

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

LÊ XUÂN TÙNG

**XÂY DỰNG HỆ THỐNG THU THẬP VÀ
QUẢN LÝ THÔNG TIN TỪ CAMERA GIÁM SÁT
DỰA TRÊN NỀN TẢNG FIWARE**

LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

HÀ NỘI - 2020

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

LÊ XUÂN TÙNG

**XÂY DỰNG HỆ THỐNG THU THẬP VÀ
QUẢN LÝ THÔNG TIN TỪ CAMERA GIÁM SÁT
DỰA TRÊN NỀN TẢNG FIWARE**

Ngành: Khoa học máy tính

Chuyên ngành: Khoa học máy tính

Mã số: 8480101.01

LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

NGƯỜI HƯỚNG DẪN KHOA HỌC:
PGS. TS. Lê Thanh Hà
TS. Ngô Thị Duyên

HÀ NỘI - 2020

MỤC LỤC

MỤC LỤC	i
LỜI CẢM ƠN	ii
LỜI CAM ĐOAN	iii
TÓM TẮT	iv
DANH MỤC HÌNH VẼ	v
DANH MỤC BẢNG BIỂU	vi
Chương 1. Giới thiệu	1
1.1. Đặt vấn đề	1
1.2. Các nghiên cứu liên quan	3
1.3. Những đóng góp của luận văn	4
1.4. Cấu trúc luận văn	5
Chương 2. Tổng quan về nền tảng FIWARE	6
2.1. Orion Context Broker	6
2.1.1. MongoDB Database	7
2.1.2. NGSI-v2 API và cách thức hoạt động	8
2.2. Keyrock Identity Manager	12
2.3. Wilma PEP Proxy	14
Chương 3. Thiết kế và cài đặt hệ thống	16
3.1. Kiến trúc tổng quan hệ thống	16
3.2. Phân hệ thu nhận và trích rút thông tin ngữ nghĩa từ camera	18
3.3. Phân hệ lưu trữ, quản lý thông tin ngữ nghĩa	21
3.4. Phân hệ quản lý định danh và quyền người dùng	23
Chương 4. Phát triển ứng dụng tìm bãi đỗ xe và đánh giá kết quả	28
4.1. Xây dựng kịch bản kiểm thử khả năng ứng dụng của hệ thống	28
4.2. Cài đặt thuật toán đếm số chỗ trống trong bãi đỗ xe	28
4.3. Thiết kế và cài đặt ứng dụng	32
4.4. Thủ nghiệm ứng dụng	36
4.5. Đánh giá hệ thống	42
KẾT LUẬN	47
TÀI LIỆU THAM KHẢO	48

LỜI CẢM ƠN

Trước tiên tôi xin dành lời cảm ơn chân thành và sâu sắc đến thầy giáo, PGS. TS. Lê Thanh Hà và cô giáo, TS. Ngô Thị Duyên – người đã hướng dẫn, khuyến khích, chỉ bảo và tạo cho tôi những điều kiện tốt nhất từ khi bắt đầu cho tới khi hoàn thành công việc của mình.

Tôi xin dành lời cảm ơn chân thành tới các thầy cô giáo khoa Công nghệ thông tin, trường Đại học Công nghệ, ĐHQGHN đã tận tình đào tạo, cung cấp cho tôi những kiến thức vô cùng quý giá và đã tạo điều kiện tốt nhất cho tôi trong suốt quá trình học tập, nghiên cứu tại trường.

Đồng thời tôi xin cảm ơn tất cả những người thân yêu trong gia đình tôi cùng toàn thể bạn bè, những người đã luôn giúp đỡ, động viên tôi những khi vấp phải những khó khăn, bế tắc.

Cuối cùng, tôi xin chân thành cảm ơn các bạn học cùng khóa đã giúp đỡ, động viên tôi trong học tập và nghiên cứu chương trình thạc sĩ tại Đại học Công nghệ, ĐHQGHN.

LỜI CAM ĐOAN

Tôi xin cam đoan rằng luận văn thạc sĩ công nghệ thông tin “Xây dựng hệ thống thu thập và quản lý thông tin từ camera giám sát dựa trên nền tảng FIWARE” là công trình nghiên cứu của riêng tôi, không sao chép lại của người khác. Trong toàn bộ nội dung của luận văn, những điều đã được trình bày hoặc là của chính cá nhân tôi hoặc là được tổng hợp từ nhiều nguồn tài liệu. Tất cả các nguồn tài liệu tham khảo đều có xuất xứ rõ ràng và hợp pháp.

Tôi xin hoàn toàn chịu trách nhiệm và chịu mọi hình thức kỷ luật theo quy định cho lời cam đoan này.

Hà Nội, ngày tháng ... năm

.....

TÓM TẮT

Hiện nay tại Việt Nam, hệ thống camera thông minh (CCTV) đã và đang được sử dụng phổ biến ở rất nhiều đơn vị và tổ chức khác nhau nhằm giám sát, tăng cường an ninh khu vực. Tất cả các hệ thống camera này đều chỉ phụ vụ mục đích lưu trữ, quan sát trực tiếp, và xem lại video khi cần. Các chức năng thông minh tự động khai thác thông tin hình ảnh hầu như không có, và cũng chưa có một nền tảng nào có thể kết nối với các hệ thống camera khác nhau để tổng hợp thông tin từ chúng. Do đó, rất cần thiết xây dựng một hạ tầng kết nối và khai thác nguồn tài nguyên dữ liệu hình ảnh từ các hệ thống CCTV này nhằm phục vụ các nhu cầu thông tin của xã hội. Nắm bắt nhu cầu trên, luận văn này tập trung phát triển một hệ thống cho phép thu nhận, trích rút, quản lý thông tin có ý nghĩa từ hệ thống camera giám sát, để từ đó có thể đưa ra các cảnh báo, phân tích hoặc phát triển các ứng dụng thông minh cho người dùng, dựa trên việc kết hợp sức mạnh của điện toán cạnh và điện toán đám mây. Ngoài ra, bài toán tìm bãi đỗ xe ô-tô trong đô thị cũng là một bài toán xã hội cấp thiết, nên luận văn này đã áp dụng một thuật toán đếm số lượng chỗ đỗ xe trống trong bãi đỗ từ dữ liệu hình ảnh dựa trên kiến trúc Mask R-CNN vào hệ thống trên, và xây dựng một ứng dụng bản đồ chỉ đường được tích hợp tính năng tìm bãi đỗ xe còn trống dựa vào các thông tin đó. Ứng dụng này không chỉ giải quyết được một vấn đề thực tế mà còn cho thấy tính khả thi, tiềm năng phát triển và ý nghĩa thực tiễn rất cao của hệ thống được đề xuất.

Từ khóa: CCTV, điện toán cạnh, điện toán đám mây, Mask R-CNN.

DANH MỤC HÌNH VẼ

Hình 1.1 Các hệ thống camera ở các vị trí khác nhau	2
Hình 2.1 So sánh tốc độ truy vấn của Mongo và SQL	8
Hình 2.2 Mô hình hoạt động của Orion Context Broker	9
Hình 2.3 Hệ sinh thái Google và FIWARE	12
Hình 2.4 Kiến trúc của Keyrock	13
Hình 2.5 Giao diện đăng nhập của Keyrock	14
Hình 2.6 Cách thức hoạt động của Wilma PEP Proxy	15
Hình 3.1 Thiết kế chi tiết của hệ thống	17
Hình 3.2 Các công việc được xử lý bởi một máy chủ cạnh	19
Hình 3.3 Thiết kế dữ liệu JSON cho bài toán tìm bãі đỗ xe	20
Hình 3.4 Thiết kế và cách thức hoạt động của phân hệ lưu trữ, quản lý thông tin ngữ nghĩa	22
Hình 3.5 Sơ đồ hoạt động của phân hệ quản lý định danh và quyền người dùng	24
Hình 3.6 Sơ đồ kết nối của hệ thống	25
Hình 3.7 Giao diện quản lý ứng dụng đã đăng ký trong Keyrock	26
Hình 3.8 Thiết lập quyền cho từng đối tượng người dùng trong Keyrock	27
Hình 4.1 Hình ảnh bãі đỗ xe là đầu vào của thuật toán	29
Hình 4.2 Quá trình nhận biết vị trí các ô đỗ xe và các xe ô-tô	30
Hình 4.3 Công thức tính độ che phủ IoU	31
Hình 4.4 Kết quả thực nghiệm của thuật toán đếm số lượng vị trí đỗ xe còn trống	31
Hình 4.5 Biểu đồ use-case của ứng dụng	32
Hình 4.6 Sơ đồ khái quát trình trích rút dữ liệu ngữ nghĩa	33
Hình 4.7 Luồng xác minh định danh và quyền của ứng dụng	34
Hình 4.8 Sơ đồ khái của quá trình hiển thị các bãі đỗ xe và số lượng ô đỗ xe trống	35
Hình 4.9 Sơ đồ khái của quá trình tìm bãі đỗ xe còn chỗ trống gần nhất	36
Hình 4.10 Chọn điểm xuất phát và điểm đến	37
Hình 4.11 Thông tin về các tùy chọn đỗ xe	38
Hình 4.12 Tính năng bãі đỗ xe không khả dụng khi đăng nhập không thành công	39
Hình 4.13 Thông tin các bãі đỗ xe còn chỗ trống đọc đường đi	40
Hình 4.14 Chỉ đường đến bãі đỗ gần điểm đến nhất và kết thúc hành trình	41
Hình 4.15 Kết quả đếm số chỗ trống trong bãі đỗ xe khi có vật cản che khuất một phần hình ảnh	43
Hình 4.16 Hệ thống cảm biến phía trên hoặc phía dưới vị trí đỗ xe	45

DANH MỤC BẢNG BIÊU

Bảng 2.1 Các giá trị của tham số georel _____	11
Bảng 2.2 Ví dụ về truy vấn theo vị trí địa lý _____	11
Bảng 2.3 Các khái niệm quản lý danh tính trong keyrock _____	13
Bảng 3.1 Các đối tượng quan trọng của một thực thể dữ liệu _____	21
Bảng 4.1 Tốc độ xử lý dữ liệu của phân hệ thu nhận và trích rút thông tin ngữ nghĩa _____	43
Bảng 4.2 Tốc độ truy vấn thông tin ngữ nghĩa _____	44
Bảng 4.3 So sánh hệ thống đề xuất với giải pháp hiện có _____	46

Chương 1. Giới thiệu

1.1. Đặt vấn đề

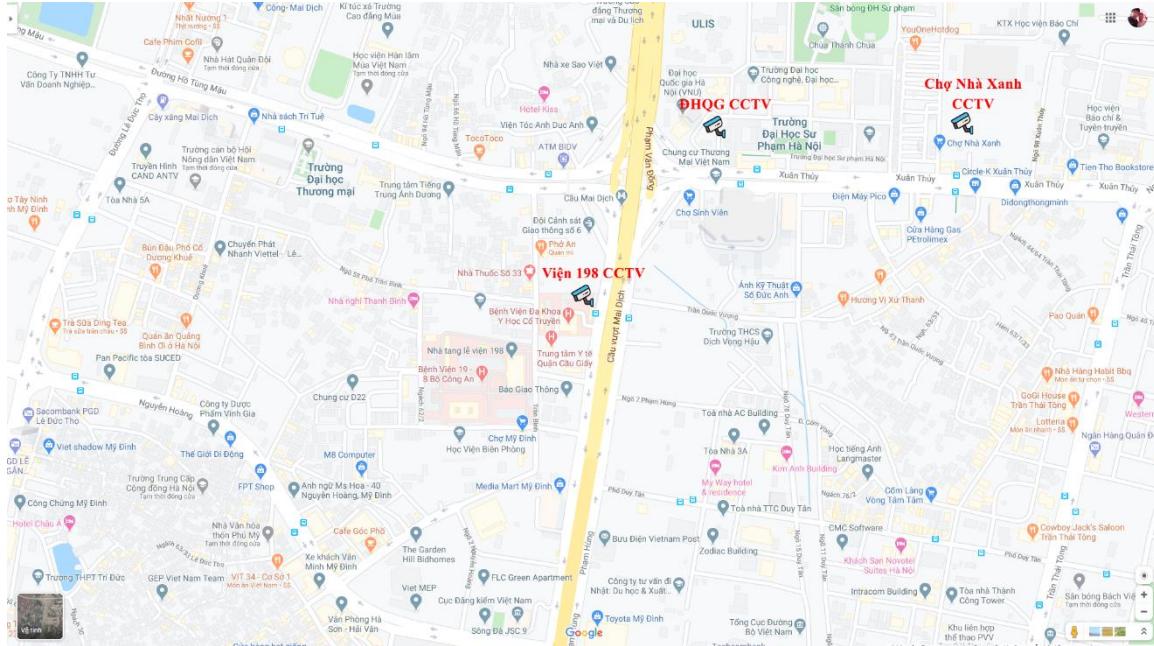
Hệ thống camera giám sát - CCTV (Closed-Circuit Television) là hệ thống sử dụng một hoặc nhiều camera để ghi lại hình ảnh theo thời gian thực tại một địa điểm cụ thể và truyền tải hình ảnh đó tới một số giới hạn các màn hình và đầu ghi hình là các NVR (Network Video Recorder) hoặc DVR (Digital Video Recorder). Trong đó, NVR sử dụng mạng internet để nhận dữ liệu từ camera còn DVR sử dụng công nghệ analog thông qua cáp đồng trục coax. Hệ thống camera giám sát sử dụng các camera IP và đầu thu NVR là hệ thống được sử dụng nhiều nhất trong các hệ thống an ninh, bảo vệ ở các thành phố lớn, đặc biệt là các khu vực công cộng nhờ các tính năng tiên tiến mà nó mang lại:

- Các camera IP và NVR được kết nối vào cùng một mạng internet để giao tiếp với nhau thông qua dây mạng hoặc wifi, đảm bảo kết nối ổn định và dễ dàng lắp đặt, mở rộng.
- Dữ liệu thu được sẽ được tiền xử lý tại camera, sau đó truyền đến đầu ghi NVR. Chính vì vậy mà việc lưu trữ dữ liệu sẽ tốn ít tài nguyên hơn.
- Tích hợp được nhiều tính năng phức tạp hơn đầu ghi analog bằng việc kết nối với trung tâm xử lý (Cloud Computing) để mở rộng việc phân tích, xử lý dữ liệu.
- Hệ thống có thể được xem và điều khiển từ xa thông qua internet, trên các nền tảng khác nhau như máy tính, điện thoại thông minh...

Tuy nhiên, những hệ thống này vẫn còn hạn chế, không thể đáp ứng được đầy đủ các nhu cầu đang ngày càng tăng của người dùng, đặc biệt là:

- Đa phần hệ thống camera được cài đặt chỉ tích hợp tính năng cơ bản là quan sát trực tiếp, lưu trữ và xem lại video khi cần. Các chức năng thông minh, tự động khai thác thông tin hình ảnh hầu như không có.
- Các tính năng thông minh dù có cũng rất sơ sài, hầu hết nhà cung cấp hệ thống chỉ tích hợp tính năng phát hiện chuyển động trong phạm vi quan sát của camera.
- Các tính năng được tích hợp hoàn toàn dựa vào nhà cung cấp hệ thống camera. Việc gửi hình ảnh lên trung tâm xử lý có nguy cơ rò rỉ dữ liệu rất lớn. Thực tế cho thấy rất nhiều video của người dùng đã bị phát tán trên các trang web không chính thống.

- Trên thực tế, các hệ thống camera được đặt ở các vị trí khác nhau, chúng được cung cấp bởi các nhà cung cấp khác nhau, điểm chung duy nhất của chúng là đều được kết nối internet (Hình 1.1 Ví dụ về các hệ thống camera ở các vị trí khác nhau). Vì vậy, để tổng hợp, sử dụng dữ liệu từ các hệ thống camera khác nhau, cần phải có một nền tảng trung gian kết nối với các hệ thống camera đó để thu nhận và quản lý dữ liệu.



Hình 1.1 Các hệ thống camera ở các vị trí khác nhau

Thực trạng này đặt ra một nhu cầu cấp thiết là thiết kế một hệ thống cho phép sử dụng hiệu quả hơn dữ liệu từ các hệ thống camera giám sát khác nhau. Hệ thống này phải đảm bảo các tiêu chí sau:

- Có thể kết nối với các hệ thống camera khác nhau, không phụ thuộc vào nhà cung cấp thiết bị.
 - Hệ thống này được coi là trung gian giữa hệ thống camera giám sát và ứng dụng. Thiết kế hệ thống phải đảm bảo tính dễ dàng kết nối (Plug and Play) ở cả phía camera và phía ứng dụng thông minh.
 - Thu nhận, lưu trữ thông tin ngữ nghĩa từ hệ thống camera một cách linh động, dễ dàng lưu trữ, tìm kiếm thông tin, tối ưu hiệu suất đọc/ghi vào cơ sở dữ liệu.
 - Hệ thống phải đảm bảo tính bảo mật thông tin và quyền riêng tư. Video thu nhận được từ camera phải tuyệt đối bảo mật. Ví dụ, việc truyền dữ liệu video lên một đám mây tập trung (Cloud) sẽ gây nguy cơ rò rỉ thông tin rất lớn, có thể gây ảnh hưởng trực tiếp tới người dùng, nên nền tảng phải có cơ chế xử lý phù hợp để giải quyết vấn đề trên.

1.2. Các nghiên cứu liên quan

Trên thế giới, hiện đã và đang có những công ty nghiên cứu và cung cấp các giải pháp thông minh tích hợp vào hệ thống camera giám sát. Công ty IC Realtime (Hoa Kỳ)[1] cung cấp các giải pháp camera an ninh có tích hợp giải pháp học sâu và tìm kiếm nội dung video dựa vào ngôn ngữ tự nhiên. Công ty Xiaomi (Trung Quốc)[2] cung cấp các thiết bị IP camera độc lập cho người dùng cá nhân, tích hợp trí tuệ nhân tạo để phát hiện chuyển động bất thường trong khu vực giám sát, và lưu trữ trực tuyến trên tài khoản đám mây của người dùng (Xiaomi Cloud). Công ty HikVision (Trung Quốc)[3] cung cấp thiết bị phần cứng bao gồm camera và các thiết bị tính toán thông minh đầu cuối (DeepinView Series Cameras and DeepinMind Series NVRs), các giải pháp tích hợp dịch vụ điện toán đám mây. Các giải pháp này có thể được ứng dụng trong thành phố thông minh, giao thông thông minh, và an ninh cộng đồng. Trung Quốc hiện đang triển khai mô hình hàng trăm triệu camera giám sát trên khắp đất nước. Đặc biệt hơn, hệ thống camera ở thành phố Guiyang đã được tích hợp công nghệ trí tuệ nhân tạo nhận diện mặt người, hệ thống này đã vượt qua bài kiểm tra của đài BBC khi cảnh sát đã tiếp cận được phóng viên của báo này, ông John Sudworth, trong vòng bảy phút.

Hệ thống camera giám sát tự động dựa trên tính toán cloud (A Cloud-Based Architecture For Smart Video Surveillance)[30] cũng được đề xuất dựa trên nền tảng FIWARE[5] có thể thu thập dữ liệu từ các camera qua môi trường mạng và xử lý tập trung các vấn đề liên quan đến an ninh và lưu trữ các sự kiện cũng như đưa ra cảnh báo sớm khi có phát hiện các hành vi bất thường liên quan đến an ninh. Đề xuất Framework xử lý dữ liệu thô cho IoT (Raw Data Processing Framework for IoT)[4] đã nêu ra một vấn đề là để phát triển các hệ thống IoT, việc lưu trữ và quản lý luồng thông tin liên tục được sinh ra từ các cảm biến, camera là một thách thức lớn. Các ứng dụng IoT yêu cầu thông tin phải được xử lý theo thời gian thực, và để bắt kịp thông tin nào có thể truy vấn được, nó phải được lưu trữ ở một cơ sở dữ liệu. Việc thêm vào cơ sở dữ liệu một lượng dữ liệu lớn và liên tục chính là nút nghẽn cổ chai đầu tiên của hệ thống IoT. Nếu ta có thể lưu trữ và quản lý dữ liệu thô (Raw data) thì sẽ là một cải tiến rất lớn cho vấn đề này. Vì vậy, nghiên cứu [4] đề xuất một Framework cho phép áp dụng công nghệ xử lý dữ liệu thô vào việc quản lý luồng dữ liệu từ các cảm biến IoT thay vì sử dụng dữ liệu SQL truyền thống để giải quyết bài toán trên. Hệ thống quản lý dữ liệu thô này gồm 4 thành phần: *In-situ Query Engine + DBMS, In-memory storage, Raw data files, và Streaming Data & Query Management. In-situ Query Engine + DBMS* nhằm mục đích tạo ra câu lệnh truy vấn

SQL truyền thống, nên bất kỳ hệ thống in-situ query engine có sẵn nào cũng có thể áp dụng cho phần này. Việc hiểu và xử lý câu lệnh truy vấn để trả về kết quả sẽ được thực hiện ở bộ phận khác. *In-memory storage* là một vùng lưu trữ có kích thước nhỏ, dùng để lưu trữ dữ liệu hiện tại hoặc những dữ liệu được sử dụng nhiều, hoặc có thể dùng làm bộ nhớ đệm. *Raw data files* chính là dữ liệu thô sinh ra từ các cảm biến IoT. Nó thường được lưu trữ dưới các định dạng quen thuộc như CSV, JSON hoặc XML. Cuối cùng, *Streaming Data & Query Management*, thực hiện việc xử lý luồng dữ liệu và chuẩn hóa, lưu trữ nó dưới dạng Raw, đồng thời xử lý câu lệnh truy vấn từ query engine và trả về kết quả. Tác giả đánh giá hệ thống này (PgRAW) và phương pháp sử dụng SQL truyền thống (PgSQL) trên các bộ dữ liệu với kích thước khác nhau, kết quả cho thấy nó cải thiện hiệu suất 99.5% cho việc thêm dữ liệu và 57% cho việc truy vấn dữ liệu so với PgSQL. Tuy hệ thống này mới chỉ giải quyết được một trong các vấn đề đặt ra ở mục 1.1, nhưng đề xuất sử dụng NoSQL cho lưu trữ, quản lý dữ liệu có ý nghĩa rất lớn và nên được áp dụng.

Ở Việt Nam, các hệ thống camera giám sát được lắp đặt ở nhiều khu vực khác nhau như bãi đỗ xe, siêu thị, ngã tư... Hầu hết các hệ thống này chỉ hỗ trợ lưu trữ video và cho phép xem trực tiếp hoặc xem lại, một phần nhỏ được tích hợp tính năng cảnh báo chuyển động bất thường – được cung cấp bởi hãng sản xuất thiết bị camera. Ngoài ra, chưa có bất kì một nền tảng thứ ba nào có khả năng kết nối với tất cả các hệ thống camera để khai thác thông tin ngữ nghĩa của chúng, nên không thể phát triển các ứng dụng xã hội thông minh, thành phố thông minh dựa trên sự kết hợp của mạng lưới rộng lớn các hệ thống camera.

1.3. Những đóng góp của luận văn

Luận văn này tập trung tìm hiểu và xây dựng một hệ thống cho phép trích rút, lưu trữ, truy xuất thông tin ngữ nghĩa từ các hệ thống camera giám sát đáp ứng đầy đủ các tiêu chí nêu ở mục 1.1, với tên gọi là hệ thống CamNet (viết tắt của Camera Networks, các phần tiếp theo của luận văn sẽ dùng tên gọi “hệ thống CamNet” để chỉ hệ thống được đề xuất). Hệ thống CamNet được xây dựng dựa trên các thư viện của nền tảng mã nguồn mở FIWARE[5], nhờ đó, thông tin từ các hệ thống camera giám sát có thể được cung cấp như một dịch vụ, các ứng dụng thông minh đa nền tảng có thể sử dụng tài khoản với các quyền phù hợp để tìm kiếm thông tin trong hệ thống cho các mục đích sử dụng khác nhau. Ngoài ra, luận văn còn phát triển một ứng dụng tích hợp hệ thống CamNet vào bản đồ HERE Maps cho Android để chỉ đường cho tài xế đến bãi đỗ xe còn trống gần nhất, nhằm đưa ra một hướng giải quyết cho một

trong những vấn đề cấp thiết của xã hội, và chứng minh khả năng ứng dụng thực tiễn của hệ thống.

1.4. Cấu trúc luận văn

Phần tiếp theo của luận văn được chia thành 3 chương như dưới đây:

Chương 2 Cung cấp kiến thức cơ bản về nền tảng FIWARE được sử dụng để xây dựng hệ thống CamNet.

Chương 3 Diễn giải thiết kế và cài đặt hệ thống.

Chương 4 Phát triển ứng dụng tìm bãi đỗ xe và đánh giá kết quả.

Cuối cùng là kết luận và hướng phát triển trong tương lai của hệ thống.

Chương 2. Tổng quan về nền tảng FIWARE

FIWARE[5] là một dự án nền tảng mã nguồn mở được đầu tư bởi liên minh châu Âu (EU). Mục tiêu của nó là tạo ra một hệ sinh thái chung tiêu chuẩn, miễn phí để hướng tới triển khai, phát triển các ứng dụng thông minh trong nhiều lĩnh vực khác nhau như thành phố thông minh, nông nghiệp thông minh, công nghiệp thông minh... Nền tảng FIWARE cung cấp những thành phần (được gọi là các GE – Generic Enablers) có thể tích hợp hoặc kết hợp với các nền tảng thứ ba để phát triển các giải pháp thông minh. Các GEs này được phát triển để giải quyết các vấn đề của một hệ thống IoT:

- Giao tiếp với các hệ thống IoT, Robots và các hệ thống bên thứ 3, để nắm bắt các cập nhật của thông tin ngữ nghĩa và chuyển đổi thông tin thành các hành động cần thiết.
- Quản lý, xuất bản (publication) và kiếm tiền (monetization), hỗ trợ việc theo dõi, quản lý việc sử dụng ứng dụng và việc xuất bản, kiếm tiền từ dữ liệu ngữ nghĩa được quản lý.
- Xử lý, phân tích và mô hình hóa thông tin ngữ nghĩa để xây dựng các ứng dụng thông minh hoặc hỗ trợ người dùng bằng cách đưa ra các quyết định thông minh.

Sử dụng nền tảng FIWARE không có nghĩa là ta phải sử dụng tất cả các GEs của nó vào hệ thống. Hệ thống CamNet áp dụng ba GEs của FIWARE, đó là: Orion Context Broker để quản lý thông tin ngữ nghĩa, Keyrock và Wilma PEP Proxy để xác thực và quản lý quyền của người dùng. Các phần tiếp theo của chương này sẽ đưa ra cái nhìn tổng quan về các GEs nêu trên.

2.1. Orion Context Broker

Orion Context Broker là một triển khai C++ của NGSIV2 REST API, để quản lý thông tin ngữ nghĩa và tính khả dụng của nó. Nó cho phép ta quản lý toàn bộ vòng đời của thông tin ngữ nghĩa. Cụ thể là:

- Truy vấn thông tin ngữ nghĩa: Orion Context Broker lưu trữ thông tin ngữ nghĩa được cập nhật từ các ứng dụng, do đó các truy vấn được giải quyết dựa trên các thông tin đó. Thông tin ngữ nghĩa bao gồm các thực thể (ví dụ: một chiếc camera giám sát) và các thuộc tính của chúng (ví dụ: vị trí của camera, số lượng người trong ảnh mà camera ghi nhận được).
- Cập nhật thông tin ngữ nghĩa: Ví dụ như gửi thông tin cập nhật về số lượng người trong khu vực.

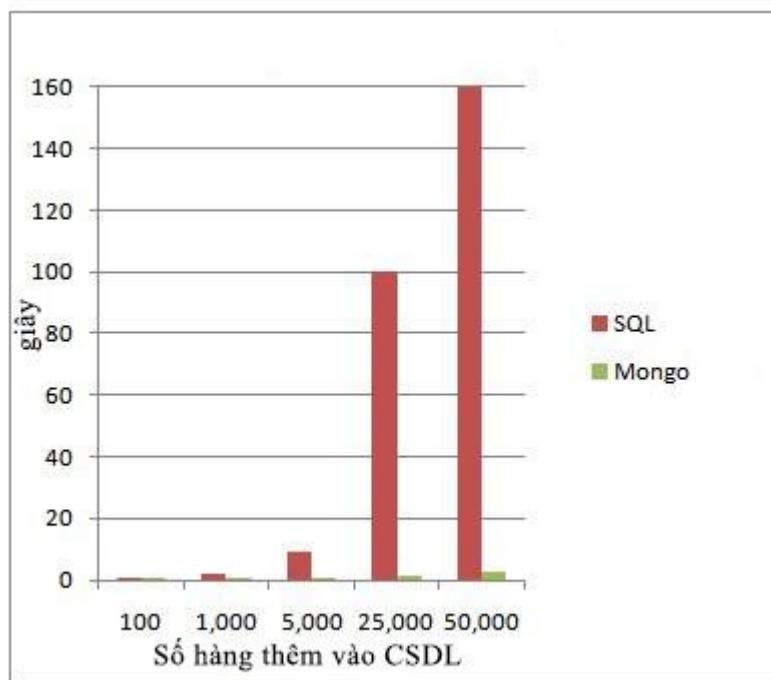
- Nhận thông báo khi có thay đổi về thông tin ngữ nghĩa: Ví dụ như khi số lượng người trong khu vực đã thay đổi.
- Thêm thông tin ngữ nghĩa của các nhà cung cấp mới: Ví dụ như thêm dữ liệu từ hệ thống camera của tòa nhà Keangnam vào hệ thống hiện có.

Orion Context Broker sử dụng cơ sở dữ liệu MongoDB và hoạt động dựa trên NGSIV2 API. Các thành phần này và cách thức hoạt động của nó sẽ được giới thiệu cụ thể hơn dưới đây.

2.1.1. MongoDB Database

MongoDB là một chương trình cơ sở dữ liệu mã nguồn mở được thiết kế theo kiểu hướng đối tượng trong đó các bảng được cấu trúc một cách linh hoạt cho phép các dữ liệu lưu trên bảng không cần phải tuân theo một dạng cấu trúc nhất định nào. Chính do cấu trúc linh hoạt này nên MongoDB có thể được dùng để lưu trữ các dữ liệu có cấu trúc phức tạp và đa dạng và không cố định. Những ưu điểm của MongoDB có thể được tóm lược như dưới đây:

- Do MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên nó rất linh hoạt trong việc lưu trữ dữ liệu, có thể thêm bất cứ gì vào cơ sở dữ liệu.
- Dữ liệu trong MongoDB không có sự ràng buộc lẫn nhau, không có *join* như trong các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) nên khi thêm, xóa hay sửa nó không cần phải mất thời gian kiểm tra xem có thỏa mãn các ràng buộc dữ liệu như trong RDBMS.
- MongoDB rất dễ mở rộng (Horizontal Scalability). Trong MongoDB có một khái niệm *cluster* là cụm các *node* chứa dữ liệu giao tiếp với nhau, khi muốn mở rộng hệ thống ta chỉ cần thêm một *node* vào *cluster*.
- Trường dữ liệu “*id*” luôn được tự động đánh *index* (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.
- Hiệu năng cao: tốc độ truy vấn (find, update, insert, delete) của MongoDB nhanh hơn hẳn so với các hệ quản trị cơ sở dữ liệu quan hệ. Với một lượng dữ liệu đủ lớn thì thử nghiệm cho thấy tốc độ insert của MongoDB có thể nhanh tới gấp 100 lần so với MySQL (Hình 2.1 So sánh tốc độ truy vấn củaMongo và SQL).

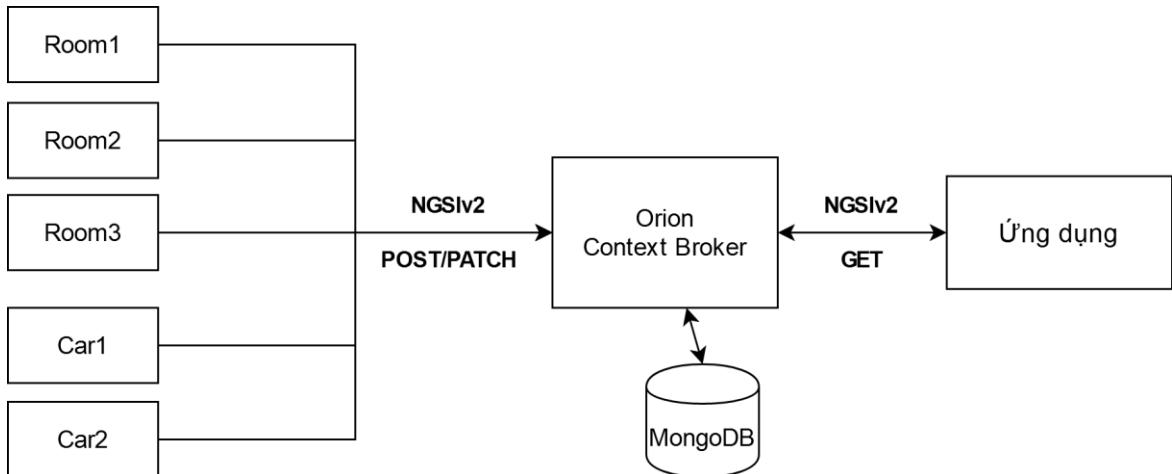


Hình 2.1 So sánh tốc độ truy vấn của MongoDB và SQL

Các tính chất trên của MongoDB rất phù hợp với việc lưu trữ, quản lý dữ liệu sinh ra từ các thiết bị IoT (là các dữ liệu phi cấu trúc), nên nó được sử dụng làm cơ sở dữ liệu cho Orion Context Broker.

2.1.2. NGSI-v2 API và cách thức hoạt động

FIWARE NGSI là viết tắt của Next Generation Service Interface, và NGSIv2 thể hiện đây là phiên bản thứ 2 (version 2). Để hiểu rõ hơn về NGSIv2 API và cách thức hoạt động của nó, hãy giả định ta có một tòa nhà với những phòng khác nhau và ta muốn sử dụng Orion Context Broker để quản lý thông tin ngữ nghĩa của chúng. Các phòng là Room1, Room2, Room3 và mỗi phòng có hai cảm biến: nhiệt độ và độ ẩm không khí. Thêm nữa, hãy giả định rằng ta có hai chiếc xe hơi (Car1 và Car2) với những cảm biến cho biết thông tin về tốc độ và vị trí. Khi đó, Orion Context Broker sẽ tương tác với cả nguồn cung cấp dữ liệu (là các cảm biến) và nguồn sử dụng dữ liệu (là các ứng dụng thông minh xử lý dữ liệu ngữ nghĩa để mô hình hóa hay đưa ra quyết định). Toàn bộ tương tác này đều thông qua NGSIv2 API (Hình 2.2 Mô hình hoạt động của Orion Context Broker).



Hình 2.2 Mô hình hoạt động của Orion Context Broker

Để làm được điều đó, FIWARE NGSI API định nghĩa được: *Mô hình dữ liệu* (data model) cho dữ liệu ngữ nghĩa, dựa trên một mô hình thông tin đơn giản sử dụng khái niệm thực thể ngữ nghĩa (context entities) và *Lớp giao tiếp dữ liệu ngữ nghĩa* (context data interface) để trao đổi thông tin bằng các hoạt động truy vấn (query), đăng ký (subscription) và cập nhật (update). Một thực thể của *mô hình dữ liệu* được thể hiện bởi một đối tượng JSON với các cú pháp sau:

- Định danh của thực thể (id) được chỉ định bởi thuộc tính *id* của đối tượng, có giá trị là một chuỗi (String).
- Loại thực thể (type) được chỉ định bởi thuộc tính *type* của đối tượng, có giá trị là một chuỗi (String).
- Các thuộc tính của thực thể (entity attributes) được chỉ định bởi các thuộc tính bổ sung, gồm có tên thuộc tính và giá trị của nó. Hiện nhiên, “*id*” và “*type*” không được sử dụng để làm tên của thuộc tính.

Dưới đây là một ví dụ của một thực thể mô hình dữ liệu:

```
{
  "id": "entityID",
  "type": "entityType",
  "attr_1": <val_1>,
  "attr_2": <val_2>,
  ...
  "attr_N": <val_N>
}
```

Giả sử phòng 1 có nhiệt độ phòng là 23 độ C, thì thực thể này được thể hiện như sau:

```
{
  "id": "Room1",
  "type": "Room",
  "temperature": 23
}
```

Lớp giao tiếp ngữ nghĩa cung cấp các API cho phép trao đổi thông tin giữa nhà cung cấp dữ liệu (Provider) với máy chủ và máy chủ với các ứng dụng sử dụng dữ liệu (Consumer). Đứng về phía nhà cung cấp dữ liệu ngữ nghĩa (là các cảm biến của phòng và xe hơi trong giả định trên), ta có các hoạt động sau:

Append: *POST /v2/entities* (nếu chưa tồn tại thực thể) hoặc *POST /v2/entities/<id>/attrs* (nếu thực thể đó đã tồn tại)

Update: *PATCH /v2/entities/<id>/attrs* để cập nhật giá trị của thuộc tính

Delete: *DELETE /v2/entities/<id>/attrs/<attrName>* để xóa các thuộc tính trong một thực thể hoặc *DELETE /v2/entities/<id>* để xóa cả thực thể

Replace: *PUT /v2/entities/<id>/attrs* để thay thế một thuộc tính

Nếu hoạt động được thực hiện thành công, ta sẽ nhận lại mã *200 OK*, còn nếu thất bại, ta sẽ nhận lại mã lỗi (mã khác *2-xx*) và nội dung lỗi.

Đứng về phía các ứng dụng truy vấn dữ liệu (query), NGSIV2 cung cấp các phương thức truy vấn theo *id*, *type*, hoặc theo các thuộc tính khác nhau, đặc biệt là khả năng truy vấn theo vị trí và đăng ký nhận thông tin khi dữ liệu thay đổi. Truy vấn theo vị trí địa lý được chỉ định bằng cách sử dụng các tham số sau:

- **georel:** được thiết kế để xác định mối quan hệ không gian giữa thực thể và một hình tham chiếu (*geometry*). Các giá trị của nó thể hiện ở bảng 2.2:

Giá trị	Ý nghĩa
near	Quan hệ <i>near(gần)</i> trả về các thực thể phải đặt ở một ngưỡng khoảng cách xác định so với mốc tham chiếu, bao gồm: <ul style="list-style-type: none"> • maxDistance: tính bằng mét, khoảng cách tối đa của các thực thể so với tham chiếu.

	<ul style="list-style-type: none"> • minDistance: tính bằng mét, khoảng cách tối đa của các thực thể so với tham chiếu.
coveredBy	Trả về các thực thể nằm trong hình học tham chiếu. Viền của hình được coi là một phần của hình.
intersects	Trả về các thực thể giao nhau với hình tham chiếu.
equals	Trả về thực thể có vị trí giống với vị trí tham chiếu.
disjoint	Trả về các thực thể không giao nhau với hình tham chiếu.

Bảng 2.1 Các giá trị của tham số georel

- *geometry*: cho phép định nghĩa hình tham chiếu dùng để truy vấn. Các hình tham chiếu có thể là *point*-một điểm trên mặt đất, *line*-một đường thẳng, *polygon*-một hình đa giác, hoặc *box*-một hình hộp giới hạn.
- *coords*: phải là một chuỗi chứa danh sách các cặp tọa độ địa lý được phân cách với nhau bằng dấu “;” theo quy định chuẩn “*Simple Location Format*”.

Tương ứng với giả định trên đầu mục, cách áp dụng truy vấn theo vị trí vào các tình huống thực tế được thể hiện ở một số ví dụ ở bảng 2.3 dưới đây:

Cú pháp truy vấn	Ý nghĩa
georel=near;maxDistance:1000&geometry=point&coords=-40.4,-3.5	coords thể hiện vị trí của xe hơi Carl, tìm kiếm các thực thể ở cách xe 1000m
georel=coveredBy&geometry=polygon &coords=25.774,-80.190;18.466,-66.118;32.321,-64.757;25.774,-80.190	coords thể hiện tọa độ hình học của ngôi nhà, tìm kiếm các thực thể có tọa độ ở trong ngôi nhà đó.

Bảng 2.2 Ví dụ về truy vấn theo vị trí địa lý

Ngoài ra, ta có thể đăng ký nhận thông tin theo *id* của thực thể. NGSIV2 cung cấp phương thức để tạo mới, xóa, và truy vấn tất cả các đăng ký đã tạo. Các thông tin đăng ký này được quản lý ở địa chỉ *<orion server>/v2/subscriptions/*. Ví dụ, khi một ứng dụng đăng ký thông tin vị trí của xe hơi Carl, thì khi thông tin này được cập nhật mới trên Orion Context Broker, nó sẽ gửi một thông báo đến ứng dụng đó. Như

vậy, ứng dụng có thể biết vị trí của chiếc xe theo thời gian thực, mà không cần phải gửi truy vấn liên tục đến Orion Context Broker.

2.2. Keyrock Identity Manager

Việc quản lý dữ liệu ngữ nghĩa đã được xử lý bởi Orion Context Broker, nhưng trung tâm dữ liệu này chưa được bảo mật. Tất cả mọi người dùng đều có thể truy vấn, sửa đổi thông tin ngữ nghĩa của nó. Vì vậy, một GE khác của FIWARE sinh ra làm trung gian giữa người dùng và Orion Context Broker để xử lý vấn đề quản lý quyền và xác thực người sử dụng. Đó chính là Keyrock Identity Manager. Tương tự như một tài khoản Google, một tài khoản Keyrock cung cấp cho ta định danh và quyền để có thể sử dụng tất cả các FIWARE GEs, bao gồm Orion Context Broker (Hình 2.3 Hệ sinh thái Google và FIWARE).



Hình 2.3 Hệ sinh thái Google và FIWARE

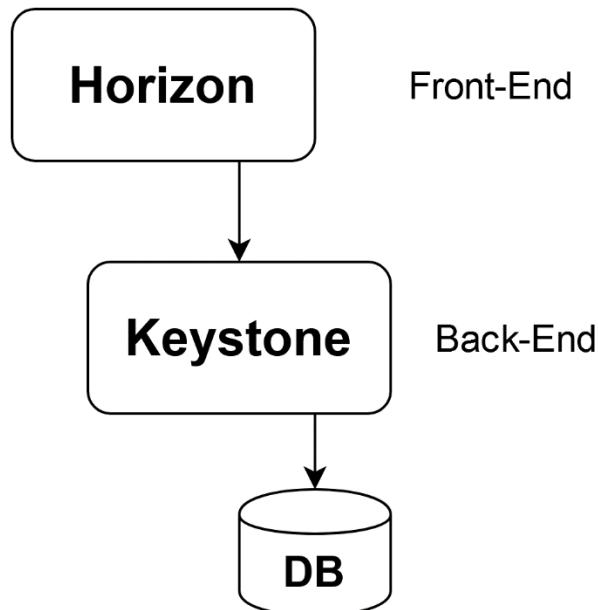
Các khái niệm về quản lý danh tính của Keyrock được thể hiện ở bảng 2.4.

Đối tượng	Mô tả
Người dùng (Users)	Có tài khoản đã đăng ký Keyrock. Có thể quản lý các tổ chức và đăng ký ứng dụng.
Tổ chức (Organizations)	Là nhóm người dùng chia sẻ tài nguyên của một ứng dụng (các quyền và vai trò). Người dùng có thể là thành viên hoặc chủ sở hữu (quản lý tổ chức).

Ứng dụng (Applications)	Có vai trò là khách hàng trong kiến trúc OAuth 2.0 và sẽ yêu cầu dữ liệu người dùng đã được bảo vệ. Có thể xác thực người dùng dựa vào thông tin xác thực của họ. Định nghĩa các quyền và vai trò để quản lý người dùng và tổ chức.
----------------------------	---

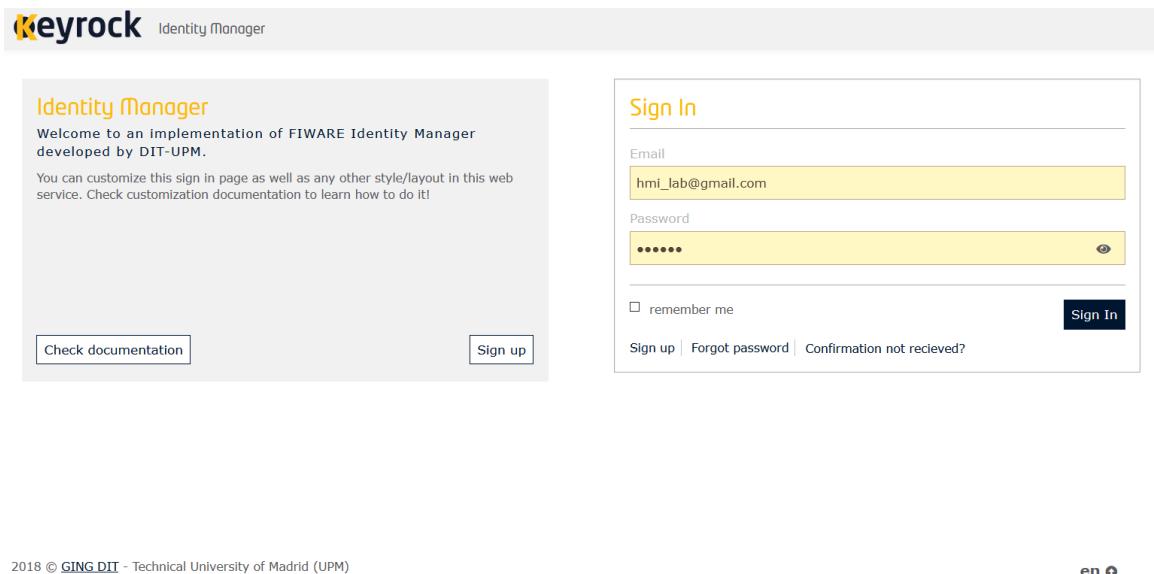
Bảng 2.3 Các khái niệm quản lý danh tính trong Keyrock

Keyrock cung cấp cả giao diện người dùng (GUI) và API. Kiến trúc của nó gồm có Front-End sử dụng *Horizon*[6] và Back-End sử dụng *Keystone*[7], nó sử dụng cơ sở dữ liệu MySQL. Khác với các yêu cầu của dữ liệu ngữ nghĩa, dữ liệu định danh và quyền người dùng có rất nhiều ràng buộc chặt chẽ, nên phù hợp hơn với cơ sở dữ liệu MySQL truyền thống (Hình 2.4 Kiến trúc của Keyrock).



Hình 2.4 Kiến trúc của Keyrock

Horizon là một dự án dựa trên nền tảng Django[8] nhằm cung cấp một bảng điều khiển trên nền tảng web cho các dịch vụ của OpenStack bao gồm Nova, Swift... Ở đây, FIWARE đã tùy biến lại *Horizon* cho phù hợp với những yêu cầu của hệ thống Keyrock. Nó bao hàm các công nghệ bảo mật Oauth2 Driver, reCAPTCHA, AuthZForce Driver và các giao diện người dùng cho tài khoản người dùng thường và tài khoản Admin (Hình 2.5 Giao diện đăng nhập của Keyrock).



Hình 2.5 Giao diện đăng nhập của Keyrock

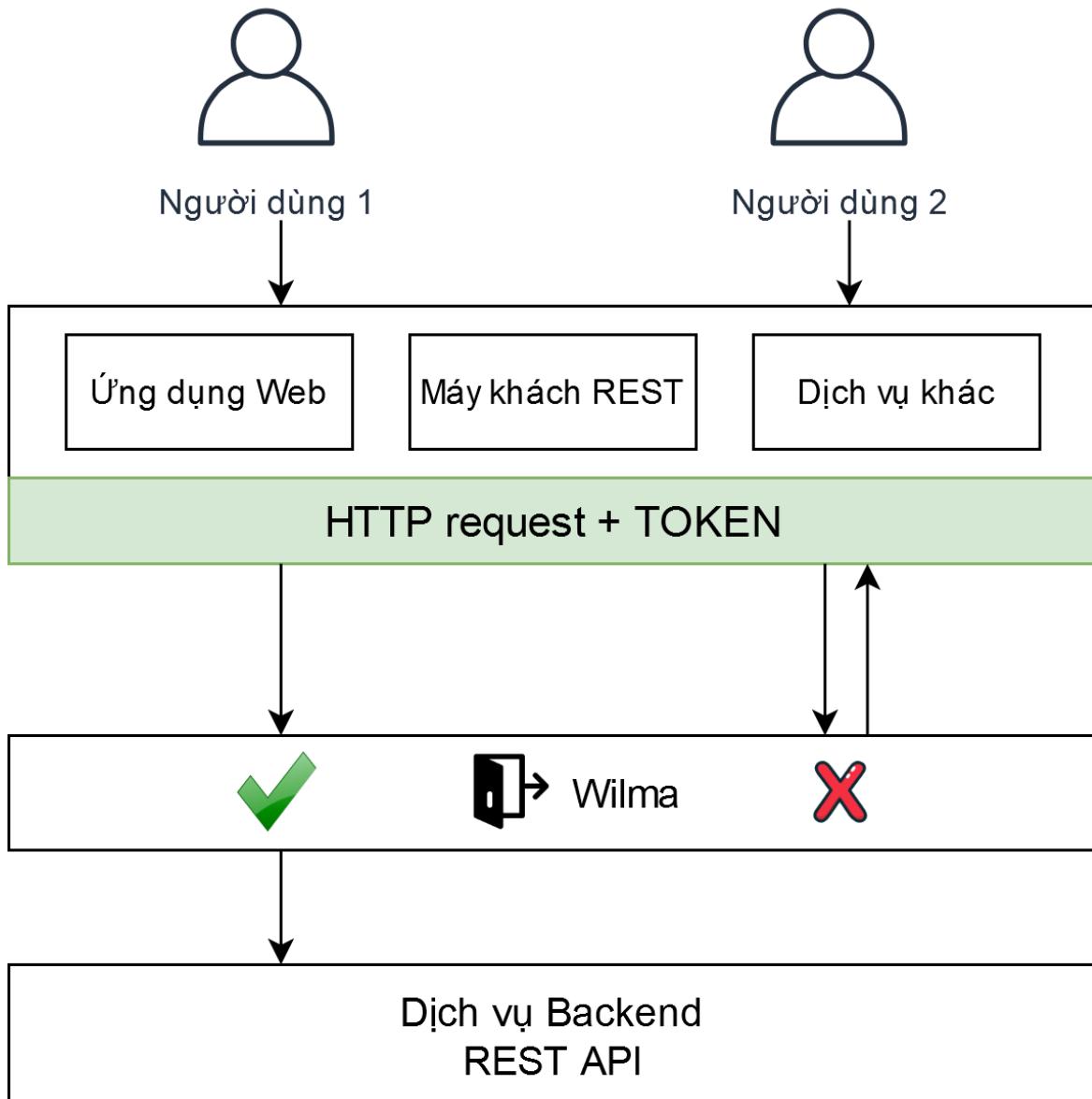
Keystone cũng được tùy biến lại từ dự án *Keystone* của OpenStack. Nó là một dịch vụ (service) cung cấp các cơ chế xác thực (authentication), ủy quyền (authorization) và khám phá dịch vụ (service discovery). Nó kết nối với một cơ sở dữ liệu MySQL và bao hàm các tiện ích bảo mật mở rộng là OAuth2, SCIM 2.0, bảo mật 2 lớp (2-factor authentication).

Kết hợp *Horizon* và *Keystone* lại, ta có một hệ thống quản lý định danh và quyền người dùng trực quan, thân thiện. Hầu hết mọi thao tác từ người quản trị hệ thống (admin) đến người dùng cuối (end-user) đều có thể thực hiện trên GUI, như tạo tài khoản, đăng ký ứng dụng cần bảo vệ, định nghĩa quyền và cấp quyền cho người dùng cuối...

2.3. Wilma PEP Proxy

Dựa vào Keyrock, ta đã có thể cho phép hoặc từ chối quyền truy cập vào tài nguyên hệ thống bằng việc xác thực tài khoản. Tuy nhiên, sau khi vượt qua quá trình xác thực tài khoản, người dùng có thể thực hiện truy vấn trực tiếp lên các dịch vụ được bảo vệ, việc này tiềm ẩn nhiều nguy cơ về bảo mật. FIWARE Wilma sinh ra để giải quyết vấn đề này. Nó là một Policy Enforcement Point (PEP) công khai đặt trước một tài nguyên được bảo mật (như là Orion Context Broker). Wilma hoạt động như một người gác cổng, để thực hiện một truy vấn đến hệ thống, người dùng hoặc các tác nhân khác chuyển yêu cầu HTTP (HTTP request) và thông tin xác thực (chứa trong token) được lấy từ Keyrock cho Wilma PEP proxy, Wilma sẽ xác thực và chuyển tiếp yêu cầu xuống dịch vụ được bảo vệ. Sau đó, Wilma nhận lại phản hồi cho yêu cầu và chuyển phản hồi đó tới người dùng. Đối với người dùng đã được xác

nhận, độ trễ so với truy cập trực tiếp là rất nhỏ và kết quả nhận được là hoàn toàn tương đồng với việc truy cập trực tiếp. Với người dùng trái phép, đơn giản là họ sẽ nhận được một phản hồi lỗi *401-Unauthorized response*. Hình 2.6 thể hiện cách thức hoạt động của Wilma PEP Proxy. Với cách thức hoạt động như vậy, mọi truy vấn đến các dịch vụ của FIWARE được ẩn đi, các tác nhân tương tác với hệ thống hoàn toàn không biết dữ liệu thực sự đang nằm ở đâu. Kết hợp Wilma PEP proxy với Keyrock, ta sẽ có một hệ thống xác thực và phân quyền người dùng hoàn chỉnh.



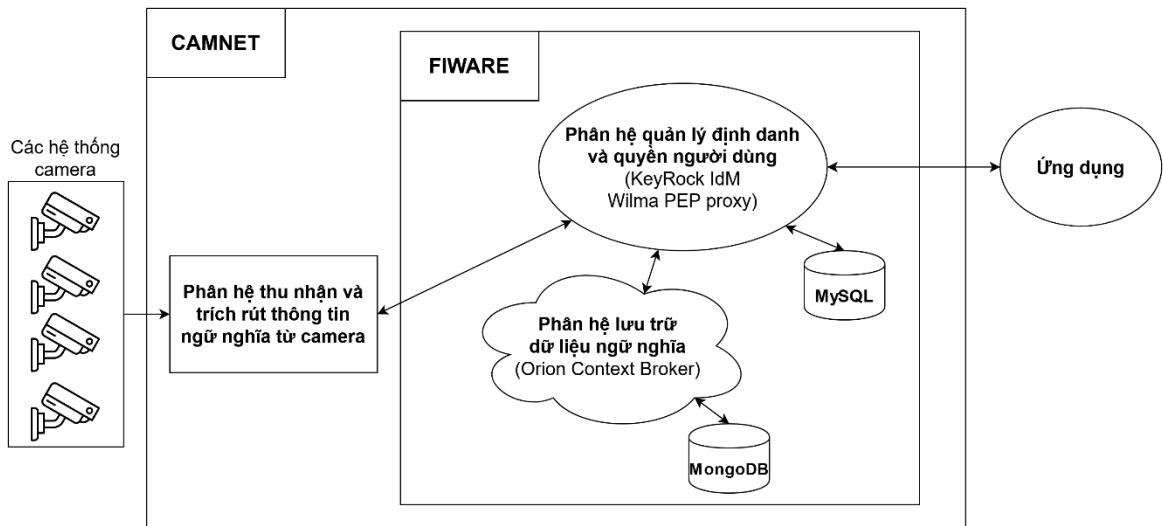
Hình 2.6 Cách thức hoạt động của Wilma PEP Proxy

Chương 3. Thiết kế và cài đặt hệ thống

Để đảm bảo hệ thống CamNet đạt được các yêu cầu như đã đề cập ở chương 1, tác giả đề xuất một thiết kế gồm 3 phân hệ: phân hệ thu nhận và trích rút thông tin ngữ nghĩa từ camera giám sát, phân hệ lưu trữ, quản lý thông tin ngữ nghĩa, và phân hệ quản lý định danh và quyền người dùng. Mô tả cụ thể của toàn bộ hệ thống và từng phân hệ được thể hiện ở các mục dưới đây.

3.1. Kiến trúc tổng quan hệ thống

Hệ thống CamNet là sự kết hợp giữa điện toán cạnh và điện toán đám mây. Trung tâm lưu trữ, quản lý và bảo mật dữ liệu phải được thiết kế để trở thành một nhà cung cấp dịch vụ (services), nên nó phải được xây dựng hoàn toàn theo mô hình điện toán đám mây, các thư viện của FIWARE được sử dụng để xây dựng nền các thành phần này. Hiện tại, có một số nền tảng cung cấp khả năng quản lý thông tin từ thiết bị IoT như Confluent[31] hay Google Cloud IoT Core[32], tuy nhiên chúng là nền tảng có tính phí và không thể can thiệp sâu vào hệ thống của chúng. Với việc sử dụng các thư viện của FIWARE, chi phí xây dựng hệ thống là miễn phí, ngoài ra, ta có thể can thiệp sâu vào cài đặt của thư viện để tùy chỉnh theo các mục đích sử dụng riêng. Khác với quản lý dữ liệu, việc thu nhận và trích rút thông tin từ camera giám sát đứng trước thách thức rất lớn về hiệu năng xử lý, do độ phức tạp của các thuật toán trích rút thông tin và khi mở rộng cài đặt hệ thống CamNet trên nhiều hệ thống camera khác nhau, số lượng camera cần phải xử lý sẽ tăng lên rất nhiều. Vì vậy, việc thu nhận và trích rút thông tin ngữ nghĩa từ hệ thống camera được xử lý trên các máy chủ cạnh đặt bên ngoài hệ thống điện toán đám mây. Hình 3.1 mô tả chi tiết thiết kế của hệ thống, mỗi khối có một vai trò duy nhất tương ứng với từng phân hệ. Trong đó, *phân hệ lưu trữ dữ liệu ngữ nghĩa* và *phân hệ quản lý định danh và quyền người dùng* sử dụng các thư viện có sẵn trong nền tảng FIWARE, các thành phần của *phân hệ thu nhận và trích rút thông tin ngữ nghĩa từ camera* và một ứng dụng trên nền tảng Android kết nối với hệ thống CamNet (được trình bày ở chương 4) do tác giả tự phát triển.



Hình 3.1 Thiết kế chi tiết của hệ thống

Thông tin ngữ nghĩa ở đây được định nghĩa là các thông tin chứa trong hình ảnh mà có thể được trích xuất dựa vào các thuật toán phân tích hình ảnh (Semantic Image Analysis). Nhiệm vụ đầu tiên của *phân hệ thu nhận và trích rút thông tin ngữ nghĩa từ camera* là thu nhận dữ liệu hình ảnh từ camera giám sát. Hình ảnh này sẽ được xử lý bởi các thuật toán phân tích ngữ nghĩa từ hình ảnh và cho kết quả là dữ liệu ngữ nghĩa của ảnh, được lưu trữ dạng thô (Raw) trên cơ sở dữ liệu NoSQL và nội dung được thiết kế phù hợp. Các thuật toán phân tích ngữ nghĩa từ ảnh rất đa dạng, tùy thuộc vào nhu cầu của người dùng như phát hiện đám đông bất thường, phát hiện đoạn đường ngập lụt... Phân hệ này đảm bảo chỉ truyền đi thông tin ngữ nghĩa trong hình ảnh chứ không truyền đi hình ảnh đó. Thứ 2 là *phân hệ lưu trữ dữ liệu ngữ nghĩa*, nó sử dụng thư viện FIWARE Orion Context Broker, được thiết kế để đảm bảo tối ưu hiệu năng đọc/ghi dữ liệu dạng thô. Hơn nữa, các phương thức truy vấn dữ liệu cũng được thiết kế linh hoạt, thuận tiện cho việc tìm kiếm thông tin theo không gian, thời gian. Thứ 3, *phân hệ quản lý định danh và quyền người dùng* kết hợp 2 thư viện của FIWARE là Keyrock Identity Manager và Wilma PEP Proxy, nhằm đảm bảo kết nối an toàn với phân hệ lưu trữ dữ liệu ngữ nghĩa. Mỗi đối tượng muốn truy vấn đến đám mây ngữ nghĩa (như phân hệ xử lý dữ liệu đầu vào muốn đẩy thông tin lên, hoặc các ứng dụng muốn tìm kiếm thông tin trong hệ thống) đều phải được xác thực định danh và quyền hạn (ví dụ như quyền đọc/ghi toàn bộ hay một phần dữ liệu). Ngoài ra, *ứng dụng* kết nối với hệ thống là các ứng dụng bất kỳ trên các nền tảng khác nhau. Chúng có thể sử dụng tài khoản định danh với quyền được cấp để tìm kiếm các thông tin có ích trong hệ thống để giải quyết các bài toán khác nhau như cảnh báo tắc đường, ngập lụt, cảnh báo khu vực hỗn loạn. Luận văn

này phát triển một ứng dụng bản đồ được kết nối với hệ thống để đưa ra tính năng tìm đến bãi đỗ xe gần điểm đến nhất cho người dùng.

3.2. Phân hệ thu nhận và trích rút thông tin ngữ nghĩa từ camera

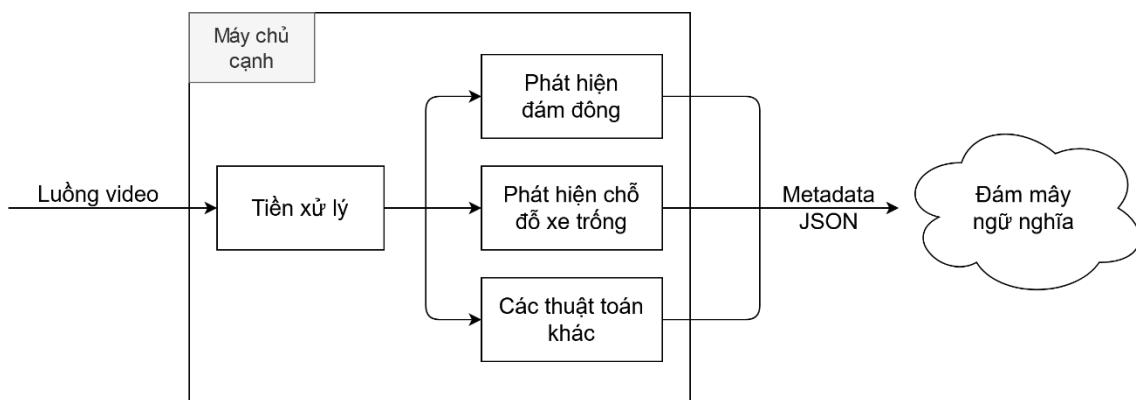
Điện toán đám mây (cloud computing) [9] là mô hình điện toán sử dụng các công nghệ máy tính và phát triển dựa vào mạng Internet. Ở mô hình điện toán này, mọi khả năng liên quan đến công nghệ thông tin đều được cung cấp dưới dạng các “dịch vụ”, cho phép người sử dụng truy cập các dịch vụ công nghệ từ một nhà cung cấp nào đó “trong đám mây” mà không cần phải có các kiến thức, kinh nghiệm về công nghệ đó, cũng như không cần quan tâm đến các cơ sở hạ tầng phục vụ công nghệ đó. Tuy nhiên, với sự phát triển nhanh chóng của các ứng dụng Internet di động và Internet of Things (IoT) (đặc biệt là các thiết bị IoT cảm biến hình ảnh), kiến trúc điện toán đám mây tập trung hiện có đang gặp phải những thách thức nghiêm trọng khi các số lượng lớn các thiết bị này cùng kết nối và truyền tải một lượng lớn dữ liệu tới máy chủ tập trung để yêu cầu dịch vụ. Các thách thức này bao gồm:

- Độ trễ: Các ứng dụng IoT có yêu cầu thời gian thực cao. Trong mô hình điện toán đám mây tập trung truyền thống, các ứng dụng gửi dữ liệu đến trung tâm dữ liệu và nhận được phản hồi, điều này làm tăng độ trễ của hệ thống.
- Băng thông: Truyền một lượng lớn dữ liệu được tạo bởi các thiết bị IoT (ví dụ các thiết bị thu nhận hình ảnh/video) lên đám mây theo thời gian thực sẽ gây ra áp lực lớn đối với băng thông mạng.
- Tính khả dụng: Khi ngày càng có nhiều dịch vụ được triển khai trên điện toán đám mây, tính khả dụng của các dịch vụ này đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày. Người sử dụng sẽ rất phụ thuộc vào sự sẵn sàng của các dịch vụ này và gây thách thức cho hệ thống điện toán đám mây tập trung cung cấp dịch vụ 24/7.
- Năng lượng: Trung tâm dữ liệu tiêu thụ rất nhiều năng lượng.
- Bảo mật và quyền riêng tư: Dữ liệu của hàng ngàn hộ gia đình có liên quan mật thiết đến cuộc sống của họ. Ví dụ, camera trong nhà truyền dữ liệu video từ trong nhà lên đám mây sẽ làm tăng nguy cơ rò rỉ thông tin cá nhân của người dùng. Với việc thực thi Quy định bảo vệ dữ liệu chung của EU (GDPR) [10], các vấn đề bảo mật dữ liệu đã trở nên quan trọng hơn đối với các công ty điện toán đám mây.

Những thách thức trên đã thúc đẩy sự phát triển của điện toán cạnh (edge computing) [11], yêu cầu xử lý dữ liệu ở rìa mạng Internet. Nó đã phát triển nhanh chóng kể từ năm 2014 với khả năng giảm độ trễ và phí băng thông, giải quyết giới

hạn của khả năng tính toán của trung tâm điện toán đám mây và tăng tính khả dụng cũng như bảo vệ quyền riêng tư và bảo mật dữ liệu. Điện toán cạnh là một mô hình mới trong đó tài nguyên của máy chủ cạnh (edge server) được đặt ở rìa Internet, gần với thiết bị IoT và người dùng cuối. Điện toán cạnh có một số lợi thế rõ ràng. Đầu tiên, một lượng lớn dữ liệu tạm thời được xử lý ở rìa mạng, chứ không phải tất cả dữ liệu được tải lên đám mây, điều này giúp giảm đáng kể áp lực lên băng thông và mức tiêu thụ điện của trung tâm điện toán đám mây. Thứ hai, xử lý dữ liệu gần hệ thống cảm biến không yêu cầu phản hồi của trung tâm điện toán đám mây thông qua mạng, điều này giúp giảm đáng kể độ trễ của hệ thống và tăng cường khả năng đáp ứng dịch vụ. Cuối cùng, máy chủ cạnh lưu trữ dữ liệu riêng tư của người dùng trên các thiết bị cạnh thay vì tải lên, giúp giảm nguy cơ rò rỉ dữ liệu mạng và bảo vệ an ninh và quyền riêng tư.

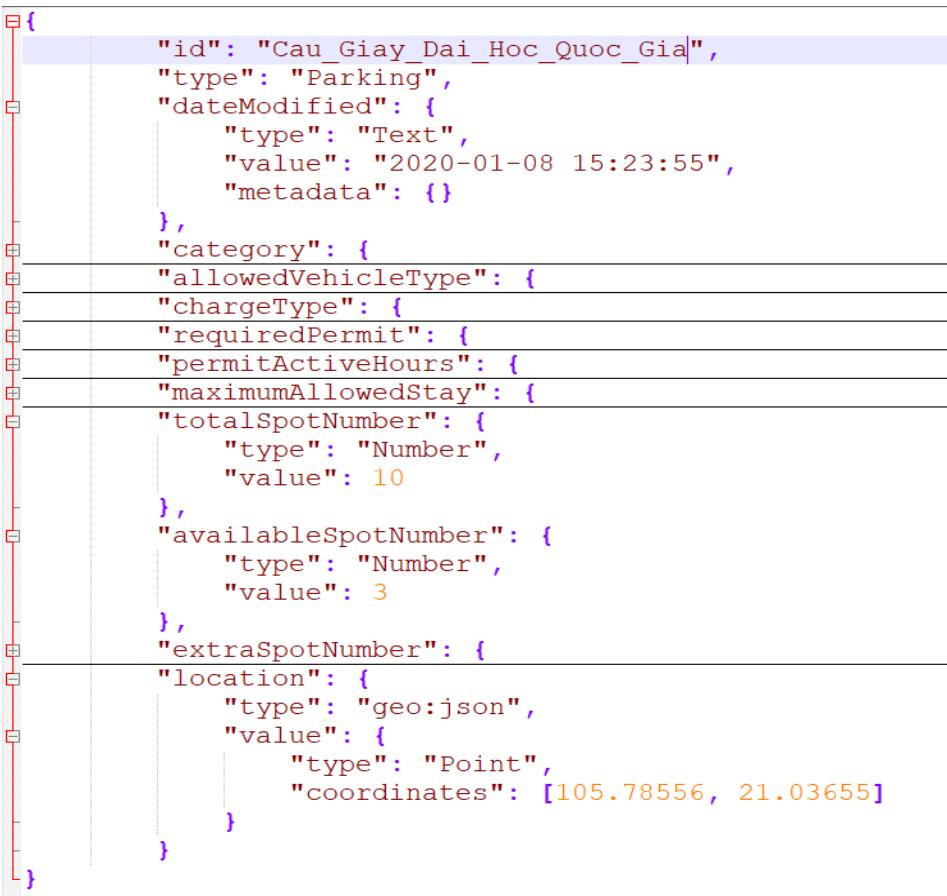
Phân hệ thu nhận và trích rút thông tin từ camera của hệ thống CamNet xử lý dữ liệu video từ các hệ thống camera khác nhau dựa theo mô hình điện toán cạnh. Mỗi khu vực camera khác nhau sẽ được lắp đặt một máy chủ cạnh để thu nhận và trích rút thông tin cần thiết. Sau đó, thông tin sẽ được đẩy lên phân hệ lưu trữ, quản lý thông tin, là một mô hình điện toán đám mây. Số lượng máy chủ cạnh có thể là rất nhiều và hiệu năng xử lý của nó phụ thuộc vào phần cứng cục bộ nên nó không ảnh hưởng đến hiệu năng chung của toàn bộ hệ thống. Hơn nữa, máy chủ cạnh có thể linh hoạt kết nối với các hệ thống camera khác nhau và đầu ra của nó là dữ liệu ngữ nghĩa cũng phải tuân thủ chặt chẽ theo quy tắc chung của hệ thống nên sẽ không có trở ngại khi đẩy dữ liệu lên đám mây lưu trữ dữ liệu. Hình 3.2 mô tả các công việc mà một máy chủ cạnh sẽ thực hiện.



Hình 3.2 Các công việc được xử lý bởi một máy chủ cạnh

Giả sử phân hệ này được cài đặt để trích rút thông tin số ô đỗ xe còn trống từ hình ảnh, một máy chủ cạnh sẽ thu nhận ảnh từ luồng video bằng thư viện FFmpeg[12], sau đó, hình ảnh sẽ được xử lý bởi thuật toán đếm chỗ đỗ xe còn trống

trong bãi đỗ. Kết quả thu được sẽ là số chỗ đỗ xe còn trống trong bãi đỗ tại vị trí có camera giám sát. Kết quả này sẽ được trình bày dưới định dạng JSON một cách hợp lý, đúng quy định của NGSIv2. Thiết kế của một dữ liệu JSON được trình bày ở hình 3.3 là một thiết kế chuẩn cho dữ liệu ngữ nghĩa trong toàn hệ thống. Với các thực thể khác (ví dụ như dữ liệu phát hiện đám đông), các đối tượng trong thực thể được thay đổi phù hợp cho từng yêu cầu khác nhau.



```

{
  "id": "Cau_Giay_Dai_Hoc_Quoc_Gia",
  "type": "Parking",
  "dateModified": {
    "type": "Text",
    "value": "2020-01-08 15:23:55",
    "metadata": {}
  },
  "category": {
    "allowedVehicleType": {},
    "chargeType": {},
    "requiredPermit": {},
    "permitActiveHours": {},
    "maximumAllowedStay": {},
    "totalSpotNumber": {
      "type": "Number",
      "value": 10
    },
    "availableSpotNumber": {
      "type": "Number",
      "value": 3
    },
    "extraSpotNumber": {
      "location": {
        "type": "geo:json",
        "value": {
          "type": "Point",
          "coordinates": [105.78556, 21.03655]
        }
      }
    }
  }
}

```

Hình 3.3 Thiết kế dữ liệu JSON cho bài toán tìm bãi đỗ xe

Ý nghĩa của những đối tượng quan trọng nhất được trình bày ở Bảng 3.1 dưới đây:

Đối tượng	Giá trị	Ý nghĩa
id	Cau_Giay_Dai_Hoc_Quoc_Gia	Định danh của thực thể, ở đây là tên địa điểm đặt camera
type	Parking	Loại thực thể, “Parking” thể hiện đây là dữ liệu của bài toán tìm bãi đỗ xe
dateModified	2020-01-08 15:23:55	Thời gian ghi nhận dữ liệu
totalSpotNumber	10	Tổng số chỗ đỗ xe
availableSpotNumber	3	Số chỗ đỗ xe còn trống

location	105.78556, 21.03655	Tọa độ địa lý của hệ thống camera, tuân theo tiêu chuẩn quốc tế
----------	---------------------	---

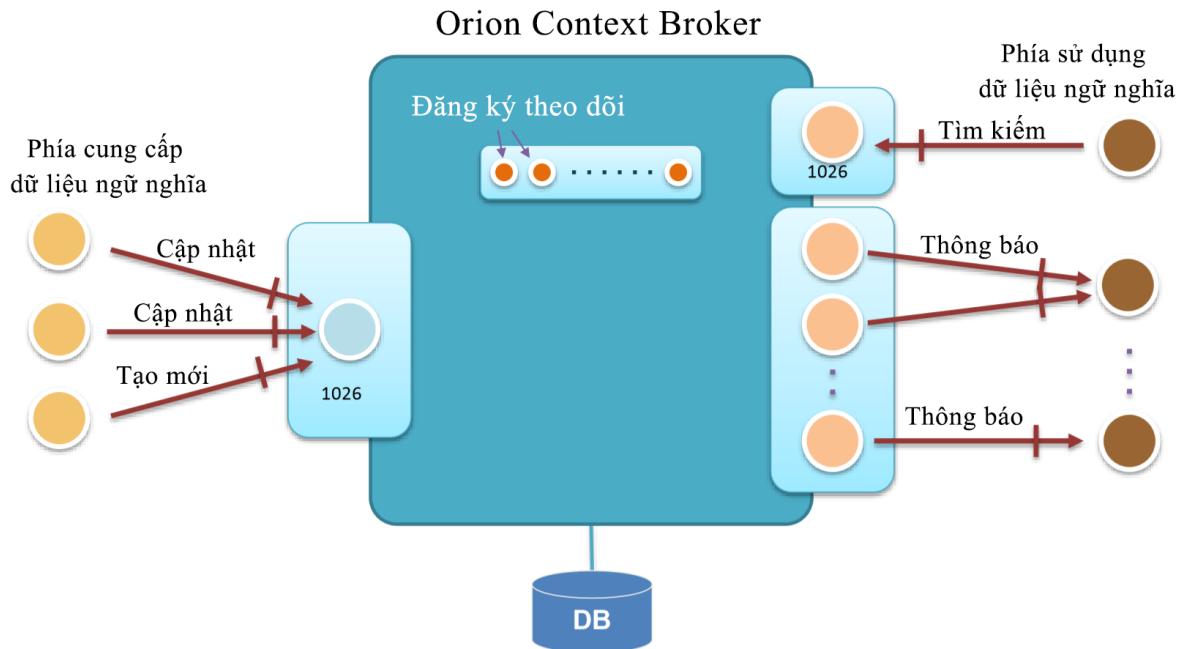
Bảng 3.1 Các đối tượng quan trọng của một thực thể dữ liệu

Dữ liệu đã được chuẩn hóa này sẽ được máy chủ cạnh đưa lên máy chủ quản lý dữ liệu ngữ nghĩa được mô tả ở mục 3.3. Toàn bộ ảnh thu được sau đó sẽ được xóa đi tại máy chủ cạnh để đảm bảo bảo mật thông tin người dùng.

3.3. Phân hệ lưu trữ, quản lý thông tin ngữ nghĩa

Phân hệ này có nhiệm vụ lưu trữ dữ liệu nhận từ máy chủ cạnh và cung cấp dữ liệu dưới dạng “dịch vụ” cho các ứng dụng thông minh. Vì vậy, nó phải được thiết kế hoàn toàn dựa trên mô hình điện toán đám mây. Vấn đề và hướng giải quyết đặt ra ở *Framework xử lý dữ liệu thô cho IoT* [4] rất phù hợp với bài toán lưu trữ, quản lý thông tin ngữ nghĩa nhận từ các máy chủ cạnh. Vì vậy, hệ cơ sở dữ liệu của phân hệ này phải là NoSQL và dữ liệu được lưu trữ ở dạng RAW (JSON, XML...). Nhiệm vụ của phân hệ này được phân định rõ ràng đối với phía nhà cung cấp dữ liệu (context producers) và bên sử dụng dữ liệu (context consumers). Đứng về phía nhà cung cấp dữ liệu, phân hệ phải cung cấp các API cho phép: thêm một thực thể, sửa các thuộc tính trong một thực thể, xóa thực thể. Đứng về bên sử dụng dữ liệu, phân hệ phải có các API đáp ứng việc tìm kiếm dữ liệu một cách linh hoạt, đặc biệt là tìm kiếm theo không gian, thời gian. Ngoài ra, đối với các ứng dụng yêu cầu thông tin theo thời gian thực, phân hệ cũng phải có cơ chế thông báo đến ứng dụng khi thực thể hoặc thuộc tính mà ứng dụng đăng ký có biến động.

Thư viện Orion Context Broker của FIWARE đáp ứng được các yêu cầu trên của phân hệ lưu trữ, quản lý thông tin ngữ nghĩa. Hình 3.4 mô tả thiết kế và cách thức hoạt động của Orion Context Broker. Như đã đề cập ở chương 2, Orion Context Broker sử dụng cơ sở dữ liệu MongoDB, mọi truy vấn đến nó được thực hiện qua NGSIV2 API.



Hình 3.4 Thiết kế và cách thức hoạt động của phân hệ lưu trữ, quản lý thông tin ngữ nghĩa

Orion Context Broker được cài đặt trên một máy chủ (server) cung cấp bởi công ty Digital Ocean với địa chỉ 165.22.62.250 và cấu hình như sau:

OS: UBUNTU 18.04.3 (LTS) X64

CPU: Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz (1 CPU)

RAM: 1GB

HDD: 25GB

Phân hệ được cài đặt bằng cách sử dụng Docker[13] và Docker Compose [14]. Orion Context Broker container và MongoDB container sẽ được liên kết với nhau nhờ Docker Compose dựa vào những thiết lập dưới đây:

```

mongo:
  image: mongo:3.6
  command: --nojournal

orion:
  image: fiware/orion

links:
  - mongo

ports:
  - "1026:1026"

```

command: -dbhost mongo

Mọi giao tiếp tới phân hệ được thực hiện qua cổng 1026. Dữ liệu ngữ nghĩa được cung cấp qua địa chỉ *165.22.62.250:1026/v2/entities*, các ứng dụng có thể đăng ký theo dõi thực thể qua địa chỉ *165.22.62.250:1026/v2/subscription*. Orion Context Broker hỗ trợ rất mạnh mẽ việc tìm kiếm theo vị trí địa lý. Ví dụ với bài toán tìm bến đỗ xe, với dữ liệu ngữ nghĩa được thiết kế như ở mục 3.2, với vị trí của xe hơi là (*lat, lon*) ta có thể dễ dàng tìm kiếm tất cả các thực thể bến đỗ xe trong bán kính *R* (m) bằng lệnh truy vấn sau đây:

```
http://165.22.62.250:1026/v2/entities?  
coords=lat,lon           // vị trí địa lý của chiếc xe  
&georel=near;maxDistance:R // tìm các thực thể trong bán kính R mét  
&type=Parking            // tìm các thực thể là bến đỗ xe  
&geometry=point          // loại vị trí địa lý, ở đây là tọa độ một điểm
```

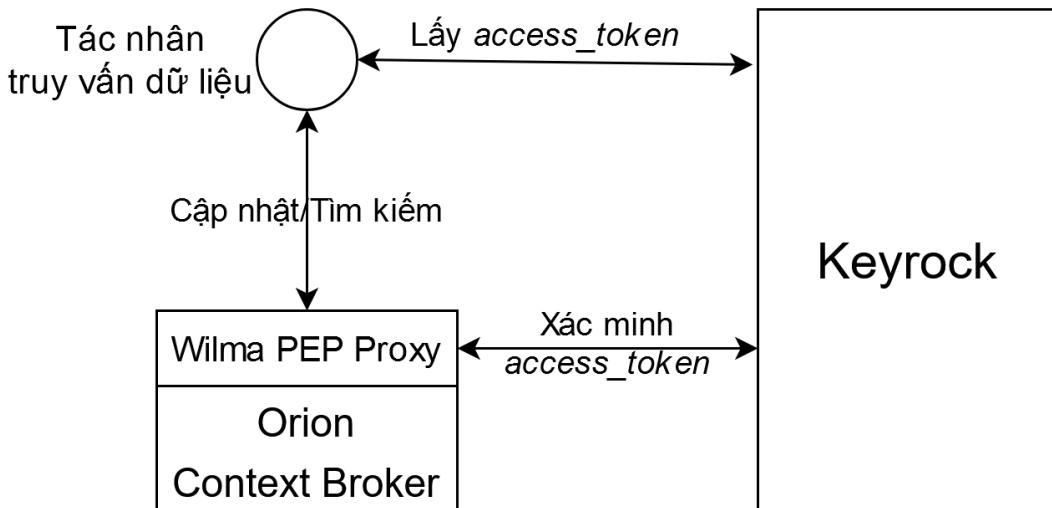
Như vậy, phân hệ quản lý thông tin ngữ nghĩa với nền tảng là Orion Context Broker đã đáp ứng được các yêu cầu đặt ra là có khả năng lưu trữ linh hoạt các dữ liệu dạng thô (RAW), cung cấp NGSIV2 API có khả năng tương thích cao với các nền tảng khác, cho phép tạo mới, xóa, cập nhật dữ liệu, truy vấn dữ liệu với các bộ lọc linh hoạt, đặc biệt là khả năng truy vấn theo vị trí địa lý có ý nghĩa rất lớn với các bài toán IoT.

3.4. Phân hệ quản lý định danh và quyền người dùng

Hiện nay, yêu cầu về bảo mật, an toàn thông tin là một yêu cầu bắt buộc đối với bất kỳ hệ thống nào, nhất là với các hệ thống dựa trên điện toán đám mây. Bảo mật hệ thống được chia ra làm hai vấn đề chính là:

- Xác minh định danh của người dùng (user identity): bất kỳ tác nhân nào truy cập vào hệ thống đều phải được cung cấp một tài khoản (account) bao gồm tên đăng nhập (username) và mật khẩu (password) hợp lệ. Không có một tài khoản hợp lệ sẽ không thể thực hiện bất kỳ hành động gì đối với hệ thống.
- Xác định quyền của người dùng (user permission): hệ thống CamNet có các nguồn cấp dữ liệu khác nhau, dữ liệu được cung cấp có thể thuộc các loại khác nhau (như dữ liệu bến đỗ xe, dữ liệu đám đông...), và các ứng dụng tìm kiếm thông tin trong hệ thống cũng có các mục tiêu tìm kiếm khác nhau. Vì vậy, mỗi đối tượng tương tác với hệ thống cần có một bộ các quyền khác nhau, phù hợp với yêu cầu, mục đích của từng đối tượng.

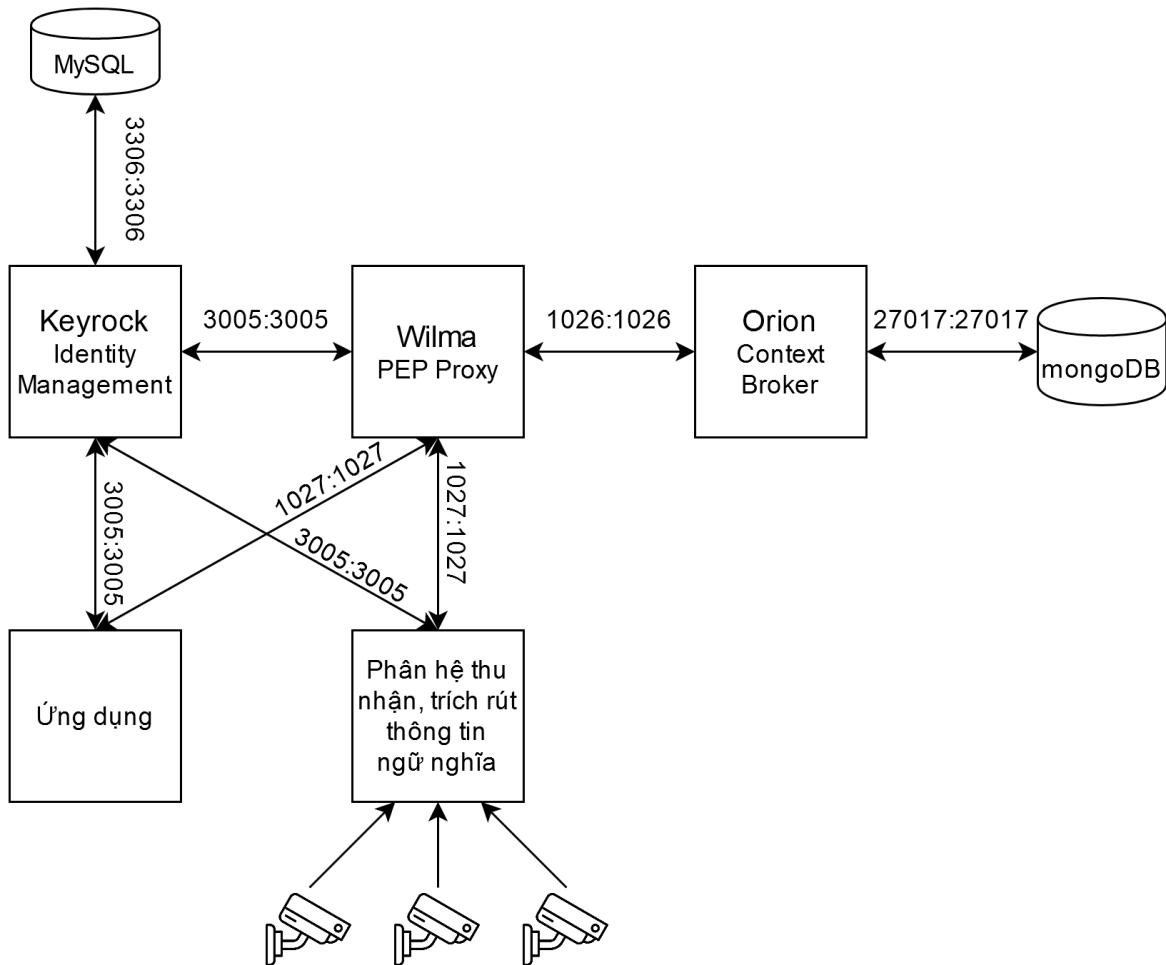
Với việc sử dụng FIWARE Orion Context Broker cho trung tâm quản lý dữ liệu, ta có thể tích hợp 2 thư viện của FIWARE là Keyrock Identity Manager và Wilma PEP Proxy để tạo một bức tường bảo mật cho hệ thống. Nó bao hàm cả chức năng xác minh định danh người dùng và xác định quyền người dùng nêu trên, sơ đồ hoạt động của phân hệ quản lý định danh và quyền người dùng được thể hiện ở hình 3.5.



Hình 3.5 Sơ đồ hoạt động của phân hệ quản lý định danh và quyền người dùng

Trung tâm dữ liệu Orion Context Broker hoàn toàn được ẩn đi ở phía tác nhân sử dụng hệ thống. Mọi giao tiếp với Orion Context Broker được thực hiện thông qua Wilma PEP Proxy. Trước tiên, các tác nhân phải yêu cầu *access_token* từ Keyrock với một tài khoản hợp lệ, sau đó, chúng sẽ sử dụng *access_token* này để gửi yêu cầu tới Wilma PEP proxy, Wilma sẽ thực hiện việc xác minh *access_token* và chuyển yêu cầu đến Orion Context Broker, nhận phản hồi và chuyển phản hồi về phía tác nhân yêu cầu.

Trong luận văn này, Keyrock và Wilma được cài đặt trên cùng một máy chủ với Orion Context Broker. Tuy nhiên khi chi phí cho phép, ta có thể đặt Orion Context Broker ở một máy chủ riêng và Keyrock + Wilma ở một máy chủ hoàn toàn khác để tăng cường khả năng bảo mật. Sơ đồ kết nối chi tiết và các cổng giao tiếp của từng thành phần của toàn bộ hệ thống CamNet được thể hiện ở hình 3.6.



Hình 3.6 Sơ đồ kết nối của hệ thống

Sau khi cài đặt Keyrock và Wilma, thực tế, chúng vẫn chưa được kết nối với Orion Context Broker. Tài khoản *admin* của Keyrock đã được định nghĩa trong khi cài đặt, vì vậy, ta có thể sử dụng tài khoản *admin* để thực hiện liên kết Keyrock với Orion và thiết lập các quyền phù hợp cho từng nhóm người dùng. Đầu tiên, để một ứng dụng thuộc FIWARE được bảo vệ bởi Keyrock, nó phải được đăng ký trong Keyrock Applications. Vì vậy, Orion Context Broker phải được thêm vào như một ứng dụng thuộc Keyrock. Trong luận văn này, Orion Context Broker được thêm vào Keyrock như một ứng dụng với tên gọi là *Orion Service*. (Hình 3.7 Giao diện quản lý ứng dụng đã đăng ký trong Keyrock).

The screenshot shows the Keyrock Identity Manager web application. At the top, there's a navigation bar with the Keyrock logo, the text "Identity Manager", and a user profile for "keyrock admin". Below the header is a "Main menu" sidebar with links to Home, Organizations, Applications, Notify, Administrators, and Users.

The main content area is titled "Orion Service" and includes tabs for "edit" and "manage roles". It displays the following details:

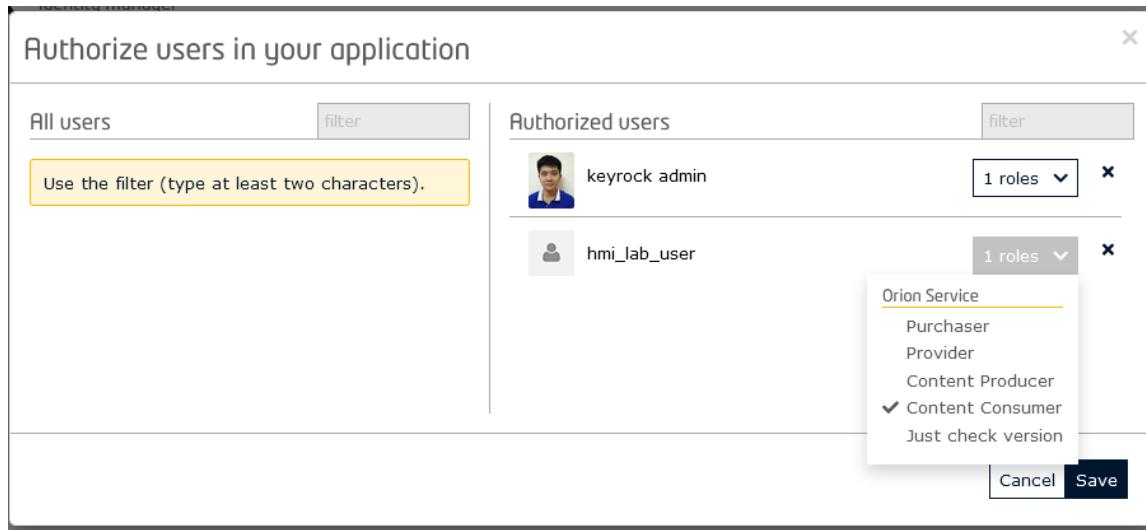
- Description:** Orion protected by OAuth2 and Keyrock
- Url:** http://localhost:1026
- Callback Url:** http://localhost:1026
- OAuth2 Credentials:** A section with a question mark icon.
- PEP Proxy:** A section with a question mark icon.
- IoT Sensors:** A section with a question mark icon.
- Trusted applications:** A table with a "filter" input and a "+ Add" button. It contains a single row: "This application does not have any trusted application."
- Authorized users:** A table with a "filter" input and an "Authorize" button. It lists two users: "keyrock admin" and "hmi_lab_user".
- Authorized organizations:** A table with a "filter" input and an "Authorize" button. It contains a single row: "This application does not have any authorized organizations."

At the bottom of the page, there's a footer with the text "2018 © GINGER-DIT - Technical University of Madrid (UPM)" and a language selection dropdown showing "en" and "es".

Hình 3.7 Giao diện quản lý ứng dụng đã đăng ký trong Keyrock

Sau đó, ta phải cài đặt các quyền cho từng đối tượng người dùng, đó là:

- Quyền chỉ thêm mới hoặc cập nhật dữ liệu (Content Producer): Đối với phía nhà cung cấp dữ liệu ngữ nghĩa là các máy chủ cạnh, nhiệm vụ của nó là cung cấp dữ liệu cho hệ thống, nên chỉ có việc thêm mới hoặc cập nhật dữ liệu được cho phép.
- Quyền truy vấn dữ liệu (Content Consumer): Phía các ứng dụng kết nối với hệ thống CamNet chỉ được cấp quyền truy vấn dữ liệu. Ngoài ra, ta có thể thêm các giới hạn quyền nữa cho từng ứng dụng như giới hạn loại thực thể có thể truy vấn (ví dụ giới hạn `type=Parking` đối với các ứng dụng tìm kiếm bãi đỗ xe) hay giới hạn theo vị trí của hệ thống camera. Ví dụ ở hình 3.8, người dùng có tên `hmi_lab_user` được cấp quyền *Content Consumer* nên nó được phép tìm kiếm dữ liệu được quản lý bởi hệ thống.



Hình 3.8 Thiết lập quyền cho từng đối tượng người dùng trong Keyrock

Đối với người dùng thường, Keyrock cung cấp một giao diện trực quan để thực hiện các công việc cơ bản: đăng nhập, tạo tài khoản, tìm lại mật khẩu, sau khi đăng nhập, người dùng có thể quản lý tài khoản, tra cứu, quản lý thông tin nhóm và các ứng dụng của mình. Sau khi được thêm vào ứng dụng *Orion Service* và thiết lập các quyền bởi *admin*, tài khoản người dùng có thể được sử dụng để yêu cầu *access_token* và dùng *access_token* để gửi các yêu cầu lên hệ thống.

Chương 4. Phát triển ứng dụng tìm bãi đỗ xe và đánh giá kết quả

4.1. Xây dựng kịch bản kiểm thử khả năng ứng dụng của hệ thống

Xã hội phát triển, thu nhập tăng lên cùng với những ưu đãi của nhà nước giúp cho việc sở hữu một chiếc ô-tô trở nên không quá khó khăn. Số lượng xe lớn dẫn đến việc tìm kiếm một chỗ đỗ xe trở thành một vấn đề không nhỏ với các tài xế, và nó càng khó khăn hơn tại các đô thị lớn như Hà Nội và TP. Hồ Chí Minh. Nó được coi là một vấn đề xã hội cấp thiết, có ảnh hưởng lớn tới cuộc sống của người dân. Mặt khác, các bãi đỗ xe tại các đô thị lớn hiện nay ngoài lực lượng quản lý chính là nhân viên bảo vệ thì hầu hết đều bổ sung thêm hệ thống camera giám sát để tăng cường an ninh cho khu vực, nên việc kết nối vào các hệ thống sẵn có này là hoàn toàn khả thi. Hai yếu tố trên tạo ra động lực để xây dựng một phần mềm tích hợp khả năng tìm kiếm bãi đỗ xe còn chỗ trống vào ứng dụng bản đồ - ứng dụng đã quen thuộc với các tài xế ô-tô.

Trong luận văn này, tác giả phát triển một ứng dụng tích hợp hệ thống CamNet vào bản đồ HERE Maps để chỉ đường cho tài xế đến bãi đỗ xe còn chỗ trống gần điểm đến nhất. Một thuật toán đếm số chỗ trống trong bãi đỗ xe đề xuất ở mục 4.2 sẽ được tích hợp vào phân hệ nhận diện, trích rút dữ liệu từ camera để tạo dữ liệu về trạng thái các bãi đỗ xe. Ứng dụng đưa đến người dùng cuối được xây dựng trên nền tảng Android và HERE Maps SDK, chi tiết thiết kế và cài đặt ứng dụng được đề cập ở mục 4.3.

4.2. Cài đặt thuật toán đếm số chỗ trống trong bãi đỗ xe

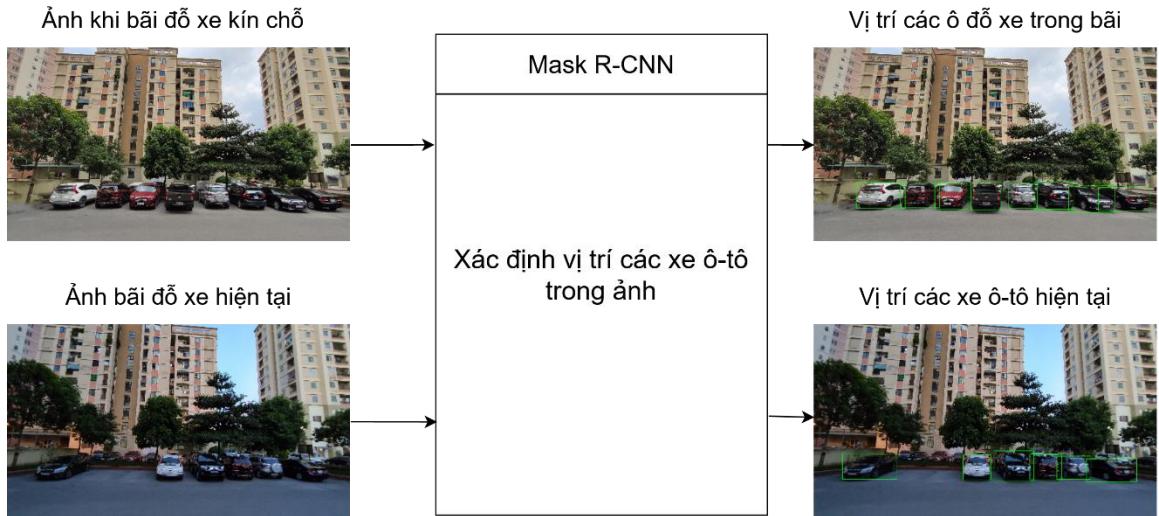
Bài toán này có đầu vào là một ảnh thu nhận được từ camera giám sát ghi lại hình ảnh hiện tại của bãi đỗ xe, đầu ra là số lượng chỗ trống trong bãi đỗ xe đó. Thực nghiệm trên camera giám sát cho bãi đỗ xe ngoài trời của khu đô thị Handiresco, Cố Nhuế 2, Bắc Từ Liêm, Hà Nội, ta có thể thu được hình ảnh làm đầu vào cho thuật toán như hình 4.1 dưới đây.



Hình 4.1 Hình ảnh bãi đỗ xe là đầu vào của thuật toán

Sau khi có được dữ liệu đầu vào, thuật toán sẽ thực hiện ba bước để trích rút thông tin số lượng ô đỗ xe còn trống, đó là: bước 1, xác định vị trí các ô đỗ xe có trong ảnh; bước 2, xác định vị trí các xe ô-tô có trong ảnh; bước 3, xác định ô đỗ xe đã có xe đỗ hay chưa, dựa vào việc tổng hợp kết quả của bước 1 và 2. Vị trí của một ô đỗ xe hoặc một chiếc xe ô-tô được thể hiện bởi một hình bao chữ nhật (*bounding boxes*).

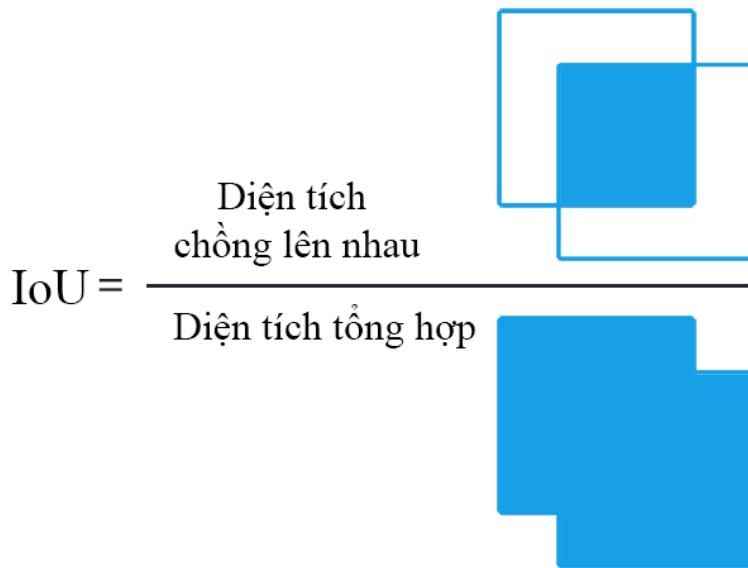
Vị trí các ô đỗ xe thường được phân biệt bằng vạch sơn trắng, nhưng trong đa số trường hợp bãi đỗ xe ngoài trời, vạch sơn này rất mờ, hơn nữa, độ phân giải của ảnh từ camera là không cao, nên không thể dựa vào nó để xác định vị trí (ví dụ ở hình 4.1, vạch sơn thậm chí không thể xác định được bằng mắt thường). Để xác định được vị trí các ô đỗ xe có trong ảnh, ta đưa ra nhận định: khi bãi đỗ xe không còn chỗ trống, thì vị trí của từng chiếc xe đỗ trong bãi chính là vị trí của ô đỗ xe. Như vậy, để tìm *bounding box* của các ô đỗ xe, ta sẽ lấy *bounding box* của các xe ô-tô trong bãi khi bãi không còn chỗ trống. Để tìm *bounding box* của các xe ô-tô trong bãi, ta cần sử dụng một thuật toán phát hiện đối tượng từ ảnh (object detection algorithms).



Hình 4.2 Quá trình nhận biết vị trí các ô đỗ xe và các xe ô-tô

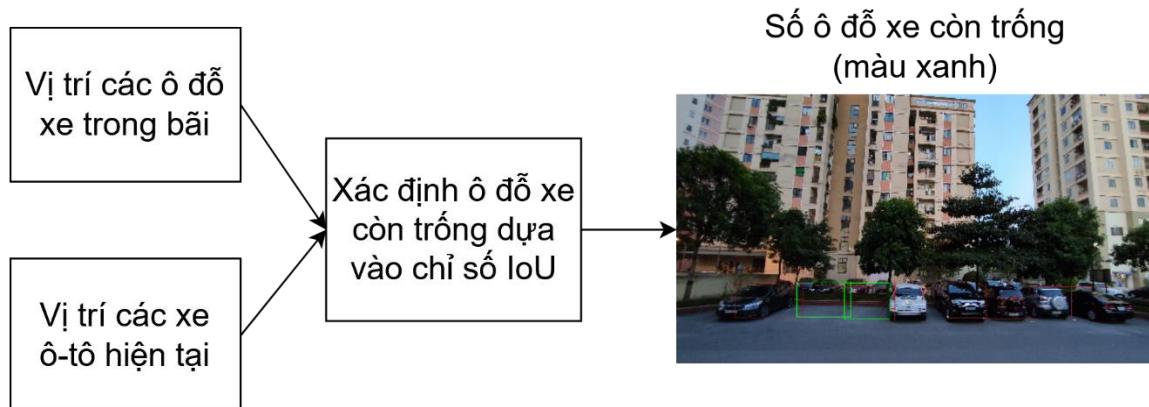
Các hướng tiếp cận sử dụng học sâu tỏ ra ưu việt cho bài toán này với các lợi thế về độ chính xác, độ ổn định khi các đối tượng trong ảnh xoay theo các hướng khác nhau. Trong thực nghiệm này, tác giả sử dụng kiến trúc Mask R-CNN [20] và một mô hình huấn luyện trước (pre-trained model) được huấn luyện trên tập dữ liệu COCO [29] để phát hiện các xe ô-tô có trong hình ảnh. Cài đặt mã nguồn mở của Mask R-CNN trên Python 3, Keras, và TensorFlow được phát triển bởi Waleed Abdulla [22]. Hình 4.2 mô tả quá trình nhận biết vị trí các ô đỗ xe và các xe ô-tô. Đối với mỗi đối tượng phát hiện được trong ảnh, ta lấy được ba thông tin quan trọng: loại đối tượng được phát hiện dưới dạng số nguyên, mô hình này được huấn luyện để phát hiện được 80 loại đối tượng khác nhau, nhưng trong đó, ta chỉ quan tâm đến các thể loại là xe ô-tô như Car, Truck; điểm số tin cậy (confidence score) của việc phát hiện đối tượng; và *bounding box* của đối tượng trong ảnh, gồm các tọa độ (x,y) của điểm ảnh.

Cuối cùng, ta cần xác định vị trí ô đỗ xe đã có xe đỗ hay chưa. Sau bước 1 và 2, ta có 2 mảng *bounding boxes*: 1 mảng cho vị trí các chỗ đỗ xe, 1 mảng cho vị trí các xe ô-tô hiện có trong ảnh. Vị trí của camera là cố định nên hình ảnh thu nhận được có kích thước, góc độ là cố định, vì vậy ta có thể đưa ra nhận định: nếu một *bounding box* của một vị trí đỗ xe bị che phủ bởi một *bounding box* của một chiếc xe, thì vị trí đó đã có xe đỗ. Chỉ số che phủ được sử dụng là Intersection Over Union hay IoU [21], nó được tính toán bằng cách tìm lượng điểm ảnh mà 2 đối tượng trùng nhau và chia cho số lượng điểm ảnh được bao phủ bởi cả 2 đối tượng, như thể hiện ở hình 4.3.



Hình 4.3 Công thức tính độ che phủ IoU

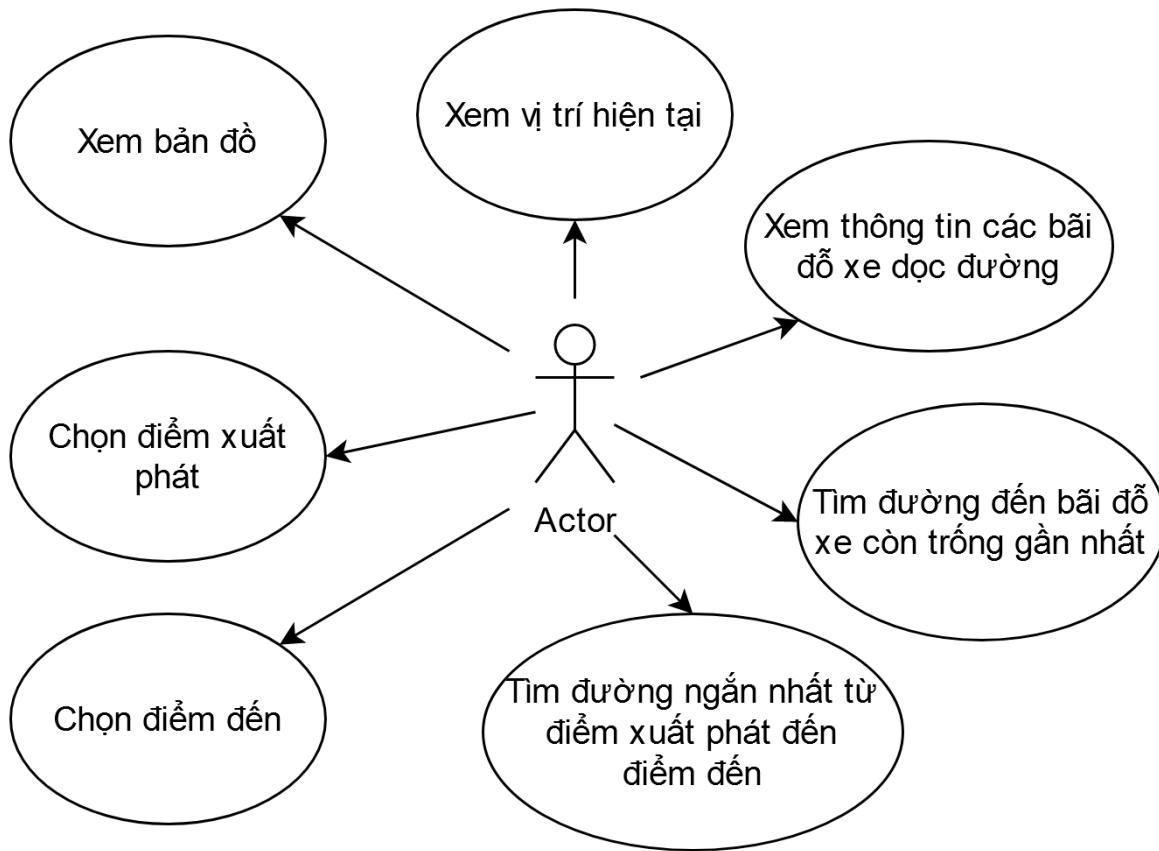
Chỉ số IoU này sẽ cho biết mức độ một *bounding box* của xe ô-tô chồng lên một *bounding box* của vị trí đỗ xe. Từ đó có thể kết luận xe có ở trong bãi đỗ hay không. Nếu chỉ số IoU thấp, trong luận văn này đặt mức là < 0.2 , chứng tỏ chiếc xe không chiếm phần lớn vị trí đỗ xe, thể hiện xe sắp rời đi hoặc bị ảnh hưởng bởi xe đỗ bên cạnh. Trái lại, nếu chỉ số IoU cao thì chứng tỏ vị trí đó đã có xe đỗ. Từ kết quả này, ta đếm được số lượng vị trí còn trống trong bãi đỗ. Hình 4.4 dưới đây mô tả phương pháp phát hiện ô đỗ xe còn trống, các ô màu đỏ thể hiện vị trí đỗ xe không còn trống, ô màu xanh thể hiện vị trí đang còn trống.



Hình 4.4 Kết quả thực nghiệm của thuật toán đếm số lượng vị trí đỗ xe còn trống

4.3. Thiết kế và cài đặt ứng dụng

Là một ứng dụng chỉ đường nên các tính năng bắt buộc phải có của ứng dụng là: xem bản đồ, xem vị trí hiện tại, chọn điểm xuất phát, chọn điểm đến, tìm đường đi ngắn nhất từ điểm xuất phát tới điểm đến. Ngoài ra, ta sẽ tích hợp thêm tính năng hiển thị trạng thái các bãi đỗ xe trên đường đi và chỉ đường tới bãi đỗ xe còn chỗ trống gần điểm đến nhất, các tính năng này được xây dựng dựa trên thông tin lấy được từ hệ thống CamNet. Hình 4.5 thể hiện biểu đồ use-case của ứng dụng.

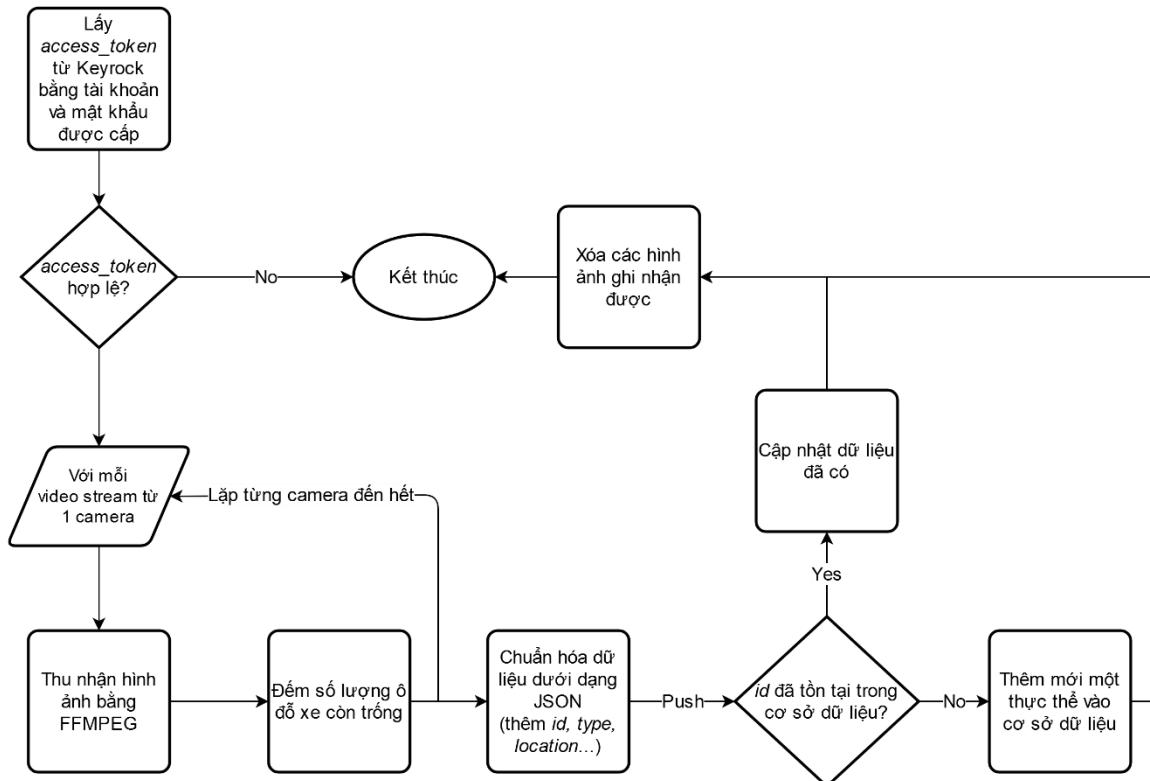


Hình 4.5 Biểu đồ use-case của ứng dụng

Ứng dụng được phát triển trên nền tảng Android 9, Android SDK 28. Các tính năng cơ bản của một bản đồ đề cập trên đây được cung cấp bởi HERE SDK cho Android [15]. HERE SDK phiên bản premium không chỉ cung cấp đầy đủ các API cần thiết cho các tính năng của ứng dụng mà nó còn rất linh hoạt, mang lại khả năng tùy biến cao. Nó cho phép nhà phát triển thêm các đối tượng tùy chỉnh vào bản đồ, nhờ vậy mà việc thêm những hình ảnh, số lượng chỗ trống của các bãi đỗ xe vào bản đồ trở nên dễ dàng hơn.

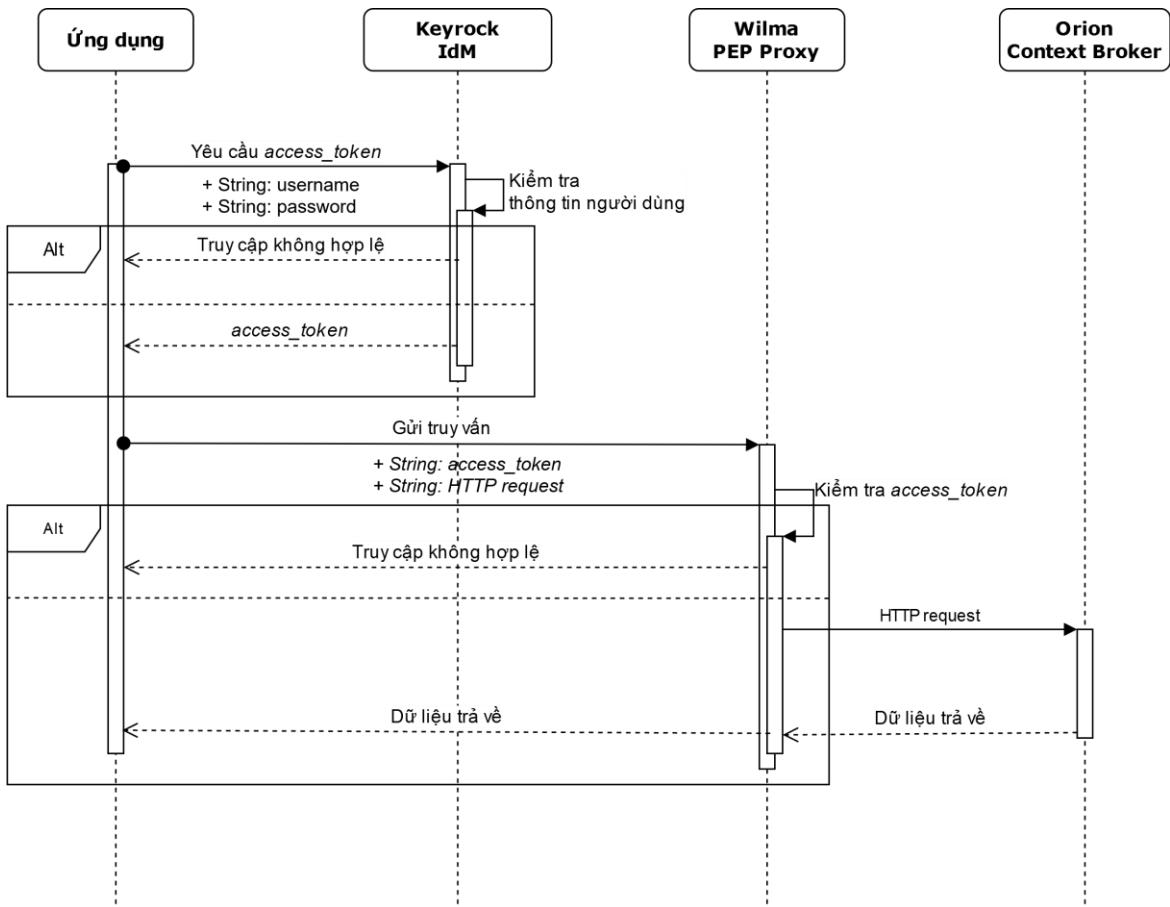
Ở phân hệ thu nhận, trích rút thông tin ngữ nghĩa, một máy chủ cạnh được kết nối với hệ thống camera giám sát và có thể thu nhận được các luồng video của các camera. Nó được đăng nhập vào hệ thống CamNet để lấy *access_token* với quyền

“provider”. Mỗi một khoảng thời gian nhất định, với mỗi luồng video, máy chủ cạnh sẽ ghi nhận lại một hình ảnh. Hình ảnh này được đưa qua thuật toán đếm số chỗ đỗ xe còn trống được đề xuất ở mục 4.2 để lấy thông tin số lượng ô đỗ xe còn trống có trong ảnh. Kết quả thu được sẽ được chuẩn hóa dạng JSON như đã đề cập ở chương 3 và được đẩy lên phân hệ lưu trữ, quản lý dữ liệu ngữ nghĩa. Ảnh thu nhận được từ camera sẽ được xóa tại máy chủ cạnh sau khi xử lý thành công. Hình 4.6 thể hiện sơ đồ khái của quá trình trích rút dữ liệu ngữ nghĩa được thực hiện ở máy chủ cạnh.



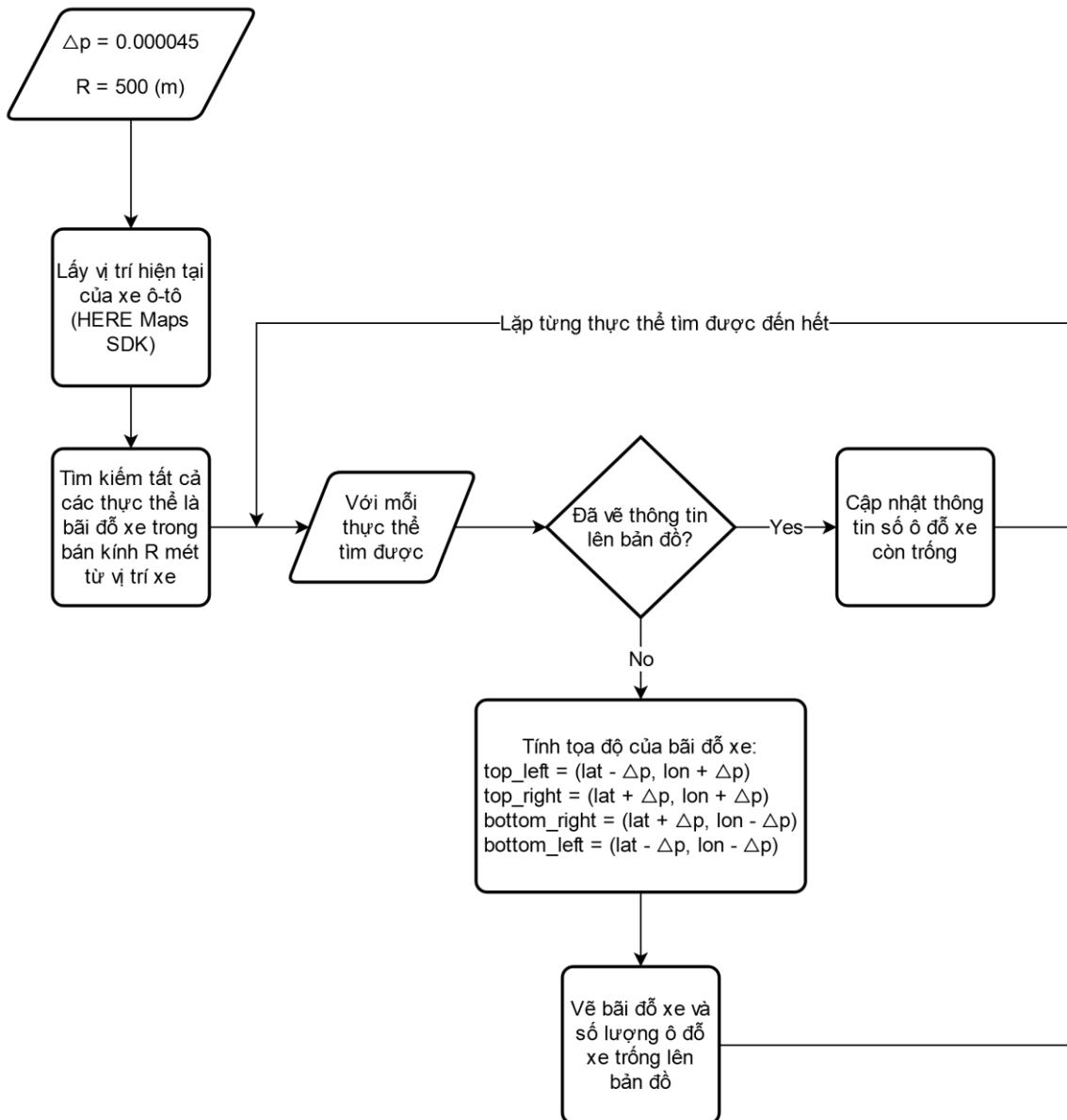
Hình 4.6 Sơ đồ khái quát quá trình trích rút dữ liệu ngữ nghĩa

Như vậy, dữ liệu ngữ nghĩa về thông tin bãi đỗ xe đã hiện hữu ở máy chủ quản lý thông tin ngữ nghĩa. Việc của ứng dụng Android là tìm kiếm các thông tin cần thiết để phát triển tính năng tìm bãi đỗ xe. Cũng như ở máy chủ cạnh, việc đầu tiên ứng dụng phải làm là đăng nhập vào hệ thống để lấy *access_token* với quyền “*consumer*” – quyền cho phép tìm kiếm thông tin ngữ nghĩa. Quá trình đăng nhập và lấy thông tin được thể hiện ở biểu đồ ở hình 4.5 dưới đây.



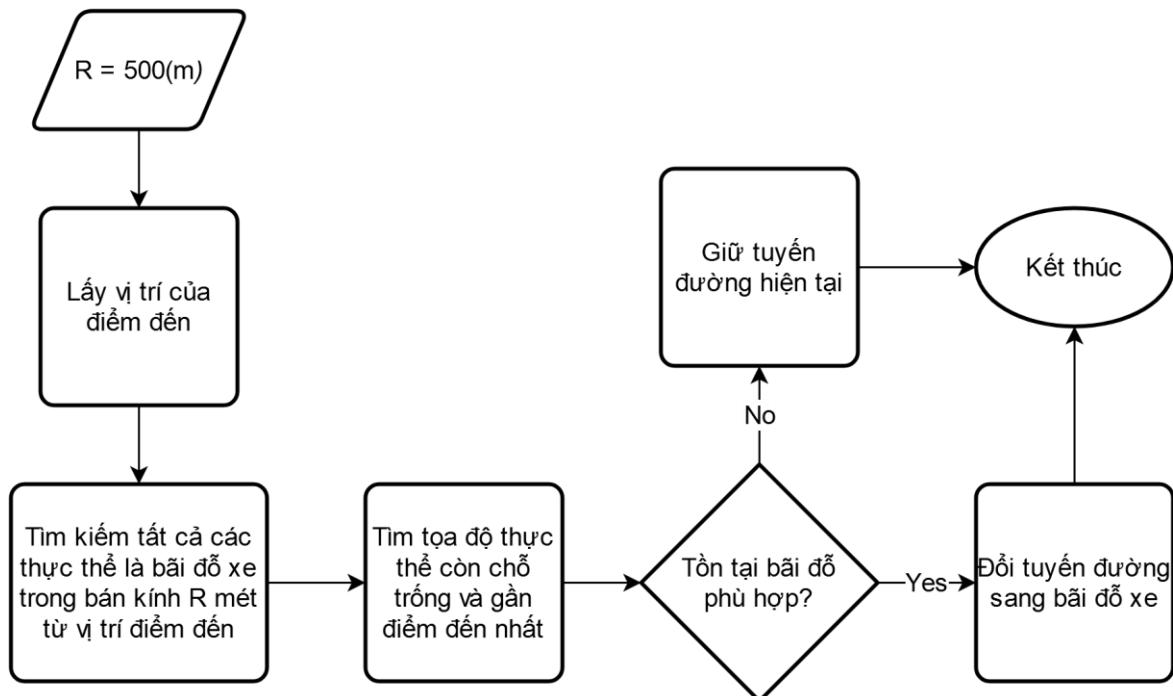
Hình 4.7 Luồng xác minh định danh và quyền của ứng dụng

Tính năng hiển thị tất cả các bãi đỗ xe và số lượng chỗ đỗ xe còn trống là sự kết hợp của vị trí hiện tại của chiếc xe (dữ liệu từ HERE maps) và khả năng tìm kiếm theo không gian của hệ thống quản lý thông tin ngữ nghĩa. Vị trí địa lý của các bãi đỗ xe (hay chính là vị trí của hệ thống camera) là một điểm (*point*) với 2 tọa độ (*latitude*, *longitude*), nhưng trên bản đồ, bãi đỗ xe được thể hiện bởi một hình (*shape*). Vì vậy, ta cần tính toán ra một hình vuông với kích thước phù hợp và có tâm là vị trí của bãi đỗ xe bằng cách cộng/trừ các tọa độ với một hằng số $\Delta p = 0.000045$ với đơn vị là khoảng cách tọa độ địa lý, tương đương với 5 mét. Ứng dụng sẽ tìm kiếm tất cả các bãi đỗ xe trong bán kính $R = 500$ (mét) kể từ vị trí hiện tại của xe ôtô trong máy chủ quản lý thông tin ngữ nghĩa. Sơ đồ khối của quá trình hiển thị tất cả các bãi đỗ xe và số lượng ô đỗ xe trống được trình bày ở hình 4.8 dưới đây, điều kiện cần là ứng dụng đã có *access_token* hợp lệ.



Hình 4.8 Sơ đồ khái của quá trình hiển thị các bãi đỗ xe và số lượng ô đỗ xe trống

Tính năng tìm kiếm và chỉ đường đến bãi đỗ xe còn chỗ trống gần nhất cũng lấy dữ liệu bãi đỗ xe từ trung tâm quản lý dữ liệu, sau khi có tọa độ của bãi đỗ xe, API tìm đường của HERE SDK được gọi để tìm đường đi ngắn nhất đến vị trí đó. Thông tin chỉ đường cũ tới điểm đến được hủy bỏ và thay bằng thông tin chỉ đường mới đến bãi đỗ xe gần nhất. Bán kính tìm kiếm xung quanh điểm đến vẫn là $R = 500$ (mét). Sơ đồ khái của quá trình tìm bãi đỗ xe còn chỗ trống gần nhất được trình bày ở hình 4.9 dưới đây.



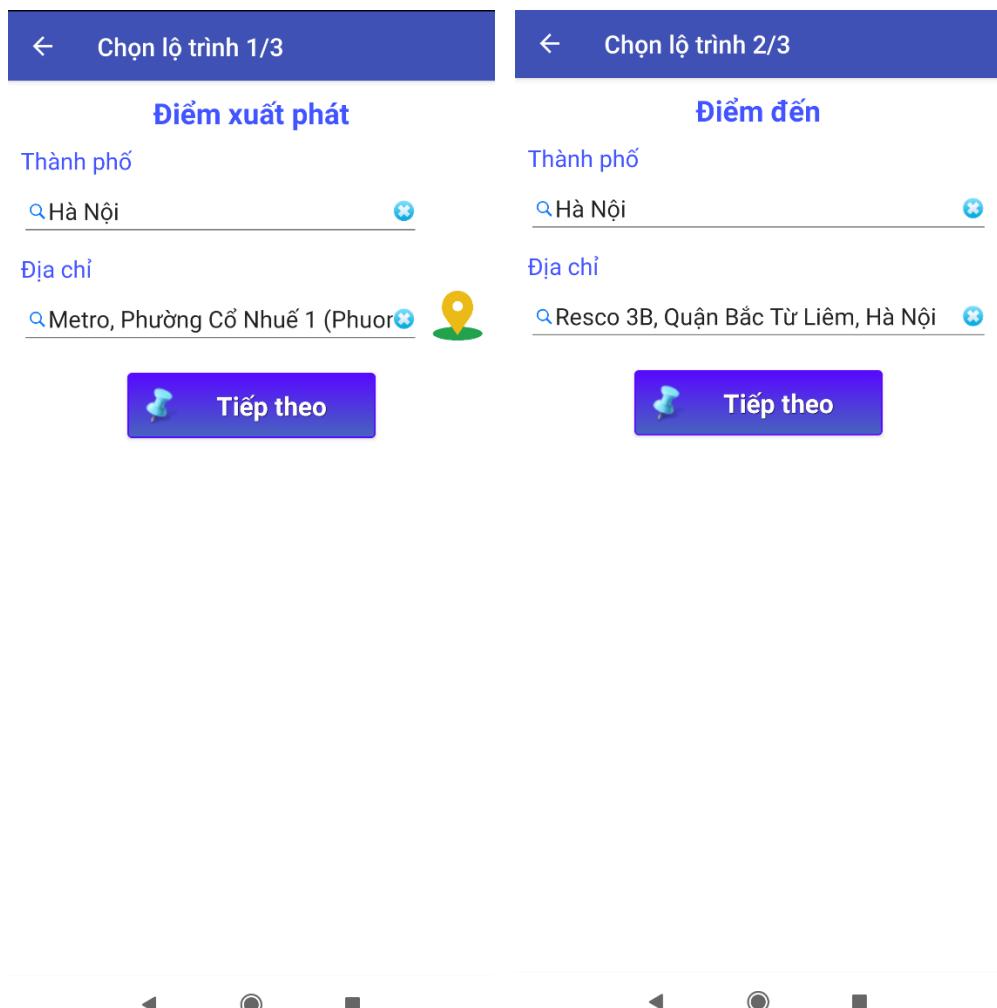
Hình 4.9 Sơ đồ khái của quá trình tìm bãi đỗ xe còn chỗ trống gần nhất

4.4. Thủ nghiệm ứng dụng

Như đã trình bày ở các phần trên, ứng dụng bản đồ HERE Maps sau khi được tích hợp hệ thống CamNet có thể hiển thị thông tin của tất cả các bãi đỗ xe trên đường đi và tìm đường đến bãi đỗ xe gần điểm đến nhất khi người dùng gần kết thúc chặng đường. Giả định rằng, điểm xuất phát của tài xế là siêu thị Metro, Bắc Từ Liêm, Hà Nội với tọa độ $(21.054266, 105.780990)$, đích đến là tòa nhà OCT3-B, khu đô thị Resco, Xuân Đỉnh, Bắc Từ Liêm, Hà Nội với tọa độ $(21.067474, 105.783576)$; hai khu vực đỗ xe có gắn hệ thống camera giám sát kết nối với hệ thống CamNet là tòa nhà OCT3-A, khu đô thị Resco với tọa độ $(21.064345, 105.782880)$ và tòa nhà OCT3-B9, khu đô thị Resco với tọa độ $(21.067419, 105.783088)$. Ứng dụng được cài đặt trên điện thoại thông minh với cấu hình như sau:

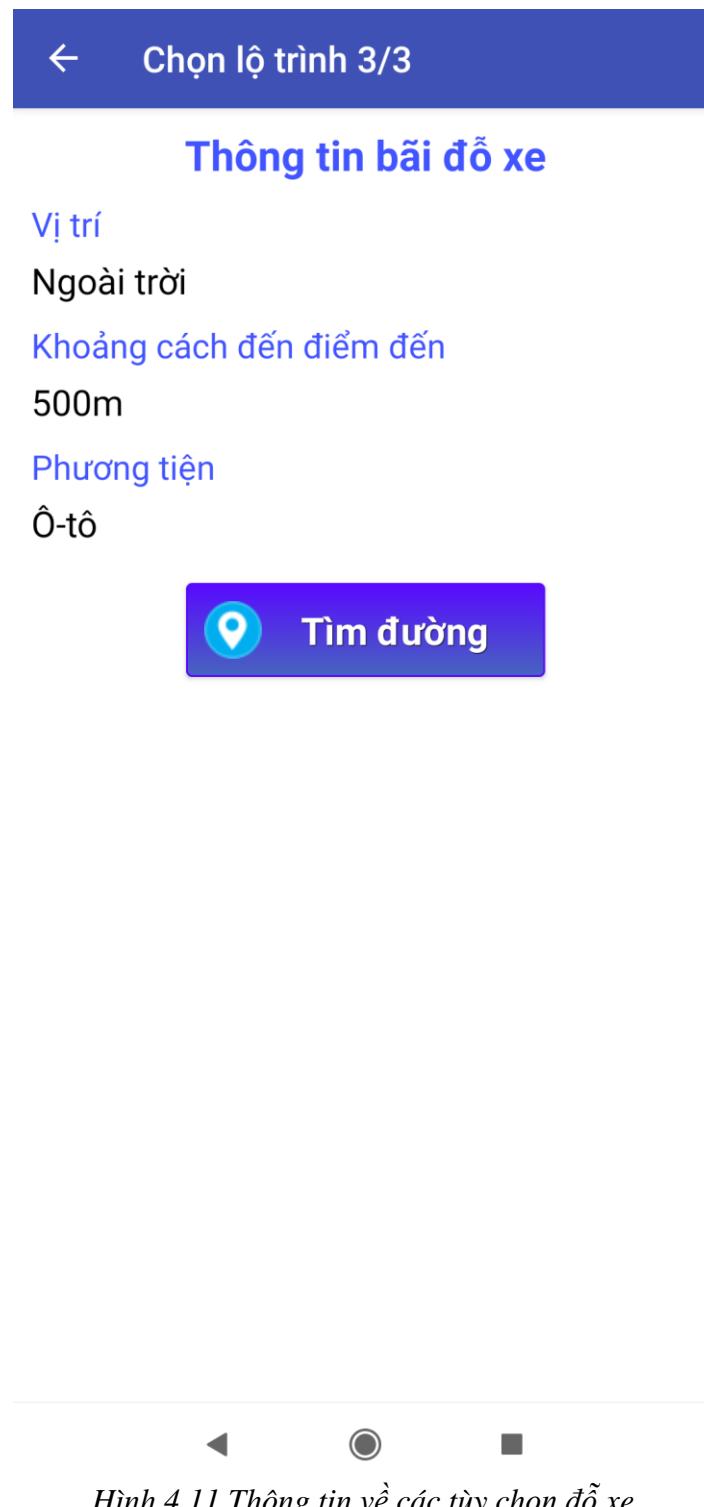
Tên thiết bị: Xiaomi Mi 9 Lite
Hệ Điều Hành: MIUI 11 trên nền tảng Android 10
CPU: Qualcomm Snapdragon 710 (8CPUs)
RAM: 6GB
Bộ nhớ trong: 64GB

Đầu tiên, chúng ta sẽ có một giao diện để chọn điểm xuất phát và điểm đến (Hình 4.10 Chọn điểm xuất phát và điểm đến).



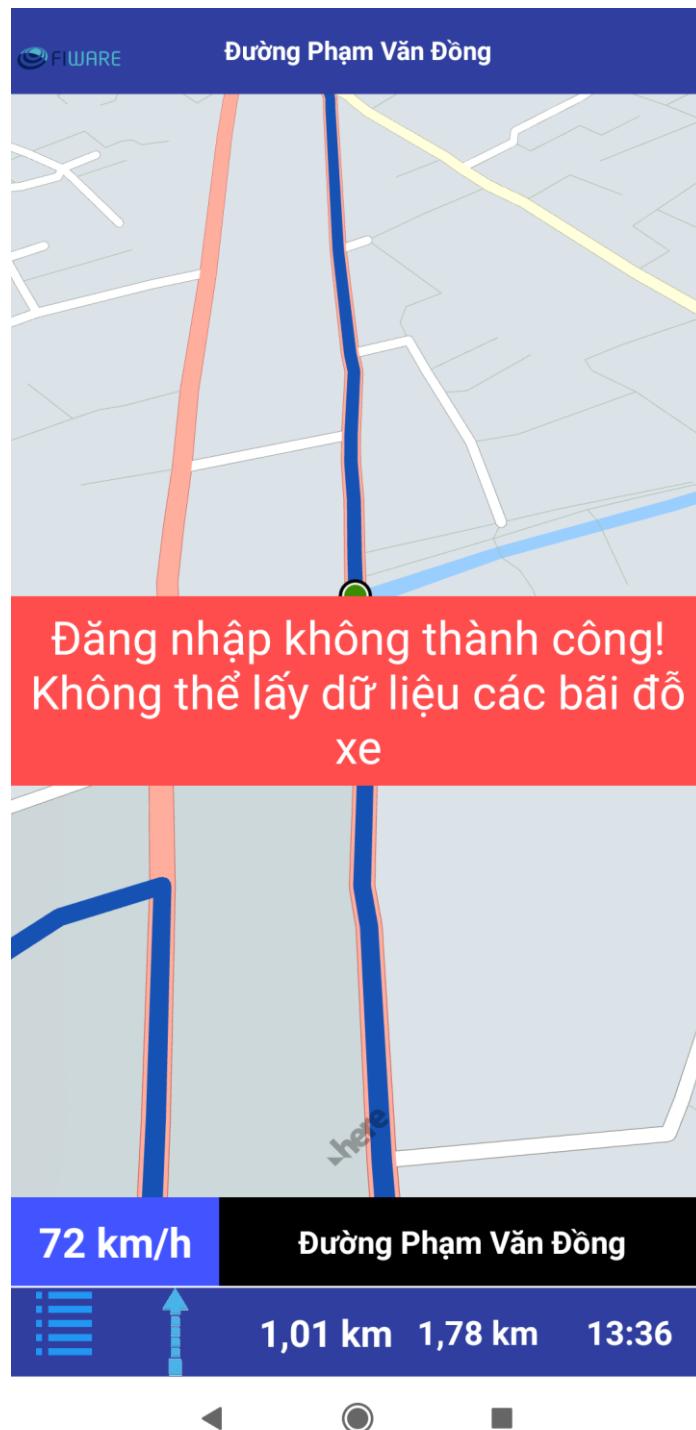
Hình 4.10 Chọn điểm xuất phát và điểm đến

Sau đó, thông tin về các tùy chọn đỗ xe đã được cài đặt trước được đưa ra cho người dùng, bao gồm: loại bãi đỗ xe: ngoài trời, khoảng cách tối đa từ bãi đỗ đến điểm đến: 500m, loại xe: Ô-tô. Hình 4.11 là màn hình ứng dụng thể hiện các thông tin đó. Nút bấm “Tìm đường” cho phép ta bắt đầu tìm kiếm hành trình trên bản đồ HERE Maps.



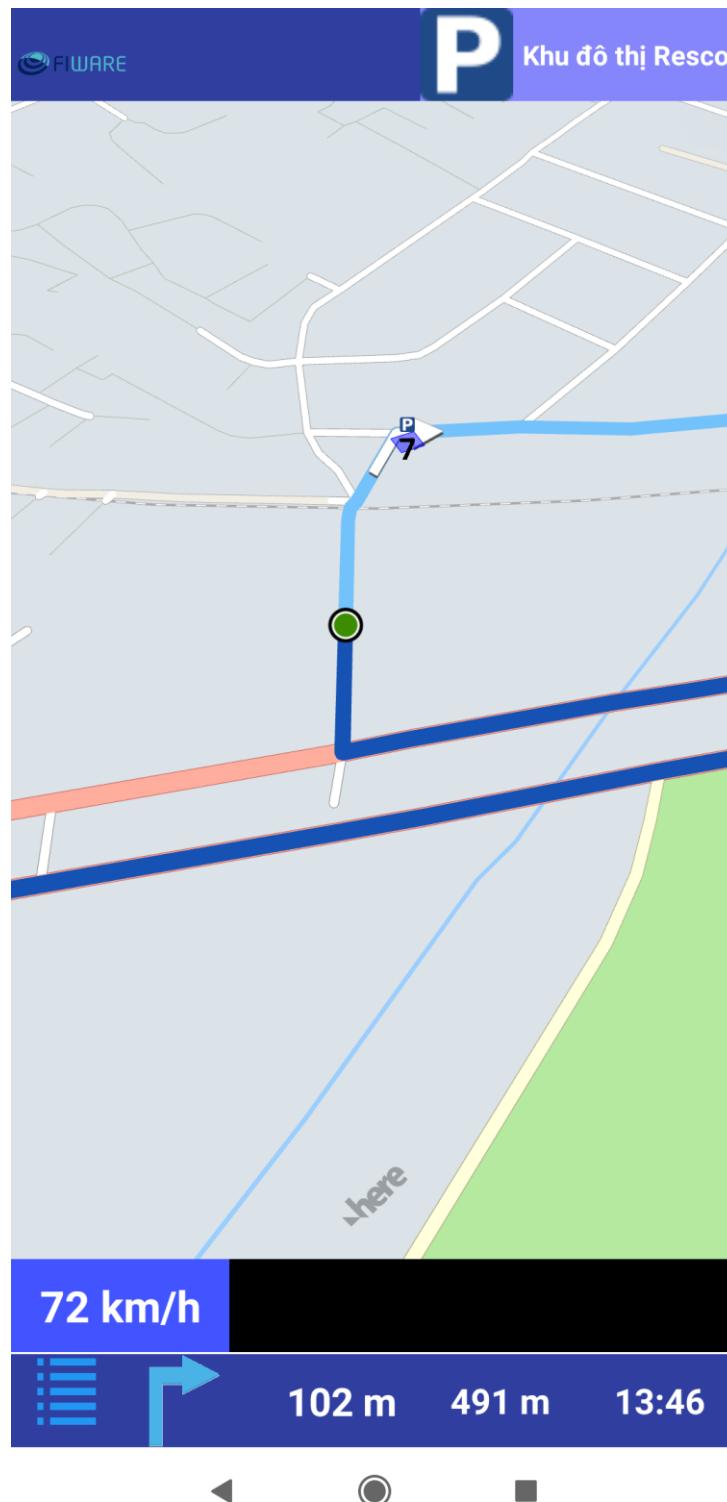
Hình 4.11 Thông tin về các tùy chọn đỗ xe

Khi bắt đầu hành trình, việc đầu tiên nó làm là đăng nhập vào hệ thống dịch vụ cung cấp dữ liệu ngữ nghĩa, sử dụng tài khoản được cung cấp bởi quản trị viên hệ thống (ở đây, tài khoản của ứng dụng chính là *hmi_lab_user* được đề cập ở chương 3). Nếu đăng nhập không thành công, một thông báo sẽ hiện lên cho biết các tính năng liên quan đến bãi đỗ xe thông minh hiện không khả dụng (Hình 4.12 Tính năng bãi đỗ xe không khả dụng khi đăng nhập không thành công)



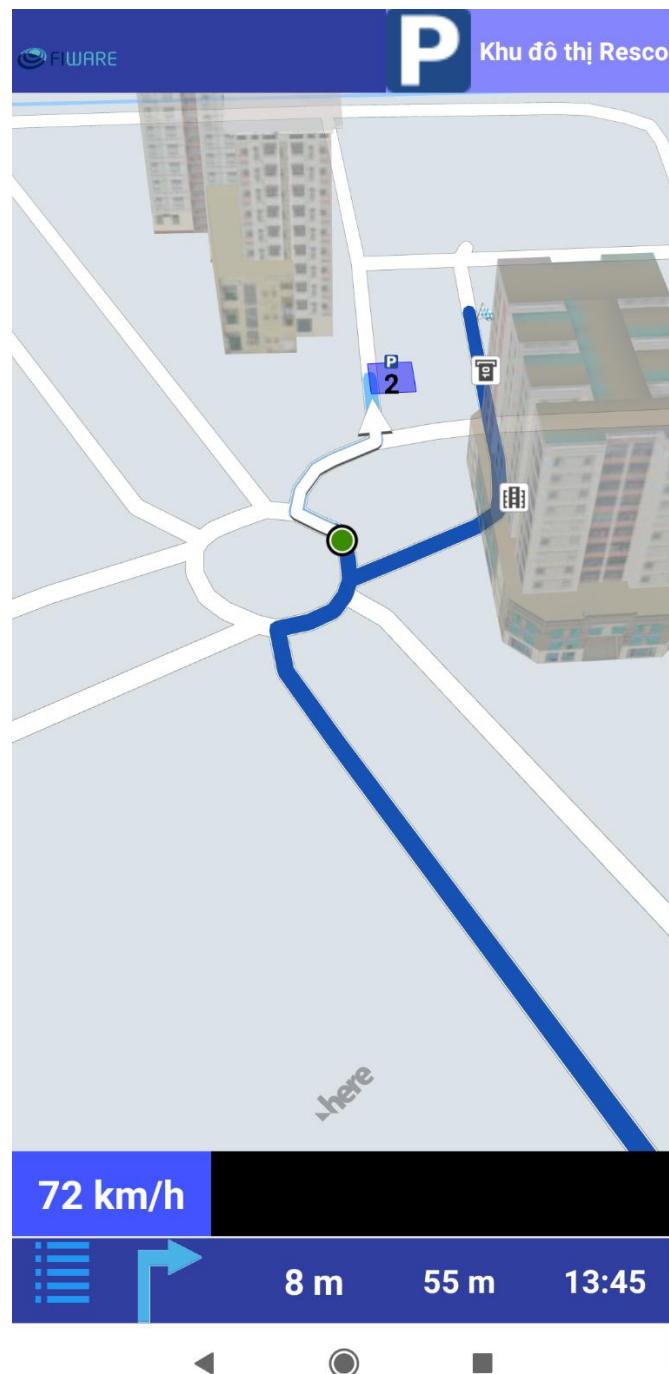
Hình 4.12 Tính năng bãi đỗ xe không khả dụng khi đăng nhập không thành công

Trong quá trình di chuyển, thông tin trực tuyến của các bãi đỗ xe được hiển thị trên bản đồ bằng các ô màu xanh và thông tin về số lượng chỗ đỗ xe còn trống trong bãi, như hình 4.13 thể hiện bãi đỗ xe tại tòa nhà OCT3-A còn thừa 7 chỗ trống.



Hình 4.13 Thông tin các bãi đỗ xe còn chỗ trống đọc đường đi

Khi xe ô-tô còn cách điểm đến 500m, thuật toán tìm kiếm và chỉ đường đến bãi đỗ xe còn chỗ trống gần nhất được khởi chạy. Khi tìm thấy bãi đỗ xe phù hợp, bản đồ sẽ tạo một chỉ dẫn đường đi mới tới bãi đỗ xe cho tài xế. Nếu không có bãi đỗ xe nào phù hợp, chỉ dẫn cũ được tiếp tục thực hiện. Hành trình kết thúc khi xe đến bãi đỗ hoặc đến điểm đến trong trường hợp không tìm được bãi đỗ xe phù hợp. Hình 4.14 cho thấy một bãi đỗ tại khu đô thị Resco còn dư 2 chỗ trống, nên ứng dụng chuyển hướng chỉ đường đến bãi đỗ đó. Hành trình kết thúc khi xe ô-tô đến bãi đỗ xe thay vì địa điểm đã chọn khi bắt đầu hành trình.



Hình 4.14 Chỉ đường đến bãi đỗ gần điểm đến nhất và kết thúc hành trình

4.5. Đánh giá hệ thống

Đầu tiên, hệ thống CamNet đảm bảo việc bảo mật thông tin người dùng ở cả 2 phía: phía camera cung cấp dữ liệu ngữ nghĩa và phía ứng dụng sử dụng dữ liệu ngữ nghĩa. Ở phía hệ thống camera và máy chủ cạnh, dữ liệu ảnh được xử lý để trích rút thông tin dưới dạng dữ liệu JSON, sau đó toàn bộ ảnh thu nhận được đều được xóa ngay tại máy chủ cạnh. Hơn nữa, khi đẩy dữ liệu lên máy chủ quản lý thông tin ngữ nghĩa, máy chủ cạnh cũng cần đăng nhập với tài khoản hợp lệ được cấp quyền “producer”, việc này đảm bảo người sử dụng máy chủ cạnh cũng không thể nào lấy được dữ liệu trên hệ thống. Ở phía ứng dụng, để tìm kiếm thông tin ngữ nghĩa cần thiết, nó phải đăng nhập vào hệ thống với tài khoản được cấp quyền “consumer”. Việc này đảm bảo chỉ những ứng dụng tin cậy được cấp quyền thì mới có thể sử dụng dữ liệu từ hệ thống, và ứng dụng cũng không có quyền ghi ngược lại dữ liệu vào hệ thống. Ngoài ra, tất cả các giao tiếp với phân hệ quản lý thông tin ngữ nghĩa (Orion Context Broker) đều đi qua lớp trung gian là Wilma PEP proxy, nên địa chỉ thực của Orion Context Broker là hoàn toàn ẩn đồi với tất cả các bên liên quan.

Thuật toán đếm số chỗ trống trong bãi đỗ xe được đánh giá với dữ liệu từ một bãi đỗ xe ngoài trời của khu đô thị Handiresco, Cổ Nhuế 2, Bắc Từ Liêm, Hà Nội (đã đề cập ở mục 4.2, hình 4.1). Bãi đỗ xe nằm trong khu vực có lưu lượng giao thông thấp, gồm có 8 ô đỗ xe nằm trên một hàng ngang và một camera giám sát được đặt cố định, vuông góc với bãi đỗ. Camera này là camera góc rộng 120 độ, ghi nhận video với độ phân giải FullHD (1920x1080) chuẩn H.264. Thuật toán được đánh giá trên bộ dữ liệu 500 hình ảnh được thu nhận ngẫu nhiên vào các thời điểm khác nhau, gồm có 400 hình ảnh với điều kiện thời tiết tốt (trời nắng hoặc mây mù), ảnh đủ sáng, không có vật cản che khuất hoặc có vật cản che một phần nhỏ các xe ô-tô và 100 hình ảnh với các điều kiện phức tạp: trời mưa, trời tối, có xe ô-tô đang lưu thông trên đường hoặc vật cản che khuất phần lớn vị trí ô đỗ xe. Kết quả đánh giá trên tập 400 hình ảnh đủ ánh sáng, các xe ô-tô không bị che khuất nhiều, độ chính xác của thuật toán lên đến 97.7%. Hình 4.15 cho thấy ngay cả khi có ô-tô đỗ trên đường, che khuất một phần hình ảnh thì thuật toán vẫn đưa ra kết quả chính xác. Tuy nhiên, khi đánh giá trên toàn bộ tập dữ liệu 500 hình ảnh nêu trên, độ chính xác của thuật toán giảm xuống còn 82.2%. Framework Mask-RCNN không phát hiện được vị trí xe ô-tô trong hầu hết các trường hợp trời tối, trời mưa, ảnh không đủ ánh sáng, hoặc khi có vật thể che khuất phần lớn xe ô-tô. Điều đó làm cho việc tính toán số ô trống trong bãi đỗ có kết quả sai lệch. Việc vật thể khác che khuất xe ô-tô trong ảnh thường là do có phương tiện giao thông đi rất sát với vị trí xe ô-tô đang đỗ. Việc này xảy ra trong thời gian

ngắn, nên sau khi ô-tô không còn bị che khuất, thuật toán sẽ đưa ra được kết quả chính xác. Kết quả của thuật toán này là thấp hơn so với phương pháp được đề xuất trong nghiên cứu *Deep Learning for Decentralized Parking Lot Occupancy Detection* [16] đạt được độ chính xác 90% đối với bộ dữ liệu có các điều kiện thời tiết khác nhau là nắng, âm u, trời mưa.



Hình 4.15 Kết quả đếm số chỗ trống trong bãi đỗ xe khi có vật cản che khuất một phần hình ảnh

Phân hệ thu nhận và trích rút thông tin ngữ nghĩa từ camera với thuật toán đếm số lượng chỗ đỗ xe trống trong bãi đỗ được cài đặt thử nghiệm trên một máy tính với hệ điều hành Ubuntu 18.04 LTS, CPU Core i7 2.9GHz, 16GB RAM, 512GB SSD. Máy tính và camera giám sát được kết nối vào cùng một mạng Wifi trên băng tần 5GHz. Để đánh giá hiệu quả xử lý của phân hệ này, tác giả thực hiện việc thu nhận liên tục 100 hình ảnh từ camera giám sát bằng FFMPEG và xử lý hình ảnh đó bằng thuật toán đề xuất ở mục 4.2. Tốc độ xử lý dữ liệu được thể hiện ở bảng 4.1 dưới đây.

Tiêu chí	Tốt nhất (milli giây)	Xấu nhất (milli giây)	Trung bình (milli giây)
Thu nhận hình ảnh bằng FFMPEG	1816	4202	2077
Đếm số lượng chỗ đỗ xe còn trống	9197	12922	9916

Bảng 4.1 Tốc độ xử lý dữ liệu của phân hệ thu nhận và trích rút thông tin ngữ nghĩa

Từ bảng kết quả có thể nhận thấy, thời gian xử lý trung bình rất sát với thời gian xử lý tốt nhất nhận được, chứng tỏ hiệu năng của phân hệ là ổn định. Tổng thời gian xử lý trung bình cho cả 2 công việc là thu nhận hình ảnh và trích rút thông tin là khoảng 12 giây, hoàn toàn đủ khả năng đáp ứng yêu cầu tìm kiếm thông tin thời gian thực cho bài toán tìm chỗ đỗ xe.

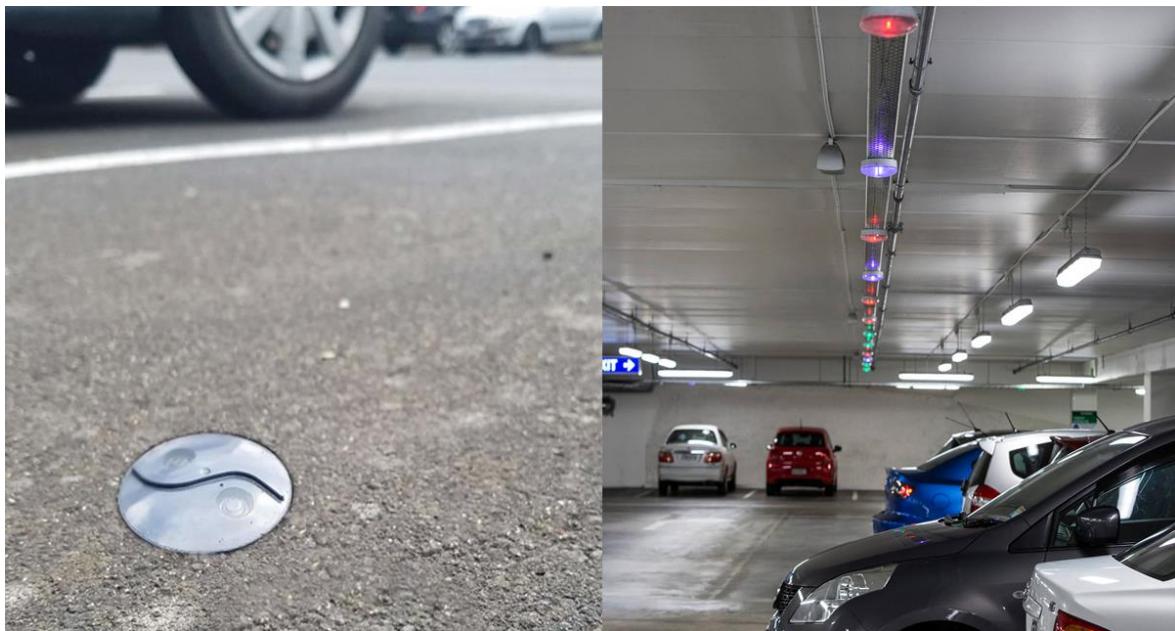
Đối với phân hệ quản lý thông tin ngữ nghĩa (Orion Context Broker) và phân hệ quản lý định danh và quyền người dùng (Keyrock + Wilma), chúng được cài đặt thử nghiệm trên một máy chủ Ubuntu 18.04 cung cấp bởi Digital Ocean, với 1 CPU 2.30GHz, 1GB RAM, 25GB HDD. Để đánh giá khả năng hoạt động của các phân hệ này, tác giả giả định toàn bộ hệ thống được kết nối với 100 camera IP, từ đó, Orion Context Broker sẽ phải quản lý 100 thực thể. Sau đó, ta đánh giá hiệu năng của phân hệ này với 2 tiêu chí: thời gian tạo mới thực thể và thời gian cập nhật dữ liệu cho 1 thực thể, bằng cách thực hiện liên tục từng tiêu chí trên 100 thực thể. Thời gian tìm kiếm thông tin dựa vào vị trí địa lý của xe hơi cũng được đánh giá bằng cách thực hiện tìm kiếm 100 lần với các số liệu vị trí và bán kính tìm kiếm khác nhau. Số liệu ghi nhận được thể hiện ở bảng 4.2 dưới đây.

Tiêu chí	Tốt nhất (milli giây)	Xấu nhất (milli giây)	Trung bình (milli giây)	Tổng thời gian (milli giây)
Thêm mới thực thể	102	409	115	11515
Cập nhật dữ liệu của thực thể	102	161	111	11130
Tìm kiếm chỗ đỗ xe	178	285	219	21998

Bảng 4.2 Tốc độ truy vấn thông tin ngữ nghĩa

Từ dữ liệu trên, ta thấy tốc độ thêm mới/cập nhật dữ liệu ngữ nghĩa của hệ thống là rất nhanh (vài trăm mili giây). Một số trường hợp có tốc độ thêm mới thực thể chưa tốt, nhưng là chấp nhận được (409 mili giây), và trong thực tế, thực thể chỉ cần được thêm vào 1 lần, sau đó, ta chỉ cập nhật thông tin của thực thể đó. Tốc độ tìm kiếm thông tin bãi đỗ xe dựa trên vị trí và bán kính tìm kiếm cũng rất nhanh (trung bình mất 219 mili giây), hoàn toàn đáp ứng được yêu cầu tìm kiếm bãi đỗ xe theo vị trí thực của ô-tô.

Trên phương diện khả năng ứng dụng thực tiễn, các hệ thống bãi đỗ xe hiện có thường phức tạp và phải đầu tư lớn về cơ sở hạ tầng, thiết bị, ví dụ như giải pháp Smart Parking[24] của công ty Hiotron phải dùng một hệ thống các cảm biến phục vụ cho việc phát hiện chỗ đỗ xe có trống hay không. Các cảm biến này được thiết kế riêng bởi công ty và yêu cầu phải được lắp đặt theo hệ thống phía trên vị trí đỗ xe hoặc phía dưới gầm xe (Hình 4.16 Hệ thống cảm biến phía trên hoặc phía dưới vị trí đỗ xe)



Hình 4.16 Hệ thống cảm biến phía trên hoặc phía dưới vị trí đỗ xe

Bảng 4.3 dưới đây cho thấy sự so sánh giữa ứng dụng tìm bãi đỗ xe áp dụng hệ thống CamNet và giải pháp Smart Parking của công ty Hiotron.

Tiêu chí	Hệ thống CamNet	Hiotron Smart Parking
Chi phí lắp đặt	Thấp. Sử dụng hệ thống camera hiện có của bãi đỗ xe. Sử dụng máy tính hiện có làm máy chủ cạnh hoặc sử dụng 1 mini-pc chi phí thấp.	Cao. Cần lắp đặt cảm biến riêng biệt tại từng vị trí đỗ xe. Cần xây dựng nền tảng điện toán đám mây đủ mạnh để xử lý thông tin thu nhận được.
Chi phí bảo trì	Chỉ cung cấp giải pháp phần mềm (camera và máy chủ cạnh có thể được cung cấp bởi các đối tác khác nhau) nên chi phí bảo trì ít hơn.	Cung cấp cả phần cứng (cảm biến) và phần mềm, nên phải bảo trì toàn bộ.
Chi phí quản lý	Tương đương	Tương đương
Tính năng tìm chỗ đỗ xe	Có	Có
Tính năng chỉ đường	Có	Có

Khả năng mở rộng phạm vi ứng dụng	Rất cao. Có thể cài đặt các thuật toán trích rút thông tin từ hình ảnh khác nhau vào hệ thống, như phát hiện tắc đường, phát hiện hỗn loạn... và tích hợp các cảnh báo phù hợp vào ứng dụng.	Thấp. Hệ thống chỉ phục vụ cho việc tìm bãi đỗ xe.
-----------------------------------	--	--

Bảng 4.3 So sánh hệ thống đề xuất với giải pháp hiện có

Ứng dụng tích hợp hệ thống CamNet vào bản đồ HERE Maps cho thấy sự vượt bậc về chi phí lắp đặt, bảo trì hệ thống dựa vào việc tận dụng các hệ thống camera và máy tính hiện có của các bãi đỗ xe, trong khi các tính năng được cung cấp là tương tự. Ứng dụng mang tính thực tiễn cao, giải quyết được một trong những nhu cầu cần thiết của xã hội, tận dụng được lượng dữ liệu khổng lồ từ các hệ thống camera sẵn có. Ngoài ra, hệ thống CamNet có thể mở rộng thêm nhiều loại dữ liệu ngữ nghĩa để phát triển ứng dụng trên các lĩnh vực khác nhau.

KẾT LUẬN

Để làm cho các hệ thống camera giám sát trở nên “thông minh” hơn và tận dụng triệt để các thông tin mà chúng ghi nhận được, lượng dữ liệu video từ các camera này phải được thu nhận, sau đó phân tích để trích xuất các thông tin hữu ích cho mục đích đưa ra các quyết định và phản hồi thông minh. Trong luận văn này, tác giả đã đề xuất một kiến trúc để thu nhận, trích rút, quản lý thông tin ngữ nghĩa từ video dựa trên ý tưởng kết hợp điện toán cạnh và điện toán đám mây với tên gọi là “hệ thống CamNet”. Một máy chủ cạnh được kết nối với hệ thống camera giám sát để thu nhận các luồng video và trích rút thông tin có nghĩa từ chúng bằng cách sử dụng thư viện FFMPEG và một thuật toán đếm số lượng chỗ đỗ xe trống trong bãi đỗ dựa trên kiến trúc Mask RCNN. Sau đó, dữ liệu này được đẩy lên nền tảng điện toán đám mây lưu trữ, quản lý thông tin ngữ nghĩa, được xây dựng dựa trên các thư viện mã nguồn mở của nền tảng FIWARE là Orion Context Broker, Keyrock, và Wilma PEP Proxy. Với phương pháp này, thông tin từ các hệ thống camera giám sát có thể được cung cấp như một dịch vụ, các ứng dụng thông minh đa nền tảng có thể sử dụng tài khoản với các quyền phù hợp để tìm kiếm thông tin trong hệ thống cho các mục đích sử dụng khác nhau. Tiếp theo, tác giả đã phát triển một ứng dụng thông minh tích hợp hệ thống CamNet vào bản đồ HERE Maps cho Android để chỉ đường cho tài xế đến bãi đỗ xe còn trống gần nhất. Kết quả cho thấy ứng dụng chỉ đường đã sử dụng tốt dữ liệu bãi đỗ xe từ hệ thống CamNet, qua đó giải quyết được vấn đề tìm kiếm bãi đỗ xe, là một trong những vấn đề cấp thiết của xã hội. Từ đó chứng minh tính khả thi và khả năng ứng dụng thực tiễn rất cao của hệ thống CamNet.

Ứng dụng thử nghiệm cho thấy dữ liệu quản lý bởi hệ thống CamNet có tiềm năng ứng dụng rất cao, tuy nhiên độ chính xác của thuật toán phát hiện ô đỗ xe còn trống chưa ổn định với các điều kiện ngoại cảnh khác nhau. Vì vậy, đầu tiên tác giả muốn phát triển thêm và đánh giá chi tiết hơn độ chính xác của thuật toán phát hiện ô đỗ xe còn trống. Tiếp theo là phát triển các thuật toán trích rút thông tin khác như phát hiện đám đông, tắc đường, ngập lụt... Sau đó, cũng rất cần thiết phải phát triển một giao diện cho phép quản lý trạng thái của mạng lưới các camera từ xa, từ đó có thể phát hiện lỗi phát sinh và khắc phục lỗi. Việc mở rộng phạm vi ứng dụng tương đồng với việc số lượng cảm biến và số lượng thuật toán tăng cao, đòi hỏi khả năng tính toán rất lớn. Điều này thúc đẩy việc nghiên cứu tích hợp các phương pháp xử lý phân tán vào hệ thống cũng cần được quan tâm phát triển.

TÀI LIỆU THAM KHẢO

- [1] Website: <https://www.icrealtime.com/>, ngày truy cập: 12/8/2020.
- [2] Website: <https://www.mi.com/vn>, ngày truy cập: 12/8/2020.
- [3] Website: <http://hikvision.vn/>, ngày truy cập: 12/8/2020.
- [4] Mayank Patel, Minal Bhise, “Raw Data Processing Framework for IoT”, in *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*
- [5] F. Cirillo, G. Solmaz, E. L. Berz, M. Bauer, B. Cheng and E. Kovacs, “A Standard-Based Open Source IoT Platform: FIWARE”, in *IEEE Internet of Things Magazine*, vol. 2, no. 3, pp. 12-18, September 2019, doi: 10.1109/IOTM.0001.1800022.
- [6] Website: <https://wiki.openstack.org/wiki/Horizon>, ngày truy cập: 12/8/2020.
- [7] Website: <https://docs.openstack.org/keystone/latest/>, ngày truy cập: 12/8/2020.
- [8] Website: <https://docs.djangoproject.com/en/3.0/>, ngày truy cập: 12/8/2020.
- [9] P.Mell and T. Grance, “The nist definition of cloud computing”, in *National Institute of Standards and Technology*, vol. 53, no. 6, article 50, 2009.
- [10] P. Voigt, A. von dem Bussche, "The EU general data protection regulation (GDPR)", in *A Practical Guide*, Cham, Switzerland:Springer, 2017.
- [11] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, Lanyu Xu, “Edge Computing: Vision and Challenges”, in *IEEE Internet of Things Journal*, volume: 3, issue: 5, Oct. 2016.
- [12] Website: <https://ffmpeg.org/>, ngày truy cập: 12/8/2020.
- [13] Website: <https://docs.docker.com/engine/>, ngày truy cập: 12/8/2020.
- [14] Website: <https://docs.docker.com/compose/>, ngày truy cập: 12/8/2020.
- [15] Website: <https://developer.here.com/products/here-sdk>, ngày truy cập: 12/8/2020
- [16] Giuseppe Amato, Fabio Carrara, Fabrizio Falchi, Claudio Gennaro, Carlo Meghini, Claudio Vairo, “Deep Learning for Decentralized Parking Lot Occupancy Detection”, in *Expert Systems with Applications*, 2016.
- [17] J.L. Hernández Ramos, A. Skarmeta, “Security and Privacy in the Internet of Things: Challenges and Solutions”, pp. 9-15, March 2020.
- [18] José Manuel Cantera Fonseca (FIWARE Foundation), Fermín Galán Márquez (Telefónica España), Tobias Jacobs (NEC), “FIWARE-NGSI v2 Specification”, 2018.
- [19] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick, “Mask R-CNN”, in *arXiv:1703.06870v3*, 2018.

- [20] Zhaoxiu Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, Dongwei Ren, “Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression”, in *arXiv:1911.08287v1*, 2019.
- [21] Waleed Abdulla, “Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow”, 2018.
- [22] IoT based Smart Parking Systems for Smart Cities, in website: <https://www.hiotron.com/smart-parking/>, ngày truy cập: 12/8/2020.
- [23] P. Widya, Y. Yustiawan, and J. Kwon, “A oneM2M-Based Query Engine for Internet of Things (IoT) Data Streams”, in *Sensors*, vol. 18, no. 10, p. 3253, Sep. 2018.
- [24] Huiju Wang, Xiongpai Qin, Xuan Zhou, Furong Li, “Efficient query processing framework for big data warehouse: an almost join-free approach”, in *Frontiers of Computer Science*, April 2015.
- [25] Mbazaia O., Kamoun K., “CaRT: Framework for Semantic Query Correction and Relaxation”, in *Digital Economy. Emerging Technologies and Business Innovation*, ICDEc 2019.
- [26] Calavia, L., Baladrón, C., Aguiar, J. M., Carro, B. and Sánchez-Esguevillas, A, “A semantic autonomous video surveillance system for dense camera networks in smart cities”, in *Sensors 12(8)*, pp. 10407–10429. 2012.
- [27] Xiong, Y.-H., Wan, S.-Y., He, Y. and Su, D., “Design and implementation of a prototype cloud video surveillance system”, in *Journal of Advanced Computational Intelligence and Intelligent Informatics*, pp. 40–47, 2014.
- [28] Igor Ivan, Alex Singleton, Jiří Horák, Tomáš Inspektor, “The Rise of Big Spatial Data”, in *Lecture Notes in Geoinformation and Cartography*, pp 41-48, 2017.
- [29] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár, “Microsoft COCO: Common Objects in Context”, in *arXiv:1405.0312v3*, 2015.
- [30] Luis Valentín, Sergio A. Serrano, Reinier Oves García, Anibal Andrade, Miguel A. Palacios-Alonso, L. Enrique Sucar, “A Cloud-Based Architecture For Smart Video Surveillance”, in *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume XLII-4/W3, 2017.
- [31] Website: <https://www.confluent.io/>, ngày truy cập: 18/9/2020.
- [32] Website: <https://cloud.google.com/iot-core>, ngày truy cập: 18/9/2020.

Hà Nội, ngày.....tháng.....năm 2020

BẢN XÁC NHẬN ĐÃ SỬA CHỮA CÁC THIẾU SÓT CỦA LUẬN VĂN

Trường Đại học Công nghệ đã có Quyết định số 569/QĐ-ĐT ngày 28 tháng 8 năm 2020 về việc thành lập Hội đồng chấm luận văn Thạc sĩ cho học viên Lê Xuân Tùng, sinh ngày 31/08/1994, tại Vĩnh Phúc, chuyên ngành Khoa học máy tính, ngành Khoa học máy tính.

Ngày 05 tháng 09 năm 2020, Trường Đại học Công nghệ (ĐHCN) đã tổ chức cho học viên bảo vệ luận văn Thạc sĩ trước Hội đồng chấm (có biên bản kèm theo). Theo Quyết định của Hội đồng chấm luận văn Thạc sĩ, học viên phải bổ sung và sửa chữa các điểm sau đây trước khi nộp quyền luận văn cuối cùng cho Nhà trường để hoàn thiện hồ sơ sau bảo vệ:

1. Bổ sung so sánh nền tảng được sử dụng (FIWARE) với các nền tảng khác, lý do chọn FIWARE (thêm vào mục 3.1).
2. Bổ sung đánh giá kết quả thực nghiệm và so sánh với phương pháp khác (thêm vào mục 4.5).
3. Làm rõ những nội dung có sẵn và nội dung do tác giả tự thực hiện trong hệ thống CamNet (thêm vào mục 3.1).
4. Sửa một số lỗi trình bày, chính tả.

Ngày.....tháng.....năm....., học viên đã nộp bản luận văn có chỉnh sửa. Chúng tôi nhận thấy rằng nội dung, hình thức của luận văn và tóm tắt luận văn đã được sửa chữa, bổ sung theo các điểm trên của Quyết nghị.

Đề nghị Trường Đại học Công nghệ, ĐHQGHN cho phép học viên được làm các thủ tục khác để được công nhận và cấp bằng Thạc sĩ.

Xin trân trọng cảm ơn!

XÁC NHẬN CỦA THÀNH VIÊN HỘI ĐỒNG/HỘI ĐỒNG
ĐỀ NGHỊ HỌC VIÊN SỬA CHỮA LUẬN VĂN

HỌC VIÊN

CÁN BỘ HƯỚNG DẪN

XÁC NHẬN CỦA CƠ SỞ ĐÀO TẠO

QUYẾT NGHỊ
CỦA HỘI ĐỒNG CHẤM LUẬN VĂN THẠC SĨ

Căn cứ Quyết định số 569/QĐ-DT ngày 28/8/2020 của Hiệu trưởng trường Đại học Công nghệ về việc thành lập Hội đồng chấm luận văn thạc sĩ của học viên Lê Xuân Tùng, Hội đồng chấm luận văn Thạc sĩ đã họp vào hồi 11; 05 ngày 05 tháng 09 năm 2020 tại 3.42-E5...ĐH Công nghệ, ĐHQGHN

Tên đề tài luận văn: Xây dựng hệ thống thu thập và quản lý thông tin từ camera giám sát dựa trên nền tảng FIWAVE

Ngành: Khoa học máy tính

Chuyên ngành: Khoa học máy tính

Mã số: 8480101.01

Sau khi nghe học viên trình bày tóm tắt luận văn Thạc sĩ, các phản biện đọc nhận xét, học viên trả lời các câu hỏi, Hội đồng đã họp, trao đổi ý kiến và thống nhất kết luận:

1. Về tính cấp thiết, tính thời sự, ý nghĩa lý luận và thực tiễn của đề tài luận văn:

Xây dựng ứng dụng sử dụng camera để
tính yêu cầu sử dụng nhanh chóng, chính
xác hóa mặt có tính thực tiễn cao

2. Về bối cảnh, phương pháp nghiên cứu, tài liệu tham khảo, của luận văn:

Làm văn bản bối cảnh hấp hối trình bày rõ ràng
tài liệu tham khảo đầy đủ

3. Về kết quả nghiên cứu:

Kết quả thực nghiệm chỉ ra là tổng đài ứng
nguyên kín phát triển cho trong điều kiện

4. Hạn chế của luận văn (nếu có):

Chưa trình bày các giải pháp khác
Thực nghiệm chưa so sánh với p.p khác

5. Đánh giá chung và kết luận:

Chỉnh sửa theo góp ý của hội đồng

Đẹp tinh tế cao và làm văn học

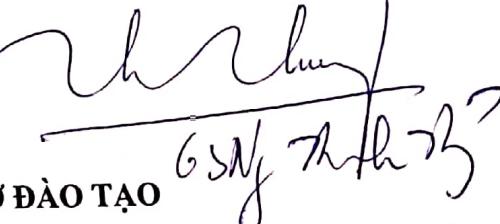
Luận văn đạt **8.0/10** điểm. Quyết nghị này được **GS. TS. Nguyễn Văn Thành** và **GS. TS. Nguyễn Thị Nhàn** thành viên của Hội đồng nhất trí thông qua.

THƯ KÝ HỘI ĐỒNG


Lê Ngọc Khoa

XÁC NHẬN CỦA CƠ SỞ ĐÀO TẠO

CHỦ TỊCH HỘI ĐỒNG


Trần Thị Nhàn
GS. TS. Nhàn

TL. HIỆU TRƯỞNG
TRƯỞNG PHÒNG ĐÀO TẠO

Nguyễn Phương Thái

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

BẢN NHẬN XÉT PHẢN BIỆN LUẬN VĂN THẠC SĨ

Họ và tên cán bộ phản biện: Nguyễn Thị Nhật Thanh

Học hàm, học vị: PGS. TS.

Chuyên ngành: Toán và khoa học máy tính

Cơ quan công tác: Trường Đại học Công nghệ, ĐHQGHN

Họ và tên học viên cao học: Lê Xuân Tùng

Tên đề tài luận văn: Xây dựng hệ thống thu thập và quản lý thông tin từ camera giám sát dựa trên nền tảng FIWARE

Chuyên ngành: Khoa học máy tính

Mã số: 8480101.01

Ý KIẾN NHẬN XÉT

Nghiên cứu và xây dựng các ứng dụng sử dụng camera giám sát đáp ứng yêu cầu sử dụng nhanh chóng, chính xác, và bảo mật là vấn đề có tính thực tiễn cao.

Trong luận văn này, học viên tìm hiểu nền tảng FIWARE được phát triển dựa trên mã nguồn mở, hỗ trợ kết hợp với các nền tảng thứ ba để phát triển các giải pháp ứng dụng thông minh. Học viên cũng phân tích và thiết kế hệ thống CAMNET hỗ trợ kết nối các camera giám sát bãi đỗ xe, cung cấp thông tin về vị trí trống, và điều hướng người sử dụng. Hệ thống được thử nghiệm tại 1 bãi đỗ xe ngoài trời của khu đô thị Handiresco, Bắc Từ Liêm, Hà Nội. Kết quả thử nghiệm cho thấy bước đầu ứng dụng có thể phát hiện chỗ trống trong bãi xe và điều hướng. Tốc độ xử lý dữ liệu và đáp ứng của hệ thống cũng được đánh giá và đáp ứng yêu cầu trong thực nghiệm này. Bên cạnh đó các vấn đề bảo mật và khả năng mở rộng mạng lưới camera được xem xét bước đầu.

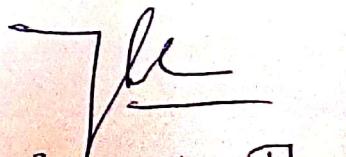
Luận văn có bố cục hợp lý, trình bày rõ ràng, tài liệu tham khảo đầy đủ.

Luận văn đáp ứng yêu cầu một luận văn thạc sĩ ngành Công nghệ thông tin. Tôi đồng ý cho học viên bảo vệ trước hội đồng.

Hà Nội, ngày 5 tháng 9 năm 2020

XÁC NHẬN CỦA CƠ QUAN CÔNG TÁC

CÁN BỘ PHẢN BIỆN


Nguyễn Thị Nhật Thanh

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh phúc

BẢN NHẬN XÉT PHẢN BIỆN LUẬN VĂN THẠC SĨ

Họ và tên cán bộ phản biện: Nguyễn Chí Thành

Học hàm, học vị: Tiến sĩ

Chuyên ngành: Công nghệ thông tin

Cơ quan công tác: Viện Công nghệ thông tin, Viện Khoa học và Công nghệ Quân sự

Họ và tên học viên cao học: Lê Xuân Tùng

Tên đề tài luận văn: Xây dựng hệ thống thu thập và quản lý thông tin từ camera giám sát dựa trên nền tảng FIWARE

Chuyên ngành: Khoa học máy tính

Mã số: 8480101.01

Ý KIẾN NHẬN XÉT

- Các thiết bị camera giám sát ngày càng trở nên phổ biến và được triển khai ở nhiều nơi, ứng dụng trong nhiều lĩnh vực khác nhau từ dân dụng đến kinh doanh, sản xuất, an ninh, quốc phòng. Tuy vậy các hệ thống camera hiện nay chủ yếu phục vụ mục đích lưu trữ, quan sát trực tiếp và xem lại khi cần, chưa có các chức năng khai thác thông tin thông minh từ hình ảnh camera và chưa có các hệ thống kết nối chia sẻ thông tin từ nhiều camera. Luận văn nghiên cứu phát triển một hệ thống cho phép thu nhận thông tin từ nhiều camera và phát triển các ứng dụng thông minh dựa trên dữ liệu thu được từ các camera. Việc nghiên cứu, làm chủ hệ thống này có ý nghĩa thực tiễn rất cao, nhất là trong lĩnh vực an ninh quốc phòng.
- Đề tài nghiên cứu không trùng lặp với các công trình khoa học, luận văn đã công bố ở trong và ngoài nước. Tài liệu tham khảo trích dẫn trung thực, rõ ràng.
- Tên đề tài phù hợp với nội dung nghiên cứu, chuyên ngành và mã số đào tạo.
- Tác giả đã tiến hành nghiên cứu các nền tảng cơ sở lý thuyết có liên quan đến đề tài, tổng quan về FIWARE, từ đó đề xuất, xây dựng hệ thống CamNet và tiến hành thử nghiệm, đánh giá. Phương pháp nghiên cứu phù hợp với chuyên ngành đào tạo.
- Luận văn đã đề xuất một kiến trúc để thu nhận, trích rút, quản lý thông tin từ camera và xây dựng hệ thống CamNet dựa trên kiến trúc đó. Dựa trên hệ thống CamNet, tác giả đã xây dựng ứng dụng tìm bối cảnh xe còn trống cho tài xế áp dụng công nghệ học sâu. Các kết quả nghiên cứu của luận văn là tài liệu tham khảo và là nền tảng để phát triển các hệ thống camera thông minh.
- Bố cục của luận văn tương đối khoa học, rõ ràng. Tuy nhiên nội dung luận văn còn một số nội dung cần bổ sung và chỉnh sửa như sau:



- + Chưa trình bày được các giải pháp khác có khả năng tương tự như FIWARE, chưa làm rõ tại sao chọn FIWARE.
- + Cần làm rõ hơn những nội dung đã có sẵn và các nội dung do tác giả tự thực hiện trong hệ thống CamNet.
- + Trong thực nghiệm không có so sánh với các phương pháp khác. Chưa có các đánh giá về độ chính xác của hệ thống.
- + Luận văn còn một số lỗi trình bày, chính tả.
- Tuy còn một số nội dung cần bổ sung, chỉnh sửa để nâng cao chất lượng của luận văn, nhưng về cơ bản luận văn đã giải quyết được mục đích và nhiệm vụ đề ra. Tôi đồng ý để tác giả Lê Xuân Tùng được bảo vệ luận văn trước Hội đồng chấm luận văn thạc sĩ.

Hà Nội, ngày 4 tháng 9 năm 2020

CÁN BỘ PHẢN BIỆN



Nguyễn Chí Thành

XÁC NHẬN CỦA CƠ QUAN CÔNG TÁC



Thượng tá Đỗ Việt Bình

