Smart IHU: an open source IoT project for Wireless Sensor Network based on the Raspberry platform – Web and mobile development

Georgios Sarigiannis

SID: 330717004

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Communications and Cybersecurity

October, 2019

THESSALONIKI – GREECE
Smart IHU: an open source IoT project for Wireless Sensor Network based on the Raspberry platform – Web and mobile development

Georgios Sarigiannis

SID: 330717004

Supervisor: Prof. Stavros Stavrinides

Supervising Committee Members:

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Communications and Cybersecurity

October, 2019

THESSALONIKI – GREECE
Acknowledgements

I would like to express my deepest appreciation to my supervisor, Dr. Stavros G. Stavrinides for his continuous support and guidance throughout the project.

I would like to thank my colleague Kostas Karaklas for his insight and dedication to the first part of the project. His extensive knowledge in matters related to hardware proved invaluable.

A thank you to the International Hellenic University for supporting this project financially and was kind enough to buy the required materials for the sensors and the nodes. I would also like to thank the IHU IT Department that was kind enough to set-up the VM server and help us determine the optimal locations around campus for the placement of the sensors.

Special thanks to my coworker and long-time friend George Fotos for his moral support as well as his tips and tricks regarding the development of the mobile app.

Finally, I would like to thank my family for their support in every step of the way, especially my brother Dr. Denis A. Sarigiannis, whose example I strive to follow.
Abstract

This dissertation was written as a part of the MSc in Communications and Cybersecurity at the International Hellenic University.

The Internet of Things has slowly but surely become an important part of every-day life for millions of users around the world and is a technology that is constantly evolving and adapting. This dissertation is essentially a project that aims to set up a wireless sensor network at the International Hellenic University Campus with the use of four (4) sensor nodes.

The nodes are based on the Raspberry Pi platform and their main function is to retrieve various forms of data such as weather conditions, noise levels, pictures and more. The data are then stored in a database created in a virtual server and afterwards displayed on a webpage built with Wordpress and a mobile application developed with Google’s Android Studio.

Georgios Sarigiannis

07/12/2018
**Contents**

ACKNOWLEDGEMENTS ........................................................................................................ III

ABSTRACT .............................................................................................................................. IV

CONTENTS ................................................................................................................................ V

1 INTERNET OF THINGS: AN INTRODUCTION ................................................................... 1

1.1 ABOUT THE INTERNET OF THINGS ........................................................................ 1

1.2 WIRELESS SENSOR NETWORKS (WSN) ................................................................. 3

  1.2.1 Star Network .................................................................................................. 4

  1.2.2 Mesh Network ............................................................................................... 5

  1.2.3 Hybrid Star – Mesh ....................................................................................... 5

1.3 SMART IHU .............................................................................................................. 6

1.4 PROJECT PLANNING .............................................................................................. 6

  1.4.1 Hardware specifications .................................................................................. 7

  1.4.2 The Virtual Server ........................................................................................ 7

  1.4.3 Software Specifications .................................................................................. 7

2 HARDWARE SPECIFICATIONS ....................................................................................... 9

2.1 THE RASPBERRY PI ............................................................................................... 9

  2.1.1 History ........................................................................................................ 9

  2.1.2 The Raspberry Pi Model 3B+ ...................................................................... 10

2.2 NODE ANALYSIS .................................................................................................... 11

  2.2.1 Node #1 – Weather Station ........................................................................ 11

  2.2.2 Node #2 – Internal Weather Node ............................................................... 13

  2.2.3 Node #3 – Campus entrance node ............................................................... 14

  2.2.4 Node #4 – The “Smart Mirror” ................................................................... 14

2.3 NODE PLACEMENT & TOPOLOGY .......................................................................... 15

  2.3.1 Node placement ............................................................................................ 15

  2.3.2 Topology ....................................................................................................... 16
3 THE VIRTUAL SERVER

3.1 UBUNTU LINUX 18.04

3.2 APACHE HTTP SERVER

3.3 MySQL

3.4 PHP

3.5 SETUP OF THE LAMP STACK

3.5.1 Installation of Apache

3.5.2 Installation of MySQL

3.5.3 Installation of PHP

4 WEB DEVELOPMENT

4.1 WEB DEVELOPMENT PROCESS

4.1.1 Information Gathering

4.1.2 Wireframe creation - Sitemap

4.1.3 Page layouts / Review / Approval

4.1.4 Content writing and management

4.1.5 Coding

4.1.6 Testing, Review and Launch

4.1.7 Maintenance and updates

4.2 DEVELOPMENT TOOLS

4.2.1 Phpstorm

4.2.2 phpMyAdmin

4.2.3 XAMPP

4.2.4 Wordpress

4.3 CODING PREPARATION

4.3.1 XAMPP Setup & Configuration

4.3.2 phpMyAdmin / database setup

4.3.3 Wordpress Setup

4.4 CODE REVIEW

5 MOBILE DEVELOPMENT

5.1 CODE REVIEW

6 INSPIRATIONAL PROJECTS

6.1 DUT – “SMART UNIVERSITY CAMPUS”
1 Internet of Things: An introduction

This chapter briefly describes the evolution of the Internet of Things (IoT) and its applications while attempting to properly define it. It also provides a brief overview of the Wireless Sensor Networks (WSN).

1.1 About the Internet of Things

Modern technologies, especially those of the information and communication sector are changing the living and working conditions and environments of the people in industrialized nations at an increasingly rapid rate. 15 years ago mobile phones were still in the market, but today they have become a product that is massively produced, a product that combines more and more capabilities and functions. Mobile telephony, internet access, satellite navigation, television, music and much more can now be found in a single device.

But apart from mobile phones, many other devices and products now have embedded logical functions and components that greatly increase their function range. For these devices, it has become increasingly possible, with the integration of sensory components, to interact with their surroundings, communicate with control units or with other objects via wired or wireless networks in order to exchange information.

Technological advancements in several areas such as network technology, electronics and sensors have laid out the foundations for the use of integrated knowledge or “technical intelligence” in devices that are now considered highly valuable, such as phones, televisions, gaming consoles, etc.

However, this is bound to change in the near future, as the rapid development of the Internet of Things could lead to its integration in many cheap, every-day products and low-value consumer goods thus making it way more popular. The Radio Frequency Identification (RFID), which is the main technology behind this, makes it possible to
turn low-cost products into “smart” objects. Future developments in the area of electronics have the potential to make the cost of RFID tags drop even less, thus making the technology more accessible.

Along with such developments, the Internet has become a global medium; a source of ever-present information, communications and entertainment, the likes of which could not be foreseen in the previous century. But again, even the Internet seems to be changing.

The Internet of today is the “Internet of Persons” or the “Internet of Information”. It gives people the ability to connect and communicate with each other in many different ways in order to exchange information. However, the addition and use of “smart” objects in networking and information technology along with their associated control systems leads to the emergence of new types of networks.

These networks are in many ways comparable to the Internet; however, they now connect more than people; inanimate objects have entered the equation and thus the term “Internet of Things” became a reality.

Apart from networking, another important feature of the Internet of Things is its autonomy. Technological developments in several areas have enabled objects to act in an autonomous or semi-autonomous way, for example in engineering, medicine, infrastructure, household operations and more.

It is widely recognized that the concept of the Internet of Things was initially proposed by Kevin Ashton in 1999 [1]. Greengaard traces the origin of the term back to the development of the LAN networks [2].

Kevin Ashton initially observed that the majority of the data on the Internet was handled by human beings. So essentially, a human user is a slow, inefficient router of data, prone to errors. The user also limits the quality and quantity of said data while sometimes he attempts to interpret it or correct it. An alternative to this approach would be to have these systems connect to sensors that make real-world measurements of events and properties directly. Essentially, systems rid themselves of the need for a human intermediary and attempt to make a direct connection to the internet in order to directly capture real-world data.

However, this idea does not constitute a definition but an observation and an important one at that. For if systems are able to access data directly from sensors, these data will most likely be more in number and less prone to errors. This field of study is known and
worked on for decades; the Sensor Networks. So what is the real difference between the IoT and M2M or D2D communication, or even CPS? Under this basis, Peter Wahe suggests the following definition [3]:

“The Internet of Things is what we get when we connect Things, which are not operated by humans, to the Internet.”

The Internet of Things differs from sensor networks in the sense that neither Things need to be sensors, nor do sensor networks need a connection to the Internet. The Internet of Things is also different from Big Data since Things do not necessarily capture or generate data, nor is it necessary for data to be stored in big data stores located in the Cloud. The Internet of Things is also different to M2M in the sense that users also can and want to access these Things too.

1.2 Wireless Sensor Networks (WSN)

Wireless Sensor Networks can be defined as self-configured and infrastructure-less wireless networks to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants and to cooperatively pass their data through the network to a main location or sink where the data can be observed and analyzed.

The network and its users make use of a base station as an interface, while information retrieval can occur via query injection and data gathering directly from the base station. Wireless Sensor Networks can comprise of a few or even more, up to thousands of nodes which communicate with each other via radio signals. Wireless sensors consist of sensing and computing devices, radio transceivers and power components.

Every node in a Wireless Sensor Network has limited resources: processing speed, disk capacity and network bandwidth. After deployment, the nodes self-organize an appropriate network infrastructure and the sensors begin the collection of data.

Wireless Sensor Networks have become extremely popular due to their flexibility and their potential for solving many different problems; they have been successfully used in the following domains:

- Military applications
Wireless Sensor Networks have different topologies, briefly explained below:

1.2.1 **Star Network**

In the star topology [4], each node connects directly to a base station. A gateway can send or receive messages to and from a number of nodes. Nodes are not allowed to communicate directly with one another. This enables the use of low-latency communications between a node and the base station. Since this topology enables a single node to handle the whole network, the base station must be within range of all individual nodes. It also helps keep the nodes’ energy consumption to a minimum and is popular because of its simplicity, although it severely lacks in terms of robustness due to a single node managing the network.

![Star Network Topology](image)
1.2.2 Mesh Network

Mesh networks [4] allow communication between nodes that are within transmission range. If a node wants to send a message to a node that is located outside of transmission range, then it needs to use another node as an intermediary. The intermediary node forwards the message to the intended node. The advantage of this topology lies in the fact that its range can be extended by simply adding more nodes, while the fact that nodes communicate with each other makes it easier to isolate a node in the event it fails. It also allows for easier error detection. Disadvantages include the large cost (network scalability can make ultimately lead it to being huge), the high energy consumption of nodes that make use of multi-hop communications and the increased time it takes a message to be delivered depending on the number of hops required.

![Picture 2: Mesh Network Topology]

1.2.3 Hybrid Star – Mesh

A combination of the star and mesh networks [4], the hybrid provides many advantages in terms of versatility, robustness and low power usage. In this topology the nodes with the lowest power consumption do not possess the ability to forward messages while other nodes have the ability to multi-hop, thus being allowed to forward messages from low power nodes to the rest of the network.
The goal of this project is to setup a Wireless Sensor Network that includes four sensor nodes placed around the campus of the International Hellenic University in a star network topology. The nodes collect sensor readings and communicate with a virtual server where the data is stored. The data are ultimately displayed on a webpage created for this purpose, as well as a mobile app.

The main aim is to demonstrate that the Internet of Things should become an integral part of every higher education institute and that the data the sensors provide can be of use to staff, students, and academics alike. It should be noted that by the time this project is finished, the International Hellenic University will be one of the few Institutes in Greece to be making use of IoT technology in its campus.

1.4 Project planning

After consulting with colleague Kostas Karaklas, the planning for the project was divided into three parts:
1.4.1 Hardware specifications

This part of the project was carried out by Kostas Karaklas with the assistance of the IT Department. It includes the studying of the various parts of the nodes, their assembly and their installation at distinct locations around the campus. Kostas also did the programming on the nodes and their services; the nodes store their data in MicroSD cards and via use of a service, send the data to the server in short intervals, depending on the sensor and the node. The IHU IT Department was responsible for setting up the Virtual Server and provide us with credentials to be able to access it via the IHU VPN.

1.4.2 The Virtual Server

After the Virtual Server was setup, the next step was to install the necessary components on it. Via use of the so-called “LAMP” stack, the server ended up running on Ubuntu Linux 18.04 with the Apache2 HTTP server, as well as MySQL and PHP version 7.2.

1.4.3 Software Specifications

The webpage was created with the use of Wordpres, one of the world’s most popular CMS, based on PHP and MySQL. It also makes use of the Bootstrap4 theme and a plugin was written in PHP in order to get the data from the sensors and display them on the page. The mobile application was programmed via the use of Android Studio, the
official IDE for Google’s Android OS. The application was programmed in Java. As of the moment, an iOS version of the app is not planned for the near future.
2 Hardware specifications

This chapter describes the devices that were used for this project, their specifications, setup and functionality. After discussing this topic extensively with Prof. Stavros Stavrinides and colleague Kostas Karaklas, a conclusion was reached that the most time- and cost-efficient technology to use as a basis for this project was the Raspberry Pi Model 3B+. The chapter also describes the nodes and their respective sensors briefly. This part of the project is covered extensively by colleague Kostas Karaklas in his dissertation, titled “Smart IHU: an open source IoT project for wireless sensor network based on the Raspberry PI platform – Physical design and implementation”.

2.1 The Raspberry Pi

2.1.1 History
The Raspberry Pi (or, in short, RPi) is a series of small, portable devices that are classified as “single-board” computers; in other words, a computer that is built entirely on one single circuit board. The concept of the Raspberry Pi was conceived by the Raspberry Pi Foundation, a charitable organization that was founded in the United Kingdom in 2009. Its goal was to promote the teaching of the basics of Computer Science in low- and middle-income countries. The mind behind the conception of the RPi was Eben Upton, former CEO of the Raspberry Pi Foundation and Raspberry Pi (Trading), who at the time was working as director of studies at the Computer Science Department at the University of Cambridge’s St. John’s College. Upton, along with Jack Lang, an affiliated lecturer at the Computer Laboratory of Cambridge and Alan Mycroft, a Computing Professor at the same institution, became concerned when they realized that the students’ interest in computer science was waning, while the majority of the applicants of these institutions did not possess adequate programming skills. And so the RPi Foundation developed a model that was cheap, portable and ultimately, powerful enough to become popular, with the hope that more young people and future students would take an interest for computer science.
The full list of the original founders is as follows:

- Eben Upton
- Pete Lomas
- Robert Mullins
- Alan Mycroft
- David Braben
- Jack Lang

The current chairman of the Raspberry Pi Foundation is David Cleevely; the charity employs 135 people out of Cambridge, UK.

2.1.2 **The Raspberry Pi Model 3B+**

The Raspberry Pi Model 3B+ [5] features a quad-core CPU (Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC), along with 1GB of LPDDR2 SDRAM. It also comes equipped with 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE, as well as Gigabit Ethernet over USB 2.0 with a maximum throughput of 300Mbps. The device has 4 USB 2.0 ports, a CSI camera port for connecting a Raspber-
ry Pi Camera, a DSI display port for connecting a Raspberry Pi touchscreen display and a Micro SD port for loading the operating system and storing data. It also offers Power-over-Ethernet support (PoE) via the use of a separate PoE HAT.

![Raspberry Pi Model 3B+ with attached HAT](image)

**Picture 6: Raspberry Pi Model 3B+ with attached HAT**

## 2.2 Node Analysis

### 2.2.1 Node #1 – Weather Station

The first and most complex node of the project essentially functions as a weather station, gathering various type of data. It is equipped with multiple sensors and its housing features an electronic lock which requires the use of a RFID card for access. The node is housed in a Stevenson screen [6] (also known as “instrument shelter”) which shelters the sensors from precipitation and heat radiation whilst allowing air to freely circulate around them. It is built out of wood and is painted white in order to reflect direct solar radiation. Some of the more sensitive sensors are stored within an IP65 rated (IEC standard 60529) plastic box which provides protection against dust and moisture. The full list of components can be found below:

- Raspberry Pi Model 3B+
- Power Supply for the Raspberry Pi / 5V – 2.5A
- MicroSD card - 64 GB. Required for the operating system and applications of the Raspberry Pi
- Adafruit Perma-Proto Full-sized Breadboard [7] - PCB used for the connections
- Adafruit Perma-Proto Half-sized Breadboard [8] - PCB used for the connections
- Adafruit BME680 [13] – sensor that measures temperature, humidity, barometric pressure and VOC gas
- Adafruit TSL2591 [9] – an advanced digital light sensor, capable of measuring infrared, full-spectrum or human-visible light separately
- Anemometer [12] – capable of measuring up to 32.4 m/s wind speed.
- RGB LCD screen [10] – used for data display
- Lock-style Solenoid [11] – an electronic lock controlled by either the keypad or the RFID sensor
- Power supply for the Solenoid / 12V – 5A
- MFRC522 module [14] – RFID reader-antenna which is used to control the solenoid
- Magnetic contact switch – a switch used to alert the user in the event that the second door of the Stevenson Screen is opened

Picture 7: Sample pictures of the Weather Station components
2.2.2 Node #2 – Internal Weather Node

The internal weather node contains sensors that again measure weather data. It will be installed in a classroom in order to monitor the environmental conditions of the IHU interior and determine the air quality of the space. It also measures temperature and humidity. It will be housed in a custom, open-style box made out of wood and plexiglass. The full list of components can be found below:

- Raspberry Pi Model 3B+
- Power Supply for the Raspberry Pi / 5V – 2.5A
- MicroSD card - 64 GB. Required for the operating system and applications of the Raspberry Pi
- Adafruit BME680 [13] – sensor that measures temperature, humidity, barometric pressure and VOC gas
- Adafruit SPG30[15] – an indoor air quality monitoring sensor capable of detecting a wide range of Volatile Organic Compounds (VOCs). The sensor also calculates eCO2 values
- RGB LCD[10] screen – used for data display
2.2.3 Node #3 – Campus entrance node

The third node of the project is located near the campus entrance. It serves a dual purpose; Take pictures of the horizon and also counts the number of vehicles that enter or exit the campus. The components are housed in plastic boxes of various sizes, similar to the IP65 rated plastic boxes used for the weather station. The full list of components can be found below:

- Raspberry Pi Model 3B+
- Power Supply for the Raspberry Pi / 5V – 2.5A
- MicroSD card - 64 GB. Required for the operating system and applications of the Raspberry Pi
- Adafruit Perma-Proto Quarter-sized Breadboard [16] - PCB used for the connections.
- Raspberry Pi Camera [17] – 8MP camera used for the pictures
- 2 x IR Distance Sensor [18] – Two SHARP sensors used to determine how far away from them objects are. Good for short-range detection

2.2.4 Node #4 – The “Smart Mirror”

The final node of the project is an internal node that will be installed near the reception of the IHU’s building A. A smart mirror is a two-way mirror with an electronic display.
behind the glass. The display will be showing the viewer different kinds of information in the form of widgets, such as weather, time, date, etc. It will be housed in a custom wooden enclosure designed as a mirror with a frame. The full list of components can be found below:

- Raspberry Pi Model 3B+
- Power Supply for the Raspberry Pi / 5V – 2.5A
- MicroSD card - 64 GB. Required for the operating system and applications of the Raspberry Pi.
- PIR Sensor [19] – A motion sensor used to detect movement and trigger the node
- Adafruit Perma-Proto HAT [20] – A HAT PCB used for the connections
- 2-way mirror
- Monitor – A 23-inch monitor for the display of data

2.3 Node placement & topology

2.3.1 Node placement

By making use of the IHU infrastructure and after determining optimal positions for each node in order to maximize its efficiency, the nodes were placed as follows:

1. The weather station is placed near building B at an elevated position in order to get more accurate readings for the environmental data. Power supply and network coverage are possible via resources from building B.

2. The campus entrance node is placed to the left of the main entrance of the campus at an elevated position in order to be able to take better pictures of the horizon and also be able to provide accurate data regarding the vehicles entering and exiting the campus. Power supply and network coverage are possible via resources from building A.
3. The magic mirror is placed near the reception of building A after determining that this point is one with high traffic – most faculty, staff or students pass by daily.

4. The internal weather node is placed inside a classroom in building A.

![Image: Node placement as seen from above (IHU campus)]

At this point it should be noted that during the time this dissertation is being written, the “Magic Mirror” sensor is not functional due to the fact that the required parts to set it up have not arrived. As such, no further mention will be made for this node in this document.

### 2.3.2 Topology

Since this project makes use of IHU’s infrastructure, all nodes are connected to the university’s network, essentially in a star-network topology. The nodes make use of different access points that all lead to the same gateway, as shown in the diagram that follows.
Picture 12: Detailed diagram of the project
3 The Virtual Server

The virtual server for this project was set up by the IT department of the IHU. It runs the 64-bit version of Ubuntu Linux, and the shared resources include 2 GB of RAM and 30 GB of disk space. These values were deemed adequate for the current needs of the project, however they may be altered in the future. Afterwards, the Apache2 HTTP server was installed, along with PHP version 7.2, as well as MySQL. Thus, the so-called “LAMP stack”, the archetypal model of web service stacks was used for this project.

3.1 Ubuntu Linux 18.04

Ubuntu is one of many Linux distribution, and a relatively new one at that. Its source code originates from an older distribution known as Debian. Although Debian is still a popular distribution, its lack of frequent updates, user-friendly installation and difficult maintenance has made users turn to Ubuntu, which provides a much better overall user experience [21]. The following features made Ubuntu a much more desirable choice when it comes to picking an open-source operating system:

- Frequent and predictable release cycles
• Better localization and accessibility
• Ease-of-use; a user-friendly experience on desktop computers
• A desire to build a community-driven project

Although the LAMP stack clearly requires a Linux distribution, picking Ubuntu was the only real choice for this project. As the setup of the operating system was performed by the IHU IT Department, this dissertation will not go into any further details regarding this subject.

### 3.2 Apache HTTP Server

Apache is an open-source, free web server software. Created by the Apache Software foundation, it powers around 46% of websites around the world. It is considered to be one of the most reliable web servers and was originally launched back in 1995[22].

The task of a web server is to serve websites on the internet. In a way, a web server acts as a middleman between server and client machines. Upon user request, it delivers content from the server to the web. However, many users might request different pages at the same time and therein lies the biggest challenge for a web server; being able to serve multiple users at the same time.

Web servers also process files that are written in several different programming languages. They turn these files into static HTML files and serve them in each user’s browser. Essentially, a web server is the key to a successful server-client communication.

![Diagram: Example of a web server’s functionality](image)

**Picture 14: Example of a web server’s functionality**
Apache was the web server of choice for this project due to the many advantages it offers:

- It is an open-source software which is free even for commercial use
- It is stable and reliable
- The software is being updated on a regular basis
- Its module-based structure makes it extremely flexible
- It is easy to configure
- Works on different platforms (both on Unix and Windows servers)
- Works very efficiently with Wordpress sites
- Boasts a wide community that provides fast and effective support

Even though Apache comes off as a stable and versatile platform, it also comes with disadvantages that hinder its functionality when it comes to dealing with heavy-traffic webpages. Finally, its many configuration options can lead to security threats as the system is left vulnerable.

Since the project is rather small in terms of scale at the moment (prone to change in the future, a revision might be required) and the infrastructure and security measures of the IHU provide us with a safe network, Apache was the web server of choice.

### 3.3 MySQL

MySQL is a Database Management System (DBM). A database is a structured collection of data and in order to perform any kind of action such as access, add, or process data that is stored in a database, a DBM such as MySQL server is necessary.

MySQL databases are relational; the data is stored in separate tables and the structures are organized into physical files in order to promote speed optimization. The SQL standard has been evolving since 1986 [24] and as of today, several different versions exist. MySQL is open-source and thus serves the project’s needs; the software can be used and modified by anyone. Advantages of MySQL include speed, reliability, flexibility and ease-of-use. It is written in C and C++ and is available in many platforms, the most notable being Windows, Linux, Mac and Unix. In terms of security, MySQL makes use of an access privilege and an encrypted password system which enables host-based verification. Despite the fact that MySQL is an efficient and user-friendly DBM,
it is not suitable for projects that deal with large size data; as the data grows, so does its performance decrease and more complex queries may take a significant amount of time to be fulfilled. MySQL also has serious issues in terms of scalability. For larger projects, this presents an issue. MySQL was picked for this project due its small size, but this might need to be revised in the future.

3.4 PHP

PHP is an acronym for Hypertext Preprocessor. It is a scripting language that is used very often for server-side web development and can be embedded into HTML. Scripting languages (other examples include JavaScript and Ruby) are a set of programming languages that are used for the automation of processes that under different circumstances would need to be executed step-by-step in a webpage’s code every time they occur. As opposed to client-side languages (like Java), PHP code is executed on the server and the HTML generated is then sent to the client. Essentially the client-side receives the results of running a script but does not have access to the underlying code. PHP is equally suitable for beginners and professional programmers, being easy to learn but at the same time offering many advanced features. PHP is the main component of WordPress and its development, from static pages to themes and plugins [23]. For the needs of this project, it was the obvious choice, since a decision was made to develop the web page using WordPress.

3.5 Setup of the LAMP stack

After installing Ubuntu on the Virtual Server, the next step would be to install the rest of the open-source software that comprise the “LAMP” stack.

3.5.1 Installation of Apache

Apache was installed via the use of Ubuntu’s package manager, *apt*:

```bash
$ sudo apt update
$ sudo apt install apache2
```

Picture 15: Apache installation step #1
Sudo commands execute operations with root privileges. After the packages were installed, the next step would be to adjust the firewall in order to allow HTTP and HTTPS traffic. The UFW firewall was already enabled during the server setup, so a check should be made to confirm that a UFW profile was created for Apache.

```
$ sudo ufw app list
```

Output
```
Available applications:
Apache
Apache Full
Apache Secure
OpenSSH
```

Picture 16: Apache installation step #2

A check on the *Apache Full* profile should show that traffic is enabled to ports 80 and 443.

```
$ sudo ufw app info "Apache Full"
```

Output
```
Profile: Apache Full
Title: Web Server (HTTP,HTTPS)
Description: Apache v2 is the next generation of the omnipresent Apache web server.
Ports:
80,443/tcp
```

Picture 17: Apache installation step #3

In order to allow incoming HTTP and HTTPS traffic for the specific profile the `ufw allow` command is used:

```
$ sudo ufw allow in "Apache Full"
```

Picture 18: Apache installation step #4
In order for the installation process to be successful, the following page should be now showing in the browser:

![Default Ubuntu 18.04 Apache web page](image-url)

**Picture 19: Default Ubuntu 18.04 Apache web page**
3.5.2 Installation of MySQL

After finishing with the web server installation, the next step was installing MySQL. Again, as with Apache, the installation process begins with the use of the `apt` command:

```
$ sudo apt install mysql-server
```

Picture 20: MySQL installation step #1

After the installation of the packages is complete, a simple security script that comes pre-installed with MySQL was executed. It removes default settings that may cause security issues and locks down access to the database system:

```
$ sudo mysql_secure_installation
```

Picture 21: MySQL installation step #2

Running the script prompts the user to answer whether they want to configure the `VALIDATE PASSWORD PLUGIN`. This feature carries some degree of risk since this project makes use of the phpMyAdmin software. phpMyAdmin automatically configures MySQL user credentials and a weak password along with the use of this plugin may cause issues. Regardless, the plugin was used along with a strong password.

```
VALIDATE PASSWORD PLUGIN can be used to test passwords and improve security. It checks the strength of password and allows the users to set only those passwords which are secure enough. Would you like to setup VALIDATE PASSWORD plugin?

Press y|Y for Yes, any other key for No:
```

Picture 22: MySQL installation step #3

After pressing `y` to enable this feature, the system asks for the preferred level of password validation. In our case, the option `1 (MEDIUM Length)` was selected. After this selection the system asks for the selection and confirmation of a password for the MySQL root user. After selecting an appropriate password, since the `VALIDATE-`
PASSWORD PLUGIN was enabled, the next system message also shows the strength level of the password that was chosen:

![Using existing password for root. Estimated strength of the password: 100 Change the password for root? ((Press y|Y for Yes, any other key for No) : n]

Picture 23: MySQL installation step #4

The next step includes removing the test database and some anonymous users that are created by default, as well as remove root logins and load the new rules that were applied in the previous steps and are implemented on the spot. Since the MySQL version this project uses is older than version 5.7, the default user authentication happens with the use of the `auth_socket` plugin. Although this method is more secure, it may cause problems when accessing the user via the phpMyAdmin application. Since this is the case, the next action would be to change the authentication method to allow for the root user to be authenticated via password. In order to achieve this, the authentication method will be changed from `auth_socket` to `my_sql_native_password`. In order to do this, firstly the MySQL prompt is accessed.

![$ sudo mysql]

Picture 24: MySQL installation step #5

The next command checks the authentication method of the existing MySQL user accounts:

```
mysql> SELECT user,authentication_string,plugin,host FROM mysql.user;
```

Picture 25: MySQL installation step #6

The received output displays the current authentication method and confirms the fact that the `auth_socket` method is used for the authentication of the root user:
In order to change the authentication method, the `ALTER USER` command will be used:

```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password';
```

After choosing a strong password, the command `FLUSH PRIVILEGES` was used in order to reload the grant tables and bring the changes into effect:

```
mysql> FLUSH PRIVILEGES;
```

By using the query from Picture 25 again it is now confirmed that the root MySQL user is authenticated via the `my_sql_native_password` method:
3.5.3 Installation of PHP

The installation once again begins with the download and setup of the required packages via the use of the `apt` system:

```
$ sudo apt install php libapache2-mod-php php-mysql
```

After the installation finishes, it was necessary to modify the way Apache serves files upon a directory request. As things stand, Apache looks for a file called `index.html` whenever a user request a directory from the server. The server should prioritize PHP files over others, so priority will be given to the file named `index.php`. In order to do this, the `dir.conf` file must be accessed with root privileges and be opened via text editor. This is done with the following command:

```
$ sudo nano /etc/apache2/mods-enabled/dir.conf
```

The `dir.conf` file has the following structure:

```
<IfModule mod_dir.c>
  DirectoryIndex index.html index.cgi index.pl index.php index.xhtml index.html
</IfModule>
```

The `index.php` file needs to be moved to the first position, directly after `DirectoryIndex`. After successfully moving the file, the Apache web server needs to be restarted in order for the changes to apply. In order to restart Apache, the following command needs to be executed:
The status of the apache2 service can be checked by using the `systemctl` command:

```
$ sudo systemctl status apache2
```

The following output is displayed. Pressing Q exits this output:

**Sample Output**
- `apache2.service` - LSB: Apache2 web server
  - Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
  - Drop-In: /lib/systemd/system/apache2.service.d
    - apache2-systemd.conf
  - Active: active (running) since Tue 2018-04-23 14:28:43 EDT; 45s ago
  - Process: 13581 ExecStop=/etc/init.d/apache2 stop (code=exited, status=0/SUCCESS)
  - Process: 13605 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
  - Tasks: 6 (limit: 512)
  - CGROUP: /system.slice/apache2.service
    - /usr/sbin/apache2 -k start
    - /usr/sbin/apache2 -k start
    - /usr/sbin/apache2 -k start
    - /usr/sbin/apache2 -k start
    - /usr/sbin/apache2 -k start
    - /usr/sbin/apache2 -k start

In order to be able to get the most out of PHP and to be able to develop our web page and plugins properly, the package named `php-cli` will also need to be installed. This package is a command line interpreter for the PHP scripting language. It provides the `/usr/bin/php` command interpreter which is useful for testing PHP scripts from a shell, as well as performing general shell scripting tasks. In order to install it, the following command was executed:

```
$ sudo apt install php-cli
```
At this point the installation of PHP has been completed successfully. All that remains is to test that the system has been configured properly for PHP. In order to do this, a basic PHP script called info.php was created. The script was stored in the web root directory, in order for Apache to locate it and serve it correctly. After visiting the correct url (in our case http://sensornet.ihu.edu.gr/info.php) a page that provides basic information regarding the server from the perspective of PHP. This is used for debugging and in order to ensure the correct application of the settings. The file was removed after confirmation that everything was installed properly.

```php
<?php
phpinfo();
?>
```

After successfully testing the PHP settings, the setup of the LAMP stack is now complete. More work will be needed later in order to install phpMyAdmin and Wordpress but this is part of the web development process and is described in the next chapter.
4 Web Development

This chapter describes the process of developing the webpage of the project, which is one of the main goals of this dissertation. As it has been stated before, the page was created with Wordpress, while the database management system of choice was phpMyAdmin. The initial testing of the page’s and plugin’s functionality was done using XAMPP, while the code was written in JetBrains’ PhpStorm IDE.

4.1 Web development process

This section briefly describes the steps that were followed for the development process. Unlike most people think, web development[25] is not entirely code-based, if anything, coding is just a small part of the whole process.

4.1.1 Information Gathering

This part of the process involves the initial planning and the setting of goals. The goal was to create a web page that would be simple, easy in the eye yet efficient in the way it transmits information to the user. Its target audience would initially be users that are directly or indirectly related to the IHU, such as guests, students, staff and faculty. The webpage would provide all the basic information regarding the project, as well as information regarding the IHU. It would also go on to provide basic information regarding the sensors in use, the node location, while also providing a link for the mobile application. Finally, it would be mobile-friendly. The data from the nodes would be displayed on the home page.

4.1.2 Wireframe creation - Sitemap

At this stage it is time to create data in order to assess how the webpage will look like, as well as pick a technology stack that will be used for the development process. As far as the technology stack is concerned, as it has been mentioned often so far, the decision was made to develop the page with WordPress based on the use of the LAMP stack. The theme of choice was Bootstrap 4. The programming language would be PHP and the coding would be done with the use of PhpStorm. Regarding the sitemap, the main
page would come with a top menu which provides access to multiple pages which provide information about the campus, the nodes and their components, images of their locations, as well as a blog for the latest news regarding the IoT technology, which will be maintained by the IHU. Finally, the page will provide information and background regarding the mobile application along with a link for the .apk file, as well as contact information for the developer and content manager.

4.1.3 Page layouts / Review / Approval
At this stage of development, all the visual content of the webpage is created, gathered and shown to colleagues in order to agree on the final content. A rough layout of the webpage is designed, along with a choice for a logo, as well as relevant images from the nodes, the campus and the components that were used in the project. The final step of this process included discussing the selected content with colleague Kostas Karaklas and Prof. Stavros Stavrinides, who both approved the selection.

4.1.4 Content writing and management
Although this step of the process might appear trivial, it is crucial in order to attract visitors and keep them interested in navigating around the website. This step included the writing of texts about the nodes, the campus, the project and the Internet of Things as a topic in general. Headlines were also picked along with other content that aim to provide the user with a wholesome and comprehensible navigating experience.

4.1.5 Coding
This step involves the actual coding of the plugin that was created in order to display the sensors’ reading in real time on the webpage. It also involved the creation of the database, along with the frontend coding in order to make the webpage user-friendly and mobile-friendly.

4.1.6 Testing, Review and Launch
The functionality of the plugin and the webpage was tested in a local environment with the use of the XAMPP cross-platform web server solution pack. For the database management phpMyAdmin was used, a free software tool written in PHP which is responsible for handling the administration of MySQL over the Web. After the webpage func-
tionality was tested successfully, the page was reviewed by the author’s colleagues. Finally, the webpage was launched on the IHU Virtual Server.

4.1.7 Maintenance and updates
This stage refers to the future of the project; the WordPress CMS, along with the theme and the installed plugins need to be kept updated on a regular basis; the content also needs to be kept up-to-date with the latest news regarding the technology of the Internet of Things. This is a continuous process, to be handled by the author and future students of the Department willing to assist in keeping this project alive.

4.2 Development tools

4.2.1 PhpStorm
Phpstorm is a commercial, cross-platform IDE built by JetBrains[26]. It provides the developer with a PHP, HTML and Javascript editor along with on-the-fly code analysis, error prevention and automated refactorings for PHP and Javascript code. The application can be extended by plugins that are developed by JetBrains or a developer can write their own plugin. Since the author of this dissertation already works as a web developer, the IDE choice was easy enough to make.

4.2.2 phpMyAdmin
phpMyAdmin is a free software tool written in PHP, responsible for handling the administration of MySQL over the Web[27]. It supports a wide range of operations on MySQL and MariaDB. The tool comes with a user interface while providing the developer with the ability to execute any SQL statement. The software is extremely well-documented and the development team also provides excellent support.

4.2.3 XAMPP
XAMPP is a free, open-source cross-platform web server solution stack package developed by Apache Friends[28]. It mainly consists of the Apache HTTP Server, MariaDB database, as well as interpreters for scripts written in PHP and Perl. XAMPP is an excellent choice for testing since it provides all the features that are installed in the Virtual server, which made the transition from a localhost environment to the live server possible.
4.2.4 Wordress

Wordress is currently the most popular Content Management System (CMS) in the world, powering more than 30% of the websites globally[29]. A content management system is a web application that allows website owners, authors and editors to manage their website and publish content. Wordress makes use of PHP and MySQL which are both widely supported. Even though CMS platforms were originally used for blogging, any WordPress site can be turned into anything a developer desires. Wordress is open-source and free for everyone, while its interface is flexible and extremely user-friendly. It also boasts an excellent community and support with millions of plugins and themes created by users and developers alike.

The advantages of Wordress include its low cost, the easy installation and update process, simplicity of management, custom design and functionality, as well as a very active and helpful community.

However, Wordress also comes with a couple of disadvantages; since it powers a huge number of webpages globally, Wordress also comes under attack from hackers very often. The installation of a security plugin is strongly recommended. Another disadvantage has to do with the third-party plugins and themes. Even though many of those are created by other developers or users and they are free of charge, sometimes they can turn out to be buggy. Finally, a Wordress page with too many installed plugins may become laggy and produce long loading times.

4.3 Coding preparation

This chapter will briefly describe the steps taken to install and configure the tools that will be used for the web development process.

4.3.1 XAMPP Setup & Configuration

After downloading the executable file, XAMPP offers a list of components that can be installed. Even though all components are installed, for this project we will mostly need PHP, phpMyAdmin and MySQL.
After successfully completing the installation, the XAMPP control panel launches. The services that need to be launched from the control panel are Apache and MySQL.
4.3.2  phpMyAdmin / database setup
As it was mentioned in the previous chapter, installing XAMPP automatically installed phpMyAdmin. After the installation of Wordpress and the creation of its database, the following SQL script is executed in order to create the database tables for the sensor nodes.
The script essentially creates three tables, one for each node. After creating the tables, the database of the project looks like this:
4.3.3 Wordpress Setup

Before beginning the plugin development, Wordpress needs to be setup. After downloading the Wordpress files from the official webpage and since XAMPP and phpMyAdmin is already installed, the Wordpress files need to be placed in the local XAMPP folder, inside the *htdocs* subfolder.
The file structure of Wordpress looks like this:

![Wordpress file structure](image1)

Picture 45: Wordpress file structure

Since phpMyAdmin is already installed and as it was mentioned in the previous chapter the database is already setup, the next step would be to access the localhost location from the browser and begin the main installation.

![Wordpress installation](image2)

Picture 45: Wordpress installation
After the installation is successful, the only thing that remains before beginning the plugin development is choosing a Wordpress theme. For the needs of this project, the Bootstrap4 theme was chosen and then installed in the webpage.

![Wordpress admin dashboard – Bootstrap 4 theme](image)

**Picture 46: Wordpress admin dashboard – Bootstrap 4 theme**

### 4.4 Code Review

The logic behind the development of the plugin is rather simple; the main use of the plugin is to pull sensor data from the database via a GET request in order to query the database over time. The response of this GET request is a JSON string. This method applies for every sensor node table in the database. This dissertation will not discuss the frontend development of the webpage in detail, but will rather review the backend code that displays the node data live on the webpage.
The file structure of the plugin looks like this:

![Wordpress plugin file structure](image)

- The `api` folder contains the controllers and the models of the plugin.
- The `public` folder contains files related to the frontend.
- The `templates` folder contains templates that are used in the frontend.

The `SmartIHUplugin.php` file contains a class that initializes the plugin, includes the files that are necessary for the initialization process and declares global variables that are used in the plugin. It also contains the internal Wordpress path of the plugin and the url of the API endpoint.
Within the public folder the file structure looks like this:

- The assets folder contains Javascript and CSS files that are used for the development of the frontend.
- The pages folder is of no use since in order to display the data from the nodes on the webpage, we use shortcodes.
- The shortcodes displays the data from the nodes in data table view. Using shortcodes in wordpress provides the developer with the ability to display the data in whichever page of the site he wishes.
  - Within the shortcodes folder there are three PHP files, one for each node.
Picture 49: Example of a shortcode PHP file (Node #1)

This way the data tables of the nodes can be added in this way in any page around the Wordpres page:
Finally, the SmartIHUFrontend.php contains a class that is called by SmartIHUPlugin.php and contains methods that include the styles and scripts that are used by the shortcodes. The class also sets the full-width template for the pages that will be using the shortcodes. Finally, it creates the shortcodes.
Finally, the *api* folder structure looks like this:

![api folder structure](image)

- The *SmartIHUApiRegistrar.php* file creates the API endpoints for every controller:
- The *models* folder contains the models for every node as well as methods for data manipulation and storing.
class NodeOne
{
    // Database table name
    private $tableName = "ai_p_nodeone";

    // Object properties
    public $id;
    public $general_error_message;
    public $general_error;
    public $node_name;
    public $readings;
    public $node_entry_date;
    public $anemometer;
    public $bme680;
    public $tsi2591;
    public $created_at;

    public function setData($data)
    {
        $this->general_error_message = $data->general_error_message;
        $this->general_error = $data->general_error;
        $this->node_name = $data->node_name;
        $this->readings = $data->readings;
        $this->node_entry_date = $data->node_entry_date;
        $this->anemometer = json_encode($data->anemometer);
        $this->bme680 = json_encode($data->bme680);
        $this->tsi2591 = json_encode($data->tsi2591);
        $this->created_at = date("Y-m-d H:i:s");
    }

    // Index
    function index()
    {
        // Select all query
        $query = "SELECT
            n.id, n.general_error_message, n.general_error, n.node_name, n.readings, n.node_entry_date, n.anemometer, n.bme680, n.tsi2591
        FROM
            " . $this->tableName . " n
        ORDER BY
            n.id DESC";

        global $wpdb;
        return $wpdb->get_results($query);
    }

    // Store
    function store()
    {
        $data = [
            "general_error_message" => $this->general_error_message,
            "general_error" => $this->general_error,
            "node_name" => $this->node_name,
            "readings" => $this->readings,
            "node_entry_date" => $this->node_entry_date,
            "anemometer" => $this->anemometer,
            "bme680" => $this->bme680,
            "tsi2591" => $this->tsi2591,
            "created_at" => $this->created_at
        ];

        global $wpdb;
        $result = $wpdb->insert($this->tableName, $data);

        // If insert was successful, assign last inserted id to category id
        if ( $result ) {
            $this->id = $wpdb->insert_id;
            return true;
        }

        return false;
    }
}
The v1 folder contains the controllers that expose the public APIs for data management of the nodes and use the models from the models folder.

```php
include(plugin_dir_path(__FILE__) . 'models/NodeOne.php');
include_once 'Controller.php';

class NodeOneController extends Controller{

    protected $base = 'nodeone';

    public function index() {
        // Initialize object
        $node = new NodeOne();
        // Query properties
        $results = $node->index();
        $data = [];
        foreach ($results as $result) {
            $data[] = [
                "id" => $result->id,
                "general_error_message" => $result->general_error_message,
                "general_error" => $result->general_error,
                "node_name" => $result->node_name,
                "readings" => $result->readings,
                "node_entry_date" => $result->node_entry_date,
                "anemometer" => json_decode($result->anemometer),
                "bme680" => json_decode($result->bme680),
                "ts12591" => json_decode($result->ts12591)
            ];
        }
        return new WP_REST_Response($data, 200);
    }

    public function store() {
        // Get user input and decode it.
        $data = json_decode(file_get_contents("php://input"));
        // Initialize object
        $node = new NodeOne();
        $node->setData($data);
        if ($node->store()) {
            return new WP_REST_Response($this->createResponseData($node), 200);
        }
        return new WP_REST_Response("Error", 400);
    }

    private function createResponseData($node) {
        return [
            "id" => $node->id,
            "general_error_message" => $node->general_error_message,
            "general_error" => $node->general_error,
            "node_name" => $node->node_name,
            "readings" => $node->readings,
            "node_entry_date" => $node->node_entry_date,
            "anemometer" => $node->anemometer,
            "bme680" => $node->bme680,
            "ts12591" => $node->ts12591
        ];
    }
```

Picture 54: Sample controller file: NodeOneController.php
Picture 55: Live display of the node data
5 Mobile Development

This logic behind the mobile development of the Smart IHU app is very similar to the web-based Wordpress plugin. With the use of the GET endpoints, the data from every node is retrieved from the database in JSON format and is then converted into Java objects via the Gson library. The generated objects are then displayed on the frontend. This mobile app was developed with the use of Google’s Android Studio. The coding language used is Java.

5.1 Code Review

The main file structure of the mobile app is the following:

<table>
<thead>
<tr>
<th>Folder</th>
<th>Date</th>
<th>Time</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>controllers</td>
<td>14-Apr-19</td>
<td>17:05</td>
<td>File folder</td>
</tr>
<tr>
<td>helpers</td>
<td>14-Apr-19</td>
<td>16:43</td>
<td>File folder</td>
</tr>
<tr>
<td>interfaces</td>
<td>31-Jan-19</td>
<td>20:20</td>
<td>File folder</td>
</tr>
<tr>
<td>models</td>
<td>14-Apr-19</td>
<td>16:20</td>
<td>File folder</td>
</tr>
<tr>
<td>navigation</td>
<td>14-Apr-19</td>
<td>15:54</td>
<td>File folder</td>
</tr>
<tr>
<td>singletons</td>
<td>31-Jan-19</td>
<td>20:17</td>
<td>File folder</td>
</tr>
<tr>
<td>views</td>
<td>16-Feb-19</td>
<td>15:04</td>
<td>File folder</td>
</tr>
</tbody>
</table>

Picture 56: Mobile app file structure

- The controllers folder contain methods that call the API endpoints which are exposed by the Wordpress plugin. The controller files contain the index() method which calls the respective API which in turn returns the list of the nodes. The folder also contains the ActivityInitializer; a class which is extended by every controller and contains methods related to the User Interface and methods that initialize the requests.
package blastek.net.smartihumobile.controllers;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotationSuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotation.SuppressLint;
import android.annotationSuppressLint;
import android.annotation.SuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotation.SuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotationSuppressLint;
import android.annotation Dayton
package blastek.net.smartthumbble.controllers;

import android.os.Bundle;
import android.util.Log;
import android.view.MenuItem;
import com.android.volley.Request;
import com.google.gson.Gson;
import com.google.gson.reflect.TypeToken;
import java.lang.reflect.Type;
import java.util.List;
import blastek.net.smartthumbble.R;
import blastek.net.smartthumbble.helpers.ApiUrls;
import blastek.net.smartthumbble.helpers.FragmentConstants;
import blastek.net.smartthumbble.interfaces.ResponseListener;
import blastek.net.smartthumbble.interfaces.Router;
import blastek.net.smartthumbble.models.NodeOne;
import blastek.net.smartthumbble.views.nodeone.NodeOneListFragment;

public class NodeOneController extends ActivityInitializer implements Router {

    private String activityTitle;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // This is the main activity so get user's related data as early as possible
        // and keep them in Shared Preferences file.
        activityTitle = getString(R.string.node_one_results);
        index();
    }

    @Override
    public void index() {
        ResponseListener<String> responseListener = new ResponseListener<String>() {
            @Override
            public void onResponse(String response) {
                try {
                    Type listType = new TypeToken<List<NodeOne>>(){}.getType();
                    List<NodeOne> list = new Gson().fromJson(response, listType);
                    NodeOneListFragment listFragment = NodeOneListFragment.newInstance(list);
                    getSupportFragmentManager().beginTransaction().add(R.id.fragment_container, listFragment,
                            FragmentConstants.NODEONE_LIST_FRAGMENT).commit();
                    fadeOutFragmentContainer();
                } catch (Exception e) {
                    Log.e("index", e.getMessage() + " + e.getCause());
                }
            }
            @Override
            public void onErrorResponse() {
                updateOnErrorResponse(activityTitle);
            }
        };
        initRequest(ApiUrls.NODE_ONE_ROOT, responseListener);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here.
        // The action bar will automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        Picture 58: Sample picture of NodeOneController.java
• The **helpers** folder contains classes with methods for the controllers, f.e. the **ApiUtils** class that contains the strings to the API endpoints and methods in order to set the headers for every request.

```java
package blastek.net.smartihumobile.helpers;
import android.content.SharedPreferences;
import java.util.HashMap;
import java.util.Map;

/**
 * Created by Blastblitz.
 */
public class ApiUtils {

    // ApiUrls
    public static final String API_V1_URL = "http://10.8.2.2/smartihu/wp-json/api/v1";
    public static final String MESSAGE = "message";
    public static final String API_PREFERENCES = "apiPreferences";

    // Resource
    public static final String NODE_ONE_ROOT = API_V1_URL + "/nodeone";
    public static final String NODE_TWO_ROOT = API_V1_URL + "/nodetwo";
    public static final String NODE_THREE_ROOT = API_V1_URL + "/nodethree";

    public static Map<String, String> getApiHeaders(SharedPreferences apiPreferences) {
        Map<String, String> headers = new HashMap<>();
        headers.put("Accept", "application/json");
        return headers;
    }
}
```

Picture 59: Sample helper file - ApiUtils.java

• The **interfaces** folder contains two interfaces which are implemented by the controllers.
  
  o The **ResponseListener** interface is executed at the end of the API requests
  
  o The **Router** interface contains an index() method for the service of the requests.
The `models` folder contains the models of the nodes and the data fields for each node. The models have been created in a way that allows the data that have been returned from the APIs to be parsed by the Gson Library.
```java
package blastek.net.smartihomemobile.models;

import com.google.gson.annotations.Expose;
import com.google.gson.annotations.SerializedName;
import java.io.Serializable;

public class NodeThree implements Serializable {
    @Expose
    private String id;
    @SerializedName("general_error_message")
    private String generalizedErrorMessage;
    @SerializedName("general_error")
    private String generalError;
    @SerializedName("node_name")
    private String nodeName;
    @SerializedName("node_entry_date")
    private String nodeEntryDate;
    private Camera camera;
    @SerializedName("car_count")
    private CarCount carCount;

    public String getGeneralizedErrorMessage() {
        return generalizedErrorMessage;
    }

    public String getGeneralError() {
        return generalError;
    }

    public String getNodeName() {
        return nodeName;
    }

    public String getNodeEntryDate() {
        return nodeEntryDate;
    }

    public Camera getCamera() {
        return camera;
    }

    public CarCount getCarCount() {
        return carCount;
    }
}
```

Picture 62: Sample NodeThree.java
The navigation folder contains a class that creates the navigation drawer.
The **singleton**s folder contains the VolleySingleton class which is the implementation of the Volley Library for HTTP requests. It was deemed preferable to use the Singleton pattern in order to have a single Volley instance for the whole application.

Finally, the views folder contains classes that extend the fragment() class and contain methods for the projection of the nodes in lists with the use of RecyclerViewAdapter().
package blastek.net.smartihomobile.views.nodeonelist;

import android.content.Context;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v7.widget.DividerItemDecoration;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import java.io.Serializable;
import java.util.List;

import blastek.net.smartihomobile.R;
import blastek.net.smartihomobile.models.NodeOne;

public class NodeOneListFragment extends Fragment {

    private static String NODENONE_LIST = "nodesOneList";
    private List<NodeOne> nodeOneResultList;
    private NodeOneRecyclerItemClickListener nodeOneRecyclerItemClickListener;

    public NodeOneListFragment() {
    }

    public static NodeOneListFragment newInstance(List<NodeOne> nodeslist) {
        NodeOneListFragment fragment = new NodeOneListFragment();
        Bundle arguments = new Bundle();
        arguments.putSerializable(NODENONE_LIST, nodeslist);
        fragment.setArguments(arguments);
        return fragment;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Bundle arguments = getArguments();
        if (arguments != null) {
            nodeOneResultList = (List<NodeOne>) arguments.getSerializable(NODENONE_LIST);
        }
    }

    @Override
    public void onResume() {
        super.onResume();
        getActivity().setTitle(getString(R.string.node_one_results));
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.node_one_list_fragment, container, false);
        // Set the adapter
        Context context = view.getContext();
        RecyclerView recyclerView = view.findViewById(R.id.recyclerView);
        recyclerView.setLayoutManager(new LinearLayoutManager(context));
        recyclerView.addItemDecoration(new DividerItemDecoration(view.getContext(), DividerItemDecoration.VERTICAL));
        recyclerView.setAdapter(new NodeOneRecyclerAdapter(nodeOneResultList, nodeOneRecyclerItemClickListener));
        recyclerView.setAdapter(new NodeOneRecyclerAdapter(nodeOneResultList, nodeOneRecyclerItemClickListener));
        return view;
    }
}
package blastek.net.smarthumobile.views.nodeone;

import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import com.google.gson.JsonObject;
import com.google.gson.JsonParser;

import java.util.List;

import blastek.net.smarthumobile.R;
import blastek.net.smarthumobile.helpers.Fields;
import blastek.net.smarthumobile.models.Anemometer;
import blastek.net.smarthumobile.models.BME;
import blastek.net.smarthumobile.models.TSL;

public class NodeOneRecyclerViewAdapter extends RecyclerView.Adapter<NodeOneRecyclerViewAdapter.ViewHolder> {
    private final List<NodeOne> mValues;

    NodeOneRecyclerViewAdapter(List<NodeOne> items) {
        mValues = items;
    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.node_one_fragment, parent, false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(final ViewHolder holder, int position) {
        final NodeOne node = mValues.get(position);
        if (node != null) {
            Anemometer anemometer = node.getAnemometer();
            BME bme680 = node.getBME();
            TSL tsl2591 = node.getTSL();

            holder.mNodeNameTv.setText("Node name: " + node.getNodeName());
            holder.mNodeEntryDateTv.setText("Node entry date: " + node.getNodeEntryDate());

            holder.mAnemometryTv.setText("Anemometer: \n\n + Wind Speed: \n + \n + MCP3408 output: \n + MCP3408 output: ");

            holder.mBme680Tv.setText("BME680: \n\n + Temperature: \n + \n + Humidity: \n + \n + Gas: \n + BME680.getGas();

            holder.mTsl2591Tv.setText("TSL2591: \n\n + l\n + \n + \n + \n + \n + ");
        }
    }

    public class ViewHolder extends RecyclerView.ViewHolder {
        public ViewHolder(View itemView) {
            super(itemView);
        }
    }
}

Picture 66: Sample NodeOneRecyclerViewAdapter.java
Node1 Results

Node name: RPI-node_1

Node entry date: 2019-01-11 13:14:32

Anemometer:

Wind Speed: 0.06532258
MCP3008 output: 125

BME680:

Temperature: "20.458"
Humidity: "37.435"
Gas: "734683"

TSL2591:

Lux: "0.2937600000000001"
Ir: "5"
Full Spectrum: "327690"

Node name: RPI-node_1

Node entry date: 2019-02-12 09:19:16

Picture 67: Sample picture of Node1 results in the mobile app
6 Inspirational Projects

The idea behind this project was conceived earlier this year, but it didn’t come out of nowhere. There are currently many other IoT projects that are being developed in a larger scale and are playing a crucial role to the betterment of the lives of citizens around the world. The writer of this dissertation also works as a developer in one. It is only fair that honorable mention is made to some of the projects that kicked this project into life, proved inspiring and ultimately helped in its completion.

6.1 DUT – “Smart University Campus”

The Democritus University of Thrace in cooperation with COSMOTE was the first university in Greece to launch a “Smart University Campus” located in Xanthi[30]. The project makes use of the Narrow-Band Internet of Things (NB-IoT). It is considered a pilot program that takes full advantage of the capabilities of NB-IoT technology, thus managing to promote solutions via smart applications around the campus.

More specifically, the following solutions were implemented:

- Air Quality Monitoring: special sensors for air quality measurement were installed; the measurements include real-time temperature, humidity, pressure,
various gases and particles. They also enable optimal design and implementation of appropriate actions for the reduction of atmospheric pollution.

- **Smart Tank Fuel Management:** This application makes use of a gas oil level meter which has been installed in order to prevent over-consumption, fuel theft and valuation of invoices. The application is based on the solution provided by the Greek company Fuelics and is being implemented in cooperation with Ericsson.

- **Water Quality Measurement:** Ensures the quality of drinking water at the campus and is being implemented in collaboration with Greek company Wings.

- **Smart Lighting:** makes use of NB-IoT technology in order to provide assistance with light adaption depending on the season and time of day in order to drastically reduce power consumption.

The use of the NB-IoT technology provides an easier and more-cost effective interconnection between network devices as no special equipment is required. This technology also offers better indoor coverage since it makes use of a mobile telephony network and is considered to be superior to other IoT technologies in terms of quality and safety.

### 6.2 The ICARUS Project

ICARUS is a project funded from the European Union’s Horizon 2020 Research and Innovation programme[31]. Its main objective is to develop integrated tools and strategies for urban impact assessment in support of air quality and climate change governance in EU Member States leading to the design and implementation of appropriate abatement strategies to improve the air quality and reduce the carbon footprint in European cities. The project will develop detailed policies and measures for air pollution and climate control for the short and medium term (until ca. 2030). For the long term perspective (2050 and beyond) ICARUS will develop visions of green cities and explore pathways on how to start realizing these visions.

The ICARUS project boasts a detailed webpage where the whole scope of the project is explained in detailed, along with deliverables, newsletters, news and events, as well as a forum where citizens and scientists alike may discuss ongoing research or concerns regarding environmental issues.

The ICARUS project has held campaigns in most of Europe’s major cities, where volunteers make use of sensors in their households and their every-day life in order to
measure environmental data that will assist in improving the environmental conditions in the long run. Namely, every household that participated in a campaign made use of a uHoo household sensor, while every volunteer used a Garmin vivosmart watch every day for a week while performing their daily routine. Data gathered by the sensors is exported via a data portal developed by UPCOM and then is made available to scientists for further analysis.

![uHoo household sensor and the uHoo app](image)

**Picture 69:** Sample picture of the uHoo household sensor and the uHoo app

![Garmin vivosmart](image)

**Picture 70:** Sample picture of a Garmin vivosmart used in the ICARUS project

ICARUS will also deliver a Decision Support System (DSS), a web-based, flexible and interactive platform aimed at assisting stakeholders in the selection, application and evaluation of the available datasets and tools for urban impact assessment in support of air quality and climate change governance at different spatial and temporal scales and taking into account the specific regulatory context.
Finally, the ICARUS project is currently developing a mobile application, the ICARUS UCT (User-Centric-Tools), essentially a platform where every user may register daily activities such as walking, jogging, running, driving their car, as well as track activities like recycling, waste management or other tasks that are related to environmental awareness.

The proposed integrated system deployment and application in several cities across Europe is expected to have a positive impact on local society and the economy contributing to lower the health costs associated to environmental burden.
7 Conclusions

The final chapter of this dissertation aims to provide some insight on the project and its impact on the author, his colleagues and the campus. It describes the reasons why this project was undertaken, the process, the journey, the impact the data can have, as well as the problems that arose along the way, or even before it even started.

7.1 The Internet of Things
After this journey has come to its end, it is the author’s belief that the Internet of Things already has a significant impact on the world and ultimately, it is the future. From households to offices, from small towns to large cities, from manufacturing to transportation, from healthcare to the environment, from agriculture to energy, the technology of the Internet of Things has found its way into everyone’s daily life. As of today, billions of devices are connected to the internet and keep producing data that can be used to change the lives of every citizen across the world for the better. This technology was used for this project with the aim of producing environmental data that will help better the lives of every guest, student, staff and faculty that spends a good portion of their day in the International Hellenic University.

7.2 The Nodes and the Network
The setup of the nodes and the sensor network proved to be a unique experience. The hands-on approach on this part of the project helped the author gain good knowledge of how the sensors and the network operate. It also went a long way to show how a cheap and affordable product like the Raspberry Pi can prove to be a powerful asset when used correctly.

7.3 Environmental Data
The data gathered in this project will be of immense value for the IHU for a variety of reasons. Firstly, it can determine the environmental conditions around the campus area;
that way the administration can take steps to improve indoor conditions based on the
data of the classroom sensor. The weather station provides the university with meteorological data as well as environmental quality data which can be used by the IHU itself in other projects, or can be provided to stakeholders of other projects for scientific research.

### 7.4 Development

The development of this project, web and mobile, has assisted the author in various way. Web development helped him gain a better understanding of the Bootstrap4 theme, while the plugin development proved quite challenging. The greatest benefit was gained from the mobile development since the author had no prior experience in the specific field.

### 7.5 Problems and hindrances

From the author’s point of view, the most serious of problems occurred with the arrival of the sensor parts. Due to the fact that the majority of these parts are manufactured abroad and for bureaucratic reasons regarding their order, the amount of time lost on this process took a huge toll in the time it took for the project to complete. Another problem that occurred was the fact that the sensors appear to be quite fragile. Several of them arrived damaged and were thus deemed inoperable which resulted in more delays and time lost.
7.6 The future of the project

The project still appears to have a lot of potential. From the author’s view, every aspect of it can be expanded and improved. The network will benefit from the addition of more sensors that can enhance its effectiveness and the impact it has on the campus grounds. Network security can be enhanced in order to prevent any malicious attempts to exploit the nodes. The webpage can be enriched with more sections and attract even more visitors who can gain a better understanding of the value of the Internet of Things as a technology. The mobile application can be improved as well, with an iOS version being the next step.

Bibliography


December 2018].


[18] https://www.adafruit.com/product/164, "IR DISTANCE SENSOR INCLUDES


Appendix A

Project tools
- PHPStorm
- XAMPP
- phpMyAdmin
- Wordpress
- Google Android Studio