VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Toni Haukkala

UG, UX AND WORDPRESS DEVELOPMENT FOR IOT AND EMBEDDED SYSTEMS

Technology
2022

VAASAN AMMATTIKORKEAKOULU
Tietotekniikka

TIIVISTELMÄ

Tekijä	Toni Haukkala
Opinnäytetyön nimi	UG, UX ja WordPress kehitys IoT ja sulautettuihin järjestelmiin.
Vuosi	2022
Kieli	englanti
Sivumäärä	33
Ohjaaja	Smail Menani

Opinnäytetyön tarkoituksena on lähettää sensoridataa Arduino Nanolta BLE:n yli ja vastaanottaa saapuvaa dataa hyödyntäen Python-ohjelmarajapintaa. Graafisena käyttöliittymänä kirjautuneelle käyttäjälle toimii WordPress. Data esitetään numerona ja kaaviona.

Työ pyörii suurimmaksi osaksi WordPressin ympärillä. Arduinossa on valmis ohjelma jonka tarkoituksena on lähettää sensoridataa. Data vastaanotetaan Raspberry Pi:lle, jossa pyörii Python-ohjelmarajapinta. Käyttäjä pystyy seuraamaan sensoridataa WordPressissä reaaliajassa.

Työ saatiin onnistuneesti suoritettua vaatimusten mukaisesti. Verkkosivut luotiin ja sensori data vastaanotettiin onnistuneesti. Sensoridata näkyi ja päivittyi onnistuneesti kaavion muodossa verkkosivuilla.

Opinnäytetyötä voidaan tulevaisuudessa hyödyntää mahdollisesti monelle eri laitteelle, jos niiden dataa halutaan seurata.

Avainsanat WordPress, Raspberry Pi, Arduino, Python, BLE, Elementor

UNIVERSITY OF APPLIED SCIENCES
Tietotekniikka

ABSTRACT

Author	Toni Haukkala
Title	UG, UX and WordPress Development for IoT and Embedded Systems
Year	2022
Language	English
Pages	33
Name of Supervisor	Smail Menani

The purpose of this thesis was to develop a system to send sensor data from Arduino Nano over BLE and receive sensor data using the Python interface. WordPress acts as a user interface for logged user. Data is presented as a numeric value and as a chart format.

This project revolves mostly around WordPress. Arduino has finished software that sends sensor data. Data is received using Raspberry Pi that uses the Python interface. The user can see sensor data in WordPress in real time.

This project was successfully completed as required. The website was created and sensor data was successfully received. The sensor data was displayed and updated in a chart form on the website.

This thesis can be utilized in the future for many different devices if you want to see their data in real time.

Keywords WordPress, Raspberry Pi, Arduino, Python, BLE, Elementor

CONTENTS

LIST OF CONCEPTS AND ABBREVIATIONS

API	Application programming interface
BLE	Bluetooth Low Energy
HTML	Hypertext Markup Language
IoT	Internet of Things
JSON	JavaScript Object Notation
REST	Representational State Transfer
RRD	Round Robin Database
RRDTool	Round Robin Database Tool

LIST OF FIGURES

1 INTRODUCTION

The impact of IoT (Internet of Things) has grown significantly in the world over the recent years. Because of this, a number of different devices are connected to the internet. Prime examples of IoT being used in the world are in home automation (thermostats, light systems and smoke detectors), security systems (cameras and motion sensors) and health care systems (heart monitors and sleep monitors). The rise of IoT can make everyday life easier and bring security and safety into people's lives.

In this thesis two devices are used, Arduino Nano 33 BLE Sense and Raspberry Pi 3b+. Using coded software inside Arduino and BLE, sensor data is transferred in Raspberry. The Raspberry Pi uses the Python code to connect to Arduino and receive the data. Then data is transmitted to the WordPress where the user can choose to see data from a specific sensor in real time.

This project explores the idea of a user who wants to track a specific sensor data in real time from a website. For example, if a user grows chili, then temperature, humidity and luminosity would be the desired metrics to be tracked. Their status would then be seen from a website in real time.

In Chapter 2 technologies, systems and devices used in this project are introduced and how these things co-operate together and make the working product. Every step of how the project was planned is explained in Chapter 3 and all the different execution phases and how the project was implemented is covered in Chapter 4. All the different challenges and obstacles that occurred during different development phases of this project and how they were handled are described in Chapter 5. The subject of thesis was offered by and was done for Delektre Ltd.

1.1 Delektre Oy

Delektre Ltd was established in 2010 to serve high tech companies, research centres and universities in idea commercialisation. This path led to own product development and research into non-invasive optical measurement on soft tissue. During the years the Delektre Ltd has grown its partnership network around the world and the major part of the sales is exported to other countries.

2 TECHNOLOGIES AND DEVICES

Devices that were used in this project were Arduino Nano 33 BLE Sense and Raspberry Pi 3b+. Notable technologies used were BLE, Python, JavaScript, WordPress and a plugin for WordPress called Elementor.

2.1 Internet of Things

”The Internet of Things (IoT) encapsulates a vision of a world in which billions of objects with embedded intelligence, communication means, and sensing and actuation capabilities will connect over IP (Internet Protocol) networks.” (Cirani, S., Ferrari, G., Picone, M., Veltri, L. 2018, p. 1).

IoT (Internet of Things) is a network of physical objects that have capabilities of connecting and exchanging data with each other. The most basic architecture of IoT is a three-layer architecture as shown in Figure 1. The Perception layer is responsible for gathering information and sensing other smart objects in the environment. The network layer is responsible for transmitting and processing data. It connects to other smart objects, devices and servers. The Application layer is responsible for delivering specific services to the user depending on the application.

2.2 Arduino Nano 33 BLE Sense

Arduino Nano 33 BLE Sense is the smallest available 3.3V AI enabled board Arduino has to offer with a form factor of 45x18mm (Arduino.cc, 2021 a). The board comes with a variety of different kind of sensors that enables the measurement of temperature, humidity, luminosity, pressure, noise, velocity and movement. Arduino Nano 33 BLE Sense can be seen in **Figure 1**.

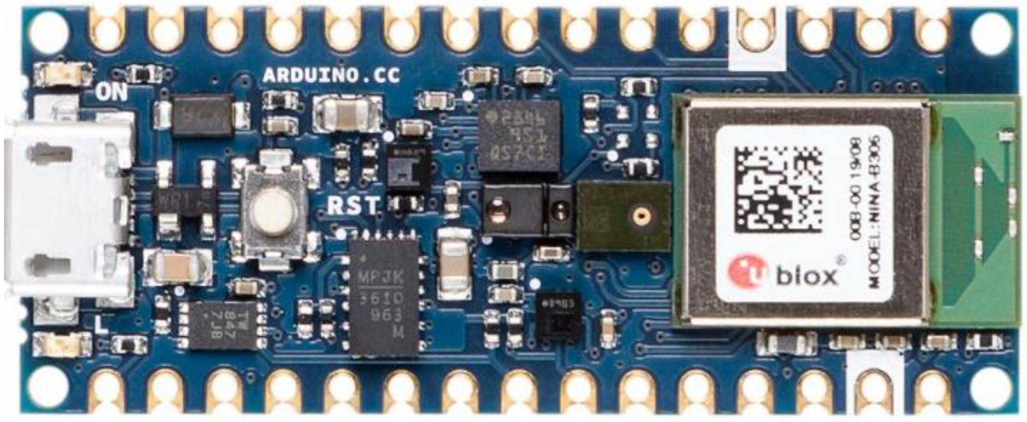


Figure 1. Arduino Nano 33 BLE Sense.

2.3 Bluetooth and BLE

Bluetooth is a wireless way to communicate between devices. It uses radio frequency to share data over a short distance. Bluetooth has been utilized in many areas for example, changing the audio streaming by stripping away wires so media could be consumed wirelessly (Bluetooth, 2021 a). The Bluetooth data transfer is used on many devices and some of these include sport trackers and pc accessories (Bluetooth, 2021 b). The Bluetooth location services can be used to locate certain items, other points of interest and is an excellent wayfinding tool in a large building (Bluetooth, 2021 c). The Bluetooth mesh networking enables communication between many devices, and this is an ideal way to create control, monitoring and automation systems (Bluetooth, 2021 d).

Bluetooth Low Energy (BLE) is a wireless way to communicate with devices that is optimized for low power use at low data rates (Arduino.cc, 2021 b).

2.4 Raspberry Pi 3 model B+

Raspberry Pi is a small sized low-cost computer that can be used for example to explore and learn different kinds of programming languages or anything to do with computing. Raspberry Pi is capable to do anything a normal computer does and possibilities with it are endless.

In this thesis we used Raspberry Pi 3 model B+ that can be seen in **Figure 2**. It is the final version of Raspberry's third generation single board computer (Raspberry Pi, 2021).

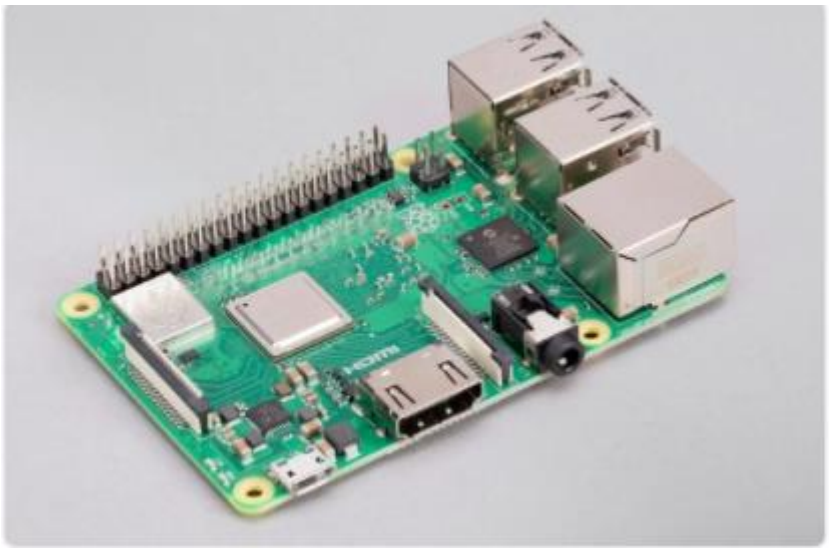


Figure 2. Raspberry Pi 3 model B+.

2.5 Python

Python is easy to learn programming language that emphasizes code readability. This comes from not having to deal with brackets and using significant whitespace. Usually, users favor Python because of the edit-test-debug cycle. Because there is no compilation step, this increases productivity (Python, 2021).

2.6 HTML

HTML short for Hypertext Markup Language, is the main markup language of the web and used for creating web pages. HTML is considered easy to learn and being straightforward. Usually using HTML alone is not enough for building a proper website, so it is a good idea to use other languages such as CSS and JavaScript (Hostsinger Tutorials, 2021).

2.7 JavaScript

JavaScript is a programming language that allows the use of more complex features on web pages. Basically, this includes all dynamically updating content for example self-updating charts that were used in this work.

2.8 Visual Studio Code

Visual Studio Code is a free source-code editor made by Microsoft. It is available for Windows, Linux and MacOS. In this thesis Visual Studio code was used to create code for self-updating charts.

2.9 VNC Viewer

VNC Viewer is a graphical desktop sharing system that can be used from the local computer or phone. It allows the remote control of a computer as though sitting in front of it. In this thesis VNC Viewer was used to obtain a remote connection to Raspberry Pi.

2.10 WordPress

WordPress is an open-source software that can be used to build a website. It is used between individual users and big companies and more than 38% of the websites on the internet are done with WordPress. Reasons why WordPress is used because its free and easy to use. It has over 50 000 plugins and 5000 themes for free which makes WordPress extensible. WordPress has also many available commercial plugins and themes.

2.11 Elementor

Elementor is a free plugin for WordPress and is the world’s leading WordPress website builder. It replaces the basic WordPress editor that enables the making of more complex websites more easily.

2.12 RRDTool

RRDTool (Round Robin Database Tool) is an opensource tool that handles time series data. RRDTool creates Round Robin Databases. When a database is created, the size and structure are determined at the beginning. When new data is saved, old one is discarded to keep the database the same size.

3 DESIGN

In this chapter architecture of the project is introduced; how the systems and devices communicate with each other and how sensor data is handled and how the website was planned.

Regular team meetings were held all throughout the different phases of the project. In these meetings the progress of the project was discussed, how to continue the project after a specific point and if tips or help would be needed.

3.1 Architecture

As it can be seen in **Figure 3**, Arduino Nano has many sensors and has the possibility of having more sensors if added. Arduino Nano transmits the sensor data with BLE to Raspberry Pi and Raspberry Pi uses the Python code to connect to Arduino Nano and receive the sensor data. After the sensor data is received, it is stored to the data storage. WordPress gets the sensor data from data storage and shows the data in a form of self-updating chart. The user can then open the webpage and see the sensor data in real time.

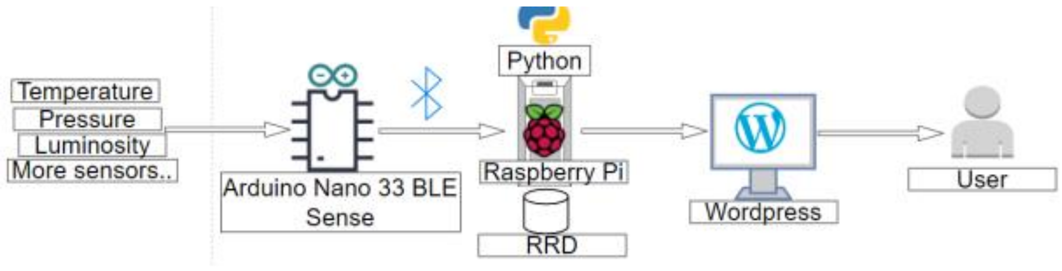


Figure 3. Architecture of the work.

3.2 Sending and Receiving Sensor Data

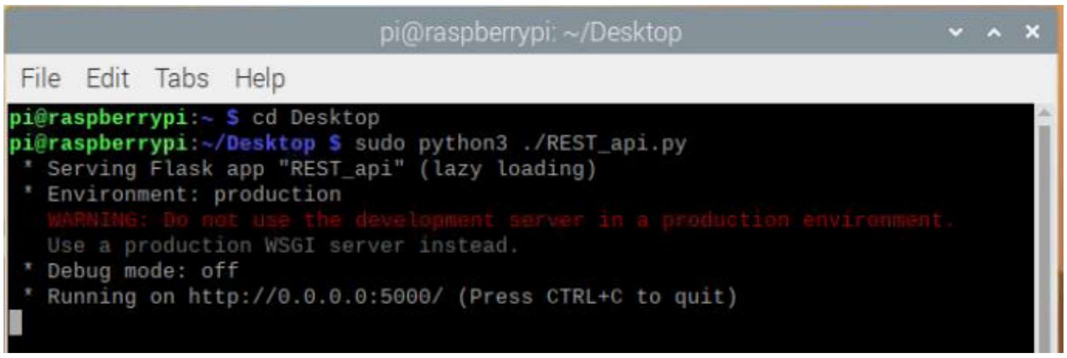
Sending and receiving sensor data was made possible using three Python files. These files were created beforehand and were made for this project. The Python files are as follows: create_rrd.py, REST_api.py and BLE_receiver.py. These files were run inside Raspberry Pi.

3.2.1 create_rrd.py

The database is created with create_rrd.py. This needs to be done only once if there are no changes made.

3.2.2 REST_api.py

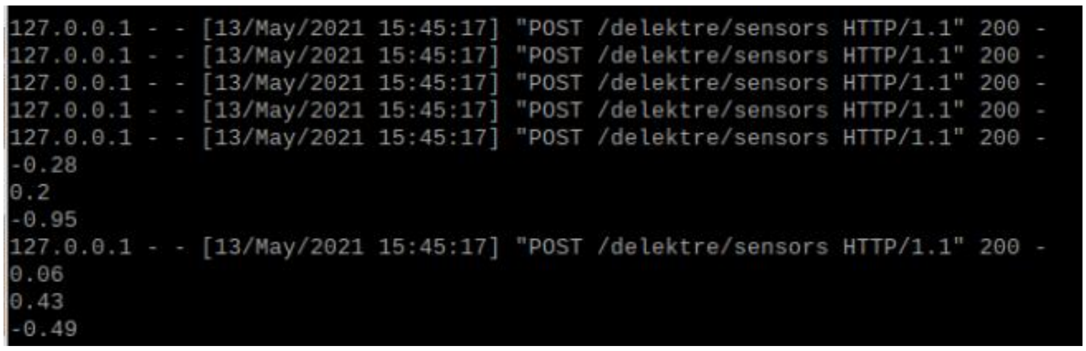
REST_api.py is used for communication. Information is sent or received from the database. How the communication is established can be seen in **Figure 4**.



```
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ sudo python3 ./REST_api.py
 * Serving Flask app "REST_api" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```

Figure 4. Establishing communication.

With a POST request, data is sent to API (Application programming interface) from where API stores it to the database. With GET request API receives data from the database and sends it to WordPress. Sensor data being sent is shown **Figure 5**.

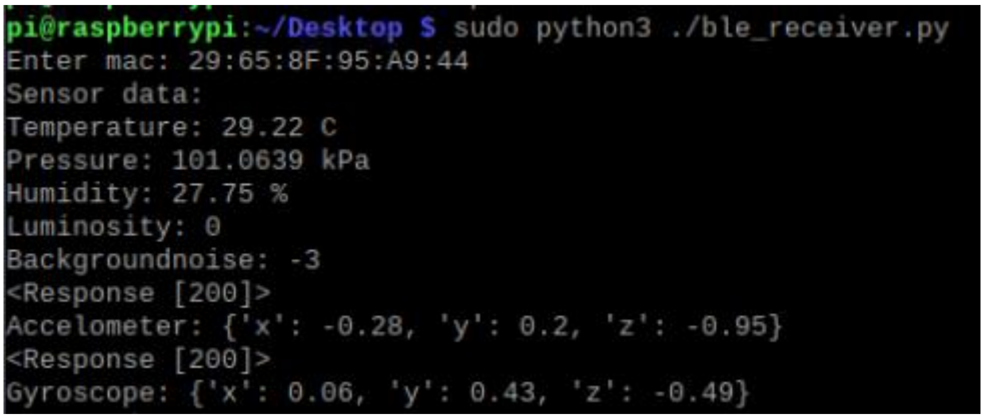


```
127.0.0.1 - - [13/May/2021 15:45:17] "POST /delektre/sensors HTTP/1.1" 200 -
127.0.0.1 - - [13/May/2021 15:45:17] "POST /delektre/sensors HTTP/1.1" 200 -
127.0.0.1 - - [13/May/2021 15:45:17] "POST /delektre/sensors HTTP/1.1" 200 -
127.0.0.1 - - [13/May/2021 15:45:17] "POST /delektre/sensors HTTP/1.1" 200 -
127.0.0.1 - - [13/May/2021 15:45:17] "POST /delektre/sensors HTTP/1.1" 200 -
-0.28
0.2
-0.95
127.0.0.1 - - [13/May/2021 15:45:17] "POST /delektre/sensors HTTP/1.1" 200 -
0.06
0.43
-0.49
```

Figure 5. Sensor data being sent to the database (Numbers -0.28, 0.2, -0.95, 0.06, 0.43 and -0.49 being data values).

3.2.3 Ble_receiver.py

In Ble_receiver.py Arduino Nano is a server and Raspberry Pi is a client. Data is received from Arduino and sent to API with a POST request. Measured data with Arduino can be seen in **Figure 6**.



```
pi@raspberrypi:~/Desktop $ sudo python3 ./ble_receiver.py
Enter mac: 29:65:8F:95:A9:44
Sensor data:
Temperature: 29.22 C
Pressure: 101.0639 kPa
Humidity: 27.75 %
Luminosity: 0
Backgroundnoise: -3
<Response [200]>
Accelometer: {'x': -0.28, 'y': 0.2, 'z': -0.95}
<Response [200]>
Gyroscope: {'x': 0.06, 'y': 0.43, 'z': -0.49}
```

Figure 6. Measured data from Arduino Nano.

3.3 Getting Familiar with WordPress and Elementor

Even though WordPress is advertised as an easy-to-use website building tool, watching video tutorials of WordPress and Elementor made learning much faster. Building a website with Elementor consists of three main blocks: sections, columns and widgets. Sections are the largest blocks and within sections there are columns and within columns there are widgets.

Creation with Elementor starts by adding a section; this is base for the webpage. Then Columns are added. Columns are inside the Section and to define where widgets can be added, the right type of column has to be chosen. Finally, widgets can be added. Widgets can be, for example the main title text or something that has a function when clicked for example a button.

3.4 Planning the Website

Planning the website started by figuring out how the requirements would be met: what kind structure and features the website should have, how to get the sensor data from the database and see it in a chart form. Also, how to make charts self-updating in real time had to be determined. The structure and features desired for the website were sketched on paper. JavaScript was chosen as the method to get sensor data from database. The simple architecture is shown in **Figure 7**.

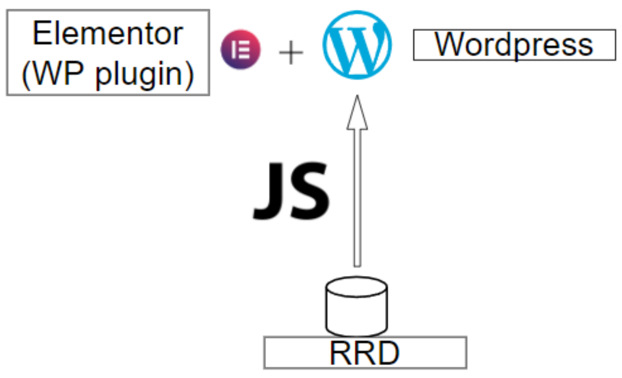


Figure 7. Simple architecture figure showing Elementor and JavaScript.

4 IMPLEMENTATION

The project started by installing a LAMP server and WordPress to Raspberry Pi and installing necessary plugins for WordPress. Self-updating charts and website were created, as well. VNC Viewer was used to remote control Raspberry Pi. The architecture is shown in **Figure 8**.

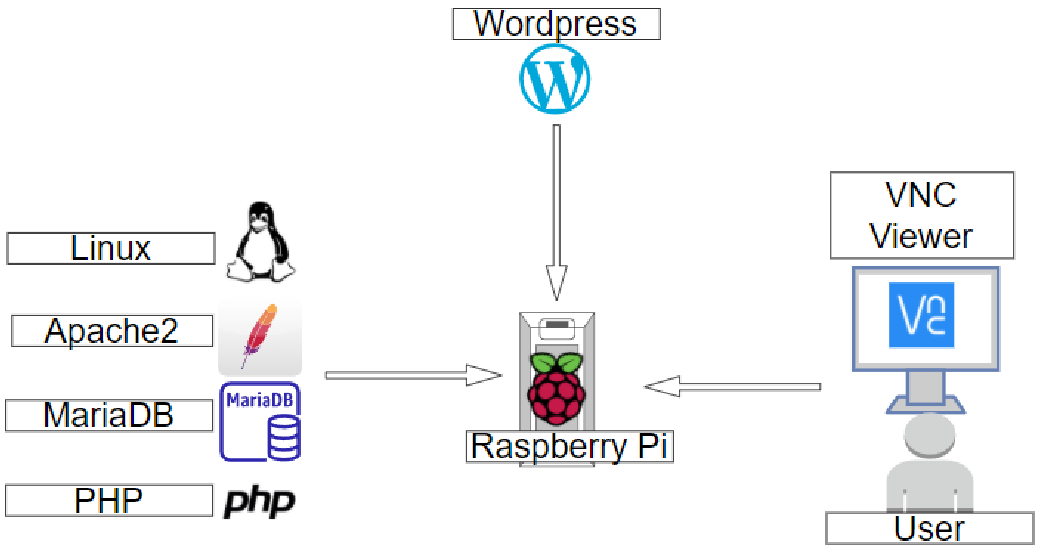


Figure 8. Architecture of LAMP server and WordPress within Raspberry Pi and User connecting with VNC Viewer.

4.1 Installing LAMP Server

A LAMP server is a collection of open-source code programs and in order to run WordPress on Raspberry PI a LAMP server is required. As shown in **Figure 7**, LAMP consists of Linux, Apache2, MySQL and PHP. This is also called LAMP Server Stack and is the foundation for Linux hosted websites. Programs required for LAMP server can be seen in **Figure 9**.

Figure 9. LAMP server programs.

4.1.1 Installing Apache2

Apache is the most commonly used web server on Linux systems. Installing Apache2 can be done with the following command:

```
sudo apt install apache2 -y
```

4.1.2 Installing PHP

PHP (PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language. Installing PHP can be done with the following command:

```
sudo apt install php7.4 php7.4-mysql php-common php7.4-cli
php7.4-json php7.4-common php7.4-opcache libapache2-mod-
php7.4
```

4.1.3 Install MariaDB

MariaDB is a fork of MySQL and a good replacement for it. In this thesis MariaDB was used, and it can be installed with a following command:

```
sudo apt install mariadb-server php-mysql -y
```

4.2 Installing WordPress

Installing WordPress takes place by navigating to the following folder:

```
cd /var/www/html/
```

and deleting index.html with the sudo rm command.

After that the latest copy of WordPress can be downloaded with the following commands:

```
sudo wget http://WordPress.org/latest.tar.gz
```

and

```
sudo tar xzf latest.tar.gz
```

Then the files from WordPress folder need to be moved to HTML folder with the following command:

```
sudo mv WordPress/* .
```

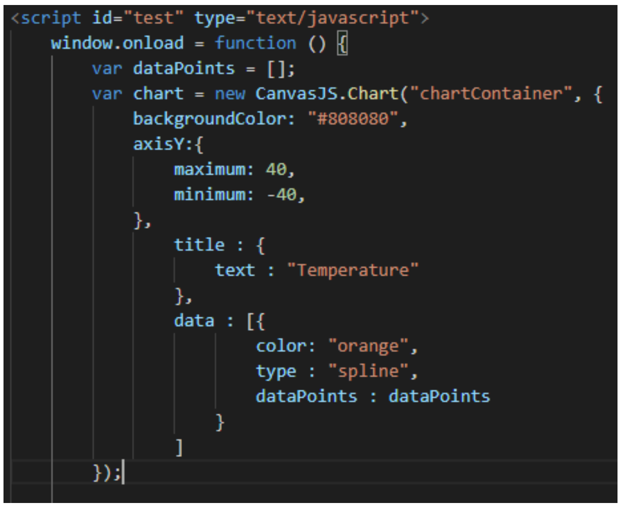
Then it is required to change the ownership of web pages folder and files to Apache2:

```
sudo chown -R www-data:www-data /var/www/html
```

4.3 Creating Code for Charts

The code for uploading the charts was created using HTML and JavaScript. A simple way to understand how these languages work together is to think HTML being a structure (human or tiger) and JavaScript being movements (handshake, jumping).

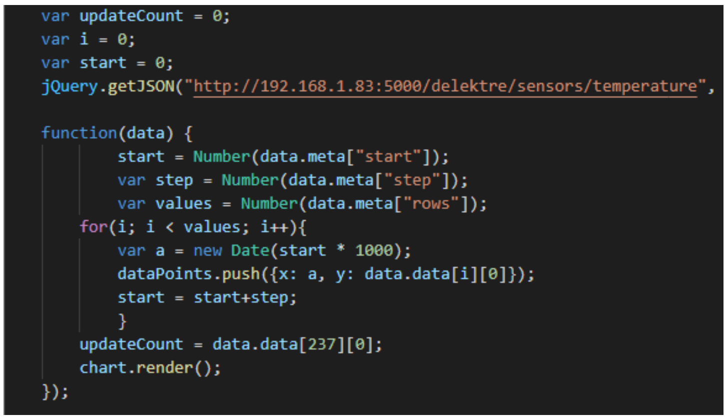
The code can be divided in to three sections. The first section is structure of the chart and all cosmetic additions to it. As can be seen in the **Figure 10** the charts color, axis length and type are determined.



```
<script id="test" type="text/javascript">
    window.onload = function () {
        var dataPoints = [];
        var chart = new CanvasJS.Chart("chartContainer", {
            backgroundColor: "#808080",
            axisY:{
                maximum: 40,
                minimum: -40,
            },
                title : {
                    text : "Temperature"
                },
                data : [{
                        color: "orange",
                        type : "spline",
                        dataPoints : dataPoints
                    }
                ]
        });
```

Figure 10. Chart structure is defined.

The second section of the code is about receiving information from the database. jQuery.getJSON is used to get data from the database. Using *for()* loop it is possible to go through the database and get all the desired values one by one. This section of the code can be seen in **Figure 11.**

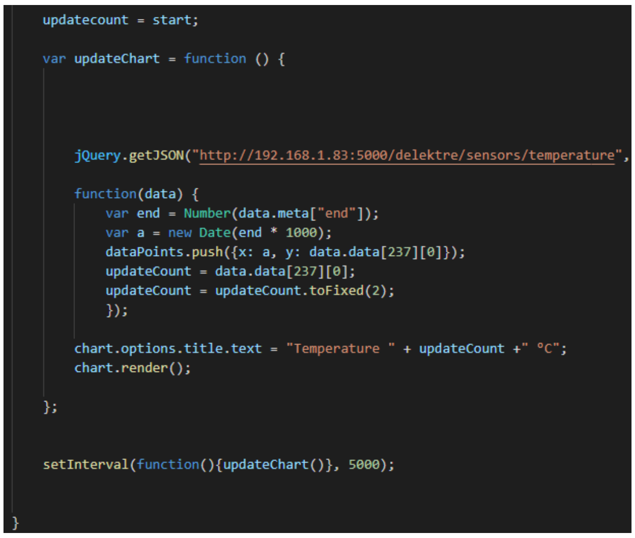


```
    var updateCount = 0;
    var i = 0;
    var start = 0;
    jQuery.getJSON("http://192.168.1.83:5000/delektre/sensors/temperature",

    function(data) {
            start = Number(data.meta["start"]);
            var step = Number(data.meta["step"]);
            var values = Number(data.meta["rows"]);
        for(i; i < values; i++){
            var a = new Date(start * 1000);
            dataPoints.push({x: a, y: data.data[i][0]});
            start = start+step;
            }
        updateCount = data.data[237][0];
        chart.render();
    });
```

Figure 11. Getting values from the database.

The third section is for chart updating. As it can be seen in **Figure 12** *Updatechart()* gets new data every 5 seconds. The latest data is picked and saved to the array.



```
    updatecount = start;

    var updateChart = function () {



        jQuery.getJSON("http://192.168.1.83:5000/delektre/sensors/temperature",

        function(data) {
            var end = Number(data.meta["end"]);
            var a = new Date(end * 1000);
            dataPoints.push({x: a, y: data.data[237][0]});
            updateCount = data.data[237][0];
            updateCount = updateCount.toFixed(2);
            });

        chart.options.title.text = "Temperature " + updateCount +" °C";
        chart.render();

    };


    setInterval(function(){updateChart()}, 5000);


}
```

Figure 12. Chart gets updated every 5 seconds.

In **Figure 10**, **Figure 11** and **Figure 12** excerpts from the temperatures code are shown, but each sensor that was measured for this project had to have their own individual code. The codes themselves did not differ that much from functionality. Text and units had to be changed in the code for the corresponding sensor. For example, temperature would have Celsius and background noise would have decibel.

4.4 First Version

The first version of the webpage had a startup view of homepage that can be seen in **Figure 13** and underneath there was sensor selection menu seen in **Figure 14**.

Figure 13. Homepage of the website.

At the sensor selection menu there were buttons and each button would take the user to a new webpage where self-updating chart of the sensor can be seen.

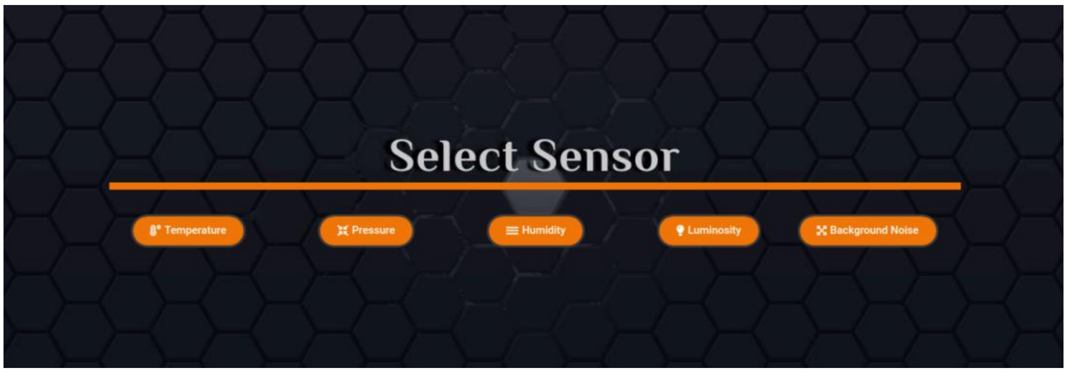


Figure 14. Sensor selection menu.

Each chart had the title of the corresponding sensor. It was also possible to see data from the past hour. There was also a button to go back to the homepage. An example of chart can be seen in **Figure 15**.

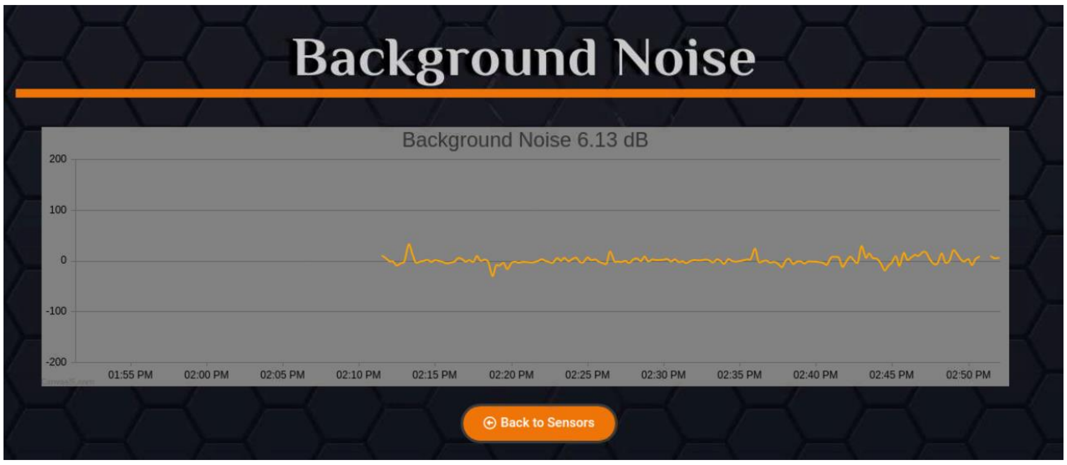


Figure 15. Chart of Background Noise.

4.5 Second Version

Later it was decided that the webpage would be used to measure only temperature, humidity and luminosity. Also, the webpage theme would be about chili farming in a balcony. So the background was changed to more fitting. In **Figure 16** the new homepage can be seen.

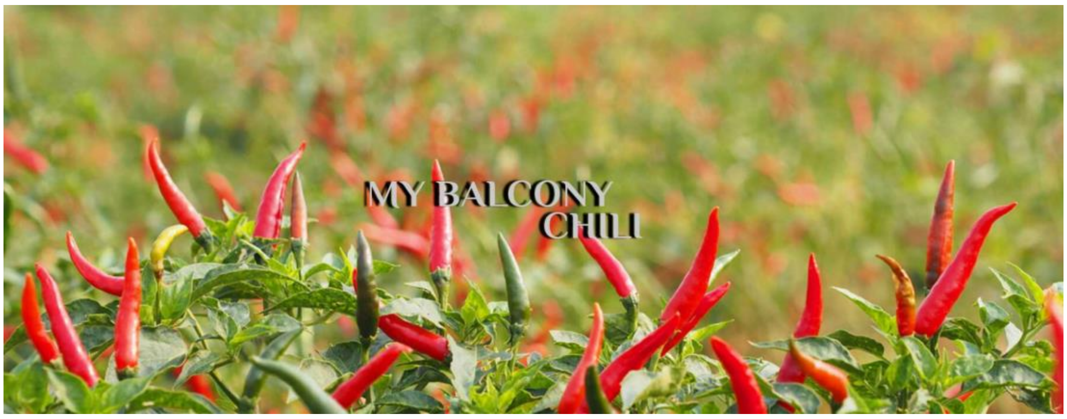


Figure 16. New homepage.

In **Figure 17** the new sensor selection menu can be seen. Since the theme was about chili farming, the background of the webpage was a picture of chilis. The colour of buttons were changed to red, as well.

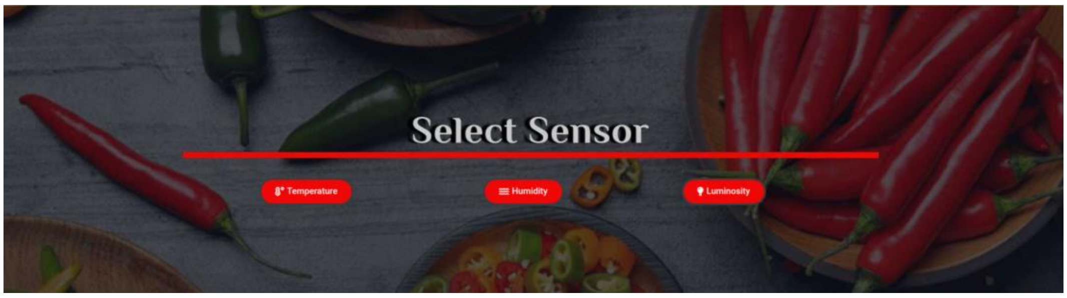


Figure 17. New sensor selection menu.

The lines of the charts were changed to red, as well to, fit with the theme. This gave the website a more pleasant look. In **Figure 18** the tracking of luminosity sensor data can be seen.

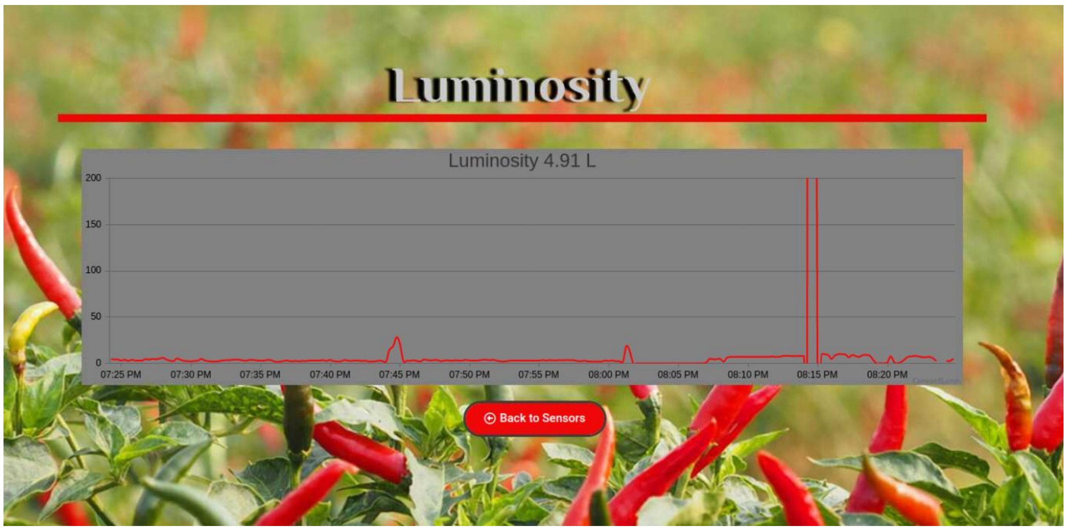


Figure 18. Luminositys sensor data.

4.6 Final Version

It was decided that for the demo video of the working webpage all three sensors that were being measured would be on the same page. This meant that all buttons from the webpage were removed. All three sensor data charts can be seen in **Figure 19**.

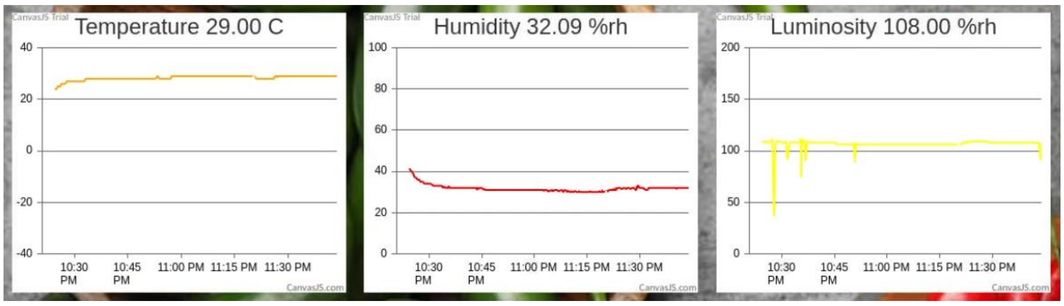


Figure 19. Temperature, humidity and luminosity on the same page.

4.7 Timeline

Figure 20 shows the timeline of the project. After the first team meeting the next part was to setup Raspberry Pi, Arduino, install necessary softwares and test the connection between Arduino and Raspberry Pi. The website was planned and the demo version was created. The team meeting was held after each new version of the project was completed.

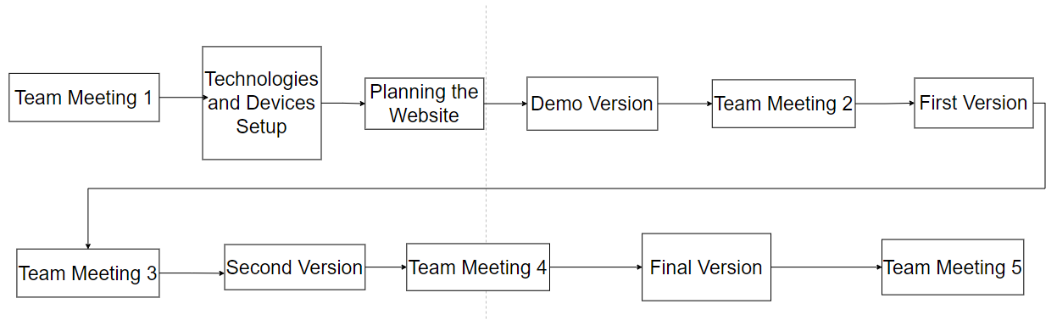


Figure 20. Timeline of the project.

5 DEVELOPMENT PHASES

Some of the challenges faced during development of the project and how they were handled are discussed here. For example, how the demo version was created and what kind of problems it had, or why Visualizer and WpDataTables were not used. It is also explained what problems there were with the charts.

5.1.1 Demo Version

The demo version of the project was created using Elementor following along tutorial and all the charts were uploaded using plugin for WordPress called Visualizer. The way charts were uploaded from Visualizer was using an option to upload data directly from URL and a chart was created. Data from each sensor was uploaded to one page (home page) as a form of bar chart as a proof of concept that the receiving of sensor data was working. However, refreshing the web page was required to monitor most recent data and Visualizer had a limited number of different types of charts for free. For these two major reasons the next version of the project had to have a different approach to upload charts.

5.1.2 Use of Visualizer

Visualizer is a plugin for WordPress that was used during the demo version of the project. It made the creation of charts easy by uploading the charts from URL. However, the page had to be manually refreshed each time new measurement from sensors wanted to be seen on the chart. Because of this Visualizer was quickly abandoned.

5.1.3 Use of WpDataTables

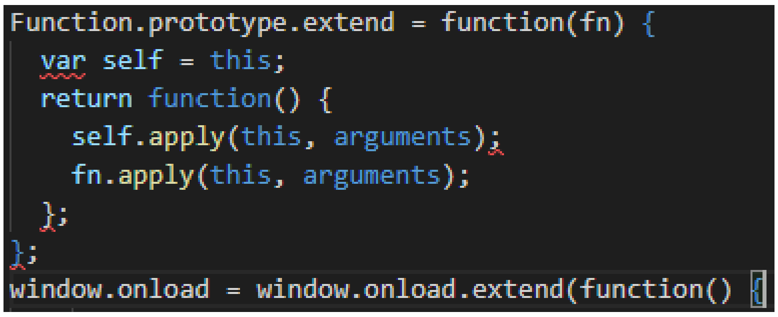
Using WordPress plugin called wpDataTables was also considered since it is marketed as being the best-selling WordPress table plugin. However, this idea was discarded quickly because wpDataTables required a specific file format to work.

5.2 Uploading Charts

Uploading charts to WordPress did not work at the beginning. When the code was launched on the home computer using Visual Studio Code, the charts were working perfectly but in WordPress the chart would show up on the web page after many page refreshes (sometimes over 20+ refreshes). The solution was simple but hard to find but eventually the problem was solved by changing ”$” symbol to ”jQuery”. This was because “$” is not associated with jQuery.

5.3 Multiple Charts on Same Page

When the final version of the webpage was being done, a problem occurred with multiple charts showing on the same page. If more than one chart was added to the page, only the first one would be seen. This, however, was fixed by adding the following lines inside the code seen in **Figure 21**.



```
Function.prototype.extend = function(fn) {
  var self = this;
  return function() {
    self.apply(this, arguments);
    fn.apply(this, arguments);
  };
};
window.onload = window.onload.extend(function() {
```

Figure 21. Lines that were added at start of the code.

6 CONCLUSION

This thesis was more of a proof of concept that this method can be used for many other sensors if required. For example, one could track one's own heart rate with the right kind of sensor/equipment. Even though the rise of IoT can make everyday life easier and make people feel more safe and secure, it can also make devices more vulnerable if connections are not secure enough. In this project making secure connections was not the main priority due to data being harmless. The project met all the requirements. For me the hardest part of this thesis was to make codes self-updating without the need of manually refreshing webpage. My personal knowledge expanded in WordPress development and also in JavaScript. I think potential for this kind of system is great, but it would need more updating. In the future WordPress could have control system where different sensors can be managed. This would add scalability if for example there were 100 sensors. If the information handled is sensitive, investing in security would be logical.

REFERENCES

Arduino.cc, 2021 a. Arduino Nano 33 BLE Sense. Accessed 02.06.2021 https://store.arduino.cc/arduino-nano-33-ble-sense

Arduino.cc, 2021 b. ArduinoBLE library. Accessed 02.06.2021 https://www.arduino.cc/en/Reference/ArduinoBLE

Bluetooth, 2021 a. LEARN ABOUT BLUETOOTH Audio Streaming. Accessed 02.06.2021 https://www.bluetooth.com/learn-about-bluetooth/solutions/audio-streaming/

Bluetooth, 2021 b. LEARN ABOUT BLUETOOTH Data Transfer. Accessed 02.06.2021

https://www.bluetooth.com/learn-about-bluetooth/solutions/data-transfer/

Bluetooth, 2021 c. LEARN ABOUT BLUETOOTH Location Services. Accessed 02.06.2021 https://www.bluetooth.com/learn-about-bluetooth/solutions/location-services/

Bluetooth, 2021 d. LEARN ABOUT BLUETOOTH Device Networks. Accessed 02.06.2021 https://www.bluetooth.com/learn-about-bluetooth/solutions/device-networks/

Cirani, S., Ferrari, G., Picone, M., Veltri, L. 2018. Internet of Things. United States. John Wiley and Sons Ltd.

Hostsinger Tutorials, 2021. What is HTML? The Basics of Hypertext Markup Language Explained. Accessed 02.06.2021 https://www.hostinger.com/tutorials/what-is-html

Python, 2021. What is Python? Executive Summary. Accessed 02.06.2021

https://www.python.org/doc/essays/blurb/

Raspberry Pi, 2021. Raspberry Pi 3 Model B+. Accessed 02.06.2021

https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/