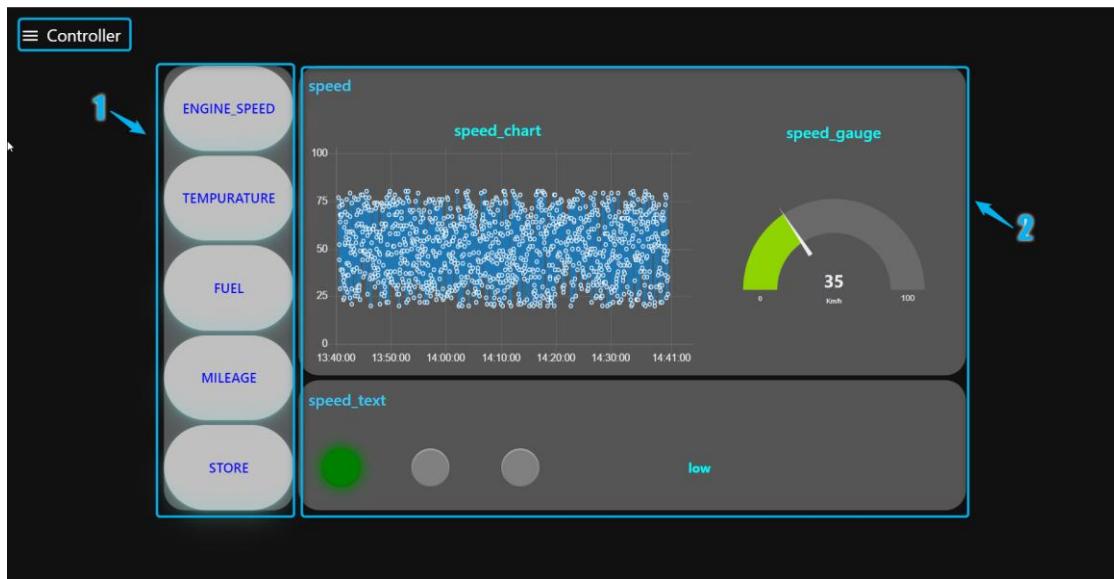
Trong giao diện trang “Home”, các thông số vật lý đặc trưng của động cơ được thu thập thông qua chuẩn J1939 sẽ được thể hiện. Các thông số của động cơ sau khi được chuyển đổi từ CAN sẽ được lưu vào các tệp lưu trữ dữ liệu, GUI sẽ đọc dữ liệu trực tiếp từ các tệp lưu trữ ấy và hiển thị lên trang “home”, thứ tự hiển thị lần lượt là:

1. Đồng hồ thể hiện thời gian thực.
2. Bản đồ thu nhỏ, thể hiện vị trí hiện tại của phương tiện.
3. Đồ thị hiển thị lượng nhiên liệu hiện tại của phương tiện, đơn vị (%).
4. Đồ thị hiển thị nhiệt độ hiện tại của động cơ, đơn vị (°C).
5. Đồ thị thể hiện quãng đường mà phương tiện đã đi được, đơn vị (Km).
6. Đồ thị hiển thị vận tốc của phương tiện ở thời điểm hiện tại, đơn vị (Km/h).

Vì GUI đọc dữ liệu trực tiếp từ các tệp lưu trữ mà không thông qua bất cứ giao thức truyền thông nào nhò vạy hạn chế được tối đa thời gian trễ và không cần phụ thuộc vào đường truyền mạng, đồng thời các tệp lưu trữ sẽ liên tục được ghi đè dữ liệu, tránh gây hiện tượng tràn bộ nhớ.

4.4.2 Giao diện trang Controller

Ở trang “Controller” người dùng có thể theo dõi chi tiết các thông số của động cơ, đồng thời có thể truy xuất các giá trị lớn nhất, nhỏ nhất và giá trị trung bình của một thông số vật lý bất kỳ từ dữ liệu đã được lưu trữ trên cơ sở dữ liệu.

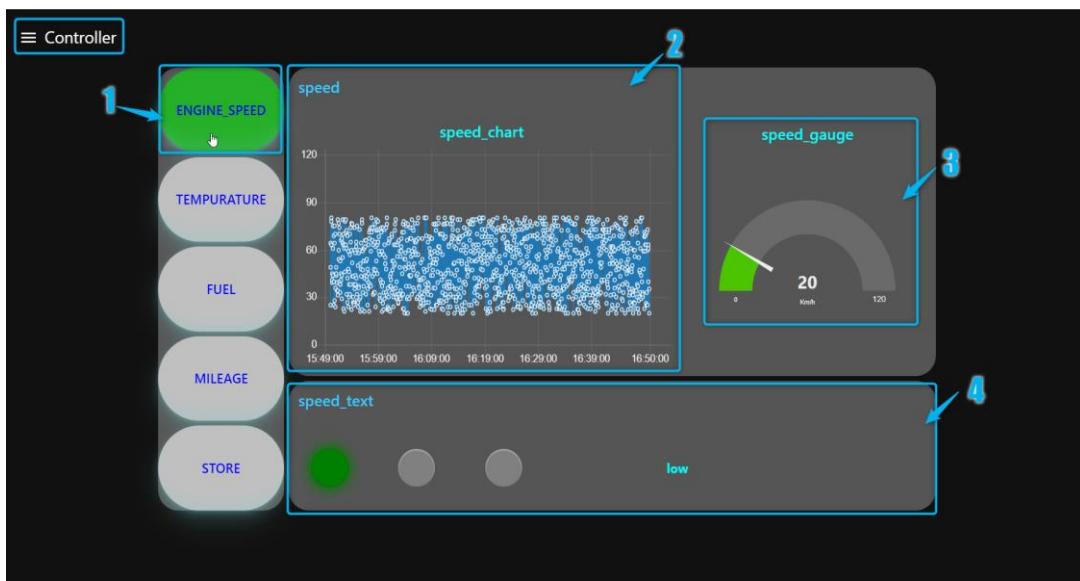


Hình 4.27: Các thành phần trong trang Controller.

Giao diện “Controller” gồm hai phần chính:

- Nhóm các nút nhấn nằm phía trái màn hình (đánh số 1 trên hình) đây là nhóm các nút nhấn để người dùng có thể lựa chọn thông số vật lý mà mình muốn theo giờ gồm các thông số (tốc độ của xe, nhiệt độ động cơ, nhiên liệu hiện tại động cơ, quãng đường đi được của phương tiện, Trích xuất lịch sử dữ liệu), trực hoành tương ứng với giá trị thời gian theo kiểu (giờ, phút, giây).
- Sau khi chọn thông số bằng các nút nhấn, chi tiết của thông số đó sẽ được hiện ra phía bên tay phải màn hình (đánh số 2 trên hình ...). Khi người dùng chọn trang “Controller” thông số mặc định là tốc độ của xe.

4.4.2.1 Tốc độ xe

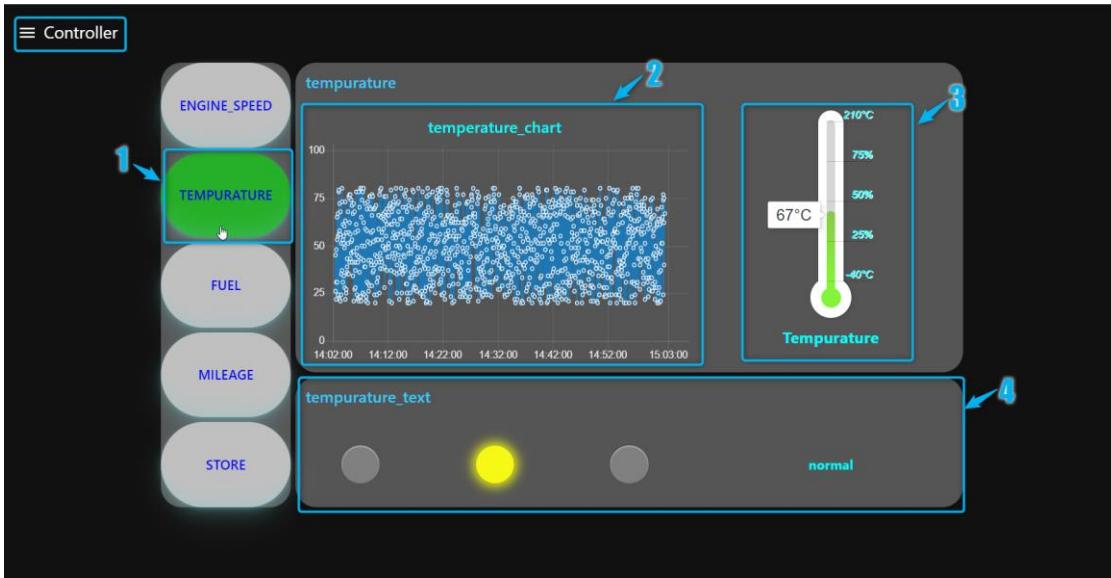


Hình 4.28: Thông số tốc độ xe

Giao diện chi tiết về thông số tốc độ của xe được hiện ra khi người dùng bấm chọn nút “Vehicle speed”. Giao diện tốc độ của xe bao gồm các thành phần chính.

1. Nút chọn theo dõi tốc độ của xe.
2. Biểu đồ hiển thị tốc độ của xe trong vòng 1 giờ tính từ khi người dùng chọn theo dõi tốc độ của xe. Biểu đồ được thể kế là dạng biểu đồ điểm với trực tung tương ứng với giá trị của tốc độ của xe (có giới hạn từ 0-120 Km/h).
3. Gauge thể hiện tốc độ của động cơ ở thời điểm hiện tại.
4. Dãy đèn báo gồm 3 đèn đỏ, xanh, vàng. Ba đèn này giúp người dùng có thể nhận biết được khi tốc độ của xe vượt quá ngưỡng cho phép. Ba đèn báo có các giá trị giới hạn theo thứ tự xanh, vàng, đỏ lần lượt là 0-40 (Km/h), 41-100 (km/h), 100-120 (km/h).
5. Chú thích thể hiện trạng thái của tốc độ của xe gồm các mức độ cảnh báo tốc độ thấp, vừa, cao lần lượt tương ứng với các ngưỡng 0-40 (Km/h), 41-100 (km/h), 100-120 (km/h).

4.4.2.2 Nhiệt độ động cơ.



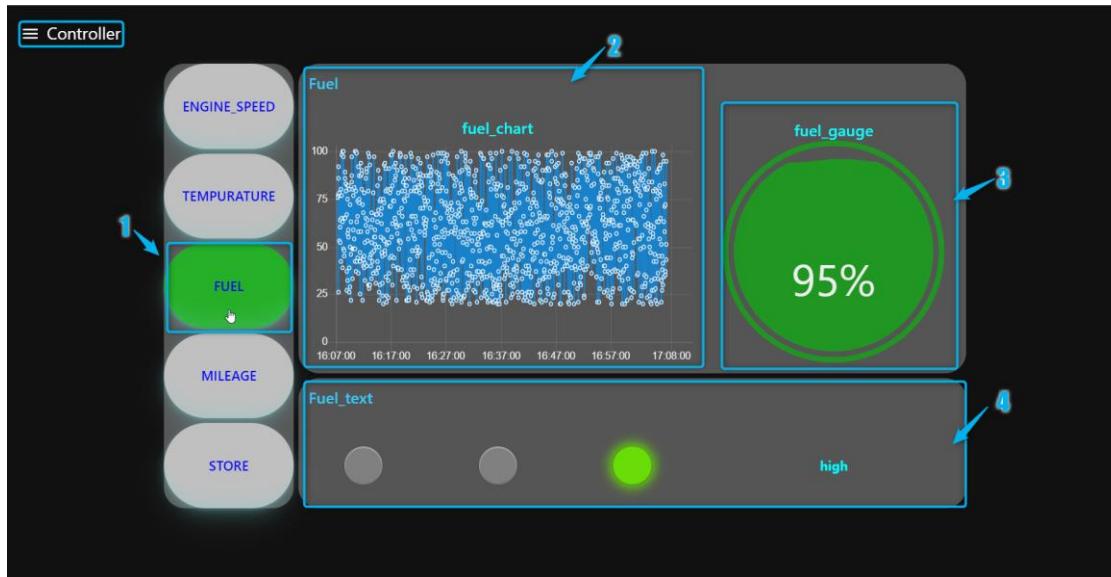
Hình 4.29: Thông số nhiệt độ động cơ

Giao diện chi tiết về thông số nhiệt độ động cơ được hiện ra khi người dùng bấm chọn nút “temperature”. Giao diện nhiệt độ động cơ bao gồm các thành phần chính.

1. Nút chọn theo dõi nhiệt độ động cơ.
2. Biểu đồ hiển thị nhiệt độ động cơ trong vòng 1 giờ tính từ khi người dùng chọn theo dõi nhiệt động cơ. Biểu đồ được thể kế là dạng biểu đồ điểm với trực tung tương ứng với giá trị của nhiệt độ động cơ (có giới hạn từ -40°C-210°C), trực hoành tương ứng với giá trị thời gian theo kiểu (giờ, phút, giây).
3. Gauge thể hiện nhiệt độ của động cơ ở thời điểm hiện tại, dạng gauge được chọn có biểu tượng thang đo nhiệt độ giúp người dùng dễ hình dung về đại lượng vật lý nhiệt độ, giới hạn của gauge là từ -40 -210(°C), gauge có khả năng chuyển đổi màu sắc tùy thuộc vào nhiệt độ tại các thời điểm khác nhau.
4. Dãy đèn báo gồm 3 đèn đỏ, xanh, vàng. Ba đèn này giúp người dùng có thể nhận biết được khi nhiệt độ động cơ vượt quá ngưỡng cho phép. Ba đèn báo có các giá trị giới hạn theo thứ tự xanh, vàng, đỏ lần lượt là -40-20 (°C), 21-70 (°C), 71-210 (°C).

- Chú thích thể hiện trạng thái của nhiệt độ động cơ gồm các mức độ cảnh báo nhiệt độ thấp, vừa, cao lần lượt tương ứng với các ngưỡng là -40-20 (°C), 21-70 (°C), 71-210 (°C).

4.4.2.3 Mức độ nhiên liệu động cơ.



Hình 4.30: Thông số mức độ nhiên liệu động cơ

Giao diện chi tiết về mức độ nhiên liệu động cơ được hiện ra khi người dùng bấm chọn nút “Fuel”. Giao diện mức độ nhiên liệu còn lại của động cơ bao gồm các thành phần chính.

- Nút chọn theo dõi mức độ nhiên liệu động cơ.
- Biểu đồ hiển thị mức độ nhiên liệu động cơ trong vòng 1 giờ tính từ khi người dùng chọn theo dõi mức độ nhiên liệu động cơ. Biểu đồ được thể kê là dạng biểu đồ điểm với trục tung tương ứng với giá trị mức độ nhiên liệu động cơ, mức độ nhiên liệu này đã được chuyển đổi về dạng đơn vị % với giới hạn từ 0%-100%, trục hoành tương ứng với giá trị thời gian theo kiểu (giờ, phút, giây).
- Gauge thể hiện mức độ nhiên liệu ở thời điểm hiện tại, dạng gauge được chọn có biểu tượng quả cầu chứa đựng lượng chất lỏng tương ứng với lượng nhiên liệu của động cơ giúp người dùng dễ hình dung về đại lượng vật lý nhiên liệu, giới hạn của gauge là từ 0-100 (%).

4. Dãy đèn báo gồm 3 đèn đỏ, xanh, vàng. Ba đèn này giúp người dùng có thể nhận biết được khi mức nhiên liệu của xe thấp quá ngưỡng cho phép. Ba đèn báo có các giá trị giới hạn theo thứ tự đỏ, vàng, xanh lần lượt là 0-20(%), 21-70 (%), 71-100 (%).
5. Chú thích thể hiện trạng thái nhiên liệu của động cơ gồm các mức độ cảnh báo mức nhiên liệu thấp, vừa, cao lần lượt tương ứng với các ngưỡng là 0-20(%), 21-70 (%), 71-100 (%).

4.4.2.4 Quãng đường đi được.



Hình 4.31: Thông số quãng đường đi được

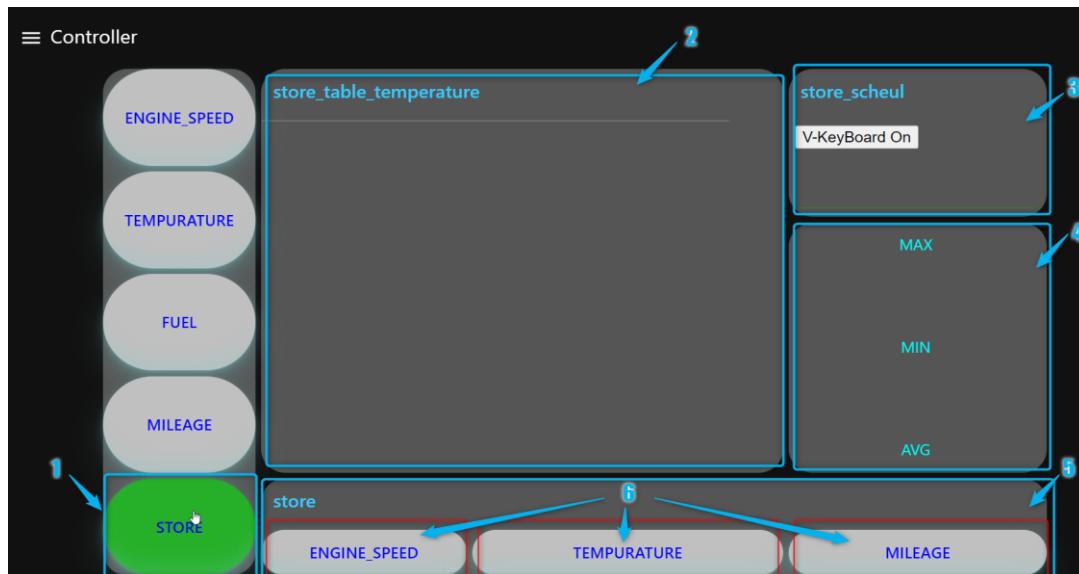
Giao diện chi tiết về quãng đường đi được hiện ra khi người dùng bấm chọn nút “Mileage”. Giao diện về quãng đường đi được gồm các thành phần chính.

1. Nút chọn theo dõi quãng đường đi được.
2. Biểu đồ thể hiện quãng đường phương tiện đã đi được, do quãng đường đi là một đại lượng chỉ có thể tăng vì thế ta chọn biểu đồ dạng cột đơn với trục tung tương ứng với các giá trị khoảng cách (Km) khi biểu đồ đạt đến các ngưỡng giới hạn sẽ có thông báo đến người dùng tùy mức độ.
3. Đồng hồ đo quãng đường đi được của phương tiện, hiển thị ở dạng số giúp người dùng dễ quan sát hơn.

4. Thang thê hiện quãng đường đi với các mức được chia tỷ lệ giúp người dùng có thể dễ hình dung được các mức độ cần bảo trì ở xe, tương ứng với 3 mức độ chính: Mức độ thường (0-100km), mức độ bảo dưỡng định kỳ (100-200km), mức độ bắt buộc bảo dưỡng (hơn 200km).
5. Dãy đèn báo gồm 3 đèn đỏ, xanh, vàng. Ba đèn này giúp người dùng có thể nhận biết trạng thái cần bảo trì của động cơ. Ba đèn báo có các giá trị giới hạn theo thứ tự xanh, vàng, đỏ lần lượt là mức độ thường 0-100 (Km), độ bảo dưỡng định kỳ 100-200(Km), hơn 200 km mà xe chưa được bảo dưỡng.
6. Chú thích giúp người dùng có thể nhận biết trạng thái cần bảo trì của động cơ: là mức độ thường 0-100 (Km), độ bảo dưỡng định kỳ 100-200(Km), hơn 200(Km) mà xe chưa được bảo dưỡng.

4.4.2.5 Lưu trữ các thông số của động cơ.

Ở giao diện này người dùng có thể truy xuất lại dữ liệu thu thập bằng cách chọn thời gian muốn theo dõi lại bằng bàn phím và dùng. Giao diện gồm các thông tin.

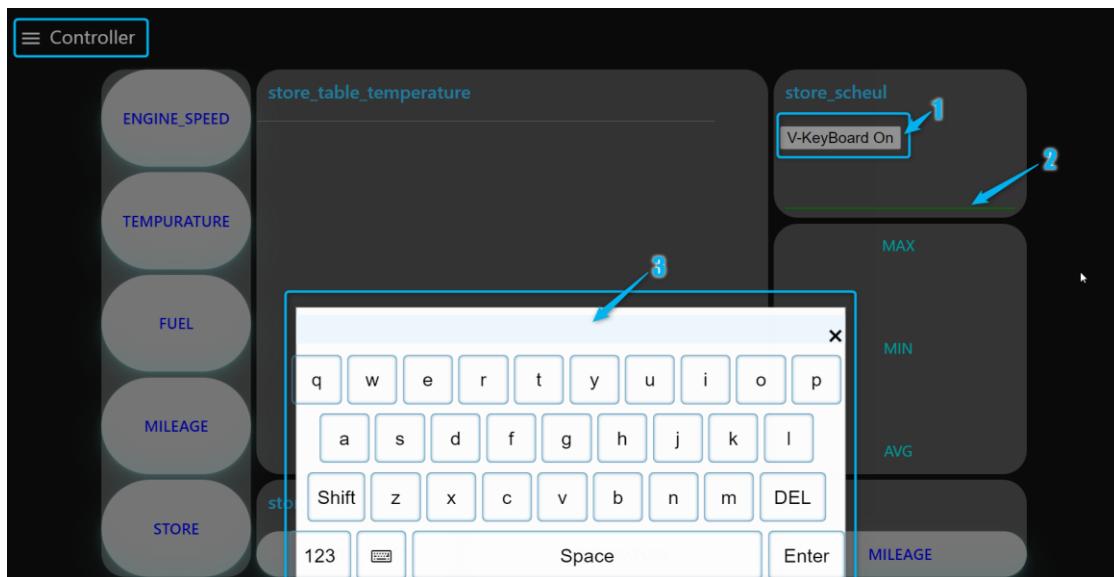


Hình 4.32: Chức năng lưu trữ các thông số

1. Nút nhấn chọn chế độ trích xuất dữ liệu lưu trữ. Bảng thể hiện dữ liệu lưu trữ gồm các trường thông tin: Dữ liệu muốn theo dõi (chọn thông qua các nút nhấn), ngày tháng thu thập dữ liệu, thời gian thu thập dữ liệu.

2. các nút nhấn chọn ngày tháng trích xuất dữ liệu, khi người dùng chọn nhập dữ liệu một bàn phím ảo sẽ hiện ra giúp người dùng có thể nhập ngày tháng mà mình muốn trích xuất dữ liệu.
3. Các thông số đặc trưng của toàn bộ dữ liệu được trích xuất, gồm có giá trị lớn nhất của đại lượng được trích xuất, giá trị nhỏ nhất của đại lượng trích xuất và giá trị trung bình của đại lượng cần trích xuất.
4. Khung chọn dữ liệu muốn theo dõi, người dùng có thể chọn các dữ liệu mà mình muốn theo dõi trên khung này thông qua các nút nhấn.
5. Các nút nhấn lựa chọn dữ liệu muốn trích xuất gồm ba nút nhấn tương ứng với ba đại lượng: tốc độ của xe, nhiệt độ động cơ, quãng đường đi được.

Khi người dùng nhập ngày tháng mong muốn trích xuất dữ liệu, GUI sẽ gửi yêu cầu truy xuất đến cơ sở dữ liệu (phpMyadmin) với ngày đúng với ngày mà người dùng nhập, từ đó sẽ trích xuất được bản dữ liệu theo đúng thông số ngày mà người dùng mong muốn, nếu ngày người dùng nhập không đúng hoặc không có dữ liệu của ngày đó thì sẽ không có dữ liệu được trả về.



Hình 4.33: Trích xuất dữ liệu lưu trữ

Khi người dùng muốn trích xuất dữ liệu chọn vào nút trích xuất dữ liệu (Đánh số 1 ở Hình 4.32), sau đó nhập ngày tháng muốn trích xuất, khi người dùng chọn vào ô nhập dữ

liệu một bàn phím ảo sẽ hiện lên (số 3 ở Hình 4.33), người dùng sẽ nhập ngày tháng thông qua bàn phím ảo này, cuối cùng chọn dữ liệu cần trích xuất.

ID	Engine_Speed	Date	Time
120	73	4/5/2023	5:53:40
121	78	4/5/2023	5:53:45
122	33	4/5/2023	5:53:50
123	91	4/5/2023	5:53:55
124	61	4/5/2023	5:54:0
125	25	4/5/2023	5:54:5
126	65	4/5/2023	5:54:10
127	30	4/5/2023	5:54:15
128	30	4/5/2023	5:54:20
129	5	4/5/2023	5:54:25
130	40	4/5/2023	5:54:30
131	49	4/5/2023	5:54:35
132	45	4/5/2023	5:54:40
133	42	4/5/2023	5:54:45
134	20	4/5/2023	5:54:50

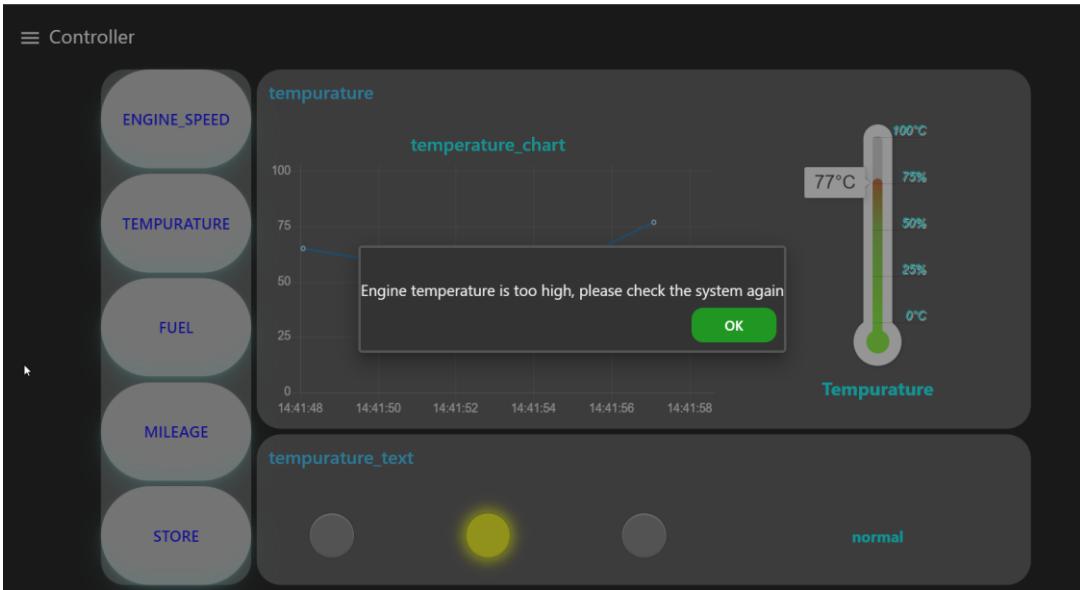
Hình 4.34: Hiển thị các giá trị lưu trữ theo ngày

Trên hình là bảng trích xuất dữ liệu tốc độ của xe ngày 4/5/2023, ta có thể thấy bảng hiển thị thông số ở phía tay trái (Hình 4.34), đồng thời ta có thể biết tốc độ của xe cao nhất trong ngày 4/5/2023 là 99 Km/h, tốc độ của xe thấp nhất trong ngày 4/5/2023 là 2 Km/h, tốc độ của xe trung bình trong ngày 4/5/2023 là 53 Km/h.

4.4.2.6 Các giao diện cảnh báo khi vượt quá ngưỡng

- Cảnh báo nhiệt độ động cơ vượt quá ngưỡng

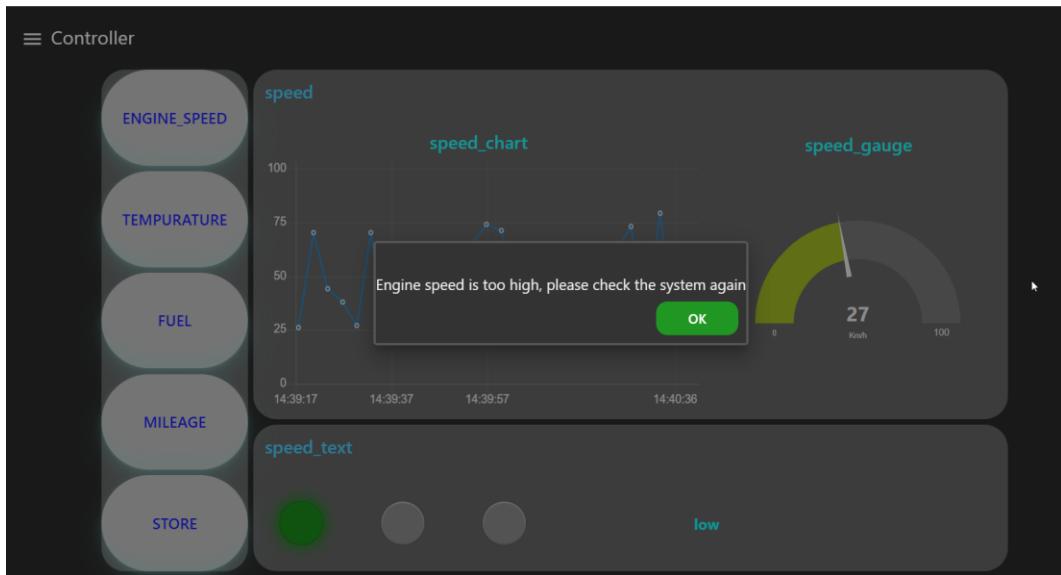
Khi nhiệt độ động cơ vượt quá ngưỡng cho phép một thông báo sẽ được hiện lên và nhắc nhở người dùng động cơ của phương tiện đang vượt quá ngưỡng giới hạn cho phép. Cảnh báo sẽ hiện trên màn hình chính cho đến khi người dùng tắt cảnh báo.



Hình 4.35: Cảnh báo nhiệt độ động cơ vượt quá ngưỡng cho phép

- Cảnh báo tốc độ của xe vượt quá ngưỡng

Khi tốc độ của xe vượt quá ngưỡng cho phép một thông báo sẽ được hiện lên và nhắc nhở người dùng tốc độ của xe của phương tiện đang vượt quá ngưỡng giới hạn cho phép. Cảnh báo sẽ hiện trên màn hình chính cho đến khi người dùng tắt cảnh báo.



Hình 4.36: Cảnh báo tốc độ của xe vượt quá ngưỡng cho phép

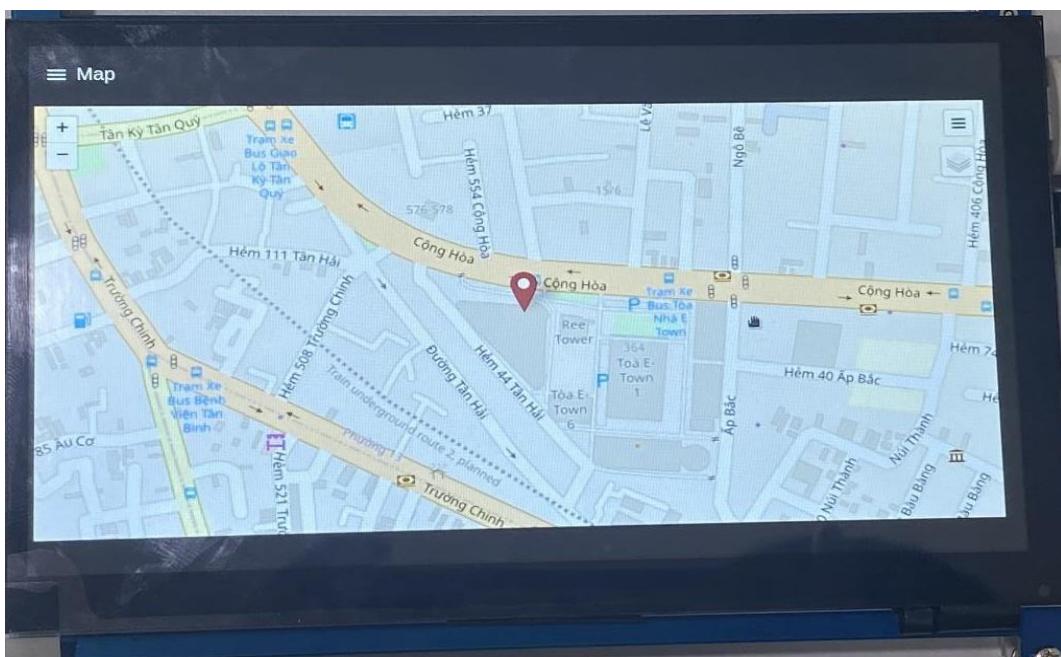
- Cảnh báo nhiên liệu động cơ thấp quá ngưỡng cho phép

Trong Hình 4.36 thể hiện khi nhiên liệu động cơ thấp quá ngưỡng cho phép một thông báo sẽ được hiện lên và nhắc nhở người dùng nhiên liệu động cơ thấp quá ngưỡng cho phép, cần được bổ sung thêm nhiên liệu. Cảnh báo sẽ hiện trên màn hình chính cho đến khi người dùng tắt cảnh báo.



Hình 4.37: Cảnh báo nhiên liệu thấp quá ngưỡng cho phép.

4.4.3 Map



Hình 4.38: Giao diện bản đồ

Trong giao diện trang “Map”, người dùng có thể thấy được vị trí của phương tiện, xem được bản đồ với nhiều chế độ khác nhau.

GUI sẽ lấy giữ liệu kinh độ và vĩ độ của xe được gửi về từ module GPS qua đó xác định được vị trí của xe và hiển thị lên bản đồ. Khi xe di chuyển thì chấm đỏ sẽ di chuyển theo vị trí hiện tại của xe, vị trí hiện tại của xe sẽ được đánh dấu đỏ như Hình 4.38 đã thể hiện.

4.5 ĐÁNH GIÁ HỆ THỐNG

Các chức năng hoạt động của toàn bộ hệ thống đã thực thi một cách ổn định như việc truyền, gửi và chuyển đổi các giá trị từ CAN frame được gửi từ ECU qua central gateway (Raspberry Pi) các thông số như nhiệt độ động cơ, tốc độ xe, quãng đường đi được và mức nhiên liệu của xe. Ngoài ra, do việc sử dụng module đọc CAN trên Raspberry có giá thành không cao nên việc đòi hỏi các linh kiện có thể đáp ứng được việc đọc đúng chính xác về yêu cầu thời gian đọc frame được gửi ra của ECU không thể đáp ứng được.

Trong các frame thì đối với frame truyền của giá trị quãng đường đi được thì không thể thay đổi hay can thiệp vào. Vì giá trị đọc được này được sử dụng mặc định lưu trong bộ nhớ của ECU và cần phải dùng dụng cụ thực tế hay một máy mô phỏng thực tế để khởi động động cơ của công ty Bosch mới có thể tác động đến biến quãng đường đi được này từ đó mới tăng được giá trị, nên các giá trị để có thể đưa ra thông báo, cảnh báo sẽ là hoàn toàn dựa trên mô phỏng truyền giả một giá trị để kiểm tra hoạt động.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 KẾT LUẬN

Thông qua đề tài “Thiết kế và thi công hệ thống giám sát và chuẩn đoán cho xe cơ giới” với mô hình có thể hoạt động bình thường và thực hiện được đầy đủ các chức năng được đề ra thông qua các yêu cầu hệ thống ban đầu từ đó nhóm đã nhận thấy rằng đối với đề tài này sẽ có các ưu điểm và nhược điểm như sau:

+ Ưu điểm:

- Hệ thống, mô hình hoạt động, thực hiện được các chức năng theo yêu cầu người dùng.

- Hệ thống hoạt động với các tính năng ổn định, đáp ứng được các tiêu chí yêu cầu về thời gian dưới 5 giây đối với một mô hình truyền gửi thực tế trên xe thật.

- Các chức năng cơ bản mà mô hình cũng như hệ thống có thể thực hiện đều có thể ứng dụng vào trong thực tế, hiện nay về ý tưởng mô hình đã được tích hợp vào một số hệ thống của các hãng xe hiện nay. Về cơ bản việc sử dụng ECU từ công ty Bosch đã thể hiện được phần nào về công dụng hoạt động trong thực tế của đề tài.

+ Nhược điểm:

- Đọc CAN từ Raspberry Pi vẫn còn trễ với một thời gian ngắn khi gửi liên tục từ ECU.

- Cơ sở dữ liệu còn bị hạn chế dung lượng lưu trữ.

- Chưa có thực sự đảm bảo về bảo mật, an toàn khi truyền gửi các dữ liệu.

- Hiển thị phần nội dung truy suất dữ liệu trên GUI còn hạn chế, chưa có khả năng thể hiện dữ liệu cũ dưới dạng đồ thị.

- Chưa tối ưu hóa được các hiển thị giao diện bản đồ.

5.2 HƯỚNG PHÁT TRIỂN

Mô hình chỉ là một phần nhỏ đối với các chức năng cũng như hoạt động của một hệ thống thực trên xe, nên về hướng phát triển sẽ có rất nhiều để nhằm nâng cao và phát triển trong tương lai.

- Cải thiện về cách sử dụng các chuẩn truyền thông để truyền gửi dữ liệu tốt hơn trong môi trường mạng wifi yếu hoặc không có đối với web và app.
- Đảm bảo cải thiện về độ bảo mật dữ liệu khi truyền gửi sử dụng các giao diện người dùng.
- Phát triển thêm phân quyền từ đó có thể xem xét mở rộng thêm với việc quản lý mỗi tài khoản đối với mỗi xe khác nhau.
- Kết hợp dữ liệu thu thập với trí tuệ nhân tạo để hệ thống có thể tự tính toán và đoán trước được .
- Dưa tín hiệu điều khiển ngược về ECU, từ đó có thể điều khiển được động cơ phương tiện.

TÀI LIỆU THAM KHẢO

- Das, A. (2019 , July 10). *Use Neo 6M GPS Module with Raspberry Pi and Python.* Retrieved from <https://sparklers-the-makers.github.io/blog/robotics/use-neo-6m-module-with-raspberry-pi/>
- Falch, M. (2022, March). *J1939 Explained - A Simple Intro.* (CSS Electronics) Retrieved from <https://www.csselectronics.com/pages/j1939-explained-simple-intro-tutorial>
- Hertz, D. (2020, February 03). *How to Use a GPS Receiver With Raspberry Pi 4.* (EETech Media, LLC) Retrieved from <https://maker.pro/raspberry-pi/tutorial/how-to-use-a-gps-receiver-with-raspberry-pi-4>
- Linh, L. (2022, September 22). *Phát triển nông nghiệp ứng dụng công nghệ cao.* Retrieved from <https://dangcongsan.vn/khoa-hoc-va-cong-nghe-voi-su-nghiep-cong-nghiep-hoa-hien-dai-hoa-dat-nuoc/diem-nhan-khoa-hoc-va-cong-nghe/phat-trien-nong-nghiep-ung-dung-cong-nghe-cao-563993.html>
- Python Contributor. (2022, November 24). *Python-can 4.1.0 documentation.* Retrieved from <https://python-can.readthedocs.io/en/v4.1.0/api.html>
- Random Nerd Tutorials. (n.d.). *Getting Started with Node-RED on Raspberry Pi.* Retrieved from <https://randomnerdtutorials.com/getting-started-node-red-raspberry-pi>
- Vector Informatik GmbH. (2020). *Solutions for CAN .* (Vector) Retrieved from <https://www.vector.com/int/en/products/solutions/networks/can/#c224051>
- W3Schools. (2023). *JavaScript Tutorial.* (JAVASCRIPT) Retrieved from <https://www.w3schools.com/js/default.asp>
- W3Schools. (2023). *React Tutorial.* (REACT) Retrieved from <https://www.w3schools.com/react/default.asp>
- w3schools.io. (2023). *Dart Language.* (Dart) Retrieved from <https://www.w3schools.io/languages/dart-tutorials/>
- WaveShare. (n.d.). *Module RS485 CAN HAT for Raspberry Pi.* Retrieved from https://www.waveshare.com/wiki/RS485_CAN_HAT