

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN 1

-----□□□□-----



BÀI TẬP LỚN

Môn học: IoT và ứng dụng

Tên lớp: D22CNPM02

Nhóm bài tập lớn : 8

Tên đề tài: Bài đồ xe thông minh

Danh sách thành viên nhóm:

1. Nguyễn Ngọc Thị Hạnh MSV: B22DCCN832
2. Nguyễn Viết Quyên MSV: B22DCCN676
3. Nguyễn Quyết Tiến MSV: B22DCCN724
4. Nguyễn Tân Thành MSV: B22DCCN796

MỤC LỤC

I. Tài liệu SRS.....	1
1. Giới thiệu.....	1
a) Mục tiêu hệ thống.....	1
b) Phạm vi triển khai.....	1
c) Tiêu chí thành công (KPIs).....	1
d) Kết quả mong đợi.....	2
2. Mô tả tổng quan.....	2
a) Môi trường hoạt động.....	2
b) Môi trường triển khai.....	3
3. Yêu cầu chức năng.....	3
4. Yêu cầu phi chức năng.....	4
5. Ràng buộc kỹ thuật và môi trường.....	5
6. Mô hình hóa yêu cầu.....	7
II. Lý thuyết và công nghệ liên quan.....	9
1. Internet vạn vật (IoT) là gì?.....	9
2. Mô hình 4 lớp.....	10
3. Tổng quan về phần cứng.....	11
a) ESP32:.....	11
b) Cảm biến siêu âm (SRF05):.....	12
c) Servo motor (SG90):.....	13
4. Blynk Cloud.....	13
5. Cấu trúc của hệ thống bãi đỗ xe thông minh.....	14

I. Tài liệu SRS

1. Giới thiệu

a) Mục tiêu hệ thống

- Vấn đề thực tế: Ở các đô thị lớn, nhu cầu đỗ xe, đặc biệt là cho ô tô đang còn thiếu sót rất nhiều về mặt quản lý, người lái xe phải mất nhiều thời gian để đi tìm chỗ trống và việc phát hiện chỗ xe sai cách gây tốn diện tích cùng với số lượng xe nhiều khiến người quản lý gặp nhiều trở ngại trong việc kiểm soát bãi đỗ xe.
- Mục tiêu:
 - Giám sát tự động: ghi nhớ biển số xe và cấp phép các xe đến và đi mỗi khi ra vào bãi đỗ
 - Tự động hóa: hệ thống đóng mở thanh chắn tự động sử dụng servo với cảm biến siêu âm
 - Tối ưu hóa: phát hiện vị trí chỗ trống hoặc đã có xe bằng cảm biến siêu âm đặt tại từng vị trí đỗ.
 - Hiển thị thông tin chỗ trống cho tài xế
 - Quản lý, thống kê số lượng xe, thời gian đỗ, tình trạng hoạt động của cảm biến.
- Kì vọng: tiết kiệm 30% chi phí thuê nhân viên, nâng cao hiệu suất và quản lý hiệu quả cao.

b) Phạm vi triển khai

- Số lượng thiết bị:
 - 5 cảm biến siêu âm HC SR04
 - 1 servo đóng mở
 - Bộ kít arduino (board test, dây cắm)
 - Vị mạch ESP32
 - 1 thiết bị hiển thị thông tin
- Môi trường hoạt động:
 - Ngoài trời → chịu mưa, nắng điều kiện thiếu sáng.
 - Trong nhà: yêu cầu độ chính xác cao và chống nhiễu tốt (vì tín hiệu siêu âm dễ bị phản xạ).

c) Tiêu chí thành công (KPIs)

Tiêu chí	Mục tiêu mong muốn
Độ chính xác	Tỷ lệ phát hiện trạng thái có xe/ trống $\geq 95\%$

Độ trễ	<= 0,2 giây - Thời gian hệ thống hiển thị từ khi phát hiện <=5 giây - Đóng mở thanh chắn
Độ tin cậy	Uptime >98% hệ thống hoạt động ổn định, không mất kết nối
Chi phí triển khai	Tổng chi phí dự kiến <= 500.000 đ cho toàn bộ hệ thống
Khả năng mở rộng	Dễ dàng thêm cảm biến mới, không ảnh hưởng hệ thống hiện tại

d) Kết quả mong đợi

- Tự động hóa quy trình quản lý bãi đỗ xe → tiết kiệm thời gian, tối ưu hiệu suất
- Người quản lý giám sát qua màn hình
- Dữ liệu thu thập từ biển số phục vụ cho việc theo dõi quản lý xe (ví dụ phòng ngừa trộm cắp, truy xuất thông tin).

2. Mô tả tổng quan

a) Môi trường hoạt động

- Điều kiện ngoài trời / trong nhà:
 - + Nhiệt độ & ẩm: chọn thiết bị -20°C...70°C, vỏ IP65+ (mưa, bụi), gioăng cao su + cổng cáp có gland chống nước.
 - + Bức xạ mặt trời/UV: vỏ nhựa cần UV-resistant; tránh lóa nắng nếu dùng IR/camera; che mưa cho cảm biến siêu âm.
 - + Ăn mòn/khói bụi: bãi gần đường lớn/biển → vít, bracket inox 304/316; lọc bụi cho hộp điện.
 - + Rung/va đập: vị trí lắp thấp dễ bị cán/vá chạm → chọn cảm biến chống va đập (IK08+) hoặc đặt cao trên trụ.
- Nhiều điện tử & phản xạ tín hiệu:
 - + Nhiều RF: khu thương mại/ tầng hầm Wi-Fi dày đặc → khảo sát site, chọn kênh ít nhiễu; cân nhắc LoRa/NB-IoT cho các điểm xa/góc khuất.
 - + Siêu âm: mặt đường ướt, vũng nước, pô xe nóng gây phản xạ/sai số → đặt cảm biến góc 10–15°, khoảng cách an toàn; lọc trung bình trượt + ngưỡng hysteresis.
 - + Từ trường/kim loại (nếu dùng cảm biến từ): thay đổi do cốt thép sàn → cần hiệu chuẩn tại chỗ từng ô.
- Cấp nguồn & chống sét:
 - + Nguồn AC chập chờn: gateway/LED nên có UPS 15–30 phút; ô rải rác

- + dùng pin + năng lượng mặt trời (duty cycle thấp).
- + Tiết kiệm năng lượng: node slot ngủ deep-sleep, đánh thức theo chu kỳ 5–10s; truyền gói ngắn (MQTT QoS 0/1)
- + Sét lan truyền: đường cáp và tín hiệu dài → SPD/TVS, nối đất đúng chuẩn; cáp ngoài trời đi ống.

Nếu bỏ qua các ràng buộc này: cảm biến nhanh hỏng, dữ liệu nhiều, mất kết nối hoặc... hết pin sau vài tuần

b) Môi trường triển khai

- Không gian vật lý:
- + Kịch bản trong nhà (tầng hầm, TTTM): âm cao, nhiều bê-tông/kim loại, Wi-Fi dày đặc → chú ý phản xạ siêu âm, chọn kênh RF ít nhiễu, bố trí gateway theo “ô lưới”.
- + Kịch bản ngoài trời (sân cơ quan, khu dân cư): nắng mưa, bụi, tia UV → vỏ IP65+, che mưa cho sensor, cáp đi ống; tính đến chớp sét lan truyền.
- + Vị trí lắp thiết bị:
 - Cảm biến chỗ đỗ: cao 1.2–1.8 m, nghiêng 10–15°, tránh ngay trước ống xả.
 - Gateway: gần nguồn AC ổn định, thông thoáng, dễ bảo trì.
 - Barie: có vùng an toàn người đi bộ + cảm biến chống kẹt.
- Hạ tầng điện – nguồn:
 - + Nguồn AC cho gateway/barie, kèm UPS 15–30 phút.
- Hạ tầng mạng:
 - + Wi-Fi nội bộ cho gateway; ưu tiên Ethernet nếu được.
 - + Vùng phủ: khảo sát RSSI, lập heatmap tối thiểu trước khi lắp đặt tràn.
- Nền tảng phần mềm/Cloud
 - + Cloud (Blynk): dùng MQTT/HTTPS; bật TLS, quản lý token.
 - + CSDL: time-series cho trạng thái chỗ trống + bảng giao dịch/nhật ký sự kiện.
 - + Triển khai theo môi trường: DEV (mock), UAT (site thử 4 ô), PROD (toàn bãi).

3. Yêu cầu chức năng

- Thu thập dữ liệu cảm biến:
 - + Mỗi chỗ đỗ xe được gắn cảm biến siêu âm để đo khoảng cách, từ đó xác định chỗ trống hay có xe
 - + Cảm biến được cấu hình đo định kỳ (ví dụ 5s) và gửi dữ liệu kèm theo timestamp + ID vị trí
 - + Camera nhận dạng biển số(LPR) được đặt tại cổng ra/vào để ghi lại biển số xe và thời gian ra vào

- Gửi dữ liệu về Blynk:
 - + Dữ liệu cảm biến được gửi về từ ESP32 qua giao thức MQTT
 - + ESP32 có thể xử lý sơ bộ(lọc nhiễu, xác nhận trạng thái hợp lệ) rồi đồng bộ lên cloud
 - + Hệ thống có cơ chế retry hoặc lưu tạm(buffer) khi mất mạng để dữ liệu không bị mất
- Lưu trữ và phân tích dữ liệu:
 - + Lưu lại lịch sử input trên cloud Blynk
 - + Lưu lại thông tin về lịch sử ra vào lên database (sqlite), hệ thống sử dụng nó để có thể phân tích: số lượng chỗ trống hiện tại, thời gian trung bình xe đỗ, biểu đồ lưu lượng xe thời giờ/ngày
- Hiển thị dữ liệu qua ứng dụng website:
 - + Hệ thống cung cấp website giám sát trực tuyến cho người dùng
 - + Các chức năng hiển thị:
 - Bản đồ bãi xe hiển thị trạng thái từng ô(xanh = trống, đỏ = có xe)
 - Bảng thống kê theo thời gian thực số chỗ trống còn lại
 - Thông báo hoặc cảnh báo khi bãi xe đầy
 - Lịch sử ra/vào cùng biển số xe
- Điều khiển và ra lệnh ngược lại thiết bị:
 - + Hệ thống có thể gửi lệnh điều khiển ngược lại các thiết bị:
 - Mở đóng barie tự động khi xe ra/vào
 - Điều chỉnh trạng thái chỗ trống
 - + Việc điều khiển này yêu cầu xác thực người và phân quyền(ví dụ: chỉ nhân viên mới mở được barie thủ công)
- Các chức năng quản lý và bảo mật khác(mở rộng)
 - + Quản lý người dùng: tạo tài khoản, phân quyền(người dùng thường, nhân viên, quản trị)
 - + Bảo mật dữ liệu: mã hóa đường truyền(HTTPS, SSL/TLS), kiểm tra xác thực API (dùng cho việc giao tiếp giữa Blynk và web application)
 - + Cập nhật phần mềm từ xa(OTA) cho các thiết bị IoT để bảo trì hệ thống

4. Yêu cầu phi chức năng

- Hiệu năng
 - + Độ trễ cập nhật trạng thái: <3s kể từ khi xe vào/ra khỏi chỗ đỗ cho đến khi hiển thị trên bản đồ
 - + Thời gian nhận diện biển số: <5s để camera ghi nhận hình ảnh và nhận diện được biển số, cuối cùng là xử lý ở hệ thống và lưu vào DB
 - + Tần suất lấy mẫu cảm biến: 5s/lần

- + Khả năng chịu tải: Hệ thống xử lý được đồng thời yêu cầu từ 100% số lượng cảm biến và camera trong giờ cao điểm mà không bị treo
- Bảo mật
 - + Mã hóa: sử dụng HTTPS/TLS cho mọi kết nối giữa Blynk Cloud và Web Application
 - + Xác thực: mọi truy cập API phải được xác thực (dùng JWT)
 - + Phân quyền: hệ thống được chia thành các cấp độ phân quyền như: quản lý, nhân viên, khách hàng
- Độ tin cậy
 - + Tỉ lệ truyền dữ liệu thành công: >99% từ cảm biến về Esp32
 - + Thời gian hoạt động (Uptime): >99% mỗi tháng
 - + Khả năng lưu trữ đệm (Buffer): Esp32 phải lưu trữ được dữ liệu của ít nhất 1 giờ khi mất kết nối internet
- Khả năng mở rộng
 - + Hỗ trợ tăng thêm 50% số lượng cảm biến mà không thay đổi kiến trúc, gateway
 - + Kiến trúc: sử dụng kiến trúc microservice để các thành phần như quản lý xe, thanh toán, hiển thị có thể được nâng cấp độc lập
- Chi phí và năng lượng
 - + Băng thông mạng
 - Lưu lượng cảm biến: dữ liệu cực nhỏ, chỉ gửi khi có sự kiện thay đổi trạng thái
 - Lưu lượng camera LPR: <100Kb cho mỗi lượt xe ra vào → Xử lý nhận dạng biển số ngay tại gateway, chỉ gửi dữ liệu văn bản(biển số, thời gian) và 1 ảnh về cloud, không truyền video thô
 - + Chi phí hạ tầng:
 - Chi phí vận hành cloud: <=1USD/chỗ đỗ/tháng ở quy mô 200 chỗ đỗ → Sử dụng các dịch vụ đám mây có quản lý như Managed MQTT broker và các hàm serverless để xử lý dữ liệu phù hợp với dữ liệu biến động của bãi xe (cao điểm vào ban ngày và thấp điểm vào ban đêm)

5. Ràng buộc kỹ thuật và môi trường

STT	Nhóm ràng buộc	Mô tả chi tiết	Ảnh hưởng/ Giải pháp
1	Môi trường hoạt	- Hệ thống hoạt động ngoài	- Dùng cảm biến có vỏ

	động	<p>trời, chịu ảnh hưởng của nắng, mưa, bụi.</p> <ul style="list-style-type: none"> - Nhiệt độ dao động: -20°C → 70°C. - Độ ẩm cao, có thể ngưng tụ nước. - Có khả năng nhiều sóng Wi-Fi tại khu vực đông thiết bị. 	<p>IP65 trở lên (chống nước/bụi).</p> <ul style="list-style-type: none"> - Chọn ESP32 có khả năng hoạt động ổn định trong điều kiện ngoài trời. - Che chắn, lắp hộp bảo vệ và thử nghiệm thực tế để xác định vị trí lắp tối ưu. - Nếu nhiều nhiều, xem xét chuyển sang LoRa hoặc NB-IoT.
2	Nguồn cấp và năng lượng	<ul style="list-style-type: none"> - Một số khu vực xa nguồn điện, cần giải pháp cấp nguồn linh hoạt. - Gateway và barie cần nguồn ổn định, cảm biến có thể dùng pin. 	<ul style="list-style-type: none"> - Dùng nguồn DC ổn định hoặc UPS mini cho gateway/barie. - Cảm biến dùng pin + chế độ tiết kiệm năng lượng (sleep mode). - Với mô hình ngoài trời, có thể bổ sung pin năng lượng mặt trời.
3	Ràng buộc pháp lý	<ul style="list-style-type: none"> - Phải tuân thủ quy định tần số vô tuyến hợp pháp ở Việt Nam (~920 MHz cho LoRa). - Dữ liệu thu thập (biển số, hình ảnh) cần tuân thủ chính sách bảo mật. 	<ul style="list-style-type: none"> - Sử dụng tần số được phép hoạt động trong băng ISM. - Mã hóa dữ liệu khi truyền qua mạng (MQTT/HTTPS). - Thêm thông báo “Khu vực có camera giám sát”, chỉ dùng dữ liệu cho mục đích quản lý.
4	Bộ nhớ và xử lý	<ul style="list-style-type: none"> - ESP32/STM32 có RAM và Flash hạn chế. - Không phù hợp cho thuật toán phức tạp hoặc AI nhận dạng tại chỗ. 	<ul style="list-style-type: none"> - Chỉ xử lý những phép tính đơn giản (đọc cảm biến, gửi dữ liệu). - Các tác vụ nặng (phân tích dữ liệu, thống kê) đưa

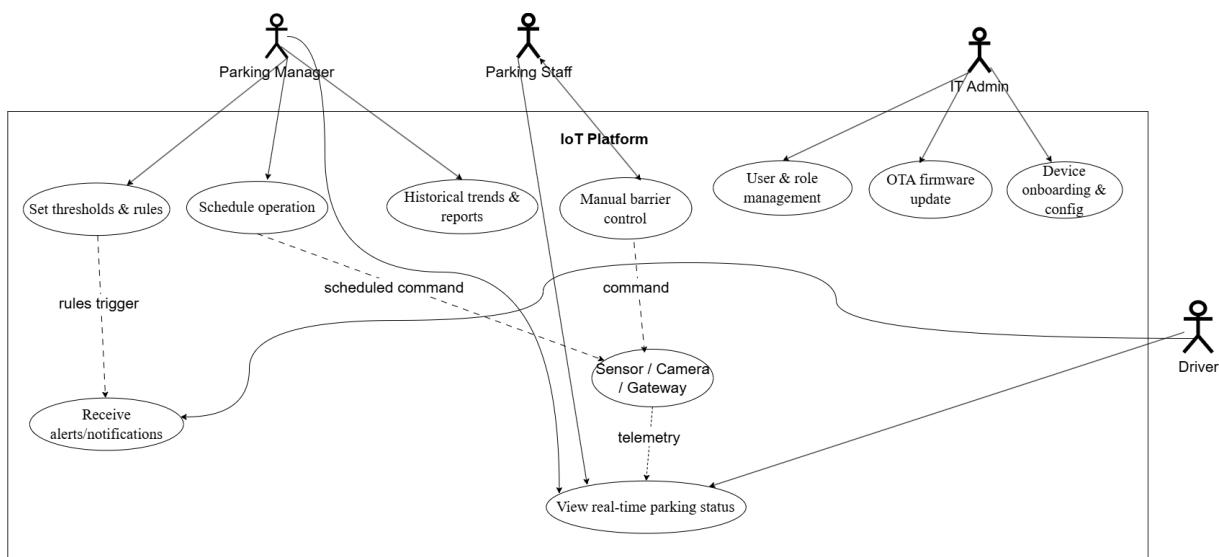
			lên Cloud hoặc Server để xử lý.
5	Dung lượng pin và tuổi thọ thiết bị	<ul style="list-style-type: none"> - Cảm biến dùng pin phải hoạt động nhiều tháng liên tục. - Nếu cấu hình sai, pin sẽ nhanh hết hoặc treo thiết bị. 	<ul style="list-style-type: none"> - Giới hạn tần suất gửi dữ liệu (5–10s/lần). - Dùng chế độ deep-sleep. - Theo dõi mức pin qua dashboard để thay định kỳ.
6	An toàn cơ khí và điện	<ul style="list-style-type: none"> - Cổng barie có thể gây nguy hiểm nếu đóng khi xe chưa qua. - Mạch điều khiển barie và cảm biến cần cách ly điện tốt. 	<ul style="list-style-type: none"> - Trang bị cảm biến chống kẹt / cảm biến quang ở barie. - Dùng relay có optocoupler để cách ly mạch điều khiển. - Có nút dừng khẩn cấp (Emergency Stop) cho người vận hành.
7	Bảo trì và mở rộng	<ul style="list-style-type: none"> - Thiết bị ngoài trời dễ bám bụi, hỏng hóc do thời tiết. - Khi mở rộng số lượng ô đỗ, hệ thống phải dễ tích hợp thêm cảm biến mới. 	<ul style="list-style-type: none"> - Thiết kế mô-đun, dễ thay thế từng cảm biến/gateway. - Giao thức truyền dữ liệu MQTT giúp thêm node mới nhanh chóng. - Lập kế hoạch bảo trì định kỳ (1–2 tháng/lần).

6. Mô hình hóa yêu cầu

- Actors:
 - Parking Manager: Theo dõi báo cáo, đặt quy tắc cảnh báo, xem lịch sử ra vào
 - Parking Staff: Mở barie thủ công, xử lý lỗi, giám sát trực tiếp
 - Driver: Xem chỗ trống, nhận thông báo, ra/vào bãi xe
 - IT Admin: Cấu hình thiết bị, cập nhật firmware, phân quyền người dùng
- Các use case chính:
 - U1: View real-time parking status: Hiển thị trạng thái từng chỗ đỗ

xe(trống/có xe)

- U2:Receive alerts/notifications: Nhận cảnh báo bãi đầy hoặc lỗi cảm biến
- U3: Manual barrier control: Nhân viên điều khiển barie thủ công khi cần
- U4: Set thresholds & rules: Đặt ngưỡng cảnh báo
- U5: Schedule operation: Thiết lập lịch đóng/mở hệ thống hoặc đèn báo
- U6: Historical trends & reports: Xem báo cáo lượng xe, thời gian đỗ trung bình
- U7: Device onboarding & config: Thêm mới, cấu hình cảm biến hoặc camera
- U8: OTA firmware update: Cập nhật phần mềm thiết bị từ xa
- U9: User & role management: Tạo tài khoản, phân quyền người dùng
- U1-U2-U3-U5 bao phủ monitoring & control
- U4 mô hình hóa rules/thresholds kích hoạt cảnh báo(mỗi quan hệ <include/trigger> tới U2)
- U6 cho analytics
- U7-U8-U9 cho vận hành & bảo trì hệ thống
- Quan hệ:
 - Hardware → U1 thể hiện thiết bị gửi telemetry để cập nhật trạng thái
 - U3/U5 → Hardware thể hiện điều khiển thời gian thực hoặc theo lịch
 - U4 → U2 nhấn mạnh cảnh báo phải phát sinh từ ngưỡng



II. Lý thuyết và công nghệ liên quan

1. Internet vạn vật (IoT) là gì?

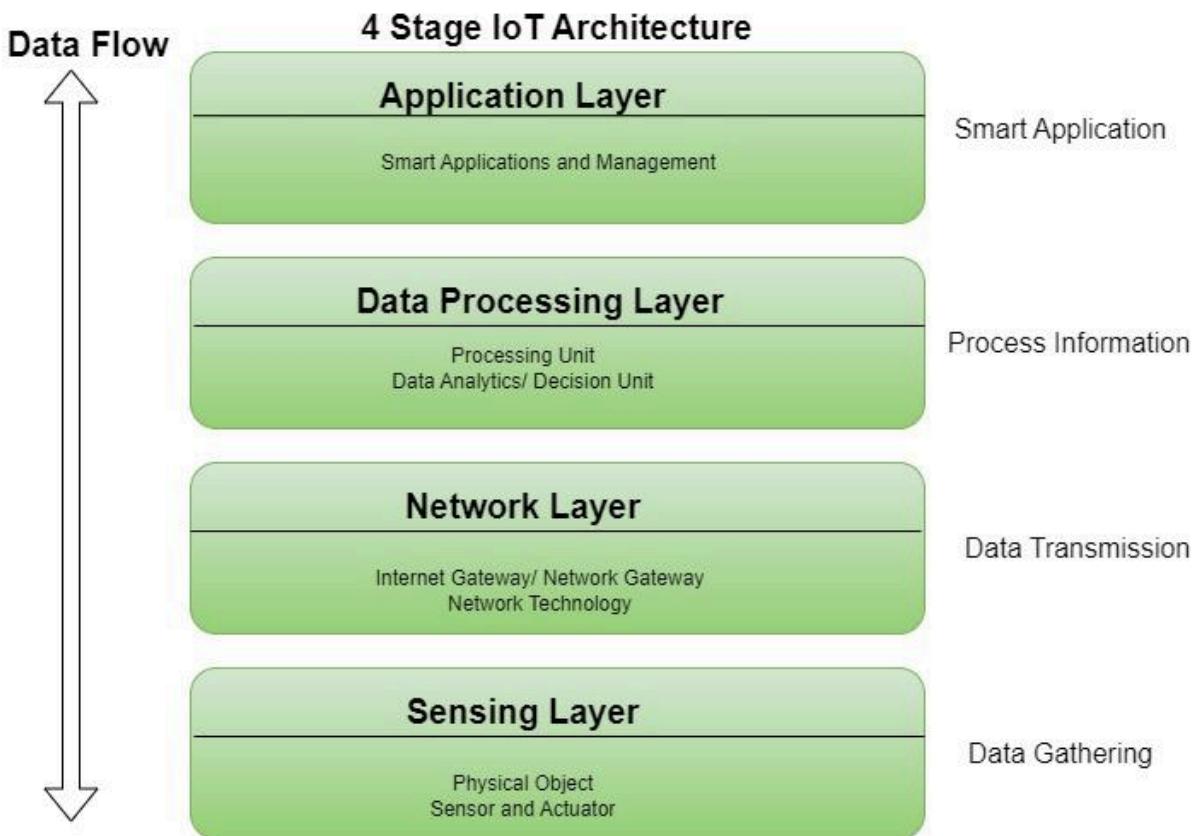
Thuật ngữ IoT hay Internet vạn vật đề cập đến mạng lưới tập hợp các thiết bị thông minh và công nghệ tạo điều kiện thuận lợi cho hoạt động giao tiếp giữa thiết bị và đám mây cũng như giữa các thiết bị với nhau. Nhờ sự ra đời của chip máy tính giá rẻ và công nghệ viễn thông băng thông cao, ngày nay, chúng ta có hàng tỷ thiết bị được kết nối với internet. Điều này nghĩa là các thiết bị hàng ngày như bàn chải đánh răng, máy hút bụi, ô tô và máy móc có thể sử dụng cảm biến để thu thập dữ liệu và phản hồi lại người dùng một cách thông minh.

Internet vạn vật tích hợp “vạn vật” với Internet mỗi ngày. Các kỹ sư máy tính đã và đang thêm các cảm biến và bộ xử lý vào các vật dụng hàng ngày kể từ những năm 90. Tuy nhiên, tiến độ ban đầu rất chậm vì các con chip còn to và cồng kềnh. Loại chip máy tính công suất thấp gọi là thẻ tag RFID, lần đầu tiên được sử dụng để theo dõi các thiết bị đắt đỏ. Khi kích cỡ của các thiết bị điện toán dần nhỏ lại, những con chip này cũng trở nên nhỏ hơn, nhanh hơn và thông minh hơn theo thời gian.

Chi phí tích hợp công suất điện toán vào trong các vật dụng nhỏ bé hiện nay đã giảm đáng kể. Ví dụ: bạn có thể thêm khả năng kết nối với các tính năng của dịch vụ giọng thoại Alexa vào các MCU tích hợp sẵn RAM chưa đến 1 MB, chẳng hạn như cho công tắc đèn. Nguyên cả một ngành công nghiệp đã bất ngờ xuất hiện với trọng tâm xoay quanh việc trang bị các thiết bị IoT khắp mọi ngóc ngách căn nhà, doanh nghiệp và văn phòng của chúng ta. Những vật dụng thông minh này có thể tự động truyền và nhận dữ liệu qua Internet. Tất cả những “thiết bị điện toán vô hình” này và công nghệ liên quan được gọi chung là Internet vạn vật.

2. Mô hình 4 lớp

Với kiến trúc này sẽ được chia thành 4 lớp khác nhau:



- Lớp Cảm biến (Sensing Layer): Lớp cảm biến là lớp đầu tiên của kiến trúc Internet of Things và chịu trách nhiệm thu thập dữ liệu từ nhiều nguồn khác nhau. Lớp này bao gồm các cảm biến và cơ cấu chấp hành được đặt trong môi trường để thu thập thông tin về nhiệt độ, độ ẩm, ánh sáng, âm thanh và các thông số vật lý khác. Các giao thức truyền thông có dây hoặc không dây kết nối các thiết bị này với lớp mạng.
- Lớp Truyền thông (Network Layer): Lớp mạng của kiến trúc IoT chịu trách nhiệm cung cấp khả năng giao tiếp và kết nối giữa các thiết bị trong hệ thống. Nó bao gồm các giao thức và công nghệ cho phép các thiết bị kết nối, giao tiếp với nhau và với mạng internet rộng hơn.
- Lớp Xử lý dữ liệu (Data processing Layer): Lớp xử lý dữ liệu của kiến trúc IoT để cập đến các thành phần phần cứng và phần mềm chịu trách nhiệm thu thập, phân tích và diễn giải dữ liệu từ các thiết bị IoT. Lớp này có nhiệm vụ nhận dữ liệu thô từ các thiết bị, xử lý và cung cấp dữ liệu đó cho việc phân tích hoặc hành động tiếp theo. Lớp xử lý dữ liệu bao gồm nhiều công nghệ và công cụ khác nhau, chẳng hạn như hệ thống quản lý dữ liệu, nền tảng phân tích và thuật toán học máy. Những công cụ này được sử dụng để trích xuất thông tin chi tiết có ý nghĩa từ dữ liệu và đưa ra quyết định dựa trên dữ liệu đó.

- Lớp Ứng dụng (Application Layer): Lớp ứng dụng của kiến trúc IoT là lớp trên cùng, tương tác trực tiếp với người dùng cuối. Nó chịu trách nhiệm cung cấp các giao diện và chức năng thân thiện với người dùng, cho phép họ truy cập và điều khiển các thiết bị IoT. Lớp này bao gồm nhiều phần mềm và ứng dụng khác nhau như ứng dụng di động, cổng thông tin web và các giao diện người dùng khác được thiết kế để tương tác với cơ sở hạ tầng IoT bên dưới. Nó cũng bao gồm các dịch vụ phần mềm trung gian (middleware) cho phép các thiết-bị-và-hệ-thông-IoT khác nhau giao tiếp và chia sẻ dữ liệu một cách liền mạch. Lớp ứng dụng cũng bao gồm các khả năng phân tích và xử lý cho phép dữ liệu được phân tích và chuyển đổi thành những hiểu biết có ý nghĩa. Điều này có thể bao gồm các thuật toán học máy, công cụ trực quan hóa dữ liệu và các khả năng phân tích nâng cao khác.

3. Tổng quan về phần cứng

a) ESP32



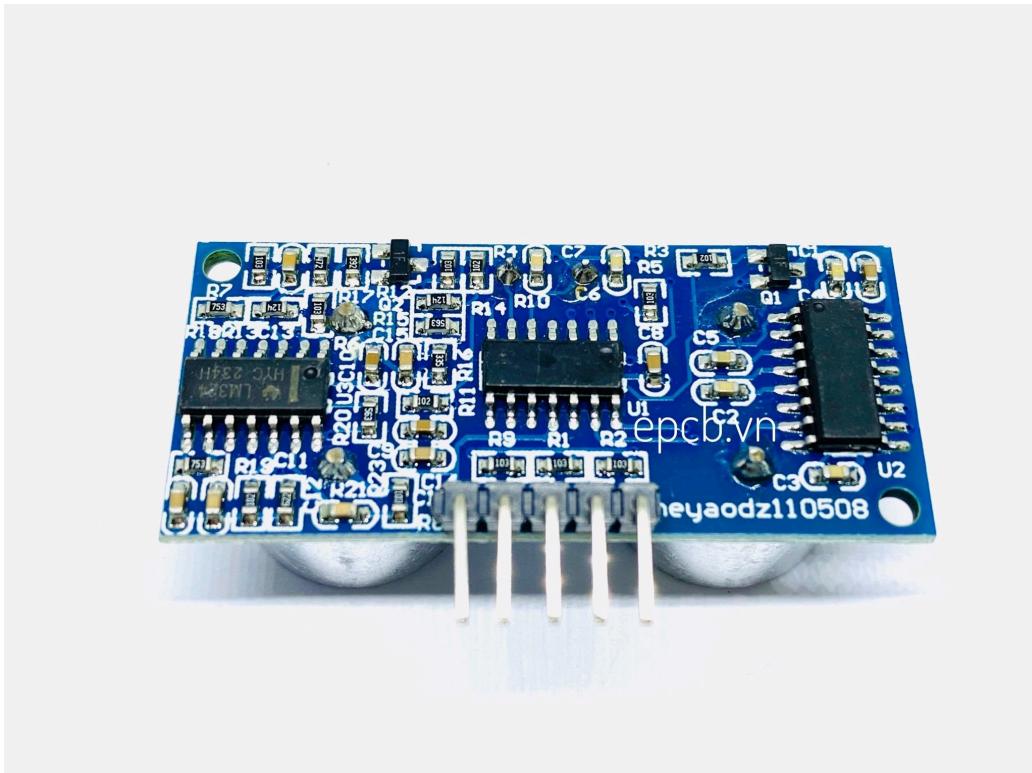
ESP32 là một dòng vi điều khiển (microcontroller) System-on-Chip (SoC) chi phí thấp và tiết kiệm năng lượng được phát triển bởi Espressif Systems. Điểm nổi bật chính của nó là việc tích hợp sẵn các module Wi-Fi và Bluetooth trên cùng một con chip, giúp nó trở thành một lựa chọn lý tưởng cho các dự án Internet of Things (IoT).

- Vi điều khiển: Lõi xử lý Tensilica Xtensa LX6 32-bit cung cấp hiệu năng mạnh mẽ để xử lý các tác vụ phức tạp.
- Kết nối không dây: Hỗ trợ Wi-Fi (802.11 b/g/n) và Bluetooth (cô điện và BLE -

Bluetooth Low Energy)

- Ngoại vi: Tích hợp nhiều chân GPIO (General Purpose Input/Output), các chuẩn giao tiếp như SPI, I2C, UART, cùng với bộ chuyển đổi ADC (Analog-to-Digital) và DAC (Digital-to-Analog).
- Chip nạp chương trình (CH340/CP2102): Đây là các chip chuyển đổi tín hiệu từ USB sang UART, đóng vai trò trung gian để máy tính có thể giao tiếp và nạp chương trình cho vi điều khiển ESP32 qua cổng USB.

b) Cảm biến siêu âm (SRF05)



Cảm biến siêu âm SRF05 được sử dụng để đo khoảng cách đến một vật thể mà không cần tiếp xúc. Nguyên lý hoạt động của nó dựa trên việc phát và thu sóng siêu âm (sóng âm có tần số cao hơn ngưỡng tai người nghe được).

- Nguyên lý hoạt động:
 - + Cảm biến phát ra một xung sóng siêu âm ngắn từ chân TRIG (Trigger).
 - + Sóng này di chuyển trong không khí, khi gặp vật cản sẽ phản xạ lại.
 - + Cảm biến nhận lại sóng phản xạ này tại chân ECHO.
 - + Vì điều khiển sẽ đo khoảng thời gian (t) từ lúc phát xung đi cho đến khi nhận được xung về.
- Tính toán khoảng cách: Khoảng cách (d) được tính bằng công thức:

$$d = \frac{v * t}{2}$$

Trong đó:

- + v là vận tốc của sóng âm trong không khí (xấp xỉ 343 m/s).

- + t là thời gian đo được.
- + Chúng ta chia cho 2 vì thời gian t là tổng thời gian sóng đi và về.

c) Servo motor (SG90)



Servo motor (động cơ servo) là một loại động cơ điện đặc biệt cho phép điều khiển chính xác vị trí góc quay của trục động cơ. Servo SG90 là một loại servo mini rất phổ biến, giá rẻ và thường được sử dụng trong các dự án nhỏ.

- Cấu tạo: Bao gồm một động cơ DC, một hộp số giảm tốc, một biến trở (potentiometer) để theo dõi vị trí và một mạch điều khiển.
- Nguyên lý điều khiển: Vị trí của trục servo được điều khiển bằng cách gửi một chuỗi xung tín hiệu điều khiển (PWM - Pulse Width Modulation) đến dây tín hiệu của nó. Độ rộng của xung sẽ quyết định góc quay của servo.
 - + Ví dụ, với servo SG90, xung rộng 1ms tương ứng với góc 0 độ, 1.5ms tương ứng với góc 90 độ (vị trí trung tâm), và 2ms tương ứng với góc 180 độ.
- Ứng dụng: Thường được dùng trong các cơ cấu cần sự chuyển động chính xác như cánh tay robot, cơ cấu lái của xe mô hình, hoặc điều khiển các van, khớp nối.

4. Blynk Cloud

- Định nghĩa: Blynk IoT là một nền tảng IOT platform giúp bạn dễ dàng kết nối và điều khiển các thiết bị IoT từ xa qua internet. Server Blynk đóng vai trò trung gian, xử lý các yêu cầu từ ứng dụng Blynk IOT và các thiết bị IoT như ESP32
- Ưu điểm:
 - + Dễ sử dụng: Blynk có giao diện thân thiện và trực quan, giúp bạn dễ dàng tạo các dự án IoT mà không cần nhiều kiến thức chuyên sâu về lập trình.
 - + Đa nền tảng: Ứng dụng Blynk IOT hoạt động trên cả Android và iOS, cho phép bạn giám sát và điều khiển thiết bị từ bất kỳ thiết bị di động nào.
 - + Thời gian thực: Dữ liệu từ các thiết bị IoT được cập nhật liên tục và hiển thị ngay trên ứng dụng Blynk IOT, giúp bạn giám sát và phản hồi kịp thời.
 - + Thư viện phong phú: Blynk hỗ trợ nhiều loại vi điều khiển như ESP32, Arduino, và Raspberry Pi, với thư viện phong phú và dễ tích hợp.
 - + Bảo mật: Sử dụng mã xác thực (Auth Token) để kết nối và bảo vệ thông tin giữa ứng dụng Blynk IOT và các thiết bị IoT.
- Nhược điểm:
 - + Giới hạn miễn phí: Phiên bản miễn phí của Blynk có giới hạn về số lượng widget và thiết bị mà bạn có thể sử dụng. Để sử dụng nhiều hơn, bạn cần nâng cấp lên phiên bản trả phí.
 - + Phụ thuộc internet: Blynk yêu cầu kết nối internet liên tục để hoạt động. Điều này không phù hợp cho các ứng dụng cần hoạt động ngoại tuyến.
 - + Chi phí nâng cấp: Để sử dụng đầy đủ các tính năng và không bị giới hạn, bạn cần trả phí để nâng cấp tài khoản.

5. Cấu trúc của hệ thống bãi đỗ xe thông minh

Hệ thống được xây dựng theo mô hình 4 lớp:

- **Lớp cảm biến:**
 - + ESP32 (Kit Wifi + Bluetooth, CH340/CP2102): vi điều khiển trung tâm, chịu trách nhiệm thu thập dữ liệu cảm biến và điều khiển thiết bị chấp hành.
 - + SRF05 – Module cảm biến siêu âm: phát hiện sự có mặt của phương tiện tại các vị trí đỗ xe bằng cách
 - + SG90 – Servo motor: điều khiển thanh chắn (barie) tự động mở khi xe vào/ra.

⇒ Các thiết bị này sẽ gửi tín hiệu thời gian thực đến bộ điều khiển trung tâm để

xử lý sự kiện.

- **Lớp mạng:** ESP32 kết nối Internet thông qua Wi-Fi và sử dụng giao thức MQTT để truyền dữ liệu. Dữ liệu cảm biến được gửi đến nền tảng xử lý đám mây Blynk Cloud để quản lý sự kiện và điều khiển từ xa.
- **Lớp xử lý dữ liệu (Processing Layer):** Blynk Cloud: đảm nhận vai trò xử lý backend, quản lý luồng dữ liệu, sự kiện vào/ra bãi đỗ và giao tiếp với giao diện ứng dụng. ⇒ Hệ thống xử lý trung gian có thể thực hiện các chức năng nâng cao như phát hiện bất thường, thống kê công suất sử dụng bãi xe.
- **Lớp ứng dụng (Application Layer):** Giao diện người dùng Web Application: cho phép khách hàng theo dõi tình trạng bãi xe, quản lý được quyền kiểm soát barie và truy xuất dữ liệu.