ISSN 2278-3091
Volume 9 No.2, March - April 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse133922020.pdf
https://doi.org/10.30534/ijatcse/2020/133922020

Cloud-Based Indoor Positioning Service for Indoor Navigation System

David William[1], Felix Andrian Nugroho[2], Gede Putra Kusuma[3]

[1]Computer Science Department, BINUS Graduate Program - Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia, 11480, david.william@binus.ac.id
[2]Computer Science Department, BINUS Graduate Program - Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia, 11480, felix.nugroho@binus.ac.id
[3]Computer Science Department, BINUS Graduate Program - Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia, 11480, inegara@binus.edu

ABSTRACT

With technology is becoming more reachable and integrated in our daily lives, new concepts began to grow. One of those is the concept of a smart city, where architects are racing to design the most functional and aesthetically pleasing buildings. This circumstance poses a threat where bigger buildings, means a bigger chance of people getting lost inside a building. Unlike outdoors where we got GPS to help locate our position in this world. Inside a building GPS signals tend to be heavily interfered, thus making it unreliable. This paper proposes cloud-based indoor positioning service, that can be implemented for indoor navigation system. The service uses an android device to pick up the Received Signal Strength Indicator (RSSI) and send it to the server to transform it from (X, Y) points obtained by fingerprinting techniques into a global coordinate based on 2D-to-2D transformation matrix. We used 21 x 10 meters of room for the testing. The result suggest that this kind of technique is doable, and it reports good availability with the trade-off of response time and throughput at higher request counts.

Key words: BLE Positioning, Bluetooth Beacon, Fingerprinting, Indoor Positioning, Indoor Navigation.

1. INTRODUCTION

Outdoor positioning right now is very accessible for every person in the world. It uses Global Positioning System (GPS) not only to locate a device like a smartphone or a specific GPS tracker device, but also to enhance business performance in certain aspects [1]. However, the indoor positioning system cannot take advantage of the GPS Satellite due to many disadvantages, for example, a wall that interferes with the signal from the satellite into the device, places where GPS signals cannot reach like a basement. Bluetooth also becoming available around the world, with annual device shipment with Bluetooth compatibility around 5.4 billion in 2023 [2] as seen in Figure 1.

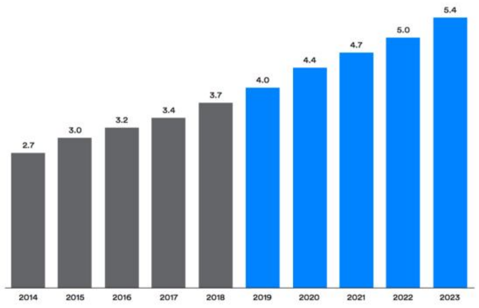


Figure 1: Bluetooth Availability Graph

With that in mind, another technology is required and Bluetooth Low Energy (BLE) is selected. It allows the integration of Bluetooth transmitters into small devices so it can store more energy for a very long period of time. With the implementation of this technology, users will get many benefits as this may solve a lot of common problems in the current society. Some of the more obvious tasks are knowing the position of a given asset, tracking people's paths, and many more. If implemented correctly, indoor positioning system can work side by side with outdoor positioning system, and acts may complement each other in its application.

Indoor positioning systems not only brings ease to end-users, but it also opens new opportunities in different sectors. Such example would be the implementation of similar technology in an offline clothing store [3], that utilizes a WiFi-based positioning system to analyze the pathways of customers in a clothing store, which does not only allow acquisition of important data and information, but also enable research regarding the result to optimize the product placement of said store in order to further increase its sales, traffic volume, and customer satisfaction. The afore mentioned example is only a peek of how the application of indoor positioning system may benefit a lot of different industrial sectors should they be implemented effectively.

However, further research on this field has found some disadvantages by using the traditional Wi-Fi based positioning compared to using Bluetooth based positioning. Among those are the fact that Bluetooth is designed for low power operations, thus making it able to run on Complementary Metal-Oxide Semiconductor (CMOS) coin cell battery which is easier to install when compared to the effort needed to install a whole Wi-Fi infrastructure. To further strengthen this argument, [4] has stated that the use of BLE may improve the positioning system's accuracy.

Due to the popularity in indoor positioning system, a lot of research have been focused on this topic. However, most of those researches focus only on making a highly accurate positioning algorithm, while none consider implementing these positioning methods.

This paper will propose a cloud-based indoor positioning service that can be implemented for indoor navigation system, along with a working prototype of an application for indoor positioning system. Cloud or commonly referred as cloud computing is a large pool of computer connected in a single network. It connects to the internet so anyone can access it virtually. It is a virtually shared servers that provide storage, applications, services, hosting, data analysis, and many more [5]. The service will be deployed on google cloud platform before its API will be accessed by an android application to obtain its global coordinates in latitude and longitude unit.

In the cloud, weighted sum is used to determine the local position of the request based on the RSSI that is sent from the phone. After that, affine transformation will be used to transform local coordinates into global coordinates (latitude and longitude) using a 2D-to-2D transformation matrix algorithm. It should be noted that the study only covers 21 X 10 meters of room and controlled environment. Other types of environments can be subject to fluctuations of data. The RSSI can be measured from periodic broadcasted signals. In Indoor Positioning System, there are three important factors to consider: the arrangement of the transmitters and receivers, the RSSI analysis and the wireless technology that will be used for the deployment [6].

2. RELATED WORKS

With many interferences in radio waves, indoor positioning needs special technique because signals like magnetic fields, reflective surface, and other radio waves like WIFI, microwave, phone cellular can cause wrong propagation of the signals. Even if there are many people inside a room can also interfere the radio waves from the beacon [7].

Using BLE for Indoor Positioning System is relatively cheap, many papers already do some research about it with many available techniques but rarely convert from local coordinates to global coordinates. In the following we present the related work considering the main technique used in each paper.

Fingerprinting is basically creating a fingerprint for each room that we used. It involves the division of a map into multiple segments or a grid. It also used massive deployment of fixed BLE beacon devices in a room with a distance around 2 meters between them. Then we take several references point using a smartphone as the device to capture the BLE beacons RSSI. This reference point contains many such values and the data could be saved and these values are called fingerprints. However, there are drawbacks using this algorithm. It requires memory to store the fingerprint and it is also sensitive to the change of positioning environment [8].

In [9], the fingerprinting technique is used with the Euclidean distance correction algorithm and it can achieve a positioning error to only 1.58 meters in 44 meters long and 22 meters wide room. 34 Bluetooth beacons are used and placed 5 meters away from the neighboring one and adjusted to 4dBm and the refreshing rate is 10/3Hz.

In [10], another technique is used. There are two-phase procedure, offline training and online locating. In offline training, it uses piecewise fitting based on the lognormal distribution model to train the propagation model of RSSI for every BLE reference nodes with Gaussian filter to pre-process the receiving signals in different sampling points. In online locating, weighted sliding window is used to reduce fluctuations of the real-time signals. And calculating the distance with weighted filter based on triangle trilateral relations theorem. The experiments show that the probability of locating an error of fewer than 1.5 meters is higher than 80%.

Another paper [11] also use BLE for indoor positioning system with MQTT and IoT device. It uses RSSI measurements and trilateration of fixed points BLE beacons. With that technique, Least Square (LSQ) method is used to calculate the mobile device location. The result concluded the error is less than 1.5 meters which is good.

Another experiment with Kalman-Based Fusion is used in [12] where it can outperform both trilateration and dead reckoning in terms of accuracy. The fusion is between the trilateration method to measure RSSI and Kalman based filter. Once the RSSI is accepted, the Kalman filter is applied. The experiments show that the error is less than 1 meter with Kalman-Based fusion. However, using an android app to detect BLE device and put the algorithm in android consumes much energy.

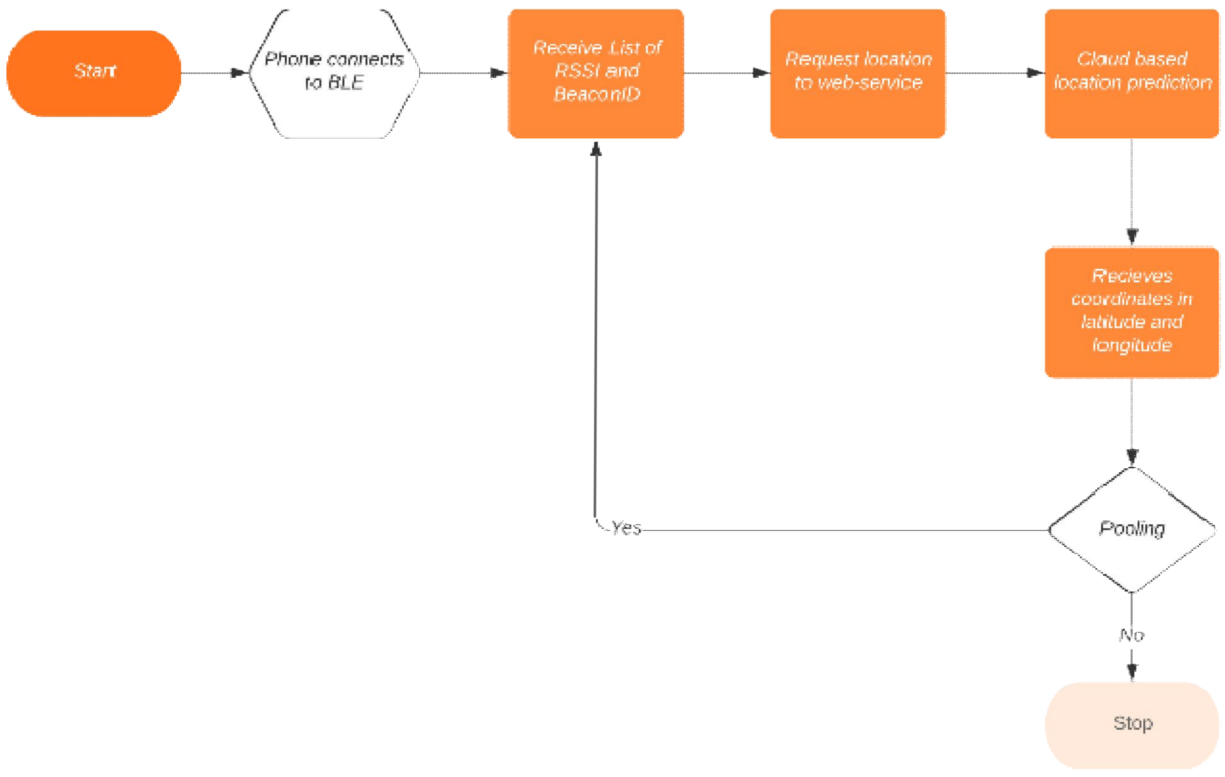

Start
Phone connects to BLE
Receive List of RSSI and BeaconID
Request location to web-service
Cloud based location prediction
Recieves coordinates in latitude and longitude
Pooling
Yes
No
Stop

Figure 2: Application's Flowchart

The indoor navigation system consists of three main components, cloud, smartphone application (Android), and Bluetooth beacons, as follows:

A. Cloud Server

Application will be deployed using Google Cloud Platform. All data will be saved in a database using SQL (Structured Query Language) which is also deployed on the said server using CloudSQL provided by the cloud server in use.

B. Smartphone Application (Android)

A smartphone with an Android operating system is used to collect the Received Signal Strength Indicator (RSSI). This smartphone must be equipped with Bluetooth 4.0 adapter so it can scan the beacon signal and the minimum API level is 18 which is Android 4.3 (Jelly Bean). It uses Android devices because it is the most widely used operating system in the world. Xiaomi Redmi Note 7 with Android 10 and Samsung Galaxy S7 Edge with Android Oreo is used during this experiment to get the RSSI from BLE. Kotlin is used to build the apps and Android Studio as a framework.

C. Bluetooth Beacon

Our system used beacon manufactured by Radioland with type nRF5182 with iBeacon protocol. It uses 32-bit ARM® Cortex™-M0 CPU with 256/128 KB flash and 32/16 KB RAM.

3. METHODOLOGY

There are two main steps here, first Android smartphone is used with installed custom app to listen to every Bluetooth device with filter so only the registered MAC Address can be scanned with the app. After a while, list of RSSI and Beacon ID that are collected by the device is then sent to the cloud for second steps that is further processing with weighted sum to locate with nearest beacon with dataset and affine matrix to transform local coordinate into global coordinate.

The flow diagram in Figure 2 shows the overall flow of the application made in this paper. It describes the architecture in which this application will use not only a device and a collection of Bluetooth low energy beacons, but also integrates a web-service to do the calculation. The detail of operations done in the web-service can be observed on the swimlane diagram on Figure 3.

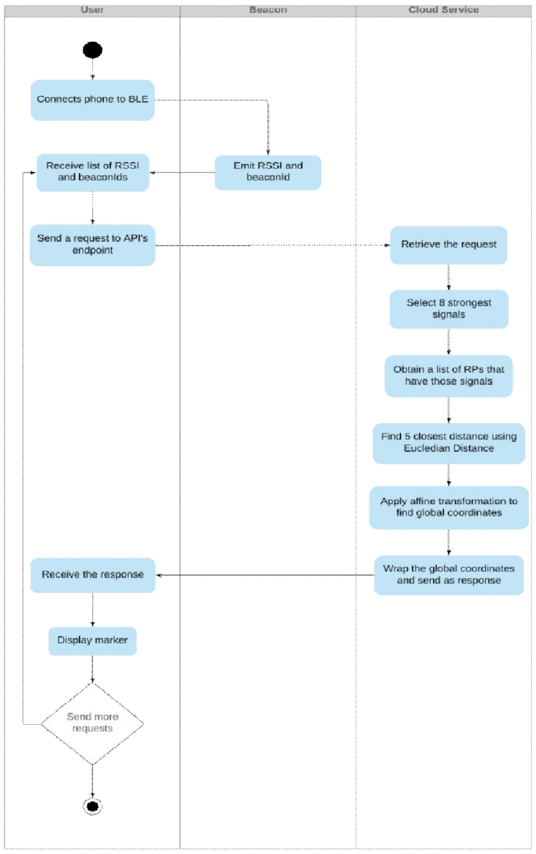

User
Beacon
Cloud Service
Connects phone to BLE
Receive list of RSSI and beaconIds
Emit RSSI and beaconId
Send a request to API's endpoint
Retrieve the request
Select 8 strongest signals
Obtain a list of RPs that have those signals
Find 5 closest distance using Euclidean Distance
Apply affine transformation to find global coordinates
Wrap the global coordinates and send as response
Receive the response
Display marker
Send more requests

Figure 3: Application's Swimlane Diagram

Note that in Figure 3, the third step done by the cloud-based service will need reference point data which were obtained by fingerprinting method. To do this, 23 beacons were installed in the testing environment, and 155 points in the area were selected. Then using the coordinate from each of those points and the list of signal strength each beacon transmit to that particular point was a reference point data made. Figure 4 depicts the map of testing environment use in this study, along with examples of point placements.

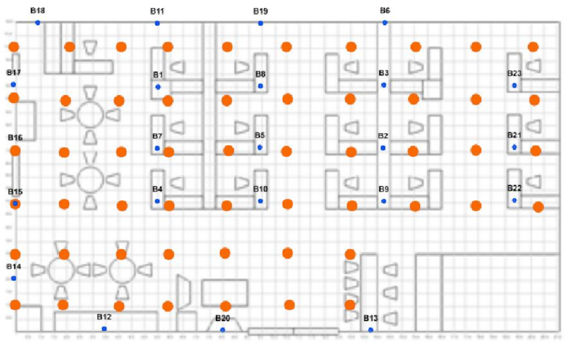


Figure 4: Map of the Used Testing Location

Blue dots on Figure 4 symbolize each beacon placed in the testing environment, while the orange dots symbolize each reference point. Each point has their coordinates in x-y axis, where $0 \leq x \leq 1750$, and $0 \leq y \leq 950$.

Each beacon represented by the blue dots in Figure 4 will emit their own signal. The strength of these will then be captured by an android phone as a list of RSSI, where the device will then make a web-request to a cloud server, passing the list of RSSI and their respective beacon Id in the request.

The cloud-based service will then receive the request and find 8 closest beacons to the query's current location. This process is done by retrieving the list of RSSI passed in the web-request, and sorting the list based on strongest signal strength, since this should indicate how far is a beacon to the user's current location. The 8 selected beacons will then be used in the process of finding all reference points that may correspond with the query, which will help find the distance of the user's location to each of these reference points.

Distance from user's location to each reference point is calculated by using Euclidean distance in formula (1)

$$D(Q, RP_i) = \sqrt{\sum_{b=1}^{b<=n} (pRSSI_b - qRSSI_b)} \qquad (1)$$

Where RP_i used to symbolized a reference point number i, Q indicates the query's current location, $qRSSI_b$ indicates the signal strength obtained by user from a beacon with index b, $pRSSI_b$ indicates the strength of signal received by reference point i from the beacon with index b; where n is a number ranging from 1 to the number of signals received by a reference point.

The result of the above calculation will be a list of distances, that depicts how far the user is to each reference point in the map. The service will then find 5 closest reference points to the user and find the user's coordinate in the current map by using *weighted sum* to find the x-axis and y-axis respectively.

$$W = \frac{1}{D} \qquad (2)$$

$$(X, Y) = \left(\frac{\sum_{i=1}^{5} (Wi * Xi)}{\sum_{i=1}^{5} W_i}, \frac{\sum_{i=1}^{5} (Wi * Yi)}{\sum_{i=1}^{5} W_i} \right) \qquad (3)$$

Table 1: Data Example

	X	Y	Latitude	Longitude
Reference Point 1	0	0	-6.201784	106.781861
Reference Point 2	1000	0	-6.201905	106.781773
Reference Point 3	0	1000	-6.201795	106.781778

In order to display a marker on the map however, it is required that we provide a point in latitude and longitude, and to do this, we need to do a 2D-to-of transformation by translating, rotating, and scaling our local coordinates to that of the global coordinates. This is done by applying affine transformation. Affine transformation is a linear mapping method that preserves points, straight lines, and in the case of this study; a plane, by multiplying the source matrix with a linear transformation matrix (scale operations and rotations), before applying vector addition to depicts the translation process. Affine transformation usually can be represented by

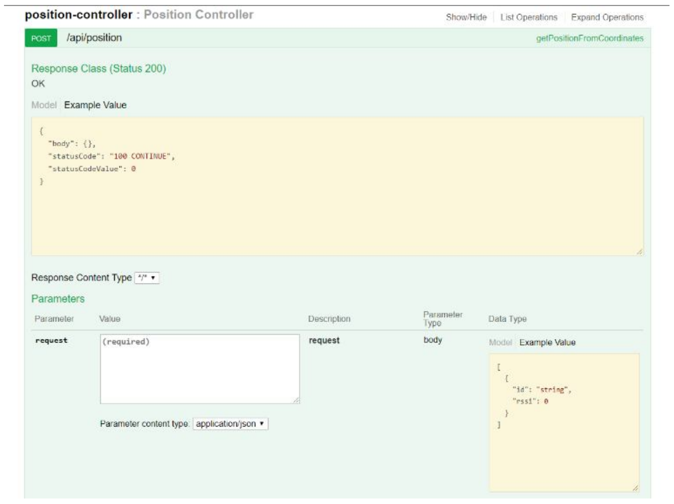
$$T = A \cdot S + B \tag{4}$$

To find both the linear transformation matrix and translation matrix stated above, we need to find at least three points in our local map, as seen in Table 1, that correspond with their counterpart on the global map. The collection of points in our local map will form the matrix and will represent the source plane, while the matrix will contain numbers that are obtained from the global map in latitude and longitude measurement to describe our target plane.

The value of the user's position in latitude and longitude will then be passed on to the user's device, where an application will use those values to display a marker on a map. The device will then make more requests to the service in a similar manner as stated before to obtain its most recent position. These requests will be made using the pooling strategy, where there will be a fixed amount of time, in which the device will make an API call each time the time threshold is met.

4. RESULTS

In the implementation of the indoor-positioning system made in this paper, one of the most crucial steps is to make the API for data processing. This API endpoint will calculate the request that consists of signal strengths and return the global coordinate of its request. The request body, response body, and the endpoint of this API will be depicted in Figure5.



position-controller : Position Controller

Show/Hide | List Operations | Expand Operations

POST /api/position getPositionFromCoordinates

Response Class (Status 200)

OK

Model Example Value

```
{
  "body": {},
  "statusCode": "100 CONTINUE",
  "statusCodeValue": 0
}
```

Response Content Type */*

Parameters

Parameter	Value	Description	Parameter Type	Data Type
request	(required)	request	body	Model Example Value

```
[
  {
    "id": "string",
    "rssi": 0
  }
]
```

Parameter content type: application/json

Figure 5: API Endpoint

This service will be deployed on the Google Cloud Platform (GCP) ecosystem, using Google App Engine (GAE) as its platform as seen in Figure 6. The usage of GAE is because it offers automatic scaling for web applications, in which it will allocate more resources for the web application according to demands.

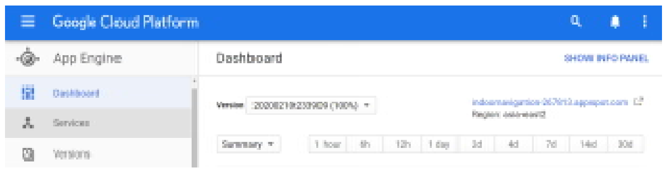
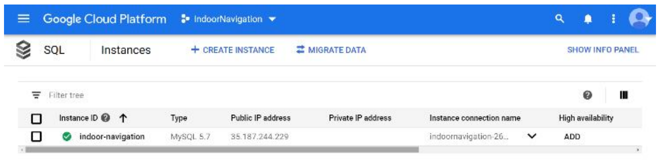


Figure 6: Deployment on Google Cloud Platform's App Engine

Processing the requests made to this API will also require all beacons and reference points data. For accessibility and scalability purposes, the database made by using MySQL as its DBMS (Database Management System) will also be hosted on the GCP server using yet another google-developed feature, which is CloudSQL, as seen in Figure 7.



Instance ID	Type	Public IP address	Private IP address	Instance connection name	High availability
indoor-navigation	MySQL 5.7	35.187.244.229		indoornavigation-26...	ADD

Figure 7: Online Hosted Database Using CloudSQL

In order to assess the application's performance, a set of tests is done to the API's endpoint. The tests will be done using Apache Jmeter, which is a pure Java desktop application, designed for load testing and to measure performance-related aspect of an endpoint [13], while using load testing as the testing methodology.

Some aspects that will be taken into consideration for these tests are the number of concurrent users, the API's response time, availability, and throughput while it is being flooded by the number of requests made from those users.

Table 2: Performance Testing

Users	Response Time (s)	Availability (%)	Throughput (/min)
25	2.7	100	8.5
50	4.1	100	6.8
100	7.7	100	6.6

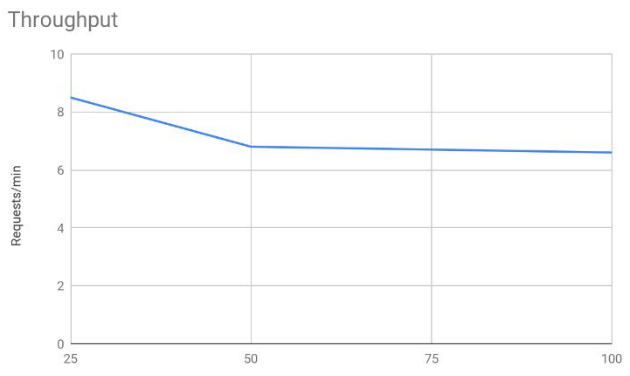


Figure 9: Throughput Graph

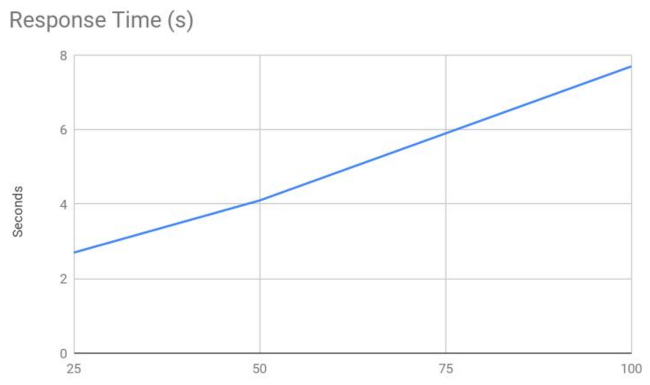


Figure 8: Response Time Graph

Table 2 shows the result of load testing with 3 different scenarios:

- 25 concurrent users
- 50 concurrent users
- 100 concurrent users

Figure 8 displays the graph of response time in seconds with each 25, 50, and 100 users, while Figure 9 depicts the throughput of each scenario using request/min as its unit of measurement.

Even though the service returns a mediocre result from the tests, as expected the response time went up accordingly in comparison to its concurrent user. The throughput also experiences similar changes, this may be impacted by the stale amount of availability in 100%, as the number means there is not a single request that obtain failure as its response, thus reduces both the throughput and the response time of said request. This means that the web-based service is having difficulties keeping up with the flood of request coming, while serving other requests.

It can also be observed from the test results that the service did worse when there is an increase of concurrent requests. It is possible that this can be caused by the limitation in computing capabilities of the computing engine. One way to solve this is to apply a load balancer. There are several algorithms to approach load balancing that allows scaling on demand for cloud computing and serve the purpose to help the service split the traffic to some node in order to have better performance, despite the heavy traffic[14].

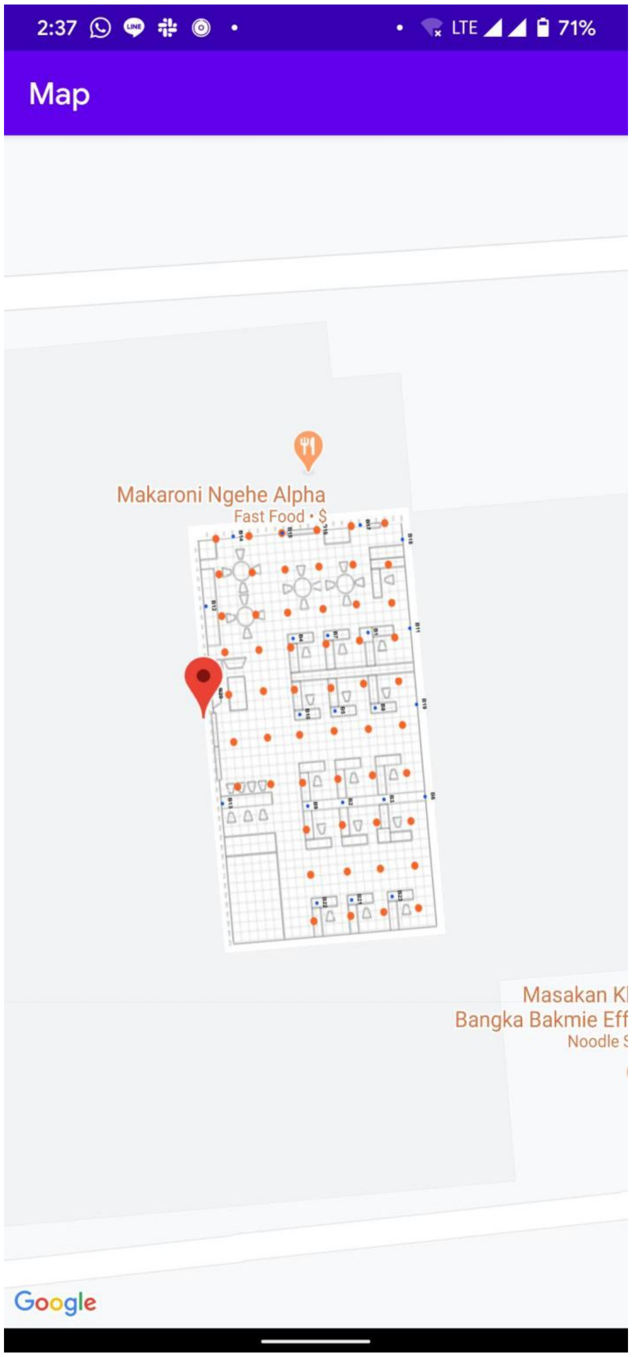


Figure 10: Application's Interface

From the android side, as observed in Figure 10, a marker is then displayed to display user current position based on the latitude and longitude. The positioning system used in this paper itself have accuracy ranging around less than 1m to 3m. There are a lot of factors that may affect this result, one of them are because the proposed method uses Weighted Sum algorithm which reportedly have low accuracy. Another reason may be because of the placement of beacons in the testing environment. The beacons used in this research is placed fairly close to the ground in around 1m from ground level of the testing environment, this may affect the accuracy as lower placement means higher chance of the signal being interfered by objects and/or crowds.

5. CONCLUSION

From the application of fingerprinting and cloud-based calculation of global coordinates, it can be concluded that affine transformation is proven to be usable in obtaining global coordinates of latitude and longitude, given a set of local coordinates in X and Y axes. Using cloud-based calculation also preserve the battery on the smartphone.

The result of this study shows that the implementation of cloud-based indoor positioning system is indeed possible, even though it reports an average performance both in response time and in throughput when there is a significant amount of request to the API. Another aspect that is open for development is that the prototype made in this paper has yet to implement indoor navigation system. Therefore, for future work in cloud service section, it is suggested that performance improvement is done by applying a load-balancing strategy in order to make the service capable of handling more concurrent requests, without sacrificing both its response time and its throughput. It is not only improving the performance of the service, but also implement this service directly to a working indoor navigation system.

REFERENCES

1. H. R. M. Sapry, A. F. Muzaffar, A. R. Ahmad, and S. Baskaran. **The Implementation of Global Position System (GPS) among the Cement Transporters and its Impact to Business Performance**, *International Journal of Advanced Trends in Computer Science and Engineering*, Vol. 9, pp. 12-16, Feb 2020. https://doi.org/10.30534/ijatcse/2020/0391.12020
2. S. Zeadally, F. Siddiqui, and Z. Baig. **25 Years of Bluetooth Technology**, *MDPI Future Internet*, Vol. 1, pp. 194, Sept 2019. https://doi.org/10.3390/fi11090194
3. Z. Jianyong, H. Luo, and C. Zili. **RSSI Based Bluetooth Low Energy Indoor Positioning**, *International Conference on Indoor Positioning and Indoor Navigation*, pp. 526-533, Oct 2014. https://doi.org/10.1109/IPIN.2014.7275525
4. A. A. Kalbandhe and S. Patil. **Indoor Positioning System using Bluetooth Low Energy**, *International Conference on Computing, Analytics, and Security Trends*, Dec 2016.
5. G. Sunil, S. Aluvala, N. Yamsani, K. R. Chythanya, and S. Yalabaka. **Security Enhancement of Genome Sequence Data in Health Care Cloud**, *International Journal of Advanced Trends in Computer Science and Engineering*, Vol. 8, pp. 328-332, April 2019.
6. K. Mekki, E. Bajic, and F. Meyer. **Indoor Positioning System for IoT Device based on BLE Technology and MQTT Protocol**,*5th IEEE World Forum on Internet of Things*, April 2019.

7. Z. Yang, C. Wu, and Y. Liu. **Locating in Fingerprint Space: Wireless Indoor Localization with Little Human Intervention**, *Mobicom '12: Proceedings of the 18th annual international conference on Mobile computing and networking.* pp.206-280, Aug 2012. https://doi.org/10.1145/2348543.2348578
8. H. Li and H. Ma. **A Low Complexity Low Power Indoor Positioning System Based on Wireless Received Signal Strength**, *IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom),* Nov 2018.
9. C. Paterna, C. Ague, P. Aspas, and P. Bullones. **A Bluetooth Low Energy Indoor Positioning System with Channel Diversity, Weighted Trilateration and Kalman Filtering.***SENSORS,* vol. 12, Dec 2017.
10. Z. Jianyong, C. Zili, L. Haiyong, and L. Zhaohui. **RSSI Based Bluetooth Low Energy Indoor Positioning**, *2014 International Conference on Indoor Positioning and Indoor Navigation, Oct 2014.* https://doi.org/10.1109/IPIN.2014.7275525
11. Y. Wang, Q. Yang, G. Zhang, and P. Zhang. **Indoor Positioning System Using Euclidean Distance Correction Algorithm with Bluetooth Low Energy Beacon**, *2016 International Conference on Internet of Things and Applications (IOTA).*pp. 243-247, Jan 2016. https://doi.org/10.1109/IOTA.2016.7562730
12. J. I. Glitza, P. Zhang, M. Abdelaal, and O. Theel. **An Improved BLE Indoor Localization with Kalman-Based Fusion: An Experimental Study**, *SENSORS, v*ol. 5, April 2017.
13. S. S. Patil, and S. D. Joshi. **Identification of Performance Improving Factors for Web Application by Performance Testing**, *International Journal of Emerging Technology and Advanced Engineering,* vol. 2, Aug 2012.
14. S. Karimunnisa, and V. S. Kompalli.**Cloud Computing: Review on Recent Research Progress and Issues**, *International Journal of Advanced Trends in Computer Science and Engineering,* vol. 8, April 2019. https://doi.org/10.30534/ijatcse/2019/18822019