

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA ĐIỆN - ĐIỆN TỬ  
BỘ MÔN VIỄN THÔNG**



**ĐỀ CƯƠNG LUẬN VĂN TỐT NGHIỆP**

## **CÔNG NGHỆ UHF RFID**

**SINH VIÊN THỰC HIỆN:**

**NGUYỄN DUY LUÂN - 1511889**

**NGUYỄN VĂN BẢO SƠN - 1512852**

**GIẢNG VIÊN HƯỚNG DẪN:**

**TS. TRỊNH XUÂN DŨNG**

**Thành phố Hồ Chí Minh, tháng 12 năm 2018**

## LỜI CẢM ƠN

Sau gần 4 năm học tập và rèn luyện tại **Trường Đại học Bách Khoa TP.HCM**, chúng em đã tích lũy cho mình rất nhiều kiến thức cũng như kinh nghiệm sống để làm hành trang cho mình trước khi trở thành một kỹ sư trong tương lai. Bên cạnh những nỗ lực và cố gắng của bản thân mình, chúng em đã nhận được rất nhiều sự quan tâm và giúp đỡ của quý **Thầy Cô, Gia đình và Bạn bè**. Với lòng biết ơn sâu sắc nhất, chúng em xin chân thành gửi đến quý **Thầy Cô Trường Đại học Bách Khoa TP.HCM**, quý **Thầy Cô Khoa Điện - Điện Tử** - những người đã cùng với tri thức và tâm huyết truyền đạt vốn kiến thức quý báu của mình cho chúng em trong suốt thời gian học tập tại trường. Đặc biệt, chúng em xin gửi lời cảm ơn chân thành nhất đến Thầy hướng dẫn của mình - **TS. Trịnh Xuân Dũng** - người đã tận tình hướng dẫn và giúp đỡ chúng em trong quá trình thực hiện đề cương luận văn này.

Qua đây, chúng em cũng xin gửi đến gia đình và bạn bè của mình lời cảm ơn chân thành và đặc biệt là cha mẹ chúng em, người đã luôn ủng hộ và hỗ trợ để em có thể hoàn thành tốt đề cương luận văn này.

Trong quá trình thực hiện đồ án môn học chắc chắn không thể tránh khỏi những sai sót, do vậy chúng em rất mong nhận được những ý kiến đóng góp quý báu của quý Thầy Cô để chúng em học hỏi thêm nhiều kinh nghiệm và hoàn thiện đề cương luận văn của mình. Sau cùng, chúng em xin kính chúc quý **Thầy Cô Trường Đại học Bách Khoa TP.HCM** và **Thầy Trịnh Xuân Dũng** được dồi dào sức khỏe, đạt được nhiều thành công trong cuộc sống và luôn giữ vững ngọn lửa đam mê của mình trong việc truyền đạt kiến thức cho thế hệ mai sau.

Thành phố Hồ Chí Minh, tháng 12 năm 2018

Nhóm sinh viên thực hiện

## TÓM TẮT

IoT (Internet of Things) hay Mạng lưới vạn vật kết nối internet là một thuật ngữ không còn gì lạ lẫm trên thế giới, bởi lợi ích của nó trong tất cả các lĩnh vực của cuộc sống. Internet of things hiện nay có thể được hình dung là bao gồm nhiều mạng, thiết bị kết nối với nhau dựa trên hàng loạt thông số kỹ thuật và tiêu chuẩn. RFID có thể trở thành cầu nối bằng cách cung cấp dữ liệu xác định đối tượng cụ thể tại một địa điểm cụ thể và thời gian chính xác. Thẻ tag đảm bảo thiết bị có một định danh để có thể được nhận dạng bởi các đặc tính duy nhất. Đó cũng chính là mức độ giao tiếp cơ bản đặc trưng của RFID và điều này hoàn toàn phù hợp với những gì mà IoT đòi hỏi.

Trong đó, Công nghệ Siêu cao tần thụ động (UHF – ultra high frequency) là một trong những công nghệ đang nổi lên như là những tiêu chuẩn có khả năng được ứng dụng rộng rãi trong thế giới IoT. Trong báo cáo này, chúng ta sẽ nghiên cứu sâu vào Công nghệ UHF RFID và Ứng dụng của nó trong cuộc sống.

# Mục lục

Lời cảm ơn	i
Tóm tắt	ii
Mục lục	iii
Danh sách hình vẽ	vi
Danh sách bảng	viii
Danh mục từ viết tắt	ix
<b>1 Giới thiệu tổng quan đề tài</b>	<b>1</b>
1.1 Ý tưởng nghiên cứu, phạm vi nghiên cứu và đối tượng nghiên cứu . . . . .	1
1.1.1 Ý tưởng nghiên cứu . . . . .	1
1.1.2 Phạm vi nghiên cứu . . . . .	2
1.1.3 Đối tượng nghiên cứu . . . . .	2
1.2 Tổng quan về kiến trúc của hệ thống . . . . .	2
<b>2 Cơ sở lý thuyết</b>	<b>4</b>
2.1 Công nghệ RFID UHF . . . . .	4
2.1.1 Công nghệ RFID là gì? . . . . .	4
2.1.2 Lịch sử phát triển của công nghệ RFID . . . . .	5
2.1.2.1 Thời kỳ đầu của RFID . . . . .	5

2.1.2.2	Phát hiện các vật thể riêng biệt . . . . .	6
2.1.2.3	RFID phát triển trên toàn cầu . . . . .	8
2.1.3	Phân loại hệ thống RFID . . . . .	9
2.1.3.1	Các dải tần số RFID . . . . .	9
2.1.3.2	Các hệ thống chủ động, thụ động và BAP RFID . . . . .	11
2.2	Cơ sở dữ liệu MongoDB . . . . .	12
2.2.1	NoSql là gì? . . . . .	12
2.2.2	MongoDB là gì? . . . . .	13
2.2.3	Ưu điểm của MongoDB . . . . .	14
2.2.4	Nhược điểm . . . . .	14
<b>3</b>	<b>Thiết kế và thực hiện phần cứng</b>	<b>16</b>
3.1	Thiết kế kiến trúc phần cứng . . . . .	16
3.1.1	Các thành phần . . . . .	16
3.1.2	Yêu cầu thiết kế . . . . .	16
3.2	Thực hiện phần cứng . . . . .	16
<b>4</b>	<b>Thiết kế và thực hiện phần mềm</b>	<b>17</b>
4.1	Thiết kế kiến trúc phần mềm . . . . .	17
4.1.1	Các thành phần . . . . .	17
4.1.2	Yêu cầu thiết kế . . . . .	17
4.2	Thực hiện phần mềm . . . . .	17
4.2.1	Cài đặt và cấu hình MongoDB . . . . .	17
4.2.1.1	Cài đặt MongoDB . . . . .	17
4.2.1.2	Cấu hình MongoDB . . . . .	18
4.2.2	Tạo Server truy xuất MongoDB . . . . .	20
4.2.2.1	Cài đặt NodeJS và Tạo Project . . . . .	20
4.2.2.2	Cài đặt module mongoose . . . . .	20
4.2.2.3	Cài đặt module express . . . . .	20

4.2.2.4	Tạo web server dùng express . . . . .	22
4.2.2.5	Tạo file Controller, Model và Routes . . . . .	23
4.2.3	Tạo ứng dụng web hiển thị trực quan MongoDB . . . . .	27
<b>5</b>	<b>Kết quả vận hành thực tế</b>	<b>36</b>
<b>6</b>	<b>Đánh giá và Hướng phát triển</b>	<b>38</b>
6.1	Đánh giá . . . . .	38
6.1.1	Ưu điểm . . . . .	38
6.1.2	Khuyết điểm . . . . .	38
6.2	Hướng phát triển . . . . .	38
	<b>Tài liệu tham khảo</b>	<b>38</b>

# Danh sách hình vẽ

1.1	Kiến trúc cơ bản của một hệ thống RFID. . . . .	1
1.2	Tổng quan về kiến trúc của hệ thống. . . . .	3
2.1	Thiết bị IFF (bên trái), thiết bị RFID (tích cực) hiện đại ngày nay. . . . .	6
2.2	Các mốc thời gian quan trọng trong giai đoạn đầu của RFID. . . . .	6
2.3	Các phương pháp điều khiển truy cập thông thường và điều khiển truy cập RFID. . . . .	7
2.4	Những mốc thời gian quan trọng từ năm 1960 đến 1990. . . . .	8
2.5	Những mốc thời gian quan trọng từ năm 1990 đến nay. . . . .	9
2.6	Phân loại hệ thống RFID theo tần số. . . . .	10
2.7	So sánh UHF và LF, HF. . . . .	11
2.8	So sánh các hệ thống chủ động, thụ động và BAP RFID. . . . .	13
2.9	CSDL MongoDB. . . . .	13
2.10	So sánh tốc độ ghi của MongoDB và MySQL. . . . .	15
4.1	Hướng dẫn cài đặt MongoDB trên Ubuntu. . . . .	18
4.2	Khởi chạy MongoDB. . . . .	18
4.3	Truy xuất MongoDB. . . . .	19
4.4	Xem tất cả các database MongoDB. . . . .	19
4.5	Tạo project nodejs. . . . .	20
4.6	Mô hình mongoose, nodejs, mongodb. . . . .	21
4.7	Cài đặt module mongoose. . . . .	21
4.8	Cài đặt module express. . . . .	22

4.9	Khởi chạy ứng dụng NodeJS. . . . .	27
4.10	File package.json. . . . .	28
4.11	Chạy ứng dụng web. . . . .	35
5.1	Giao diện web tại địa chỉ <b>http://localhost:3000/</b> . . . . .	36
5.2	Xóa trực tiếp document. . . . .	37



# Danh sách bảng

2.1	Các ứng dụng tiêu biểu dùng công nghệ RFID LF và HF . . . . .	9
-----	---	---

# Danh mục từ viết tắt

RFID Radio Frequency IDentification

LF Low Frequency

HF High Frequency

UHF Ultra-High Frequency

CSDL Cơ sở dữ liệu

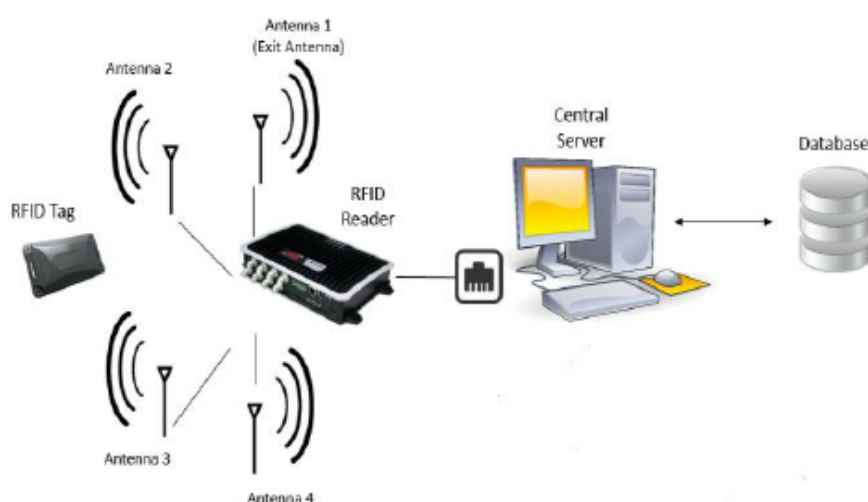
# Chương 1

## Giới thiệu tổng quan đề tài

### 1.1 Ý tưởng nghiên cứu, phạm vi nghiên cứu và đối tượng nghiên cứu

#### 1.1.1 Ý tưởng nghiên cứu

Công nghệ RFID - Radio Frequency Identification (Nhận dạng tần số sóng vô tuyến) đang trở nên phổ biến với sự đa dạng các ứng dụng trong kỉ nguyên công nghệ 4.0. Với kiến trúc đơn giản, hệ thống RFID được sử dụng trong nhiều lĩnh vực: kiểm soát ra vào, nhận diện khách hàng, động vật, đồ vật,...Nhất là trong hệ thống quản lý hàng hóa, kho bãi, chuỗi cung ứng,...



Hình 1.1: Kiến trúc cơ bản của một hệ thống RFID.

Trong đó, công nghệ UHF RFID có thể đọc đồng thời các đặc tính của các thẻ điện tử trường gần và trường xa. Ngoài loại thẻ kiểm soát truy cập thẻ ID truyền thống (kích

## CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN ĐỀ TÀI

---

thước thẻ tín dụng), công nghệ UHF RFID cung cấp nhiều loại thẻ điện tử để lựa chọn, có thể được cài đặt trong bất kỳ tài liệu nào và đang trong tình trạng tốt để theo dõi các mặt hàng, Container và phương tiện để cải thiện độ chính xác của hậu cần và ứng dụng theo dõi. Hiện nay, hầu hết các sản phẩm RFID UHF trên thị trường đều tập trung vào các ứng dụng như chuỗi cung ứng và quản lý hậu cần.

### 1.1.2 Phạm vi nghiên cứu

Vì sự giới hạn của thời gian và kiến thức, trong đề tài này, chúng ta sẽ nghiên cứu và ứng dụng một hệ thống UHF RFID đơn giản với đầy đủ kiến trúc của một hệ thống RFID thường thấy.

**Phần cứng** Các kiến thức về hệ thống UHF RFID, antenna, tag.

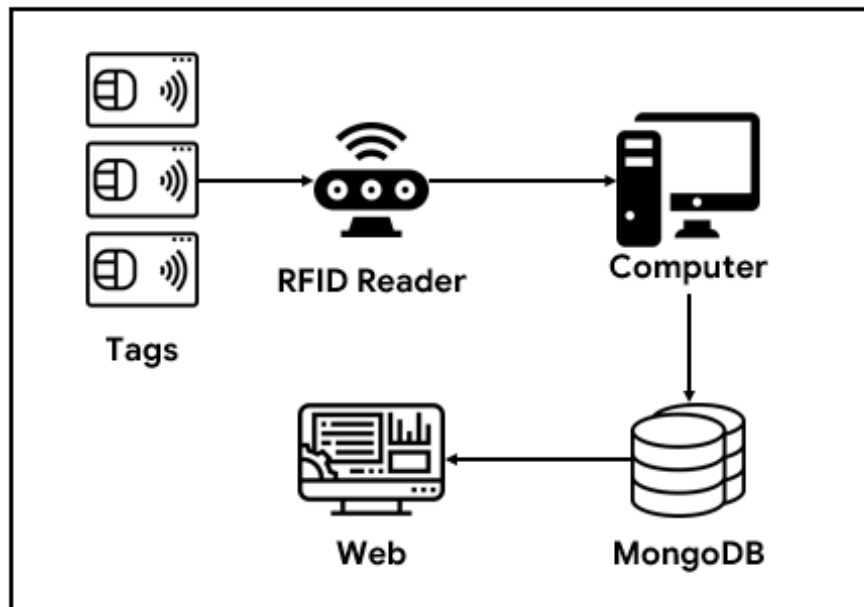
**Phần mềm** Các kiến thức về đọc và xử lý dữ liệu RFID, xây dựng hệ thống Web Sever, lập trình Web, làm việc với cơ sở dữ liệu MongoDB.

### 1.1.3 Đối tượng nghiên cứu

- Đọc thẻ tag.
- Xử lý dữ liệu trên tag.
- Cơ sở dữ liệu MongoDB.
- Web Service với NodeJS.
- Kết nối Web Server với CSDL MongoDB.
- Lập trình UI/UX Web.

## 1.2 Tổng quan về kiến trúc của hệ thống

Như có thể thấy trong hình 1.2, các dữ liệu dùng để nhận dạng từ các tag riêng biệt sau khi được đọc bằng Reader RFID sẽ được xử lý và lưu vào hệ cơ sở dữ liệu MongoDB. Sau đó được kết nối với Web Sever NodeJs và hiển thị trên giao diện web cho người dùng quan sát.



Hình 1.2: Tổng quan về kiến trúc của hệ thống.

# Chương 2

## Cơ sở lý thuyết

### 2.1 Công nghệ RFID UHF

Công nghệ RFID viết tắt bởi Radio Frequency Identification là một trong những công nghệ nhận dạng dữ liệu tự động tiên tiến nhất hiện nay có tính khả thi cao và áp dụng trong thực tế rất hiệu quả. RFID đang hiện diện trong rất nhiều lĩnh vực tự động hóa, rất nhiều ứng dụng quản lý và các mô hình tổ chức khác nhau nhằm đem lại những giải pháp nhận dạng dữ liệu tự động tối ưu và hiệu quả hơn.

#### 2.1.1 Công nghệ RFID là gì?

Công nghệ RFID (Radio Frequency Identification) cho phép một thiết bị đọc thông tin chứa trong chip không tiếp xúc trực tiếp ở khoảng cách xa, không thực hiện bất kỳ giao tiếp vật lý nào hoặc giữa hai vật không nhìn thấy. Công nghệ này cho ta phương pháp truyền, nhận dữ liệu từ một điểm đến điểm khác.

Kỹ thuật RFID sử dụng truyền thông không dây trong dải tần sóng vô tuyến để truyền dữ liệu từ các tag (thẻ) đến các reader (bộ đọc). Tag có thể được đính kèm hoặc gắn vào đối tượng được nhận dạng chẳng hạn sản phẩm, hộp hoặc giá kệ (pallet). Reader scan dữ liệu của tag và gửi thông tin đến cơ sở dữ liệu có lưu trữ dữ liệu của tag. Chẳng hạn, các tag có thể được đặt trên kính chắn gió xe hơi để hệ thống thu phí đường có thể nhanh chóng nhận dạng và thu tiền trên các tuyến đường.

Dạng đơn giản nhất được sử dụng hiện nay là hệ thống RFID bị động làm việc như sau: reader truyền một tín hiệu tần số vô tuyến điện từ qua anten của nó đến một con chip. Reader nhận thông tin trở lại từ chip và gửi nó đến máy tính điều khiển đầu đọc và xử lý thông tin lấy được từ chip. Các chip không tiếp xúc không tích điện, chúng hoạt động bằng cách sử dụng năng lượng nhận từ tín hiệu được gửi bởi reader.

### 2.1.2 Lịch sử phát triển của công nghệ RFID

Lịch sử RFID đánh dấu từ những năm 1930 nhưng công nghệ RFID có nguồn gốc từ năm 1897 khi Guglielmo Marconi phát ra radio. RFID áp dụng các nguyên tắc vật lý cơ bản như truyền phát rad ng điện từ truyền và nhận dạng dữ liệu khác nhau.

Để hiểu rõ hơn về sự giống nhau này, hình dung một trạm radio phát ra âm thanh hoặc âm nhạc qua một bộ phát. Dữ liệu này cần phải mã hóa sang dạng sóng radio có tần số xác định. Tại những vị trí khác nhau, người nghe có một máy radio để giải mã dữ liệu từ trạm phát (âm thanh hoặc âm nhạc). Mọi người đều nhận biết được sự khác nhau về chất lượng sóng radio khi ngồi trên xe hơi. Khi di chuyển càng xa bộ phát tín hiệu thu được càng yếu. Khoảng cách theo các hướng hoặc các vùng mà sóng radio phát ra có thể bao phủ được xác định bởi điều kiện môi trường, kích thước và năng lượng của anten tại mỗi đường giao tiếp. Sử dụng thuật ngữ RFID, có chức năng như một trạm truyền gọi là một transponder (tag) được tạo thành từ 2 thuật ngữ transmitter và responder; vật có chức năng như radio gọi là reader (bộ đọc) hay interrogator. Anten xác định phạm vi đọc (range).

Ba thành phần tag, reader và anten là những khối chính của một hệ thống RFID. Khi thay đổi về năng lượng, kích thước, thiết kế anten, tần số hoạt động, số lượng dữ liệu và phần mềm để quản lý và xuất dữ liệu tạo ra rất nhiều ứng dụng. Công nghệ RFID có thể giải quyết rất nhiều bài toán kinh doanh thực tế.

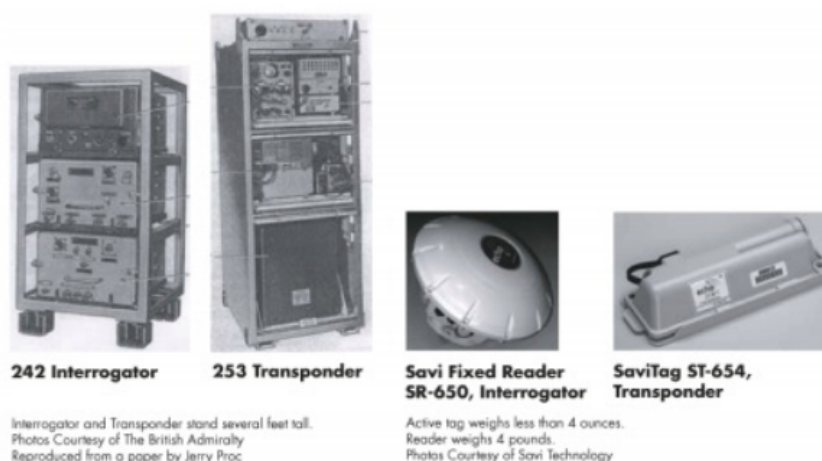
#### 2.1.2.1 Thời kỳ đầu của RFID

Các máy thu GPS ngày nay cực kì chính xác, nhờ vào thiết kế nhiều kênh hoạt động song song của chúng. Các máy thu 12 kênh song song (của Garmin) nhanh chóng khóa vào các quỹ vệ tinh khi mới bật lên và chúng duy trì kết nối bền vững, thậm chí trong tán lá rậm rạp hoặc thành phố với các tòa nhà cao tầng. Trạng thái của khí quyển và các nguồn gây sai số khác có thể ảnh hưởng tới độ chính xác của máy thu GPS. Các máy thu GPS có độ chính xác trung bình trong vòng 15 mét.

Các máy thu mới hơn với khả năng WAAS (Wide Area Augmentation System) có thể tăng độ chính xác trung bình tới dưới 3 mét. Không cần thêm thiết bị hay mất phí để có được lợi điểm của WAAS. Người dùng cũng có thể có độ chính xác tốt hơn với GPS vi sai (Differential GPS, DGPS) sửa lỗi các tín hiệu GPS để có độ chính xác trong khoảng 3 đến 5 mét. Cục Phòng vệ Bờ biển Mỹ vận hành dịch vụ sửa lỗi này. Hệ thống bao gồm một mạng các đài thu tín hiệu GPS và phát tín hiệu đã sửa lỗi bằng các máy phát hiệu. Để thu được tín hiệu đã sửa lỗi, người dùng phải có máy thu tín hiệu vi sai bao gồm cả ăng-ten để dùng với máy thu GPS của họ.

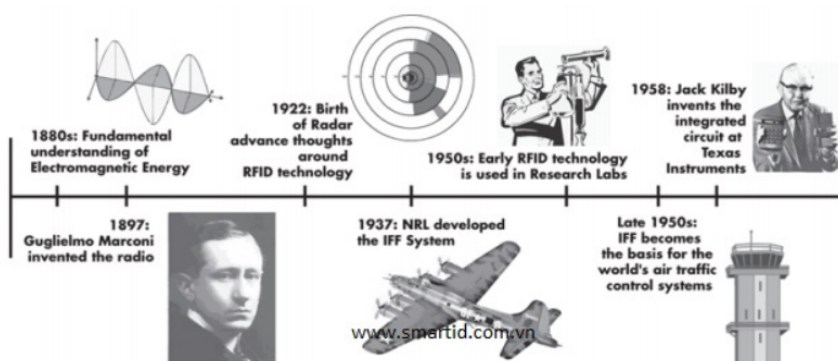
Những công nghệ mới những sản phẩm này gọn hơn và rẻ hơn như: công nghệ tích hợp trong IC, chip nhớ lập trình được, vi xử lý, những phần mềm ứng dụng hiện đại ngày nay và những ngôn ngữ lập trình làm cho công nghệ RFID đang có xu hướng chuyển sang lĩnh vực thương mại rộng lớn.

Cuối thập kỉ 60 đầu thập kỉ 70 nhiều công ty như Sensormatic and Checkpoint Systems giới thiệu những sản phẩm mới ít phức tạp hơn và ứng dụng rộng rãi hơn. Những công ty này bắt đầu phát triển thiết bị giám sát điện tử (electronic article surveillance EAS) để bảo vệ và kiểm kê sản phẩm như quần áo trong cửa hàng, sách trong thư viện. Hệ thống



Hình 2.1: Thiết bị IFF (bên trái), thiết bị RFID (tích cực) hiện đại ngày nay.

RFID thương mại ban đầu này chỉ là hệ thống RFID tag một bit (1-bit tag) giá rẻ để xây dựng, thực hiện và bảo hành. Tag không đòi hỏi nguồn pin (loại thụ động) dễ dàng đặt vào sản phẩm và thiết kế để khởi động chuông cảnh báo khi tag đến gần bộ đọc, thường đặt tại lối ra vào, phát hiện sự có mặt của tag.



Hình 2.2: Các mốc thời gian quan trọng trong giai đoạn đầu của RFID.

### 2.1.2.2 Phát hiện các vật thể riêng biệt

Suốt thập kỷ 70, công nghiệp sản xuất, vận chuyển bắt đầu nghiên cứu và phát triển những dự án để tìm cách dùng IC dựa trên hệ thống RFID. Có nhiều ứng dụng trong công nghiệp tự động, xác định thú vật, theo dõi lưu thông. Trong giai đoạn này tag có IC tiếp tục phát triển và đặc tính: bộ nhớ ghi được, tốc độ đọc nhanh hơn và khoảng cách đọc xa hơn.

Đầu thập niên 80 công nghệ phức tạp RFID được áp dụng trong nhiều ứng dụng: đặt tại đường ray ở Mỹ, đánh dấu thú vật trên nông trại ở châu Âu. Hệ thống RFID còn dùng trong nghiên cứu động vật hoang dã đánh dấu các loài nguy hiểm. Vào thập niên 90, hệ thống thu phí điện tử trở nên phổ biến ở Thái Bình Dương: Ý, Tây Ban Nha, Bồ Đào







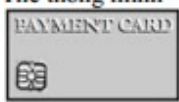
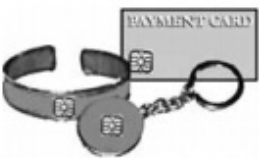
## CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Nha... và ở Mỹ: Dallas, New York và New Jersey. Những hệ thống này cung cấp những dạng truy cập điều khiển phức tạp hơn bởi vì nó còn bao gồm cả máy trả tiền.

Đầu năm 1990, nhiều hệ thống thu phí ở Bắc Mỹ tham gia một lực lượng mang tên EZPass Interagency Group (IAG) cùng nhau phát triển những vùng có hệ thống thu phí điện tử tương thích với nhau. Đây là cột mốc quan trọng để tạo ra những ứng dụng tiêu chuẩn. Hầu hết những tiêu chuẩn tập trung các đặc tính kỹ thuật như tần số hoạt động và giao thức giao tiếp phần cứng.

E-Zpass còn là một tag đơn tương ứng với một tài khoản trên một phương tiện. Tag của xe sẽ truy cập vào đường cao tốc của hệ thống thu phí mà không phải dừng lại. E-Z Pass giúp lưu thông dễ dàng hơn và giảm lực lượng lao động để kiểm soát vé và thu tiền.

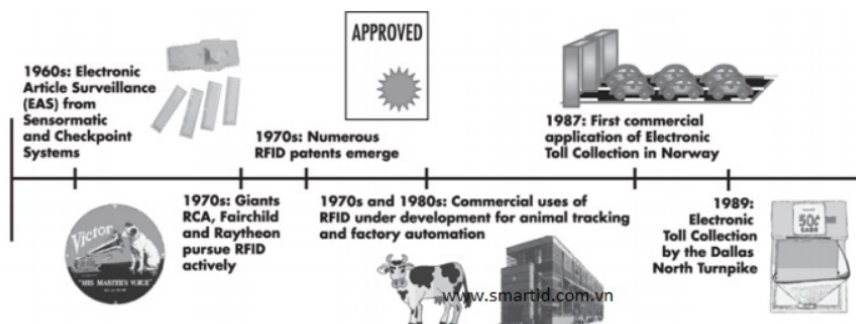
Cùng vào thời điểm này, khóa (card RFID) sử dụng phổ biến thay thế cho các thiết bị máy móc điều khiển truy nhập truyền thống như khóa kim loại và khóa số. Những sản phẩm này còn được gọi là thẻ thông minh không tiếp xúc cung cấp thông tin về người dùng, trong khi giá thành thấp để sản xuất và lắp trình. Hình 1.3 so sánh các phương pháp điều khiển truy cập thông thường và điều khiển truy cập RFID:

<b>Chìa khóa kim loại</b> 	<ul style="list-style-type: none"><li>▪ Không cần nguồn điện</li><li>▪ Dễ sử dụng</li></ul>	<ul style="list-style-type: none"><li>▪ Dễ dàng copy</li><li>▪ Khóa có thể bị mất</li><li>▪ Dễ bị trộm</li></ul>
<b>Khóa kết hợp</b> 	<ul style="list-style-type: none"><li>▪ Có thể dễ dàng thay đổi sự kết hợp</li><li>▪ Không có chìa khóa nên không lo bị mất hay bị đánh cắp</li></ul>	<ul style="list-style-type: none"><li>▪ Đắt hơn khóa kim loại</li><li>▪ Dễ bị tấn công</li></ul>
<b>Thẻ đóng dấu</b> 	<ul style="list-style-type: none"><li>▪ Không thể nhân lên dễ dàng như khóa kim loại</li></ul>	<ul style="list-style-type: none"><li>▪ Sử dụng kỹ thuật cũ ít linh hoạt</li></ul>
<b>Thẻ dùng dải từ trường</b> 	<ul style="list-style-type: none"><li>▪ Khó copy</li><li>▪ Có sẵn bộ đọc card</li></ul>	<ul style="list-style-type: none"><li>▪ Sử dụng lâu thẻ sẽ hư</li><li>▪ Việc lắp đặt yêu cầu cơ sở IT</li></ul>
<b>Thẻ thông minh</b> 	<ul style="list-style-type: none"><li>▪ Cùng một thẻ có thể sử dụng cho nhiều ứng dụng</li><li>▪ Có khả năng bảo mật cao hơn thẻ dùng dải từ trường</li></ul>	<ul style="list-style-type: none"><li>▪ Đắt hơn thẻ từ trường</li></ul>
<b>RFID</b> 	<ul style="list-style-type: none"><li>▪ Như thẻ thông minh</li><li>▪ Không cần phải tiếp xúc</li><li>▪ Có thể gắn lên sản phẩm và dưới da</li></ul>	<ul style="list-style-type: none"><li>▪ Đắt hơn thẻ thông minh</li></ul>

Hình 2.3: Các phương pháp điều khiển truy cập thông thường và điều khiển truy cập RFID.

Điều khiển truy nhập RFID tiếp tục có những bước tiến mới. Các nhà sản xuất xe hơi đã

dùng tag RFID trong gần một thập kỷ qua cho hệ thống đánh lửa xe hơi và nó đã làm giảm khả năng trộm cắp xe.



Hình 2.4: Những mốc thời gian quan trọng từ năm 1960 đến 1990.

### 2.1.2.3 RFID phát triển trên toàn cầu

Cuối thế kỷ 20, số lượng các ứng dụng RFID hiện đại bắt đầu mở rộng theo hàm mũ trên phạm vi toàn cầu. Dưới đây là một vài bước tiến quan trọng góp phần đẩy mạnh sự phát triển này. Texas Instrument đi tiên phong ở Mỹ năm 1991, công ty đã tạo ra một hệ thống xác nhận và đăng ký Texas Instrument (TIRIS). Hệ thống TI-RFID (Texas Instruments Radio Frequency Identification System) n tận cho phát triển và thực hiện những lớp mới của ứng dụng RFID. Châu Âu đã bắt đầu công nghệ RFID từ rất sớm.

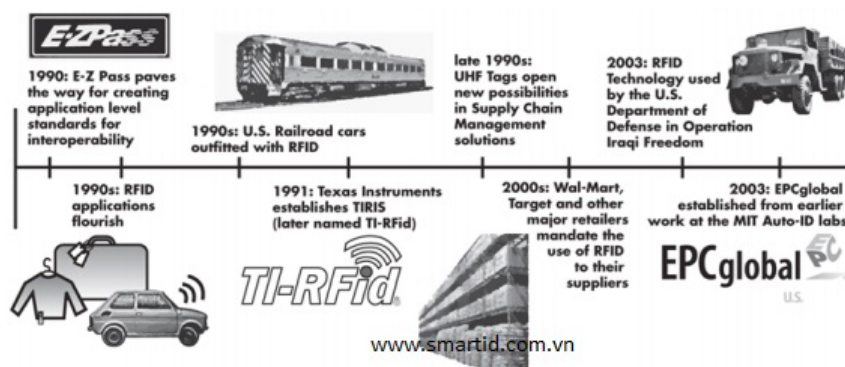
Ngay cả trước khi Texas Instrument giới thiệu sản phẩm RFID, vào năm 1970 EM Microelectronic-Marin một công ty của The Swatch Group Ltd đã thiết kế mạch tích hợp năng lượng thấp cho những đồng hồ của Thụy Sĩ. Năm 1982 Mikron Integrated Microelectronics phát minh ra công nghệ ASIC và năm 1987 phát triển công nghệ đặc biệt liên quan đến việc xác định thẻ thông minh. Ngày nay EM Microelectronic và Philips Semiconductors là hai nhà sản xuất lớn ở châu Âu về lĩnh vực RFID.

Cách đây một vài năm các ứng dụng chủ yếu của thẻ RFID thụ động, như minh họa trong bảng 2.2 mới được ứng dụng ở tần số thấp (LF) và tần số cao (HF) của phổ RF. Cả LF và HF đều giới hạn khoảng cách và tốc độ truyền dữ liệu. Cho những mục đích thực tế khoảng cách của những ứng dụng này đo bằng inch. Việc giới hạn tốc độ ngăn cản việc đọc của ứng dụng khi hàng trăm thậm chí hàng ngàn tag cùng có mặt trong trường của bộ đọc tại một thời điểm. Cuối thập niên 90 tag thụ động cho tần số siêu cao (UHF) làm cho khoảng cách xa hơn, tốc độ cao hơn, giá cả rẻ hơn, tag thụ động này đã vượt qua những giới hạn của nó; Với những thuộc tính thêm vào hệ thống RFID dựa trên UHF được lựa chọn cho những ứng dụng dây chuyền cung cấp như quản lý nhà kho, kiểm kê sản phẩm.

Cuối những năm 1990 đầu năm 2000, các nhà phân phối như Wal-Mart, Target, Metro Group và các cơ quan chính phủ như U.S. Department of Defense (DoD) bắt đầu phát triển và yêu cầu việc sử dụng RFID bởi nhà cung cấp. Vào thời điểm này EPCglobal được thành lập, EPCglobal đã hỗ trợ hệ thống mã sản phẩm điện tử (Electronic Product Code Network EPC) hệ thống này đã trở thành tiêu chuẩn cho xác nhận sản phẩm tự động.

LF	HF
Điều khiển truy nhập	Xác định động vật
Xác định hàng hóa trên máy bay	Thanh toán tiền
Chống trộm xe hơi	Giám sát điện tử
Đánh dấu tài liệu	Định thời cho thể thao

Bảng 2.1: Các ứng dụng tiêu biểu dùng công nghệ RFID LF và HF



Hình 2.5: Những mốc thời gian quan trọng từ năm 1990 đến nay.

### 2.1.3 Phân loại hệ thống RFID

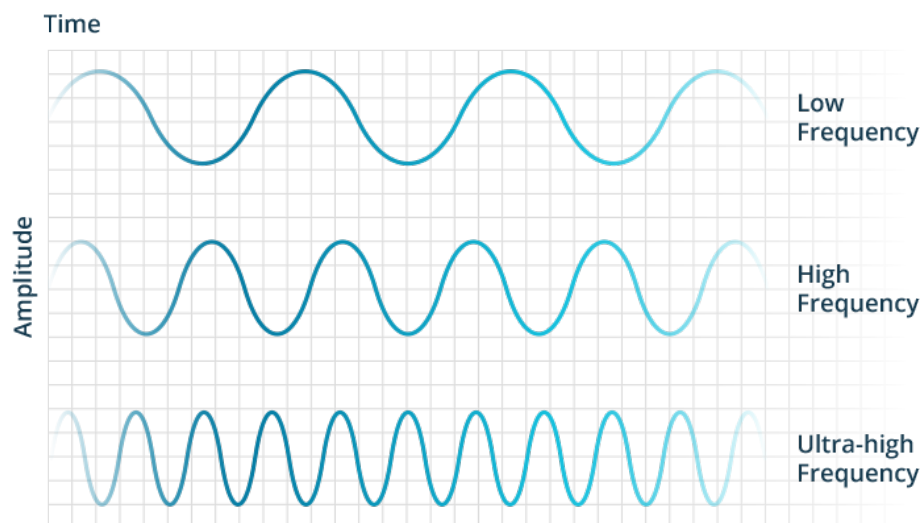
Hệ thống RFID có thể được chia nhỏ theo các nhóm tần số mà chúng hoạt động: tần số thấp, tần số cao, và tần số cực cao. Ngoài ra còn có hai loại của hệ thống RFID là hệ thống chủ động và hệ thống thụ động.

#### 2.1.3.1 Các dải tần số RFID

Tần số phụ thuộc vào kích cỡ của các sóng radio được sử dụng trong việc giao tiếp giữa các thành phần hợp thành nên hệ thống. Hệ thống RFID trên toàn thế giới hoạt động chủ yếu theo ba nhóm: tần số thấp (LF), tần số cao (HF) và tần số siêu cao (UHF). Các sóng radio truyền khác nhau tại mỗi dải tần số với nhiều thuận lợi và hạn chế trong mối liên kết với việc sử dụng mỗi nhóm tần số.

Nếu một hệ thống RFID hoạt động tại một dải tần số thấp, thì nó sẽ có phạm vi đọc ngắn hơn và tốc độ đọc dữ liệu cũng chậm hơn, nhưng lại tăng khả năng đọc trên các bề mặt chất lỏng, trong phạm vi gần hoặc trên các bề mặt kim loại. Nếu một hệ thống hoạt động tại một dải tần số cao, nhìn chung nó sẽ có tốc độ chuyển dữ liệu nhanh hơn, và phạm vi đọc cũng sẽ dài hơn so với các hệ thống sử dụng dải tần số thấp, tuy nhiên nó lại nhạy cảm với sự can thiệp của sóng radio gây ra bởi các chất lỏng và các chất kim loại trong môi trường.

**LF RFID** Nhóm LF bao phủ tần số trong khoảng từ 30KHz tới 300 KHz. Các hệ thống LF RFID tiêu biểu thì hoạt động tại 125 KHz, mặc dù có một vài hệ thống hoạt động tại



Hình 2.6: Phân loại hệ thống RFID theo tần số.

134 KHz. Nhóm tần số này đáp ứng phạm vi đọc ngắn chỉ khoảng 10 cm, và có tốc độ đọc chậm hơn so với các dải tần số cao, tuy nhiên nó lại không dễ bị tác động bởi sự can thiệp của sóng radi

Các ứng dụng LF RFID bao gồm việc kiểm soát tài sản và theo dõi vật nuôi.

Các tiêu chuẩn cho hệ thống theo dõi động vật LF được miêu tả rõ trong ISO 14223 và ISO/IEC 18000-2. Dải LF không được công nhận như một ứng dụng thực sự trên toàn thế giới bởi một vài khác biệt trong các mức độ tần số và năng lượng trên toàn thế giới.

**HF RFID** Nhóm HF hoạt động trong phạm vi từ 3 MHz đến 30 MHz. Đa số các hệ thống HF RFID hoạt động tại 13,56 MHz với phạm vi đọc vào khoảng giữa 10 cm tới 1 mét. Các hệ thống HF thí nghiệm độ nhạy vừa phải để can thiệp.

HF RFID thường được sử dụng cho việc bán vé, thanh toán, và các ứng dụng chuyển dữ liệu.

Có một vài tiêu chuẩn HF RFID được đặt ra như tiêu chuẩn ISO 15693 cho việc theo dõi các mặt hàng, hay các tiêu chuẩn ECMA-340 và ISO/IEC 18092 cho Near Field Communication (NFC) – thiết bị tầm ngắn được sử dụng trong việc trao đổi dữ liệu giữa các thiết bị. Các tiêu chuẩn HF khác bao gồm: tiêu chuẩn ISO/IEC 14443 A và tiêu chuẩn ISO/IEC 14443 cho công nghệ MIFARE được sử dụng trong các thẻ thông minh và các thẻ ở khoảng cách gần; và tiêu chuẩn JIS X 6319-4 cho FeliCa – một hệ thống thẻ thông minh thường được sử dụng trong các thẻ tính tiền điện tử.

**UHF RFID** Nhóm UHF hoạt động trong phạm vi từ 300 MHz đến 3 GHz. Các hệ thống tuân theo tiêu chuẩn UHF Gen2 cho hệ thống RFID sử dụng nhóm từ 860 đến 960 MHz. Trong khi có một số khác biệt giữa tần số từ vùng này đến vùng khác, nhưng hệ thống UHF Gen2 RFID tại hầu hết các nước lại hoạt động trong khoảng giữa 900 và 915 MHz.

Phạm vi đọc của các hệ thống UHF thụ động có thể ở trong khoảng 12 mét, và UHF RFID có tốc độ chuyển dữ liệu nhanh hơn so với LF hay HF. UHF RFID thường nhạy

cảm nhất với các sự can thiệp, nhưng nhiều nhà chế tạo sản phẩm UHF đã tìm ra các cách để thiết kế thẻ, ăng-ten, và đầu đọc nhằm duy trì được hiệu suất cao ngay cả trong các môi trường có điều kiện khó khăn. Các thẻ UHF thụ động dễ chế tạo và chi phí sản xuất cũng rẻ hơn so với các thẻ LF và HF.

UHF RFID được sử dụng trong đa dạng các ứng dụng, từ quản lý việc kiểm kê hàng hóa trong các hoạt động bán lẻ tới chống hàng được phẩm giả, hay cài đặt cấu hình cho các thiết bị không dây. Phần lớn các dự án RFID mới đang sử dụng UHF đều trái với LF và HF, tạo ra sự phân khúc nhanh nhất trong thị trường RFID.

Nhóm UHF được quy định bởi một tiêu chuẩn toàn cầu gọi là tiêu chuẩn EPCglobal Gen2 (ISO 18000-6C) UHF.

UHF	HF và LF
<ul style="list-style-type: none"> <li>• Tiêu chuẩn Gen2 duy nhất trên toàn thế giới</li> <li>• Gấp 20 lần phạm vi và tốc độ của HF</li> <li>• Các nhãn có giá từ 5 cent đến 15 cent (năm 2012)</li> <li>• Công nghệ gắn thẻ lên hàng hóa</li> </ul>	<ul style="list-style-type: none"> <li>• Nhiều tiêu chuẩn cạnh tranh</li> <li>• HF dựa trên công nghệ NFC đảm bảo cho việc thanh toán</li> <li>• Các nhãn, thẻ, vi mạch có giá từ 50 cent đến 2 đô</li> <li>• Được sử dụng cho các công việc thu hồi, bán vé, hay thanh toán</li> </ul>

Hình 2.7: So sánh UHF và LF, HF.

### 2.1.3.2 Các hệ thống chủ động, thụ động và BAP RFID

**Các hệ thống RFID chủ động** Tại các hệ thống RFID chủ động, các thẻ tự có cho riêng mình hệ thống điều khiển và nguồn năng lượng. Thường thường thì nguồn năng lượng đó chính là Pin. Các thẻ chủ động sẽ truyền đi các tín hiệu của chúng để chuyển thành thông tin rồi được lưu trữ trong các mạch vi xử lý của chúng.

Hệ thống RFID chủ động hoạt động điển hình trong nhóm UHF và cung cấp phạm vi đọc lên đến 100 mét. Nhìn chung, các thẻ chủ động được sử dụng cho nhiều mục đích như các xe đường sắt, các container lớn tái sử dụng, và các tài sản khác cần được theo dõi ở khoảng cách dài.

Có hai loại thẻ chủ động chính là: các hệ thống tiếp sóng và đèn hiệu. Các hệ thống tiếp sóng được đánh thức khi chúng nhận được tín hiệu radio từ một đầu đọc, và sau đó năng lượng sẽ bật và phản ứng lại bằng việc truyền tín hiệu quay trở lại. Bởi vì tiết kiệm Pin, nên hệ thống tiếp sóng sẽ không tích cực phát ra sóng radio cho đến khi chúng nhận được tín hiệu từ đầu đọc.

Các đèn hiệu được sử dụng ở đa số các hệ thống định vị thời gian thực (RTLS), cốt để việc theo dõi chính xác vị trí của một tài sản được thực hiện liên tục. Không giống với các hệ thống tiếp sóng, đèn hiệu không được bật bởi tín hiệu của đầu đọc. Thay vào đó, chúng nhả ra các tín hiệu tại các khoảng thời gian được đặt trước. Phụ thuộc vào cấp độ của việc yêu cầu chính xác địa điểm, các đèn hiệu có thể được thiết lập để nhả ra tín hiệu mỗi lần trong vài giây, hoặc một lần mỗi ngày. Mỗi tín hiệu đèn hiệu được nhận bởi các

ăng-ten đầu đọc được đặt xung quanh chu vi của các khu vực đang bị giám sát, và liên kết với thông tin ID của thẻ và vị trí.

**Hệ thống RFID thụ động** Trong các hệ thống RFID thụ động, đầu đọc và các ăng-ten đầu đọc gửi tín hiệu radio đến thẻ. Thẻ RFID sau đó sử dụng tín hiệu được truyền để bật nguồn năng lượng, và phản xạ năng lượng ngược trở lại đầu đọc.

Các hệ thống RFID thụ động có thể hoạt động ở các nhóm tần số thấp (LF), tần số cao (HF), hay tần số siêu cao (UHF). Khi các phạm vi của hệ thống thụ động bị giới hạn bởi năng lượng của backscatter của thẻ (tín hiệu radio phản xạ từ thẻ quay trở lại đầu đọc), thì phạm vi của chúng ít hơn 10 mét. Bởi các thẻ thụ động không yêu cầu nguồn năng lượng hay hệ thống điều khiển, mà chỉ yêu cầu một thẻ chip và ăng-ten, nên chúng thường có chi phí rẻ hơn, nhỏ hơn và dễ chế tạo hơn các thẻ chủ động.

Các thẻ thụ động có thể được đóng gói theo nhiều cách khác nhau, tùy thuộc vào mỗi yêu cầu ứng dụng cụ thể. Lấy một ví dụ, chúng có thể được gắn lên một chất nền, hoặc kẹp vào giữa một lớp chất dính và một nhãn giấy để tạo ra các thẻ RFID thông minh. Các thẻ thụ động cũng có thể được gắn vào đa dạng các thiết bị hoặc đóng gói để tạo ra thẻ có thể chịu được các điều kiện nhiệt độ cực đoan hay các chất hóa học gay gắt.

Các giải pháp RFID thụ động hữu ích cho nhiều ứng dụng, và thường được triển khai để theo dõi hàng hóa trong chuỗi cung ứng, để kiểm kê số lượng tài sản trong nền công nghiệp bán lẻ, để xác nhận các sản phẩm như dược phẩm,... và để gắn vào năng lực RFID trong hàng loạt các thiết bị. Hệ thống RFID thụ động thậm chí có thể được sử dụng trong các nhà kho và các trung tâm phân phối, mặc dù phạm vi của nó ngắn hơn, bằng việc đặt các đầu đọc tại các điểm nhất định để theo dõi quá trình vận chuyển tài sản.

**Các hệ thống trợ Pin thụ động (Battery-Assisted Passive-BAP)** Các thẻ BAP RFID là một loại thẻ thụ động kết hợp với một đặc trưng vô cùng quan trọng của thẻ chủ động. Trong khi đa số các thẻ thụ động RFID sử dụng năng lượng từ tín hiệu của đầu đọc RFID để kích hoạt con chip của thẻ và tán xạ chúng tới đầu đọc, thì các thẻ BAP lại sử dụng nguồn năng lượng tích hợp (thường là Pin) để kích hoạt con chip, bởi vậy toàn bộ nguồn dữ liệu được lưu trữ từ đầu đọc đều có thể được sử dụng cho việc tán xạ. Không giống các hệ thống tiếp sóng, các thẻ BAP không tự có cho chúng hệ thống điều khiển.

## 2.2 Cơ sở dữ liệu MongoDB

### 2.2.1 NoSql là gì?

NoSQL là 1 dạng CSDL mã nguồn mở và được viết tắt bởi: None-Relational SQL hay có nơi thường gọi là Not-Only SQL.

NoSQL được phát triển trên Javascript Framework với kiểu dữ liệu là JSON và dạng dữ liệu theo kiểu key và value.

NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu sót cũng như hạn chế



	Hệ thống RFID chủ động	Hệ thống RFID thụ động	Hệ thống trợ Pin thụ động (BAP)
Nguồn năng lượng thẻ	Bên trong thẻ	Năng lượng truyền từ đầu đọc qua RF	Thẻ sử dụng nguồn năng lượng bên trong để kích hoạt, và năng lượng được chuyển từ đầu đọc qua RF để tán xạ
Thẻ Pin	Có	Không	Có
Giá trị của thẻ năng lượng	Liên tục	Chỉ trong phạm vi đầu đọc	Chỉ trong phạm vi đầu đọc
Yêu cầu độ mạnh của tín hiệu từ đầu đọc tới thẻ	Rất thấp	Rất cao	Vừa phải
Giá trị độ mạnh tín hiệu từ thẻ tới đầu đọc	Cao	Rất thấp	Vừa phải
Phạm vi giao tiếp	Dài (100 mét hoặc hơn)	Ngắn (lên đến khoảng 10 mét)	Trung bình (lên đến khoảng 100 mét)
Khả năng cảm biến	Khả năng liên tục giám sát và ghi chép cảm biến đầu vào	Khả năng đọc và chuyển các giá trị cảm biến chỉ khi thẻ được cung cấp bởi đầu đọc	Khả năng đọc và chuyển các giá trị cảm biến chỉ khi thẻ nhận tín hiệu RF từ đầu đọc

Hình 2.8: So sánh các hệ thống chủ động, thụ động và BAP RFID.

của mô hình dữ liệu quan hệ RDBMS (Relational Database Management System - Hệ quản trị cơ sở dữ liệu quan hệ) về tốc độ, tính năng, khả năng mở rộng,...

Với NoSQL bạn có thể mở rộng dữ liệu mà không lo tới những việc như tạo khóa ngoại, khóa chính, kiểm tra ràng buộc .v.v ...

NoSQL bỏ qua tính toàn vẹn của dữ liệu và transaction để đổi lấy hiệu suất nhanh và khả năng mở rộng.

NoSQL được sử dụng ở rất nhiều công ty, tập đoàn lớn, ví dụ như FaceBook sử dụng Cassandra do FaceBook phát triển, Google phát triển và sử dụng BigTable,...

### 2.2.2 MongoDB là gì?



Hình 2.9: CSDL MongoDB.

MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng.

MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh. Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là collection thay vì bảng.

So với RDBMS thì trong MongoDB collection ứng với table, còn document sẽ ứng với row, MongoDB sẽ dùng các document thay cho row trong RDBMS.

Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định.

Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB.

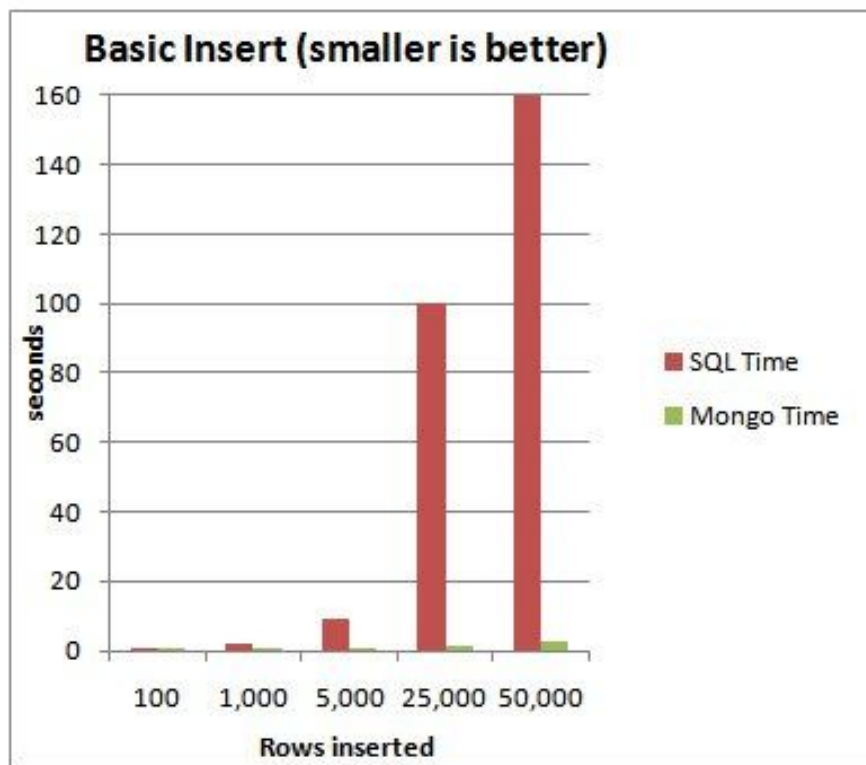
### 2.2.3 Ưu điểm của MongoDB

- Do MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ có các kích cỡ và các document khác nhau, linh hoạt trong việc lưu trữ dữ liệu, nên bạn muốn gì thì cứ insert vào thoải mái.
- Dữ liệu trong MongoDB không có sự ràng buộc lẫn nhau, không có join như trong RDBMS nên khi insert, xóa hay update 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  $idx$  luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.
- Khi có một truy vấn dữ liệu, bản ghi được cached lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.
- 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 (RDBMS). 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.

### 2.2.4 Nhược điểm

- Một ưu điểm của MongoDB cũng chính là nhược điểm của nó. MongoDB không có các tính chất ràng buộc như trong RDBMS nên khi thao tác với MongoDB thì phải hết sức cẩn thận.
- Tổn bộ nhớ do dữ liệu lưu dưới dạng key-value, các collection chỉ khác về value do đó key sẽ bị lặp lại. Không hỗ trợ join nên dễ bị dư thừa dữ liệu.





Hình 2.10: So sánh tốc độ ghi của MongoDB và MySQL.

- Khi insert/update/remove bản ghi, MongoDB sẽ chưa cập nhật ngay xuống ổ cứng, mà sau 60 giây MongoDB mới thực hiện ghi toàn bộ dữ liệu thay đổi từ RAM xuống ổ cứng điều này sẽ là nhược điểm vì sẽ có nguy cơ bị mất dữ liệu khi xảy ra các tình huống như mất điện...

## Chương 3

# Thiết kế và thực hiện phần cứng

### 3.1 Thiết kế kiến trúc phần cứng

#### 3.1.1 Các thành phần

#### 3.1.2 Yêu cầu thiết kế

### 3.2 Thực hiện phần cứng

## Chương 4

# Thiết kế và thực hiện phần mềm

### 4.1 Thiết kế kiến trúc phần mềm

#### 4.1.1 Các thành phần

Kiến trúc phần mềm của hệ thống gồm nhiều thành phần:

- Đọc và xử lý data từ các Tags.
- Tạo Server có kết nối với CSDL MongoDB để lưu data.
- Tạo Web service hiển thị trực quan các thông tin từ CSDL MongoDB.

#### 4.1.2 Yêu cầu thiết kế

- Đáp ứng thực hiện đầy đủ các thành phần trong kiến trúc phần mềm của hệ thống.
- Đảm bảo hệ thống vận hành chính xác, hiệu quả, độ trễ thấp, kết nối bền vững.
- Giao diện web thân thiện, dễ truy cập, ít tốn tài nguyên.

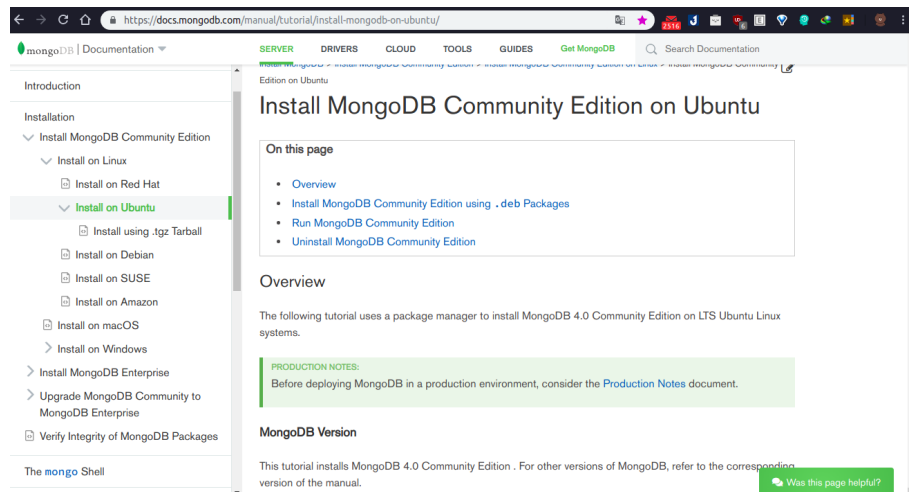
### 4.2 Thực hiện phần mềm

#### 4.2.1 Cài đặt và cấu hình MongoDB

##### 4.2.1.1 Cài đặt MongoDB

Hệ điều hành sử dụng trong đề cương: Ubuntu 18.04.1 LTS.

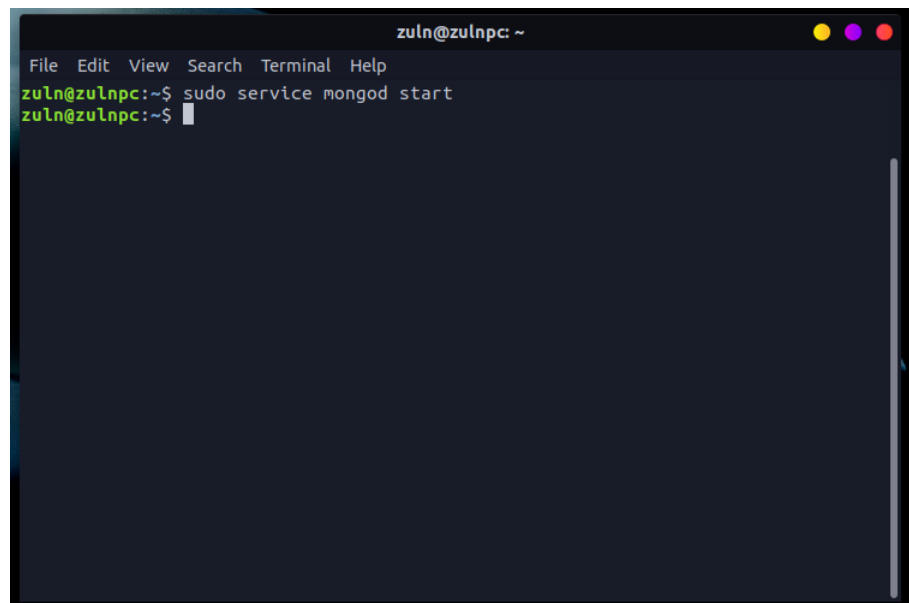
Tiến hành truy cập trang chủ [https : //docs.mongodb.com/manual/tutorial/install – mongodb – on – ubuntu/](https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/) và cài đặt theo các hướng dẫn.



Hình 4.1: Hướng dẫn cài đặt MongoDB trên Ubuntu.

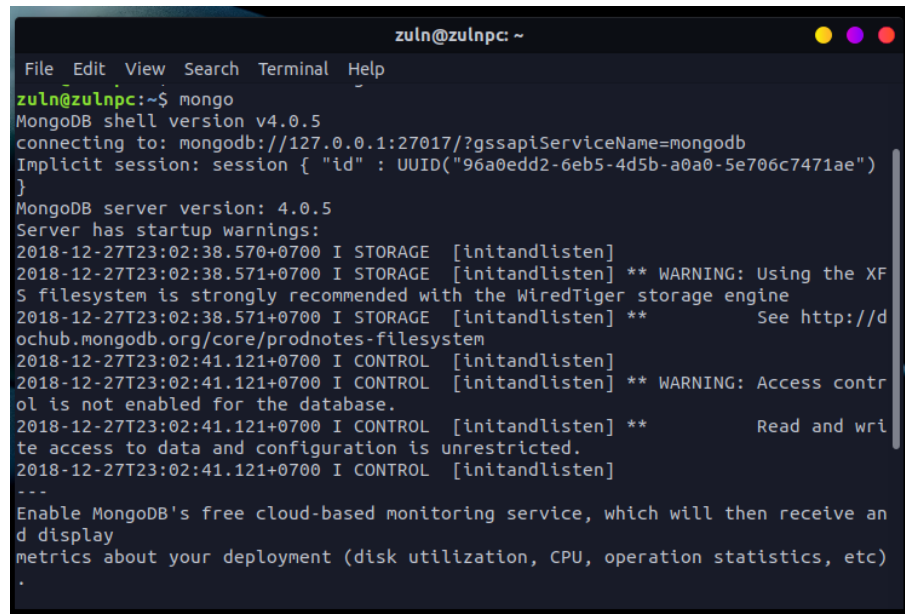
### 4.2.1.2 Cấu hình MongoDB

Mở Terminal và gõ **sudo service mongod start** để khởi chạy MongoDB.



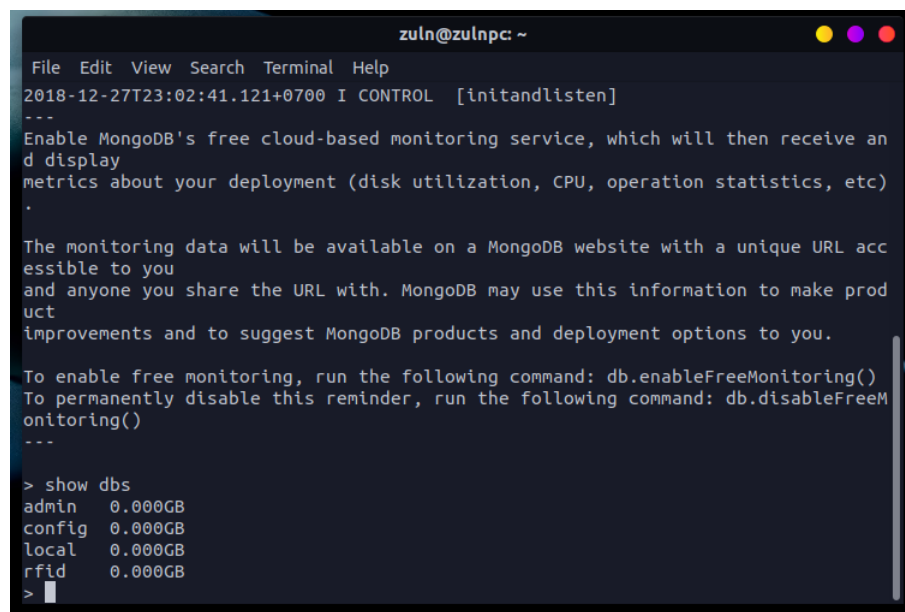
Hình 4.2: Khởi chạy MongoDB.

Gõ **mongo** để truy xuất MongoDB. Gõ **show dbs** để xem tất cả các database trong máy.



```
zuln@zulnpc: ~  
File Edit View Search Terminal Help  
zuln@zulnpc:~$ mongo  
MongoDB shell version v4.0.5  
connecting to: mongodb://127.0.0.1:27017/?gssapiServiceName=mongodb  
Implicit session: session { "id" : UUID("96a0edd2-6eb5-4d5b-a0a0-5e706c7471ae")  
}  
MongoDB server version: 4.0.5  
Server has startup warnings:  
2018-12-27T23:02:38.570+0700 I STORAGE [initandlisten]  
2018-12-27T23:02:38.571+0700 I STORAGE [initandlisten] ** WARNING: Using the XFS  
filesystem is strongly recommended with the WiredTiger storage engine  
2018-12-27T23:02:38.571+0700 I STORAGE [initandlisten] ** See http://d  
ochub.mongodb.org/core/prodnotes-filesystem  
2018-12-27T23:02:41.121+0700 I CONTROL [initandlisten]  
2018-12-27T23:02:41.121+0700 I CONTROL [initandlisten] ** WARNING: Access contr  
ol is not enabled for the database.  
2018-12-27T23:02:41.121+0700 I CONTROL [initandlisten] ** Read and wri  
te access to data and configuration is unrestricted.  
2018-12-27T23:02:41.121+0700 I CONTROL [initandlisten]  
---  
Enable MongoDB's free cloud-based monitoring service, which will then receive an  
d display  
metrics about your deployment (disk utilization, CPU, operation statistics, etc)  
.
```

Hình 4.3: Truy xuất MongoDB.



```
zuln@zulnpc: ~  
File Edit View Search Terminal Help  
2018-12-27T23:02:41.121+0700 I CONTROL [initandlisten]  
---  
Enable MongoDB's free cloud-based monitoring service, which will then receive an  
d display  
metrics about your deployment (disk utilization, CPU, operation statistics, etc)  
.  
  
The monitoring data will be available on a MongoDB website with a unique URL acc  
essible to you  
and anyone you share the URL with. MongoDB may use this information to make prod  
uct  
improvements and to suggest MongoDB products and deployment options to you.  
  
To enable free monitoring, run the following command: db.enableFreeMonitoring()  
To permanently disable this reminder, run the following command: db.disableFreeM  
onitoring()  
---  
  
> show dbs  
admin 0.000GB  
config 0.000GB  
local 0.000GB  
rfid 0.000GB  
> 
```

Hình 4.4: Xem tất cả các database MongoDB.

### 4.2.2 Tạo Server truy xuất MongoDB

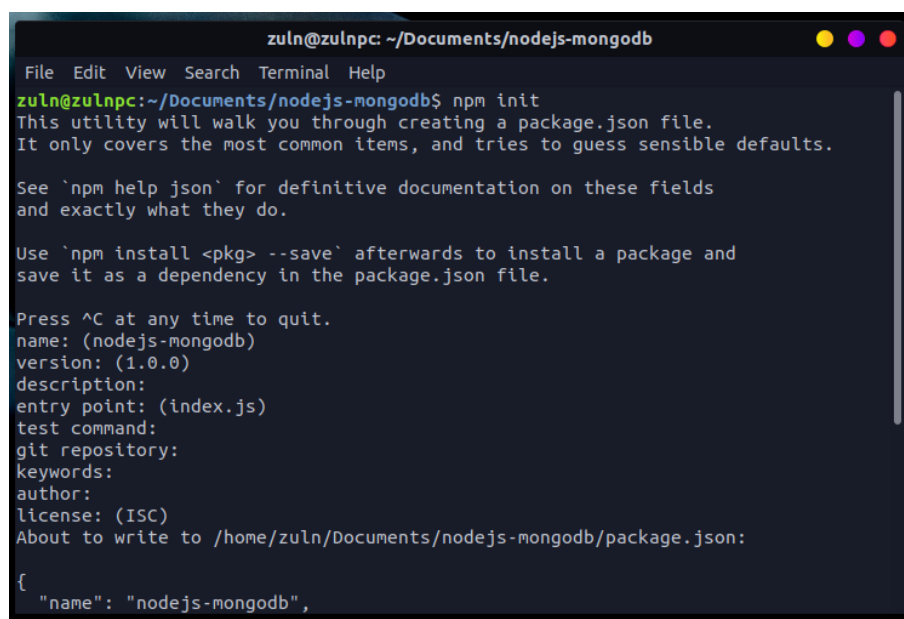
Ngôn ngữ lập trình NodeJS được sử dụng. Nhiệm vụ của server là kết nối với MongoDB để truy xuất dữ liệu bằng các phương thức GET/POST của giao thức HTTP.

#### 4.2.2.1 Cài đặt NodeJS và Tạo Project

Yêu cầu: hệ điều hành đã cài đặt NodeJS.

Tiến hành tạo thư mục chứa project có tên **nodejs-mongodb**.

Mở Terminal và gõ **npm init** để tạo project nodejs.



```
zuln@zulnc: ~/Documents/nodejs-mongodb
File Edit View Search Terminal Help
zuln@zulnc:~/Documents/nodejs-mongodb$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg> --save` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
name: (nodejs-mongodb)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to /home/zuln/Documents/nodejs-mongodb/package.json:
{
  "name": "nodejs-mongodb",
```

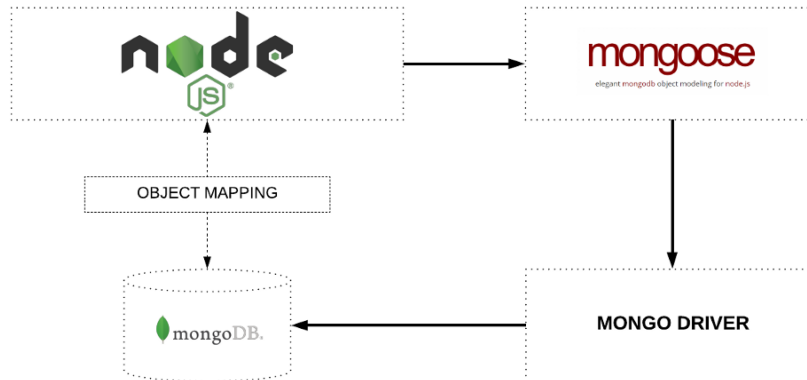
Hình 4.5: Tạo project nodejs.

#### 4.2.2.2 Cài đặt module mongoose

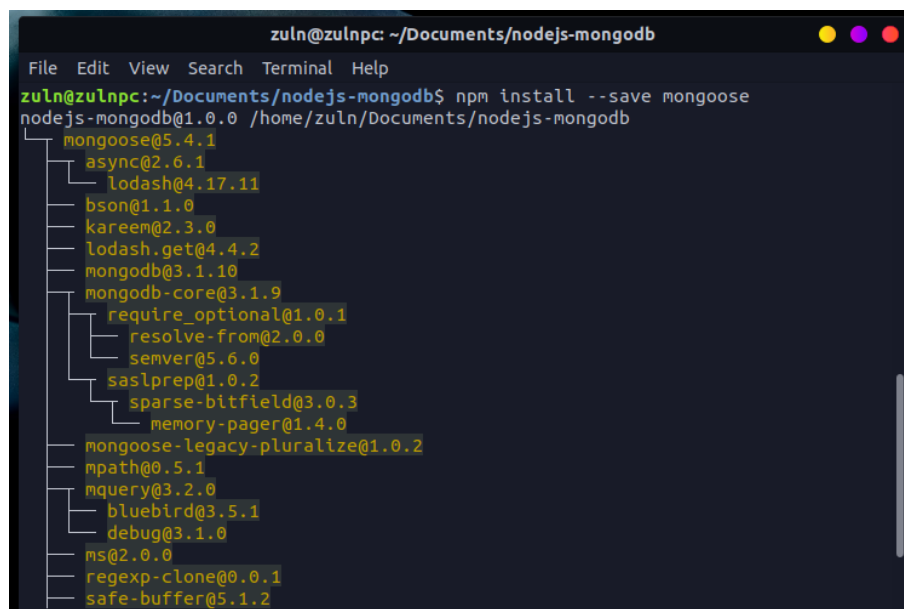
Mongoose là một thư viện mô hình hóa đối tượng (Object Data Model - ODM) cho MongoDB và Node.js. Nó quản lý mối quan hệ giữa dữ liệu, cung cấp sự xác nhận giản đồ và được sử dụng để dịch giữa các đối tượng trong mã và biểu diễn các đối tượng trong MongoDB. Gõ **npm install --save mongoose**

#### 4.2.2.3 Cài đặt module express

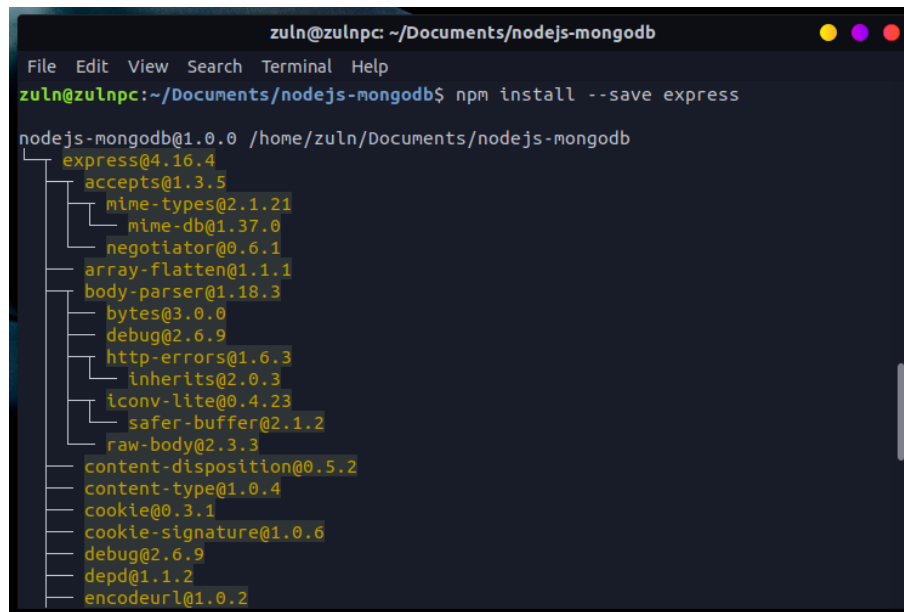
Express framework được cài đặt và được sử dụng để tạo một máy chủ web. Gõ **npm install --save express**



Hình 4.6: Mô hình mongoose, nodejs, mongodb.



Hình 4.7: Cài đặt module mongoose.



Hình 4.8: Cài đặt module express.

#### 4.2.2.4 Tạo web server dùng express

Trong thư mục project tạo file **app.js**

```
var express = require('express'), //Import the express module

app = express(),

port = process.env.PORT || 8000,

mongoose = require('mongoose'), //Import the mongoose module

Tag = require('./restapi/models/tagModel'),

bodyParser = require('body-parser'); //Import the body-parser module

// Get Mongoose to use the global promise library
mongoose.Promise = global.Promise;

//Set up default mongoose connection
mongoose.connect('mongodb://localhost/rfid');

app.use(bodyParser.urlencoded({ extended: true }));

app.use(bodyParser.json());

var routes = require('./restapi/routes/tagRoutes');

routes(app);
```



```
app.use(function(req, res) {  
    res.status(404).send({url: req.originalUrl + ' not found'})  
});  
  
app.listen(port);  
  
console.log('rfid - RESTful web services with Nodejs started on: ' + port);
```

---

Tạo thư mục **restapi**. Trong thư mục **restapi** tạo các thư mục con **controller**, **models** và **routes**.

### 4.2.2.5 Tạo file Controller, Model và Routes

Trong thư mục **models** tạo file **tagModel.js** để tạo các Schema cho tag.

---

```
'use strict';  
  
var mongoose = require('mongoose'); //Require Mongoose  
  
//Define a schema  
var Schema = mongoose.Schema;  
  
var TagSchema = new Schema({  
    Ant_Id: { type: String, required: 'Ant_Id cannot be left blank.' },  
    PC: { type: String, required: 'PC cannot be left blank.' },  
    EPC: { type: String, required: 'EPC cannot be left blank.', index: {unique: true}},  
    Date: {type: Date, required: true, default: Date.now }  
});  
// Compile model from schema  
module.exports = mongoose.model('Tags', TagSchema);  
  
//Define a schema  
var LogSchema = new Schema({  
    Ant_Id: { type: String, required: 'Ant_Id cannot be left blank.' },  
    PC: { type: String, required: 'PC cannot be left blank.' },  
    EPC: { type: String, required: 'EPC cannot be left blank.' },
```

```
Date: {type: Date, required: true, default: Date.now }
});
// Compile model from schema
module.exports = mongoose.model('Logs', LogSchema);
```

---

Trong đó, AntID, PC, EPC, Date là những thông tin cần có trong tag database.

Trong thư mục **controller** tạo file **tagController.js** để quản lý các phương thức truy xuất database.

---

```
'use strict';

var mongoose = require('mongoose');

var Tag = mongoose.model('Tags');

var Log = mongoose.model('Logs');

exports.tags = function(req, res) {
  Tag.find({}, function(err, tag) {
    if (err)
      res.send(err);
    res.json(tag);
  });
};

exports.add = function(req, res) {
  var new_tag = new Tag(req.body);
  new_tag.save(function(err, tag) {
    if (err)
      res.send(err);
    res.json(tag);
  });
};

exports.gettag = function(req, res) {
  Tag.findById(mongoose.Types.ObjectId(req.query.tagId), function(err, tag) {
    if (err)
      res.send(err);
    res.json(tag);
  });
};

exports.update = function(req, res) {
  var id = mongoose.Types.ObjectId(req.query.tagId);
  Tag.findOneAndUpdate({_id: id}, req.body, {new: true}, function(err, tag) {
    if (err)
```

```

        res.send(err);
        res.json(tag);
    });
};

exports.delete = function(req, res) {
    var id = mongoose.Types.ObjectId(req.query.tagId);
    Tag.remove({
        _id: id
    }, function(err, tag) {
        if (err)
            res.send(err);
        res.json({ message: 'Tag deleted successfully' });
    });
};

////////for logs

exports.logs = function(req, res) {
    Log.find({}, function(err, log) {
        if (err)
            res.send(err);
        res.json(log);
    });
};

exports.add_log = function(req, res) {
    var new_log = new Log(req.body);
    new_log.save(function(err, log) {
        if (err)
            res.send(err);
        res.json(log);
    });
};

exports.getlog = function(req, res) {
    Log.findById(mongoose.Types.ObjectId(req.query.logId), function(err, log) {
        if (err)
            res.send(err);
        res.json(log);
    });
};

exports.update_log = function(req, res) {
    var log_id = mongoose.Types.ObjectId(req.query.logId);
    Log.findOneAndUpdate({_id: log_id}, req.body, {new: true}, function(err,
        log) {

```

```
    if (err)
        res.send(err);
    res.json(log);
  });
};

exports.delete_log = function(req, res) {
  var log_id = mongoose.Types.ObjectId(req.query.logId);
  Log.remove({
    _id: log_id
  }, function(err, log) {
    if (err)
        res.send(err);
    res.json({ message: 'Tag deleted successfully' });
  });
};
```

---

Trong thư mục **routes** tạo file **tagRoutes.js** để quản lý các đường dẫn dùng để truy xuất database.

---

```
'use strict';

module.exports = function(app) {

  var tag = require('../controllers/tagController');

  var log = require('../controllers/tagController');

  app.route('/tags')

    .get(tag.tags)

    .post(tag.add);

  app.route('/tags/:tagId')

    .get(tag.gettag)

    .put(tag.update)

    .delete(tag.delete);

  app.route('/logs')

    .get(log.logs)

    .post(log.add_log);

  app.route('/logs/:logId')
```

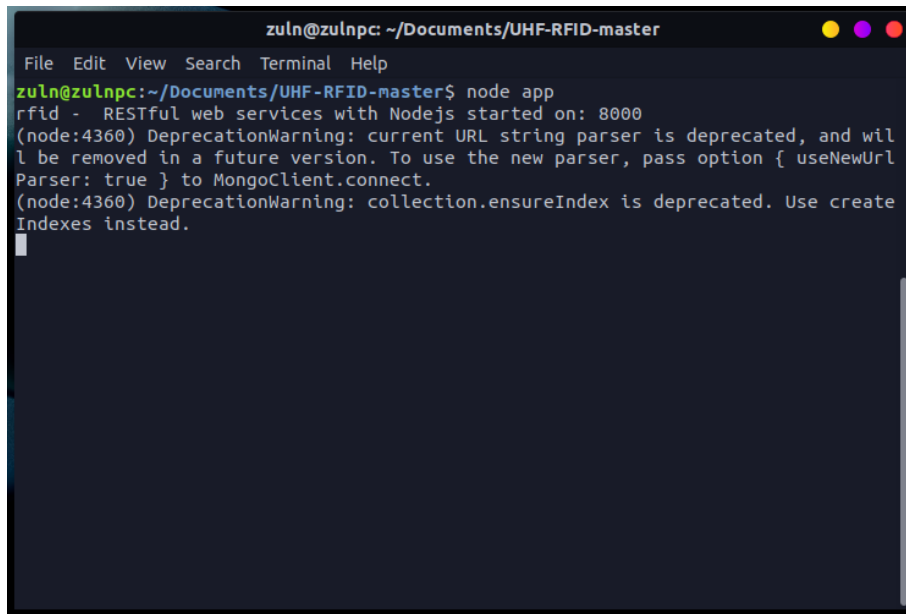
```
.get(log.getlog)

.put(log.update_log)

.delete(log.delete_log);
};
```

---

Trong thư mục chứa project, mở Terminal và gõ **node app.js** để khởi chạy ứng dụng.



Hình 4.9: Khởi chạy ứng dụng NodeJS.

### 4.2.3 Tạo ứng dụng web hiển thị trực quan MongoDB

Module được sử dụng **express-generator**.

Tiến hành update Express và Express-generator, mở terminal và gõ **npm update -g express**.

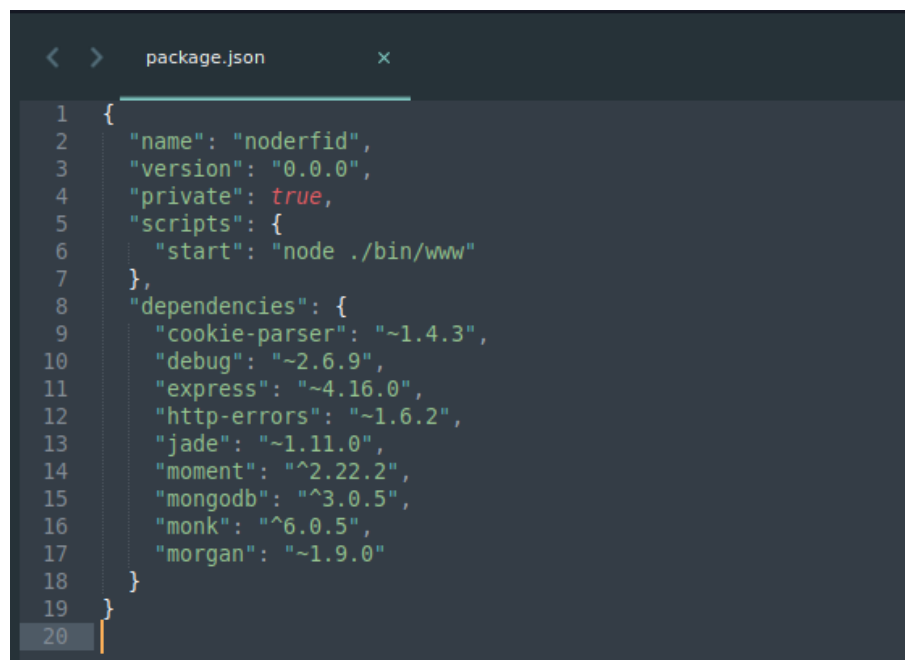
Gõ tiếp **npm update -g express-generator**.

Cuối cùng, gõ **express uhf-rfid-web** để tạo project express. Trong file **package.json** chỉnh sửa để thêm một số module cần cho MongoDB.

Mở thư mục **views**, bắt đầu với file **layout.jade**

---

```
doctype html
html
  head
    title= title
    meta(name='viewport', content='width=device-width, initial-scale=1')
```



Hình 4.10: File package.json.

```
link(rel='shortcut icon', type='image/png' href='/images/database.png')
link(rel='stylesheet', href='/stylesheets/style.css')
body
  block content
  script(src='http://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js')
  script(src='/javascripts/global.js')
```

---

Mở `index.js` để tạo web chính

---

`extends layout`

`block content`

`h1= title`

`h3 Welcome to our test`

`// Wrapper`

`#wrapper`

`// TAG INFO`

`#tagInfo`

`h2 Tag Info`

`p`

`strong Id:`

`| <span id='tagInfoId'></span>`

`br`

`strong AntennaId:`

`| <span id='tagInfoAntennaId'></span>`

```
    br
    strong PC:
    | <span id='tagInfoPc'></span>
    br
    strong EPC:
    | <span id='tagInfoEpc'></span>
    br
    strong Date:
    | <span id='tagInfoDate'></span>
// /TAG INFO

// TAG LIST

#tagList
h2 Tag List
table
  thead
    th ID
    // th ANTENNA ID
    // th PC
    th EPC
    // th DATE
    th DELETE?
  tbody
// /TAG LIST

// /FOOTER
// #footer
//   h5 This project
// /WRAPPER
```

---

Mở app.js và chỉnh sửa như sau, app.js sẽ vận hành tất cả các thao tác để tạo web hiển thị thông tin từ database cho người dùng.

---

```
var createError = require('http-errors');
var express = require('express');
var path = require('path');
var cookieParser = require('cookie-parser');
var logger = require('morgan');

// Database
var mongo = require('mongodb');
var monk = require('monk');
var db = monk('localhost:27017/rfid');
var indexRouter = require('./routes/index');
var tagsRouter = require('./routes/tags');
var logsRouter = require('./routes/logs');
```

```
var app = express();

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'jade');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

// Make our db accessible to our router
app.use(function(req, res, next){
  req.db = db;
  next();
});

app.use('/', indexRouter);
app.use('/tags', tagsRouter);
app.use('/logs', logsRouter);

// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render the error page
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;
```

---

Trong thư mục **routes**, mở file **index.js**, **index.js** sẽ chứa đường dẫn truy cập vào web hiển thị thông tin.

---

```
var express = require('express');
var router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'UHF RFID' });
});
```



```
});  
router.get('/logs', function(req, res, next) {  
  res.render('index_logs', { title: 'UHF RFID LOGS' });  
});  
  
module.exports = router;
```

---

Trong thư mục public/javascripts, tạo file global.js. Trong đây sẽ chứa các function truy cập database và truy xuất dữ liệu.

---

```
// Taglist data array for filling in info box  
var tagListData = [];  
  
// DOM Ready =====  
$(document).ready(function() {  
  
  // Populate the tag table on initial page load  
  populateTable();  
  // Tag link click  
  $('#tagList table tbody').on('click', 'td a.linkshowtag', showTagInfo);  
});  
  // Add User button click  
  $('#btnAddTag').on('click', addTag);  
  // Delete User link click  
  $('#tagList table tbody').on('click', 'td a.linkdeletetag', deleteTag);  
  
// Functions =====  
  
// Fill table with data  
function populateTable() {  
  
  // Empty content string  
  var tableContent = '';  
  
  // jQuery AJAX call for JSON  
  var express = require('express');  
  var router = express.Router();  
  
  /* GET home page. */  
  router.get('/', function(req, res, next) {  
    res.render('index', { title: 'UHF RFID' });  
  });  
  router.get('/logs', function(req, res, next) {  
    res.render('index_logs', { title: 'UHF RFID LOGS' });  
  });  
  
  module.exports = router;  
  $.getJSON( '/tags/taglist', function( data ) {  
    // Stick our user data array into a taglist variable in the global object
```

```

tagListData = data;
    // For each item in our JSON, add a table row and cells to the content
    string
    $.each(data, function(){
        tableContent += '<tr>';
        tableContent += '<td><a href="#" class="linkshowtag" rel="' + this._id +
            '">' + this._id + '</a></td>';
        // tableContent += '<td>' + this.Ant_Id + '</td>';
        // tableContent += '<td>' + this.PC + '</td>';
        tableContent += '<td>' + this.EPC + '</td>';
        // tableContent += '<td>' + this.Date + '</td>';
        tableContent += '<td><a href="#" class="linkdeletetag" rel="' + this._id
            + '">DELETE</a></td>';
        tableContent += '</tr>';
    });

    // Inject the whole content string into our existing HTML table
    $('#tagList table tbody').html(tableContent);
});
};

// Show Tag Info
function showTagInfo(event) {

    // Prevent Link from Firing
    event.preventDefault();

    // Retrieve username from link rel attribute
    var thisTagName = $(this).attr('rel');

    // Get Index of object based on id value
    var arrayPosition = tagListData.map(function(arrayItem) { return
        arrayItem._id; }).indexOf(thisTagName);
    // Get our Tag Object
    var thisTagObject = tagListData[arrayPosition];
    //Handle Date Data
    var date = new Date(thisTagObject.Date);
    var a = date.toDateString() + ' ' + date.toTimeString();

    //Populate Info Box
    $('#tagInfoId').text(thisTagObject._id);
    $('#tagInfoAntennaId').text(thisTagObject.Ant_Id);
    $('#tagInfoPc').text(thisTagObject.PC);
    $('#tagInfoEpc').text(thisTagObject.EPC);
    $('#tagInfoDate').text(a);
    // $('#tagInfoDate').text(thisTagObject.Date);
};

// Add Tag

```

```
function addTag(event) {
    event.preventDefault();

    // Super basic validation - increase errorCount variable if any fields are
    // blank
    var errorCount = 0;
    $('#addTag input').each(function(index, val) {
        if($(this).val() === '') { errorCount++; }
    });

    // Check and make sure errorCount's still at zero
    if(errorCount === 0) {

        // If it is, compile all user info into one object
        var newTag = {
            'Ant_Id': $('#addTag fieldset input#inputTagAntennaId').val(),
            'PC': $('#addTag fieldset input#inputTagPC').val(),
            'EPC': $('#addTag fieldset input#inputTagEPC').val(),
        }

        // Use AJAX to post the object to our adduser service
        $.ajax({
            type: 'POST',
            data: newTag,
            url: '/tags/addtag',
            dataType: 'JSON'
        }).done(function( response ) {

            // Check for successful (blank) response
            if (response.msg === '') {

                // Clear the form inputs
                $('#addUser fieldset input').val('');

                // Update the table
                populateTable();

            }
            else {

                // If something goes wrong, alert the error message that our service
                // returned
                alert('Error: ' + response.msg);

            }
        });
    }
    else {
        // If errorCount is more than 0, error out
    }
}
```

```
        alert('Please fill in all fields');
        return false;
    }
};

// Delete Tag
function deleteTag(event) {

    event.preventDefault();

    // Pop up a confirmation dialog
    var confirmation = confirm('Are you sure you want to delete this tag?');

    // Check and make sure the user confirmed
    if (confirmation === true) {

        // If they did, do our delete
        $.ajax({
            type: 'DELETE',
            url: '/tags/deletetag/' + $(this).attr('rel')
        }).done(function( response ) {

            // Check for a successful (blank) response
            if (response.msg === '') {
            }
            else {
                alert('Error: ' + response.msg);
            }

            // Update the table
            populateTable();

        });

    }
    else {

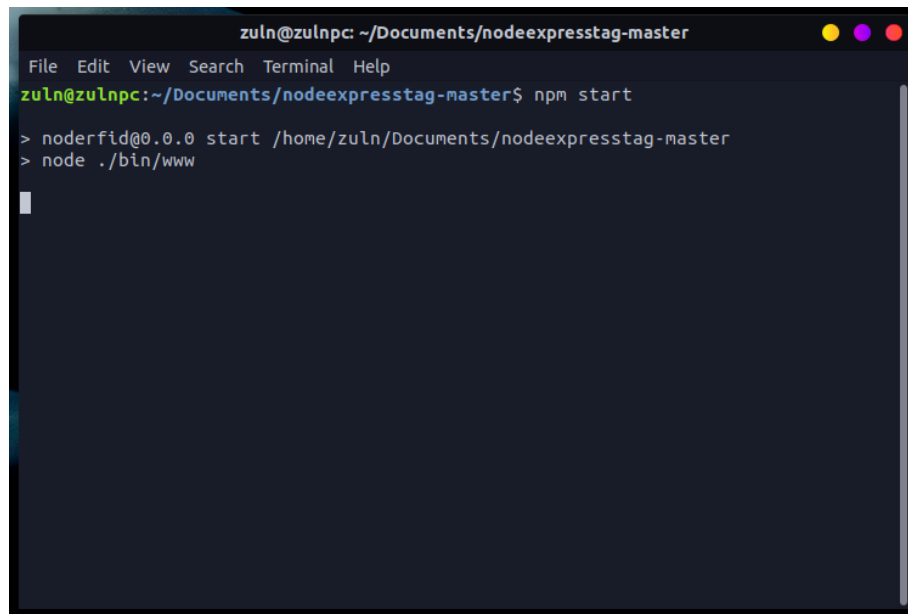
        // If they said no to the confirm, do nothing
        return false;

    }

};
```

---

Để chạy ứng dụng web, mở Terminal và chạy **npm start**



```
zuln@zulnpc: ~/Documents/nodeexpresstag-master
File Edit View Search Terminal Help
zuln@zulnpc:~/Documents/nodeexpresstag-master$ npm start
> nodelfid@0.0.0 start /home/zuln/Documents/nodeexpresstag-master
> node ./bin/www
```

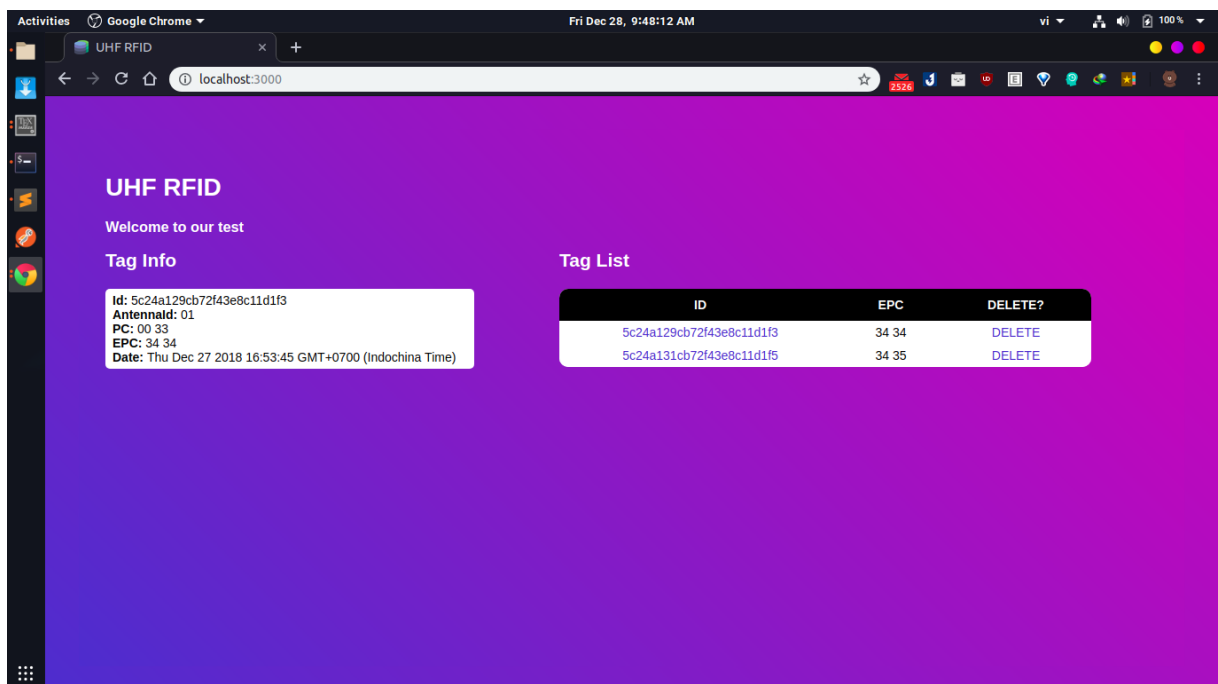
Hình 4.11: Chạy ứng dụng web.

# Chương 5

## Kết quả vận hành thực tế

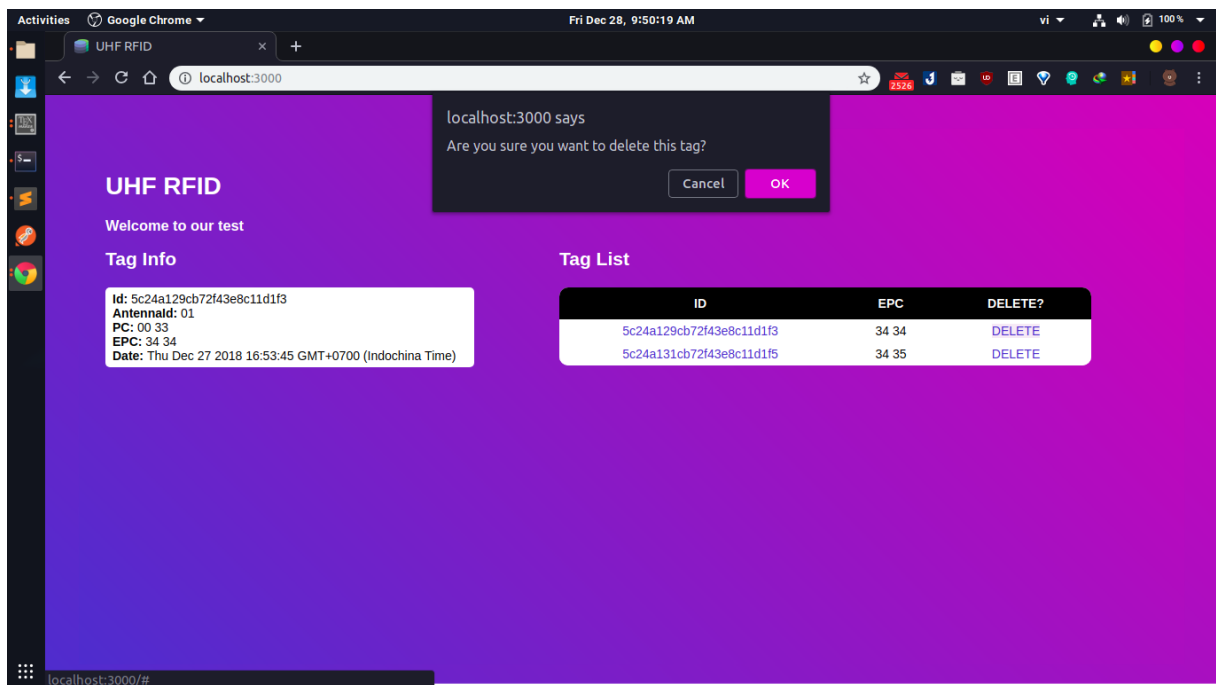
Truy cập **http://localhost:3000/** vào giao diện web.

Người dùng có thể CLICK vào Id để xem các thông tin trong tag.



Hình 5.1: Giao diện web tại địa chỉ **http://localhost:3000/**.

Người dùng có thể Click vào DELETE để xóa trực tiếp document.



Hình 5.2: Xóa trực tiếp document.

# Chương 6

## Đánh giá và Hướng phát triển

### 6.1 Đánh giá

#### 6.1.1 Ưu điểm

- Ứng dụng đáp ứng được các yêu cầu cơ bản của hệ thống UHF RFID.
- Giao diện web quan sát trực quan, thẩm mỹ.

#### 6.1.2 Khuyết điểm

- Độ trễ chưa đạt yêu cầu.
- Ứng dụng web chưa realtime.
- Ứng dụng web chỉ mới truy cập được trên phía localhost, chưa public trên Internet.
- Ứng dụng chưa có giao diện đăng nhập, bảo mật thông tin.

### 6.2 Hướng phát triển

Các hướng phát triển:

- Tạo giao diện web hoàn chỉnh, gồm: đăng nhập, đăng kí, xác thực người dùng, tối ưu truy xuất dữ liệu,...
- Đẩy ứng dụng lên Internet giúp người dùng dễ truy cập.



# Tài liệu tham khảo

- [1] *How to create a nodejs mongodb rest api and test with postman*, URL: <http://programmerblog.net/nodejs-mongodb-rest-api/>
- [2] *Creating a Simple RESTful Web App with Node.js, Express, and MongoDB*, URL: <https://closebrace.com/tutorials/2017-03-02/creating-a-simple-restful-web-app-with-nodejs-express-and-mongodb>
- [3] *Công nghệ RFID là gì ? lịch sử phát triển của RFID*, URL: <http://smartid.com.vn/cong-nghe-rfid-la-gi-lich-su-phat-trien-cua-rfid.html>
- [4] *Ứng dụng và ưu điểm của công nghệ RFID UHF*, URL: <http://vn.rfidtagcn.com/info/application-and-advantages-of-rfid-uhf-technol-27459131.html>