

HỆ THỐNG BÃI ĐỖ XE THÔNG MINH

Đề xuất dự án ứng dụng công nghệ AIoT

Ngày 22 tháng 5 năm 2025

Tóm tắt nội dung

Báo cáo này trình bày đề xuất phát triển hệ thống bãi đỗ xe thông minh ứng dụng công nghệ AIoT, nhằm tự động hóa và tối ưu hóa quy trình quản lý bãi đỗ xe. Giải pháp đề xuất kết hợp các công nghệ nhận diện hình ảnh (biển số xe, khuôn mặt người lái) và hệ thống cảm biến IoT để theo dõi tình trạng chỗ đỗ trong thời gian thực, loại bỏ thao tác thủ công và giảm thiểu sai sót trong quá trình vận hành. Các tính năng chính bao gồm: tự động nhận diện phương tiện ra vào, giám sát trạng thái chỗ đỗ, lưu trữ dữ liệu trên nền tảng cloud và cung cấp giao diện dashboard trực quan cho người quản lý. Hệ thống có thể triển khai linh hoạt tại nhiều môi trường khác nhau như trường học, khu dân cư, bãi xe công cộng, góp phần xây dựng hạ tầng đô thị thông minh và hiện đại.

Mục lục

1	Giới thiệu	3
1.1	Bối cảnh	3
1.2	Giải pháp	3
1.3	Phát Biểu Bài toán	4
1.3.1	Dữ liệu đầu vào (input)	4
1.3.2	Kết quả đầu ra (ouput)	4
1.3.3	Giới hạn hệ thống	4
1.3.4	Phạm vi triển khai	5
1.4	Đóng góp chính của đề tài	5
2	Kiến trúc phần mềm	5
2.1	Thiết bị đầu vào (Input Layer)	6
2.2	Tầng xử lý trung tâm (Processing Layer)	6
2.3	Thiết bị điều khiển - phản hồi (Output Layer)	6
2.4	Nền tảng lưu trữ	7
3	Công nghệ	8
3.1	Công nghệ sử dụng	8
3.1.1	Hệ thống phần cứng	8
3.1.2	Ứng dụng AI phân tích hình ảnh	15
3.1.3	Giao diện người dùng và quản trị (chưa quyết định)	16
3.1.4	Kết nối và nền tảng lưu trữ (chưa quyết định)	16
3.2	Thiết bị	16
4	Kế hoạch thực hiện	18
4.1	Phân công nhân lực	18
4.2	Timeline	19
4.3	Chi phí thực hiện (chi phí thiết bị, chi phí làm)	19

1 Giới thiệu

1.1 Bối cảnh

Hiện nay, nhiều bãi đỗ xe tại các khu vực như chung cư, trường học, tòa nhà văn phòng đã bắt đầu áp dụng phần mềm và thẻ từ (hoặc thẻ RFID) để quản lý việc ra vào. Tuy nhiên, các hệ thống này chủ yếu chỉ dừng lại ở mức cơ bản: người dùng quét thẻ khi vào – ra, hệ thống ghi lại thời gian và biển số xe (nếu có camera phụ trợ), sau đó lưu dữ liệu cục bộ. Chúng không kết nối với các cảm biến để giám sát tình trạng từng vị trí đỗ, không cung cấp thông tin theo thời gian thực cho người dùng hoặc người quản lý, và thường thiếu khả năng đồng bộ dữ liệu lên nền tảng trực tuyến để truy xuất khi có sự cố.

Hệ quả là việc vận hành vẫn phụ thuộc nhiều vào thao tác thủ công, dễ xảy ra sai sót như mất thẻ, quên quét thẻ, hoặc ghi nhận sai thời gian. Đồng thời, người dùng cũng gặp bất tiện khi không thể biết trước bãi còn chỗ hay không, dẫn đến mất thời gian di chuyển và tìm kiếm chỗ đỗ. Trong bối cảnh đô thị hóa nhanh chóng, nhu cầu tự động hóa và tối ưu hóa quản lý bãi đỗ xe trở nên cấp thiết hơn bao giờ hết.

Công nghệ AIoT – kết hợp giữa trí tuệ nhân tạo (AI) và Internet vạn vật (IoT) – mang đến giải pháp khả thi để xây dựng các bãi đỗ xe thông minh: tự động nhận diện xe qua camera, cập nhật tình trạng chỗ đỗ theo thời gian thực, hỗ trợ giám sát tập trung và nâng cao trải nghiệm người dùng, hướng tới một hệ thống hạ tầng đô thị hiện đại và hiệu quả hơn.

1.2 Giải pháp

Nhóm đề xuất phát triển một hệ thống **Bãi đỗ xe thông minh**, nhằm tự động hóa và tối ưu hóa quy trình quản lý bãi xe. Hệ thống bao gồm các chức năng chính sau:

- **Tự động nhận diện biển số xe** hoặc **khuôn mặt tài xế** khi phương tiện ra vào, giúp loại bỏ thao tác thủ công như quét thẻ hay ghi tay.
- **Ghi nhận thời điểm vào – ra một cách chính xác**, đồng bộ dữ liệu theo thời gian thực.
- **Giám sát tình trạng chỗ đỗ (còn/trống)** thông qua cảm biến đặt tại từng vị trí và hiển thị bằng hệ thống đèn báo trực quan.

- **Lưu trữ toàn bộ dữ liệu** về phương tiện, người dùng và lịch sử ra vào trên nền tảng cloud (ví dụ như Firebase hoặc ThingsBoard), hỗ trợ truy xuất và phân tích khi cần.
- **Cung cấp giao diện dashboard trực quan**, giúp ban quản lý dễ dàng theo dõi tình trạng bãi xe, thống kê lưu lượng phương tiện và kiểm tra lịch sử hoạt động.

Hệ thống này không chỉ giúp **rút ngắn thời gian xử lý, giảm sai sót do con người**, mà còn có thể **triển khai linh hoạt** tại các khu vực như **trường học, khu dân cư, bãi xe công cộng hoặc cơ quan hành chính**, góp phần xây dựng hạ tầng đô thị thông minh và hiện đại.

1.3 Phát Biểu Bài toán

1.3.1 Dữ liệu đầu vào (input)

- Hình ảnh/video từ camera tại lối vào và lối ra bãi xe.

1.3.2 Kết quả đầu ra (output)

- Kết quả nhận diện biển số hoặc khuôn mặt tại thời điểm vào – ra.
- Ghi nhận và lưu trữ thời điểm vào – ra xe.
- Hiển thị trạng thái chỗ đỗ (còn trống hoặc đã chiếm chỗ) theo thời gian thực.
- Giao diện trực quan cho người quản lý và người dùng.
- Dữ liệu được lưu trữ và truy xuất thông qua nền tảng cloud.

1.3.3 Giới hạn hệ thống

Hệ thống nhận diện khuôn mặt và biển số xe có các giới hạn như:

- **Điều kiện môi trường:** Độ chính xác giảm trong điều kiện ánh sáng yếu (ban đêm, bóng râm) hoặc thời tiết bất lợi (mưa, sương mù), ảnh hưởng đến khả năng trích xuất đặc trưng.
- **Chất lượng và vị trí camera:** Yêu cầu camera độ phân giải đủ và lắp đặt đúng góc nhìn để tránh nhận diện sai hoặc bỏ sót.
- **Tài nguyên phần cứng:** Cần đủ RAM, dung lượng lưu trữ và tốc độ mạng để đảm bảo hiệu suất thời gian thực.

- **Độ trễ:** Yêu cầu độ trễ xử lý thấp để đảm bảo tính tức thời trong kiểm soát ra vào.

1.3.4 Phạm vi triển khai

Hệ thống được thiết kế hướng đến việc triển khai tại các công ty, doanh nghiệp, nhà máy hoặc khu công nghiệp – nơi có nhu cầu kiểm soát người và phương tiện ra vào một cách tự động, chính xác và không cần can thiệp thủ công.

Cụ thể, hệ thống có thể được tích hợp vào cổng kiểm soát an ninh, trạm gác hoặc khu vực đỗ xe nội bộ. Mô hình hoạt động phù hợp với các đơn vị có sẵn hạ tầng mạng cục bộ ổn định và có khả năng đầu tư cơ bản vào thiết bị nhận diện như ESP32-CAM, camera, cảm biến và màn hình hiển thị.

Hiện tại, hệ thống chưa tích hợp chức năng thu phí. Tuy nhiên, đây là một khả năng có thể mở rộng trong tương lai nếu hệ thống được ứng dụng tại các bãi giữ xe công cộng hoặc khu vực dịch vụ có yêu cầu thanh toán tự động.

1.4 Đóng góp chính của đề tài

- Đề xuất mô hình hệ thống bãi đỗ xe thông minh tích hợp công nghệ **AIoT**, ứng dụng được vào thực tế.
- Tích hợp **nhận diện hình ảnh (biển số/khuôn mặt)** và **cảm biến IoT** để tự động hóa hoàn toàn quy trình vận hành.
- Thiết kế giao diện dashboard giúp quản lý dễ dàng theo dõi, giám sát và truy xuất lịch sử.
- Định hướng mở rộng và triển khai tại các bãi giữ xe trường học, tòa nhà, trung tâm thương mại...
- Hướng đến việc **giảm sai sót con người, nâng cao hiệu quả quản lý và tối ưu trải nghiệm người dùng**.

2 Kiến trúc phần mềm

Hệ thống được thiết kế dựa trên mô hình phân tầng, bao gồm các thành phần chính: thiết bị đầu vào (người dùng và camera ESP32-CAM), tầng xử lý trung tâm (AI model,

ESP32 và Django backend), và tầng điều khiển - phản hồi (thiết bị IoT, Dashboard và lưu trữ).

2.1 Thiết bị đầu vào (Input Layer)

- Người dùng (User) tương tác trực tiếp với hệ thống thông qua camera.
- **ESP32-CAM**:
 - Ghi nhận hình ảnh đầu vào (biển số hoặc khuôn mặt).
 - Gửi hình ảnh tới **mô hình AI** để xử lý nhận diện.

2.2 Tầng xử lý trung tâm (Processing Layer)

- **AI Model (cục bộ hoặc cloud)**:
 - Tiếp nhận hình ảnh từ ESP32-CAM.
 - Thực hiện nhận diện khuôn mặt hoặc biển số xe.
 - Gửi kết quả (hoặc đường dẫn ảnh nếu có dùng Cloudinary) về cho backend.
 - Có thể tải ảnh lên **Cloudinary**, nhận lại URL ảnh phục vụ lưu trữ/hiển thị.
- **ESP32 (vi điều khiển trung gian)**:
 - Nhận kết quả xử lý từ backend Django.
 - Truyền dữ liệu đến các thiết bị IoT điều khiển.
 - Phản hồi dữ liệu cảm biến và trạng thái thiết bị về cho backend.
- **Backend Django**:
 - Giao tiếp với Dashboard qua API.
 - Nhận dữ liệu nhận diện từ AI model.
 - Điều phối hoạt động của hệ thống (mở barrier, phát âm báo, hiển thị OLED...).
 - Lưu trữ/truy xuất dữ liệu trạng thái và nhật ký hệ thống lên **Firestore**.

2.3 Thiết bị điều khiển - phản hồi (Output Layer)

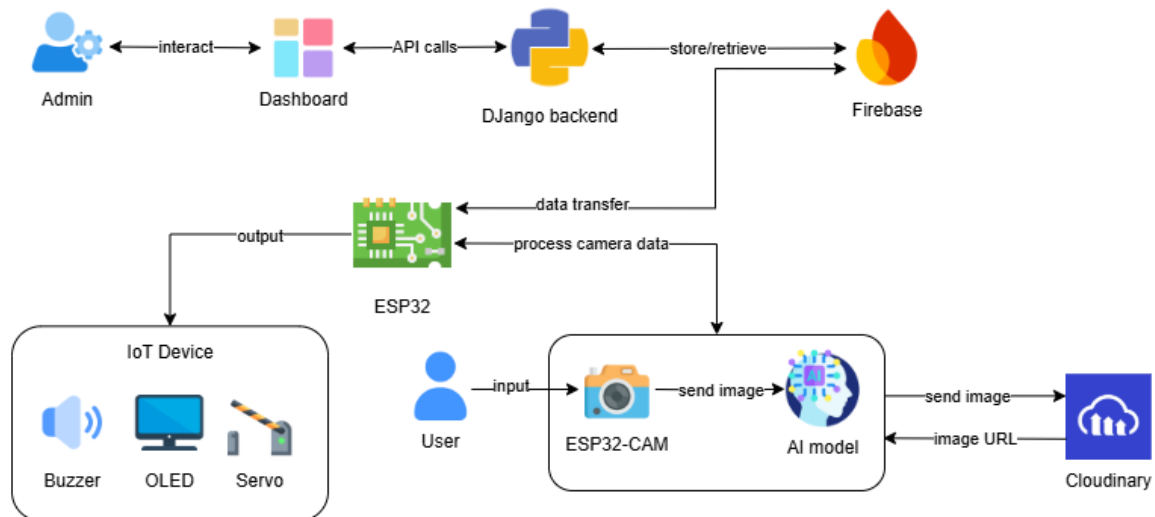
- **Thiết bị IoT gồm**:
 - **Servo (Barrier)**: mở/đóng rào chắn tự động.
 - **OLED Display**: hiển thị thông tin trực quan (số ô trống, lỗi, trạng thái...).
 - **Buzzer**: cảnh báo âm thanh cho các trạng thái khác nhau.
- **Dashboard (giao diện quản trị cho Admin)**:

- Giao tiếp với backend qua API.
- Hiển thị dữ liệu thời gian thực và lịch sử ra vào.
- Cho phép admin giám sát toàn bộ hệ thống và các thiết bị.

2.4 Nền tảng lưu trữ

- **Firebase:**
 - Lưu trữ dữ liệu hệ thống, nhật ký hoạt động, và thông tin người dùng.
 - Hỗ trợ đồng bộ dữ liệu và hiển thị dashboard.
- **Cloudinary:**
 - Dùng để lưu trữ ảnh chụp từ ESP32-CAM nếu cần gửi đi và truy xuất nhanh bằng URL.

Hệ thống vận hành dựa trên luồng dữ liệu: **User** → **ESP32-CAM** → **AI Model** → **Django Backend** → **ESP32** → **IoT Device**, kết hợp với giao diện **Dashboard** cho quản trị và các nền tảng lưu trữ giúp theo dõi, điều khiển và phân tích hoạt động một cách hiệu quả và an toàn.



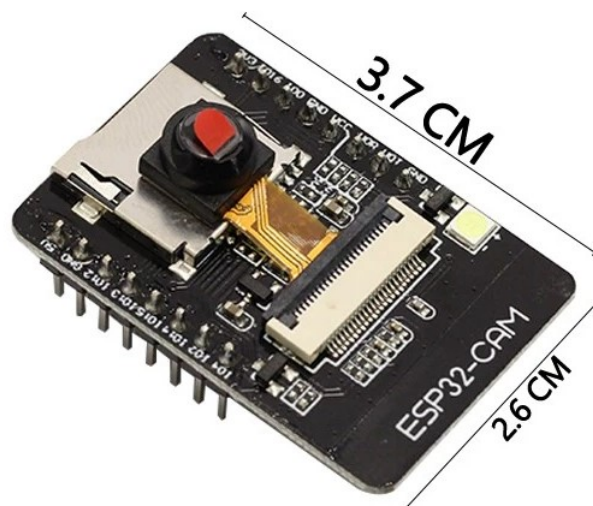
Hình 1: Sơ đồ kiến trúc hệ thống

3 Công nghệ

3.1 Công nghệ sử dụng

3.1.1 Hệ thống phần cứng

1. Camera ESP32-CAM (AI Thinker ESP32-CAM)



Hình 2: ESP32-CAM

ESP32-CAM là một module phát triển nhỏ gọn của hãng **AI. Thinker**, tích hợp sẵn:

Wi-Fi, Bluetooth, camera OV2640, khe cắm thẻ microSD, GPIO đa năng.

Module này được thiết kế chuyên dụng cho các ứng dụng thị giác máy tính, IoT, nhận diện khuôn mặt, giám sát an ninh, ...

ESP32-CAM là lựa chọn phổ biến nhờ giá rẻ, tính năng mạnh, và khả năng chạy AI trực tiếp trên thiết bị.

Thành phần	Thông tin chi tiết	Ghi chú
Vi xử lý chính	ESP32-WROOM-32	Xử lý chương trình nạp vào
Tốc độ xung nhịp	Lên tới 240 MHz	Xử lý được 240 triệu lệnh mỗi giây
RAM	520 KB SRAM nội	Lưu dữ liệu tạm thời khi chương trình đang chạy, tự động mất khi tắt nguồn
Flash	4 MB (SPI flash)	Lưu trữ chương trình được nạp vào chip, không mất khi tắt nguồn
Camera	OV2640, độ phân giải tối đa 1600x1200 (UXGA), hỗ trợ nhiều chế độ JPEG	Có LED flash trắng tích hợp (nối chân GPIO nếu dùng), không tự động lấy nét, hình bị mờ nếu đặt sai khoảng cách.
GPIO	~9 chân GPIO khả dụng	Các chân đa năng trên chip, cho phép nhận tín hiệu vào và gửi tín hiệu ra
Nguồn hoạt động	3.3V	Khi dùng nguồn 5V, cấp nguồn vào chân VCC/5V, bo mạch ESP32-CAM có sẵn mạch ổn áp AMS1117, chuyển 5V thành 3.3V cấp cho chip

Thành phần	Thông tin chi tiết	Ghi chú
Dòng tiêu thụ	160–250mA khi hoạt động	Nên dùng nguồn ổn định từ MB102 hoặc pin sạc 5V - 1A trở lên
Cổng USB	Không có	Cần dùng USB to UART để nạp code
Kích thước	27 x 40.5 x 4.5 mm	

Hệ thống sử dụng **2 module ESP32-CAM** cho mỗi cổng (vào và ra):

- Một module quét và nhận diện khuôn mặt tài xế
- Một module quét và nhận diện biển số xe

Mỗi ESP32-CAM hoạt động độc lập, được nạp sẵn chương trình, luôn bật camera và model nhận diện, có khả năng:

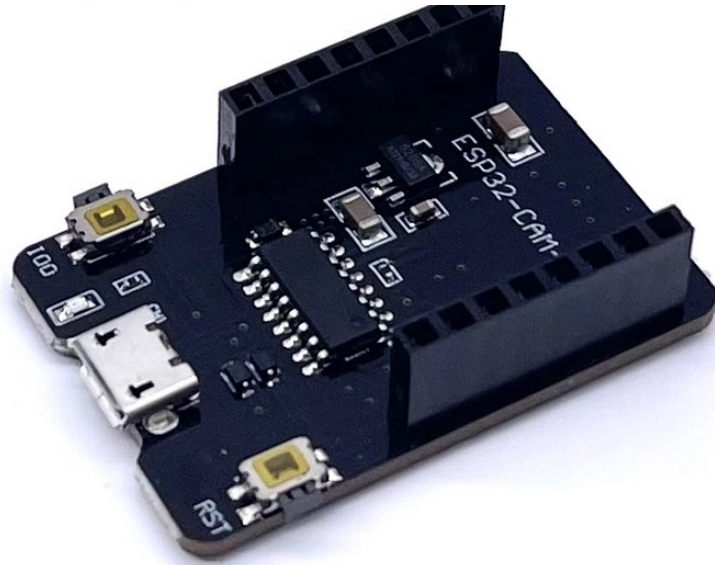
- Nhận diện sự xuất hiện của xe
- Tự động chụp ảnh
- Thực hiện nhận diện khuôn mặt hoặc biển số trực tiếp trên thiết bị
- Gửi kết quả nhận diện kèm timestamp về ESP32 trung tâm qua giao thức như **HTTP POST** hoặc **MQTT**

ESP32 trung tâm và 4 ESP32-CAM sẽ kết nối cùng một mạng wifi nội bộ, ESP32 trung tâm sẽ chạy server nhẹ để lắng nghe và mỗi ESP32-CAM đóng vai trò client.

Cấp nguồn: ESP32-CAM khi bật camera và Wi-Fi có thể tiêu thụ **160–300mA**, vì vậy cần tránh cấp nguồn chung cho cả 5 module. Ta sẽ sử dụng **sạc dự phòng** để cấp nguồn riêng cho ESP32-CAM. Ưu tiên sử dụng loại sạc dự phòng có 2 cổng USB-A, mỗi ESP32-CAM sẽ được cấp nguồn từ 1 cổng, cụ thể:

- Dây đỏ (VCC): nối từ cổng USB-A của sạc dự phòng đến chân VCC/5V của ESP32-CAM
- Dây đen (GND): nối từ cổng USB-A của sạc dự phòng đến chân GND của ESP32-CAM
- Chân GND khác của ESP32-CAM sẽ nối dây đến GND rail của breadboard

Nạp code:



Hình 3: Đế nạp ESP32-CAM

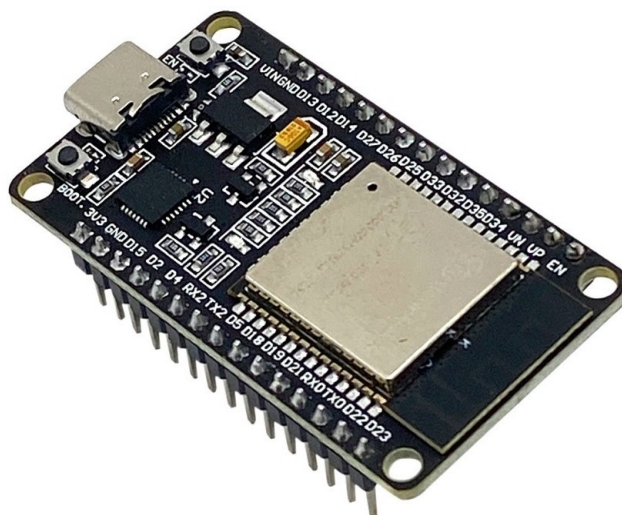
- Vì ESP32-CAM không có cổng USB, nên ta cần thiết bị trung gian để nạp code là **Đế Nạp ESP32-CAM**, sau khi nạp ESP32-CAM sẽ chạy độc lập và không mất code khi tắt nguồn. Đây là module chuyển đổi tín hiệu USB và UART, cho phép kết nối giữa máy tính và các thiết bị không có cổng USB trực tiếp như ESP32-CAM.
- Ngoài ra, khi cắm cáp USB từ sạc dự phòng vào đế, ESP32-CAM sẽ được cấp nguồn điện, không cần dây USB-to-dupont.

Lưu ý khi vừa nạp code, ESP32-CAM đang ở chế độ bootloader, cần ấn nút *reset* để chip khởi động lại chế độ chạy bình thường.

Đối với mô hình AI, sau khi train, ta sẽ chuyển sang định dạng nhẹ và export model ra dạng `.c` hoặc `.h` rồi include vào code.

2. ESP32 trung tâm (ESP32 DevKit V1)

ESP32 trung tâm là **bộ điều khiển chính** của toàn hệ thống, xử lý và điều phối giữa các thiết bị. Board sử dụng chip **ESP32-WROOM-32**, có thể xử lý nhiều tác vụ song song.



Hình 4: ESP32 DevKit V1

Thành phần	Thông tin chi tiết	Ghi chú
Chip chính	ESP32-WROOM-32	
RAM	520 KB	
Flash	4MB SPI Flash	
Tốc độ xử lý	Lên đến 240MHz	
Điện áp logic	3.3V	Thiết bị gửi tín hiệu 5V vào ESP32 GPIO có thể gây cháy GPIO, cần sử dụng mạch chia áp
USB-to-Serial	Tích hợp sẵn CP2102	Có thể nạp code trực tiếp qua cổng USB-C / microUSB
GPIO khả dụng	~25 chân, đủ dùng cho nhiều ngoại vi đồng thời	
Wi-Fi / Bluetooth	Tích hợp sẵn, dùng được ở chế độ Station và Access Point	Với ESP32 và 4 ESP32-CAM, ta sẽ dùng chế độ Station, nghĩa là cùng kết nối wifi có sẵn

3. Breadboard

Là bảng mạch kết nối không cần hàn, được sử dụng để phân phối nguồn điện và kết nối các thiết bị với nhau.

Cho phép kết nối linh kiện một cách linh hoạt thông qua dây jumper, giúp xây dựng và tổ chức hệ thống điện tử một cách gọn gàng, dễ bảo trì và dễ mở rộng.

Cấp nguồn:

- Trong hệ thống này, breadboard sẽ được cấp nguồn trực tiếp từ adapter 5V – 3A thông qua cáp chia nguồn.
- Mỗi đầu ra được gắn jack DC cái chuyển sang dây jumper:
 - Dây đỏ (VCC) được nối vào rail 5V của breadboard
 - Dây đen (GND) được nối vào rail GND
- Dây jumper là loại dây cắm sẵn đầu, chuyên dùng để kết nối các thành phần trong mạch điện tử mà không cần hàn, đặc biệt là trong hệ thống kết nối qua breadboard.

Loại dây	Đầu cắm	Ứng dụng
Male-Male	Đầu cắm kim 2 bên	Dùng để cắm từ chân này sang chân khác trên breadboard hoặc từ breadboard đến board mạch (ESP32, MB102...)
Male-Female	Một đầu kim, một đầu lỗ	Dùng để nối module/cảm biến (có chân đực) với breadboard hoặc board mạch
Female-Female	Hai đầu lỗ	Dùng để nối giữa 2 thiết bị đều có chân đực , ví dụ ESP32-CAM và FTDI , hoặc giữa module logic với module khác

4. Màn hình OLED

Là màn hình đơn sắc kích thước 0.96 inch, giao tiếp bằng chuẩn **I2C (SDA, SCL)** với ESP32 trung tâm (thường là GPIO21 và GPIO22). Trong hệ thống này ta sử dụng màn hình OLED với IC điều khiển SSD1306, điện áp hoạt động 5V (có tương thích 3.3V).

Có chức năng hiển thị:

- Khi xe vào: Hiển thị số chỗ còn trống, hoặc báo “đã hết chỗ” nếu không còn slot đỗ
- Khi xe ra: Hiển thị lời chào hoặc hiển thị lỗi nếu quá trình nhận diện không hợp lệ

Chân OLED	Kết nối đến	Ghi chú
VCC	VCC của breadboard	Cấp điện 5V từ MB102 / breadboard
GND	GND của breadboard	Mass
SCL	GPIO22 của ESP32	Tạo xung đồng bộ, do ESP32 điều khiển, cho OLED biết khi nào nhận 1 bit
SDA	GPIO21 của ESP32	Truyền dữ liệu đọc và ghi

5. Buzzer

Sử dụng **2 buzzer loại passive 5V**, mỗi chiếc được điều khiển bởi ESP32 trung tâm thông qua chân GPIO.

Passive buzzer là loại không có mạch dao động tích hợp bên trong, vì vậy **có thể phát được nhiều loại âm thanh khác nhau**. ESP32 sẽ tạo ra các tín hiệu tần số khác nhau (qua lệnh `tone()`), giúp buzzer phát ra các âm báo tùy biến:

- Âm báo thành công
- Âm báo lỗi

Giúp người dùng **phân biệt ngữ cảnh thông qua loại âm phát ra**.

Buzzer passive hoạt động ổn định ở **điện áp 5V**, được cấp nguồn từ breadboard thông qua module MB102, và điều khiển bằng tín hiệu logic 3.3V từ ESP32 mà không cần mạch

khuếch đại.

Chân buzzer	Kết nối đến	Ghi chú
Chân dương (+)	5V từ breadboard (MB102)	Cấp nguồn hoạt động; không nên lấy từ ESP32
Chân âm (–)	GPIO bất kỳ của ESP32	Dùng để phát âm bằng tín hiệu <code>tone()</code>
GND chung	GND breadboard	GND của buzzer và ESP32 phải nối chung

6. Servo motor MG90S

MG90S là một loại **servo mini** với **moment xoắn**, có cấu trúc **bánh răng kim loại**, bền hơn và chịu lực tốt hơn, phù hợp để điều khiển cơ cấu vật lý như **thanh chắn**.

Servo hoạt động ở **điện áp 5V**, tiêu thụ dòng khoảng **250–400mA** khi **tải nặng**, do đó cần cấp nguồn ổn định **từ MB102** qua breadboard để tránh sụt áp hoặc làm **ESP32** **reset** đột ngột.

ESP32 **trung tâm điều khiển servo** qua **tín hiệu PWM** từ một chân GPIO bất kỳ (thường dùng GPIO13 hoặc GPIO14). Tín hiệu PWM xác định góc quay của servo (trong khoảng 0° – 180°).

Chân servo	Kết nối đến	Ghi chú
VCC (đỏ)	5V từ MB102 qua rail breadboard	Cấp nguồn chính
GND (nâu)	GND chung trên breadboard	Cực âm
Signal (vàng)	GPIO13 (hoặc bất kỳ) trên ESP32	Điều khiển PWM

3.1.2 Ứng dụng AI phân tích hình ảnh

- **Nhận diện khuôn mặt**

- **Phát hiện khuôn mặt:** Sử dụng mô hình **YOLOv6** để phát hiện vùng chứa khuôn mặt (bounding box) trong ảnh. Mô hình có tốc độ nhanh, độ chính xác cao, phù hợp xử lý ảnh thời gian thực.
- **Nhận diện khuôn mặt:** Dựa trên các vector đặc trưng được tạo bởi mô hình **FaceNet**, so sánh với database để xác định danh tính.

- **Nhận diện biển số xe**

- **Phát hiện biển số xe:** Sử dụng mô hình **YOLOv8** để phát hiện vùng chứa biển số xe trong ảnh.
- **Nhận dạng ký tự biển số:** Cắt vùng biển số từ ảnh dựa trên kết quả phát hiện, áp dụng thuật toán nhận dạng ký tự **Tesseract OCR** để trích xuất chuỗi ký tự.

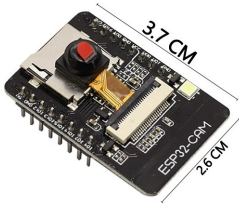
3.1.3 Giao diện người dùng và quản trị (chưa quyết định)

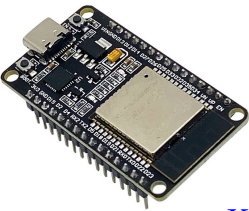




- **Frontend:** Sử dụng các công nghệ phổ biến HTML, CSS, JavaScript để xây dựng giao diện thân thiện, dễ sử dụng cho tài xế và quản trị viên.
- **Backend:** Sử dụng Django làm framework backend, xử lý các logic nghiệp vụ như xác thực người dùng, quản lý dữ liệu xe và vị trí bãi đỗ, cung cấp API cho frontend và thiết bị IoT kết nối.

3.1.4 Kết nối và nền tảng lưu trữ (chưa quyết định)

- **MQTT (Message Queuing Telemetry Transport):**
- **Firebase:**

3.2 Thiết bị

STT	Tên thiết bị	Số lượng	Giá/cái	Hình ảnh	Nguồn liên kết
1	AI Thinker ESP32-CAM	4	165,000		Xem tại đây

STT	Tên thiết bị	Số lượng	Giá/cái	Hình ảnh	Nguồn liên kết
2	ESP32 DevKit V1 (CP2102, Type-C)	1	97,000		Xem tại đây
3	Màn hình OLED 0.96 inch	2	55,000		Xem tại đây
4	Servo motor MG90S	2	37,000		Xem tại đây
5	Buzzer	2	5,000		Xem tại đây
6	Adapter 5V - 3A	1	52,000		Xem tại đây

STT	Tên thiết bị	Số lượng	Giá/cái	Hình ảnh	Nguồn liên kết
7	Bộ chia nguồn DC male	1	26,000		Xem tại đây
8	Jack DC cái có dây 5.5x2.1mm	5	3,000		Xem tại đây
9	Breadboard	1	17,000		Xem tại đây
10	Dây jumper sợi/loại	40	66,000		Xem tại đây
11	Đế Nạp ESP32-CAM	1	30,000		Xem tại đây

4 Kế hoạch thực hiện

4.1 Phân công nhân lực

Thành viên	Vai trò chính	Nhiệm vụ cụ thể
Cao Uyển Nhi	Trưởng nhóm, quản lý tiến độ	Phụ trách AI – Nhận diện biển số/khuôn mặt Tổng hợp báo cáo, điều phối nhóm, hỗ trợ kỹ thuật Thu thập dữ liệu, huấn luyện mô hình, triển khai xử lý ảnh
Trần Thị Cát Tường	Phụ trách IoT – Cảm biến và cloud	Kết nối cảm biến, lập trình ESP32, gửi dữ liệu lên Firebase/ThingsBoard
Võ Lê Việt Tú	Thiết kế phần cứng – Mạch, cảm biến	Lắp ráp hệ thống, bố trí camera và sensor thực tế
Lưu Thanh Thuý	Giao diện & báo cáo	Thiết kế dashboard, lập trình hiển thị và làm tài liệu

4.2 Timeline

Tuần	Công việc chính
1–2	Nghiên cứu tài liệu, xác định yêu cầu, phân công nhiệm vụ
3–4	Thu thập dữ liệu khuôn mặt/biển số, khảo sát thiết bị cần thiết
5–6	Triển khai mô hình AI nhận diện, kết nối cảm biến với ESP32
7–8	Thiết kế giao diện dashboard, xây dựng cơ sở dữ liệu trên cloud
9–10	Tích hợp toàn hệ thống: AI + IoT + giao diện
11	Chạy thử, hiệu chỉnh, test toàn bộ quy trình thực tế
12	Viết báo cáo, chuẩn bị slide thuyết trình, hoàn thiện demo

4.3 Chi phí thực hiện (chi phí thiết bị, chi phí làm)

Tên thiết bị	Số lượng	Đơn giá	Thành tiền
AI Thinker ESP32-CAM	4	165,000	660,000
ESP32 DevKit V1 (CP2102, Type-C)	2	97,000	194,000

Tên thiết bị	Số lượng	Đơn giá	Thành tiền
Màn hình OLED 0.96 inch	2	55,000	110,000
Servo motor MG90S	2	37,000	74,000
Buzzer	2	5,000	10,000
Module nguồn MB102	2	17,000	34,000
Breadboard	2	17,000	34,000
Dây jumper	40 sợi	66,000	66,000
Đế Nạp ESP32-CAM	4	30,000	120,000
Pin Sạc Dự Phòng	4	245,000	980,000
Vật tư lắp đặt, khung mô hình	-	-	200,000
Tổng chi phí			2,482,000