THỬ NGHIỆM MỘT HỆ THỐNG ĐỊNH VỊ TRONG NHÀ SỬ DỤNG WI-FI VÀ THUẬT TOÁN K-LÁNG GIỀNG GẦN NHẤT

***Lê Đinh Phú Cường*[1]**
***Phạm Hồng Xuân*[2]**

TÓM TẮT

Nghiên cứu này đề cập đến định vị trong nhà sử dụng chuẩn không dây IEEE 802.11 (Wi-Fi) dựa trên thuật toán máy học k-NN (k-Nearest Neighbours) đáp ứng giảm chi phí so với các công nghệ không dây trong nhà khác. Mục đích của nghiên cứu này là kiểm tra một số vấn đề việc định vị vị trí trong nhà dựa trên dấu vết vị trí ảnh hưởng đến độ chính xác định vị đạt được tại khu vực thí nghiệm. Các tác động của hành vi của con người đối với phân bố RSSII được khảo sát và phân tích. Hệ thống có thể phát triển trong nhà cho dịch vụ bãi đậu xe thông minh trong tương lai.

***Từ khóa**: Wi-Fi, RSSI, k-NN (k-Nearest Neighbours), back-end, NoSQL, Redis, Postman, JSON, Chanalyzer 4*

1. Giới thiệu

Công nghệ định vị trong nhà là một trong các công trình nghiên cứu quan trọng trong thời đại cách mạng công nghiệp 4.0. Trong đời sống xã hội ngày nay, việc sử dụng điện thoại thông minh và các dịch vụ dựa trên vị trí đã và đang góp phần thúc đẩy sự phát triển của các hệ thống định vị. Trước đây, các tín hiệu GPS (Global Positioning System) bị hạn chế bởi các vật liệu bê tông xây dựng tạo nên kết quả ảnh hưởng không khả thi cho việc xác định vị trí trong nhà. Ngày nay, xác định vị trí trong nhà bằng cách sử dụng Wireless Ethernet IEEE 802.11 (Wi-Fi) được bố trí theo nhiều mô hình tương đối bao phủ toàn bộ trong nhà và được sử dụng phổ biến ở mọi nơi trong khu vực công cộng. Wi-Fi đáp ứng để thay thế những hạn chế của hệ thống định vị toàn cầu (GPS) ở trung tâm thành phố hoặc trong môi trường trong tòa nhà có diện tích lớn hoặc nhỏ. Tín hiệu Wi-Fi được dùng cho việc triển khai hạ tầng ít tốn kém về mặt chi phí và cả về mặt thời gian. Đặc biệt, vị trí trong nhà dựa trên dấu vết vị trí chỉ liên quan đến đặc điểm cường độ tín hiệu nhận được (RSSI). Quan trọng hơn, sơ đồ định vị vị trí dựa trên Wi-Fi không có yêu cầu về đầu tư thêm phần cứng chuyên dụng.

Các hệ thống Wi-Fi được phát triển với phạm vi tín hiệu, khả năng truyền thông và tốc độ dữ liệu có thể được tạo ra trong nhiều mô hình lan truyền sóng vô tuyến. Hơn nữa, mối quan hệ giữa các giá trị RSSI từ nhiều Wi-Fi được phân biệt rõ ràng. Để hiểu và mô hình hiệu suất của các hệ thống định vị trong nhà dựa trên dấu vết vị trí, bài nghiên cứu này thu thập các giá trị RSSI, độ lệch tiêu chuẩn, sự biến đổi theo thời gian của chúng và sự độc lập của RSSI từ nhiều điểm truy cập Wi-Fi. Nói chung, việc phân phối RSSI được phân bố bởi công suất phát sóng (đơn vị được tính theo dBm) qua nghiên cứu trong thí nghiệm này.

[1]Trường Đại học Yersin Đà Lạt
Email: ledinhphucuong.dalat@gmail.com
[2]Trường Cao Đẳng Bách Khoa Nam Sài Gòn

Quá trình truyền dẫn tín hiệu trong nhà rất khó khăn vì ảnh hưởng của môi trường đa kênh và các hiệu ứng lan truyền như phản xạ, nhiễu xạ và tán xạ.

Bên cạnh công nghệ thông tin di động và công nghệ Wi-Fi, các công nghệ tần số sóng vô tuyến tạo ra kỹ thuật lấy dấu vết RSSI cho định vị vị trí có thể so sánh với các công nghệ khác bao gồm:

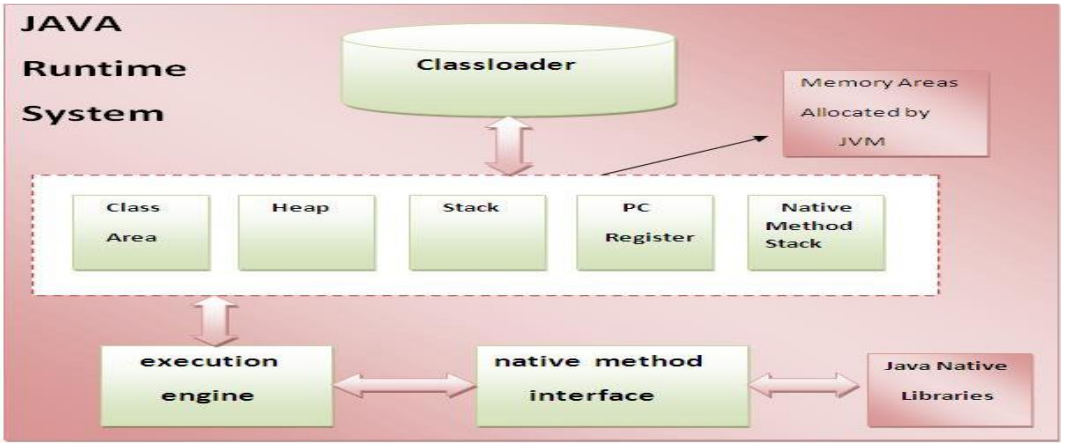
- *Bluetooth*: mặc dù có yêu cầu về kết cấu hạ tầng thấp so với Wi-Fi, nó có thể đạt được độ chính xác trong phạm vi 1m.
- *Radio* cũng có thể được sử dụng cho định vị. Tuy nhiên, yêu cầu của phần cứng chuyên dụng và thực tế là các thiết bị có thể được đặt tại trong vùng ngoại ô.

2. Môi trường phát triển hệ thống

Môi trường phát triển hệ thống được trình bày như sau: Khảo sát nhu cầu nghiên cứu thực tiễn qua việc đánh giá so sánh công nghệ hoặc các ứng dụng khác nêu trên của các công ty thiết kế hệ thống thiết bị và tích hợp sử dụng phần mềm quản lý tự động bãi đậu xe sử dụng mã nguồn đóng. Bên cạnh, chọn lựa ra được ngôn ngữ lập trình nguồn mở: Java để làm công cụ thiết kế phần mềm như là một ngôn ngữ lập trình hướng đối tượng (*OOP-Object Oriented Programming*) [1].

JAVA là một ngôn ngữ lập trình và là một Platform [2], [3]. Java là một ngôn ngữ lập trình có tính bảo mật cao, hướng đối tượng, bậc cao và mạnh mẽ. Platform: bất cứ môi trường phần cứng hoặc phần mềm nào mà trong đó một chương trình chạy thì được biết đến như một Platform. Với môi trường runtime riêng cho mình JRE và API, Java được gọi là Platform.

JVM *(viết tắt của Java Virtual Machine)* là một thiết bị trường tựu (*ảo*) có thể giúp máy tính chạy các chương trình trình Java [3]. Nó cung cấp môi trường *runtime* mà trong đó Java Bytecode có thể được thực thi. JVM là có sẵn cho nhiều nền tảng (*Windows, Linux,…*). JVM, JRE và JDK là phụ thuộc nền tảng, bởi vì cấu hình của mỗi OS (*hệ điều hành*) là khác nhau. Nhưng, Java là độc lập nền tảng. Cấu trúc JVM theo hình 1.

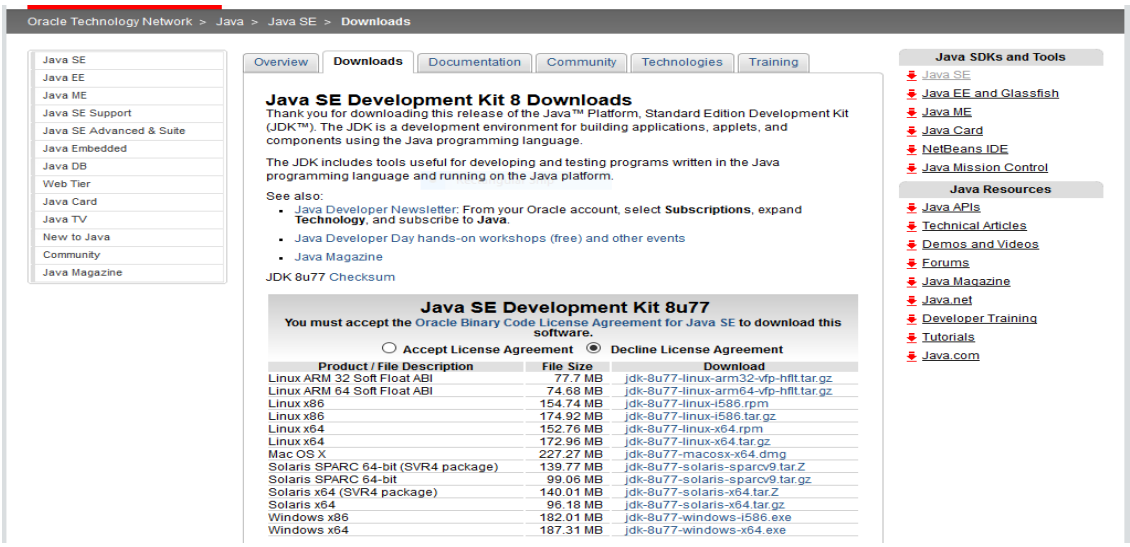


Hình 1: *Cấu trúc JVM*

2.1. Cài đặt trình biên dịch và soạn thảo mã lệnh (viết code)

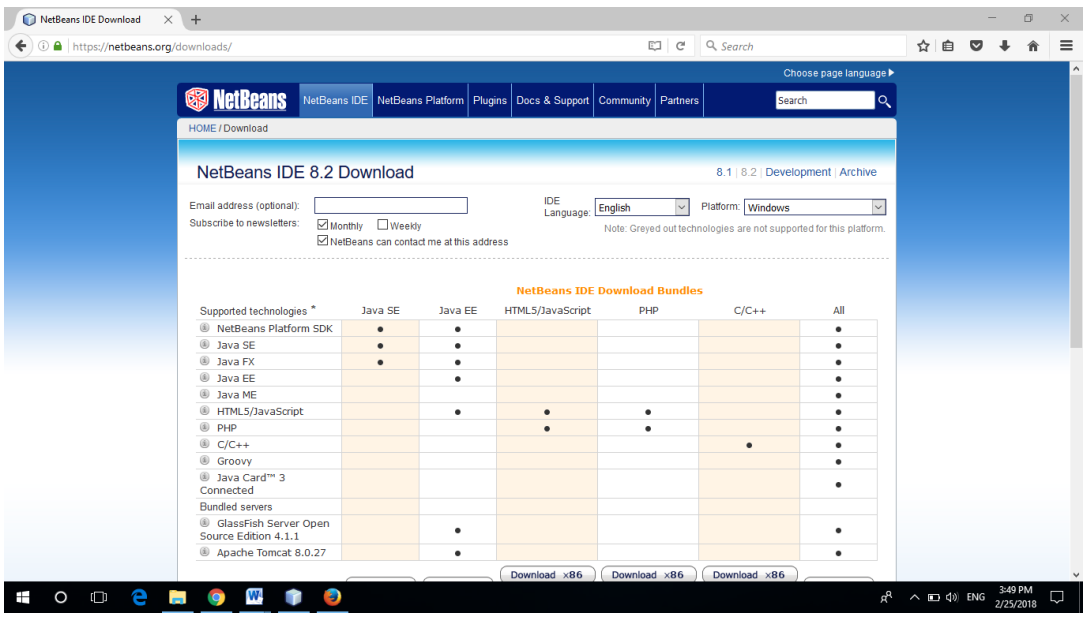
a) Tải và cài đặt trình trình biên dịch, thông dịch

Đó là Java SE Development Kit 8 (JDK), đây là bộ công cụ phát triển ứng dụng bằng ngôn ngữ lập trình Java. Trong JDK chứa các công cụ và chương trình (trình biên dịch, thông dịch…). Truy cập trang Web: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html, tiếp theo tải và cài đặt JDK 8 trên nền Windows x64 *jdk-8u77-windows-x64.exe (187.31MB),* theo hình 2 như sau:



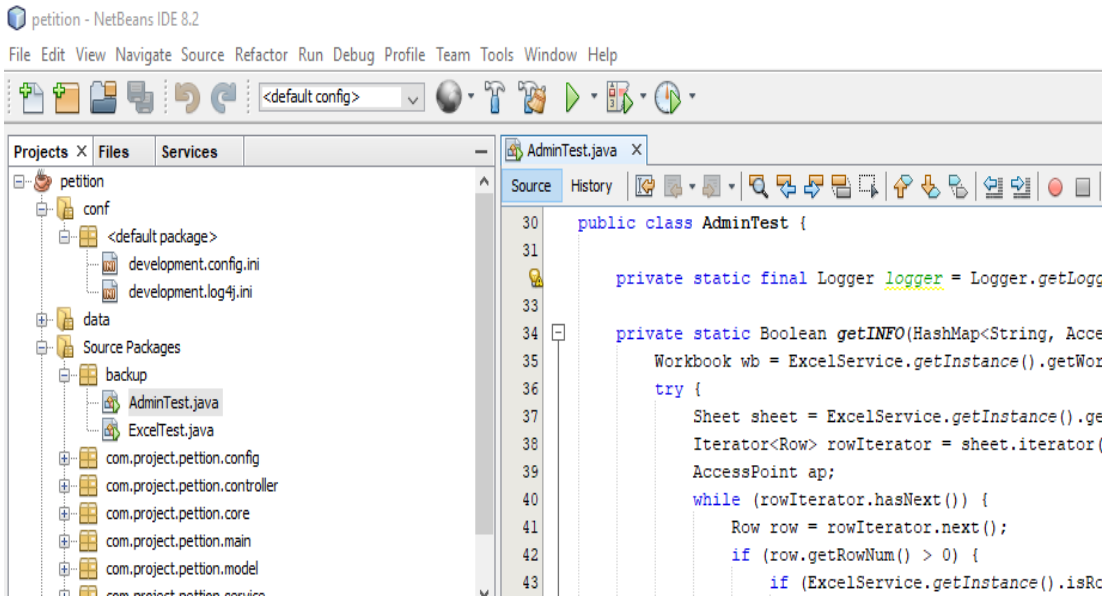
Hình 2: *Cài đặt trình biên dịch*

b) Tải trình biên dịch Netbean [4]



Hình 3: *Chương trình soạn thảo mã lệnh*

c) Viết chương trình



Hình 4: *Công cụ Netbean viết chương trình*

2.2. Ứng dụng nghiên cứu được phát triển back-end

Phát triển back-end là việc xử lý mọi logic nghiệp vụ phức tạp ở ẩn ở phía sau, giúp cho hệ thống hoạt động trơn tru. Dữ liệu của người dùng, thuật toán phân tích đều nằm ở back-end [5]. (Ví dụ: trên trang face, khi bạn post 1 status, để status ấy được lưu trữ thì cần back-end, để status ấy hiển thị cho bạn bè của bạn xem thì cũng cần back-end, để status ấy lưu những react: love, phẫn nộ,… cũng cần back-end).

a) Kỹ năng back-end: Để trở thành phát triển back-end thì cần biết ngôn ngữ phía Server cũng như biết thao tác với cơ sở dữ liệu. Ứng dụng trong nghiên cứu này thực hiện bởi ngôn ngữ *server-side* để viết *back-end*: Java.

b) Cơ sở dữ liệu: Trong bài báo này, *Server* được viết bằng ngôn ngữ Java trao đổi dữ liệu thông qua hệ quản trị cơ sở dữ liệu mang hình thức *NoSQL* qua việc xử lý dữ liệu bởi *Cache Redis* sử dụng *key-value* lưu giữ dữ liệu. Việc lưu trữ dữ liệu trên *RAM* mang lại khả năng truy cập dữ liệu với tốc độ cao, cập nhật và loại bỏ dữ liệu nhanh chóng. Sau khi tương và xử lý dữ liệu, *Server* trả về *API* theo chuẩn *Java Servlet API* dưới dạng *JSON* cho *Client* xử lý trên giao diện. Hiện tại ứng dụng dừng lại ở việc lấy kết quả của *Server* sau khi xử lý, và sử dụng công cụ *Postman* để thực hiện việc gọi *API* từ *Server* và kiểm tra (*testing*) trả về kết quả trên giao diện *Postman*.

- *NoSQL* là một khái niệm chỉ về một lớp các hệ cơ sở dữ liệu không sử dụng mô hình quan hệ (*RDBMS*) [6]. *RDBMS* vốn tồn tại khá nhiều nhược điểm như có hiệu năng không tốt nếu

kết nối dữ liệu nhiều bảng lại hay khi dữ liệu trong một bảng là rất lớn. Có các loại *NoSQL* cơ bản: Key–value data stores (***Redis***, Dynomite, MemcacheDB), Document–based (Apache CouchDB, MongoDB), Graph–based data-stores (Neo4j, InfiniteGraph, DEX). NoSQL có các đặc điểm sau:

o NoSQL lưu trữ dữ liệu của mình theo dạng cặp giá trị "key–value". Sử dụng số lượng lớn các node để lưu trữ thông tin.

o Chấp nhận dữ liệu bị trùng lặp do một số node sẽ lưu cùng thông tin giống nhau.

o Phi quan hệ - không có ràng buộc nào cho việc nhất quán dữ liệu.

o Có hiệu suất cao (high performance) và tính sẵn sang cao (high availability).

- *Redis* là một hệ thống lưu trữ *key-value* rất mạnh mẽ và phổ biến hiện nay [7]. Redis nổi bật bởi việc hỗ trợ nhiều cấu trúc dữ liệu cơ bản (*hash, list, set, sorted set, string*) giúp việc thao tác với dữ liệu tốt hơn các hệ thống cũ như *memcached* rất nhiều. Bên cạnh lưu trữ *key-value* trên RAM giúp tối ưu hiệu suất, *redis* còn có cơ chết sao lưu dữ liệu trên đĩa cứng cho phép phục hồi dữ liệu khi gặp sự cố.

✓ Redis là một lựa chọn khi cần đến một server lưu trữ dữ liệu đòi hỏi tính mở rộng cao (*scaleable*) và chia sẻ bởi nhiều tiến trình, nhiều ứng dụng và nhiều server khác nhau. Chỉ riêng cơ chế tương tác giữa các tiến trình đã khó khăn. Do đó, việc có thể tương tác *cross-platform, cross-server,* và *cross-application* đã làm Redis trở thành một lựa chọn cho thực hiện nhiều công việc khác nhau. Tốc độ cực cao của Redis cũng có thể để làm *caching layer*.

- *Postman* là một *App Extensions*, cho phép làm việc với các *API*, nhất là *REST*, giúp hình rất nhiều cho việc testing [8]. Hỗ trợ tất cả các phương thức *HTTP* (*GET, POST, PUT*...). Postman cho phép lưu lại các lần sử dụng. Sử dụng nó khá đơn giả, chỉ cần điền *URL* của *API*, chọn phương thức, thêm tham số cần thiết và nhấn *Send*.

- *JSON (JavaScript Object Noattion)* là một định dạng hoán vị dữ liệu nhanh [8]. Chúng dễ dàng cho chúng ta đọc và viết. Dễ dàng cho thiết bị phân tích và phát sinh. Cơ sở dựa trên tập hợp của ngôn ngữ lập trình *JavaScript*, tiêu chuẩn *ECMA-262*. *JSON* là một định dạng kiểu text mà hoàn toàn độc lập với các ngôn ngữ hoàn chỉnh, thuộc họ hàng với các họ hàng C, gồm có *C, C++, C#, Java, JavaScript*… Những đặc tính đó đã tạo nên *JSON* một ngôn ngữ hoán vị dữ liệu lý tưởng. JSON được xây dựng trên 2 cấu trúc:

✓ Tập hợp của các cặp tên và giá trị *name-value*. Trong những ngôn ngữ khác nhau, đây được nhận thấy như là 1 đối tượng (*object*), sự ghi (*record*), cấu trúc (*struct*), từ điển (*dictionary*), bảng băm (*hash table*), danh sách khóa (*keyed list*), hay mảng liên hợp.

✓ Tập hợp các giá trị đã được sắp xếp. Trong hầu hết các ngôn ngữ, *this* được nhận thấy như là một mảng, vector, tập hợp hay là 1 dãy sequence.

3. Ứng dụng

3.1. Location fingerprinting

Mô hình hệ thống định vị dựa trên dấu vết vị trí thường có 2 giai đoạn: giai đoạn ngoại tuyến (offline) và giai đoạn trực tuyến (online) hay còn gọi là giai đoạn định vị.

➢ *Bước 1*: Thiết bị máy tính xách tay (*Laptop*) sử dụng để đo các giá trị *RSSI* (đơn vị dBm) và tương ứng vị trí phân bố của nó, sau đó được sử dụng trong giai đoạn ngoại tuyến. Dữ liệu này được thu thập từ 3 điểm truy cập không dây (*AP, viết tắt Access Point*) được bố trí trong nhà tại các vị trí cần quan tâm, đây là khu vực thí nghiệm để thu thập dữ liệu từ các *AP*.

Cách thức thu nhận thông số RSSI: sử dụng công cụ "Chanalyzer 4" để scan các vị trị thu thập RSSI, chuyển thành cơ sở dữ liệu qua Microsoft Office Excel.

➢ *Bước 2*: Quá trình định vị, thiết bị *Laptop* đo lường giá trị *RSSI* ở một số vị trí không xác định trong thời gian tức thời trong giai đoạn trực tuyến. Ứng dụng của người dung thiết bị (*User*) được áp dụng một thuật toán *k-NN* đưa ra giá trí *RSSI* và vị trí tương ứng để ước tính vị trí hiện tại bằng cách so sánh sử dụng dữ liệu *RSSI* phân bố đã thu thập tại bước 1 để đưa ra một dữ liệu *RSSI* duy nhất và vị trí so với dữ liệu trước đó.

Cách thức thu nhận thông số RSSI: sử dụng công cụ "Chanalyzer 4" để scan các vị trị thu thập RSSI, sau đó gửi các thông số RSSI ghi nhận được lên Server để lưu trữ hoặc ghi ra tập tin theo cấu trúc định sẵn (Microsoft Office Excel).

3.2. Thuật toán k-Nearest Neighbours

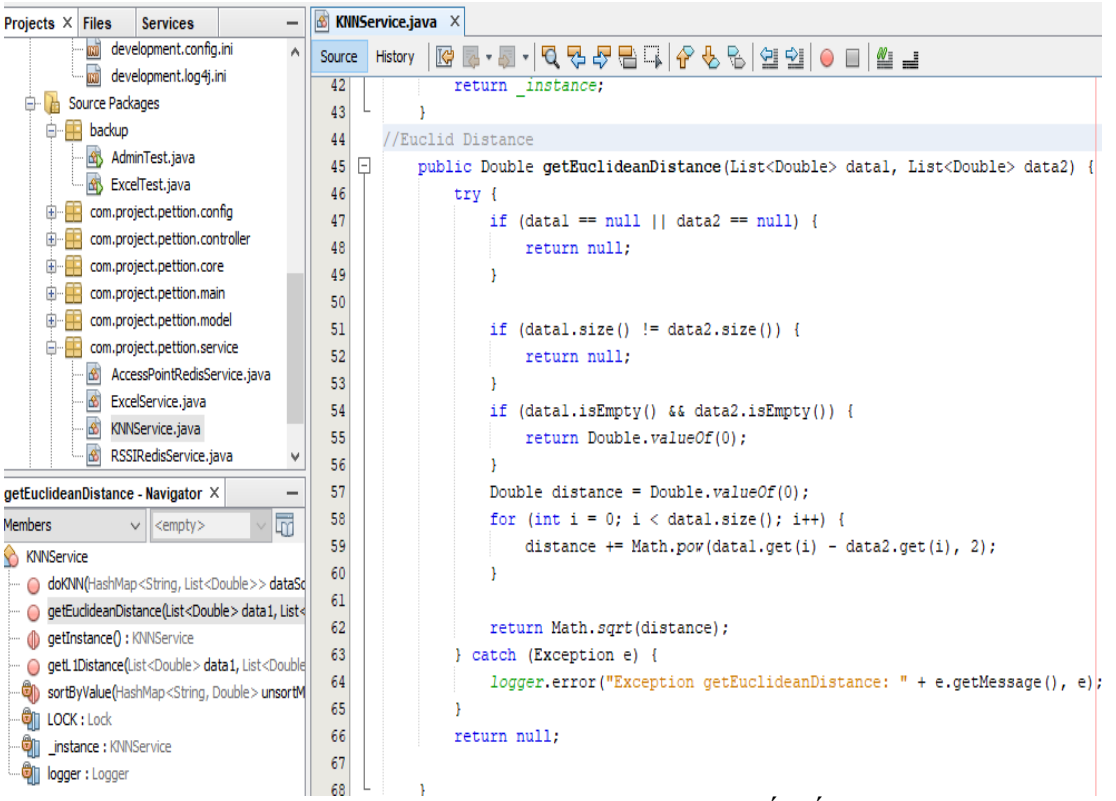
k-Nearest Neighbours (*k-NN*) là một trong những thuật toán học không giám sát (*Supervised-Learning)* đơn giản nhất trong học máy. Khi huấn luyện, thuật toán này không học bất kỳ điều gì từ dữ liệu huấn luyện, mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. *k-NN* có thể áp dụng vào cả hai loại của bài toán học không giám sát là phân loại (*Classification*) và hồi quy (*Regression*) [9].

Về nguyên tắc, có hai bước thực hiện k-láng giềng gần nhất (*k-NN*) [9]:

➢ Bước 1: chỉ số k trên bản đồ tín hiệu có các $i_1, i_2, ..., i_k$ của $r_{i1}, r_{i2}, ..., r_{ik}$ là những điểm gần nhất được tính toán bởi khoảng cách Euclid, tại thí nghiệm việc đo là không xác định là:

$$d_i = \sqrt{\sum_{k=1}^{n} (RSS_{ik} - RSS_k)^2}$$

➢ Bước 2: Lấy giá trị trung bình của k láng giềng gần nhất có khoảng cách Euclid tối thiểu theo hình 5.



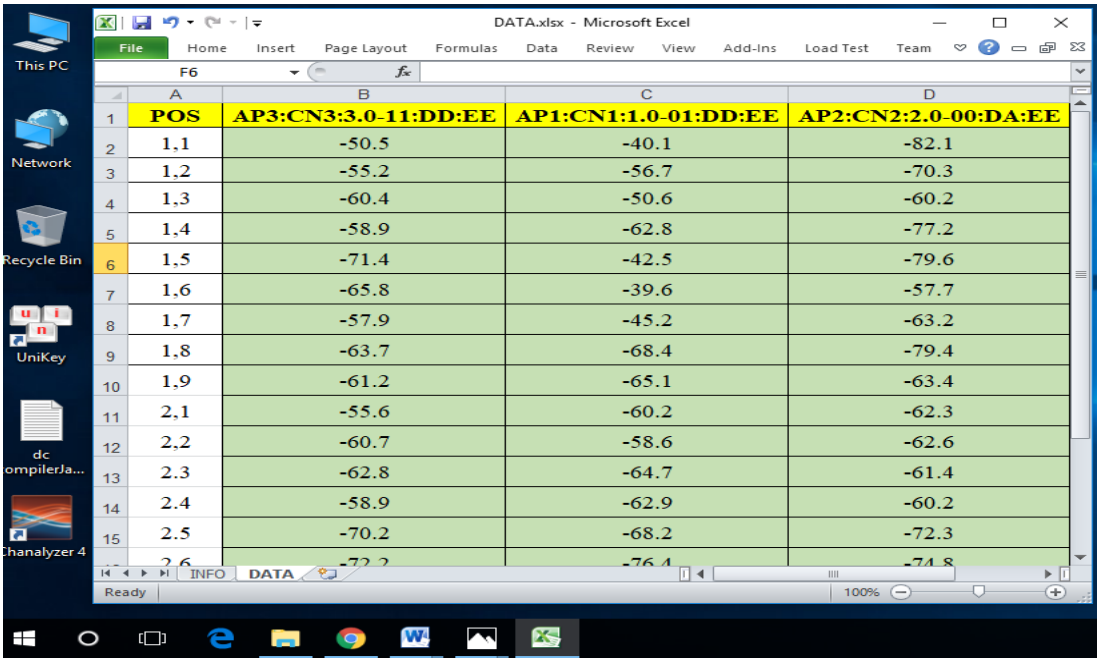
Hình 5: *Khoảng cách Euclid áp dụng trong thiết kế ứng dụng*

3.3. Ứng dụng

Trong ứng dụng nghiên cứu này, mô hình của hệ thống định vị trong nhà hoạt động trên thiết bị người dùng và dữ liệu được cập nhật trong một máy chủ *localhost* theo phát triển hình thức *back-end* để thực hiện các thí nghiệm.

Về nguyên tắc, ứng dụng hoạt động ở hai giai đoạn lấy mẫu từ xa, bao gồm giai đoạn ngoại tuyến (*offline phase*) và giai đoạn trực tuyến (*online phase*).

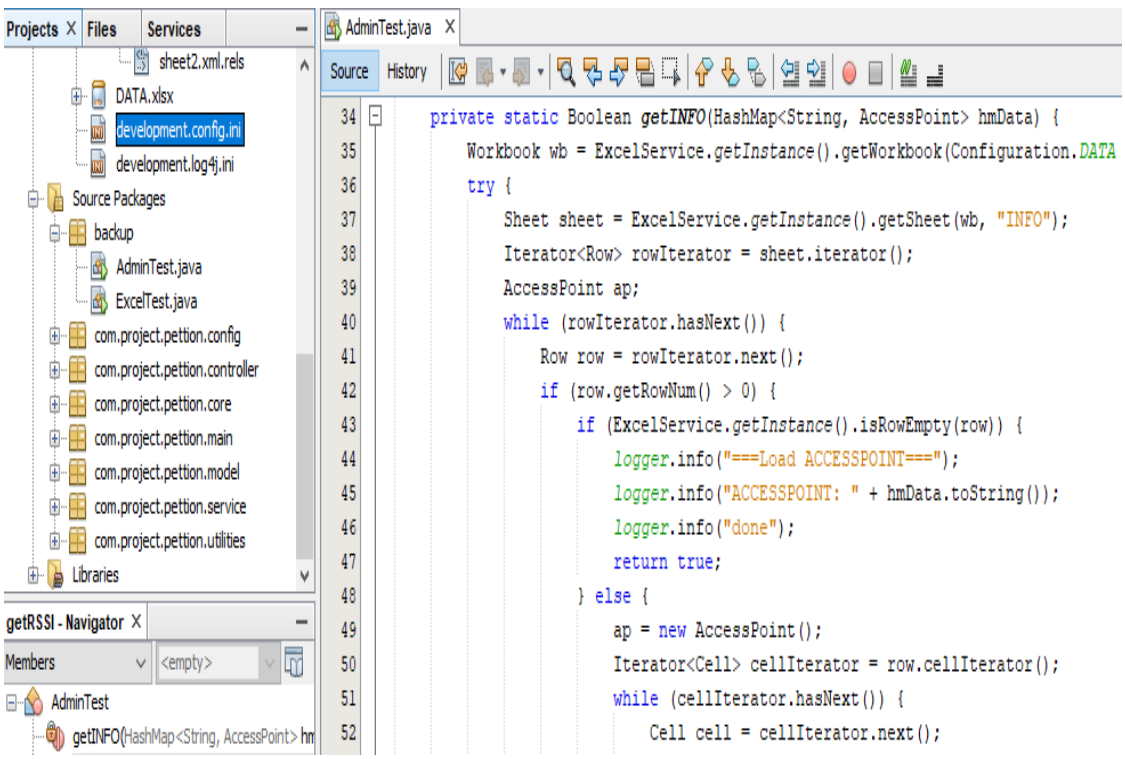
Trong giai đoạn ngoại tuyến (*offline phase*), đo và thu thập điểm truy cập Wi-Fi bởi giá trị *RSSI* ở các vị trí khác nhau trong nhà tại khu vực thí nghiệm. Trong mỗi điểm RSSI của tất cả các *AP* hiện có được thu thập trong một khoảng thời gian xác định và sau đó giá trị trung bình được ghi nhận và lưu trữ vào tập tin bảng tính ***DATA.xlsx*** (*tập tin bảng tính này phải bắt buộc theo thiết kế mẫu trong bài nghiên cứu này*) của phần mềm văn phòng *Microsoft Office Excel 2010* theo hình 6 và hình 7 thể hiện đoạn viết chương trình để thiết kế tập tin bảng tính.



DATA.xlsx - Microsoft Excel

	A	B	C	D
1	POS	AP3:CN3:3.0-11:DD:EE	AP1:CN1:1.0-01:DD:EE	AP2:CN2:2.0-00:DA:EE
2	1,1	-50.5	-40.1	-82.1
3	1,2	-55.2	-56.7	-70.3
4	1,3	-60.4	-50.6	-60.2
5	1,4	-58.9	-62.8	-77.2
6	1,5	-71.4	-42.5	-79.6
7	1,6	-65.8	-39.6	-57.7
8	1,7	-57.9	-45.2	-63.2
9	1,8	-63.7	-68.4	-79.4
10	1,9	-61.2	-65.1	-63.4
11	2,1	-55.6	-60.2	-62.3
12	2,2	-60.7	-58.6	-62.6
13	2,3	-62.8	-64.7	-61.4
14	2,4	-58.9	-62.9	-60.2
15	2,5	-70.2	-68.2	-72.3

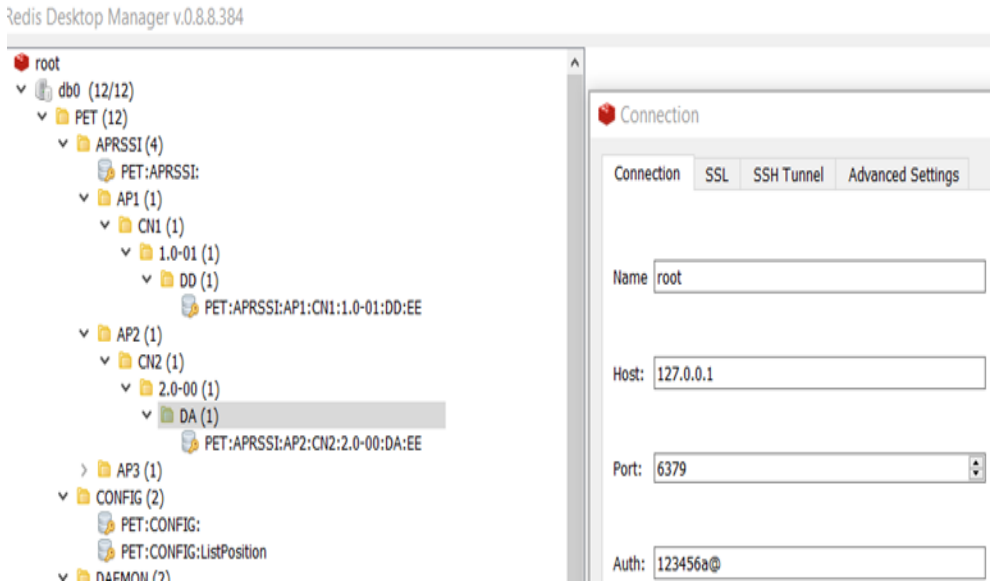
Hình 6: *Tập tin bảng tính DATA.xlsx*



```
34  private static Boolean getINFO(HashMap<String, AccessPoint> hmData) {
35      Workbook wb = ExcelService.getInstance().getWorkbook(Configuration.DATA
36      try {
37          Sheet sheet = ExcelService.getInstance().getSheet(wb, "INFO");
38          Iterator<Row> rowIterator = sheet.iterator();
39          AccessPoint ap;
40          while (rowIterator.hasNext()) {
41              Row row = rowIterator.next();
42              if (row.getRowNum() > 0) {
43                  if (ExcelService.getInstance().isRowEmpty(row)) {
44                      logger.info("===Load ACCESSPOINT===");
45                      logger.info("ACCESSPOINT: " + hmData.toString());
46                      logger.info("done");
47                      return true;
48                  } else {
49                      ap = new AccessPoint();
50                      Iterator<Cell> cellIterator = row.cellIterator();
51                      while (cellIterator.hasNext()) {
52                          Cell cell = cellIterator.next();
```

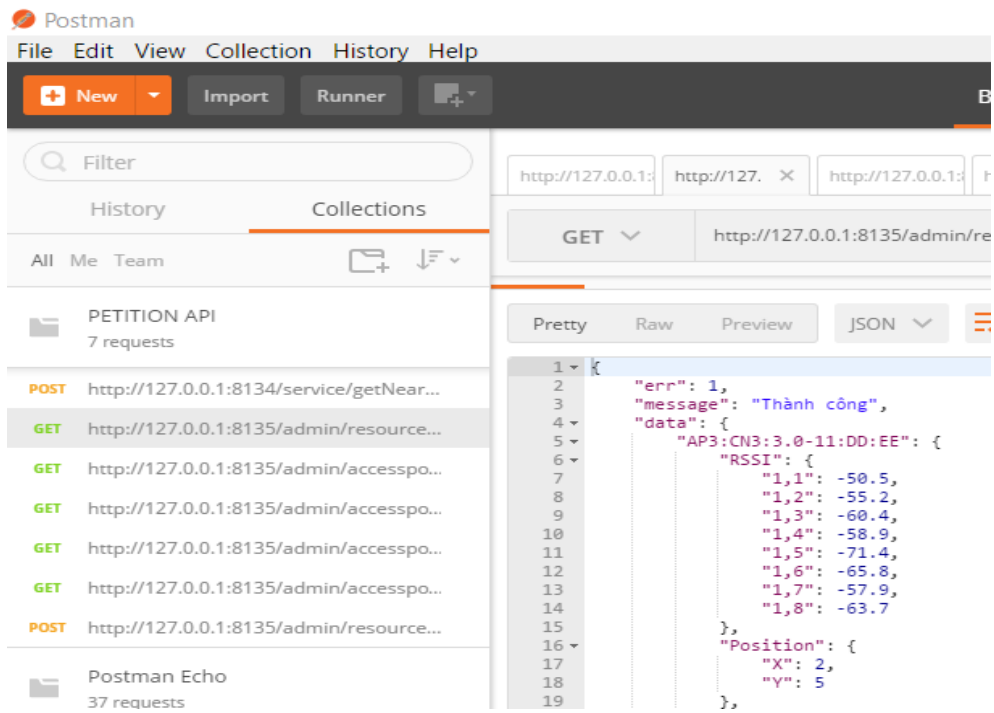
Hình 7: *Thiết kế tập tin bảng tính DATA.xlsx*

Sau đó tập tin này sẽ được lưu trữ lên trên máy chủ Back-end (*back-end Server*) để dùng cho giai đoạn Online qua công cụ hỗ trợ xử lý dữ liệu *Redis Desktop Mangerment* theo hình 8.

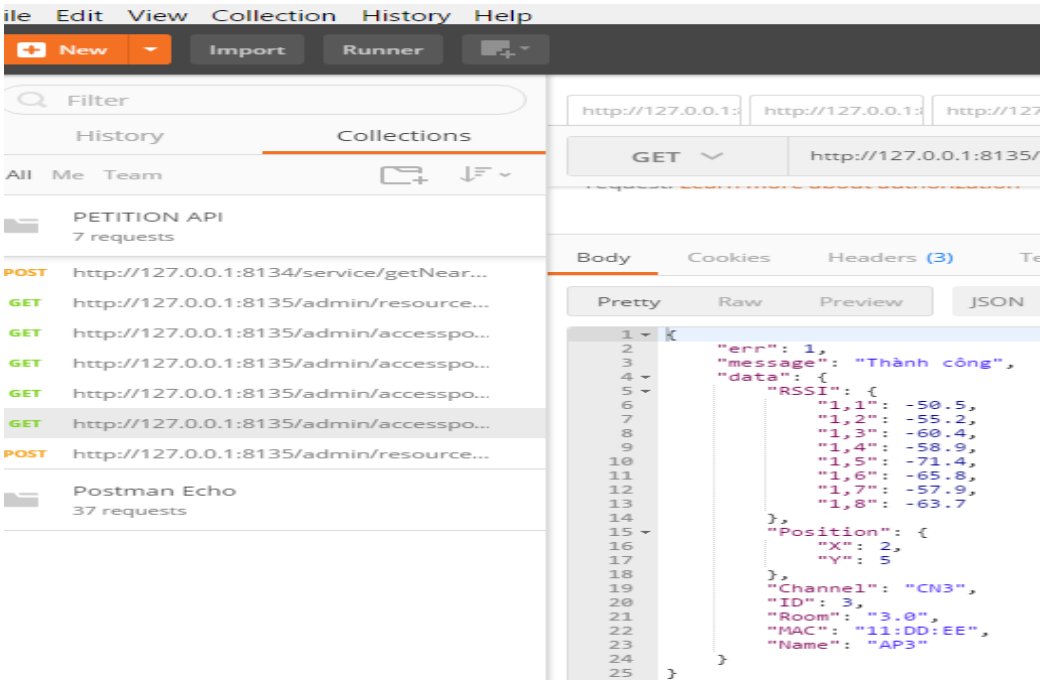


Hình 8: *Công cụ Redis Desktop Mangerment xử lý dữ liệu*

Tiếp theo trong giai đoạn trực tuyến (*online phase*), thời điểm này ứng dụng xác định vị trí và giá trị *RSSI* thực tế. Giá trị *RSSI* của tất cả các *AP* được đo và quá trình này được lấy từ máy chủ *back-end* qua công cụ nguồn mở *Postman* theo hình 9 và hình 10.



Hình 9: *Phương thức GET của Postman*



Hình 10: *Phương thức GET của Postman*

Sau đó, kết quả trả về cho người dùng là 2 giá trị được sử dụng thuật toán *k-NN* mục đích so sánh với các giá trị RSSI đã lưu trên máy chủ back-end: giá trị *RSSI* và giá trị vị trí (*X*,*Y*) tương ứng theo **thuật toán k-NN mô tả.** Cụ thể: Công cụ xuất kết quả *Postman* qua phương thức *Post* xuất kết quả người dùngcó giá trị vị trị(*X=1,Y=4*) theo hình 11.

Thuật toán k-NN mô tả:

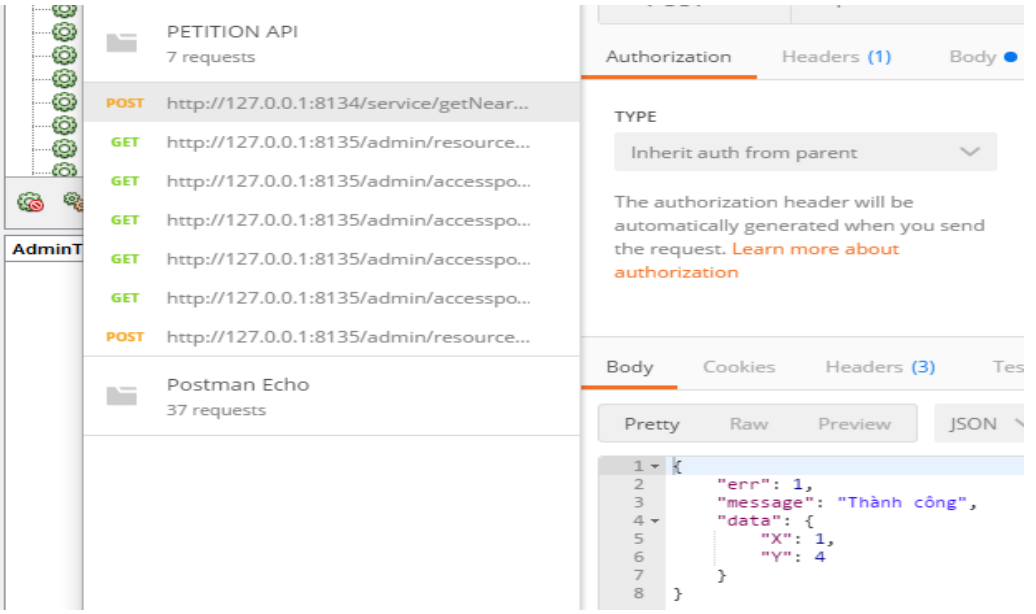
Sử dụng KNN áp dụng công thức "khoảng cách *Euclid"* để xác định điểm có chỉ số ISSI tốt nhất.

- Đầu vào (*Input*): 1 danh sách các chỉ số ISSI có sẵn (Input1). Một danh sách các chỉ số ISSI thu nhận được (Input2).
- Đầu ra (*Output*):

Output: a POINT has greatest ISSI (p)

Pseudocode:
listResult
IF length(Input1) != length(Input2)
Return false
ELSE
FOR t in Input1 DO
Tmp = SQRT(Input1[t]^2 + Input2[t]^2)
Push tmp to listReult
END FOR

SORT listResult DESC
Return first item of listResult
ENDIF



Hình 11: *Công cụ Postman xuất kết quả*

4. Kết quả

Bài nghiên cứu đã phát triển một ứng dụng định vị không dây trong nhà cho máy tính xách tay (*Laptop*). Bên cạnh đã cài đặt các *AP* dành riêng cho khu vựcđịnh vị thí nghiệm tại các vị trí cụ thể để cải thiện độ chính xác của vị trí. Định vị bằng tín hiệu Wi-Fi rất dễ thực hiện và đòi hỏi chi phí thấp hơn so với các hệ thống định vị khác. Bài nghiên cứu áp dụng một thuật toán máy học đơn giản nhất nhưng đạt hiệu quả để lọc tín hiệu lỗi và tìm vị trí của thiết bị truy cập đứng tại ví trị bất kỳ. Hệ thống này có thể được chỉnh sửa phù hợp để áp dụng cho các bãi đậu xe trong nhà.

Ngoài ra, về mặt chuyên ngành công nghệ thông tin nói riêng thì bài báo đã khái quát qua sử dụng ngôn ngữ lập trình Java để làm công cụ thiết kế phần mềm như là một ngôn ngữ lập trình hướng đối tượng (*Object Oriented Programming*) theo hình thức phát triển Back-end được phát triển mạnh mẽ trong thời gian gần đây. Bên cạnh đó cũng giới thiệu được cơ sở dữ liệu logic (*NoSQLDatabase*) mới khắc phục được nhiều nhược điểm hệ cơ sở dữ liệu trước đây qua việc xử lý dữ liệu bởi *Cache Redis* sử dụng key-value, chương trình Build bởi *Postman* để kiểm tra (*testing*) trả về kết quả trong thời gian ngắn trên giao diện Postman.

TÀI LIỆU THAM KHẢO

1. Trần Tiến Dũng (2005), *Giáo trình lý thuyết và bài tập Java*, Nhà xuất bản Lao động - Xã hội, Hà Nội

2. J. Gosling, B. Joy, G.Steele, G. Bracha, A. Buckley (2015), *The Java Language Specification*, Java SE 8 Edition, Oracle America

3. Vietut.Vn (2016), "*Java"*, https:// https://viettuts.vn/java, (ngày truy cập 01/01/2018)

4. The Apache Software Foundation (2017), "*Netbean"*, https://netbeans.org/, (ngày truy cập 01/01/2018)

5. Phạm Minh Hướng (2017), "*Back End Development"*, https://viblo.asia/p/, (ngày truy cập 01/01/2018)

6. Vinahost.Vn (2016), "*NoSQL"*, https://vinahost.vn/ac/knowledgebase/232/, (ngày truy cập 05/01/2018)

7. Bùi Thị Ngọc (2017), "*Postman"*, https:// viblo.asia/p/, (ngày 10/01/2018)

8. Douglas Crockford (2018), "*JSON"*, https://www.json.org/json-vi.html, (ngày truy cập 05/01/2018)

9. N. T. Thuong, H. T. Phong, D. D. Thuan, P. V. Hieu, D. T. Loc (2016), *Android Application for Wifi based Indoor Position: System Design and Performance Analysis*, ISBN: 978-1-5090-1723-2, IEEE publisher

TESTING AN INDOOR NAVIGATION SYSTEM USING WI-FI AND THE K-NEAREST NEIGHBOURS ALGORITHM

ABSTRACT

This study deals with indoor positioning using the Wireless Ethernet IEEE 802.11 (Wi-Fi) standard based on the k-NN machine learning algorithm that has a distinct advantage of low cost over other indoor wireless technologies. The aim of this study is to examine several aspects of location fingerprinting based indoor positioning that affect positioning accuracy achieved in the performed experiments. The impacts of human behavior on RSSI distribution are explored and analyzed. The system can develop indoor for further intelligent parking services.

Keywords*: Wireless Ethernet, RSSI, k-NN (k-Nearest Neighbours), back-end, NoSQL, Redis, Postman, JSON,Chanalyzer 4*

(Received: 6/8/2018, Revised: 15/11/2018, Accepted for publication: 11/9/2019)