University College London

ELEC0033: Internet Of Things

Final project report

Authors:

Sreethyan Aravinthan
SN: 17034996

Constantina Papavasileiou
SN: 17001295

Mindaugas Jarmolovičius
SN: 17139494

Miriam Robledo
SN: 17080614

May 4, 2020

Contents

1 Executive Summary **2**

2 Business Case **2**
2.1 Business Concept . . . 2
2.2 Value Chain . . . 3
2.3 Key Competitors . . . 4
2.4 Barriers to Entry . . . 5
2.5 Cost Breakdown . . . 5
2.5.1 Research & Development . . . 6
2.5.2 Product . . . 6
2.5.3 Installation & Maintenance . . . 6
2.6 Sales Strategy . . . 7
2.6.1 Packages . . . 7
2.6.2 Future . . . 7
2.7 Financing . . . 7
2.7.1 Seed Capital . . . 7
2.7.2 Seed Capital . . . 7

3 Design Methodology and Implementation **8**
3.1 The Problem . . . 8
3.2 Background Theory . . . 9
3.2.1 IoT Stack . . . 9
3.2.2 Communication Protocol for sensor data . . . 10
3.2.3 Methods for checking changes in state for devices connected to processor . . . 10
3.2.4 Low Power Mode . . . 11
3.2.5 Wi-Fi Activity . . . 11
3.2.6 Cryptography . . . 11
3.2.7 Checksum . . . 12
3.3 System Design . . . 12
3.3.1 Layer 1 . . . 12
3.3.2 Layer 2 . . . 15
3.3.3 Layer 3 . . . 16
3.4 Metrics for evaluating the performance . . . 16

4 Experimental Results **17**

5 Engineering Trade-offs **17**
5.1 Sensor selection . . . 17
5.1.1 Temperature and Humidity . . . 17
5.1.2 Air Quality . . . 17
5.2 Cryptography Choice . . . 18
5.3 Cloud computing . . . 18
5.4 Wireless Power consumption . . . 19

Appendices **20**

1 Executive Summary

University Campuses offer a wide range of coffee shops and cafeteria for its faculty and students that are usually run by the same catering service and within a short distance from each other. The university body uses these establishments to buy food or coffee but also as a place to gather with acquaintances, hold meetings or work. Because of the high demand for this service, the university's cafés tend to get quite crowded and become unpleasant for those who were planning to sit in the café. Furthermore, the long queues can force students and staff to be late to their lectures or commitments. This is the problem that Coffee Time, an Internet of Things application, is trying to solve. At the moment student's decision of which café to choose is limited by proximity to lecture facilities and menu choice, but what if there were an app that would allow them to identify which café will have the shortest queue or the best ambience and experience to sit down and work or meet with people. This application would not also help students and faculty make the best decision about what café to go in the moment, or where to meet the next day but also it would help catering services organise their staff's schedules in accordance with peak times and predictions.
This report details the creation and commercialisation of Coffee Time. First, a business case will explain the potential of the product in the market by explaining the business concept and other considerations such as financial requirements, profit, competitors and barriers to entry.
The second chapter will focus on the design methodology and implementation for Coffee Time. After defining the problem, background theory will be examined to justify design decisions and finally, the overall system design for this project will be explained.
The final chapter will discuss the engineering trade-offs that had to be considered when developing the project and what benefits were gained in term of performance at the expense of other performance metrics.
Our report concludes that the Coffee Time IoT system solves an existing and not previously tackled problem filling in a niche in the market. We suggest as future work for the system to be further developed and commercialised.

2 Business Case

2.1 Business Concept

Coffee time aims to help companies better organise their staff, offers and resources to accommodate more efficiently to student's usage times. Simultaneously, Coffee Time aids students select what café they want to go to, spend less time in queues and have an overall better experience in the café therefore improving student satisfaction.
The business concept for Coffee Time can be summarised under the Business Model Canvas [REF], a strategic management tool that describes the product's value proposition, infrastructure, customers, and finances.

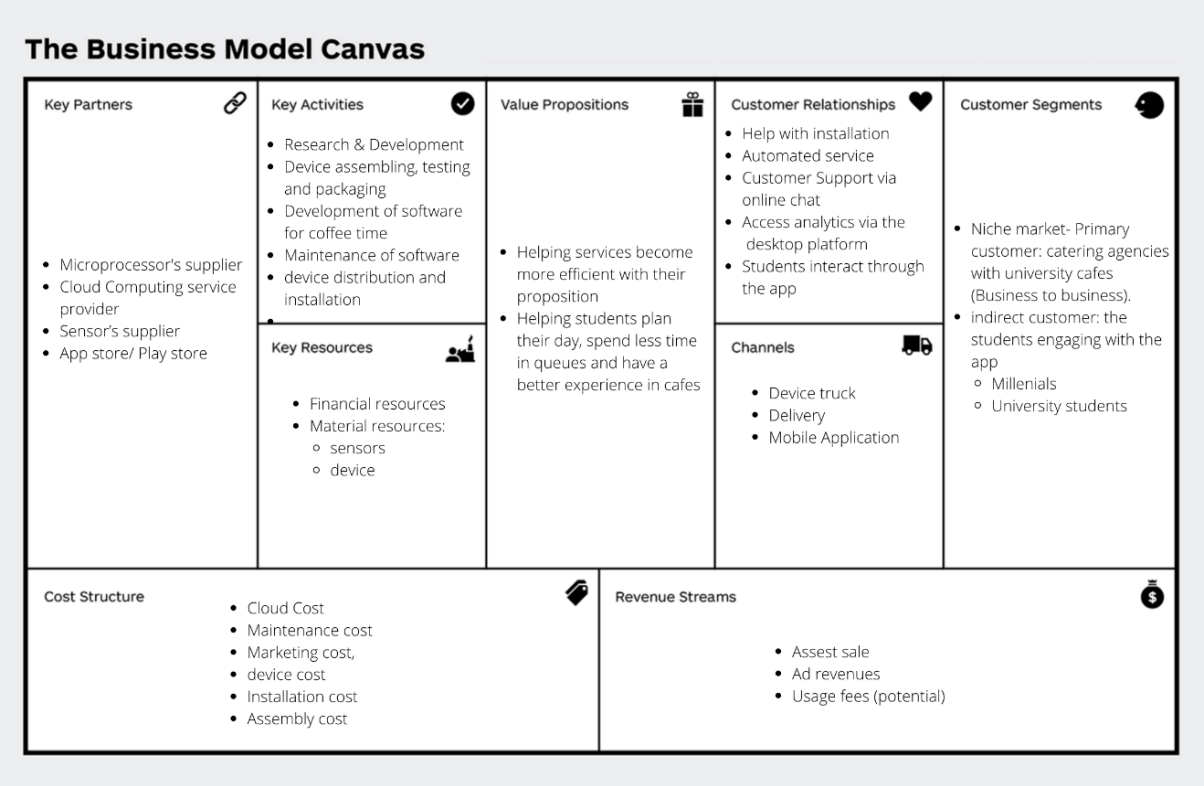


Figure 2.1.1: *The Business Model Canvas*

2.2 Value Chain

By examining the value chain for IoT systems we determined at what steps the Coffee Time business could create additional value in accordance with its resources available. It was decided that the hardware and connectivity would be outsourced, however a smart object would be created to sense data in the cafes, in the system enablers step an app and desktop platform will be developed and offered to the customers including data analytics of the data gathered by the coffee time smart object and predictions for ambiance in the near future. Lastly, while packaging, provisioning and branding will be outsourced, coffee time will handle the customer relationships, billing and fault handling.

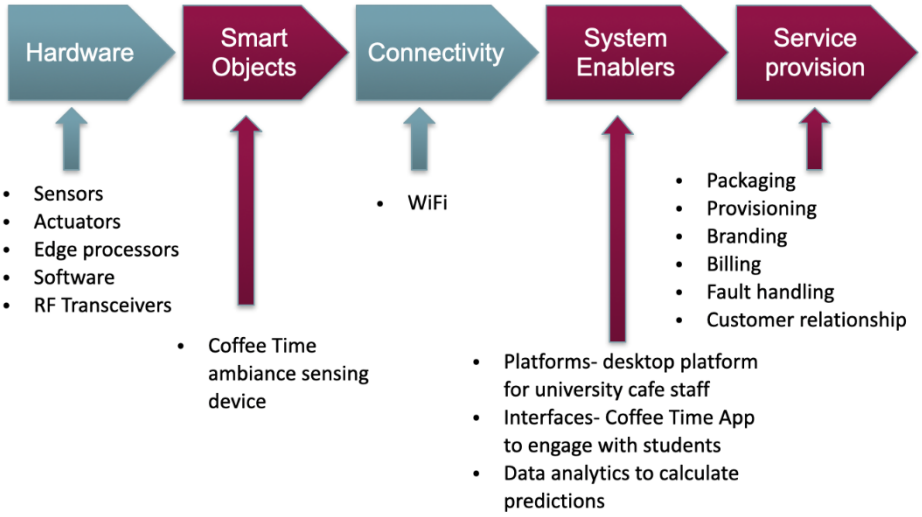


Figure 2.2.1: *Value chain diagram*

2.3 Key Competitors

There is no competitor in the market that focuses exclusively in improving the ambiance of cafes, however there are a wide range of IoT retail or home devices that measure similar features to Coffee Time but that are however, marketed to different customers.

Nest Learning Thermostat

Nest Learning Thermostat is a Google home IoT device that measures the temperature in the house, memorizes patterns and automatically lowers the temperature when the house is not being used in order to minimize energy costs [1].

Footbot Air Quality Monitor

Footbot is a smart building IoT device used in workplaces and homes to ensure quality indoor air. The product measures chemical pollutants, particular matter, humidity and temperature and uses artificial intelligence to optimize HVAC systems in order to save energy and improve comfort [2].

Real-time people counting system SensMax WebKit TS

This product is marketed to retail stores, shopping centres and outdoor facilities. It is a wireless real-life people counter to monitor how many people are in a store during the day. It can use the collected data to predict how many people will be in the store the next day [3].

2.4 Barriers to Entry

The Porter's Five Forces framework will analyse the competitive rivalry in the market Coffee Time will enter in order to identify any barriers to entry and identify the intensity of the competition to predict the likelihood of profitability Coffee Time will obtain (Porter, 2008). Each force is given a number from 1 to 5 to determine the level of competitiveness.

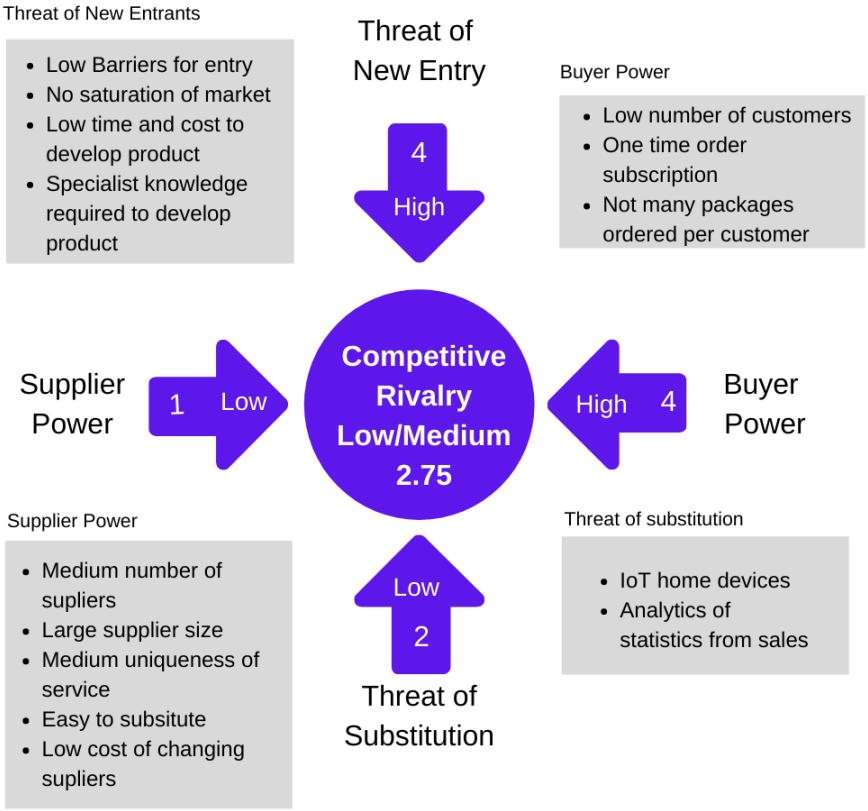


Figure 2.4.1: *Porter's five forces framework*

Overall, the competitive rivalry is between low and medium. The threat of entry is high as it is a new market that has not been very explored and there is no saturation and barriers are low, however it does require specialist knowledge to create the product. Buyers have a high power, since there is a limited number of universities in the UK and the product only needs to be bought one time. However, there is a low threat of substitution as it is very unlikely to find another service that improves café ambiance. Lastly supply power is low as there are many providers for sensors and microprocessors and it would be easy to substitute a provider at a low cost.

2.5 Cost Breakdown

Coffee time is integrating the software and the hardware Assembly of the device is outsourced to a soldering and assembling plant.

2.5.1 Research & Development

The R&D phase involves the realisation of the final product, testing, prototype building and business plan development. The resulting cost is an aggregate of the work hours spent and the material costs (hardware and software) of the prototypes and testing. On completing the R&D phase, a final product is conceived, and its prototype created.

Research and Development	Quantity	Amount
Research (market, technology)	Workhours cost (30h)	£ 600.00
Software development	Workhours cost (8h)	£ 160.00
Application development	Workhours cost (15h)	£ 300.00
Materials (hardware components)	Purchase cost	£ 1,000.00
	Total	**£ 2,060.00**

Table 2.5.1: *R&D aggregate costing table.*

2.5.2 Product

The product consists of the device and the public mobile application. The cost of the device can be split into hardware cost and software cost outlined in Table 2.5.2.

Description	Model Name	Amount
Microcontroller, WiFi Occupation	ESP8266	£ 5.51
Temperature & Humidity	BME280	£ 7.09
Air Quality	SGP30	£ 4.71
Housing Case with Screen	-	£ 8.00
Wall mount	OtS[1]	£ 2.00
	Total	**£ 27.31**

Table 2.5.2: *Hardware Costing table for one unit.*

2.5.3 Installation & Maintenance

In addition to the device cost (which is per unit) there are costs stemming from the installation and maintenance of the products. Please note these are estimated values.

Description	Quantity	Amount
Installation	per café	£ 30.00
Cloud Storage[2]	12T per year	£ 150.00
Device Maintenance	per device per year	£ 1.00
Application Maintenance (15%)[2]	per year	£ 45.00

Table 2.5.3: *Installation and Maintenance Costing table.*

The installation cost entails the device delivery, mounting on the wall and device-application setup. The devices will be delivered by the employee performing the installation and hence these tasks' cost translates to work-hours cost.

[1] Off the Shelf item
[2] About 15% of initial development cost

2.6 Sales Strategy

The pricing mechanism used is initially dynamic as based on the market conditions real-time market pricing will lead to higher revenues. Once Coffee time is established and the market more mature it is possible to switch to a fixed pricing model.

2.6.1 Packages

Packages are designed for and promoted to coffee chains. They are designed to be enticing for their value and entail high client investment. Their aim is building initial and long-term partnerships with the industry, while serve as promotional material to the product as it is seen more by prospective clients. It is expected that larger coffee chains will opt for higher tier value packages as they will be more advantageous compared to a lower quantity standard price product.

2.6.2 Future

As the company's presence in the market is maturing, it is expected the cost of the product and marketing costs will decrease. The company's outreach will be boosted by the market reputation and the existing clients' product use; hence the marketing efforts can be reduced. On increasing demand by the addition of new clients, the cost of purchasing parts from suppliers susceptible to decrease as orders increase in size. As a result, the profit margin will increase and can be invested to cover previous debt and/or expansion strategies. The packages contents and value are due to be reconsidered in line with the new revenue streams.
Return on Investment:

$$Rol = \frac{Net\ return\ on\ investment}{Net\ cost} \times 100$$

2.7 Financing

2.7.1 Seed Capital

As a first step, a founding capital to develop the idea is required. This high-risk funding is aimed at conducting market research, product research, development and a business plan until an initial revenue is generated. The amount should cover the Research and Development costs. The source of seed capital is provided by savings of the founding group, or angel investors in the form of a loan.

2.7.2 Seed Capital

Once the company has gained a sales record, additional funding, called Series A financing can be sought through the following channels, in exchange of partial ownership [4]. Coffee time's strategy should at this point, focus on longer terms goals.

- Bank Loan
- Venture Capital
- Crowdfunding

This funding can be used towards scaling up or expanding the products to different markets.

3 Design Methodology and Implementation

3.1 The Problem

Universities with medium or large campuses offer many cafes to their students. For example, UCL's has the George Farha Café, Engineering Café, IoE, Gordon's Café, Bloomsbury, Print Room, to name a few. These establishments are open during working hours to the student and academic body and are used to grab a drink on the go due to their proximity to lecture rooms and libraries, or as a place to sit down and have a break, eat lunch or conduct group meetings. As a result, these places tend to get quite crowded, creating long queues, and a loud ambiance therefore creating an unpleasant area to be in. UCL students usually have limited time to buy coffee and just cannot go through all the cafés until they find one that has the shortest queue. The problem is that the students and staff do not know how busy the cafés are at a given time. Similarly, the cafe does not know when students will be most free during the day and can be understaffed at peak times and over staffed during more calm hours. To address this problem we have developed Coffee Time, an IoT package that offers a phone application (app) showing how busy cafés are or will be in near future. An IoT device collects multiple metrics from café's environment which is sent to the cloud system. Inside the cloud raw information is analysed and processed to be represented on the app.

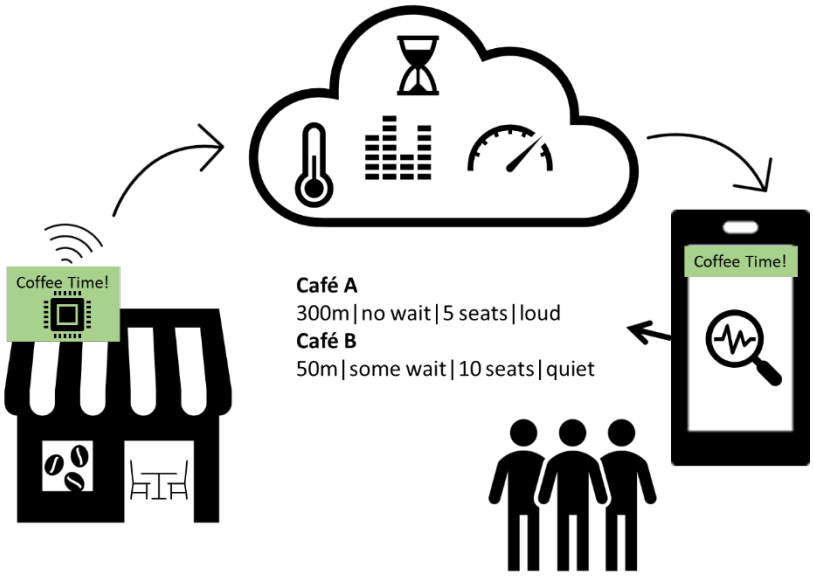


Figure 3.1.1: *A simplified overall system illustration.*

Some previous studies to detect crowds and people count have been proposed by using complicated computer vision to analyse people count [5]. However, such a solution would require complex and sophisticated programs, and powerful hardware, which brings other problems with battery life and overall project expenses. Furthermore, this project was not only aiming to measure how crowded an establishment was but also the overall ambiance of the place. Since ambience is quite an abstract concept, it was important to define what factors can determine and measure the ambience of an environment. These factors were determined through the use of literature and can be seen on Figure 3.1.2.

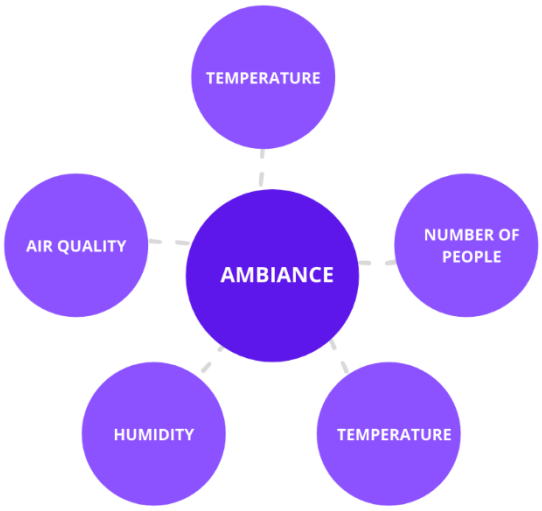
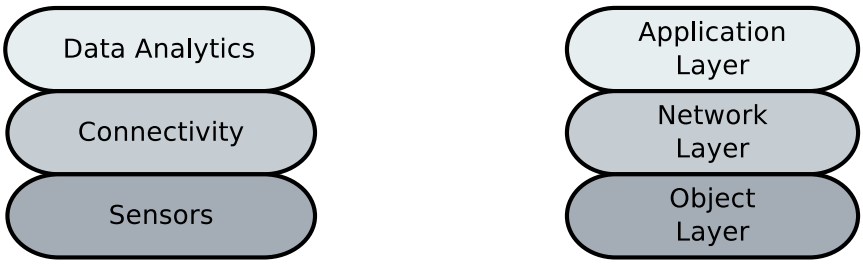


Figure 3.1.2: *Composition of environment ambiance in a café.*

Therefore, Coffee Time will be measuring air quality which includes CO_2, TVOC, H_2, Ethanol, Temperature and Humidity readings; and WiFi activity. We expect to see correlation between air quality and café's activity due to CO2 and TVOC increase from people and the coffee making process. WiFi activity should correlate with amount of people staying inside and using their phones or laptops.

3.2 Background Theory

3.2.1 IoT Stack



(a) *Model 1* ***(b)*** *Model 2*

Figure 3.2.1: *IoT conceptual models*

There are different versions for the IoT stack and many of these versions will agree in many areas. However, there is no formal standard for the IoT stack. Figure 3.2.1a and Figure 3.2.1b are two such examples of the IoT stack. These models may seem similar but they are different. Below an explanation into both the models will be done and which model was chosen for this project.

Model 1
This model contains three layers. The bottom layer is used for the sensors connected to the IoT device. The middle layer is for connectivity. Finally, the top layer is for data analytics. In this model each layer is specific as to what should take place in each layer.

Model 2
This model is more generalised or abstracted than the other model. Like the other model this model also has three layers. The top layer of the model is for cloud services such as real time streaming or database storage. This also encapsulates data analytics. Thus called the application layer. The middle layer is for the gateway. The bottom layer is known as the object layer and it can be for sensors or motes.

Chosen Model
For this project the model 1 was chosen as the IoT stack. This was more specific for this project as we looked at the sensors that can be used to get the data for each metric. The connectivity layer looks at sending the data to the server over Wi-Fi. Finally, data analytics will be done on the collected data to see if there is any correlation and to give predications as to which cafe to go to.

3.2.2 Communication Protocol for sensor data

SPI — Serial Peripheral Interface benefits from full duplex communication and high throughput, up to 10Mbps. It is excellent for peripheral such as screens, ADC, flash memory and similar low latency and high throughput requiring components. However, it requires four wires (six with power and ground) to communicate, and one extra wire for each additional peripheral.

I^2C — Inter-Integrated Circuit is only half duplex and operate on master-slave serial bus. Single master can communicate with up to 127 peripherals that connect the same common bus. It has lower throughput of 10-1000kbps and due to nature of half duplex, to get response from a peripheral the master has to send a request first. This is ideal for this project as sensors only require few hundred bits to communicate with microprocessor, and sensor does not send continuous data stream. In addition, it is much easier to set up at hardware level as it only requires all peripherals to be connected with the same two wire bus (four with power and ground).

3.2.3 Methods for checking changes in state for devices connected to processor

Polling and interrupts are two methods that can be used to see if a device has reached a state. These are useful techniques that are used by the microcontroller to detect a change in state for a device that is connected to the processor. Below the two methods will be described.

Polling
Polling checks for the change in state of the connected device by simply checking if the state of the device has changed to the set value in a loop. This section of code ill be a subset of the main program. For example there can be a loop where there will be some code to carry out a certain operation, then code to check if the state has changed and what to do if that is the case and then there can be some other code for some other operation. This entire code will be repeated until an exit condition is met.

Interrupts

Interrupts work differently. Instead of always checking if the state of the external device has changed or not, the processor will carry out the main program but when a change does occur then the processor is interrupted or notified that there is a change. Once the current instruction has finished executing the interrupt is handled by the processor using a Interrupt Service Routine (ISR) and the processor will go back to executing the main program.

Chosen method
For this project interrupts was chosen as the method for seeing if the state of the device has changed. This was because the main program will not have to check continuously if the value has changed, it will be notified when it does. In addition to this polling will have a delay when checking if there is a change to the state of the device since the loop will contain other code which is not responsible for checking the state of the device.

3.2.4 Low Power Mode

3.2.5 Wi-Fi Activity

IEEE 802.11 standard describes that Wi-Fi modules has eight operation modes. Monitor mode allows Wi-Fi module capturing Wi-Fi packets that are transmitted around it. These packets can provide information such as source and destination MAC addresses, data rate, Signal-to-noise ratio (SNR) and a number of other fields about the network. Packets are grouped into three types: management, control and data. Management and control packets are dedicated for coordination, access point authentication and any other operation that does not include user sent data. These packets can be used to detect how many uniquely connected devices are in the surrounding area.

3.2.6 Cryptography

This project uses symmetric encryption which uses one and the same key to encrypt and decrypt a message. As a cryptographic key is flashed into IoT device's ROM, this key is never sent over non-secure media, therefore, no need for asymmetric encryption. An unpredictable random nonce is provided by the server upon new connection as shown in Figure 3.2.2. It is used as an input to the chiper which ensures that identical messages with different nonce would yield a completely different encrypted data. The same nonce value is then reused inside the cloud to decrypt the message that was encrypted at the IoT device. Without such a mechanism a replay attack is possible. If a malicious source captures an encrypted message and resends it to the cloud, the cloud will not be able to differentiate between legitimate and repeated messages.

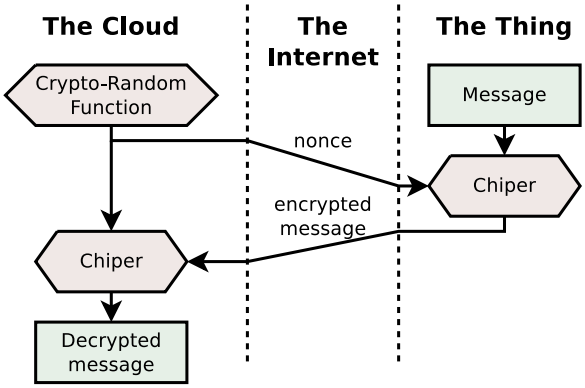


Figure 3.2.2: *Diagram representing cryptographic nonce and encrypted message exchange between the could and the thing.*

3.2.7 Checksum

This project uses cyclic redundancy check (CRC) error-detecting code. This is a very simple and fast algorithm that creates seemingly a random result from provided input. A message can be processed with this algorithm where the result is attached to the message before sending it. When this message is received, it can be processed again with the same algorithm and the calculated result will be compared to attached CRC result. If the results are different then there was an error when the data was transmitted.

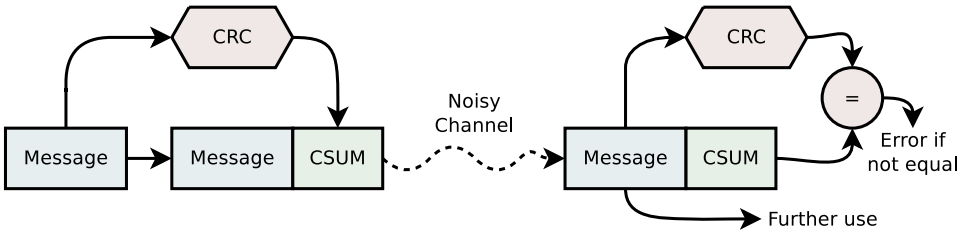


Figure 3.2.3: *Block diagram of CRC checksum logic*

3.3 System Design

This section describes the overall design of the implemented IoT device. The description starts from explaining the implementation of the sensor layer and then moving on to the connectivity layer and finally the data analytics layer.

3.3.1 Layer 1

The IoT core component is `Feather HUZZAH ESP8266` module containing a `ESP8266` microcontroller with built-in Wi-Fi. This module also has a built-in Lithium Polymer (LiPo) battery charging circuit, USB connectivity with serial converter to charge and program microcontroller and linear converter to supply 3.3V rail. The air quality `SGP30` and temperature-humidity `BME280` sensors were both connected via I^2C bus. An additional `IO16` pin was connected to the reset pin, which enables deep-sleep mode wakeup timer. Figure 3.3.1 shows the circuit diagram connecting all the components of the IoT device.

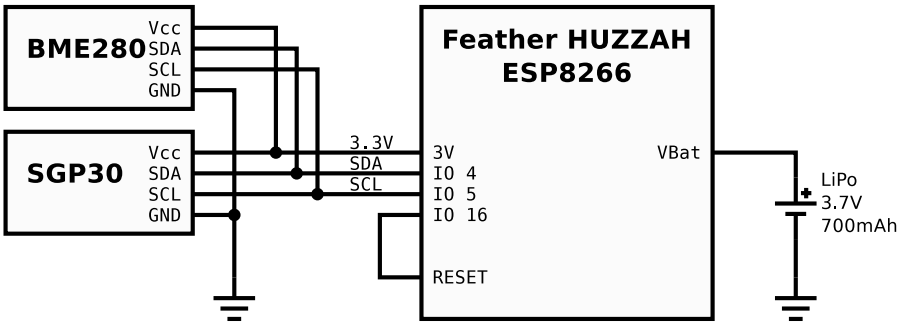


Figure 3.3.1: *Circuit of IoT device.*

With this configuration, even in deep-sleep mode both sensor modules are still powered as 3.3V bus is not affected. This is intentional as both sensors have low-power modes. `BME280` sensor outputs temperature, humidity and pressure readings. This sensor has three modes, sleep which does no operation and is at the lowest power, forced which perform one measurement, stores the results and returns to sleep and normal mode for perpetual cycling of measurements. Additional oversampling and IIR filter settings can be set. The weather monitoring configuration that was recommended by the datasheet was used, this uses forced mode with one sample per minute without oversampling and IIR filter. This achieved reasonable noise and performance with average current consumption of $160nA$.

`SGP30` sensor was equipped to measure carbon dioxide (from 400 ppm), total volatile organic compounds (in ppb range), relative H_2 and ethanol values. After the sensor has been powered it goes into low power sleep mode. When the microcontroller sends a measurement request the high current measurement mode is activated. Internally the sensor takes a sample at 1Hz. This was used to calculate a baseline. During experimentation, it was seen that it takes around 16 seconds since booting to read an accurate value. This lead to the decision of not disconnecting the power from this sensor during sleep as it will not be able to give an accurate value when requested by the microcontroller. This sensor also features an on-chip humidity compensation calculation, however, it requires the absolute humidity value to be provided by the microcontroller. This value was calculated from the `BME280` sensor readings using the following equation (Equation 1).

$$AH = 216.7 \times \frac{\frac{RH}{100} \times 6.112^{\frac{17.62t}{243.12+t}}}{273.15 + t} \tag{1}$$

Where AH is absolute humidity in g/m^3, RH is relative humidity in percent, and t is temperature in $°C$.

Figure 3.3.2 shows a flowchart for the implemented software on the IoT device. The program starts by enabling the Wi-Fi monitor mode which executes an ISR every time the microcontroller's Wi-Fi module receives a Wi-Fi packet. This packet is processed and the Wi-Fi packet type counter is increased as described in subsubsection 3.2.5. This stores three count for each packet type that has been captured during the period before the payload was composed. A one second delay before composing the payload was used to give enough time to capture the average Wi-Fi activity. During a small experiment, while continuously measuring packets, it was concluded that one second is sufficient to

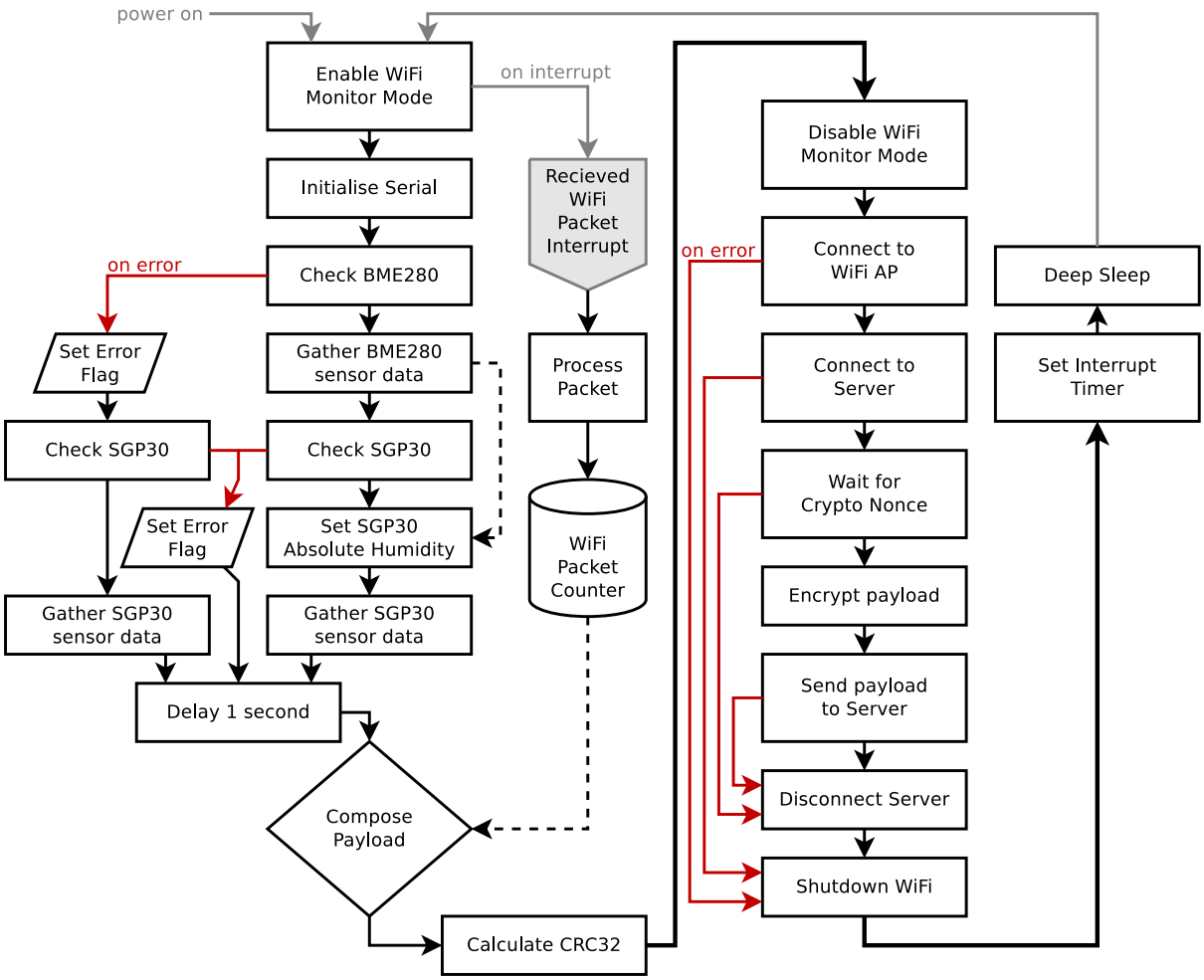


Figure 3.3.2: *Functional diagram of IoT device program. Dashed line indicates data taken by logic block.*

represent average Wi-FI activity.

After the main program enables the Wi-Fi monitoring routine, the serial and the sensors are initialised. Serial block represents Universal Asynchronous Receiver-Transmitter (UART) connection setup which is used for debugging purposes. Sensor initialisation include connecting them via I^2C, retrieving their parameter correcting if they are not right. If either of the sensors do not respond or indicate a problem, an error flag is set high for that sensor.

The program then goes onto gathering sensor readings and composes a payload message with CRC32 checksum. If `BME280` sensor is faulty, setting absolute humidity for `SGP30` is skipped. Then the Wi-Fi mode is changed to connect to an access point which allowed the IoT device to be connected to the server. Upon a new TCP connection, server sends a cryptographic nonce which microcontroller uses to encrypt payload and send it to server, this logic is explained in subsubsection 3.2.6. Afterwards the server and Wi-Fi connections are gradually disconnected, meaning it sends end of connection packet to TCP and de-authentication instruction to Wi-Fi. Then a timer and deep sleep mode is set. When timer time expires, program is reset.

3.3.2 Layer 2

	Content Name	Size and type
	Device ID	8bits
	Payload length*	8bit unsigned integer
Encrypted Payload **24-bytes**	Temperature Reading	32bit float
	Humidity Reading	32bit float
	Pressure Reading	32bit float
	Wi-Fi Managed packets	16bit unsigned integer
	Wi-Fi Control packets	16bit unsigned integer
	Wi-Fi Data packets	16bit unsigned integer
	CO_2 Reading	16bit unsigned integer
	H_2 Reading	16bit unsigned integer
	TVOC Reading	16bit unsigned integer
	Ethanol Reading	16bit unsigned integer
	Status Flags	8bits
	reserved.	8bits
	CRC32 Checksum	32bits

Table 3.3.1: *List of message content sent from IoT device to server via TCP socket.* ***** *Payload length is represented in multiples of 12bits, value of 2 in this case.*

The breakdown of the message that will be sent to the server is shown in Table 3.3.1.

Sending data in raw binary was chosen as suppose to encoded with data interchange format such as JSON or XML due to bandwidth efficiency, computation time and simplicity of implementation with C language. ChaCha20 cipher was used for encryption with 12 byte nonce. This nonce size requires payload to be in multiples of 12 bytes blocks which is the reason for having ”reserved” byte in payload.

Each IoT device is designed to have a unique built-in symmetric key. Device ID is not encrypted and used by server to lookup key dictionary and decrypt the payload. Checksum is also crucial in this case to ensure that used key is correct, because when using an incorrect key resulting message would be ”garbage” meaning that the message and CRC would not match.

Since this IoT device will not be sending any sensitive information and encryption may seem unnecessary. However, this solution ensures that received information was authenticated. This also prevents malicious parties from sending false readings to show that certain cafe is more or less busy, which could have financial advantage.

3.3.3 Layer 3

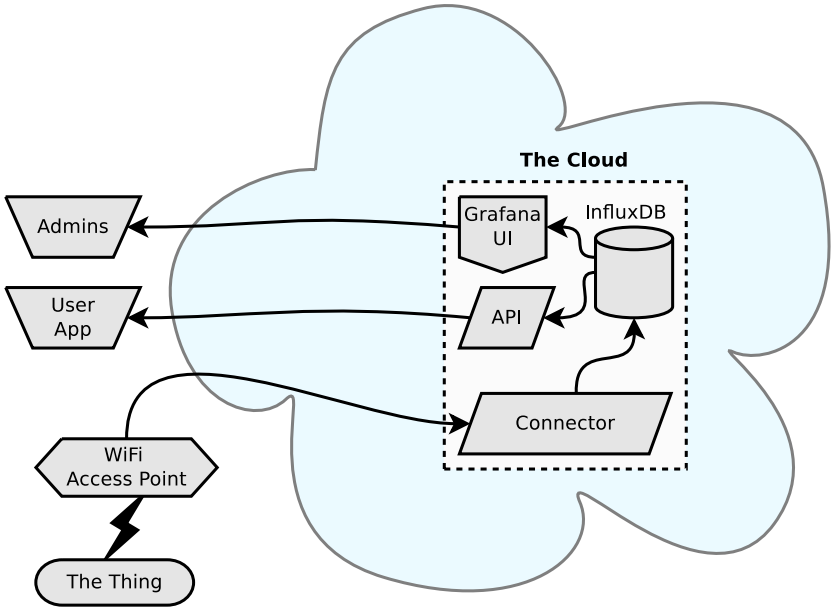


Figure 3.3.3: *General flow diagram of whole system connectivity.*

The cloud contains four main parts: connector, database, API and Grafana User Interface, general block diagram is shown in Figure 3.3.3. Connector is responsible for receiving raw TCP packets, decrypting payload and sending formatted content to Influx Database. This database is chosen because of its time series nature, real-time monitoring, rich documentation and advanced capabilities to analyse data. Data from database is then used by Grafana UI for this platform administrators to perform analysis and monitoring. API block is a simple HTTP server that formats and serves all necessary data from database to user application, such as day predictions and current readings. These responses are formatted with JSON. This block also acts as a security layer so database could not be accessed directly.

3.4 Metrics for evaluating the performance

As ambiance includes multiple metrics as shown in Figure 3.1.2, different evaluation methods are used. Air Quality, humidity and temperature metrics are measured directly by sensors, therefore their performance can be evaluated with calibrated sensor. Number of people cannot be measured directly by any sensors. To estimate this number prediction is made from trained machine learning model that include all metrics, including WiFi activity. In order to train and test performance of this model, a number of methods of gathering actual number of people are considered.

1. Requesting purchase log from café. This log would only need to have list of timestamps that indicate when a purchase was made. Even if number of purchases might not be directly show number of people in café, it would be a very good approximation.

2. Changing IoT device program to continuously stay in monitor mode, including channel hopping, unlike with currently designed program where monitor mode only

active for one second on a single channel. Further analysis on WiFi packets could be done to count unique MAC addresses being broadcast from clients. This would provide a number of unique devices that are in café and have WiFi enabled (not necessary connected). Assuming a majority of people have WiFi enabled, it would a reasonably good approximation of number of people present in café.

3. Use a commercial solution to count people. Such solution may include device placed at every entrance that counts entering and leaving people with a laser interrupt; or a device with camera and computer vision algorithm. This would give most accurate results, however it includes high additional expenses.

4 Experimental Results

5 Engineering Trade-offs

5.1 Sensor selection

This section describes different sensor options and reasons for specific choices.

5.1.1 Temperature and Humidity

A simple thermistor was not used as humidity sensor is also required and most of the module solutions has integrated both sensors. Three I^2C modules were considered: `HTU21D`, `Si7021` and `HDC2080` as these were the only modules that meet requirements and were available from the supplier. All of modules has similar accuracy of $\pm 2 - 3\%$ for relative humidity and $\pm 0.2 - 0.4\ ^\circ C$. All sensors had relatively similar sleep current however, different measuring current and time as shown in Table 5.1.1. `HDC2080` was chosen due to the lowest measuring time which means it's the most power efficient and does not delay microprocessor as long further saving energy.

Sensor	Sleep Current		Measuring Current		Measuring Time
	Typ.	Max	Typ.	Max	Typ.
HDC2080	50 nA	100 nA	$550\mu A$	$730\mu A$	1.27ms
Si7021	60 nA	62 nA	$150\mu A$	$180\mu A$	28ms
HTU21D	20 nA	140 nA	$450\mu A$	$500\mu A$	58ms
BME280	100 nA	300 nA	$350\mu A$	$370\mu A$	8ms

Table 5.1.1: *Power consumption of temperature and humidity sensors [6, 7, 8, 9].*

Unfortunately, after system setup `HDC2080` sensor has failed to respond and a new replacement sensor was needed. Due to availability from alternative supplier and relatively good specification, `BME280` was chosen, which in addition include atmospheric pressure readings which were recorded but not used for analysis.

5.1.2 Air Quality

For air quality, three sensors where considered, `MQ135`, `SGP30` and `CCS811`. `MQ135` lacked I^2C interface, has poor efficiency and requires additional circuit to control heater [10]. `CCS811` has 1.2mW average power draw measuring once every 60 seconds, whereas `SGP30`

takes 6.5mW at minimum which is significantly more. `SGP30` has higher maximum measurement range of up to 60000ppm CO_2 and 60000ppb TVOC comparing to 8192ppm and 1187ppb that `CCS811` can provide [11, 12]. `SGP30` was selected due supplier unavailability to provide `CCS811` sensor.

5.2 Cryptography Choice

A number of different cryptography algorithms have been considered. Arduino library repository provides a library with most of chryptographic algorithm implemented including their benchmarks. Results are shown in Table 5.2.1. Used ESP8266 does not have most commonly used AES hardware acceleration therefore this table should be relevant even when it was produced using much less capable AVR processor. Software implemented AES is outperformed by ChaCha and Speck, the best result showing Speck with ECB mode. However, as ECB mode lacks of diffusion ChaCha20 was chosen instead. 20 over 12 round ChaCha was selected due to relatively low time penalty over much higher system lifetime.

Algorithm	Encryption time per byte	Key Setup
AES192 ECB	$39.94\mu s$	$165.34\mu s$
AES256 ECB	$46.61\mu s$	$217.79\mu s$
AES192 GCM	$116.38\mu s$	$1485.56\mu s$
AES256 GCM	$123.04\mu s$	$1760.28\mu s$
ChaCha12	$10.38\mu s$	$43.74\mu s$
ChaCha20	$14.87\mu s$	$43.74\mu s$
ChaChaPoly	$41.20\mu s$	$902.36\mu s$
Speck 192 ECB	$10.03\mu s$	$264.63\mu s$
Speck 256 ECB	$10.31\mu s$	$275.26\mu s$
Speck 256 GCM	$86.74\mu s$	$646.88\mu s$

Table 5.2.1: *Benchark result for various encryption algorithms perfromed on Atmel ATmega328P microprocessor at 16MHz [13].*

5.3 Cloud computing

In this section a local versus cloud computing is discussed. The only processing that is done locally on microcontroller is absolute humidity calculation as it not a complicated function. Data analysis could possibly be done directly on microcontroller, however that would require being implemented on low level code which is expensive. In addition, as gather data is quite small in size, it has little benefit on processing data locally. However, if computer vision would be used with a more powerful microprocessor, such as ARM, a local processing would much more suitable solution due to high bandwidth requirement for video streaming. It would be a trade-off of accessible bandwidth and local IoT device price. Another trade-off would need to be considered between price of microprocessor power consumption as newer processors consume less power but are more expensive; and battery size.

5.4 Wireless Power consumption

This section will go over trade-offs of wireless power consumption versus range and throughput.
As data sent to server requires very low bandwidth, two other options were considered to be used for data transmission: Bluetooth LE and LoRaWAN. Both methods would consume significantly less power as both transmission requires less energy; and it takes less time to establish connection and transmit data. Bluetooth has a negative side of requiring nearby local gateway that would connect IoT device to the cloud. This defeats the purpose of IoT device and one of such gateways would need to be placed in every café eliminating the need of IoT device itself. Same argument applies to ZigBee or other low range wireless communication. LoRaWAN was strongly considered as a candidate as it would eliminate the need of existing local WiFi infrastructure as there might be difficulties getting WiFi access for an IoT device. In addition a single LoRa gateway could serve numerous cafés in 2-3km radius, assuming urban environment. The choice of not using LoRa was made due to WiFi capabilities required to be present to measure WiFi activity, therefore another communication method is redundant and just increase expenses.

References

[1] *Nest Learning Thermostat - Programs Itself, Helps Save Energy*. 2020. URL: https://store.google.com/product/nest_learning_thermostat_3rd_gen (visited on 04/07/2020).

[2] Foobot. *Foobot - Because Indoor Air Quality Matter*. 2020. URL: https://foobot.io (visited on 04/07/2020).

[3] SensMax LTD. *Real-time people counting system SensMax WebKit TS*.

[4] *Series A, B, C Funding: How It Works*. 2020. URL: https://www.investopedia.com/articles/personal-finance/102015/series-b-c-funding-what-it-all-means-and-how-it-works.asp (visited on 05/01/2020).

[5] R. Kobayashi and P. Kawamoto. "On the Development of a Customizable Crowd Sensing System for Public Spaces Using IoT Cloud Services". In: *2018 IEEE International Congress on Internet of Things (ICIOT)*. 2018, pp. 176–179. DOI: 10.1109/ICIOT.2018.00033.

[6] Texas Instruments. *HDC2080 Sensor Datasheet*. 2019. URL: https://www.ti.com/lit/ds/symlink/hdc2080.pdf (visited on 04/03/2020).

[7] Silicon Laboratories Inc. *Si7021-A20 Sensor Datasheet Rev. 1.2*. 2016. URL: https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf (visited on 04/03/2020).

[8] Inc Measurement Specialties. *HPC199-3 HTU21D(F) Sensor Datasheet*. 2013. URL: https://cdn-shop.adafruit.com/datasheets/1899_HTU21D.pdf (visited on 04/03/2020).

[9] Bosch Sensortec. *BME280 Sensor Datasheet*. 2018. URL: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme280-ds002.pdf (visited on 04/03/2020).

[10] *MQ-135 GAS SENSOR Technical Data*. URL: https://www.olimex.com/Products/Components/Sensors/Gas/SNS-MQ135/resources/SNS-MQ135.pdf (visited on 04/03/2020).

[11] Sensirion AG. *SGP30 Datasheet*. 2019. URL: https://www.sensirion.com/fileadmin/user_upload/customers/sensirion/Dokumente/9_Gas_Sensors/Datasheets/Sensirion_Gas_Sensors_SGP30_Datasheet.pdf (visited on 04/03/2020).

[12] ams AG. *CCS811 Datasheet*. 2016. URL: https://cdn-shop.adafruit.com/product-files/3566/3566_datasheet.pdf (visited on 04/03/2020).

[13] *Arduino Cryptography Library*. 2018. URL: https://rweather.github.io/arduinolibs/crypto.html (visited on 04/03/2020).

Appendices