

# INTERNET OF THINGS

## REPORT ASSIGNMENT 1

STUDENT NAME: TRAN QUANG HUY

STUDENT ID: GCD18457

Class: GCD0602

# ASSIGNMENT 1 FRONT SHEET

Qualification	TEC Level 5 HND Diploma in Computing		
Unit number and title	Unit 43: Internet of Things		
Submission date		Date Received 1st submission	
Re-submission Date		Date Received 2nd submission	
Student Name	Tran Quang Huy	Student ID	GCD18457
Class	GCD0602	Assessor name	Tran Trong Minh
<b>Student declaration</b> I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.			
		Student's signature	Huy

## Grading grid

[illegible]

☐ **Summative Feedback:**

☐ **Resubmission Feedback:**

**Grade:**

**Assessor Signature:**

**Date:**

**Internal Verifier's Comments:**

**Signature & Date:**

# Contents

<b>INTRODUCTION .....</b>	<b>1</b>
<b>Part I. What aspects of IoT are necessary and appropriate when designing software applications.....</b>	<b>2</b>
1. <i>The various forms of IoT functionality. ....</i>	<i>2</i>
2. <i>The standard architecture, frameworks, tools, hardware and APIs available for use in IoT development. ....</i>	<i>13</i>
3. <i>The impact of common IoT architecture, frameworks, tools, hardware and APIs in the software development lifecycle.....</i>	<i>21</i>
4. <i>The specific forms of IoT architecture, frameworks, tools, hardware and APIs for different problem-solving requirements.....</i>	<i>25</i>
5. <i>The specific forms of IoT architecture and justify their use when designing software applications.....</i>	<i>31</i>
<b>Part II. The plan for an appropriate IoT application using common architecture, frameworks, tools, hardware and APIs .....</b>	<b>31</b>
1. <i>The architecture, frameworks, tools, hardware and API techniques available to develop IoT applications....</i>	<i>36</i>
2. <i>The specific problem to solve using IoT. ....</i>	<i>45</i>
3. <i>The most appropriate IoT architecture, frameworks, tools, hardware and API techniques to include in an application to solve this problem. ....</i>	<i>47</i>
4. <i>Apply the selected techniques to create an IoT application development plan .....</i>	<i>52</i>
5. <i>The multiple iterations of your IoT application and modify each iteration with enhancements gathered from user feedback and experimentation.....</i>	<i>60</i>
<b>CONCLUSION .....</b>	<b>66</b>
<b>References .....</b>	<b>67</b>

## TABLE OF FIGURES

Figure 1. The history of Internet of things. (avancer, n.d.) .....	2
Figure 2. How exponentially the IoT is growing. (justcreative, n.d.) .....	3
Figure 3. IoT architecture. (avsystem, n.d.) .....	13
Figure 4. Stage 1: Things, sensors and controllers (avsystem, n.d.) .....	14
Figure 5. Stage 2: Gateways and data acquisition. (avsystem, n.d.).....	15
Figure 6. Stage 3: Edge analytics. (avsystem, n.d.) .....	16
Figure 7. Stage 4: Datacenter/ cloud platform. (avsystem, n.d.).....	16
Figure 8. SDLC for IoT Product .....	21
Figure 9. The architecture of health care monitoring system .....	27
Figure 10. IoT Based Traffic Management System. ....	28
Figure 11. Traffic Management system architecture .....	29
Figure 12. Using Raspberry PI to make traffic management system .....	30
Figure 13. Greeparking's architecture .....	47
Figure 14. Work Breakdown Structure .....	55

## TABLE OF TABLES

Table 1. The advantages and disadvantages of Blynk .....	51
Table 2. Project Charter .....	54
Table 3. Project cost.....	56
Table 4. Equipment cost for prototype.....	57
Table 5. Project risk.....	58
Table 6. Summary of Greeparking version .....	65

## TABLE OF PICTURES

Picture 1. Manufacturing IoT. (themanufacturer, n.d.) .....	6
Picture 2. Automotive IoT. (sam-solutions, n.d.) .....	7
Picture 3. Transportation and Logistics. (techtipsnapps, n.d.) .....	8
Picture 4. Retail IoT. (42gears, n.d.) .....	9
Picture 5. Public Sector IoT. (publictechnology, n.d.) .....	10
Picture 6. Healthcare IoT. (koreabiomed, n.d.) .....	11
Picture 7. General Safety Across All Industries IoT. (newgenapps, n.d.) .....	12
Picture 8. IoT Framework.....	17
Picture 9. IoT Tool. ....	18
Picture 10. IoT Hardware .....	19
Picture 11. IoT APIs. ....	20
Picture 12. Smart Health care monitoring system.....	25
Picture 13. Structure for doctor.....	26
Picture 14. Structure for Patient.....	27
Picture 15. Basic elements of IoT architecture. ....	31
Picture 16. Intelligent lighting.....	33
Picture 17. KAA IoT. (youtube, n.d.).....	36
Picture 18. Cisco IoT System. (techrepublic, n.d.) .....	37
Picture 19. Zetta IoT. (isaax, n.d.) .....	37
Picture 20. Microsoft Azure IoT. (medium, medium, n.d.) .....	38
Picture 21. Google Cloud Platform IoT. (google, n.d.) .....	38
Picture 22. ThinkSpeak IoT. (mathworks, n.d.) .....	39
Picture 23. Mainflux IoT. (medium, medium, n.d.).....	40
Picture 24. System On Chip. (dzone, n.d.) .....	41
Picture 25. Industrial Microcontroller (PLC and RTU).....	41
Picture 26. Programable Logic Controller .....	42
Picture 27. Single Board Computer.....	42
Picture 28. Arduino IDE.....	43
Picture 29. Eclipse IoT.....	43
Picture 30. Raspbian IoT. ....	44
Picture 31. Home Assistant.....	44
Picture 32. Parking in Da Nang city.....	45
Picture 33. Traffic jams at Da Nang city. ....	45
Picture 34. GreeParking logo .....	46
Picture 35. Location place to parking using GreeParking .....	46
Picture 36. Blink application .....	48
Picture 37. Widget Box on blink app.....	49
Picture 38. Libraries Blynk to connect any hardware .....	50
Picture 39. How blynk work .....	51
Picture 40. User satisfied result .....	62
Picture 41. Result of User's thinking this project will solve problem .....	63
Picture 42. Feedback on system .....	64
Picture 43. Which part did user find most relevant.....	65

# INTRODUCTION

Over the past few years, IoT has become one of the most important technologies of the 21st century. Now that we can connect everyday objects kitchen appliances, cars, thermostats, baby monitors—to the internet via embedded devices, seamless communication is possible between people, processes, and things.

By means of low-cost computing, the cloud, big data, analytics, and mobile technologies, physical things can share and collect data with minimal human intervention. In this hyper-connected world, digital systems can record, monitor, and adjust each interaction between connected things. The physical world meets the digital world and they cooperate.

With technical advancements, our interaction with information systems is changing, both at work and during leisure time. Information, sensor, and network technology are becoming increasingly small, more powerful, and more frequently used. People no longer only encounter information technology at common points in their lives, such as in offices or at desks, but as information and communication infrastructures, which are present in increasing areas of everyday life. These infrastructures are characterized by the fact that they not only include classic devices.



# Part I. What aspects of IoT are necessary and appropriate when designing software applications

## 1. The various forms of IoT functionality.

### 1.1. What is IoT?

The Internet of Things (IoT) is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

The definition of the Internet of Things has evolved due to the convergence of multiple technologies, real-time analytics, machine learning, commodity sensors, and embedded systems. Traditional fields of embedded systems, wireless sensor networks, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of Things. In the consumer market, IoT technology is most synonymous with products pertaining to the concept of the "smart home", covering devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smartphones and smart speakers. (wikipedia, n.d.)

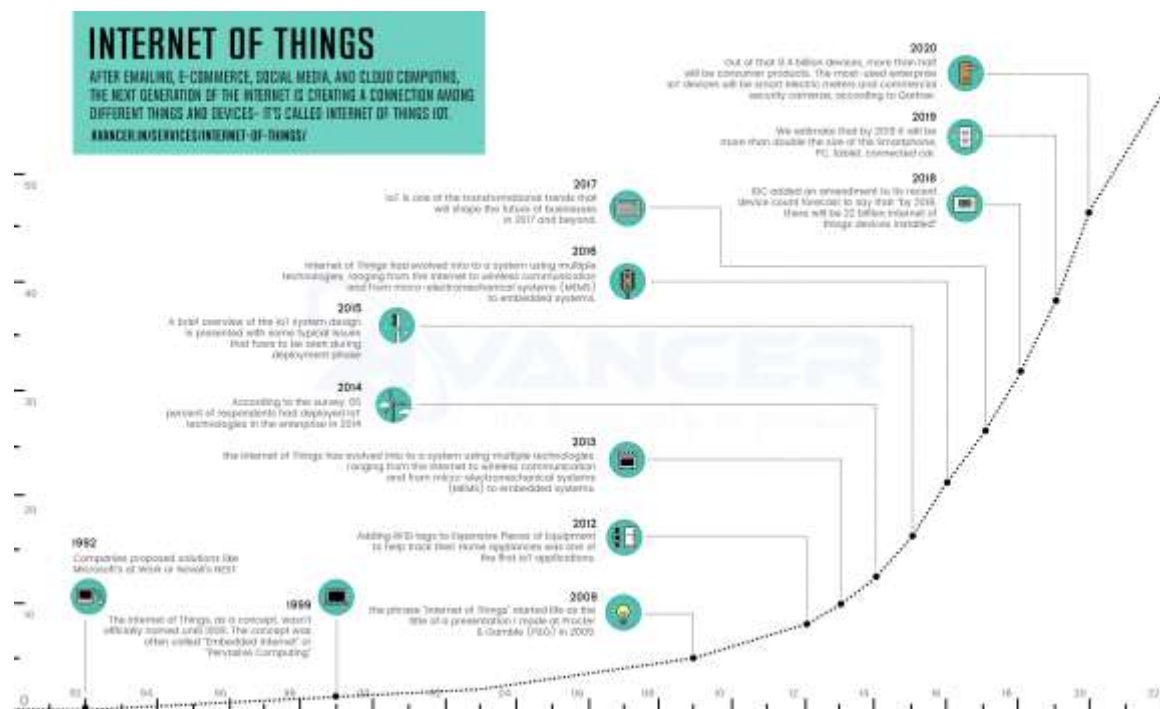


Figure 1. The history of Internet of things. (avancer, n.d.)



The Internet of Things (IoT) describes the network of physical objects “things” that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. These devices range from ordinary household objects to sophisticated industrial tools. With more than 7 billion connected IoT devices today, experts are expecting this number to grow to 10 billion by 2020 and 22 billion by 2025.

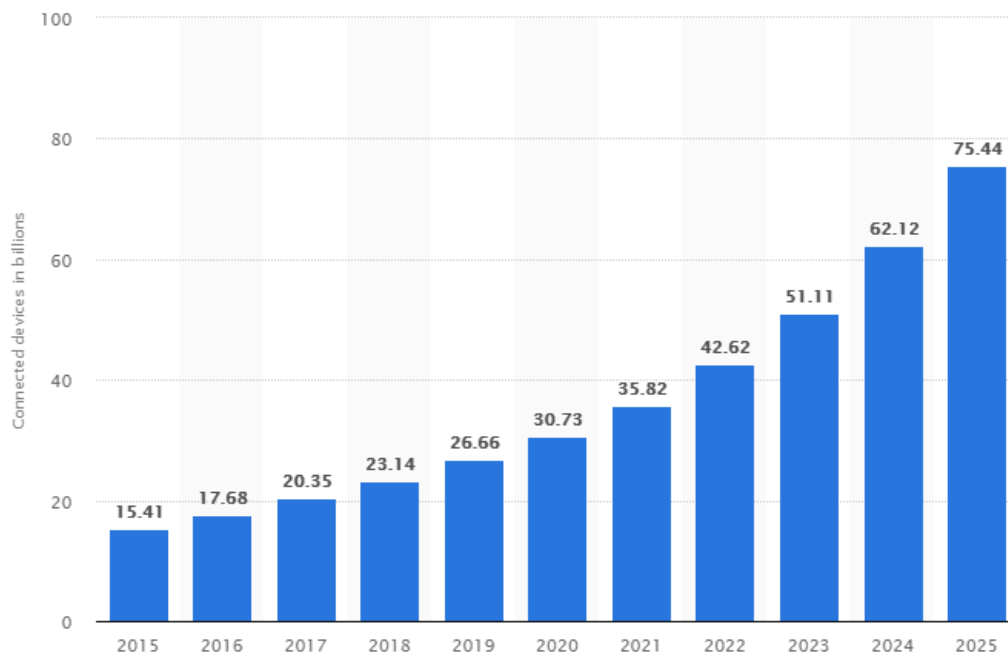


Figure 2. How exponentially the IoT is growing. (justcreative, n.d.)

## 1.2. What Technologies Have Made IoT Possible?

While the idea of IoT has been in existence for a long time, a collection of recent advances in the number of different technologies has made it practical.

- **Access to low-cost, low-power sensor technology:** Affordable and reliable sensors are making IoT technology possible for more manufacturers.
- **Connectivity:** A host of network protocols for the internet has made it easy to connect sensors to the cloud and to other “things” for efficient data transfer.
- **Cloud computing platforms:** The increase in the availability of cloud platforms enables both businesses and consumers to access the infrastructure they need to scale up without actually having to manage it all.
- **Machine learning and analytics:** With advances in machine learning and analytics, along with access to varied and vast amounts of data stored in the cloud, businesses can gather insights faster and more easily. The emergence of these allied technologies continues to push the boundaries of IoT and the data produced by IoT also feeds these technologies.
- **Conversational artificial intelligence (AI):** Advances in neural networks have brought natural-language processing (NLP) to IoT devices (such as digital personal assistants Alexa, Cortana, and Siri) and made them appealing, affordable, and viable for home use.

## 1.3. What Is Industrial IoT?

**Industrial IoT (IIoT)** refers to the application of IoT technology in industrial settings, especially with respect to instrumentation and control of sensors and devices that engage cloud technologies. Recently, industries have used machine-to-machine communication (M2M) to achieve wireless automation and control. But with the emergence of cloud and allied technologies (such as analytics and machine learning), industries can achieve a new automation layer and create new revenue and business models. IIoT is sometimes called the fourth wave of the industrial revolution, or Industry 4.0. The following are some common uses for IIoT:

- Smart manufacturing
- Preventive and predictive maintenance
- Smart power grids
- Smart cities
- Connected and smart logistics
- Smart digital supply chains

## 1.4. What Are IoT Applications?

IoT applications use machine learning algorithms to analyze massive amounts of connected sensor data in the cloud. Using real-time IoT dashboards and alerts, you gain visibility into key performance indicators, statistics for mean time between failures, and other information. Machine learning–based algorithms can identify equipment anomalies and send alerts to users and even trigger automated fixes or proactive countermeasures.

With cloud-based IoT applications, business users can quickly enhance existing processes for supply chains, customer service, human resources, and financial services. There's no need to recreate entire business processes.

The ability of IoT to provide sensor information as well as enable device-to-device communication is driving a broad set of applications. The following are some of the most popular applications and what they do.

- **Create new efficiencies in manufacturing through machine monitoring and product-quality monitoring:** Machines can be continuously monitored and analyzed to make sure they are performing within the required tolerances. Products can also be monitored in real-time to identify and address quality defects.
- **Improve the tracking and “ring-fencing” of physical assets:** Tracking enables businesses to quickly determine asset location. Ring-fencing allows them to make sure that high-value assets are protected from theft and removal.
- **Use wearables to monitor human health analytics and environmental conditions:** IoT wearables enable people to better understand their own health and allow physicians to remotely monitor patients. This technology also enables companies to track the health and safety of their employees, which is especially useful for workers employed in hazardous conditions.
- **Drive efficiencies and new possibilities in existing processes:** One example of this is the use of IoT to increase efficiency and safety in fleet management. Companies can use IoT fleet monitoring to direct trucks, in real-time, to improve efficiency.
- **Enable business process changes:** An example of this is the use of IoT devices to monitor the health of remote machines and trigger service calls for preventive maintenance. The ability to remotely monitor machines is also enabling new product-as-a-service business models, where customers no longer need to buy a product but instead pay for its usage.

**Organizations best suited for IoT are those that would benefit from using sensor devices in their business processes:**

- **Manufacturing:**

Manufacturers can gain a competitive advantage by using production-line monitoring to enable proactive maintenance on equipment when sensors detect an impending failure. Sensors can actually measure when production output is compromised. With the help of sensor alerts, manufacturers can quickly check equipment for the accuracy or remove it from production until it is repaired. This allows companies to reduce operating costs, get better uptime, and improve asset performance management.



*Picture 1. Manufacturing IoT. (themanufacturer, n.d.)*

- **Automotive:**

The automotive industry stands to realize significant advantages from the use of IoT applications. In addition to the benefits of applying IoT to production lines, sensors can detect impending equipment failure in vehicles already on the road and can alert the driver with details and recommendations. Thanks to aggregated information gathered by IoT-based applications, automotive manufacturers and suppliers can learn more about how to keep cars running and car owners informed.

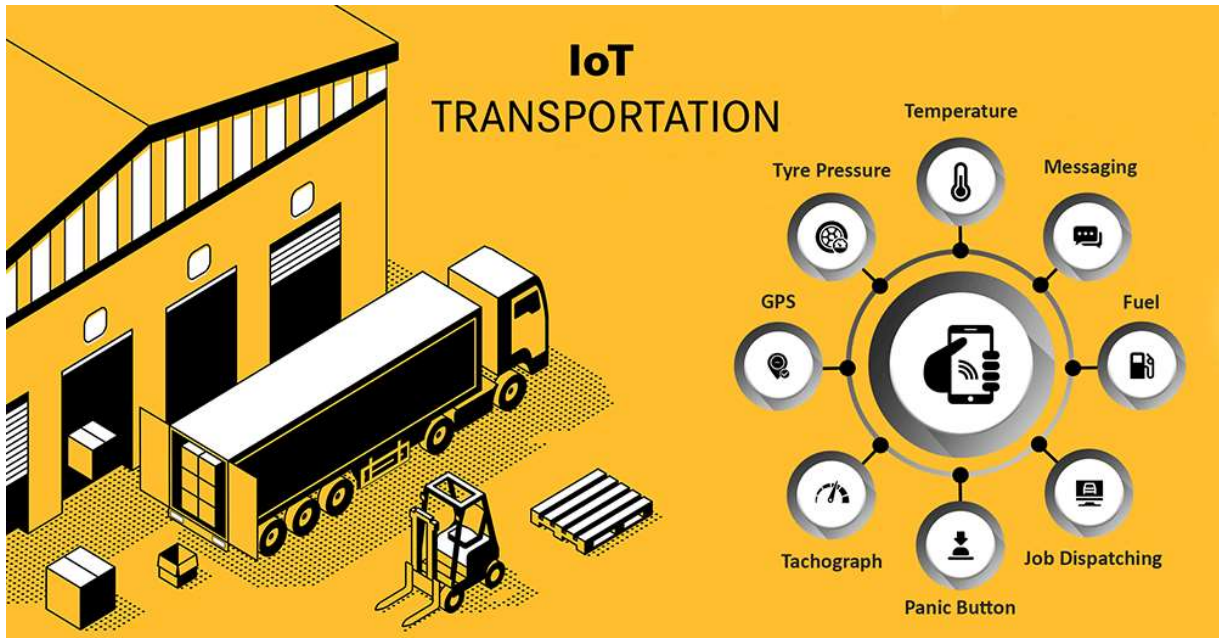


Picture 2. Automotive IoT. (sam-solutions, n.d.)



- **Transportation and Logistics:**

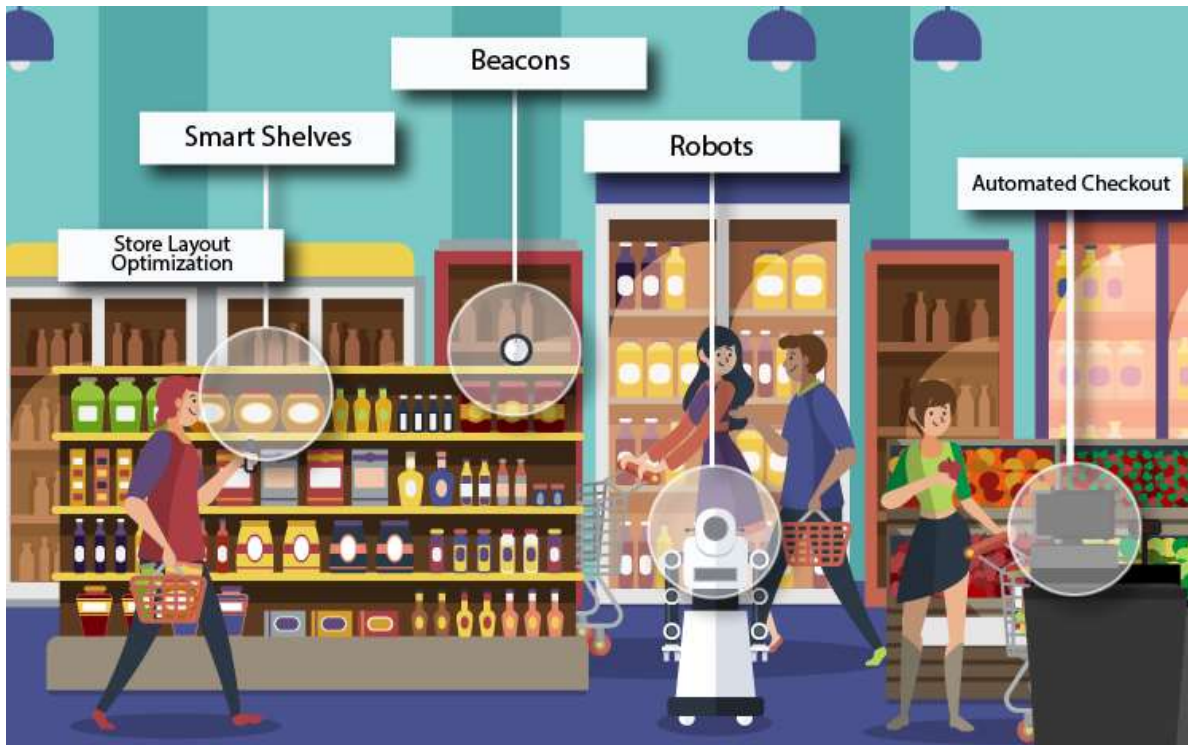
Transportation and logistical systems benefit from a variety of IoT applications. Fleets of cars, trucks, ships, and trains that carry inventory can be rerouted based on weather conditions, vehicle availability, or driver availability, thanks to IoT sensor data. The inventory itself could also be equipped with sensors for track-and-trace and temperature-control monitoring. The food and beverage, flower, and pharmaceutical industries often carry temperature-sensitive inventory that would benefit greatly from IoT monitoring applications that send alerts when temperatures rise or fall to a level that threatens the product.



Picture 3. Transportation and Logistics. (techtipsnapps, n.d.)

- **Retail:**

IoT applications allow retail companies to manage inventory, improve customer experience, optimize the supply chain, and reduce operational costs. For example, smart shelves fitted with weight sensors can collect RFID based information and send the data to the IoT platform to automatically monitor inventory and trigger alerts if items are running low. Beacons can push targeted offers and promotions to customers to provide an engaging experience.



Picture 4. Retail IoT. (42gears, n.d.)



- **Public Sector:**

The benefits of IoT in the public sector and other service-related environments are similarly wide-ranging. For example, government-owned utilities can use IoT based applications to notify their users of mass outages and even smaller interruptions of water, power, or sewer services. IoT applications can collect data concerning the scope of an outage and deploy resources to help utilities recover from outages with greater speed.



*Picture 5. Public Sector IoT. (publictechnology, n.d.)*

- **Healthcare:**

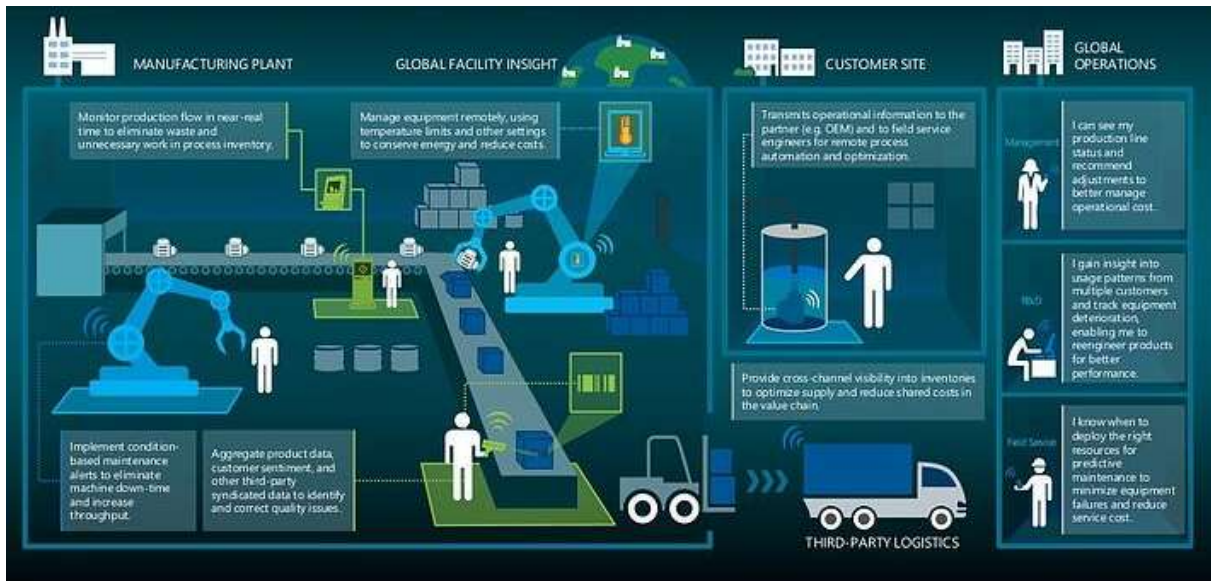
IoT asset monitoring provides multiple benefits to the healthcare industry. Doctors, nurses, and orderlies often need to know the exact location of patient-assistance assets such as wheelchairs. When a hospital's wheelchairs are equipped with IoT sensors, they can be tracked from the IoT asset monitoring application so that anyone looking for one can quickly find the nearest available wheelchair. Many hospital assets can be tracked this way to ensure proper usage as well as financial accounting for the physical assets in each department.



Picture 6. Healthcare IoT. (koreabiomed, n.d.)

- **General Safety Across All Industries:**

In addition to tracking physical assets, IoT can be used to improve worker safety. Employees in hazardous environments such as mines, oil and gas fields, and chemical and power plants, for example, need to know about the occurrence of a hazardous event that might affect them. When they are connected to IoT sensor based applications, they can be notified of accidents or rescued from them as swiftly as possible. IoT applications are also used for wearables that can monitor human health and environmental conditions. Not only do these types of applications help people better understand their own health, they also permit physicians to monitor patients remotely.



Picture 7. General Safety Across All Industries IoT. (newgenapps, n.d.)

## 2. The standard architecture, frameworks, tools, hardware, and APIs are available for use in IoT development.

### 2.1. The architecture of IoT

While every IoT system is different, the foundation for each Internet of Things architecture as well as its general data process flow is roughly the same. First of all, it consists of the Things, which are objects connected to the Internet which by means of their embedded sensors and actuators are able to sense the environment around them and gather information that is then passed on to IoT gateways. The next stage consists of IoT data acquisition systems and gateways that collect the great mass of unprocessed data, convert it into digital streams, filter and pre-process it so that it is ready for analysis. The third layer is represented by edge devices responsible for further processing and enhanced analysis of data. This layer is also where visualization and machine learning technologies may step in. After that, the data is transferred to data centers which can be either cloud-based or installed locally. This is where the data is stored, managed and analyzed in-depth for actionable insights.

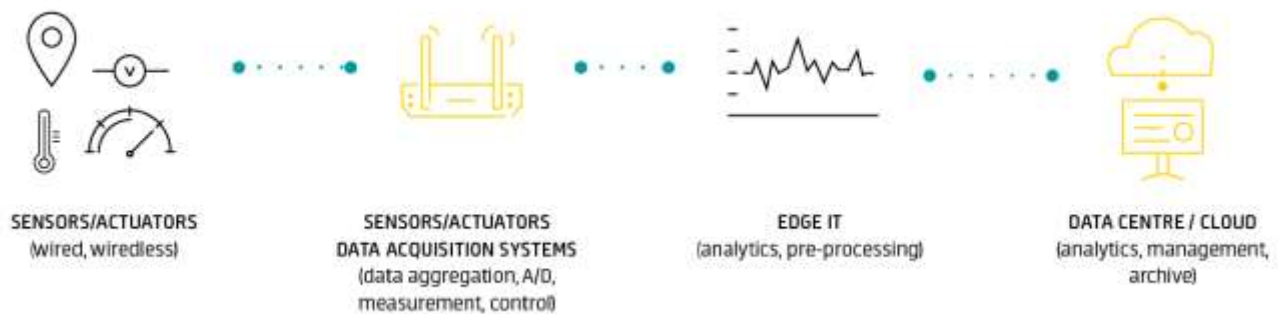


Figure 3. IoT architecture. (avsystem, n.d.)

## Things, sensors and controllers

A thing in the context of “Internet of Things”, should be equipped with sensors and actuators thus giving the ability to emit, accept and process signals.

As the basis for every IoT system, connected devices are responsible for providing the essence of the Internet of Things which is the data. To pick up physical parameters in the outside world or within the object itself, they need sensors. These can be either embedded in the devices themselves or implemented as standalone objects to measure and collect telemetry data.

What is also important is that the connected objects should not only be capable of communicating bidirectionally with their corresponding gateways or data acquisition systems but also being able to recognize and talk to each other to gather and share information and collaborate in real-time to leverage the value of the whole deployment. In the case of resource-constrained and battery-operated devices particularly, achieving this is not an easy task since such communication requires lots of computing power and consumes precious energy and bandwidth. Therefore, robust architecture can only enable effective device management when it uses fit for purpose, secure and lightweight communication protocols, such as Lightweight M2M which has become a leading standard protocol for the management of low power lightweight devices which are typical for many IoT use cases



Figure 4. Stage 1: Things, sensors and controllers (avsystem, n.d.)



## Gateways and data acquisition

The data from the sensors start in the analog form which needs to be aggregated and converted into digital streams for further processing. Data acquisition systems perform these data aggregation and conversion functions.

Although this layer still functions in close proximity with sensors and actuators on given devices, it is essential to describe it as a separate IoT architecture stage as it is crucial for the processes of data collection, filtering and transfer to edge infrastructure and cloud-based platforms. Given the massive volume of input and output that million-device deployments may generate, capabilities for the aggregation, selection, and transportation of data should be in the spotlight. As intermediaries between the connected things and the cloud and analytics, gateways and data acquisition systems provide the necessary connection point that ties the remaining layers together.

Another aspect that gateways support is security. Because the gateways are responsible for managing the information flow in both directions, with the help of proper encryption and security tools they can prevent IoT cloud data leaks as well as reduce the risk of malicious outside attacks on IoT devices.

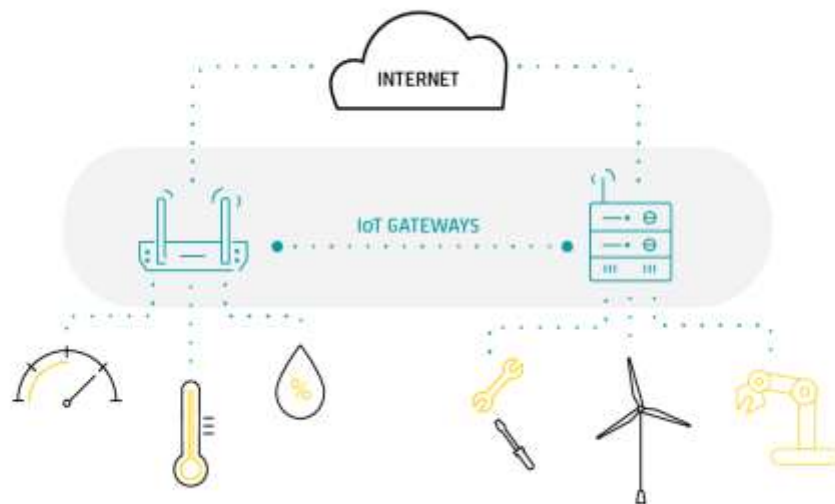


Figure 5. Stage 2: Gateways and data acquisition. (avsystem, n.d.)

## Edge analytics

Once IoT data has been digitized and aggregated, it may require further processing before it enters the data center, this is where Edge Analytics comes in.

As edge infrastructure can be located closer to the data source in physical terms, it is easier and quicker for it to act on the IoT material in real-time and provide output in the form of instant actionable intelligence. In this scenario, only the larger chunks of data which really need the power of the Cloud to be processed are forwarded there. By minimizing network exposure, security can be significantly enhanced, while reduced power and bandwidth consumption contribute to a more efficient leveraging of business resources.



Figure 6. Stage 3: Edge analytics. (avsystem, n.d.)

## Data centre / cloud platform

Data that needs more in-depth processing gets forwarded to physical data centers or cloud based systems.

If sensors are neurons and the gateway is the backbone of IoT, then the cloud is the brain in the Internet of Things body. Contrary to edge solutions, a data center or a cloud-based system is designed to store, process and analyze massive volumes of data for deeper insights using powerful data analytics engines and machine learning mechanisms that edge systems would never be able to support.

If furnished with proper user application solutions, the cloud can provide business intelligence and presentation options that help humans interact with the system, control and monitor it and make informed decisions on the basis of reports, dashboards, and data viewed in real-time.



Figure 7. Stage 4: Datacenter/ cloud platform. (avsystem, n.d.)

## 2.2. IoT Framework

IoT is a key part of a large IoT ecosystem, which promotes and links all elements in the scheme. It allows device management, handles communication protocols on software and hardware, collects/analyses information, improves information flow and intelligent apps functionality.

IoT Framework tells about the basic structure underlying an IoT solution/product.



Picture 8. IoT Framework.

There are four basic components of IoT framework:

- **Device Hardware:** Requires an idea of architecture and working of various micro-controllers along with various sensors.
- **Device Software:** Requires writing programs to configure your controller and make it act accordingly. Also requires knowledge for how API's work inside micro-controllers and how you can make libraries for programming.
- **Communication and Cloud Platform:** Requires basics of wired and wireless communication. Cloud in itself is an indispensable part of IoT and requires knowledge for how Cloud technology works and its IoT integration.
- **Cloud Application:** It is a software program where cloud-based and local components work together which have faster and easier accessibility. It serves the purpose of improving our ability to use the system to maximum potential.

IoT platform is a cloud software/platform that connects to the sensors, gateways, end-user applications or any other physical thing that has a network connectivity and is an integral component of IoT framework.

For developers, an IoT platform provides a set of ready-to-use features that greatly speed up the development of applications for connected devices as well as take care of cross-device compatibility.

Moreover, IoT platforms provide data security involving encryption, comprehensive identity management, end-to-end data flow encryption, device authentication, user access rights management, and private cloud infrastructure for sensitive data to avoid potentially compromising breaches.



### 2.3. IoT Tool

IoT Tools stands for the Internet of Things Tools. It is a network or connection of devices, vehicles, equipment applying embedded electronics, home appliances, buildings and many more. This helps in collecting and exchanging different kinds of data. It also helps the user to control the devices remotely over network.

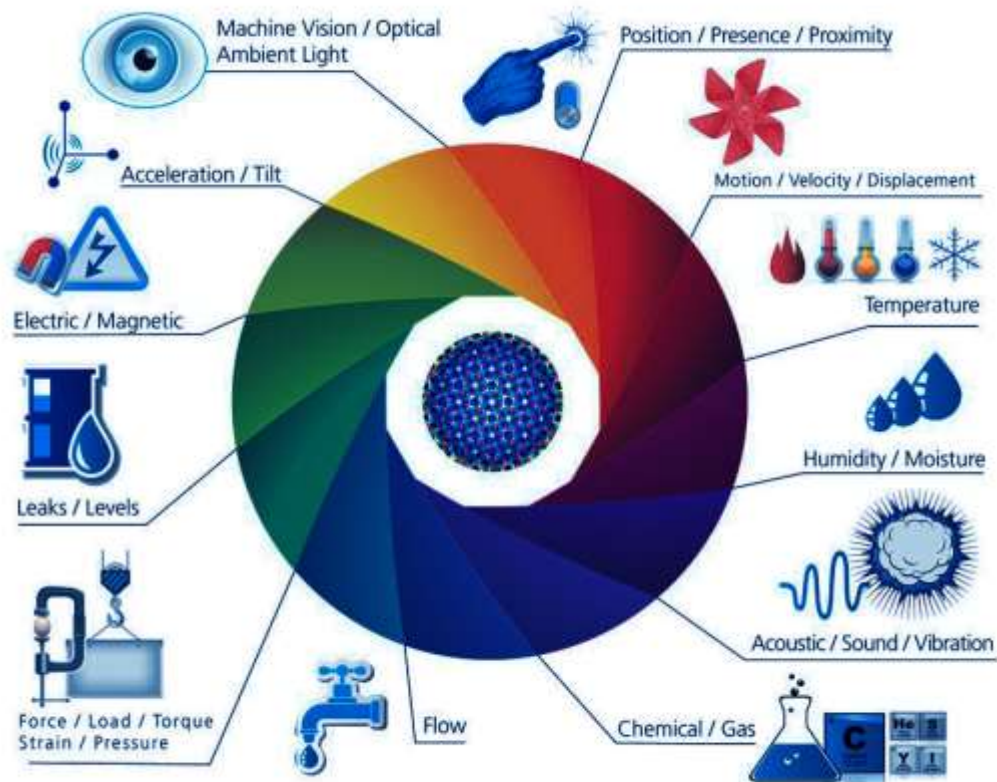
Some IoT tools that help developers in developing IoT applications and devices:



Picture 9. IoT Tool.

## 2.4. IoT Hardware

IoT Hardware includes a wide range of devices such as devices for routing, bridges, sensors, etc. These IoT devices manage key tasks and functions such as system activation, security, action specifications, communication, and detection of support-specific goals and actions. (flair, n.d.)

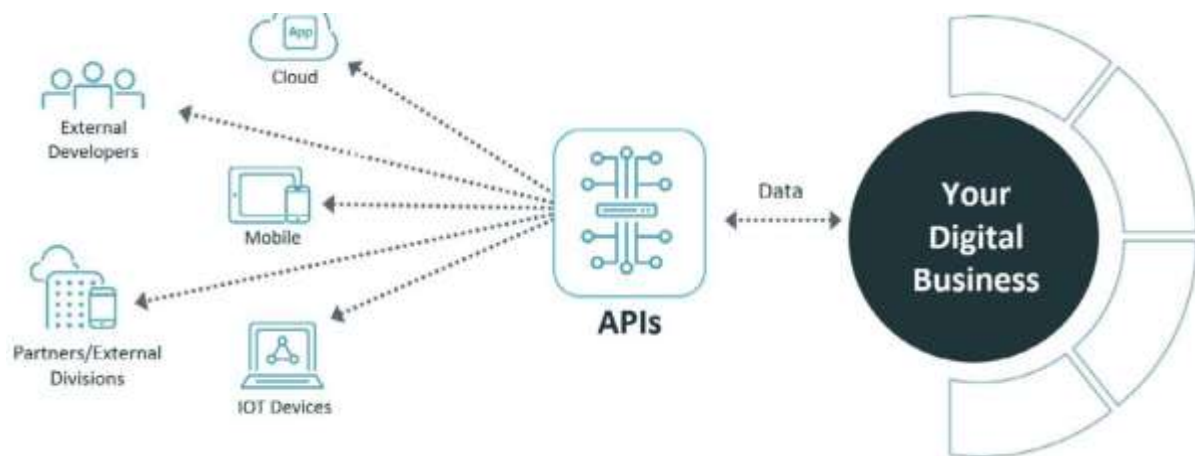


Picture 10. IoT Hardware

## 2.5. IoT APIs

The APIs allow devices to talk to each other in consistent and structured way that makes it really easy to get them to communicate. Writing about IoT we need to mention about various of protocols (CoAP, XMPP, MQTT, WAMP, OMG DDS, Stomp) and frameworks (Web RTC, ASP.NET SignalR, webSocket.org, Couchbase, Socket IO, Meteor) so it leads that IoT API management system also integrates IoT devices environment. The next very important issue is security. API Management system creates a secure user-friendly identity. It ensures secure connections to devices across mobile and the IoT environment. Besides using a platform it is much easier to identify and neutralize SQL injection, DoS attacks, and other online threats. (thecattlecrew, n.d.)

The application program (or programming) interface, or API, is arguably what really ties together the connected “things” of the “internet of things.” IoT APIs are the points of interaction between an IoT device and the internet and/or other elements within the network.



Picture 11. IoT APIs.

### 3. The impact of common IoT architecture, frameworks, tools, hardware and APIs in the software development lifecycle.

#### Typical SDLC Phases for an IoT Product (vdoo, n.d.)

The number of phases in an SDLC may vary, depending on the business and on its product goals, but is typically between five and seven phases. The following diagram describes typical track of an SDLC process for an IoT product:

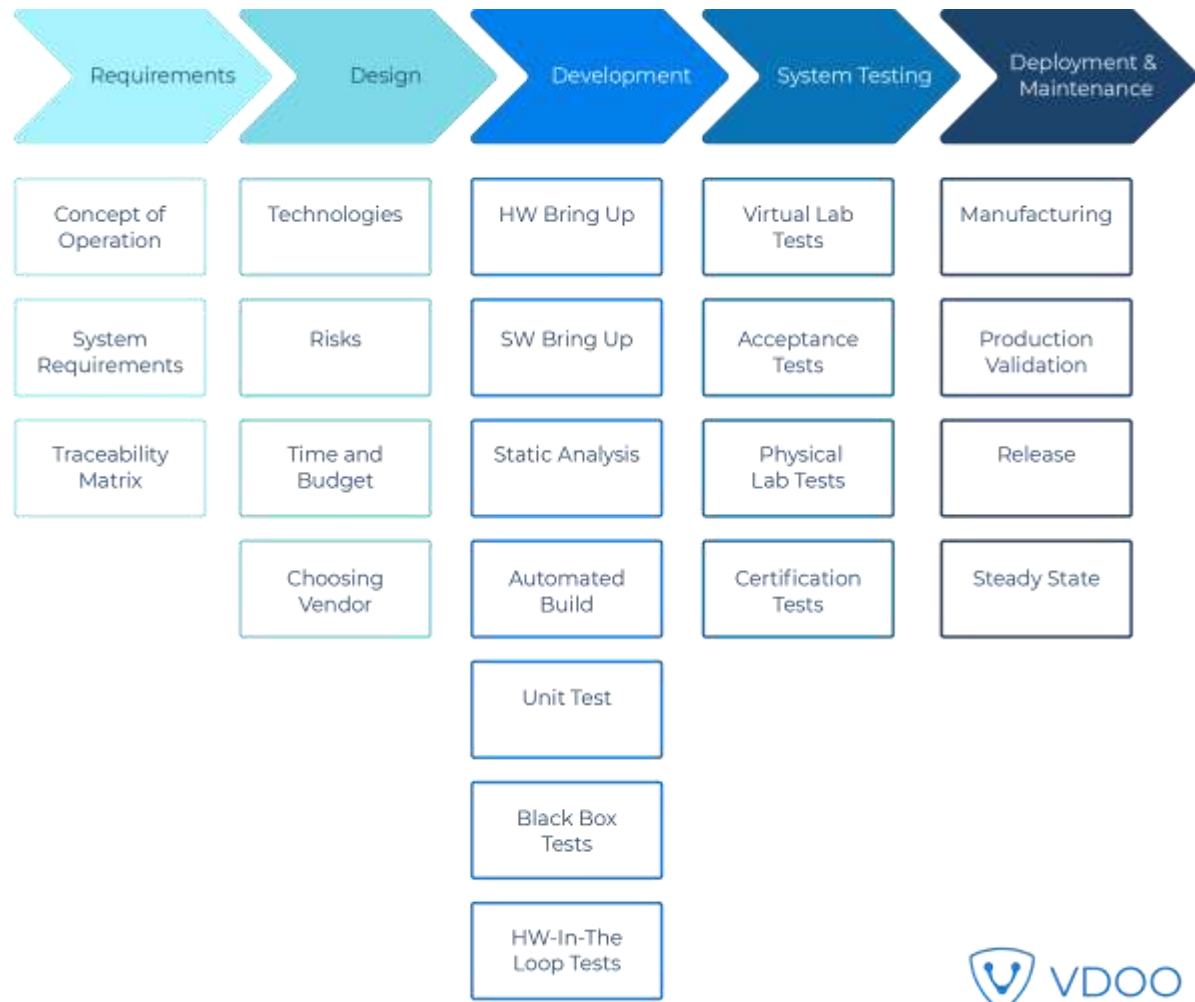


Figure 8. SDLC for IoT Product

The integration of security into the SDLC process in each and every phase is essential for the development of secure software while reducing the total cost and effort involved in security implementation. Continuous security integration into the SDLC process as described in this article will continuously improve product security and will also increase the probability that the product meets business goals.

## Phase 1: Requirements

When planning a new product, the first questions that should be answered are - what's the purpose of this product? What problems should it solve? Who will use the product? How will they use it? What is the input/output of the product?

To answer these questions properly, a well-established product management process should be conducted, including the basic steps of problem definition and outline of MRD/PRD to provide a map of arguments and features as well as user stories and security implications.

Next, the Concept of Operations should be defined, including the characteristics of the proposed system from the user's perspective. Next, the System Requirements Document (SRD) should be prepared, which is the formal statement of the system requirements, including a list of functional requirements, data requirements, system interfaces, and physical requirements. Then, the Traceability Matrix, the table that links the requirements to their origins and traces through the project life cycle, should be prepared as well.

It is essential to define the product's security requirements at this early phase.

To do that, first, a definition of the product's key security risks is needed. For example, the type of information that the product will process, its functionality, etc.

Second, a definition of the security standards or certifications that the product should meet is needed. The decision of whether to support a specific standard usually depends on the market or industry and should be received as a requirement from the product management team.

## **Phase 2: Design**

When designing the new product, among the questions that need to be answered are - what Technologies shall be used? What are the Risks in the design? What are the Time and Budget constraints?

In this phase, the system and software design documents are created, based on the requirement specifications defined in the previous phase.

The system design and the software design documents provide a detailed description of the various features and operations to support the functional requirements of the IoT product.

The design should clearly define all the architectural modules of the product, along with its communication and data flow representation as well as external and 3rd party modules.

When developing an IoT product, a Chip Vendor should be chosen as well as additional vendors for the peripheral components such as RAM memory, ROM memory, NAND flash, Audio, Switch, etc.

From a security perspective, the architecture and design should be reviewed using threat modeling techniques to point out identified potential threats, such as structural vulnerabilities, and to enumerate and prioritize them.

Threat modeling is based on four major steps: 1. Decomposing application 2. Categorizing the threats 3. Ranking the threats 4. Mitigating the threats It is highly important to map out the security threats during the early design phase, as it can be very hard to mitigate them in later phases. For example, if we decide that the product will require a secure boot mechanism, in most cases we will need to support a Root-of-Trust (RoT) to implement this requirement. In such a case, support from the chip vendor is needed for RoT integration. This integration will probably impact the basic software architecture design, therefore, early attention to this requirement is clearly more efficient.

## **Phase 3: Development**

During the development phase, the code is built, tested, integrated and managed. The developer begins the coding according to what was defined in phases 1 and 2 above (Requirements and Design).

The first step in IoT product development is the Hardware Bring Up, which includes the CAD drawing of the proposed hardware layout, the editing of the electrical circuit cycles and the measuring of the power supply noise.

The second step is the Software Bring Up, which includes integration with the hardware team, setting up the initial board state by fuse burning, boot and kernel bring up, file system bring up (in case of Linux-based devices), and the development of the company's proprietary code.

The third step is the running of the Static Analysis tools on the entire codebase, a process that provides an understanding of the code structure and existing bugs, and helps ensure that the code adheres to industry standards.

The fourth step is the Build Automation, the process of creating the software build automatically. It has associated processes that includes fetching, patching, configuration, compilation, installation, and packaging.

The final steps in the development phase are the test code steps that include Unit Tests, Black-Box Tests, and Hardware-In-The-Loop Tests. This step includes a simulation of the sensors, actuators and mechanical components that connects all the I/O of the tested ECU, long before the final system is integrated.

## Phase 4: System Testing

The focus of this phase is to verify that the product meets the technical, functional, and business requirements that were defined earlier, based on a series of tests that should cover the entire system.

A few of the major tests performed during this phase are the following:

- Virtual Lab Tests to utilize virtualization services that simulate all of the hardware dependencies required to perform a full system test. Successful simulation of the real product, although not an easy task, will significantly save cost and time in later phases.
- Acceptance Tests to evaluate the system's compliance with business requirements and assess whether it is acceptable for delivery.
- Once the tests above are completed, Physical Lab Tests are performed to allow a complete system test of the real product in its real environment.
- The last test is the Certification Test where certification-related tests are performed, such as safety, security, and network protocols.

From the security perspective, during the system testing phase, there are a few security guidelines that should be met:

- Identify critical security problems by running static and dynamic analysis of the complete final firmware image.
- Run penetration tests to check how the product handles various abuse cases such as malformed input handling, authentication/authorization bypass attempts, and overall security posture.
- Ensure that security assumptions specified during the design phase are still relevant.

## Phase 5: Deployment and Maintenance

The main focus of this phase is to make the system operational in a live environment and to maintain system operational stability.

The first step in the Deployment and Management phase is the Manufacturing step that in most cases is performed by 3rd party manufacturing companies.

The second step is the Production Validation where the device is validated under a production environment.

The third step is the Release step where the product is delivered to the market.

The fourth and last step that closes the entire SDLC process is the Steady State step, where the product is being used by customers and being maintained and monitored for its performance and for the user experience.

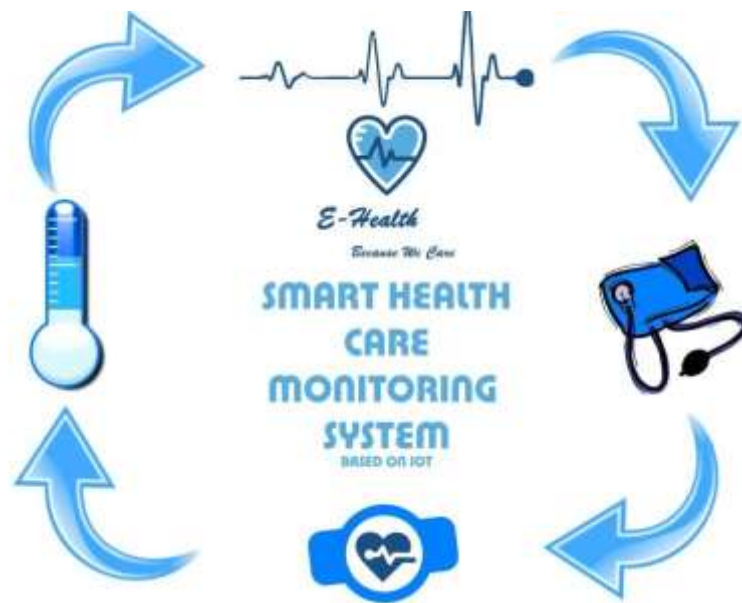
From a security requirements perspective, it is important to verify that the product manufacturing follows security standards guidelines (as defined usually by industry) and that the product is shipped with no unwanted physical ports open. Finally, it is required to keep the product software updated to ensure a healthy security system, with special attention to 3rd party vulnerability management.



#### 4. The specific forms of IoT architecture, frameworks, tools, hardware, and APIs for different problem-solving requirements.

##### 4.1. IoT Based Health Monitoring System

This project is significant in various ways because, in today's world, everyday many lives are affected because the patients are not timely and properly operated. Also for real-time parameter values are not efficiently measured in a clinic as well as in hospitals. Sometimes it becomes difficult for hospitals to frequently check patient's conditions. Also, continuous monitoring of ICU patients is very difficult. To deal with these types of situations, our system is beneficial. Our system is designed to be used in hospitals and homes also for measuring and monitoring various parameters like temperature, ECG, heart rate, blood pressure. The results can be recorded using Arduino. Also, the doctors can see those results on the android app. The system will also generate an alert notification that will be sent to the doctor. Our system is useful for monitoring health system of every person through easily attach the device and record it. In which we can analyze the patient's condition through their past data, we will recommend medicines if any emergency occurred through symbolic A.I. (arduino, n.d.)

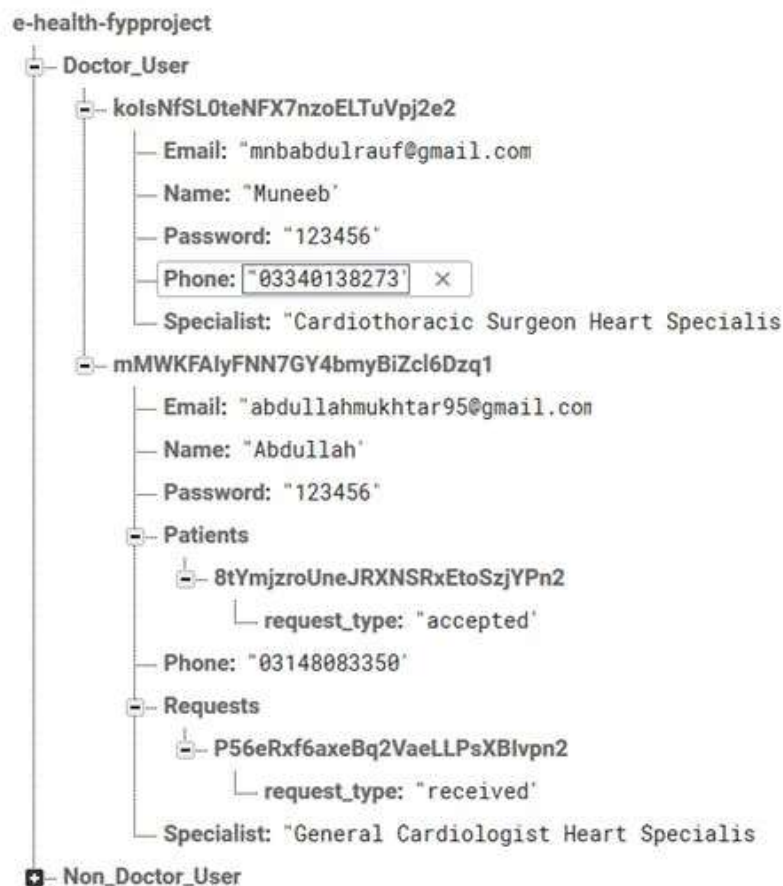


Picture 12. Smart Health care monitoring system

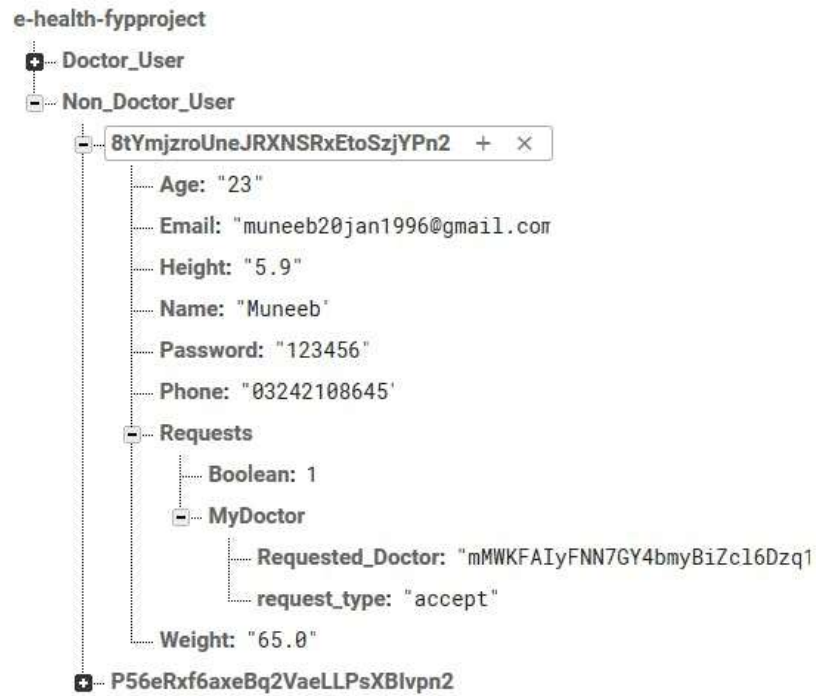


## Work Flow:

- First of all we have to make Communication between Arduino and android app using Bluetooth.
- After establishing the communication successfully start creating android app using android studio and use the code from above link after this send some data to Arduino in our scenario we have 4 modules so we have 16 combinations like patient want only temperature data or ECG, Blood pressure or temperature and ECG, etc. so for these 16 combinations we use Hexa table which starts from 0 and ends at 'F' so after reading the incoming value Arduino decides which type of data user wants.
- After that Arduino start reading that data from sensors and transfer that data to Arduino using HC05 and android receive that data and send to firebase which is real-time database below is the structure of firebase database



Picture 13. Structure for doctor



Picture 14. Structure for Patient

- The architecture of this project:

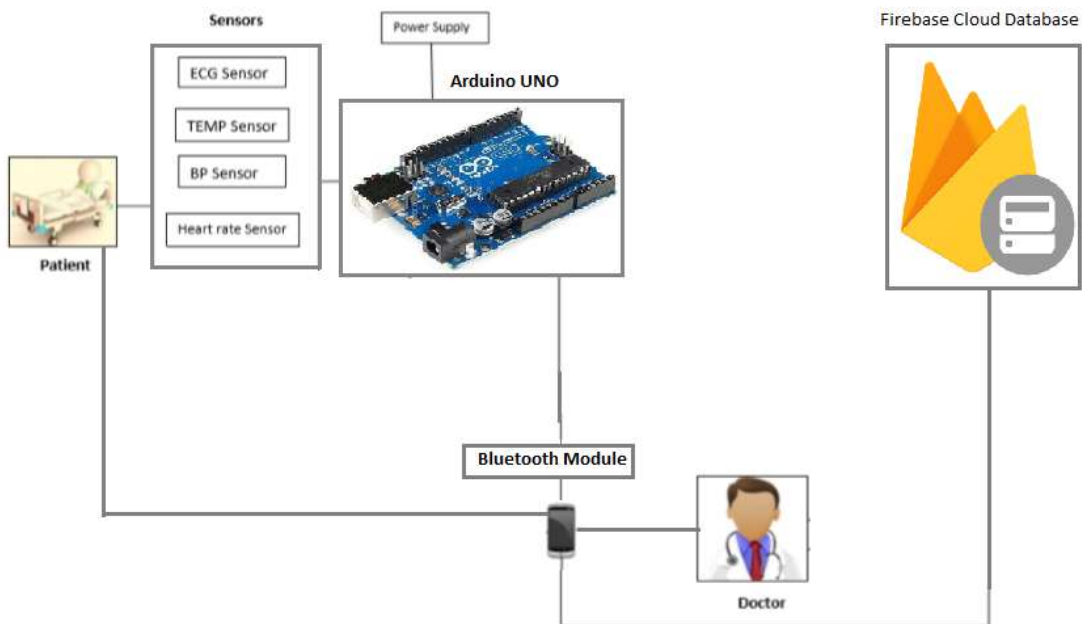


Figure 9. The architecture of health care monitoring system

## 4.2. IoT Based Traffic Management System

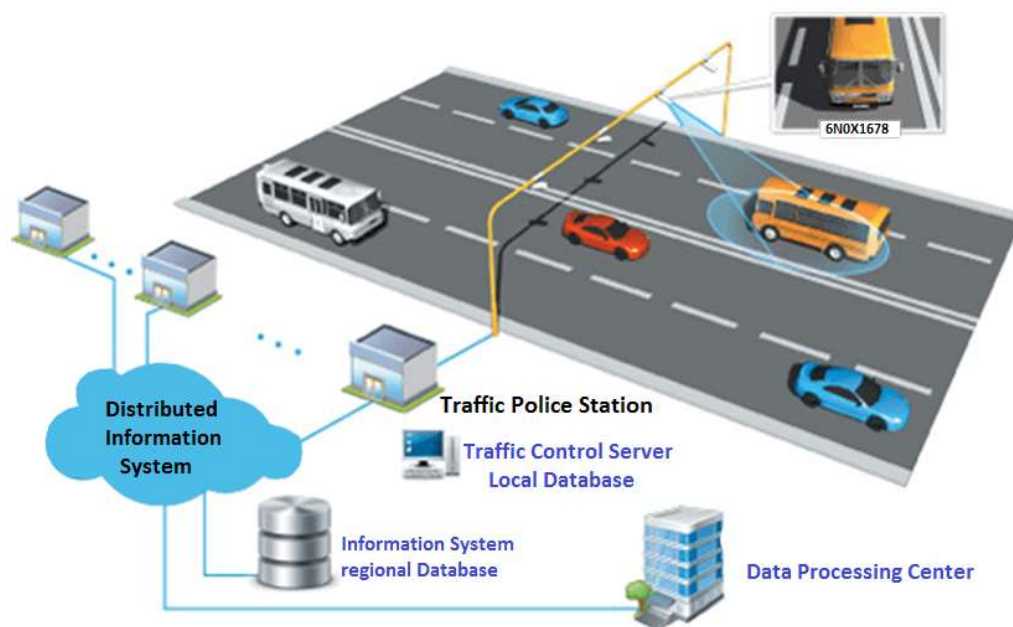


Figure 10. IoT Based Traffic Management System.

All metropolitan cities face traffic congestion problems, especially in the downtown areas. Normal cities can be transformed into “smart cities” by exploiting information and communication technologies (ICT). The paradigm of Internet of Thing (IoT) can play an important role in the realization of smart cities. This paper proposes an IOT based traffic management solution for smart cities and to coordinate with ambulance drivers to find the signal status and choose the path where traffic flow can be dynamically controlled and traffic violations are been identified by onsite traffic officers through centrally monitored or controlled through the Internet. However, the scheme proposed is general and can be used in any Metropolitan city without the loss of generality. If any ambulance will come on a signal then it will shows the green path for that ambulance and the rest of the paths are red. (pantechsolutions, n.d.)

### Important Features:

- Finding an emergency path for an emergency situation is easy.
- This IoT project can be used anywhere.
- Can identify traffic violators at night.
- It shows green light only for an ambulance, fire truck, or emergency vehicle.

### Existing System:

- It is difficult to identify Traffic Violators.
- There is no IOT based Traffic Management System.

### Proposed System:

- IoT based traffic management
- Easy to find the path for the emergency conditions in the ambulance.
- The Traffic violators are captured and send to Police.

### Advantages:

- Can be used anywhere
- No need of human power to identify violators during the night.

### System Architecture :

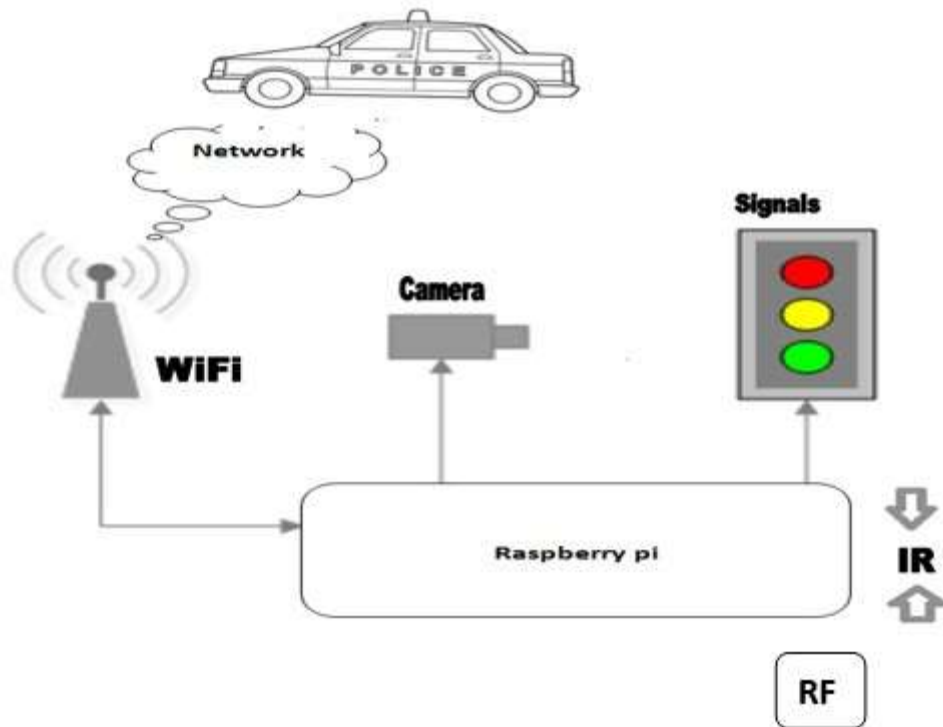


Figure 11. Traffic Management system architecture

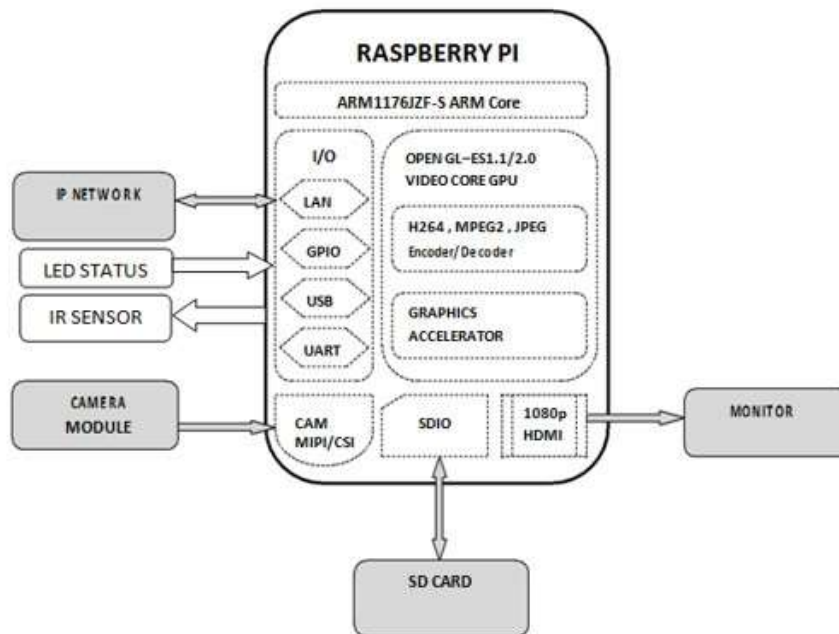


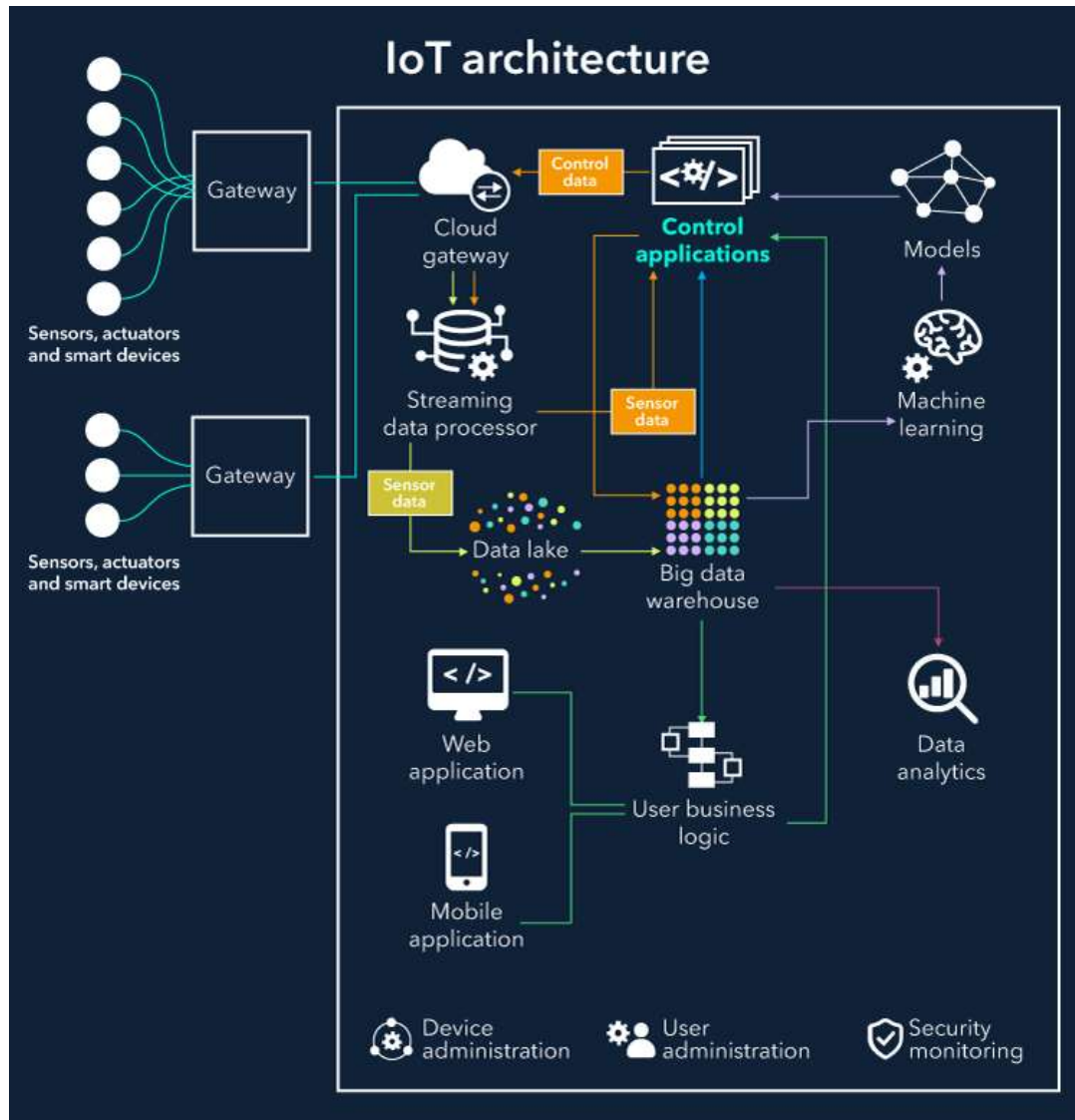
Figure 12. Using Raspberry Pi to make traffic management system

### Hardware and Software Specification:

- **Hardware Requirement:**
  - Internet Connection
  - 1 GB/2GB RAM
  - 8 GB Disk
  - Above 1.2 GHz Processor
  - Camera
  - IR sensor
- **Software Requirement:**
  - OS: Raspbian
  - Language: python

## 5. Evaluate specific forms of IoT architecture and justify their use when designing software applications.

Approach to IoT architecture is reflected in the IoT architecture diagram which shows the building blocks of an IoT system and how they are connected to collect, store and process data.



Picture 15. Basic elements of IoT architecture.



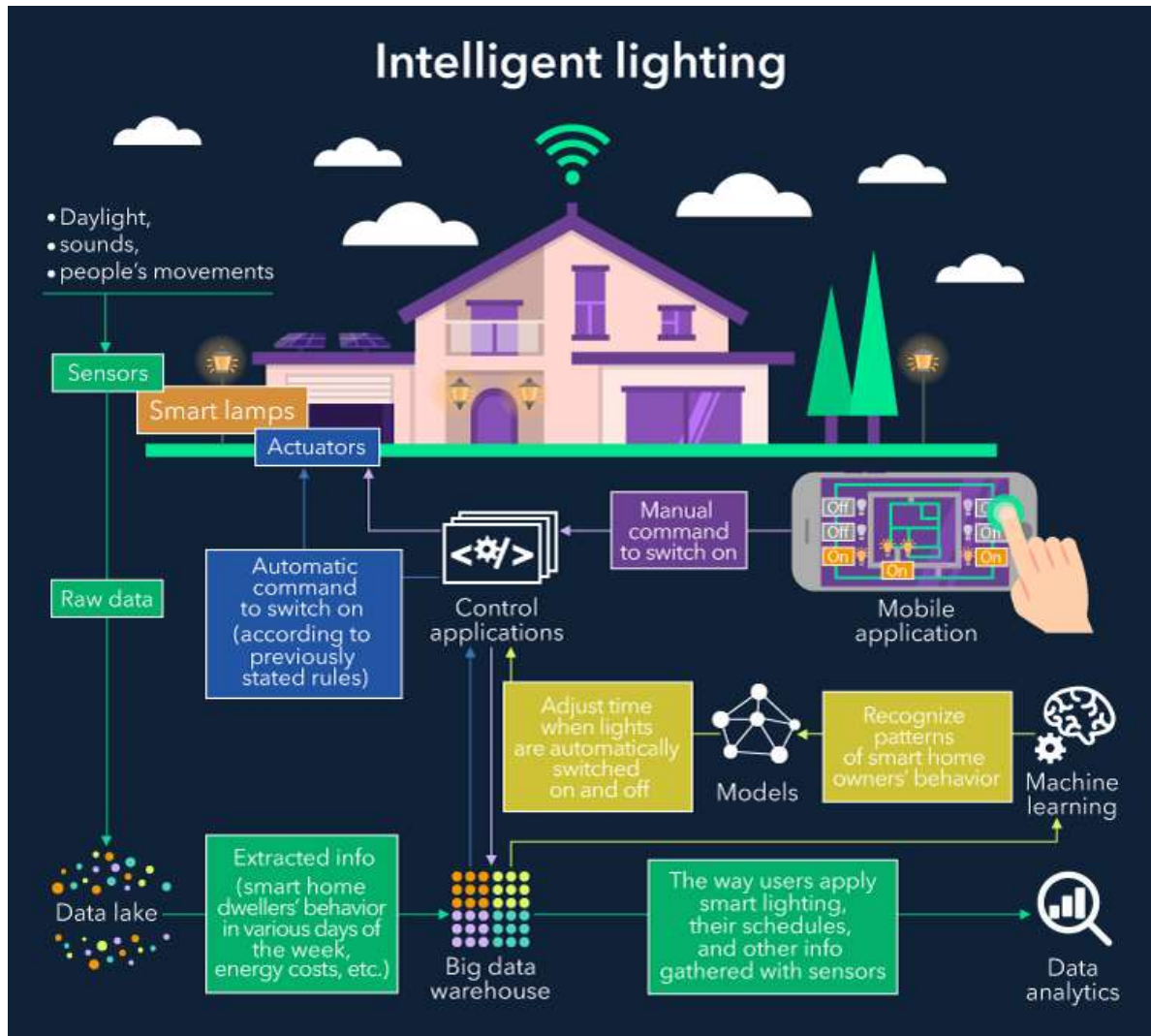
IoT architecture contains the following components:

- **Things:** equipped with sensors to gather data and actuators to perform commands received from the cloud.
- **Gateways:** for data filtering, preprocessing and moving it to the cloud and vice versa, – receiving commands from the cloud.
- **Cloud gateways** to ensure data transition between field gateways and central IoT servers.
- Streaming data processors to distribute the data coming from sensors among relevant IoT solution's components.
- **Data lake** for storing all the data of defined and undefined value.
- **Big data warehouse** for collecting valuable data.
- **Control applications** to send commands to actuators.
- **Machine learning** to generate the models which are then used by control applications.
- **User applications** to enable users to monitor control of their connected things.
- **Data analytics** for manual data processing.

IoT architecture also contains device and user management components to provide stable and secure functioning of things and control user access issues. (scnsoft, n.d.)

## Intelligent lighting

Let's see how IoT architecture elements work together by the example of smart yard lighting as a part of a smart home: a bright illustration of how an IoT solution simultaneously contributes to user convenience and energy efficiency. There are various ways a smart lighting system can function, and we'll cover basic options.



Picture 16. Intelligent lighting.



- **Basic components**

Sensors take data from the environment (for example, daylight, sounds, people's movements). Lamps are equipped with the actuators to switch the light on and off. A data lake stores raw data coming from sensors. A big data warehouse contains the extracted info smart home people's behavior in various days of the week, energy costs and more.

- **Manual monitoring and manual control**

Users control the smart lighting system with a mobile app featuring the map of the yard. With the app, users can see which lights are on and off and send commands to the control applications that further transmit them to lamp actuators. Such an app can also show which lamps are about to be out of order.

- **Data analytics**

Analyzing the way users apply smart lighting, their schedules (either provided by users or identified by the smart system) and other info gathered with sensors, data analysts can make and update the algorithms for control applications.

Data analytics also helps in assessing the effectiveness of the IoT system and revealing problems in the way the system works. For example, if a user switches off the light right after a system automatically switches it on and vice versa, there might be gaps in the algorithms, and it's necessary to address them as soon as possible.

- **Automatic control's options and pitfalls**

The sensors monitoring natural light send the data about the light to the cloud. When the daylight is not enough (according to previously stated threshold), the control apps send automatic commands to the actuators to switch on the lamps. The rest of the time the lamps are switched off.

However, a lighting system can be "baffled" by the street illumination, lamps from neighboring yards and any other sources. Extraneous light captured by sensors can make the smart system conclude that it's enough light, and lighting should be switched off. Thus, it makes sense to give the smart system a better understanding of the factors that influence lighting and accumulate these data in the cloud.

When sensors monitor motions and sounds, it's not enough just to switch on the light when movements or sounds are identified in the yard or switch all the lamps off in the silence. Movements and sounds can be produced, for example, by pets, and cloud applications should distinguish between human voices and movements and those of pets. The same is about the noises coming from the street and neighboring houses and other sounds. To address this issue, it's possible to store the examples of various sounds in the cloud and compare them with the sounds coming from the sensors.

- **Machine learning**

Intelligent lighting can apply models generated by machine learning, for example, to recognize the patterns of smart homeowners' behavior (leaving home at 8 am, coming back at 7 pm) and accordingly adjust the time when lights are switched on and off (for example, switch the lamps on 5 minutes before they will be needed).

Analyzing users' behavior in a long-time perspective, a smart system can develop advanced behavior. For example, when sensors don't identify typical movements and voices of home inhabitants, a smart system can "suppose" that smart home dwellers are on a holiday and adjust the behavior: for example, occasionally

switch on the lights to give the impression that the house is not empty (for security reasons), but do not keep the lights on all the time to reduce energy consumption.

- **User management options**

To ensure efficient user management, the smart lighting system can be designed for several users with role distribution: for example, owner, inhabitants, guests. In this case, the user with the title “owner” will have full control over the system (including changing the patterns of smart light behavior and monitoring the status of the yard lamps) and priorities in giving commands (when several users give contradicting commands, an owner’s command will be the one control apps execute), while other users will have access to a limited number of the system’s functions. “Inhabitants” will be enabled to switch on and off the lamps with no opportunity to change settings. “Guests” will be able to switch on and off the light in some parts of the house and have no access to controlling the lights, for example, near the garage.

Apart from role distribution, it’s essential to consider ownership (as soon as one system can control over 100 thousand of households, and it’s important that a dweller of a smart home manages the lighting in his yard, and not the one of a neighbor).

## Part II. The plan for an appropriate IoT application using common architecture, frameworks, tools, hardware, and APIs

1. The architecture, frameworks, tools, hardware and API techniques are available to develop IoT applications.

### 1.1. IoT Framework

- KAA IoT:



Picture 17. KAA IoT. (youtube, n.d.)

Kaa IoT is one of the most effective and rich Open Source Internet of Things Cloud Platforms, where anyone can freely implement their smart product concepts. You can manage an N number of devices connected to each other with cross-device interoperability on this platform. You can monitor your machine in actual time by providing and configuring remote devices. Kaa enables information exchange between linked devices, the IoT Cloud, information and visualization systems, as well as other elements of IoT Ecosystems

- **Cisco IoT Cloud Connect:**



Picture 18. Cisco IoT System. (techrepublic, n.d.)

Cisco IoT Cloud Connect provides robust, automated, and highly secure connectivity for the enterprise. IoT data management is done by the Cisco Kinetic IoT platform to extract, move and compute the data. As Cisco is very famous for its security services, it protects IoT deployment against threats with a secure IoT architecture.

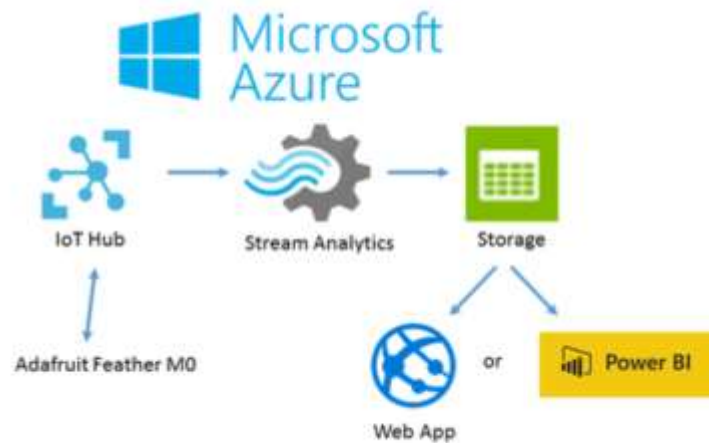
- **ZETTA IoT:**



Picture 19. Zetta IoT. (isaax, n.d.)

Zetta is nothing but a server-oriented platform developed based on the REST, NodeJS, and the Siren hypermedia-API-strip flow-based reactive programming philosophy. After being abstracted as REST APIs they are connected with cloud services. These internet services include tools for visualizing machine analytics and support such as Splunk. It builds a geo-distributed network through connectivity with systems like Heroku to endpoints like Arduino and Linux hackers.

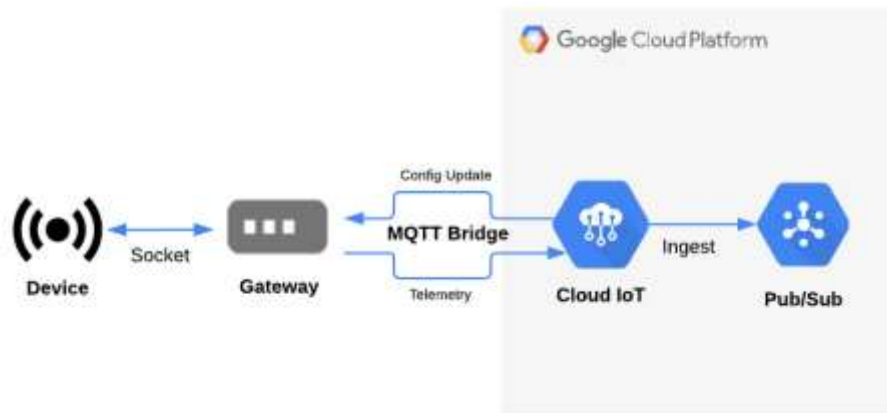
- **Microsoft Azure IoT:**



Picture 20. Microsoft Azure IoT. (medium, medium, n.d.)

Without the Microsoft Azure solution, a cloud service giant with AWS and Google Cloud platform, the comparison of the IoT platform will be not complete. The Microsoft Azure IoT Suite provides preconfigured solutions and the ability to personalize and develop new solutions to meet the project requirements. The strongest safety mechanisms, superb scalability and simple integration with your current or future systems are achieved through Microsoft Azure Internet of thing Suite.

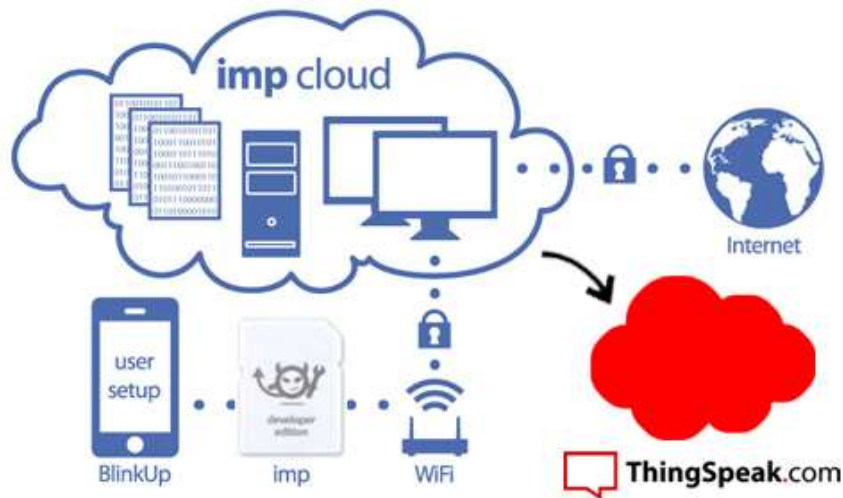
- **Google Cloud Platform IoT framework:**



Picture 21. Google Cloud Platform IoT. (google, n.d.)

Things can be done by Google. Google Cloud is one of the best IoT systems available today with its end-to-end platform. Google stands out from the others because it can process the large quantity of information using Cloud IoT Core. Due to Google's Cloud Data Studio and Big Query you get advanced analysis. With the help of Google Cloud Platform, you can accelerate your business and with that, you can speed up your device.

- **ThingSpeak:**



Picture 22. ThinkSpeak IoT. (mathworks, n.d.)

ThingSpeak is a relatively young IoT platform that tightly collaborates with MathWorks. This gives the possibility to leverage from timely MATLAB data analysis from numberless sensors. The platform comprises:

- Live data streams aggregation and analytics.
- Data recording from public channels to be further used in newly created private channels.
- Assignment of public channels to share data.
- Visualization of collected data.
- Updates of channel feed via the REST and MQTT APIs.
- MATLAB® online analytical tools for exploring patterns and relationships.
- TimeControl function that enables event-triggered alerts.

- **Mainflux:**



*Picture 23. Mainflux IoT. (medium, medium, n.d.)*

Mainflux is an open-source and patent-free IoT platform that has a rich number of advantageous tools for data collection and management, core analytics, and event schedule. No matter the industry, Mainflux provides:

- connectivity of things and users via HTTP, MQTT, WebSocket, CoAP protocols.
- Device management and provisioning.
- Container-based deployment by Docker.
- Container orchestration by Kubernetes.
- Enhanced data security with customizable API keys and scoped JWT.
- Low OPEX (operating expense) benefits.
- Both protocol and device agnostic.



## 1.2. IoT hardware

- **System On Chip (SOC):**

- **Programmable Logic Controller (PLC)**
- **Remote Terminal Unit (RTU)**



Picture 24. System On Chip. (dzone, n.d.)

SOC are electronic systems that integrate a Microcontroller Unit (MCU) as one of its components. It typically runs on a single MCU and can be 8, 16, 32 bits. It also exposes General Purpose Inputs/Outputs (GPIO) and is programmable using a toolchain that compiles the code (Firmware) and loads it in the MCU's permanent memory (SDRAM).

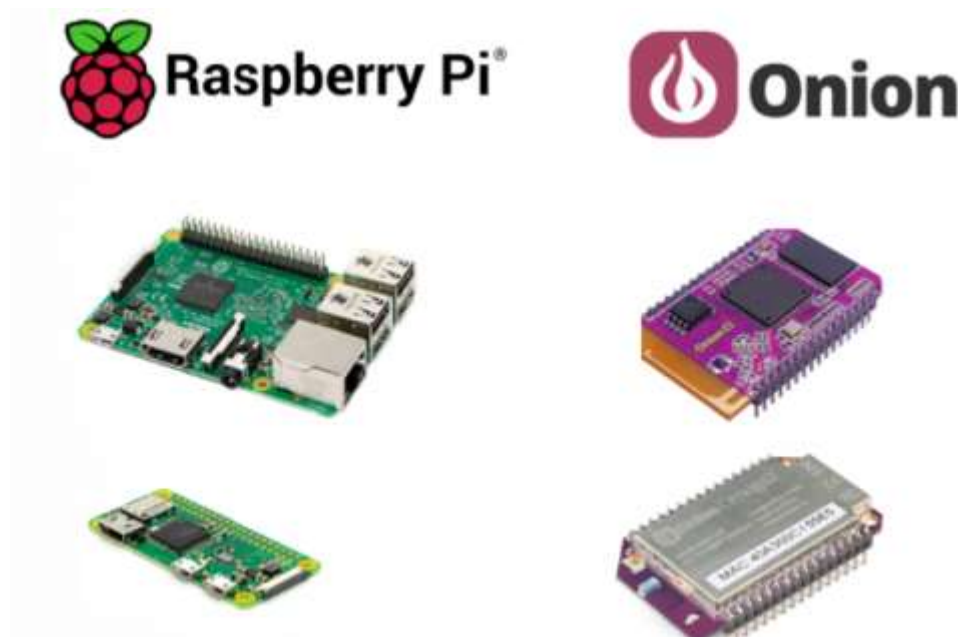
- **Industrial Microcontroller (PLC and RTU)**



Picture 25. Industrial Microcontroller (PLC and RTU).

In the industrial world, digital computers have to be rugged and adapted for the control of manufacturing processes, such as assembly lines, or robotic devices or any activity that requires high-reliability control and ease of programming and process fault diagnosis. There are two well-known types of Microcontrollers you should know. The Programmable Logic Controller and the Remote Terminal Unit.

A Programmable Logic Controller (PLC) is basically a gigantic microcontroller. It does the same things a microcontroller can do, but with higher speed, performance, and reliability where a microcontroller is really just a tiny low power CPU or computer with some output registers wired to pins.



Picture 26. Programmable Logic Controller.

The Functions of RTUs and PLCs pretty much overlap, but in terms of usage, RTUs tend to be a better fit for wide geographic telemetry, while PLCs are best suited for local area control.

- **Single Board Computer (SBC):**

A single-board computer (SBC) is a complete computer built on single circuit board, with microprocessor(s), memory, input/output (I/O) and other features required of a functional computer. Single-board computers were made as demonstration or development systems, for educational systems, or for use as embedded computer controllers. Many types of home computers or portable computers integrate all their functions onto a single printed circuit board. Unlike a desktop personal computer, single board computers often do not rely on expansion slots for peripheral functions.



Picture 27. Single Board Computer.

The Omega is a personal single-board computer created by a startup company called Onion, released on Kickstarter. It is advertised as "the world's smallest Linux Server". The system combines the tiny form factor and power-efficiency of the Arduino, with the power and flexibilities of the Raspberry Pi, you can find this SBU in a lot of device around, if you own a D-Link router you'll find out that it is powered by an Onion Omega.

### 1.3. IoT Tool

- **Arduino IDE:**



*Picture 28. Arduino IDE.*

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

- **Eclipse IoT:**



*Picture 29. Eclipse IoT.*

If you as an IoT developer are ordained to build IoT devices, Cloud platforms, and Gateways, then Eclipse IoT can be your top bet. Recognized as a collaboration of various companies and individuals who are striving towards the development and establishment of IoT open technologies, Eclipse IoT can make all your IoT dreams come true.

Allowing you to develop, promote and adopt open source IoT technologies, Eclipse IoT is an instrument that can help you gain technical expertise. Simply look into the vast assembly of services and projects delivered by the Eclipse team and you are all covered.

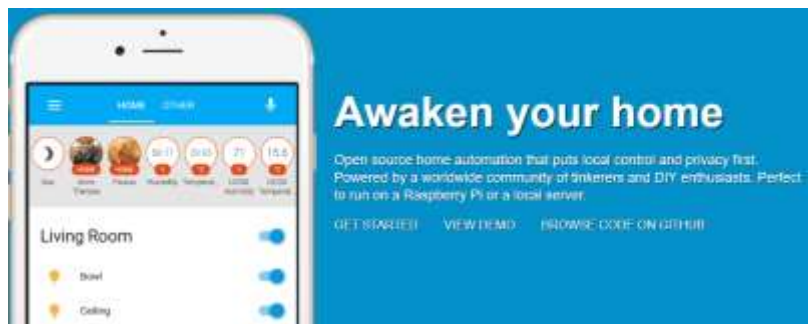
- **Raspbian:**



*Picture 30. Raspbian IoT.*

This IoT IDE is created for Raspberry Pi board by IoT tech enthusiasts. With over 35,000 packages and numerous examples along rapid installation with the use of pre-compiled software makes it an essential IoT development tool. Often, it is regarded as the best tool for Raspberry Pi for IoT app development. Another best part about this tool is that it is under constant development and has widened the reach for computing so that users can gain maximum benefits.

- **Home Assistant:**



*Picture 31. Home Assistant.*

It is an open-source tool which is primarily aimed at home automation and functions with a Python-based coding system. The IoT system developed with this tool can be easily controlled with mobile or desktop browsers. Furthermore, its set up is easy and is trusted for operations, security and privacy. The software supports any systems which are running on Python 3 and all the systems get regular updates within 2 weeks. Despite the lack of cloud components, its ability to protect the data during this internet age gives it an edge over others.

## 2. The specific problem to solve using IoT.

### 2.1. Problem

Static traffic is becoming a problem for big cities in Vietnam today. Public car parks and spots can not be said to meet the parking needs of many people because the number of vehicles exceeds the demand.



Picture 32. Parking in Da Nang city.

In Da Nang city, the key economic sector is tourism. However, according to the statistics in this city, there are already more than **70,000 individual cars** and **tourist cars, taxis** and more than **800,000 motorbikes**, but there is still no official parking. scale cars will go into operation in a total of nearly 200 parking lots will build according to the planning of Da Nang city. Consequently, vehicles parked on the road causing traffic congestion and endangering pedestrians.

It may take a lot of time for the driver to locate the parking location. This affects the work and study of people. Sometimes this also affects the security and order of the city and the parking of a car can cause traffic jams.



Picture 33. Traffic jams at Da Nang city.



## 2.2. Solution

With the **GreeParking** system, it is possible to partially solve the above parking problem by applying IoT.



Picture 34. GreeParking logo

The system will notify and show the driver the status of parking on the road where the driver wants to park so that they can make timely decisions to save more time.

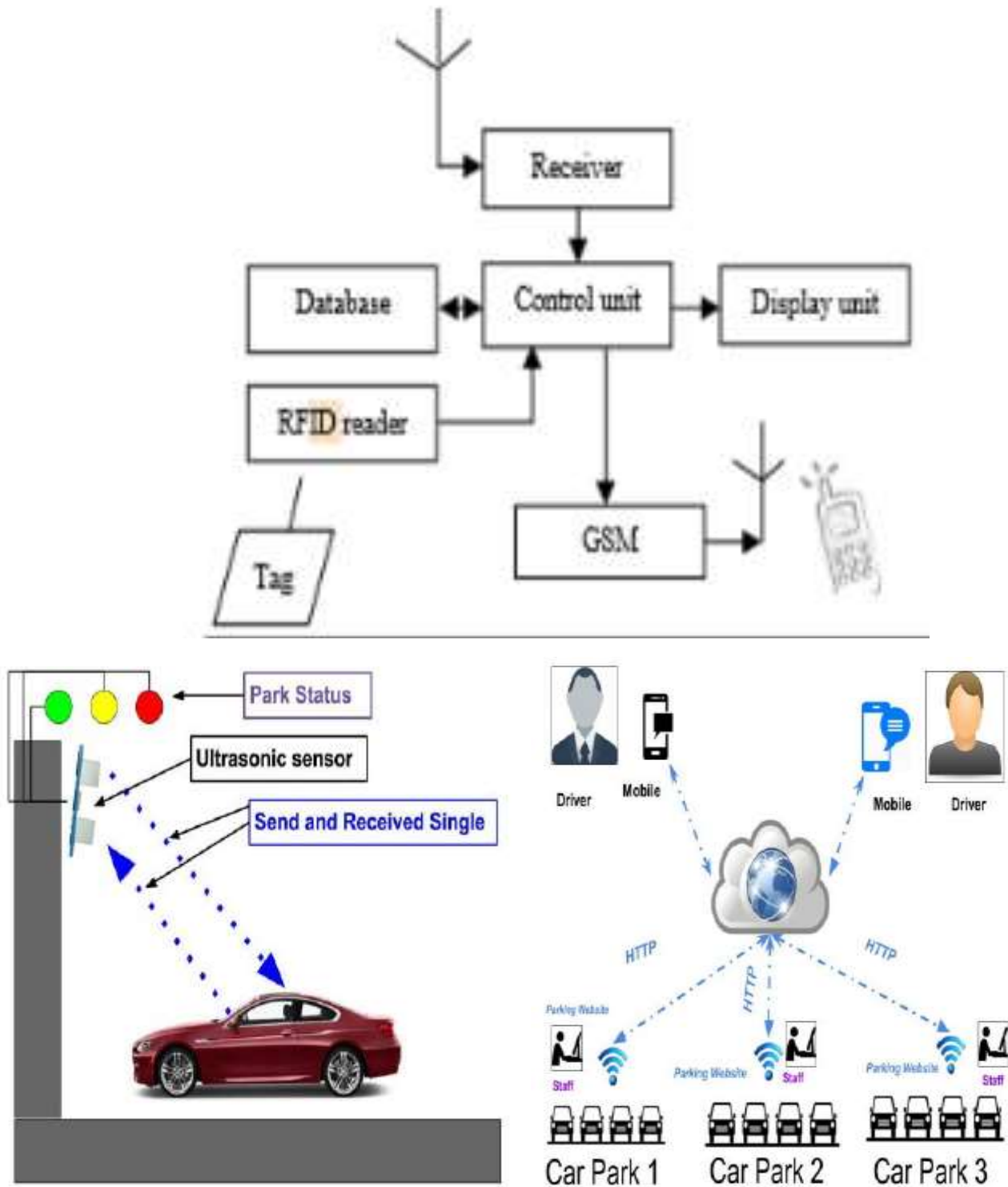


Picture 35. Location place to parking using GreeParking

3. The most appropriate IoT architecture, frameworks, tools, hardware, and API techniques to include in an application to solve this problem.

### Greeparking's architecture

Figure 13. Greeparking's architecture





## Blynk – IoT Platform

Blynk is an Internet-of-Things platform designed to make the development and implementation of smart IoT devices quick and easy. It can be used to read, store, and visualize sensor data and control hardware remotely.

Internet of Things has been all the buzz lately and more and more devices are being talked to the internet every day. With the rise of such amazing technology, the risk of security has also increased substantially. Some of the major concerns in IoT are:

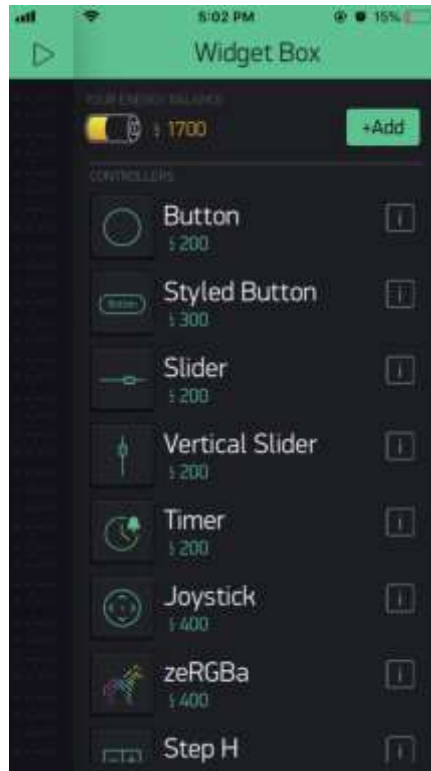
- If IoT devices are sending your data to the internet, the communication needs to be closed and encrypted which cannot be possible without using a dedicated and closed server which is really hard to manage.
- The IoT devices also need to be responsive and again, that is not possible without a server with low latency and high responsiveness.
- In IoT, the platform needs to be compatible with many different types of hardware architecture and devices, so that it doesn't restrict its users with a single type of hardware with limited capabilities.

Keeping in view the problems mentioned above, Blynk is the perfect solution for all these problems. Blynk consists of the following three major components:



Picture 36. Blynk application

**Blynk App:** The mobile app developed by Blynk works as a control panel for visualizing and controlling your hardware. It is available for both Android and iOS. The app offers a very productive interface and various different widgets for different purposes. Blynk works on a currency of its own called energy. New users get 2000 amount of Blynk energy with a free Blynk account and this energy is used to buy and deploy widgets in the projects.



Picture 37. Widget Box on blink app

**Blynk Server:** The most amazing component of the Blynk Platform which makes it all possible is the Blynk Server. Blynk offers a secure, responsive and centralized cloud service through its server which allows all of this communication between the devices. The Blynk server is also available as open-source so you can literally make your own server and make it even more secure with a little tinkering.

**Blynk Library:** The key feature of Blynk platform which makes it scalable and amazing, is the Blynk Library. The Blynk library makes it possible to connect your hardware and get it up and running in a blink. The support for multiple hardware devices including Arduino, ESP8266 and Raspberry Pi is included in the library and it also makes it possible to connect with hardware through many different ways of communication like Wi-Fi, Bluetooth, BLE, USB and GSM.

In the parking problem, the Blink Platform was used to solve this because:

- Provides similar API & user interface for all supported devices and hardware
- Connect to the server using:
  - WIFI
  - Bluetooth and BLE
  - Ethernet
  - USB (Serial)
  - GSM
- The interface utilities are easy to use by the provider
- Drag and drop interface directly without writing code
- Easily integrate and add new functions by using the virtual connection ports built into the blynk app
- Track data history
- Communication from device to device using Widgets
- Email, tweet, realtime notifications, etc.
- Blynk HTTP RESTful API: Blynk HTTP RESTful API allows us to easily read and write values to/from Pins in Blynk apps and Hardware (microcontrollers and microcomputers like Arduino, Raspberry Pi, ESP8266, Particle, etc.). Every **PUT** request will update Pin's state both in apps and on the hardware. Every **GET** request will return the current state/value on the given Pin. We also provide simplified API so you can do updates via **GET** requests.
- Supported Hardware: Blynk supports more than 400 boards already, including support for Arduino, Particle, ARM mbed, TI Energia, MicroPython, Node.js, OpenWRT and many Single Board Computers. You can add your own connection types easily.
- Libraries to connect any hardware:

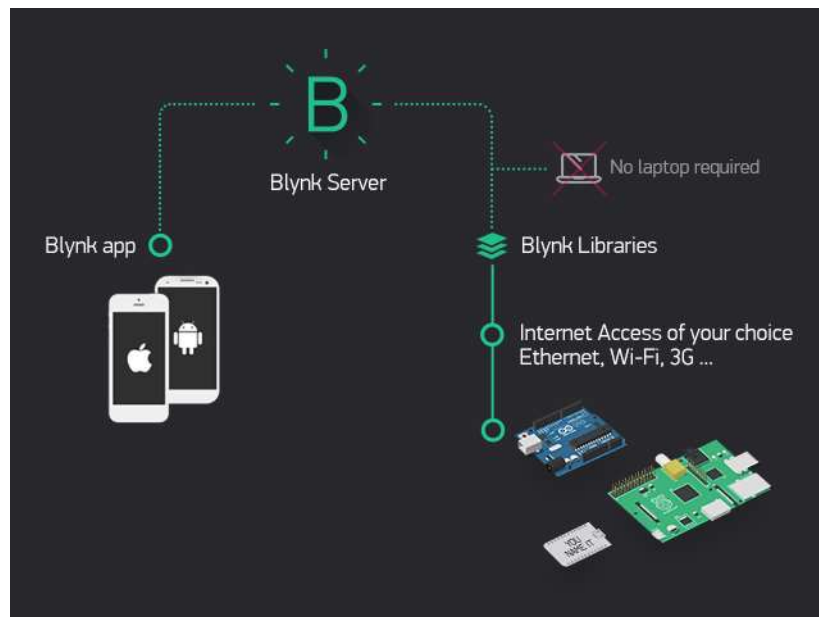


Picture 38. Libraries Blynk to connect any hardware

Advantages	Disadvantages
Really like the customizability and the ease of use.	The only smartphone app, no internet interface.
Simple and easy to use. Support for almost all hardware	Nothing maybe just a little bit expensive, but not at all
Very handy and easy to use, even for non-skilled developers	The iOS app is not so pretty or adjusted to GUI standards.
Ease of use was the main thing I was up and running in no time	The free trial is a little too much limiting in my opinion.
	You cannot monitor in your browser (only if you have your own blynk server)

Table 1. The advantages and disadvantages of Blynk

**Example:** Now imagine every time you press a Button in the Blynk app, the message travels to space the Blynk Cloud, where it magically finds its way to your hardware. It works the same in the opposite direction and everything happens in a blynk of an eye.



Picture 39. How blynk work

### What does the User need to Blynk?

- **Hardware:** An Arduino, Raspberry Pi, or a similar development kit.
- **A Smartphone:** The Blynk App is a well-designed interface builder. It works on both iOS and Android. (blynk, n.d.)

#### 4. Apply the selected techniques to create an IoT application development plan

##### 4.1. Initiating phase

### PROJECT CHARTER

1. General Project Information				
Project Name:		GreeParking		
Executive Sponsors:		Tran Quang Huy & Head of R&D department		
Department Sponsor:		R&D		
Impact of the project:		Technology solutions to help parking car in the street		
2. Project Team				
	Name	Department	Telephone	E-mail
Project Manager:	Trần Quang Huy	R&D	0795541090	<a href="mailto:tranquanghuy@gmail.com">tranquanghuy@gmail.com</a>
Team Members:	Huỳnh Thái Hiếu	R&D	0124579852	<a href="mailto:huynhthaihiu@gmail.com">huynhthaihiu@gmail.com</a>
	Nguyễn Hà Kiều My	R&D	0567489142	<a href="mailto:nguyenhakiemy@gmail.com">nguyenhakiemy@gmail.com</a>
	Dương Minh Phúc	R&D	0564897456	<a href="mailto:minhphuc@gmail.com">minhphuc@gmail.com</a>
	Lê Hạnh Dung	R&D	0697812313	<a href="mailto:hanhdung@gmail.com">hanhdung@gmail.com</a>
	Nguyễn Quang Ngọc	Marketing	0798411320	<a href="mailto:quangngoc@gmail.com">quangngoc@gmail.com</a>
3. Stakeholders				
R&D department				
HR department				
Business department				
University of Greenwich (DN Campus)				
4. Project Scope Statement				
Project Purpose / Business Justification				
GreeParking project helps support people in parking cars on the street thereby reducing people's time and traffic jams.				



<b>Objectives (in business terms)</b>	
<ul style="list-style-type: none"><li>- Reduce the “finding” time about 10-15 mins.</li><li>- Bring new experiences to people.</li><li>- Reduce the traffic jam</li></ul>	
<b>Deliverables</b>	
<ul style="list-style-type: none"><li>- Apply IoT into the system</li><li>- Monitoring by LCD or application on mobile phone</li></ul>	
<b>Scope</b>	
<ul style="list-style-type: none"><li>- Applicable to the main street has a problem about parking.</li><li>- Technologies: IoT Sensor, Cloud, Rasa core (NLP English, Vietnamese), Java, C language.</li></ul>	
<b>Major Known Risks (including significant Assumptions)</b>	
<b>Risk</b>	<b>Risk Rating (Hi, Med, Lo)</b>
Slow progress	High
Team member’s attitude & abilities	Medium
Out of budget	High
Technical problems	High
<b>Constraints</b>	
<ul style="list-style-type: none"><li>- Limit budget for implement project.</li><li>- Working between departments is difficult.</li></ul>	
<b>External Dependencies</b>	
There is an agreement between the stakeholder and project team	
<b>5. Communication Strategy</b>	
<ul style="list-style-type: none"><li>- Update progress to team leader every day.</li><li>- Team leader report to the project manager every week.</li><li>- Keep track of milestones.</li></ul>	



6. Sign-off			
	Name	Signature	Date (MM/DD/YYYY Y)
Executive Sponsor			
Department Sponsor			
Project Manager	Tran Quang Huy	<b>Huy</b>	

Table 2. Project Charter



## 4.2. Planning phase

### 4.2.1. Project work breakdown structure (WBS)

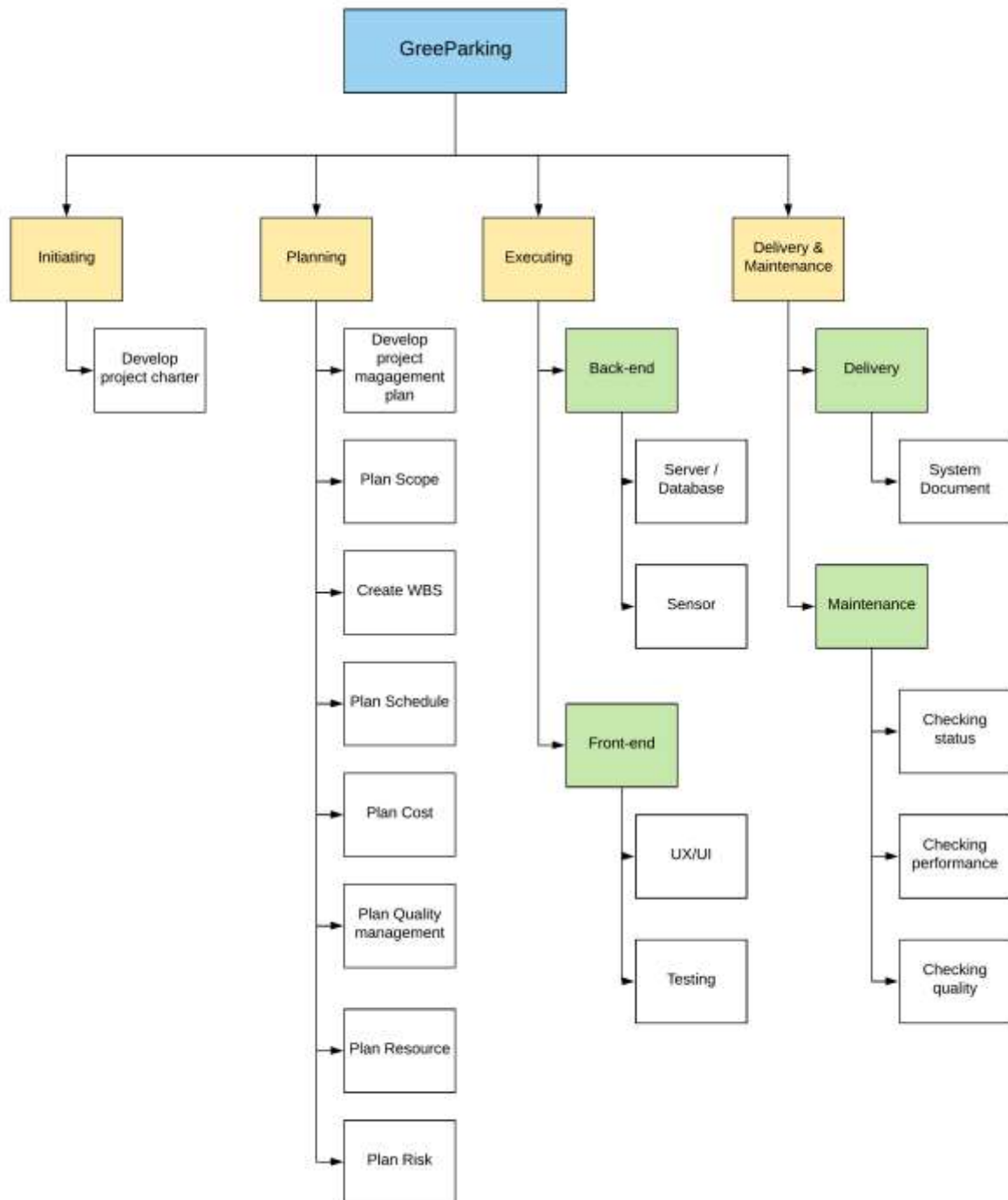


Figure 14. Work Breakdown Structure

#### 4.2.2. Project cost

Phase	Process	Members	Work hours	Cost per hour	Total
Initiating					
	Develop project Charter	5 member	24 hours	\$5	\$600
Planning					
	Develop project management plan	4 members	24 hours	\$5	\$480
	Plan Scope	3 members	16 hours	\$5	\$240
	Create WBS	2 members	12 hours	\$5	\$70
	Plan Schedule	1 members	72 hours	\$5	\$720
	Plan Cost	2 members	20 hours	\$5	\$300
	Plan Quality management	3 members	18 hours	\$5	\$270
	Plan resource	3 members	10 hours	\$5	\$150
	Plan Risk	3 members	24 hours	\$5	\$360
Executing					
Back-end	Server	2 members	130 hours	\$6	\$1.560
	Database	3 members	160 hours	\$6	\$2.880
	Sensor	2 members	120 hours	\$5	\$1.200
Front-end	UX/UI	2 members	40 hours	\$5	\$400
	Testing	3 members	18 hours	\$4	\$216
Delivery & Maintenance					
Delivery	System Document	4 members	100 hours	\$3	\$1.200
Maintenance	Checking status	1 members	40 hours	\$4	\$120
	Checking performance	1 members	40 hours	\$4	\$120
	Checking quality	2 members	60 hours	\$5	\$600
Total					\$11.486

Table 3. Project cost

### Equipment for prototype:







Item	Description	Quantity	Cost	Total
Arduino Uno R3		1	\$3.6	\$3.6
Infrared sense		7	\$0.9	\$6.3
Nodemcu esp8266 wifi module		1	\$3.8	\$3.8
Servo SG90		1	\$1.6	\$1.6
RFID RC522		1	\$1.9	\$1.9
LCD I2C 16x2		1	\$1.8	\$1.8
<b>Total</b>				<b>\$19</b>

Table 4. Equipment cost for prototype

#### 4.2.3. Project risk

Risk categories	Risk	Responsible by	Risk rating
<b>Technical</b>	Requirements	Project team	Medium
	Technology	Technical team	High
	Interfaces	Developer team	Medium
	Performance	Project team	High
	Quality	Project team	High
<b>External</b>	Customer	Project manager	Medium
	Contract	Project manager	Medium
	Market	Marketing team	High
	Supplier	Project team	Low
<b>Organizational</b>	Project Dependencies	Project team	Medium
	Logistics	Project team	Medium
	Resources	Project manager	Medium
	Budget	Project manager	High
<b>Project management</b>	Planning	Project team	Medium
	Schedule	Project team	Medium
	Estimation	Project manager	High
	Controlling	Project manager	Medium
	Communication	Project team	Low

Table 5. Project risk

### **4.3. Executing phase**

The implementation of the project consists of two main parts: back-end and front-end to create a complete product with the required features of the product.

### **4.4. Delivery and maintenance phase**

The last part of the project is to maintain and develop products that operate in the best way to serve customers in the best way.

5. The multiple iterations of your IoT application and modify each iteration with enhancements gathered from user feedback and experimentation.

#### 5.1. GreeParking survey form

In order to meet the user's requirement of Greeparking, the survey tools were applied, through interviewing and monitoring customer behavior, the questionnaire was used as the best option to meet the business requirements. Below is a sample survey:

## GreeParking

Thank you for using our system. We hope you had as much fun attending as we did organizing it.

We want to hear your feedback so we can keep improving our logistics and content. Please fill this quick survey and let us know your thoughts (your answers will be anonymous).

\*Required

How satisfied were you with the system? \*

	1	2	3	4	5	
Not very	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very much

Do you think Greeparking will solve the parking problem? \*

	1	2	3	4	5	
Not very	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Very much

What were your key take aways from this system?

Your answer

Additional feedback on system? \*

Your answer

Which part did you find most relevant? \*

	Not relevant	Relevant	Very relevant	Did not attend
Sensor	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Server	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Database	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Application	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Any overall feedback for the system?

Your answer

SUBMIT

Never submit passwords through Google Forms.



## 5.2. Analysis of user feedback and experimentation

Through the questionnaire sent to 55 users who experienced the GreeParking system, there were 52 users who answered the questionnaire and had 50 questionnaires valid. General statistics on the purpose achieved after using the question system are as follows:

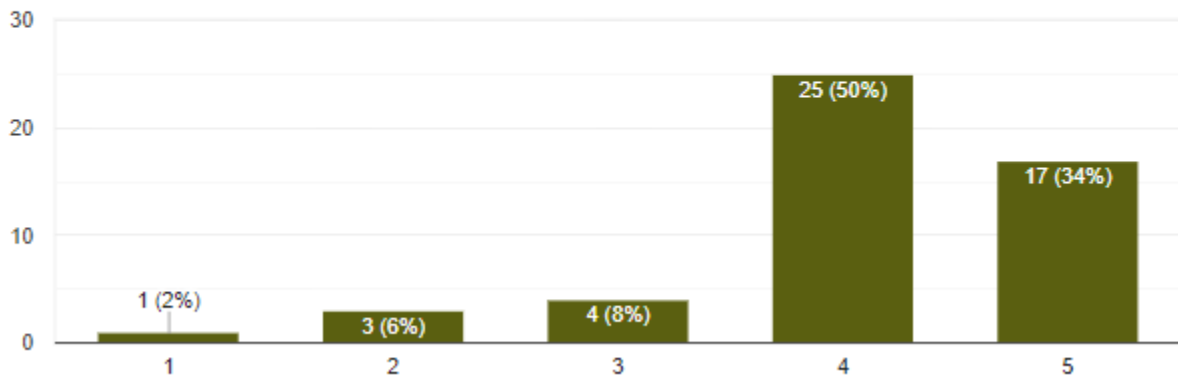
This IoT is a comprehensive system that includes hardware and software. It is important to identify key stakeholders:

- Customer: Does using the application meet the needs of customers? Is the user happy with the product or has related issues when using?
- Developer: Has there been any improvement in the project implementation, has the hardware and software functions been stabilized? Need to improve or change any hardware or software?
- Related people: Providers of software and hardware services

### The result of the survey:

#### How satisfied were you with the system?

50 responses



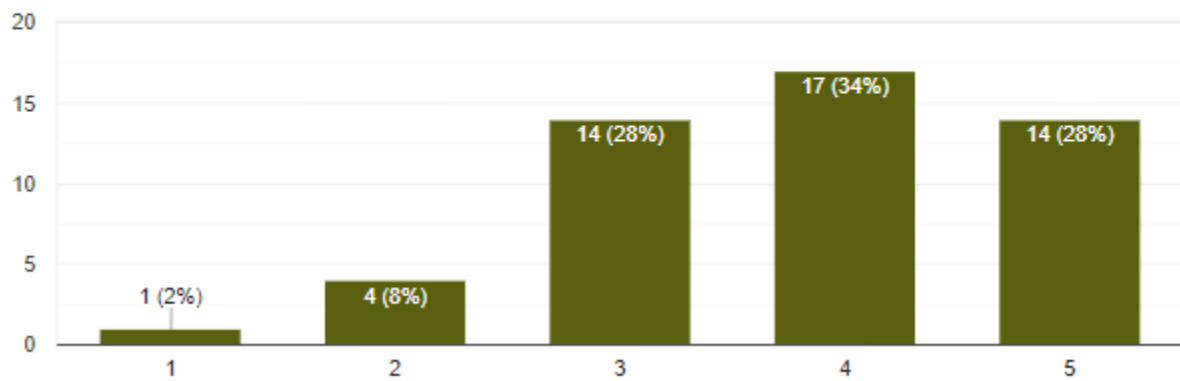
Picture 40. User satisfied result

Most survey participants have a good review when experiencing the GreeParking system.

## Do you think Greeparking will solve the parking problem?



50 responses

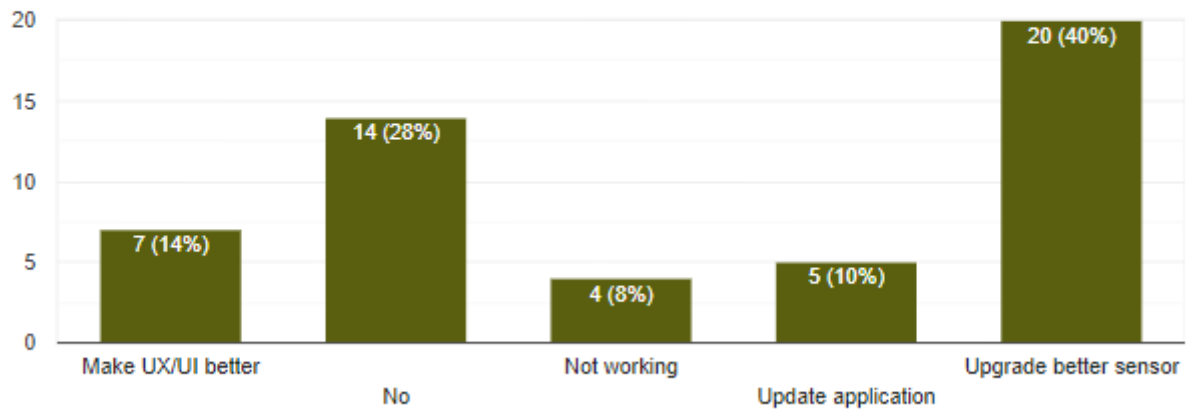


Picture 41. Result of User's thinking this project will solve the problem

Regarding the problem of parking, users have felt how the system works to solve the problem, but besides that, there are still unsolved issues.

## Additional feedback on system?

50 responses

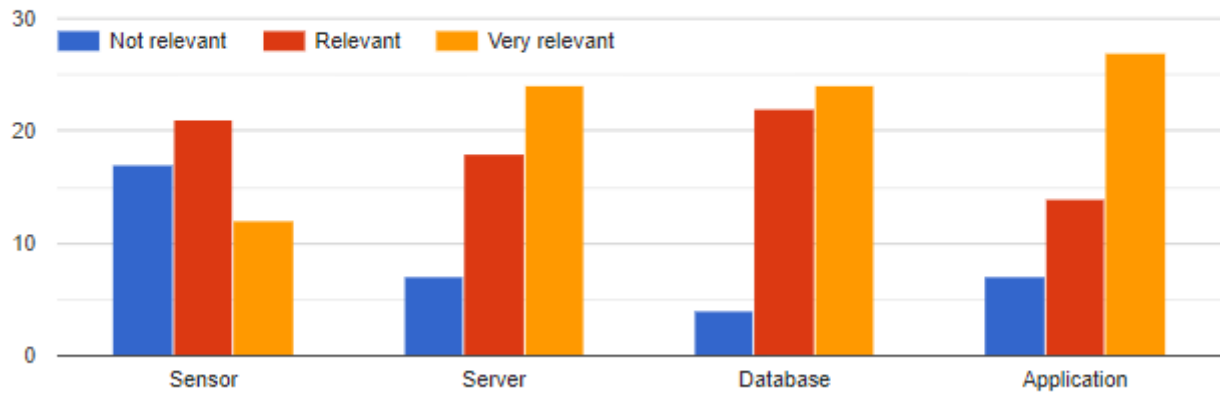


Picture 42. Feedback on system

Based on user feedback, the system needs to upgrade sensors and applications, and improve UX / UI so that users can use it more easily.

- The IR sensor used by the project is not guaranteed to be accurate. Instead of using Infrared (IR) Sensors, the next version of GreeParking will use Ultrasonic.
- Software and UX/UI to operate the system is used through the Blynk platform, so it has not yet provided a complete application to the user. The next version of GreeParking will create an application for Android phones
- The system transmits data via wifi network, because of that in some cases due to internet problems, the system will not work properly. In the next version, new hardware supporting the 3G network will be applied.

## Which part did you find most relevant?



Picture 43. Which part did the user find the most relevant

Survey results show that: Server, Database and application are self-compatible with the system. Only the sensor that people think needs to be changed does not match this system.

### Summary:

Version	Problem	Solution
Greeparking 1.0	Sensor	Change from Infrared (IR) Sensors to Ultrasonic.
	Application	Change application from Blynk to Greeparking application on Android.
	Networking	Change module Nodemcu esp8266 wifi module to Board SIM800C into Robot Fox.
Greeparking 2.0	Sensor	Upgrade Waterproof sensor.
	Application	Create application for IOS phone.
	Database	Upload data on Greeparking's cloud.
	Networking	Testing using LoRa to transfer data.
Greeparking 3.0	Energy	Use solar energy

Table 6. Summary of Greeparking version

## CONCLUSION

In a nutshell, IoT is a concept that connects all the devices to the internet and let them communicate with each other over the internet. IoT is a giant network of connected devices – all of which gather and share data about how they are used and the environments in which they are operated.

Through this report, point out the following:

- Analyze what aspects of IoT are necessary and appropriate when designing software applications: Plan an IoT application for a specific target end-user and the tests you intend to conduct with this user. This plan will be in the form of a document and will include supporting evidence and material, such as user personas and customer journey maps.
- Outline a plan for an appropriate IoT application using common architecture, frameworks, tools, hardware, and APIs: Create multiple iterations of your application and modify each iteration with enhancements gathered from user feedback and experimentation. This will follow the pathway outlined in your plan.

The IoT has the potential to dramatically increase the availability of information and is likely to transform companies and organizations in virtually every industry around the world.

As such, finding ways to leverage the power of the IoT is expected to factor into the strategic objectives of most technology companies, regardless of their industry focus.

The number of different technologies required to support the deployment and further growth of the IoT places a premium on interoperability and has resulted in widespread efforts to develop standards and technical specifications that support seamless communication between IoT devices and components. Collaboration between various standards development groups and the consolidation of some current efforts will eventually result in greater clarity for IoT technology companies.

## References

- 42gears. (n.d.). *42gears*. Retrieved from 42gears: <https://www.42gears.com/blog/how-iot-is-changing-retail-industry/>
- arduino. (n.d.). *arduino*. Retrieved from arduino: <https://create.arduino.cc/projecthub/156471/smart-health-care-monitoring-system-based-on-iot-f559b3>
- avancer. (n.d.). *avancer*. Retrieved from avancer: <http://avancer.in/history-iot-internet-things-changed-today/>
- avsystem. (n.d.). *avsystem*. Retrieved from avsystem: <https://www.avsystem.com/blog/what-is-iot-architecture/>
- blynk. (n.d.). *blynk*. Retrieved from blynk: <https://docs.blynk.cc/>
- dzone. (n.d.). Retrieved from dzone: <https://dzone.com/articles/everything-you-need-to-know-about-iot-hardware>
- flair. (n.d.). *flair*. Retrieved from flair: <https://data-flair.training/blogs/iot-hardware/>
- google. (n.d.). *google*. Retrieved from google: <https://cloud.google.com/community/tutorials/cloud-iot-gateways-rpi>
- isaax. (n.d.). *isaax*. Retrieved from isaax: <https://camp.isaax.io/en/isaax-examples/api-network-with-zetta>
- justcreative. (n.d.). *justcreative*. Retrieved from <https://justcreative.com/2018/11/19/internet-of-things-explained/>
- koreabiomed. (n.d.). *koreabiomed*. Retrieved from koreabiomed: <http://www.koreabiomed.com/news/articleView.html?idxno=3975>
- mathworks. (n.d.). *mathworks*. Retrieved from mathworks: <https://blogs.mathworks.com/iot/2014/05/06/connecting-electric-imp-to-thingspeak-iot-data-services/>
- medium. (n.d.). *medium*. Retrieved from medium: <https://medium.com/@tpponmat/azure-iot-with-python-5a00359c3d54>
- medium. (n.d.). *medium*. Retrieved from medium: <https://medium.com/mainflux-iot-platform/mainflux-open-source-iot-platform-set-up-and-usage-70bed698791a>
- newgenapps. (n.d.). *newgenapps*. Retrieved from newgenapps: <https://www.newgenapps.com/blog/8-uses-applications-and-benefits-of-industrial-iot-in-manufacturing>
- pantechsolutions. (n.d.). *pantechsolutions*. Retrieved from pantechsolutions: <https://www.pantechsolutions.net/iot-based-intelligent-traffic-management-system>
- publictechnology. (n.d.). *publictechnology*. Retrieved from publictechnology: [https://www.publictechnology.net/articles/partner\\_article/5-ways-iot-transforming-public-sector](https://www.publictechnology.net/articles/partner_article/5-ways-iot-transforming-public-sector)
- sam-solutions. (n.d.). *sam-solution*. Retrieved from sam-solution: <https://www.sam-solutions.com/blog/automotive-internet-of-things-the-iot-connected-car/>
- scnsoft. (n.d.). *scnsoft*. Retrieved from scnsoft: <https://www.scnsoft.com/blog/iot-architecture-in-a-nutshell-and-how-it-works>

techrepublic. (n.d.). *techrepublic*. Retrieved from techrepublic: <https://www.techrepublic.com/article/ciscos-iot-system-addresses-the-need-to-connect-billions-of-devices/>

techtipsnapps. (n.d.). *techtipsnapps*. Retrieved from techtipsnapps: <http://www.techtipsnapps.com/2019/02/iot-based-transportation.html>

thecattlecrew. (n.d.). *thecattlecrew*. Retrieved from thecattlecrew: <https://thecattlecrew.net/2017/07/19/thoughts-about-using-api-in-iot-scenarios/>

themanufacturer. (n.d.). *themanufacturer*. Retrieved from themanufacturer: <https://www.themanufacturer.com/articles/applications-iot-manufacturing-plants/>

vdoo. (n.d.). *vdoo*. Retrieved from vdoo: <https://www.vdoo.com/blog/integrating-security-into-the-iot-sdlc/>

wikipedia. (n.d.). *wikipedia*. Retrieved from [https://en.wikipedia.org/wiki/Internet\\_of\\_things](https://en.wikipedia.org/wiki/Internet_of_things)

youtube. (n.d.). Retrieved from youtube: <https://www.youtube.com/watch?v=wL5vXuD-3nw>