
Politecnico di Torino

Master’s Degree in ICT for Smart Societies (ICT4SS)
A.a. 03/2022

# Gateways and wearable tools for monitoring patient movements in a hospital environment

Relatori:

Prof. Monica VISINTIN
Prof. Guido PAGANA

Candidati:

Sahand HAMZEHEI

**Abstract**

In increasingly technological and connected health, medicine adapts to the changes, benefiting from them and applying the most innovative technologies to make the healthcare system increasingly efficient and effective. This research aims to track patient movement throughout the hospital departments using widely accessible technology integrated with cutting-edge algorithms. With the increasing number of patients with varying levels of urgency in hospitals, it is vital to incorporate green technology that assists hospitals in controlling patient flow. The research aims to create a real-time system that lowers the cost of utilizing an Indoor Positioning System (IPS) by giving multiple platforms that allow consumers to choose based on their preferences. In addition to purely managerial ones, the implications could also affect fragile patients requiring continuous monitoring (patients with Alzheimer, dementia, children, etc.). In this study, we did not go into these aspects, but it is a work that concerns the management of patients and the optimization of hospital management purely. First, we implement an IPS for patients, and the healthcare professional can view the patients' real-time location by using a powerful dashboard that includes all of the necessary parts. Meanwhile, tag ID is a beacon device associated with each patient by hospitals staff. In this case, by using Telegram Bot, hospitals staff can send the tag's QR code photo instead of entering data directly. The system's availability is a significant feature; the complete system is still running in the Telegram Bot if the dashboard server is down. Another point is finding the hospital's departments with the system completely independent of GPS. In this case, anytime a user selects a building as the desired destination, all of the information is recorded and stored; after a while, the statistical report is made available to identify the most frequently visited departments. Data analysis is developed that may result in a beneficial influence on the smart hospital. This study aims to have a strong impact on hospital management. It will be possible to optimize the personnel orientation according to the department's population. Still, it will also be possible to avoid the overcrowding of hospital areas, a fundamental point in recent years due to the health emergency.

# Acknowledgements

This master thesis report represents the end of my studies at Politecnico di Torino for earning a Master of Science (MSc) degree in ICT for smart societies. The experience for me while working on my thesis was both demanding and, more than that was, intellectually interesting. Completing this job would not have been possible without various people's immense and persistent assistance and direction. First, I would like to express my sincere appreciation to professors Visintin and Pagana, my supervisors at Politecnico di Torino, for their guidance in formulating my research aims to measure strategy. I would also like to thank Dr. Valeri Figini, my daily supervisor, for giving time out of her busy schedule to offer feedback on my work and assisting me to discover the cure for the numerous issues that I had throughout the building of the experimental setup.

*Sahand Hamzehei*

# Table of Contents

# List of Tables

# List of Figures

# Acronyms

**IoT**
Internet of Things

**IPS**
Indoor Positioning System

**SNR**
signal-to-noise ratios

**RFID**
Radio-frequency identification

**WLAN**
Wireless local area network

**BLE**
Bluetooth low energy

**UWB**
Ultra-wideband

**AoA**
angle of arrival

**ToF**
Time of Flight

**UX**
User Experience

**UI**

User Interface

**MQTT**

Message Queuing Telemetry Transport

**RSSI**

signal strength indicator

**TOA**

time of arrival

**MAC**

Media Access Control

**AP**

Access Point

**URL**

Uniform Resource Locator

**QoS**

Quality of Services

**JSON**

JavaScript Object Notation

**REST**

Representational State Transfer

**UF**

User Friendly

**TDA**

Time Difference of Arrival

**QR code**

Quick Response code

**PIL**

Python Imaging Library

# Chapter 1

# Introduction

Internet of Things (IoT) refers to the connected collection of anybody, anything, at any time, from any location, using any service, on any network [**1**]. IoT is characterized as a powerful branch of technology that links objects intending to smart them in today's world. It is also popular since it is acknowledged as one of the multidisciplinary fields that may collaborate with other technologies such as Energy Engagement, Smart Retail, Connected Cars, etc., to solve problems. The IoT brings together the Internet, networking systems, and mobile communication to allow a wide range of computer and cognitive operations to be carried out remotely. Because of the increasing number of Internet-connected items, the IoT might be considered one of the world's most significant technological revolutions. IoT is also a by-product of connecting physical things and people to the Internet, allowing them to communicate across a network. According to Fig. 1.1[**2**], it is expected that the number of connected devices will grow to around 75 billion by 2025, increasing IoT-connected devices. In 2015, the number was about 15 billion. While the number of connected devices is approximately 43 billion in 2022, this number is expected to expand by 9 billion in the following year.

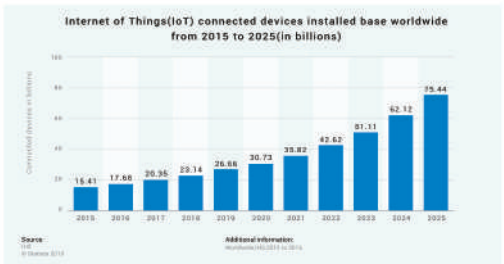


**Figure 1.1:** IoT connected devices installed base worldwide from 2015 to 2025

The blooming of Industry 4.0 has paved the way for the widespread use of the IoT in various fields, ranging from everyday living to research, technology, and industry [**3**],[**4**],[**5**]. Future applications for the IoT are predicted to span almost all industries and businesses, allowing for advancements in manufacturing techniques, information sharing systems, and business processes, among other things. An IoT system comprises several functional blocks that enable the system to perform various functions [**6**]. There are several areas in which the IoT has made significant progress throughout the years, such as,

- Wearables
- Smart Home Applications
- Health Care
- Smart Cities
- Agriculture
- Industrial Automation

One of the most effective uses of the Internet of Things is in the field of healthcare and wearable technology. The combination of these technologies forms a powerful tool that can be used to monitor the health status of individuals throughout the day. Depending on the model, they may be capable of sending an alert when the heart rate exceeds the typical threshold. Numerous sectors may benefit from IoT-based systems; one such industry is health care, where researchers might make hospitals more intelligent by adding cutting-edge techniques. That is why the IoT has much potential in the medical and health care areas [**7**]. E-health is a comparatively recent healthcare approach that relies on electronic procedures and communication; this technology is frequently considered in [**8-15**]. One of the examples is smart wearable, which allows nurses and caregivers to check their patients' health status whether they are in a hospital room or outdoors. Smart wearable devices are already owned by around 20% of US inhabitants, and the global market is predicted to increase at a compound annual growth rate of 25% over the next four years, reaching US$70 billion by 2025 [**16-17**].

This thesis is split into two different sections with the benefits of providing two independent systems to help both patients and nurses while it considers powerful devices to keep the final price within reasonable limits. Fig. 1.2 depicts the overall project phases.

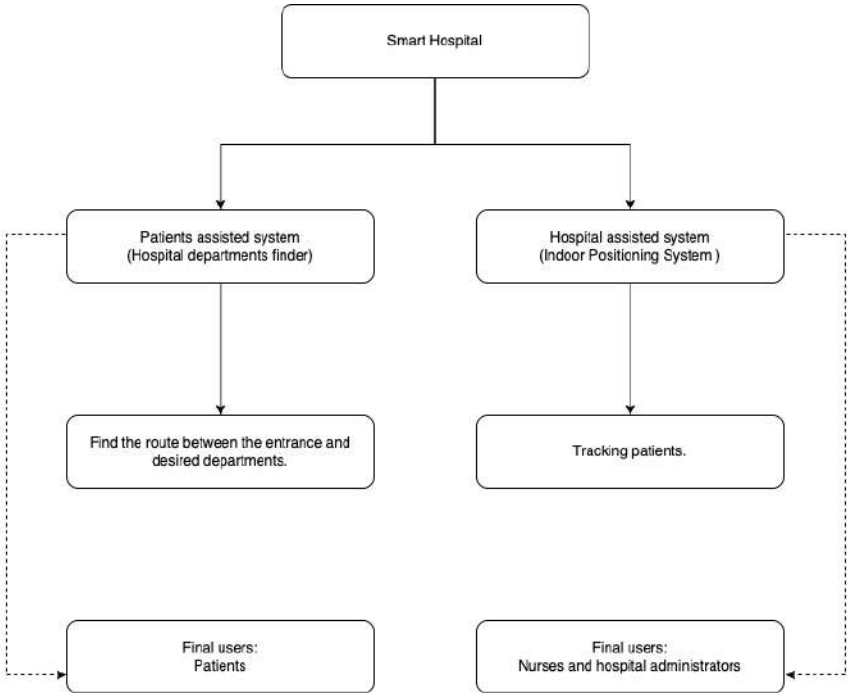


**Figure 1.2:** The two phases of the project

Due to various difficulties such as multipath, the accuracy of GPS diminishes in certain instances, resulting in errors in the case of positioning even when the patients are not physically within the building. A system that could rely on local positioning independently, without relying on the GPS, is required to deal with this challenge effectively and efficiently. The first phase is concerned with the patients trying to access specific departments when the hospital's departments are separated from one another. 'Hospital departments finder' is the name given to this system. Patients outside the hospitals may quickly reach their favored departments using the Hospital Departments Finder. Furthermore, by collecting data on which departments are most often visited, a report that emphasizes the busiest departments to the hospital's manager can be prepared. In addition to offering essential advantages to the hospital, this system is fully independent of any devices and does not incur any extra costs for the hospital to pay. A detailed representation of the Hospital department's finder system is shown in Fig. 1.3.

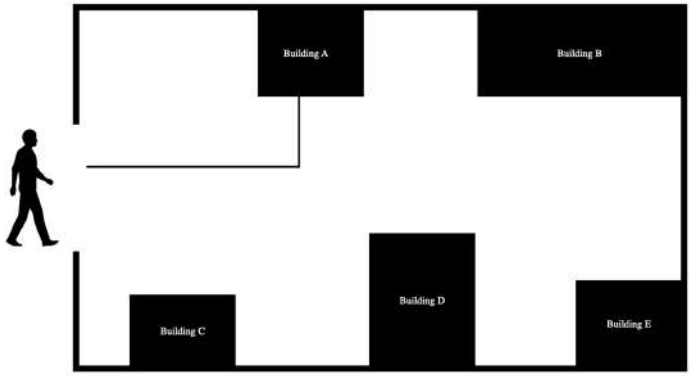


**Figure 1.3:** A workflow diagram displays the Hospital department's finder

In addition, an automated system is required to display available rooms and monitor patients periodically to manage them inside hospitals. The next phase is that the technology, known as the Indoor Positioning System (IPS), assists nurses and doctors in tracking patients' movement throughout the hospital, which takes a new look at smart wearables. This method assists them in identifying available rooms and managing the hospital population throughout the year. In this scenario, they will be able to determine the period during which the number of patients increased or decreased, and vice versa. The most significant distinction between the IPS developed in this thesis and others is its level of flexibility. The hospital's management may adjust the system quality level in the case of the interface and choose between the Telegram Bot and the Dashboard based on a budget set by the hospital. On the other hand, according to a survey of 50 people who tried our dashboard demo, 80% of them were pleased with its functionality and User Interface (UI). Fig. 1.4 illustrates the several levels of proposed IPS.

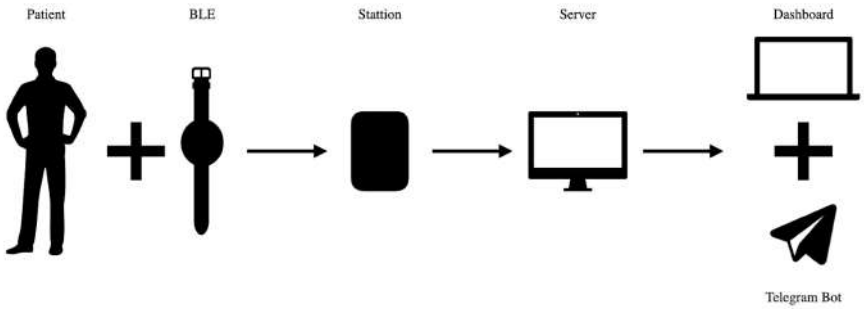


**Figure 1.4:** A workflow diagram displays the general procedures of project

Incorporating these two strategies results in patients being routed to the appropriate departments while also enabling hospital employees to keep track of the patient's location during their operation, raising the overall quality of services in the hospital more.

# Chapter 2

# Concepts definition

The atomic clock, which enables the system to measure time accurately, serves as the backbone of satellite navigation [**18**]. Some factors, such as buildings, bridges, trees, and multipath problems, might cause GPS location accuracy to be reduced, according to [**19**]. When there is an obstacle in the route of the radio signal, GPS is subject to a number of restrictions. The difficulty becomes more specific when all the obstacles are added together and form a building. Each building comprises several walls, floors, windows, and insulation sections. The crux of the matter is that signals within a building are faced with two insurmountable challenges: low signal-to-noise ratios (SNR) and multipath transmission. GPS signals are challenging to acquire and track due to low signal strength and multipath, resulting in the less precise location, according to [**20**]. GPS is a location-based system that works by transmitting and receiving signals. This system requires satellites and a receiver capable of receiving the signal broadcast by the satellites in order to compute the distance between users and display their position. However, when signal strength is diminished, the GPS performance is not precise enough in the building. That is why another technology should be developed for indoor localization.

In this case, an IPS is described as a system that is designed to handle difficulties associated with the location of users when GPS is unable to operate normally. IPS aims to combine a variety of devices to develop indoor GPS functionality that is easy to use for the final users. GPS, on the other hand, has implementation and improvement difficulties; IPS also has challenges, which are most of the time related to the different usage and structure of the building, appropriate software to have interaction with final users, keeping the system available in different situations, and the last one needing money to implement, whereas GPS is a free service. According to [**21**], there are some communication-based technologies for IPS such as RFID, WiFi, Visible light communication (VLC), and Bluetooth. All of these technologies can be used under specific limitations.

There are several approaches for developing IPS, each with its own set of functions in various environments. For instance, when indoor localization occurs in industries, the height of the interior structure becomes an issue. Additionally, the system should continuously detect students' locations in certain cases, such as schools. One criterion contributing to selecting an appropriate IPS system is the positioning precision, which may range from a few centimeters to tens of centimeters. Using different techniques is considered to develop in the IPS technologies to achieve desired requested positioning accuracy based on the specification of a building as detailed in Table 1.

## 2.1 The techniques for position determination

There is a question when the positioning is considered in any project, whether indoor or in GPS. How do satellites determine the location of the users by sending signals? This section evaluates different techniques to determine how position can be found using signals. Trilateration, triangulation, and fingerprinting are names that describe several techniques of determining position using radio signals. Different forms of placement are feasible and frequent inside a system.

### 2.1.1 Trilateration

Fig. 2.1 represents the steps taken to find the position with the trilateration technique. The initial step makes the assumption that three satellites are accessible in the two-dimensional situation, and their positions in orbit are known. In this situation, all satellites send a signal to the GPS receiver to pick it up at a given time and distance. However, the angle is unknown in this example; the distance is known. As a result, the distance between two points forms a circle, yet the GPS location may be determined anywhere within the circle's radius.
The second step is where the second satellite is considered; the behavior is the same as the first satellite, where a circle is formed but slightly different. The problem is that by considering the intersection of the two circles is possible to find the position but not precisely.
And finally, in the third step, one remaining satellite forms a circle to find the actual location. Thus, trilateration may determine an exact position by using three satellites. As seen in Fig. 2.1, each satellite is situated in the circle's center, and by adjusting the receiver, the satellite's radius (here, distance) is also adjusted.

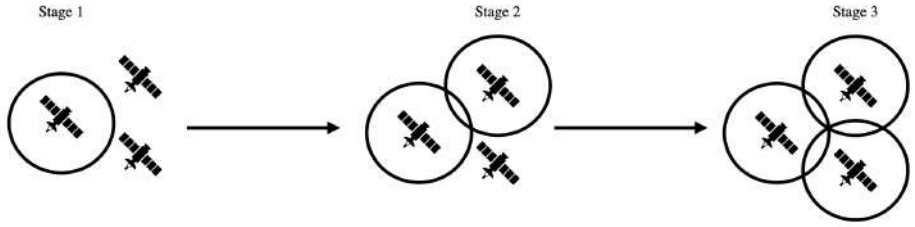


**Figure 2.1:** Trilateration

Equation. 2.1 gives the Trilateration formulation in two dimensions.

$$\begin{aligned}(x-x_1)^2+(y-y_1)^2&=r_1^2\\(x-x_2)^2+(y-y_2)^2&=r_2^2\\(x-x_3)^2+(y-y_3)^2&=r_3^2\end{aligned} \tag{2.1}$$

where $(x_i,\ y_i)$ are the coordinates of the satellite, while $(x, y)$ are the unknown positions of the users. $r$ represents the radius of each circle. It is noticeable to mention that each circle is defined by the coordinates of its center e.g. $(x_i, y_i)$, and its radius $r_i$. It can be expanded out the squares in each one as stated in Equation. 2.2

$$\begin{aligned}x^2-2x_1x+x_1^2+y^2-2y_1y+y_1^2&=r_1^2\\x^2-2x_2x+x_2^2+y^2-2y_2y+y_2^2&=r_2^2\\x^2-2x_3x+x_3^2+y^2-2y_3y+y_3^2&=r_3^2\end{aligned} \tag{2.2}$$

Equation. 2.3 and 2.4 can be obtained by subtracting the second equation from the first and the third equation from the second.

$$(-2x_1+2x_2)x+(-2y_1+2y_2)y=r_1^2-r_2^2-x_1^2-x_2^2-y_1^2+y_2^2 \tag{2.3}$$

$$(-2x_2+2x_3)x+(-2y_2+2y_3)y=r_2^2-r_3^2-x_2^2-x_3^2-y_2^2+y_3^2 \tag{2.4}$$

Equation. 2.7 represents a system when there are two unknown parameters.

$$\begin{aligned}Ax+By&=C\\Dx+Ey&=F\end{aligned} \tag{2.5}$$

where,

- $A = -2x_1 + 2x_2$
- $B = -2y_1 + 2y_2$
- $D = -2x_2 + 2x_3$
- $E = -2y_2 + 2y_3$
- $C = r_1^2 - r_2^2 - x_1^2 - x_2^2 - y_1^2 + y_2^2$
- $F = r_2^2 - r_3^2 - x_2^2 - x_3^2 - y_2^2 + y_3^2$

here $(x, y)$ are unknown coordinates in two dimensions. Equation. 2.6 demonstrates the final solution to calculate the positions.

$$\begin{aligned} x &= \frac{CE - FB}{EA - BD} \\ y &= \frac{CD - AF}{BD - AE} \end{aligned} \tag{2.6}$$

Trilateration is used for various applications, but in positioning, it is utilized to expand topographic mapping control from small local tracts to regional regions and GPS [**57**].

### 2.1.2 Triangulation

The triangulation technique is used when the distance is unknown, and by considering three points and establishing a baseline length, it is possible to measure the angles. When the length and angles are known, it is possible to determine the distance by forming the triangles, as shown in Fig. 2.2.

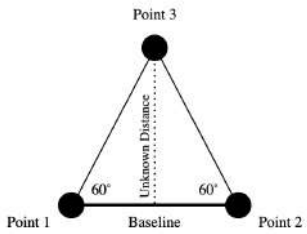


**Figure 2.2:** Triangulation

One of the critical facts regarding this technique is number of points; as it relies on a triangle to compute the angles, three points are needed to form the triangle. It is a method of computing position that is used in conjunction with other methods. In contrast to Trilateration, angles are employed to estimate the location of an object in addition to distances from the sensor in this method. Angles and a single length are required for a two-dimensional location determination. The angle of arrival (AoA) of a signal refers to the angle at which the signal arrives at the receiver. The angle is calculated based on the delay of the signal (also known as Time of Flight or, in short, ToF). It is the length of time it takes for a signal to travel between an object and the known location of a transmitter measured.

### 2.1.3 Fingerprinting

Another way of determining location is fingerprinting, which is most often used with position determination utilizing Wireless Local Area Network (WLAN) technology to determine a location. To find the distance between objects and reference locations, the fingerprint technique evaluates signal properties such as Received Signal Strength Indication (RSSI) and Media Access Control (MAC) address, rather than measuring the distance.
RSSI is a metric for determining the received power from a cell site when including noise and interference[**58**]. Equation. 2.7 represents how RSSI is calculated.

$$RSSI = P_t - P_L(d) \tag{2.7}$$

where $P_t$ is the signal transmission power and $P_L(d)$ is a path loss when the distance is equal to d and units of both are $dBm$. Equation. 2.8 represents $A$, received signal strength from a reference point when distance is $d_0$.

$$A = P_t - P_L(d) \tag{2.8}$$

RSSI is recognized as the RSSI value obtained from several measurements, as Equation. 2.10 indicates.

$$RSSI = A - 10nlog(d) \tag{2.9}$$

where d is referred to as the unknown distance.

$$d = 10^{(A-RRSI)/10n} \tag{2.10}$$

Due to its excellent accuracy compared to other techniques, fingerprinting is the most preferred localization method [**59**]. It is not dependent on line-of-sight measurements of Access Point (AP), is simple to implement, and has a wide range of use in the complicated indoor environment [**60**] [**61**]. In addition, fingerprint uses the MAC address for finding the position. MAC address is unique same as a human fingerprint. Every time devices connect to the network, it should provide a MAC address that uniquely identifies.

## 2.2 Wireless Technologies for positioning

As stated in Table 2.1[**62**], techniques discussed in 2.1 contribute with different technologies to estimate the location of users with a specific range of accuracy.

**Table 2.1:** Indoor Localization Techniques and Wireless Technologies

| **Technology** | **Technique** | **Accuracy (m)** |
|---|---|---|
| Satellite | Trilateration | 3–5 |
| WLAN | Trilateration<br>Fingerprinting<br>Triangulation | 10 (proximity)1–5 |
| Bluetooth | Trilateration<br>Fingerprinting | 2–5 |
| UWB | Trilateration<br>Triangulation | 0.01–1 |

Specific criteria such as accuracy may influence the final decision to select an appropriate IPS. Each building has some specific constraints; For instance, cost becomes critical if IPS is developed in towers. Another example is hospitals; coverage should be prioritized since the floors include various rooms. This section will discuss several technologies by providing an example of the implemented system, coverage, advantages, and disadvantages.

### 2.2.1 Satellite

Nowadays, people use their smartphones to identify their location in order to navigate to a specified location. Specific applications, such as Google Maps, display the best route between the user and the desired destination on the map. There are four unknown parameters in three dimensions: user coordinates $(x, y, z)$ and errors

that occurred during the transmission of signals. GPS consists of three different parts, which are demonstrated in Fig. 2.5.

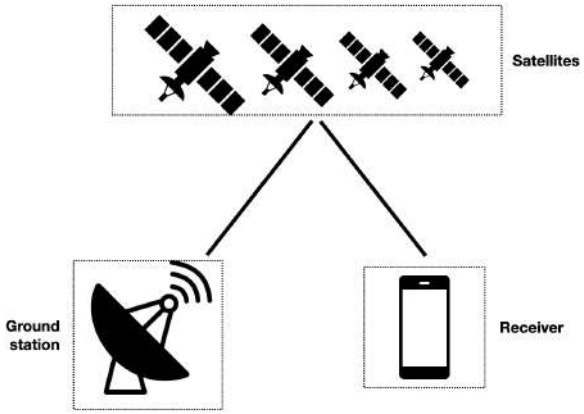


**Figure 2.3:** GPS components

- Till November 11, 2021, 30 operational satellites fly in circular orbits, not including the decommissioned on-orbit spares [**63**]. Since four unknown parameters should be calculated, at least four satellites are required to discover users' positions.
- The ground station monitors the position of satellites and communicates with them.
- A receiver can be a person who tries to find the location or navigate to a specific destination. The receiver should be listening to the signal transmitted by satellites continuously.

The receiver computes the distance between itself and satellites to find the position. Equation. 2.11 illustrates the required equation for calculating the distance.

$$\rho_j = \sqrt{(X_j - X_u)^2 + (Y_j - Y_u)^2 + (Z_j - Z_u)^2} + b_{ut} \tag{2.11}$$

where $(X_j, Y_j, Z_j)$ are the coordinates of satellites $jth$, and they are known. $(X_u, Y_u, Z_u)$ are the coordinates of the receiver and should be calculated. $\rho$ is called pseudo-range because it is not exactly a range or distance due to the error $(b_{ut})$.

The exact value of the range can be calculated by Equation. 2.12.

$$R_j = \rho_j - b_{ut} \tag{2.12}$$

However, Equation. 2.12 is not realistic since signals are constantly subject to errors due to environmental factors such as transmitting signals via the atmosphere. Nowadays, more sensitive GPS chips are utilized to receive signals from sufficient satellites to establish an object's location inside a building. However, the result of locating a point is insufficiently precise [**64**].
Kerem Ozsoy[**65**] developed an IPS based on GPS repeaters and a modified positioning algorithm. Their result indicates a maximum inaccuracy of 5 meters with a minimum error of 1 meter. Their measurement with error is shown in Table 2.2.

**Table 2.2:** Summary of Measurement Results

| **Distance from repeater(m)** | **Calculated position(m)** | **Error(m)** |
|---|---|---|
| 12 | 11 | 1 |
| 12 | 9 | 3 |
| 18 | 13 | 5 |
| 18 | 15 | 3 |
| 27 | 31 | 4 |
| 33 | 34 | 1 |
| 50 | 53 | 3 |

[**65**] results indicate that with more research, it is possible to develop an IPS based on GPS with more efficient performance in the future because an IPS error range of 1-5 meters is unacceptable in certain buildings, such as partitions in an office, where the distance between two rooms may be less than 1 meter. In general, if the distance between rooms is greater than usual or if accuracy is not critical, GPS may be a useful option due to the following advantages:

- GPS is a free service, and the application is available both in Android and iOS operational systems.
- Implementation does not require additional costs since no additional devices are needed.
- Security depends on Google company (in Google Map application), and developers are not responsible.

On the other hand, some disadvantages justify the usage of other systems:

- Coverage is just within the first floor[**62**].
- Still, the error is considerable even when the devices such as GPS repeaters are used[**65**].
- It is not flexible enough, and it is impossible to customize the User Interface.

### 2.2.2 Radio-frequency identification (RFID)

The RFID concept is based on the use of tags, with each tag being recognized from the others via the use of a mark, as demonstrated in Fig. 2.4.

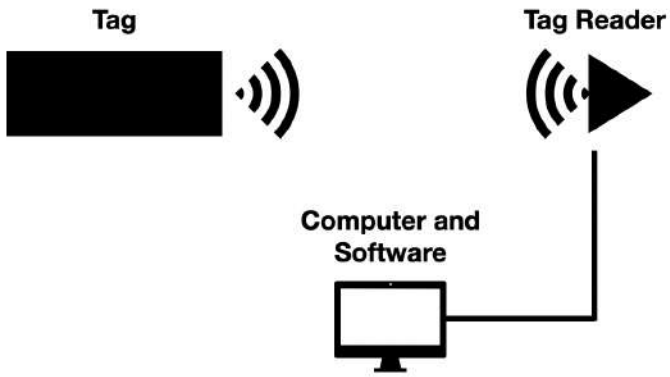


**Figure 2.4:** The operation of the RFID system

Once the Tag is placed in the vicinity of the tag reader, it extracts data of the RFID tag and sends it to the computer for further processing, such as finding the Tag's location. RFID is divided into two categories: passive and active. Each of them exhibits different characteristics over a wide variety of frequency ranges. The operational range of the device is the key point of distinction between passive and active devices. While passive RFID has a range of 1-2 meters, active RFID may be used in cases where a longer working range (10-100 meters) is necessary.

RFID in the industry may be used to provide a smart system for tracking pallet lifter. In this example, each pallet lifter has a QR code, and by scanning it, the precise moment of the pallet lifter's arrival and departure can be automatically recorded. The RFID reader can read the data contained in the RFID tags in this scenario. Another example is monitoring the health of farm animals; farmers use RFID to identify and manage livestock inventory and record critical health data for

each animal. One of the benefits of RFID is its low cost; that is why the system's usage in IoT projects has grown in recent years. Multiple ranges, scalability, and an affordable price are only a few reasons convenient researchers utilize RFID in the IPS. There are a variety of RFID applications that demonstrate the significance of that, such as book tracking in the library, toll gates, electronic Passport, and healthcare. According to [**22**], there are positive and negative points regarding RFID in hospitals,

**Advantages**

- Allows the hospital to keep track of the temperature of heat-sensitive drugs.
- RFID data can be saved, controlled, and shared in the cloud.
- Passive RFID tags are inexpensive and compact, making them ideal for attaching to small devices.

**Disadvantages**

- Installing active tags might also be costly.
- An active tag can only communicate through WiFi.
- To receive messages, active tags need a receiver infrastructure.

LANDMARC (Indoor location sensing using active RFID) [**23**] was a project that studied active RFID tags and developed an IPS system by attaching them to objects and installing readers in fixed positions.

### 2.2.3 Wireless local area network (WLAN)

Industrial protocols based on Ethernet provide several benefits, including full-duplex communication without collisions, large bandwidth, and high communication reliability [**24**]. With the proliferation of network systems, the cost of implementation became more critical than ever. That is why focusing on a wireless network is a topic that enables network systems to be implemented more effectively and with less expense.

Fig. 2.5 depicts a situation in which three separate APs are used to determine the device's position. The RSSI value between smartphone and each AP is selected; the greater the RSSI value, the closer the smartphones are to the AP's location.

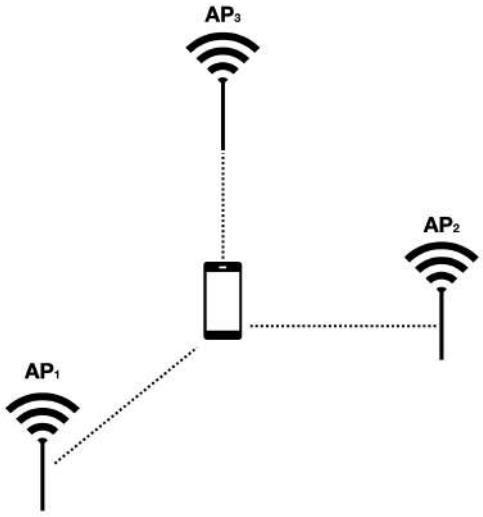


**Figure 2.5:** Localization based on WLAN technology

Positioning techniques are divided into two different categories [**25**],

- Infrastructure-based
- Infrastructure-free

The example of infrastructure-based technology are RFID [**26**], ultrasound [**27**], Bluetooth [**28**], and ZigBee [**29**]. Wi-Fi-based indoor localization has attracted researchers to use it for IPS because a Wi-Fi-based indoor localization system is both affordable and accessible [**30-31**].
The wireless indoor positioning system operates by establishing coordinates with the help of Wi-Fi access points capable of transmitting specific data. The idea is to share data wirelessly through an access point. However, there is a restriction in terms of range ; this technology has been applied in various scenarios. Some restrictions like the problem with range prevent users from working with WLAN in some sectors, such as agriculture. When the target is an IPS, the area is limited to the boundaries of the building. That is why the IPS is one of its uses. The location of an item could well be determined by examining various WLAN access points.
RADAR (an in-building RF-based user location and tracking system) [**32**] was one of the first IPS systems based on the WLAN. Many industries, from transportation and logistics to industry, offices, exhibitions, and museums, rely on indoor location-based Wi-Fi to function properly and efficiently [**33**]. Also, [**34**] mention the advantages and disadvantages of using Indoor positioning using Wi-Fi.

**Advantages**

- It is possible to make use of the existing Wi-Fi infrastructure.
- It is sufficient to have Wi-Fi activated.
- Range up to 150m.
- Detects the level of the floor.

**Disadvantages**

- Unreliable in terms of accuracy (5-15m).
- There are no consistent latency periods.
- When a smartphone is not connected to a Wi-Fi network, it uses a randomized MAC address, increasing users' privacy by allocating a randomized and autonomous MAC address instead of a static address for the connection [**67**]. However, the negative side of using a randomized MAC address is stability, which requires reconnection periodically [**68**]. That is why it cannot work with real-time systems such as IPS.

- iOS devices are unable to do client-based positioning via Wi-Fi networks.

### 2.2.4 Bluetooth

Bluetooth is a wireless communication technology standard that enables short-range connection between electronic devices [**35**]. Short-range means 10-100m in version 2. Nowadays, the smartwatch and smart band utilize the Bluetooth connection since it is integrated into the phone and connects quickly; in this situation, the range is not an issue because consumers wear the smart bands, and the phone is usually close by. Bluetooth Low Energy (BLE) is a low-power wireless technology that enables devices to communicate with one another and is intended for applications that need low power consumption [**36**]. With BLE, the device can detect when it is within range of a beacon and can compute its location if it is within range of more than two beacons at the same time [**37**].

Beacons are small wireless transmitters that use BLE technology in order to make a communicate with nearby smart devices such as tablets and smartphones [**66**]. It continuously broadcasts a constant signal that is visible to other devices. An essential factor to note about beacons is that they use a combination of letters and numbers to broadcast at regular intervals. When beacons are in range, receivers such as access points, gateways, and smartphones can detect them. Beacons communicate in one direction, which means that although they emit signals to the receiver, they cannot read any information. Beacons work across three distance ranges, with each range performing a particular function [**70**], including the following:

- Far range distances: They are explicitly designed so that the device may do measures concerning hearing a beacon, such as passing by a retail store.

- Near range distances: They are intended to work when the receiver is in the same room as the beacon, such as when entering a retail establishment.

- Near range distances: They are designed to activate when the device comes extremely near to contacting a beacon, such as the point of sale in a retail store.

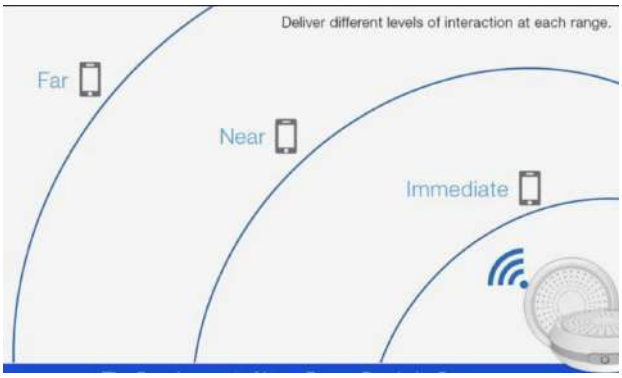


**Figure 2.6:** Different beacons ranges [**70**]

Beacons do not send any essential data; alternatively, they transmit an identifier, and receivers are responsible for processing and extracting useful data. In this case, receivers detect the broadcasted signal and convert it to a useful message.

[**38**] has developed an indoor positioning system based on BLE technology for tracking the everyday life patterns of elderly or disabled persons. Their results demonstrate that the placements of the BLE beacons on the user and the quality of the BLE beacons do not affect tracking accuracy.

In 2021, Omid Akbarzadeh [**41**] developed an IPS based on BLE systems for hospitals to control people's social distance during pandemics. Their solution aims to count individuals within hospitals using counter sensors and BLE. In the case of technology, it is possible to use a wide range of devices to detect the user's location inside the buildings. For example, Valerie Renaudin [**42**] stated that in large-scale structures such as malls, it is conceivable to achieve high-accuracy positioning without the use of beacons and simply by utilizing wearable lightweight sensors. Consequently, knowing how various technologies are used may result in improved performance. Based on the requirements of a different building, the purpose of each IPS method can be different. The technology can be determined by parameters such as cost, accuracy, and buildings' usage.

One of the applications of the BLE is in the area of health, where doctors inside the hospitals can track users or their health status can be sent to the doctors anytime needed. There is a list of advantages and disadvantages regarding the BLE.

### Advantages

- Low price
- Less complexity

### Disadvantages

- Because of the wireless transmission and reception, it is vulnerable to interception and attacks BLE [**39**].
- There are no consistent latency periods.
- Working distance is limited [**40**].

Despite this, some of the BLE's drawbacks cause issues in some sectors; nonetheless, it is worth noting that one of the reasons it is found in smartphones and other high-tech products is its pros.

### 2.2.5 Ultra-wideband (UWB)

Bluetooth, Wi-Fi, and UWB have a short-range and communicate through radio waves. The key distinction between them is that the UWB operates at a higher frequency. UWB is often the preferred IPS technology in complex and space-constrained applications such as industries tracking employees or product pallets.

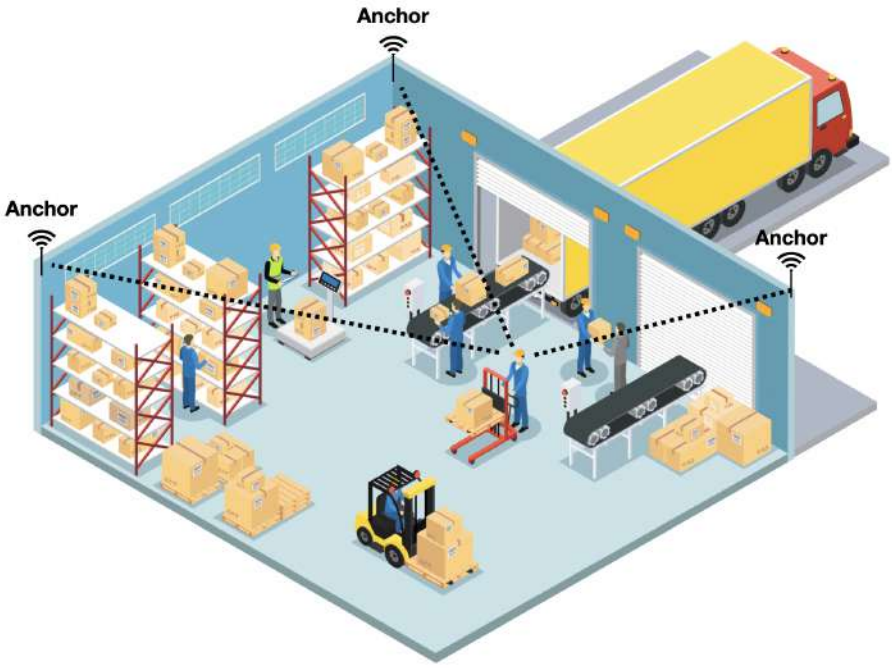


**Figure 2.7:** Localization based on UWB technology

As illustrated in Fig. 2.7, the worker is equipped with UWB tags. Each tag sends the signal to the anchor with a 3.1 to 10.6 GHz frequency range. Various tag types, such as forklift and AVG, are chosen depending on the final cost. In IPS based on UWB, the anchors play roles as receivers and capture signals transmitted by tags. Anchors collect UWB pulses emitted by tags and transmit them to the server. Anchors are electronic devices that detect UWB pulses emitted by UWB Tags and forward them to the server for calculating real-time tag positions. [**69**] mentioned the properties regarding using the UWB technology in the IPS.

- No interferences
- Transmission power 0,5 mW / 41,3 dBm/MHz (low power consumption)

- Reach 10 – 150 m (Acceptable coverage)
- Low data transfer rate (110 kbit/s – 6.8 mbit/s)

Ubisense [**43**] is one of the IPS firms that use UWB. It includes sensors scattered throughout mapped regions and tags.

## 2.3 User Interfaces

Generally, the UI for IPS is divided into two distinct platforms to enable the system to communicate. Among them is a mobile application, which should be designed for Android and iOS operating systems, and the dashboard develops and deploys on the Web for accessibility through the Web. The UI to use is determined by many considerations, including the final price, the level of security, the application of the system, and the type of user interaction necessary. For instance, when consumers at shopping malls want access to various sections, a mobile application is more helpful than a dashboard since access is easier than using a Uniform Resource Locator (URL). On the other hand, the dashboard is required for more complex tasks such as data management and analysis; as a result, it is more beneficial for situations where the IPS is designed for a company. Designed UI should be able to provide the following features:

- Capable of operating if the number of users increases.
- Customizable according to user requirements.
- Design with an appropriate User Experience (UX) rate to allow users with different knowledge to work without problems.

Generally, developing UIs for mobile and website provides competitive benefits. However, they need a nearly similar budget for implementation, including server costs, database costs, and security. Additionally, a technician is constantly on duty to monitor performance and should resolve it immediately in the event of a malfunction. This thesis provides a solution for hospitals with a website and mobile versions to add flexibility to the designed IPS and differentiate it from others.

UIs locate between final users and the system. IPS designed in this thesis for hospitals and should provide an interface for doctors and nurses to manage patients during treatment, which may take time from an hour until several days. Therefore, UIs should always be available and accessible during days and allow the final users to work anytime they want. As stated in Fig. 2.8, UIs play a role as a bridge between IPS and final users.

The website version is based on the dashboard, but this research relies on Telegram Bot for the mobile version, which provides benefits that will be discussed in 2.3.2.

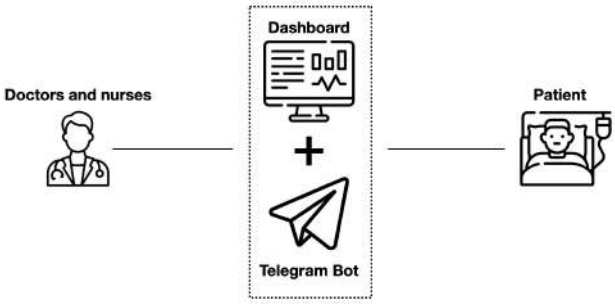


**Figure 2.8:** The role of UIs in IPS

### 2.3.1 Dashboard

The Dashboard website is utilized when there is a requirement for interaction between the system and its users. In this scenario, one component is critical, and it is referred to as UX, and it is the factor that decides how satisfied people are with the dashboard while they are using it. Once an IPS is designed and deployed in a hospital, it must be used by doctors and nurses as final users. The dashboard's responsibilities include the following:

- Managing patients (add, remove and track them during their treatment process).
- Consider security in the hospital, and patient data must be fully protected.
- Flexible to extend features in the future.

Each dashboard includes two different sections, front end, and back end. The front end is responsible for providing the eye-catching design with the most suitable functionalities based on the requirements of the final users. The back-end is in charge of developing the functionalities of the dashboard, it defines how elements described in the front-end phase works.
[**46**] provided the top seven of the programming languages which are most appropriate to develop the back-end of the website Table 2.3 shows the list of them with developed examples.

**Table 2.3:** Top 7 Programming Languages for developing back-end

| Programming Languages | Example |
|---|---|
| JavaScript | Facebook, Google, and eBay, |
| Python | Spotify, Pinterest, and Instacart |
| PHP | WordPress, MailChimp, and Flickr |
| Java | LinkedIn, IRCTC, Yahoo |
| Ruby | Airbnb, Shopify, and Slideshare |
| Golang | Dropbox, SoundCloud, and Dailymotione |
| C# | GoDaddy, Marketwatch, and Stack Overflow |

Selecting the final programming languages sometimes depends on the personal experience, provided backend with powerful functionalities, and adapting well by the front end to provide smooth service to the final users. Moreover, each programming languages for backend development have different properties.

- JavaScript is a straightforward language to learn and use when the graphical part of the website takes precedence. On the other hand, computing tasks are

heavy, and in the real-time project may result in a delay in executing the code and displaying the output.

- Python is famous for its simplicity, and also it is one of the flexible programming languages. However, it is not recommended for large-scale applications for the poor security and the lack of memory allocation.

- PHP is one of the oldest programming languages and is often challenging to learn. In addition, using PHP needs a high level of coding knowledge since non-optimized code significantly increases the time complexity. However, it is a free source and provides a high level of security, and it is a build-in Database connection, which helps to connect to the Database quickly and smoothly.

- Java is another programming language used for developing backend; it is object-oriented and straightforward, so it attracts backend developers. But for real-time projects such as IPS, it is not the best option since it is slow and increases the time execution of code.

- Ruby provides a good level of security; it is object-oriented and can be implemented on many platforms. The problem of Ruby is the same as Java; it is slow and cannot be used in IPS.

- Golang is a programming language provided by Google. It is fast with an excellent interface and easy-to-learn syntax. But poor library support makes it difficult for other developers to modify it. One of the essential points in UI development is that any developer must customize it over time. Once one developer develops the backend, it is difficult for others to change it.

- C# is one of the high-speed programming languages supporting well different libraries. The problem with C# is that it is not secure enough.

The dashboard developed in this research focuses on three aspects. First of all, it has a high level of security to guarantee the hospital in the case of protecting patients' data. Moreover, it connects well to the Database and works smoothly and fast. And in the last point, it is appropriate for a real-time system therefore executing time is acceptable. That is why PHP is the back-end programming language used to develop Dashboard in this thesis.

### 2.3.2 Telegram Bot

Telegram has grown in popularity as a social media platform over the last several years. Telegram was the seventh most downloaded app on iOS and Android in August 2021, according to [**44**], and its user base has been expanding at a pace of

**Table 2.4:** Telegram users in Italy

| | |
|---|---|
| Install penetration | 32% |
| Downloads | 4.2M |
| Unique installs | 1.4M |
| Daily Active Users | 4.6M |
| Monthly Active Users | 9.0M |

more than 40% each year since its release in 2013. Table 2.4. gives the information regarding Telegram users in Italy [**45**]

Telegram Bot is a service provided by Telegram company freely. Making Telegram Bot is easy, and bot father service is a feature that allocates users a unique token, which should be kept safe for privacy reasons.

### Advantages

- Totally free
- It can be developed by several programming languages such as Node Js, Python, PHP, Java, Go, and .NET.
- Provide BotFather, which can be quickly built and managed Bot.

As stated in Fig. 2.8, the Telegram Bot role is the same as Dashboard in this thesis, but why is it needed? As mentioned in 2.3.1, Dashboard is the most suitable UI for professional purposes. However, using Dashboard has three limitations:

- The cost of developing, implementing, and maintaining is considerable. It needs a host to store data and domain for deploying and make it accessible for all final users. Moreover, the treatment process of patients is unpredictable; as a result, it must have a high capacity to receive information.
- A specialist constantly checks the security status and prevents possible attacks.
- It is based on the URL and should be accessible through the web. However, doctors and nurses may have limited access to personal computers during the day in hospitals.

A system must be developed that operates in parallel to provide the maximum degree of satisfaction to address relevant limitations. A Token defines a Telegram Bot, a unique identifier that distinguishes it from other Bots. All that is required of the hospital is to protect the Token, which is simple to accomplish since once a Telegram Bot is built, the Token is given to the Bot's owner, and access to it is only accessible via the owner profile. Fig. 2.9 illustrates the process of creating a telegram Bot.

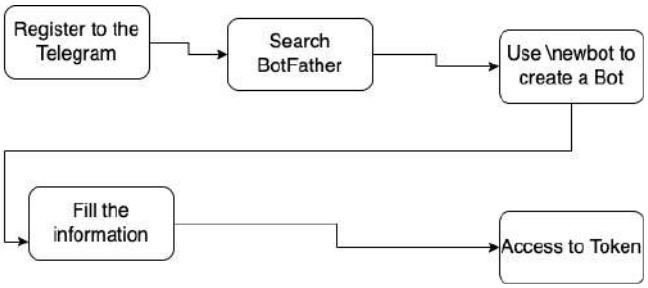


**Figure 2.9:** How to define a Telegram Bot

In the first step, an account should be created in Telegram. In the next step, by searching @$BotFather$, users can define and customize their Bot. After completing Telegram Bot registration, a Token is sent to the users. Telegram Bot is a free service that does not need a host or domain. As a result, the final price of implementing the proposed IPS may be decreased dramatically. Additionally, doctors and nurses can be accessed through personal computers and cellphones. However, Telegram Bot resolves issues with the Dashboard; it has some limits:

- It is not possible to customize it based on the hospitals' requirements.
- Connection to the Database is not easy since it needs higher knowledge in managing the Database.
- The website is a company symbol, while Telegram is just a social media.

Telegram Bot and Dashboard are developed in this research to provide the highest level of satisfaction to the final users and increase the IPS level by offering a comprehensive system.

# Chapter 3

# Implementation and Results: IPS

## 3.1 Hardware and software architecture

Each IPS is designed based on hardware and software that meets the needs of the final users. In this study, the final users of IPS are doctors and nurses working in hospitals who want to track patients' movements. Hospitals have many characteristics that may influence the final design choice:

- Each floor of hospitals consists of several rooms, and the number of rooms is more than other building types such as houses or offices.
- Distance between each room is not predictable; as a result, IPS should be supporting a different range of coverage.
- The treatment procedure for patients might take anything from a few minutes to several days. As a result, the signal transmitter must constantly operate for an extended period. Additionally, the patient population is not constant and is continually changing.

As stated in Fig. 3.1, three different devices with three communications protocols are needed to implement the system, and a Database is considered for storing, processing, and managing data for visualization through the dashboard and Telegram Bot.

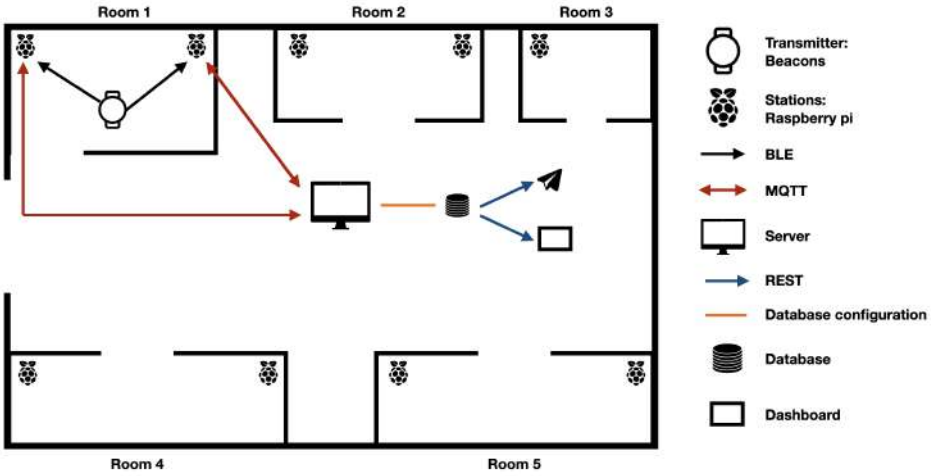


**Figure 3.1:** A workflow diagram illustrates the connection between devices

The devices are chosen based on two criteria: they must be affordable in the long run while also being strong enough to locate patients in hospitals. Table 3.1 summarizes all of the devices.

**Table 3.1:** Devices, softwares, and programming languages used in the project

| Devices | Softwares | Programming languages |
|---|---|---|
| BLE beacon | Telegram | Python |
| Raspberry Pi | TensorFlow | PHP |
| Server | Google Map | HTML |

All Devices, softwares, and programming languages mentioned in Table 3.1 will be explained in the following sections.

### 3.1.1 Transmitter: Beacons

As seen in Fig. 3.1, the initial phase involves the distribution of bracelets to each individual inside the departments by hospitals. According to [**47**], 90% of American consumers presently use a wearable fitness tracker. Nowadays, consumers embrace smart wearables and may utilize them without hesitation. This may aid the researcher in expanding the use of this technology to other fields, such as smart hospitals. Developing smart bands for the health sector may provide some questions:

- Are there any side effects associated with treatment?
- Are they suitable for use during surgery?
- How much budget should hospitals spend on beacon use?

The BLE beacon is chosen in this study since it is a green technology with minimal power consumption; hence, there is no worry about its usage even in the surgical rooms. BLE beacon fitted with Panasonic CR2032, 3V batteries enable hospitals to use them constantly for six months, after which they may be reused for months by simply replacing the battery. One of their primary benefits is affordability; they are inexpensive and readily available. The hospital receptionist presents the patient with a smart band, which includes beacons and a battery and can be distinguished from others by its unique Mac address, which is guaranteed to be unique for a Bluetooth device; thus, patients are distinct from one another. MAC address contains 48-bit with six groups of two hexadecimal digits, and they are separated by either hyphens (-) or colons(.)[**71**]. Table 3.2 represents the MAC address of the beacons used in this research.

**Table 3.2:** Example of discovering beacons

| Beacons | MAC address |
|---|---|
| Number 1 | F8:F2:C5:76:DB:92 |
| Number 2 | C4:42:06:A5:66:13 |
| Number 3 | FD:50:AA:A4:E7:9F |

One of the challenges facing hospital receptionists is assigning MAC addresses to individual patients, which may be difficult for people with limited knowledge of information technology (IT). Each beacon's MAC address is printed on it to address the issue, as seen in Fig. 3.2.

**Figure 3.2:** How to access the beacon's MAC address?

As can be seen in Fig. 3.2, the MAC address can be accessed easily by scanning the QR code. Once beacons start transmitting signals, the receiver's role begins since it must identify, separate, collect, and transmit the signals. The connection between beacons and stations is BLE, which is discussed in 2.2.4.

### 3.1.2 Stations: Raspberry pi

In the IoT project, Raspberry Pi is popular since it is able to become a bridge between two devices, communicate with various protocols, has a specific Operational System (Raspbian), and is significantly cheaper than the computer. Stations in this thesis are Raspberry Pi, a single-board computer to perform computation like the personal computer but with a different, mentioned by [**72**] in Table 3.3.

**Table 3.3:** Raspberry Pi Vs Computer

| | **Raspberry Pi** | **Computer** |
|---|---|---|
| **Memory** | Between 1 and 8 GB | Up to 4GB |
| **Storage** | Micro SD card | Hard Drive or SSD |
| **Architecture** | ARM | AMD64 |
| **Screen** | No screen | Can have a screen |

Raspberry Pi is more expensive than other gateways such as ESP32, but it can be run faster, and if the system wishes to enhance and include additional methods such as image processing, they may execute programs. Additionally, IPS can be developed by older versions of Raspberry Pi, but version 3 or above is suggested by this research. Once beacons are ready to send signals, the station can detect the BLE MAC address and separate it from the other Bluetooth names by the format defined for stations to quickly find beacons' MAC addresses. This specific format is a pattern to recognize a MAC address pattern and includes numbers and characters defined in 2.2.4. Consequently, even if several Bluetooth devices are turned on or accessible, stations can identify beacons without issue. Stations collect signals and transmit them to the server through the Message Queuing Telemetry Transport (MQTT) protocol.
MQTT is a lightweight messaging protocol used when devices want to connect to networks with limited bandwidth resources. MQTT is used for machine-to-machine (M2M) communication or IoT connections and is divided into three parts:

- Publisher
- Broker
- Subscriber

Each device should be published a message under a specific topic. For example, the topic can be written as "temp" if data is related to the temperature. This topic is stored temporarily in the Message broker and is available for the specified interval for other devices interested in the "temp" topic. Broker waits for the specific time, which defined by the programmer and if cannot find any devices for subscribing, it removes the topic (broker does not store topic or message). Generally, MQTT Brokers are classified into two categories:

- Public MQTT Brokers such as Eclipse, Mosquitto, and HiveMQ, works with TCP port 1883, and most of the time, registration for using it is not required.
- Private MQTT Brokers such as CloudMQTT enable customers to utilize their TCP Port.

This research relies on Eclipse, which uses Broker Address: https://mqtt.eclipse.org/.

Fig. 3.3. illustrates how MQTT works in this research to communicate between stations and servers.

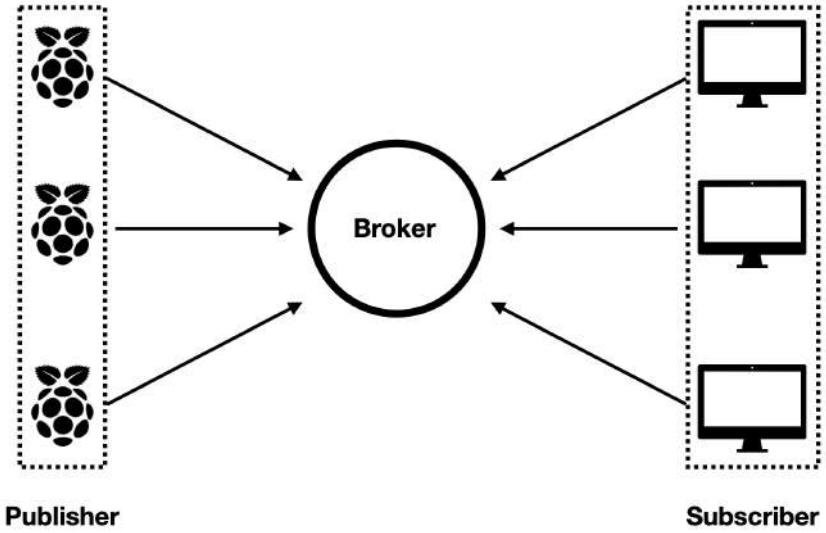


**Figure 3.3:** MQTT Diagram

When devices subscribe to the topic, they should determine the Quality of Services (QoS). In MQTT messaging, QoS is an agreement between publishers and subscribers on the assurance of a message. There are three primary levels of QoS [**73**].

- QoS 0: Each message is transmitted to a subscriber once without confirmation at this level. There is no way to determine whether subscribers receive the message.
- QoS 1: After trying to send the message, the broker awaits confirmation from the subscriber. If no response is received within the given time, the message is resent.
- QoS 2: A four-step handshake between the subscriber and broker ensures that the message is received.

At each point in time, the system starts at least one beacon to emit a signal, and when MAC patterns are established, stations detect and store them in an array in a JavaScript Object Notation (JSON) format for subsequent analysis. JSON is one of the standard text-based formats to represent data structure. It is used

in IoT projects to transmit data from the server to the client (subscriber) or vice versa. A primary advantage of JSON is security and privacy, which enables devices to communicate without sending data directly. In this case, instead of defining valuable parameters and configuration inside the code, it is possible to store it in the JSON, and then each device can read or write it directly. For example, when a topic is defined for the message, it can be stored in the JSON file, and publisher and subscriber at first open JSON file to read topic then they can use it. In addition, the Telegram Bot Token, which is discussed in 2.3.2, is also stored in JSON format. Rather than creating a list of beacons, stations also need to do two additional steps.

- Manage connected and not connected beacons
- Set the initial configuration of MQTT

Managing beacons refers to adding or removing beacons from the list. It means that when a patient leaves the hospital or the treatment process is finished, stations (Raspberry Pi) need to remove the MAC address and is available for the next patient. Moreover, stations are responsible for setting the configuration of the MQTT and storing it in JSON file, which includes:

- Beacon-id, Stations maintain the list of available beacons by adding and deleting them.
- Device-id: The device id is a list of all stations' MAC addresses that should be initialized whenever hospitals install the system. JSON files can be edited directly by IT specialists or developers to add or delete stations. Same as beacons, the MAC address of each station is also attached to the device, as stated in Fig. 3.4.
- Broker-IP: MQTT brokers manage communication between publishers and subscribers; in this case, the broker receives messages published by clients (in this case, stations), each message should be classified according to a specific topic in order to be distinguished from the others, and they also distribute to subscribers (here is the server). The final selection of the MQTT broker in this research is one of the public MQTT( Eclipse).
- Publish topic: Choosing the topic is entirely optional, and in this research is "update/sensors" as stated in Table 3.4.
- Subscribe topic: Subscribe topic should be the same as the publish topic, and it is "update/stations".
- Scan interval: The scanning interval specifies the amount of time required for stations to discover newly accessible beacons. It determines the time that

the station is looking for new beacons. For example, if IPS is developed for schools, students arrive at the school in the morning, and then by closing the doors, they are not allowed to enter. That is why time intervals can be defined by minutes. In contrast, as this research is being explored at hospitals, it is assumed that patients would arrive and go swiftly; hence, the scan intervals should be smaller than 1 second. Calculating the right time relies on various elements, including the velocity of transit to the building, the number of beacons, and the speed with which beacon owners' information is updated. In this research, as the assumption is deploying the system in the hospitals, therefore 0.5 seconds picked; consequently, after 0.5 seconds, stations are ready to detect new BLE beacons, making sense in the hospitals, particularly in the congested time.

- Send interval: The time required for stations to publish data to the server is referred to as the send interval, which is critical for doctors and nurses to maintain accurate patient location information. Signals gathered by the stations are provided to the server every 5 seconds; consequently, servers have enough time to subscribe to the topic published by the stations.

**Figure 3.4:** Stations

Table 3.4 summerize MQTT configuration, which defined by stations.

**Table 3.4:** Configuration of the MQTT

| Parameters | Values |
|---|---|
| Broker IP | Eclipse |
| Publish topic | update/sensors |
| Subscribe topic | update/stations |
| Scan interval | 0.5 (seconds) |
| Send interval | 5.0 (seconds) |

Once communication between stations and servers is done through the MQTT, the role of the server starts.

### 3.1.3 Server: Computer

In this research, the server might be a simple personal computer or a more sophisticated server. Stations are reasonable for collecting all the Mac addresses and signals, then further details such as the name, last name, and status of the patients can be assigned to the MAC address by the server. The process of connecting beacons to the stations is quick; In this case, it is based on BLE, which automatically detects the MAC address, adds it to the list, and converts it to the JSON as available MAC readable by the machine. The server is responsible for reading JSON files and performing three essential tasks on collected data:

- Store data in the Database.
- Perform Trilateration.
- Send data to the UI through Representational State Transfer (REST).

The first task of the server is to connect to the Database to store them for further processing. Generally, each Database has different properties, which help improve various aspects of IPS, such as security and time complexity. [**74**] mentioned the popularity of Databases, which is stated in Table 3.5.

**Table 3.5:** Database-Engines Ranking (February 2022)

| Database | Score |
|---|---|
| Oracle | 1251.32 |
| MySQL | 1198.23 |
| MongoDB | 485.66 |

Score refers to how often database names are searched on Google.

One of the reasons that makes Oracle's most popular Database is supporting all data types, and once an Oracle is developed, it can manage all the data by using one Database. Moreover, the backup and recovery process is smooth and can be done easily. Therefore, there is no concern regarding losing data [**75**]. On the other hand, Oracle is costly [**76**] and increases the final price of IPS, and since it is difficult to learn and does not have a User Friendly (UF) interface, it is not easy to modify by a non-expert Database manager.
In contrast, MySQL has reduced the total cost of ownership as it is entirely free, and the basic knowledge in Database is enough to work with data in MySQL. But, IPS is a real-time system that stores massive amounts of data during the running time. As a result, it needs a robust Database that works with large-scale data without problem. The issue with MySQL is that it is not as efficient at handling large database sizes[**77**].
MongoDB is another database with a score of nearly 486 based on Table 3.5. It is one of the Databases which needs high memory allocation and requires more physical memory to perform smoothly. However, it provides advantages [**78**] that represent why it is the best Database for IPS.

- It is highly scalable and can be used for a situation where there is a massive amount of data.
- It is simple to install and can be configured by anybody with a basic understanding of databases.
- MongoDB is an entirely free database. There is no expense associated with it. Because MongoDB stores data in JSON format, it is very straightforward to store arrays and objects. As mentioned in 3.1.2, data is stored in JSON format in this research. As a result, MongoDB works with the proposed IPS perfectly.

MongoDB is considered in this study to store and retrieve data, an open-source NoSQL database, which can handle documents, and supporting diverse types of data [**53**]. The station is responsible for forming the Database and initializing it for storing the data regarding users and rooms list—some attributes such as host, password, namespace, user, and port are defined in the server.

The second task of the server is to perform Trilateration. IPS has two different approaches, ranging positioning and no-ranging positioning [**48-49**]. The ranging algorithm is often used in conjunction with the received RSSI, the Time of Arrival (TOA), the Time Difference of Arrival (TDA), and the AOA positioning technique based on the signal arrival angle (angle of arrival) [**50-51**]. RSSI is a metric used to determine how effectively a device can pick up a signal from an access point or router [**33**]. One of the problems of the IPS is considering the strength

of wireless signals, which is unpredictable because of the multipath propagation and shadowing effects [**52**]. In this study, Trilateration is used to calculate the positioning of patients inside the hospitals based on the signals transmitted by beacons, which is explained in section 2.1.1.

The server's third and last responsibility is to send results, which are the patients' positions, to the Telegram Bot over REST, which will be explained in 3.1.4.

### 3.1.4 UI

Two distinct kinds of UI are explored for interacting between IPS and final users (doctors and nurses).

- Telegram Bot
- Dashboard

As discussed in 3.1.4, each UI provides specific features and limitations. Hospitals are free to choose Telegram Bot or Dashboard; however, the suggestion of this research is to use both of them to have all the features together, such as:

- Have accessibility from both smartphones and personal computers easily.
- If a problem for either Telegram Bot or dashboard occurs, there is a backup for IPS to work.

Both interfaces are developed based on specific purposes while following the same structures.

The connection between Telegram Bot and the server is based on REST, which uses HTTP requests to access and use data [**54**]. Fig. 3.5 illustrates how REST is working to exchange data between two devices.

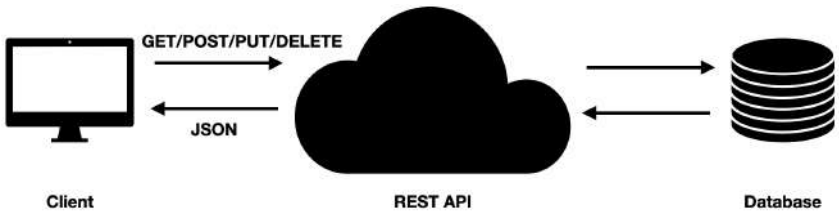


**Figure 3.5:** REST communication

Four methods are used to send a request from clients to the server.

- GET: It is a way for retrieving data from a server. As this is a read-only approach, there is no chance of data modification or loss. When nurses or doctors are interested in finding a patient's location in the hospital GET method is used.
- POST: It is a method that communicates with the server and initiates the creation of a new resource. For instance, when a new patient arrives, the POST method establishes an entity containing the patient's information, such as their name and surname.
- PUT: The most often used PUT method is to update an existing resource. For example, when the patient's name wrote wrongly, the PUT method is used to correct it.
- DELETE: The DELETE method is used to delete a resource. Patients leave the hospital; this method is used to remove it from the Database.

The Telegram Bot's purpose is to display the patients' locations while they are in hospitals. In this example, the server builds an active beacon table; If the beacon and server lose connection, the beacon MAC is removed from the table, and Telegram Bot displays an error indicating the loss of connection. Telegram Bot was built as a robust interface that allows the doctors and nurses to easily assign a MAC address to a single patient while also providing optional data such as name, surname, age, or gender.

The QR code idea enables doctors and nurses to detect the MAC address; it is an acronym for Quick Response code. It can store and save information and data, and doctors and nurses may rapidly retrieve it by scanning it through a QR code reader. However, there are two methods for assigning MAC addresses to users,

- Enter the address to the database directly.
- Send the picture of them into the Telegram Bot, and it detects the address and assigns it by using the Python Imaging Library (PIL). PIL includes lightweight image processing tools for editing, creating, and storing pictures [**78**]. It is powerful and valuable in this research since it can detect, read, and assign QR codes.

To add the name of rooms in the hospitals, staff should send the photo of the QRcode attached to the stations to the Telegram Bot as illustrated in Fig. 3.6 with the room's name, such as the surgery room. Moreover, the beacons' QRcode should also be sent by the hospital's staff to the Telegram Bot with the patient's name and last name, as stated in Fig. 3.7.


**Figure 3.6:** Adding a room to the system by Telegram Bot

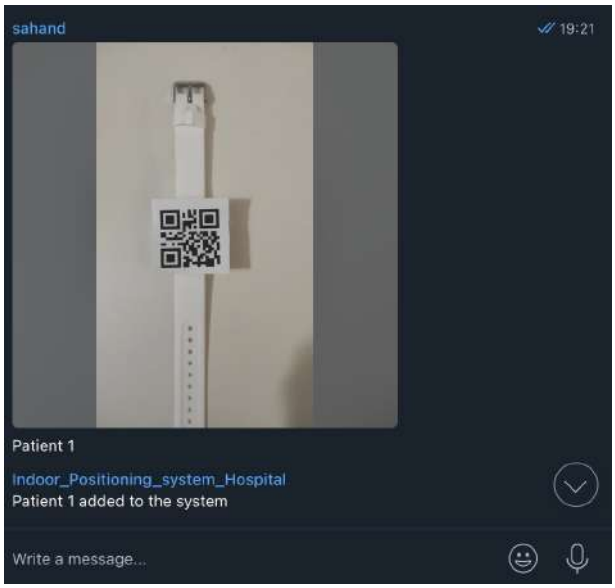


**Figure 3.7:** Adding a patient to the system by Telegram Bot

When a photo is uploaded to the Telegram Bot, it should contain a caption; otherwise, the hospital's staff will get an error message regarding the lack of a caption, as seen in Fig. 3.8.


**Figure 3.8:** Missing Caption error

Telegram Bot developed for this IPS contains different commands to allow doctors and nurses to access optional data whenever they need it. Fig. 3.9 demonstrates all the commands to handle the patients and rooms within the hospital.

- The room list provides the list of rooms determined by the photo sent.
- The patients' list is used to see the location of the patients.
- Where are all gives the position of all patients.
- Who is in the room shows patients in specific rooms.
- Delete patient is used when a patient wants to leave the hospital.
- Delete room, whenever hospital's staff wants to remove a room, they can use this command.

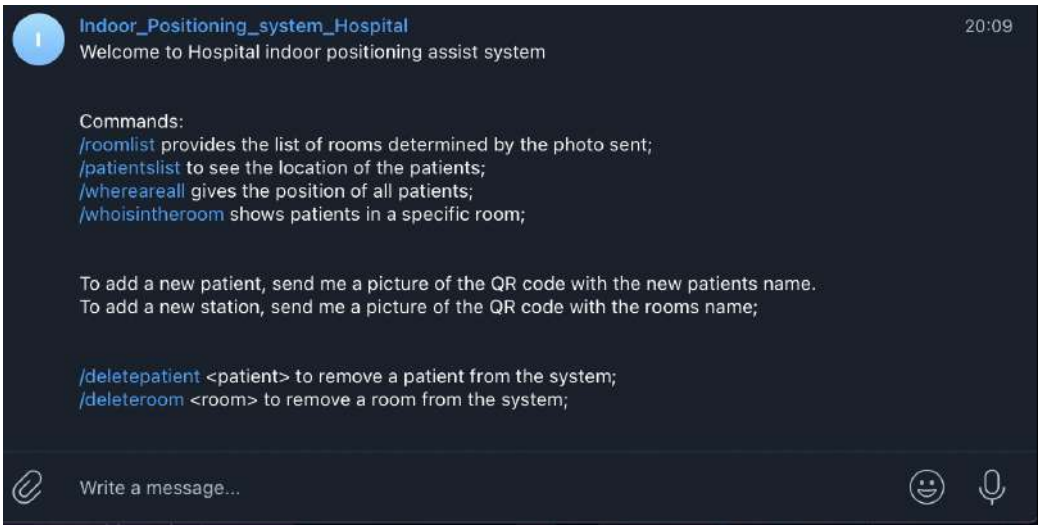


**Figure 3.9:** Telegram Bot commands list

Once registration is done, doctors and nurses can follow the location of the patients within the hospitals by using the commands. Fig. 3.10 illustrates the example of one command, which indicates the patient's location. In this case, patient number one is registered, and consequently, the MAC address is accessible and allocated to one individual. Whenever doctors and nurses use the command to understand the patient's position, Telegram Bot reacts to the command. If patient number 1 changes the position and moves to the waiting line, data is updated, and a new message is provided to the staff.


**Figure 3.10:** Result in Telegram Bot

Parallel to Telegram Bot, a robust dashboard was being developed. In terms of functionality, both perform the same functions with different advantages and disadvantages, as discussed in 2.3. A powerful dashboard is given if the hospitals desire a more professional UI. In this situation, it is feasible to manage the patients within the hospitals with any options required by considering MongoDB, in which 3.1.3 discussed the pros of using MongoDB in IPS. Rooms may be defined effortlessly, and all employees can be registered in the Dashboard to use it whenever they want (same as Telegram Bot, Dashboard is a UI developed for hospitals staff, and patients are not allowed to access it). The Dashboard has several pages.

In the first phase, after the registration, personnel can log in if their information, such as name, last name, etc., is present in the database. Fig. 3.11 illustrates the Dashboard's login page, which includes the user name and password for additional security.

After login, the hospital staff can handle the patients; Fig. 3.12 shows an example of adding new patients. The essential aspect is that fields are optional, and they depend on the decision of the hospitals' management to modify them whenever they want. MAC address section is unique and used to distinguish patients from others. All the MAC addresses are accessible in the bracelets' QRcode sticks, as stated in Fig. 3.7.

Fig. 3.13 illustrates the list of users in the hospitals, in which their location is specified; by clicking on the location icon, the room is revealed to the staff. Moreover, the search box makes it possible to quickly discover the patient in the list and delete or find their position.

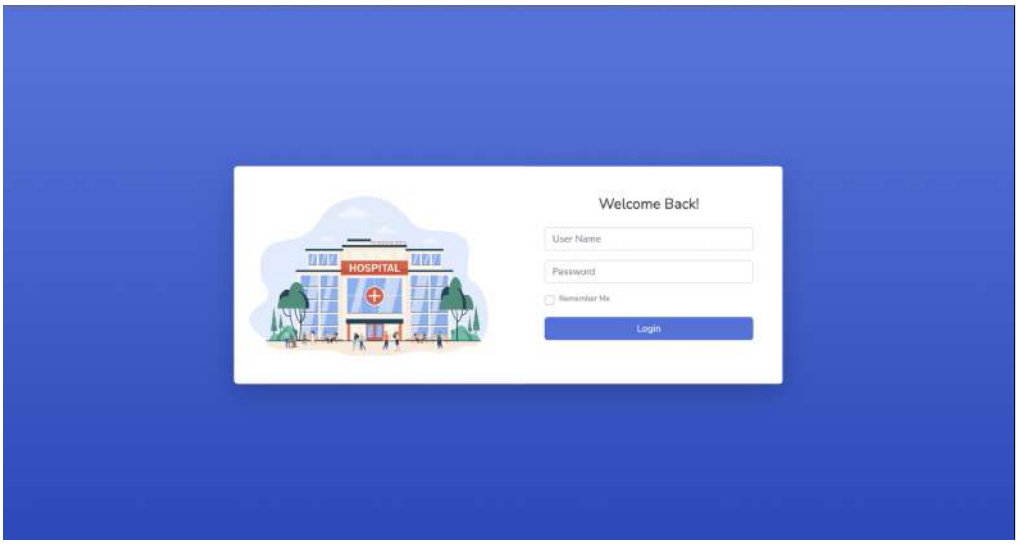


**Figure 3.11:** Dashboard Login page

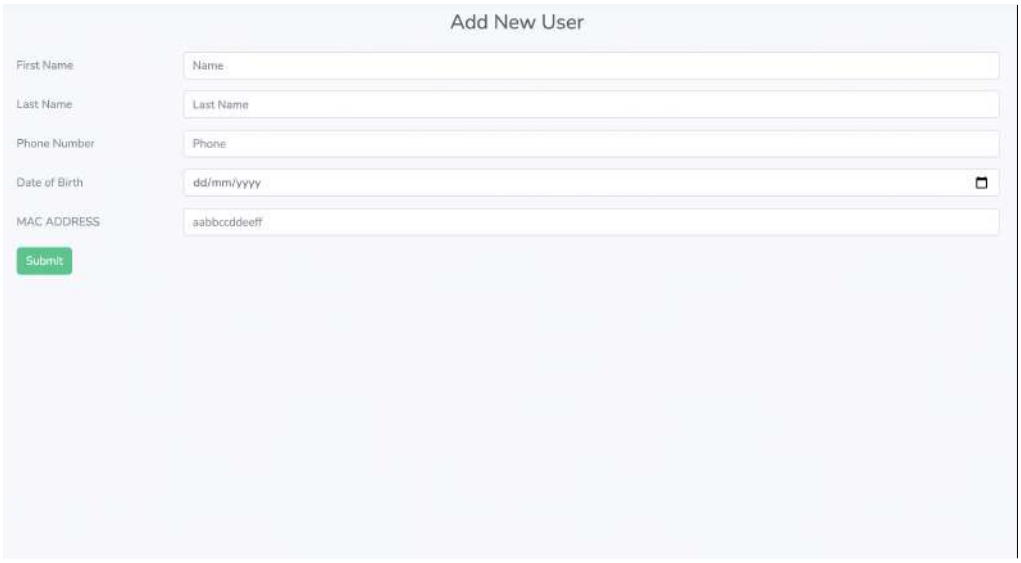


**Figure 3.12:** Dashboard "Add new user" page

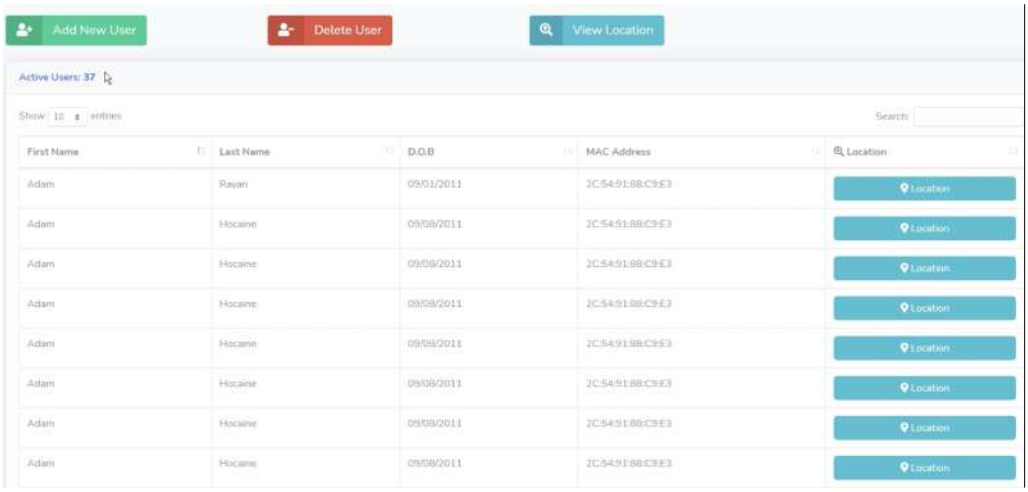


**Figure 3.13:** Managing patient inside the hospital

## 3.2 Dashboard validation

Evaluating the final user's satisfaction is an essential approach that should be considered. Fifty nurses answered the questions as a survey that was prepared on Google form regarding the dashboard performance.

According to Fig. 4.5, the UI design is the most significant aspect for the end-users. In addition, they feel that they didn't face any difficulties with accessibility, which shows that the design of the buttons and the overall structure of the dashboard have been defined completely. Color palettes assist designers in creating new colors by combining two or more colors. Additionally, palettes control the color's saturation, value (strength), and other color schemes. These characteristics of the colors influence the final appearance of the artwork [**79**]. In the UI, colors are essential since they attract users and increase UX.

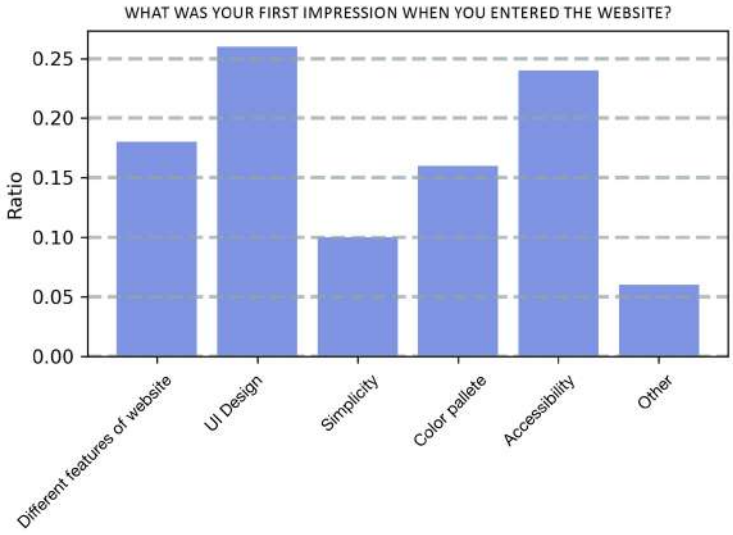


**Figure 3.14:** Dashboard first impression

Fig. 4.5 represents that most of the people who participated in the survey found the functionalities clear and emphasized not having weaknesses in this matter.

Since all the core codes of the dashboard design are synced, there were a few bugs in the final test of the dashboard. PHP was effective because it has powerful properties in IPS, which is discussed in 2.3.1 when using optimum programming languages such as JavaScript, CSS, and HTML since they are dynamic. Also, before deploying the dashboard on the web, some steps are done in different situations to test the dashboard step by step. Thanks to logical initial planning, the test time and debugging of the dashboard are completely considered.

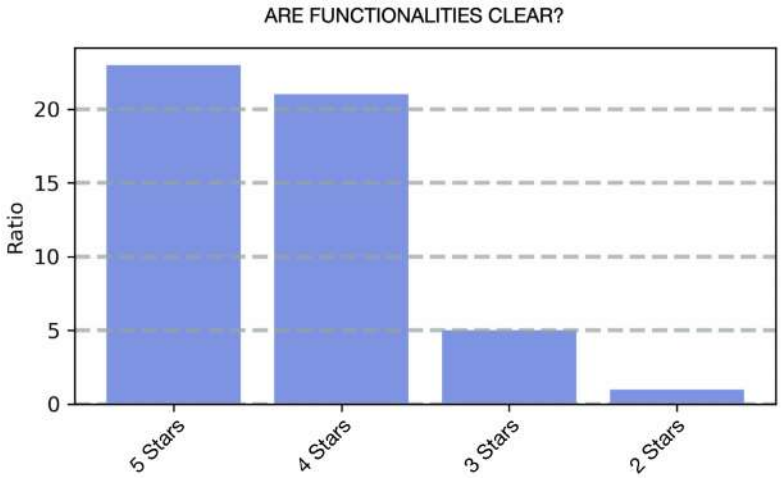


**Figure 3.15:** Functionalities

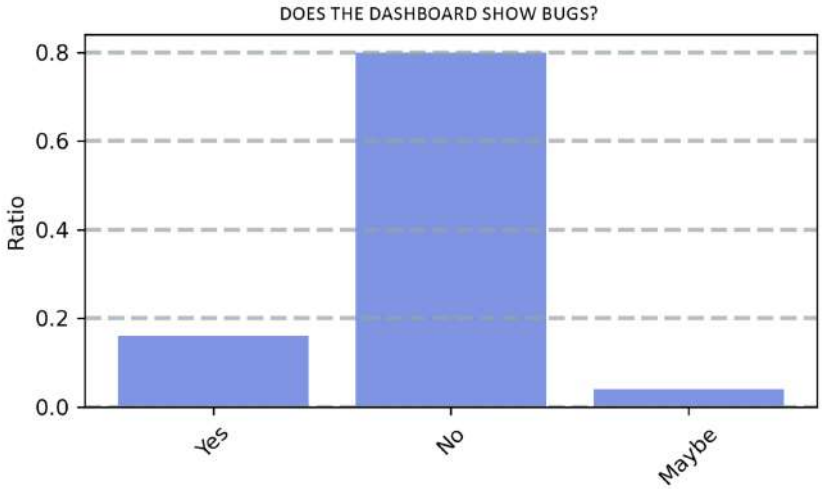


**Figure 3.16:** Bugs in the Dashboard

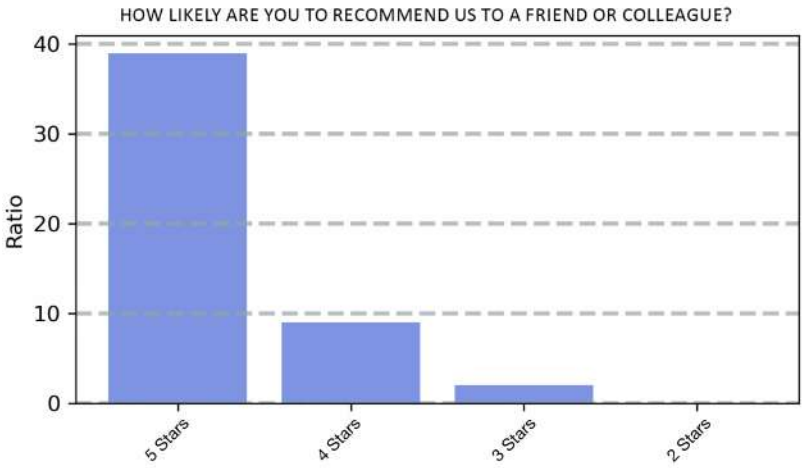


**Figure 3.17:** Users' satisfaction

As all the users who filled the form are nurses, their satisfaction and recommendation indicate that the dashboard implementation is successful.

# Chapter 4

# Implementation and Results: Hospital departments finder

Final users of Hospital departments finder are the patients who want to access the specific building in the hospital. This system relies on a web URL, enabling users to find a path between an origin and the desired destination. Hospitals are separated into buildings with various functionalities, such as the medical, physical medicine, and radiology departments. When visitors try to go to one of the buildings, two major issues occur:

- Hospitals cannot provide proper colored coding for navigation in certain circumstances, resulting in visitors' confusion.
- One of the issues that can be seen in any GPS system is the inability to track the device correctly; for example, in one of the most recent navigation systems, - Google Maps - identification of the position is calculated with a precision of 3-10 meters [**55**], which may have adverse effects when the scale of location is small.

To address these problems, it is conceivable to develop a local-based navigation system that does not depend on GPS, an internet connection, or even the activation of GPS. Visitors may reach their preferred location quickly and easily by using a smartphone and interacting with the system. In the context of geographic data structures, GeoJSON is a format that uses JSON to encode them [**56**]. The use of GeoJSON files was the most crucial component of the work. Hospital departments finder is a framework developed based on the GeoJson data format with two important geometry types. First, a polygon separates the different buildings to provide coordinates. The second is a LineString to demonstrate the specific path from a particular origin to a selective destination. The idea behind the system is

to extract the coordinates of various buildings from the main area (in this study's case, the central location is a whole hospital). If hospitals consist of different main entrances, each building can be accessed through the path by defining other start points, which the system shows. The system can guide the users on accessing each building by choosing another entrance based on preferences. Since the hospital established in this project covers different sections and regions, the system enables users to access them through a selecting list. Each list is shown respect the precedence according to the user preference selection. The study case in this work is Maria Vittoria hospital, located in Turin, Italy. In the first step, the principal coordinates of the hospital should be considered. Table 4.1 shows the latitude and longitude of the hospital.

**Table 4.1:** Coordinates of hospital under study

| latitude | longitude |
|---|---|
| 45.082080 | 7.656278 |

By using Geo Json website, it is possible to get access to any place or street by searching for it using the name or address of the building or street. The output of the first phase is the whole hospital; however, to construct the route, it is necessary to separate departments. The result of the first step is shown in Fig. 4.2. As shown in Fig. 4.2, finding the whole hospital is done completely. The second step starts when all the buildings are found. Using the polygon tools that the website provides, the border of each section is determined, then the related coordinates are extracted and made available in GeoJSON format. The fact is that the number of departments directly correlates with the number of GeoJSON files, which are drawn out with polygons. After determining the borders, the output has specific coordinates with the highlighted department, as stated in Fig. 4.3. The outcome is the GeoJSON file and demonstrates helpful information regarding the selected area, such as the total area based on the square meter, as seen in Fig. 4.1.

Moreover, patients can zoom in and change the location freely. It is important to note that it is necessary to clear all the coordinates related to the last ones to choose other departments. It means that final outputs should be independent of each other. This can be achieved by using the meta-clear option provided by the website. That is why, in order to ensure that the final findings are independent, it is crucial to clear the preceding step's resultant coordinates. Fig. 4.3 illustrates the highlighted region.

The route between the specified entry and the desired destination is determined in the next stage. This phase assumes a single point of origin and a single department; however, this model may be improved. With the help of an animated route color and the line tool, it is possible to establish the relative coordinates of two points

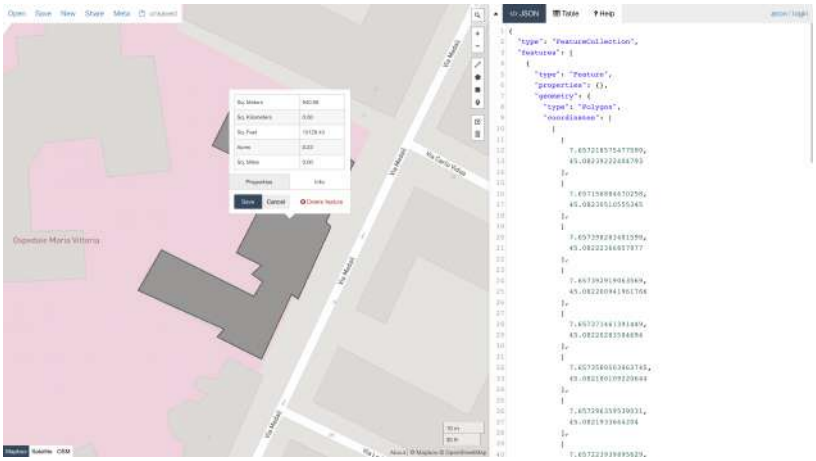

**Figure 4.1:** GeoJSON file example

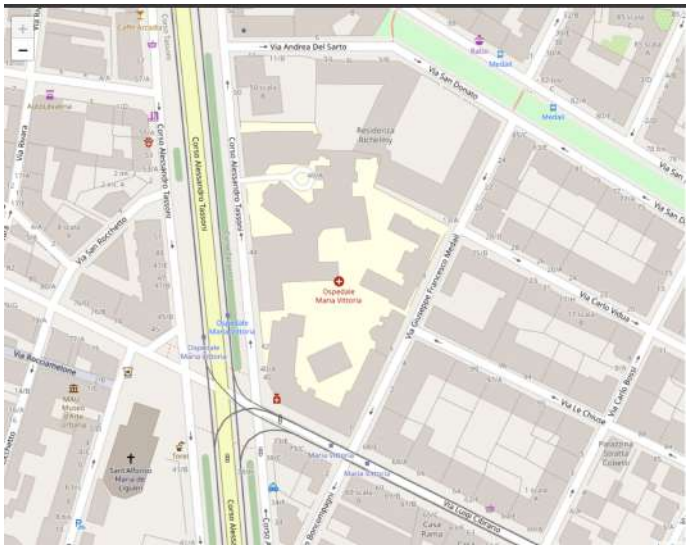


**Figure 4.2:** Coordinates of hospital under study in the map


**Figure 4.3:** Highlighted Department

and create a decent user experience. The route is also available in the GeoJSON format, like the polygon. Fig. 4.4 indicates the path that determines from the origin to the destination. However, having a different entry might vary the offered path. Patients may pick this instance when they wish to use other openings.

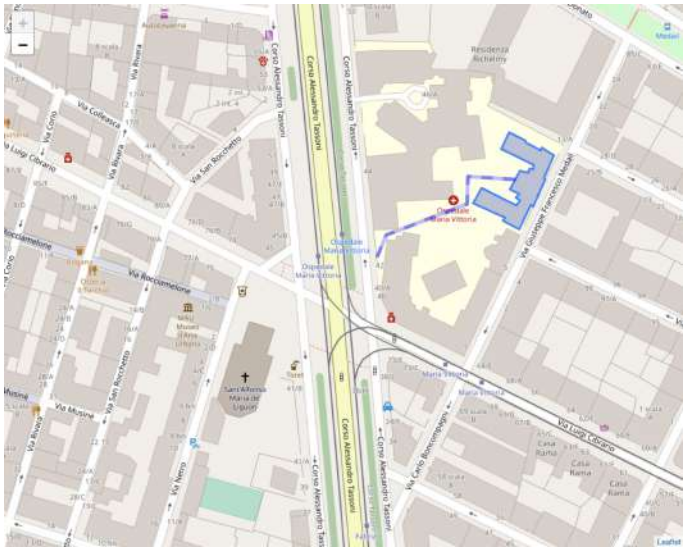

**Figure 4.4:** Route determination

Selecting multiple origins and destinations is one of the features that may help patients search among alternative pathways. TensorFlow's services were used to develop the widget, which is helpful to choice entrance and destinations as stated in Fig. 4.5. The goal is to concentrate on all available GeoJSON (both for routes and departments) and let users choose their preferences. Local storage is used to store selected sources and destinations. Consequently, since the programming saves all recommendations, the outcome is the same every time the user switches selections. As a result, one of the advantages of utilizing the system is that customers may enjoy the service as many times as they need.

One of the hospital department's finder benefits is storing data and using it for later study. In this scenario, hospital administrators and staff can locate the most inhabited departments, which is difficult to retrieve when using GPS application providers or colored coding for navigation. The point is that whenever a user picks a destination, this point is saved in the list, and at the end of the month, a *.CSV file with the relevant analysis is sent to the management. This may assist managers pick the most populous departments without surveys or extra data. Moreover, support them in controlling the quantity of personnel in the various departments and establishing a strategy for the future.

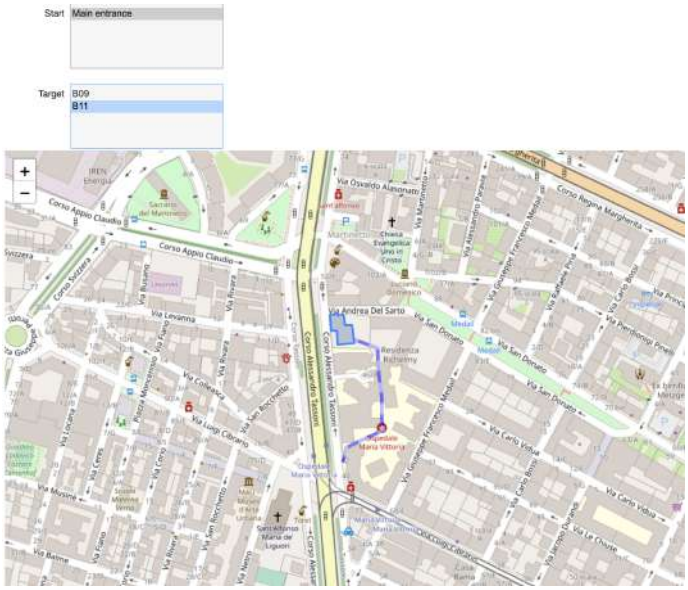


**Figure 4.5:** Widget option

# Chapter 5

# Conclusion and future extension

The proposed IPS in this research was inspired by two primary questions listed below.

- What is the final price of the IPS? Is there any possibility of managing it?
- How does design a scalable IPS work in a populated building such as hospitals?

In the first step, one of the challenges of each system, which needs interaction with final users, is the website. This study developed the Telegram Bot similar to the dashboard in the case of functionalities. This system offers the hospitals two options:

- A powerful dashboard can be used as the primary UI to interact with the doctors and nurses.
- By considering the Telegram, the final price of the implementing system reduces.

UIs designed in this research can work smoothly in the hospital since programming language and Database are entirely suitable for hospitals. Doctors and nurses can easily track the patients' positions without any problems, even during busy days, where the number of patients increases dramatically.

Hospital departments finder is a new concept in the positioning field, where without any devices, positioning can be done if the accuracy of GPS results in the wrong positioning. Further study can be developed by implementing the software, which merges both Hospital departments finder and IPS. In addition, providing a mobile application can be a good idea to reduce the final price and use it rather than Telegram Bot.

# Bibliography


[1] (S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain and K. -S. Kwak, "The Internet of Things for Health Care: A Comprehensive Survey," in *IEEE Access*, vol. 3, pp. 678-708, 2015, doi:10.1109/ACCESS.2015.2437951.) (PIMRC), Sep. 2016, pp. 1–6.

[2] `https://www.statista.com/statistics/976313/global-iot-market-size/#:~:text=The%20global%20market%20for%20Internet,around%201.6%20trillion%20by%202025.`

[3] (Sharma, N., Shamkuwar, M. Singh, I. The history, present and future with iot *Intell. Syst.* Ref. Libr. 154, 27–51 (2019).)
Kumar, A. Methods and materials for smart manufacturing: additive manufacturing, internet of things, flexible sensors and soft robotics. Manuf. Lett. 15, 122–125 (2018).

[4] Kumar, A. Methods and materials for smart manufacturing: additive manufacturing, internet of things, flexible sensors and soft robotics. *Manuf. Lett.* 15, 122–125 (2018).

[5] Ahuett-Garza, H. Kurfess, T. A brief discussion on the trends of habilitating technologies for Industry 4.0 and Smart manufacturing.*Manuf. Lett.* 15, 60–63 (2018).

[6] Sebastian, S., Ray, P.P., 2015. Development of IoT invasive architecture for complying with health of home. In: Proceedings of *I3CS, Shillong,* pp. 79–83.

[7] Z. Pang, "Technologies and architectures of the Internet-of-Things (IoT) for health and well-being," M.S. thesis, Dept. Electron. Comput. Syst., KTH-Roy. *Inst. Technol., Stockholm, Sweden,* Jan. 2013

[8] H. O. Alanazi, G. M. Alam, B. B. Zaidan and A. A. Zaidan, "Securing electronic medical records transmissions over unsecured communications: An overview for better medical governance", J. Med. *Plants Res.*, vol. 4, no. 19, pp. 2059-2074, 2010.

[9] M. S. A. Nabi, M. L. M. Kiah, B. B. Zaidan, A. A. Zaidan and G. M. Alam, "Suitability of using SOAP protocol to secure electronic medical record databases transmission", *Int. J. Pharmacol.*, vol. 6, no. 6, pp. 959-964, 2010.

[10] M. L. M. Kiah, M. S. Nabi, B. B. Zaidan and A. A. Zaidan, "An enhanced security

solution for electronic medical records based on AES hybrid technique with SOAP/XML and SHA-1", *J. Med. Syst.*, vol. 37, no. 5, pp. 9971, Oct. 2013.

[11] M. S. Nabi, M. L. M. Kiah, A. A. Zaidan and B. B. Zaidan, "Suitability of adopting S/MIME and OpenPGP email messages protocol to secure electronic medical records", Proc. 2nd Int. *Conf. Future Gener. Commun. Technol. (FGCT),* pp. 93-97, Nov. 2013.

[12] M. L. M. Kiah, A. Haiqi, B. B. Zaidan and A. A. Zaidan, "Open source EMR software: Profiling insights and hands-on analysis", *Comput. Methods Programs Biomed.*, vol. 117, no. 2, pp. 360-382, 2014.

[13] O. Alanazi, A. A. Zaidan, B. B. Zaidan, M. L. M. Kiah and S. H. Al-Bakri, "Meeting the security requirements of electronic medical records in the ERA of high-speed computing", *J. Med. Syst.*, vol. 39, no. 1, pp. 165, 2015.

[14] M. Hussain et al., "The landscape of research on smartphone medical apps: Coherent taxonomy motivations open challenges and recommendations", *Comput. Methods Programs Biomed.*, vol. 122, no. 3, pp. 393-408, 2015.

[15] M. Abdulnabi, A. Al-Haiqi, M. L. M. Kiah, A. A. Zaidan, B. B. Zaidan and M. Hussain, "A distributed framework for health information exchange using smartphone technologies", *J. Biomed. Inform.*, vol. 69, pp. 230-250, May 2017.

[16] Sana, F. et al. Wearable devices for ambulatory cardiac monitoring. *J. Am. Coll. Cardiol.* 75, 1582–1592 (2020)

[17] `https://www.polarismarketresearch.com/industry-analysis/wearable-medical-devices-market/`

[18] Rigden JS. Rabi, scientist and citizen 2ed. *Cambridge, Massachusetts: Harvard University Press;* 2000:1–283.

[19] `https://www.gps.gov/systems/gps/performance/accuracy/`

[20] Blunck, Henrik Kjærgaard, Mikkel Toftegaard, Thomas. (2011). Sensing and Classifying Impairments of GPS Reception on Mobile Devices. 350-367. $10.1007/978 - 3 - 642 - 21726 - 5_2 2$.

[21] Kunhoth, J., Karkar, A., Al-Maadeed, S. et al. Indoor positioning and wayfinding systems: a survey. *Hum. Cent. Comput. Inf. Sci.* 10, 18 (2020). https://doi.org/10.1186/s13673-020-00222-0

[22] `https://viehealthcare.com/the-pros-cons-of-rfid-technology-in-the-hospital/`

[23] L. M. Ni. Y. Liu. Y. C. Lau, and P. Patil, "LANDMARC: Indoor location sensing using active RFID," Wireless Networks, vol10, no. 6,pp. 701-710,2004

[24] Wisniewski, Lukasz Trsek, Henning Jasperneite, Juergen. (2011). WIRELESS LOCAL AREA NETWORKS AND ITS IMPACT ON CURRENT AND FUTURE PRODUCTION PROCESSES. Management and Production Engineering Review, *ISSN 2082-1344. 2. 78.*

[25] Ali MU, Hur S, Park Y (2019) Wi-Fi-based effortless indoor positioning system Using IoT sensors. Sensors 19(7):1496. https://doi.org/10.3390/s19071496

[26] Seol S, Lee EK, Kim W (2017) (2017) Indoor mobile object tracking using RFID. *Future Gener Comput Syst* 76:443–451

[27] Lindo A, García E, Ureña J, del Carmen Pérez M, Hernández Á (2015) Multiband waveform design for an ultrasonic indoor positioning system. *IEEE Sensors* J 15(12):7190–7199

[28] Zhou C, Yuan J, Liu H, Qiu J (2017) Bluetooth indoor positioning based on RSSI and Kalman filter. *Wirel Pers Commun* 96:4115–4130

[29] Aykaç M, Ergun E, Noor BA (2017) ZigBee-based indoor localization system with the personal dynamic positioning method and modified particle filter estimation. *Analog Integr Circuits Signal Process* 92(2):263–279

[30] Cui Y, Zhang Y, Huang Y, Wang Z, Fu H (2019) Novel Wi-Fi/MEMS integrated indoor navigation system based on two-stage EKF. *Micromachines* 10(3):198. https://doi.org/10.3390/mi10030198

[31] Beomju Shin, Jung Ho Lee, Taikjin Lee and Hyung Seok Kim (2012) Enhanced weighted K-nearest neighbor algorithm for indoor Wi-Fi positioning systems. *In: 2012 8th international conference on computing technology and information management (NCM and ICNIT)*, Seoul, 2012. pp 574–577

[32] P. Bahl and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. *Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, 2000, pp. 775-784 vol.2, doi: 10.1109/INFCOM.2000.832252.

[33] `https://navigine.com/blog/wifi-for-indoor-positioning-and-navigation/#:~:text=The%20wireless%20indoor%20positioning%20system,device%20using%20the%20multilateration%20approach./`

[34] `https://www.infsoft.com/blog/indoor-navigation-using-wifi-as-a-positioning-te`

[35] `https://www.britannica.com/technology/Bluetooth/`

[36] `https://www.novelbits.io/basics-bluetooth-low-energy/`

[37] `https://locatify.com/blog/indoor-positioning-systems-ble-beacons/`

[38] L. Bai, F. Ciravegna, R. Bond and M. Mulvenna, "A Low Cost Indoor Positioning System Using Bluetooth Low Energy," in *IEEE Access, vol. 8, pp.* 136858-136871, 2020, doi: 10.1109/ACCESS.2020.3012342.

[39] `https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-BLE-Bluetooth-Low-Energy.html/`

[40] `shorturl.at/hqA05`

[41] Omid Akbarzadeh, Mehrshid Baradaran, Mohammad R. Khosravi, "IoT-Based Smart Management of Healthcare Services in Hospital Buildings during COVID-19 and Future Pandemics", *Wireless Communications and Mobile Computing, vol. 2021,* Article ID 5533161, 14 pages, 2021. https://doi.org/10.1155/2021/5533161

[42] V. Renaudin et al., "Evaluating Indoor Positioning Systems in a Shopping Mall:

The Lessons Learned From the IPIN 2018 Competition," *in IEEE Access, vol. 7, pp.* 148594-148628, 2019, doi: 10.1109/ACCESS.2019.2944389.

[43] Ubisense,2018,https://ubisense.net/en/

[44] https://backlinko.com/telegram-users

[45] https://siteefy.com/telegram-statistics/

[46] https://www.geeksforgeeks.org/top-7-programming-languages-for-backend-web-dev

[47] https://codete.com/blog/wearable-fitness-technology-trends-and-statistics-202

[48] H. H. Liu and C. Liu, "Implementation of Wi-Fi signal sampling on an android smartphone for indoor positioning systems," *Sensors*, vol. 18, no. 1, p. 3, 2018.

[49] B. Maqsood and I. H. Naqvi, "Sub-nyquist rate UWB indoor positioning using power delay profile and time of arrival estimates," *in Proceedings of the Vehicular Technology Conference, Toronto, Canada*, September 2017.

[50] E. Hu, Z. L. Deng, and Y. Lu, "Efficient and robust convex relaxation methods for hybrid TOA/AOA indoor localization," *in Proceedings of the China Satellite Navigation Conference*, pp. 172–176, Shanghai, China, May 2017.

[51] H. Nawaz, A. Bozkurt, and I. Tekin, "A novel power efficient asynchronous time difference of arrival indoor localization system using CC1101 radio transceivers," *Microwave and Optical Technology Letters*, vol. 59, no. 3, pp. 550–555, 2017.

[52] A. M. Zungeru, J. Chuma, M. Mangwala, B. Sigweni, and O. Matsebe, "Signal propagation and analysis in wireless underground sensor networks," *International Journal of Engineering Research in Africa*, vol. 41, pp. 60–78, 2019.

[53] https://searchdatamanagement.techtarget.com/definition/MongoDB#:~:text=MongoDB%20is%20an%20open%20source,information%2C%20store%20or%20retrieve%20information./

[54] https://www.techtarget.com/searchapparchitecture/definition/RESTful-API/

[55] https://www.gps.gov/systems/gps/performance/accuracy/

[56] https://developer.here.com/blog/an-introduction-to-geojson

[57] https://www.prodyogi.com/2021/08/trilateration-surveying.html

[58] https://www.pcmag.com/encyclopedia/term/rssi

[59] Subedi, S., amp; Pyun, J.-Y. (2017, December 31). Practical fingerprinting localization for indoor positioning system by using beacons.*Journal of Sensors. Retrieved March 5, 2022, from https://www.hindawi.com/journals/js/2017/9742170/*

[60] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: a deep learning approach," *IEEE Transactions on Vehicular Technology, vol. 66, no. 1*, pp. 763–776, 2017.

[61] S. He and S. H. Gray Chan, "Wi-Fi fingerprint-based indoor positioning: recent advances and comparisons," *IEEE Communications Surveys and Tutorials, vol. 18, no. 1*, pp. 466–490, 2016.

[62] Obeidat, H., Shuaieb, W., Obeidat, O. et al. A Review of Indoor Localization Techniques and Wireless Technologies. *Wireless Pers Commun 119*, 289–327 (2021). https://doi.org/10.1007/s11277-021-08209-5

[63] https://lisbdnet.com/how-many-gps-satellites-are-needed-to-pinpoint-a-locatio#:~:text=In%20total%2C%20there%20are%20at,there%20are%2029%20operational%20satellites/

[64] https://www.directionsmag.com/article/1598

[65] Ozsoy, K., Bozkurt, A., amp; Tekin, I. (2013). Indoor positioning based on Global Positioning System Signals. *Microwave and Optical Technology Letters*, 55(5), 1091–1097. https://doi.org/10.1002/mop.27520

[66] https://www.wordstream.com/blog/ws/2018/10/04/beacon-technology#:~:text=The%20beacon%20device%20itself%20is,important%20place%20in%20your%20environment.

[67] https://source.android.com/devices/tech/connect/wifi-mac-randomization/

[68] https://asiatimes.com/2020/10/new-privacy-tech-has-pros-and-cons/

[69] https://www.infsoft.com/basics/positioning-technologies/ultra-wideband//

[70] https://www.mokoblue.com/what-is-beacon/

[71] https://www.geeksforgeeks.org/difference-between-mac-address-and-ip-address/

[72] https://raspberrytips.com/difference-raspberry-pi-computer/

[73] https://assetwolf.com/learn/mqtt-qos-understanding-quality-of-service#:~:text=Quality%20of%20Service%20(QoS)%20in,2%20%2D%20exactly%20once/

[74] https://db-engines.com/en/ranking/

[75] https://www.cherryroad.com/2021/10/22/oracle-database-cloud/

[76] https://www.techwalla.com/articles/advantages-disadvantages-of-oracle-sql#:~:text=Cost,times%20as%20high%20for%20Oracle./

[77] https://www.techstrikers.com/MySQL/advantages-and-disadvantages-of-mysql.php/

[78] https://acodez.in/mongodb-nosql-database/

[79] https://www.geeksforgeeks.org/python-pillow-a-fork-of-pil/

[80] https://timesofindia.indiatimes.com/most-searched-products/faq/how-to-use-colour-palettes/articleshow/83635674.cms?from=mdr/

# Appendix A

# Appendix

Add new patients to the hospitals Dashboard.

```
<?php

session_start();
$_SESSION;

include("Config.php");
include("functions.php");


$user_data = check_login($con);


if($_SERVER['REQUEST_METHOD'] == "POST")
{

    $first_name = $_POST['first_name'];
    $last_name = $_POST['last_name'];
    $dob = $_POST['DOB'];
    $mac = $_POST['MAC'];


    if(!empty($first_name) && !empty($last_name) && !empty($dob) && !empty($mac))
    {

        $user_id = uniqid(rand(), true);
        $query = "INSERT INTO user_data(user_id,first_name,last_name, DOB, MAC) VALUES('$user_id','$first_name','$last_name', '$dob','$mac')";

```

```
28          mysqli_query($con, $query);
29
30          $queryGPS = "INSERT INTO gps_data(user_id,MAC) VALUES('
    $user_id','$mac')";
31          mysqli_query($con, $queryGPS);
32
33          //header("Location: add-new-user.php");
34          //echo $query;
35
36          //die;
37
38      }
39      else
40      {
41
42          //echo "Errors";
43
44      }
45
46  }
47
48  ?>
49
50  <!DOCTYPE html>
51  <html lang="en">
52
53  <head>
54
55      <meta charset="utf-8">
56      <meta http-equiv="X-UA-Compatible" content="IE=edge">
57      <meta name="viewport" content="width=device-width, initial-scale
    =1, shrink-to-fit=no">
58      <meta name="description" content="">
59      <meta name="author" content="">
60
61      <title>Add New User</title>
62
63      <!-- Custom fonts for this template-->
64      <link href="vendor/fontawesome-free/css/all.min.css" rel="
    stylesheet" type="text/css">
65      <link
66          href="https://fonts.googleapis.com/css?family=Nunito:200,200i
    ,300,300i,400,400i,600,600i,700,700i,800,800i,900,900i"
67          rel="stylesheet">
68
69      <!-- Custom styles for this template-->
70      <link href="css/sb-admin-2.min.css" rel="stylesheet">
71
72  </head>
```

```
73
74 <body id="page-top">
75
76     <!-- Page Wrapper -->
77     <div id="wrapper">
78
79         <!-- Sidebar -->
80         <ul class="navbar-nav bg-gradient-primary sidebar sidebar-
    dark accordion" id="accordionSidebar">
81
82             <!-- Sidebar - Brand -->
83             <a class="sidebar-brand d-flex align-items-center justify
    -content-center" href="index.php">
84                 <div class="sidebar-brand-icon rotate-n-15">
85                     <i class="fas fa-laugh-wink"></i>
86                 </div>
87                 <div class="sidebar-brand-text mx-3">SB Admin <sup
    >2</sup></div>
88             </a>
89
90             <!-- Divider -->
91             <hr class="sidebar-divider my-0">
92
93             <!-- Nav Item - Dashboard -->
94             <li class="nav-item">
95                 <a class="nav-link" href="index.php">
96                     <i class="fas fa-fw fa-tachometer-alt"></i>
97                     <span>Dashboard</span></a>
98             </li>
99
100            <!-- Divider -->
101            <hr class="sidebar-divider">
102
103            <!-- Nav Item -->
104            <li class="nav-item active">
105                <a class="nav-link collapsed" href="add-new-user.php"
     data-target="#collapseTwo"
106                 aria-expanded="true" aria-controls="collapseTwo">
107                     <i class="fas fa-user-plus"></i>
108                     <span>Add New User</span>
109                 </a>
110            </li>
111
112            <!-- Nav Item -->
113            <li class="nav-item">
114                <a class="nav-link collapsed" href="delete-user.php"
    data-target="#collapseUtilities"
115                     aria-expanded="true" aria-controls="
    collapseUtilities">
```

```
116                     <i class="fas fa-user-minus"></i>
117                     <span>Delete User</span>
118                 </a>
119             </li>
120 
121             <!-- Nav Item -->
122             <li class="nav-item">
123                 <a class="nav-link collapsed" href="view-location.php
    " data-target="#collapseUtilities"
124                     aria-expanded="true" aria-controls="
    collapseUtilities">
125                     <i class="fas fa-search-location"></i>
126                     <span>View Location</span>
127                 </a>
128             </li>
129 
130             <!-- Divider -->
131             <hr class="sidebar-divider d-none d-md-block">
132 
133             <!-- Sidebar Toggler (Sidebar) -->
134             <div class="text-center d-none d-md-inline">
135                 <button class="rounded-circle border-0" id="
    sidebarToggle"></button>
136             </div>
137 
138         </ul>
139         <!-- End of Sidebar -->
140 
141         <!-- Content Wrapper -->
142         <div id="content-wrapper" class="d-flex flex-column">
143 
144             <!-- Main Content -->
145             <div id="content">
146 
147                 <!-- Topbar -->
148                 <nav class="navbar navbar-expand navbar-light bg-
    white topbar mb-4 static-top shadow">
149 
150                     <!-- Sidebar Toggle (Topbar) -->
151                     <form class="form-inline">
152                         <button id="sidebarToggleTop" class="btn btn-
    link d-md-none rounded-circle mr-3">
153                             <i class="fa fa-bars"></i>
154                         </button>
155                     </form>
156 
157                     <!-- Topbar Navbar -->
158                     <ul class="navbar-nav ml-auto">
159 
```

```
160                         <!-- Nav Item - User Information -->
161                         <li class="nav-item dropdown no-arrow">
162                             <a class="nav-link dropdown-toggle" href
    ="#" id="userDropdown" role="button"
163                                 data-toggle="dropdown" aria-haspopup
    ="true" aria-expanded="false">
164                                 <span class="mr-2 d-none d-lg-inline
    text-gray-600 small"><?php echo $user_data['user_name']; ?></span>
165                                 <img class="img-profile rounded-
    circle"
166                                     src="img/undraw_profile.svg">
167                             </a>
168                             <!-- Dropdown - User Information -->
169                             <div class="dropdown-menu dropdown-menu-
    right shadow animated--grow-in"
170                                 aria-labelledby="userDropdown">
171
172                                 <a class="dropdown-item" href="#"
    data-toggle="modal" data-target="#logoutModal">
173                                     <i class="fas fa-sign-out-alt fa-
    sm fa-fw mr-2 text-gray-400"></i>
174                                     Logout
175                                 </a>
176                             </div>
177                         </li>
178
179                     </ul>
180
181                 </nav>
182                 <!-- End of Topbar -->
183
184                 <!-- Begin Page Content -->
185                 <div class="container-fluid">
186
187                     <!-- Page Heading -->
188                     <div class="d-sm-flex align-items-center justify-
    content-center mb-4">
189                         <h1 class="h3 mb-0 text-gray-800">Add New
    User</h1>
190                     </div>
191
192                     <!-- Content Row -->
193                     <div class="row w-100 d-flex align-items-center">
194                         <form class="w-100 px-4" name = "newAdd"
    method = "post">
195                             <div class="form-group row">
196                               <label for="FirstName" class="col-sm-2
    col-form-label">First Name</label>
197                                 <div class="col-sm-10">
```

```
198                                 <input type="text" class="form-
    control" id="FirstName" name = "first_name" placeholder="Name"
    required>
199                               </div>
200                             </div>
201                             <div class="form-group row">
202                               <label for="LastName" class="col-sm-2
    col-form-label">Last Name</label>
203                               <div class="col-sm-10">
204                                 <input type="text" class="form-
    control" id="LastName" name = "last_name" placeholder="Last Name"
    required>
205                               </div>
206                             </div>
207                             <div class="form-group row">
208                                 <label for="DateOfBirth" class="col-
    sm-2 col-form-label">Date of Birth</label>
209                                 <div class="col-sm-10">
210                                   <input type="date" class="form-
    control" name = "DOB" id="DateOfBirth" required>
211                                 </div>
212                             </div>
213                             <div class="form-group row">
214                                 <label for="MacAddress" class="col-sm
    -2 col-form-label">MAC ADDRESS</label>
215                                 <div class="col-sm-10">
216                                   <input type="text" class="form-
    control" id="MacAddress" name = "MAC" placeholder="aabbccddeeff"
    required>
217                                 </div>
218                               </div>
219                             <div class="form-group row d-flex justify
    -content-start">
220                               <div class="col">
221                                 <button type="submit" class="btn btn-
    success">Submit</button>
222                                 <button type="reset" class="btn btn-
    secondary">Reset</button>
223                               </div>
224                             </div>
225                           </form>
226                 </div>
227
228             </div>
229             <!-- /.container-fluid  name = "newAdd" method = "
    post" onsubmit="return validateform()" -->
230
231         </div>
232         <!-- End of Main Content -->
```

```
233
234             <!-- Footer -->
235             <footer class="sticky-footer bg-white">
236                 <div class="container my-auto">
237                     <div class="copyright text-center my-auto">
238                         <span>Copyright</span>
239                     </div>
240                 </div>
241             </footer>
242             <!-- End of Footer -->
243
244         </div>
245         <!-- End of Content Wrapper -->
246
247     </div>
248     <!-- End of Page Wrapper -->
249
250     <!-- Scroll to Top Button-->
251     <a class="scroll-to-top rounded" href="#page-top">
252         <i class="fas fa-angle-up"></i>
253     </a>
254
255     <!-- Logout Modal-->
256     <div class="modal fade" id="logoutModal" tabindex="-1" role="
    dialog" aria-labelledby="exampleModalLabel"
257         aria-hidden="true">
258         <div class="modal-dialog" role="document">
259             <div class="modal-content">
260                 <div class="modal-header">
261                     <h5 class="modal-title" id="exampleModalLabel">
    Ready to Leave?</h5>
262                     <button class="close" type="button" data-dismiss
    ="modal" aria-label="Close">
263                         <span aria-hidden="true">  </span>
264                     </button>
265                 </div>
266                 <div class="modal-body">Select "Logout" below if you
    are ready to end your current session.</div>
267                 <div class="modal-footer">
268                     <button class="btn btn-secondary" type="button"
    data-dismiss="modal">Cancel</button>
269                     <a class="btn btn-primary" href="login.php">
    Logout</a>
270                 </div>
271             </div>
272         </div>
273     </div>
274
275     <!-- Bootstrap core JavaScript-->
```

```
276     <script src="vendor/jquery/jquery.min.js"></script>
277     <script src="vendor/bootstrap/js/bootstrap.bundle.min.js"></
        script>
278
279     <!-- Core plugin JavaScript-->
280     <script src="vendor/jquery-easing/jquery.easing.min.js"></script>
281
282     <!-- Custom scripts for all pages-->
283     <script src="js/sb-admin-2.min.js"></script>
284
285
286 </body>
287
288 </html>
```

The registration page is used to add hospital staff who will have access to the IPS and will be able to see the location of patients.

```
1  <?php
2
3  session_start();
4  $_SESSION;
5
6  include("Config.php");
7  include("functions.php");
8
9
10
11 if($_SERVER['REQUEST_METHOD'] == "POST")
12 {
13
14     $first_name = $_POST['first_name'];
15     $last_name = $_POST['last_name'];
16     $user_name = $_POST['username'];
17     $email = $_POST['email'];
18     $password = $_POST['password'];
19     $rpassword = $_POST['rpassword'];
20
21
22
23     if(!empty($first_name) && !empty($last_name) && !empty($email) &&
       !empty($password)&& !empty($rpassword))
24     {
25
26         $user_id = uniqid(rand(), true);
27         $query = "INSERT INTO ulogtab(user_id,user_name,password)
       VALUES('$user_id','$user_name','$password')";
28
```

```
29          mysqli_query($con, $query);
30
31          header("Location: login.php");
32          //echo "Inserted";
33          die;
34
35      }
36      else
37      {
38
39          //echo "Kindly Fill Out All Fields";
40
41      }
42
43  }
44
45
46
47  ?>
48
49  <!DOCTYPE html>
50  <html lang="en">
51
52  <head>
53
54      <meta charset="utf-8">
55      <meta http-equiv="X-UA-Compatible" content="IE=edge">
56      <meta name="viewport" content="width=device-width, initial-scale
        =1, shrink-to-fit=no">
57      <meta name="description" content="">
58      <meta name="author" content="">
59
60      <title>Register</title>
61
62      <!-- Custom fonts for this template-->
63      <link href="vendor/fontawesome-free/css/all.min.css" rel="
        stylesheet" type="text/css">
64      <link
65          href="https://fonts.googleapis.com/css?family=Nunito:200,200i
        ,300,300i,400,400i,600,600i,700,700i,800,800i,900,900i"
66          rel="stylesheet">
67
68      <!-- Custom styles for this template-->
69      <link href="css/sb-admin-2.min.css" rel="stylesheet">
70
71  </head>
72
73  <body class="bg-gradient-primary min-vh-100 d-flex align-items-center
        ">
```

```
74
75      <div class="container">
76
77          <div class="card o-hidden border-0 shadow-lg my-5">
78              <div class="card-body p-0">
79                  <!-- Nested Row within Card Body -->
80                  <div class="row">
81                      <div class="col-lg-5 d-none d-lg-block bg-
    register-image"></div>
82                      <div class="col-lg-7">
83                          <div class="p-5">
84                              <div class="text-center">
85                                  <h1 class="h4 text-gray-900 mb-4">
    Create an Account!</h1>
86                              </div>
87                              <form name = "signupform" class="user"
    method = "post" onsubmit="return validateform()">
88                                  <div class="form-group row">
89                                      <div class="col-sm-6 mb-3 mb-sm
    -0">
90                                          <input type="text" class="
    form-control form-control-user" id="exampleFirstName"
91                                              placeholder="First Name"
    name="first_name">
92                                      </div>
93                                      <div class="col-sm-6">
94                                          <input type="text" class="
    form-control form-control-user" id="exampleLastName"
95                                              placeholder="Last Name"
    name="last_name">
96                                      </div>
97                                  </div>
98                                  <div class="form-group">
99                                          <input type="text" class="
    form-control form-control-user" id="exampleFirstName"
100                                             placeholder="User Name"
    name="username">
101                                 </div>
102                                 <div class="form-group">
103                                     <input type="email" class="form-
    control form-control-user" id="exampleInputEmail"
104                                         placeholder="Email Address"
    name="email">
105                                 </div>
106                                 <div class="form-group row">
107                                     <div class="col-sm-6 mb-3 mb-sm
    -0">
108                                         <input type="password" class
    ="form-control form-control-user"
```

```
109                                             id="exampleInputPassword"
     placeholder="Password" name="password">
110                                     </div>
111                                     <div class="col-sm-6">
112                                         <input type="password" class
    ="form-control form-control-user"
113                                             id="exampleRepeatPassword
    " placeholder="Repeat Password" name="rpassword">
114                                     </div>
115                                 </div>
116                                 <input type = "submit" value = "
    Register Account"
117                                     class="btn btn-primary btn-user
    btn-block"/>
118
119                                 <!-- <a href="login.php" class="btn
    btn-primary btn-user btn-block">
120                                     Register Account
121                                 </a> -->
122
123                             </form>
124                             <br>
125                             <div class="text-center">
126                                 <a class="small" href="forgot-
    password.html">Forgot Password?</a>
127                             </div>
128                             <div class="text-center">
129                                 <a class="small" href="login.php">
    Already have an account? Login!</a>
130                             </div>
131                         </div>
132                     </div>
133                 </div>
134             </div>
135         </div>
136
137     </div>
138
139     <!-- Bootstrap core JavaScript-->
140     <script src="vendor/jquery/jquery.min.js"></script>
141     <script src="vendor/bootstrap/js/bootstrap.bundle.min.js"></
    script>
142
143     <!-- Core plugin JavaScript-->
144     <script src="vendor/jquery-easing/jquery.easing.min.js"></script>
145
146     <!-- Custom scripts for all pages-->
147     <script src="js/sb-admin-2.min.js"></script>
148
```

```
149
150     <script>
151         function validateform()
152         {
153             var first_name=document.signupform.first_name.value;
154             var last_name=document.signupform.last_name.value;
155             var user_name=document.signupform.username.value;
156
157             var password=document.signupform.password.value;
158
159             var firstpassword=document.signupform.password.value;
160             var secondpassword=document.signupform.rpassword.value;
161
162                 if (first_name==null || first_name=="")
163                 {
164                     alert("First Name can't be blank");
165                     return false;
166                 }
167                 else if (last_name==null || last_name=="")
168                 {
169                     alert("Last Name can't be blank");
170                     return false;
171                 }
172                 else if (username==null || username=="")
173                 {
174                     alert("User Name can't be blank");
175                     return false;
176                 }
177                 else if(password.length<6)
178                 {
179                     alert("Password must be at least 6 characters
    long.");
180                     return false;
181                 }
182
183                 if(firstpassword==secondpassword)
184                 {
185                     return true;
186                 }
187                 else
188                 {
189                     alert("password must be same!");
190                     return false;
191                 }
192         }
193     </script>
194
195 </body>
196
```

```
197
198 </html>
```

Station code

```
1  from bluepy.btle import Scanner, DefaultDelegate
2  import paho.mqtt.client as mqtt
3  import paho.mqtt.publish as publisher
4  import time
5  import json
6  import sys
7  from threading import Thread, Lock
8
9  stationId       = ""
10 devicesArray    = [""]
11 brokerIP        = "mqtt.eclipseprojects.io"
12 pubTopic        = ""
13 subTopic        = ""
14 scanInterval    = "0.5"
15 configFileName  = "station"
16 devLock         = None
17
18 class ScanDelegate(DefaultDelegate):
19
20     def __init__(self, sender):
21         DefaultDelegate.__init__(self)
22         self.__sender = sender
23
24     def handleDiscovery(self, dev, isNewDev, isNewData):
25         devLock.acquire(True)
26
27         for e in devicesArray:
28             if e == dev.addr:
29                 self.__sender.addMeasurement(e, dev.rssi)
30
31         devLock.release()
32
33
34 class Sender(Thread):
35     def __init__(self, time):
36         """ Constructor """
37         Thread.__init__(self)
38         self.__time = time
39         self.__map = {}
40
41     def run(self):
42         while(True):
43             time.sleep(self.__time)
```

```
44              payload = self.__buildPayload(self.__map)
45              print (payload)
46              publisher.single(pubTopic, payload, hostname=brokerIP)
47              self.__map = {}
48
49
50      def addMeasurement(self, name, rssi):
51          if not self.__map.has_key(name):
52              self.__map[name] = []
53
54          self.__map[name].append(int(rssi))
55
56
57      def __buildPayload(self, map):
58          payload = {}
59          payload["station-id"] = str(stationId)
60          payload["map"] = map
61
62          return json.dumps(payload)
63
64
65  def on_message(client, userdata, message):
66      jsonMsg = json.loads(message.payload.decode("utf-8"))
67      action = jsonMsg["action"]
68      userMac = jsonMsg["mac"]
69      devLock.acquire(True)
70      if action == "delete":
71          for dev in devicesArray:
72              if dev == userMac:
73                  devicesArray.remove(userMac)
74                  break
75          dumpToFile()
76      elif action == "add":
77          for m in jsonMsg["mac"] :
78              if not m in devicesArray :
79                  devicesArray.append(m)
80          dumpToFile()
81
82      else :
83          print("Invalid code")
84
85      devLock.release()
86  def on_connect(client, userdata, flags, rc):
87      print('connected')
88
89
90  def dumpToFile():
91      data = dict()
92      data["id"] = stationId
```

```
93      data["devices"] = devicesArray
94      data["broker_ip"] = brokerIP
95      data["publish_topic"] = pubTopic
96      data["subscribe_topic"] = subTopic
97      data["scan_interval"] = scanInterval
98      data["send_interval"] = sendInterval
99
100     with open(configFileName, 'w') as outfile:
101         json.dump(data, outfile, indent=4)
102
103 def main():
104     scanner = None
105     json_data = None
106
107     if len(sys.argv) != 2:
108         sys.exit("Wrong number of arguments")
109
110     print ("Initializing station")
111     json_data = json.load(open(sys.argv[1]))
112
113     global stationId
114     global devicesArray
115     global brokerIP
116     global pubTopic
117     global subTopic
118     global scanInterval
119     global sendInterval
120
121     global configFileName
122
123     global devLock
124
125     stationId    = json_data["id"]
126     devicesArray = json_data["devices"]
127     brokerIP     = json_data["broker_ip"]
128     pubTopic     = json_data["publish_topic"]
129     subTopic     = json_data["subscribe_topic"]
130     scanInterval = float(json_data["scan_interval"])
131     sendInterval = float(json_data["send_interval"])
132
133     configFileName = "../Thesis/Information.json"
134
135     client = mqtt.Client(stationId)
136     client.connect(brokerIP)
137     client.subscribe(subTopic, qos=2)
138     client.on_message=on_message
139     client.loop_start()
140
141     rssiSender = Sender(sendInterval)
```

```
142     rssiSender.daemon = True
143
144     scanner = Scanner().withDelegate(ScanDelegate(rssiSender))
145     rssiSender.start()
146
147     while (True):
148         devices = scanner.scan(scanInterval)
149
150
151 if __name__ == "__main__":
152     main()
```

Server code

```
1
2 import paho.mqtt.client as mqtt
3 import paho.mqtt.publish as publisher
4 import time
5 import json
6 import numpy as np
7 import sys
8 import copy
9 import pymongo
10 import pymongo.collection
11 import signal
12 from db_interface import DBInterface
13
14 beaconTable = {}
15 configFileContent = {}
16 beaconTableLocker = None
17 configFCLocker = None
18 configFileName = ""
19 pubTopic= ""
20 webApp = Flask(__name__)
21 database = None
22
23 def  storeConfigurationFile () :
24     print ("Saving configuration to " + configFileName)
25     f= open(configFileName, 'w')
26     json.dump(configFileContent, f, indent=4)
27     f.close()
28
29 def roomsToArray () :
30
31     arr= []
32     for p in configFileContent["positions"] :
33         arr.append (configFileContent["positions"][p])
34     return arr
```

```
35
36 def roomNameToId (rn) :
37     rid= ""
38     if rn != "" :
39         for p in configFileContent["positions"] :
40             if (configFileContent["positions"][p] == rn) :
41                 rid = p
42                 break
43     return rid
44
45 class BeaconInfo():
46
47     def __init__(self, id, sids):
48         self.__map = dict()
49         self.__id = id
50         self.__last = ""
51     def getMap(self):
52         return self.__map
53     def getId(self):
54         return self.__id
55     def getLast(self):
56         return self.__last
57     def setLast(self, last):
58         self.__last = last
59     def addMeasure(self, sid, measure):
60
61         if not self.__map.has_key(str(sid)):
62             self.__map[str(sid)] = []
63
64         self.__map[str(sid)].extend(measure)
65     def cleanInfo(self):
66         for e in self.__map:
67             self.__map[e] = []
68
69 class WebServer(Thread):
70     def __init__(self, app, ip, port):
71         Thread.__init__(self)
72         self.__flaskApp= app
73         self.__port= port
74         self.__ip = ip
75     def run(self):
76         self.__flaskApp.run(self.__ip, self.__port)
77
78 @webApp.route('/')
79 def root():
80     return Response(REST API!", status=200, content_type="text/plain
    ")
81
82 @webApp.route("/rooms", methods=["GET"])
```

```
83 def roomsGet():
84     return Response(json.dumps(roomsToArray()), status=200,
       content_type="application/json")
85
86 @webApp.route("/rooms/<rn>", methods=['GET'])
87 def getRooms (rn):
88     rooms= roomsToArray()
89     rid = roomNameToId(rn)
90
91     if rn == "" :
92         return Response("Room is empty", status=400, content_type="
       text/plain")
93     elif not rn in rooms :
94         return Response("Requested room doesn't exists", status=400,
       content_type="text/plain")
95     else :
96         ls = []
97         beaconTableLocker.acquire (True)
98         for b in beaconTable :
99             if beaconTable[b].getLast() == rid :
100                ls.append(beaconTable[b].getId())
101        beaconTableLocker.release()
102        return Response (json.dumps(ls), status=200, content_type="
       application/json")
103
104 @webApp.route("/rooms/<rn>", methods=['POST'])
105 def postRooms(rn):
106     toRet= None
107     configFCLocker.acquire(True)
108
109     rooms= configFileContent["positions"].values()
110
111     if rn == "" :
112         toRet= Response("Room is empty!", status=400, content_type="
       text/plain")
113     elif request.data == "" :
114         toRet = Response("Invalid raspberry mac", status=400,
       content_type="text/plain")
115     elif rn in rooms :
116         toRet= Response("Requested room already exists!", status=400,
       content_type="text/plain")
117     elif request.data in configFileContent["positions"]:
118         toRet = Response("Station id already associated!", status
       =400, content_type="text/plain")
119     else :
120         entry= {request.data : rn}
121         print ("Creating room  " + str(entry))
122         configFileContent["positions"][request.data]= rn
123         toSend= []
```

```
124         for k in configFileContent["devices"] :
125             toSend.append(k)
126         print ("Sending " + str(toSend))
127         payload = {
128             "action" : "add",
129             "mac" : toSend
130         }
131         brokerAddress = configFileContent["broker-ip"]
132         publisher.single (pubTopic, json.dumps(payload), hostname=
    brokerAddress)
133 
134         storeConfigurationFile ()
135         toRet= Response("", status=201, content_type="text/plain")
136     configFCLocker.release()
137     return toRet
138 
139 @webApp.route("/rooms/<rn>", methods=['DELETE'])
140 def deleteRooms(rn):
141     toRet= None
142     configFCLocker.acquire(True)
143     rooms= configFileContent["positions"].values()
144     if rn == "" :
145         toRet= Response("Room name is empty!", status=400,
    content_type="text/plain")
146     elif not rn in rooms :
147         toRet= Response("Room name  " + rn + "  doesn't exist!",
    status=400, content_type="text/plain")
148     else :
149         rid = roomNameToId (rn)
150         beaconTableLocker.acquire(True)
151 
152         del configFileContent["positions"][rid]
153         storeConfigurationFile ()
154 
155         for b in beaconTable :
156             bo = beaconTable[b]
157             if bo.getLast() == rid :
158                 bo.setLast("")
159         beaconTableLocker.release ()
160         database.delete_room_entries(rn)
161         toRet= Response("", status=200, content_type="text/plain")
162     configFCLocker.release()
163 
164     return toRet
165 
166 @webApp.route("/readings/<bid>", methods=['GET'])
167 def getReadings(bid):
168     if bid == "" :
```

```
169         return Response("Beacon id is empty!", status=400,
    content_type="text/plain")
170     elif not beaconTable.has_key(bid) :
171         return Response("Beacon id  " + bid + "  doesn't exist!",
    status=400, content_type="text/plain")
172     else :
173         beaconTableLocker.acquire(True)
174         res= json.dumps(beaconTable[bid].getMap())
175         beaconTableLocker.release ()
176         return Response(res, status=200, content_type="application/
    javascript")
177 @webApp.route("/readings/<bid>", methods=['DELETE'])
178 def deleteReadings(bid):
179     if bid == "" :
180         return Response("Beacon id is empty!", status=400,
    content_type="text/plain")
181     elif not beaconTable.has_key(bid) :
182         return Response("Beacon id  " + bid + "  doesn't exist!",
    status=400, content_type="text/plain")
183     else :
184         beaconTableLocker.acquire(True)
185         beaconTable[bid].cleanInfo()
186         beaconTableLocker.release ()
187         return Response("", status=200, content_type="text/plain")
188
189 @webApp.route("/userList", methods=["GET"])
190 def getuserList () :
191
192     user= []
193     configFCLocker.acquire (True)
194     for d in configFileContent["devices"] :
195         p= configFileContent["devices"][d]
196         user.append(p)
197     configFCLocker.release ()
198     return Response(json.dumps(user), status=200, content_type="
    application/json")
199
200 @webApp.route("/user", methods=["GET"])
201 def getuserLocations():
202
203     user = dict()
204     beaconTableLocker.acquire (True)
205     for b in beaconTable:
206         if not beaconTable[b].getLast() == "" :
207             rid = beaconTable[b].getLast()
208             user[b] = str(configFileContent["positions"][rid])
209     beaconTableLocker.release()
210     return Response(json.dumps(user), status=200, content_type="
    application/json")
```

```
211
212 @webApp.route("/user/<pid>", methods=["POST"])
213 def postuser(pid):
214     toRet= None
215     configFCLocker.acquire(True)
216     beaconTableLocker.acquire(True)
217
218     if pid == "" :
219         toRet= Response("Beacon id is empty!", status=400,
    content_type="text/plain")
220     elif beaconTable.has_key (pid) :
221         toRet= Response("Beacon with id  " + pid + "  already exists
    !", status=400, content_type="text/plain")
222     elif request.data in configFileContent["devices"]:
223         toRet = Response("Mac address  " + request.data + "  already
    in use!", status=400, content_type="text/plain")
224     else :
225         rs= roomsToArray()
226
227         beaconTable[pid]= BeaconInfo(pid, rs)
228         configFileContent["devices"].update({request.data:pid})
229         storeConfigurationFile()
230
231         payload = {
232             "action" : "add",
233             "mac" : request.data
234         }
235         brokerAddress = configFileContent["broker-ip"]
236         publisher.single (pubTopic, json.dumps(payload), hostname=
    brokerAddress)
237
238         toRet= Response('', status=201, content_type="text/plain")
239     beaconTableLocker.release()
240     configFCLocker.release()
241     return toRet
242
243 @webApp.route("/user/<pid>", methods=["DELETE"])
244 def deleteuser(pid):
245     toRet= None
246     configFCLocker.acquire(True)
247     beaconTableLocker.acquire(True)
248
249     if pid == "" :
250         toRet= Response("Beacon id is empty!", status=400,
    content_type="text/plain")
251     elif not beaconTable.has_key (pid) :
252         toRet= Response("Beacon with id  " + pid + "  doesn't exist
    !", status=400, content_type="text/plain")
253     else :
```

```
254         beaconTable.pop(pid)
255         database.delete_device_entries(pid)
256         mac_address = ""
257         for mac, name in configFileContent["devices"].items():
258             if name ==pid:
259                 mac_address = mac
260 
261         print("Deleting "+mac_address+" association with user "+pid)
262         configFileContent["devices"].pop(mac_address)
263         storeConfigurationFile()
264 
265         payload = {
266             "action" : "delete",
267             "mac" : mac_address,
268         }
269         brokerAddress = configFileContent["broker-ip"]
270         publisher.single (pubTopic, json.dumps(payload), hostname=
    brokerAddress)
271 
272         toRet= Response('', status=200, content_type="application/
    json")
273     beaconTableLocker.release()
274     configFCLocker.release()
275     return toRet
276 
277 
278 def on_message(client, userdata, message):
279     """ Structure of received message
280         {
281             "stid" : value
282             {
283                 "mac_address1" : [rssi_1, ..., rssi_n]
284                 "mac_address2" : [rssi_1, ..., rssi_n]
285                 "mac_address3" : [rssi_1, ..., rssi_n]
286                 "mac_address4" : [rssi_1, ..., rssi_n]
287             }
288         }
289     """
290     jsonMsg = json.loads(message.payload.decode("utf-8"))
291     configFCLocker.acquire (True)
292     beaconTableLocker.acquire(True)
293 
294     found = False
295     for p in configFileContent["positions"] :
296         if p == jsonMsg["station-id"] :
297             found= True
298 
299     if not found :
```

```
300         print ("Station id  " + jsonMsg["station-id"] + "  doesn't
    correspond to any registered room!")
301
302     else :
303         for mac in jsonMsg["map"] :
304             try:
305                 user = configFileContent["devices"][mac]
306                 if beaconTable.has_key(user) :
307                     beaconTable[user].addMeasure(jsonMsg["station-id
    "],  jsonMsg["map"][mac])
308
309             except(KeyError):
310                 print("Removed user " + mac)
311
312     beaconTableLocker.release()
313     configFCLocker.release ()
314
315
316
317 def main():
318     global beaconTable
319     global beaconTableLocker
320     global database
321     global configFileContent
322     global configFCLocker
323     global configFileName
324     global pubTopic
325     print ("Initializing server")
326     configFileName= "../server.json"
327     configFileContent = json.load(open(configFileName))
328     beaconTable = dict()
329     beaconTableLocker = Lock()
330     configFCLocker= Lock()
331     tmpIds= []
332     for p in configFileContent["positions"] :
333         tmpIds.append (p)
334     for b in configFileContent["devices"].values() :
335         beaconTable[b]= BeaconInfo(b, tmpIds)
336
337     broker_address = configFileContent["broker-ip"]
338     subTopic = configFileContent["subscribe_topic"]
339     pubTopic = configFileContent["publish_topic"]
340
341     print ("Init broker")
342
343     client = mqtt.Client("P1")
344     client.connect(broker_address)
345     print ("Subscription to " + broker_address + " on topic " +
    subTopic)
```

```
346     client.subscribe(subTopic)
347     client.on_message=on_message
348     client.loop_start()
349 
350     database = DBInterface(configFileContent["DB_connection_params"])
351 
352     triangulate = Triangulate(int(configFileContent["algorithm-
        interval"]), beaconTable, beaconTableLocker, database)
353     triangulate.daemon = True
354     triangulate.start()
355 
356     webServer= WebServer(webApp, configFileContent["server-ip"], int(
        configFileContent["server-port"]))
357     webServer.daemon = True
358     webServer.start()
359 
360 
361     print ("Starting loop")
362 
363     while True:
364         time.sleep(1)
365 
366 
367 
368 
369 if __name__ == "__main__":
370     main()
```

```
1 import telegram
2 from telegram.ext import Updater, CommandHandler, MessageHandler,
      Filters
3 from telegram import InlineQueryResultArticle,
      InputTextMessageContent
4 from telegram.ext import InlineQueryHandler
5 import logging, sys, json
6 import requests
7 from pyzbar.pyzbar import decode
8 from PIL import Image
9 import re
10 from qrtools import QR
11 from qrtools import qrtools
12 from PIL import Image
13 import zbarlight
14 from pyzbar.pyzbar import decode, ZBarSymbol
15 import re
16 
17 
```

```
18 logging.basicConfig(format='%(asctime)s - %(name)s - %(levelname)s -
       %(message)s',
19                     level=logging.INFO)
20 logger = logging.getLogger(__name__)
21
22 ip_address = ""
23 OKGET = 200
24 OKPOST = 201
25 OKDELETE = 200
26
27
28 mac_pattern = "^([0-9A-Fa-f]{2}[:-]){5}([0-9A-Fa-f]{2})$"
29 station_pattern = "^([0-9A-Fa-f]{6})$"
30
31
32
33 def help(bot, update, chat_data):
34
35     startText = 'Welcome to Hospital indoor positioning assist system
       \n' \
36                 '\n\n'\
37                 'Commands:\n' \
38                 '/roomlist provides the list of rooms determined by
       the photo sent;\n' \
39                 '/patientslist to see the location of the patients;\n
       ' \
40                 '/whereareall gives the position of all patients;\n'
       \
41                 '/whoisintheroom shows patients in a specific room;\n
       ' \
42                 '\n\n'\
43                 'To add a new patient, send me a picture of the QR
       code with the new patients name.\n'\
44                 'To add a new station, send me a picture of the QR
       code with the rooms name;\n'\
45                 '\n\n'\
46                 '/deletepatient <patient> to remove a patient from
       the system;\n' \
47                 '/deleteroom <room> to remove a room from the system
       ;\n' \
48                 '\n\n'\
49
50     update.message.reply_text(startText)
51
52
53 def getUserList(bot, update):
54     try:
55         req = requests.get('http://'+ip_address+':8080/userList')
56
```

```
57          if(req.status_code == OKGET):
58              txt = ""
59              msg = req.json()
60              for b in msg:
61                  txt += str(b) + "\n"
62
63              update.message.reply_text(txt)
64          else :
65              update.message.reply_text("Connection error")
66
67      except (IndexError, ValueError):
68          update.message.reply_text('Use /userlist')
69
70  def getUser(bot, update, args, chat_data):
71      try:
72          user = args[0]
73          req = requests.get('http://'+ip_address+':8080/user')
74
75          if(req.status_code == OKGET):
76              txt = user + " not at home"
77              msg = req.json()
78              if user in msg.keys():
79                  txt = str(user) + " in " + str(msg[user])
80
81              update.message.reply_text(txt)
82          else :
83              update.message.reply_text("Connection error")
84
85      except (IndexError, ValueError):
86          update.message.reply_text('Use /whereis <user>')
87
88  def getUsers(bot, update):
89      """
90          Get all user locations
91      """
92
93      try:
94          req = requests.get('http://'+ip_address+':8080/user')
95
96          if(req.status_code == OKGET):
97              txt = ""
98              msg = req.json()
99              for b in msg:
100                 txt += str(b) + " in " + str(msg[b]) + "\n"
101
102             update.message.reply_text(txt)
103         else :
104             update.message.reply_text("Connection error")
105
```

```
106     except (IndexError, ValueError):
107         update.message.reply_text('Use /whereareall')
108 
109 def getRoomList(bot, update):
110     try:
111         r = requests.get('http://'+ip_address+':8080/rooms')
112 
113         if r.status_code == OKGET:
114             msg = r.json()
115             txt = ""
116             for room in msg:
117                 txt += str(room) + "\n"
118 
119             if txt == "":
120                 txt = "No registered rooms in your service"
121 
122             update.message.reply_text(txt)
123         else :
124             update.message.reply_text("Connection error")
125 
126     except (IndexError, ValueError):
127         update.message.reply_text('Use /roomlist')
128 
129 def getRoom(bot, update, args, chat_data):
130     try:
131         room = args[0]
132         r = requests.get('http://'+ip_address+':8080/rooms/'+room)
133 
134         if r.status_code == OKGET:
135 
136             msg = r.json()
137             txt = ""
138             for user in msg:
139                 txt += str(user) + "\n"
140 
141             if txt == "":
142                 txt = "No one in " + room
143 
144             update.message.reply_text(txt)
145 
146         elif r.content == "Room is empty":
147             update.message.reply_text("You didn't specify the room!")
148 
149         elif r.content == "Requested room doesn't exists":
150             update.message.reply_text("Requested room doesn't exists
    !\nTry again!")
151 
152         else :
153             update.message.reply_text("Connection error")
```

```
154
155     except (IndexError, ValueError):
156         update.message.reply_text('Use /whoisin <room>')
157
158 def add(bot, update):
159     """ Add a new user or a new room. """
160
161     try:
162         if update.message.photo is None:
163             update.message.reply_text('no foto')
164         elif update.message.caption is None:
165             update.message.reply_text("Missing caption")
166         else:
167             img_id = update.message.photo[-1].file_id
168             newFile = bot.get_file(img_id)
169             newFile.download('qrcode.png')
170
171             text = decode(Image.open("qrcode.png"))
172
173             if len(text) == 0:
174                 update.message.reply_text("No QR code found!")
175             else:
176                 name = update.message.caption
177                 data = text[-1].data
178
179                 if re.match(mac_pattern, data):
180                     r = requests.post('http://'+ip_address+':8080/
    user/'+name, data=data)
181                     if r.status_code == OKPOST:
182                         update.message.reply_text("User "+name+"
    associated to Mac Address "+data)
183                     elif r.content=="Beacon with id  " + name + "
    already exists!":
184                         update.message.reply_text("User "+name+" is
    already associated with a device!\nTry again!")
185                     elif r.content=="Mac address  " + data + "
    already in use!":
186                         update.message.reply_text("This device is
    already associated to an user!")
187                     else :
188                         update.message.reply_text("Connection error:
    "+ r.content)
189                 elif re.match(station_pattern, data):
190                     r = requests.post('http://'+ip_address+':8080/
    rooms/'+name, data=data)
191                     if r.status_code == OKPOST:
192                         update.message.reply_text("Room: "+name+"
    associated to station "+data)
193                     elif r.content=="Requested room already exists!":
```

```
194                         update.message.reply_text("Room "+name+"
    already exist!")
195                     elif r.content=="Station id already associated!":
196                         update.message.reply_text("This station is
    already associated to a room!")
197                     else:
198                         update.message.reply_text("Connection error:
    "+r.content)
199                 else:
200                     update.message.reply_text("QR code format not
    supported.")
201
202
203
204     except (IndexError, ValueError):
205         update.message.reply_text('Inserire messaggio di errore')
206
207
208 def deleteUser(bot, update, args):
209     try:
210         user = args[0]
211
212         r = requests.delete('http://'+ip_address+':8080/user/'+user)
213
214         if r.status_code == OKDELETE:
215             update.message.reply_text("Removed " + user)
216         elif r.content=="Beacon id is empty!":
217             update.message.reply_text("Please specify the user you
    want to delete.")
218         elif r.content=="Beacon with id  " + user + "  doesn't exist
    !":
219             update.message.reply_text("This user is not associated to
     any device.")
220         else :
221             update.message.reply_text("Connection error")
222
223     except (IndexError, ValueError):
224         update.message.reply_text('Use /deleteuser <user>')
225
226 def deleteRoom(bot, update, args):
227
228     try:
229         room = args[0]
230
231         r = requests.delete('http://'+ip_address+':8080/rooms/'+room)
232
233         if r.status_code == OKDELETE:
234             update.message.reply_text("Removed " + room)
235         elif r.content=="Room name is empty!":
```

```
236             update.message.reply_text("You didn't specify the name of
    the room you want to delete.")
237         elif r.content=="Room name  " + room + "  doesn't exist!":
238             update.message.reply_text("Room "+room+" doesn't exist!")
239         else :
240             update.message.reply_text("Connection error")
241
242     except (IndexError, ValueError):
243         update.message.reply_text('Use /deleteroom <room>')
244
245 if __name__ == '__main__':
246
247     conf = json.load(open("./bot/helpbot.json"))
248     token = conf["token"]
249     ip_address = conf["ip_address"]
250     updater = Updater(token)
251     dp = updater.dispatcher
252     x = bot()
253     # dp.add_handler(CommandHandler("start", x.start, pass_chat_data=
    True))
254     dp.add_handler(CommandHandler("help", x.help))
255     dp.add_handler(CommandHandler("whereis", x.getUser))
256     dp.add_handler(CommandHandler("whereareall", x.getUsers))
257     dp.add_handler(CommandHandler("wherearealll", x.getUsers2))
258     dp.add_handler(CommandHandler("userlist", x.getUserList))
259     dp.add_handler(CommandHandler("roomlist", x.getRoomList))
260     dp.add_handler(CommandHandler("whoisin", x.getRoom, pass_args=
    True, pass_chat_data=True))
261     dp.add_handler(CommandHandler("deleteuser", x.deleteUser,
    pass_args=True))
262     dp.add_handler(CommandHandler("deleteroom", x.deleteRoom,
    pass_args=True))
263     dp.add_handler(MessageHandler(Filters.photo, x.add))
264     updater.start_polling()
265     updater.idle()
```

Hospital departments finder code

```
1 import folium
2 from folium import plugins
3 import pandas as pd
4 import ipywidgets
5 import os
6 import json
7 import datetime
8
9 UMMlocation = (45.082080, 7.656278)
```

```
map_UMM = folium.Map(location = UMMlocation, width = "75%",
    zoom_start = 17)
map_UMM

startTime=int(datetime.timedelta(minutes=6, seconds=30).total_seconds
    ())

hauseOutline = 'GeoResources/B09.geojson'
display(folium.GeoJson(hauseOutline, name="B09").add_to(map_UMM))
display(map_UMM)

testGeoJson = 'GeoResources/w09.geojson'

def switchPosition(coordinate):
  temp = coordinate[0]
  coordinate[0] = coordinate[1]
  coordinate[1] = temp
  return coordinate

with open(testGeoJson) as f:
  testWay = json.load(f)

for feature in testWay['features']:
    path = feature['geometry']['coordinates']
finalPath = list(map(switchPosition,path))
finalPath

path = 'GeoResources/w11.geojson'
folium.plugins.AntPath([[45.081744513802526, 7.656239569187165],
 [45.081856253813434, 7.656303942203522],
 [45.081899813419504, 7.656470239162445],
 [45.082019128692124, 7.656907439231873],
 [45.08216117035846, 7.656912803649902],
 [45.08214601926422, 7.657256126403808]]).add_to(map_UMM)
map_UMM


select_widget=ipywidgets.Select(
    options=['Option A', 'Option B'],
    value='Option A',
    description='Select',
    disabled=False)

def selectOption(opt):
    if opt == 'Option A':
        print('A')
    if opt == 'Option B':
        print('B')

```

```
ipywidgets.interact(selectOption, opt=select_widget)


class navigator:
    def __init__(self):
        self.geoResources = {}
        self.hospitalLocation =(45.082080, 7.656278)
        self.position = 'w'
        self.destination = 'B09'

        for root, dirs, files in os.walk('GeoResources'):
            for file in files:
                self.geoResources[file.split('.')[0]] = root+'/'+file

    def changeDestination(self,newDestination):
        self.destination = newDestination
        self.redrawMap()

    def changeStartPoint(self, newStartPoint):

        print(f'Selected Start: {newStartPoint}; Selected Target: {
    self.destination}')


    def drawPathWay(self,hospitalMap):

      def switchPosition(coordinate):
        temp = coordinate[0]
        coordinate[0] = coordinate[1]
        coordinate[1] = temp
        return coordinate

      searchString = self.position + self.destination.split('B')[1]
      with open(self.geoResources[searchString]) as f:
            testWay = json.load(f)

      for feature in testWay['features']:
        path = feature['geometry']['coordinates']

      finalPath = list(map(switchPosition,path))
      folium.plugins.AntPath(finalPath).add_to(hospitalMap)

    def drawBuilding(self,hospitalMap):
      hauseOutline = self.geoResources[self.destination]
      folium.GeoJson(hauseOutline, name="geojson").add_to(hospitalMap
    )

    def redrawMap(self):
```

```
103         hospitalMap = folium.Map(location = self.hospitalLocation,
    width = "75%", zoom_start = 17)
104         self.drawPathWay(hospitalMap)
105         self.drawBuilding(hospitalMap)
106         display(hospitalMap)
107
108
109 myNavigator = navigator()
110 def displayWay(whereTo):
111      myNavigator.changeDestination(whereTo)
112
113 def changePosition(whereFrom):
114     myNavigator.changeStartPoint(whereFrom)
115
116 selectHouse_widget=ipywidgets.Select(
117
118 options=['B09',
119     'B11'],
120     value='B09',
121     description='Target',
122     disabled=False)
123
124
125 def selectHouse(way):
126     if way == 'B09' :
127         displayWay('B09' )
128     if way == 'B11' :
129         displayWay('B11' )
130
131 selectPosition_widget=ipywidgets.Select(
132     options=['Main entrance'],
133     value='Main entrance',
134     description='Start',
135     disabled=False)
136
137 def selectPosition(position):
138     if position == 'Main entrance':
139         changePosition('w')
140
141 ipywidgets.interact(selectPosition, position=selectPosition_widget)
142 ipywidgets.interact(selectHouse, way=selectHouse_widget)
```

# Appendix B

# Appendix

**1. What was your first impression when you entered the Website?**

- Different features of website.
- UI Design
- Simplicity
- Color pallete
- Accessibility
- Other

**2. Are functionalities clear?**

- 5 Stars
- 4 Stars
- 3 Stars
- 2 Stars
- 1 Stars

**3. Does the Dashboard show bugs?**

- Yes
- No
- Maybe

**4. How likely are you to recommend us to a friend or colleague?**

- 5 Stars
- 4 Stars
- 3 Stars
- 2 Stars
- 1 Stars