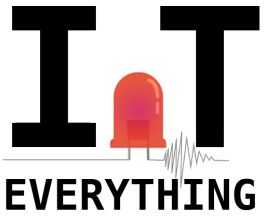


Final Report

IoT to Everything

Gabriel Ferraté Cuartero, Sebastien Kanj Bongard, Xavier Ribas Forcano, Andreu Salcedo Bosch, Oriol Sala Sucarrats, Guillem Lladós Gómez, David Vila Bagaria, Marc Gerritsen, Victor Sabartes Corral, Eloi Moliner Juanpere, Jaume Alexandre Solé Gómez, Judith Mateu Romero, Natalia Jakubiak, Victor Gomar Viana, Katarzyna Karabowitcz, Sergi Ventura i Roig

January 19, 2018

REVISION HISTORY AND APPROVAL RECORD

Revision	Date	Purpose
0	22/11/2017	Document creation
1	22/11/2017	Document revision 1
2	12/12/2017	Document revision 2
3	19/12/2017	Document revision 3
4	10/01/2018	Document revision 4
5	16/01/2018	Document revision 5
6	17/01/2018	Final Document Revision.

DOCUMENT DISTRIBUTION LIST

Name	E-mail
Developer Team	
Sebastien Kanj Bongard	sebastienkanj@gmail.com
Andreu Salcedo Bosch	andreusalbos@gmail.com
Xavier Ribas Forcano	xavierribas18@gmail.com
Oriol Sala Sucarrats	oriol0sala0sucarrats@gmail.com
Guillem Lladós Gómez	guillem.llados.gomez@gmail.com
David Vila Bagaria	david.vila.bagaria@gmail.com
Marc Gerritsen	marc@mgerritsen-it.nl
Victor Sabartes Corral	victor.sabates@alu-etsetb.upc.edu
Eloi Moliner Juanpere	e.moliner.em@gmail.com
Jaume Alexandre Solé Gómez	alexsoleg@gmail.com
Judith Mateu Romero	judith.mateu@alu-etsetb.upc.edu
Natalia Jakubiak	nataliajakubiak9@gmail.com
Victor Gomar Viana	victor.gomar@alu-etsetb.upc.edu
Gabriel Ferraté Cuartero	gabrielferrate1996@gmail.com
Katarzyna Karabowitcz	katarzyna.karabowicz@gmail.com
Sergi Ventura i Roig	serge.ventura.i@alu-etsetb.upc.edu
Professors	
Anna Calveras Auge	anna.calveras@entel.upc.edu
Jose Paradells Aspas	teljpa@entel.upc.edu

Contents

1 Document Scope

The goal of this document is to provide information about our project: IoT to Everything. The objective of our project is the creation of a device composed of a micro-controller and a series of sensors that interact with a smartphone connected to the Internet.

There will be information about time plan, general cost of the electronic device and system design, implementation and characterization. Our team have been divided into subgroups responsible for each part of the project: camera, sound, platform, interface, protocol and electronic. More details will be described by each subgroup in every chapter of the document.

2 Project Summary

The main project's goal was to design user-friendly Android application, small and cheap electronic device and to provide communication between them. The device should cost less than 2 euro and control bigger systems, like a water irrigation system.

The aim of application is to allow user to communicate to the device, giving him different instructions, like a initial configuration or a alarm configuration, and receive data from the device, like sensors or status information. There is opportunity to choose device by its localization. From each of them, user can get information about temperature and humidity and also choose date and time to set the alarm.

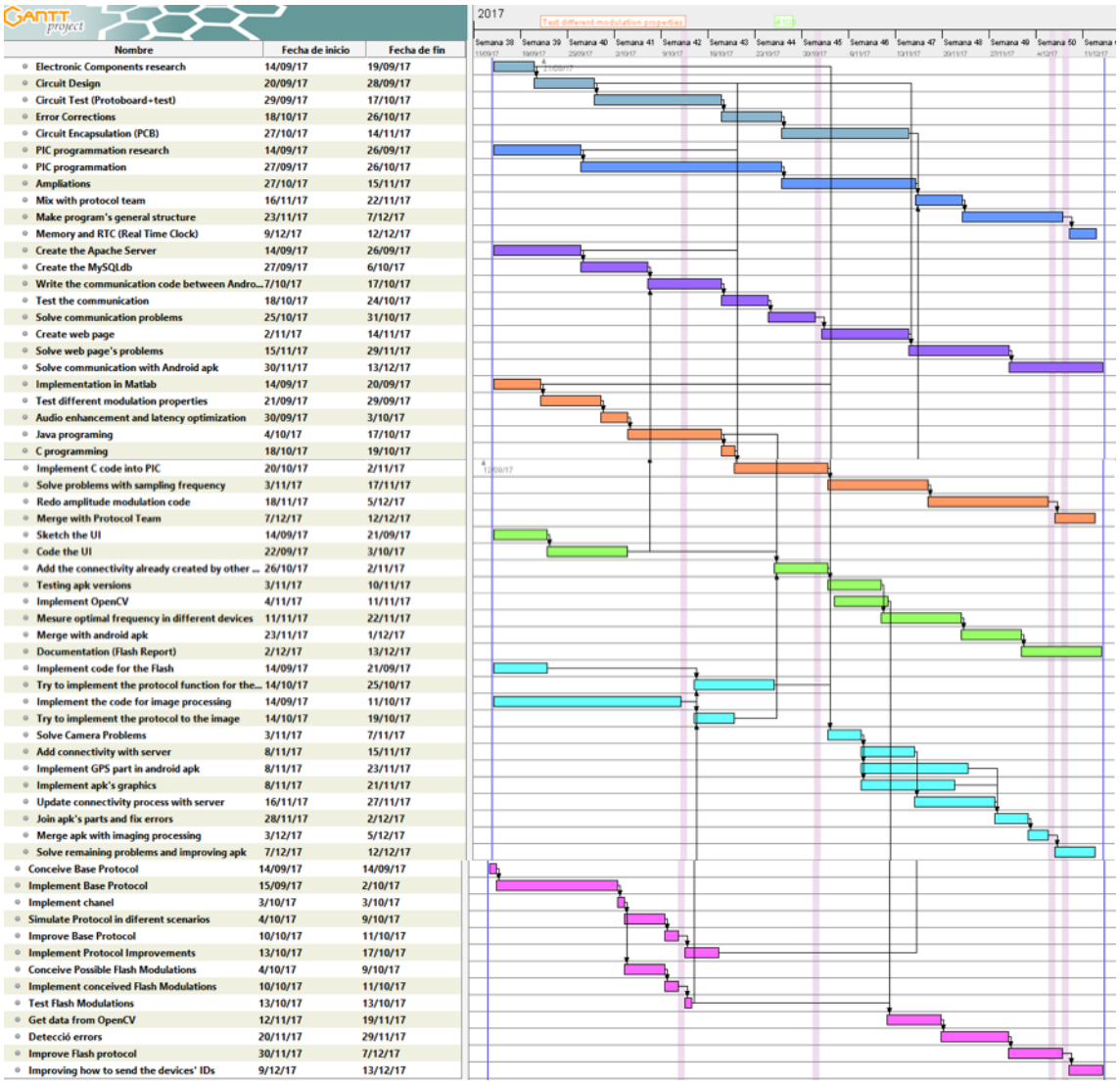
The device is composed of a micro-controller and a series of sensors that interact with a smartphone connected to the Internet. The objective of this product is to connect an offline system to the cloud, where the data will be stored. The communications between the device and the smartphone will be made by image using a device's LED and smartphone's flash and camera, and sound using device's microphone and the smartphone's speakers respectively.

3 Time Plan Updated

Firstly, we separated into groups so each one could start developing a different part of the project. We thought about how many people should in each team and how much time it would be needed to implement each part assigned to have an estimated time plan.

However, as the semester went by, we couldn't exactly follow the time plan estimated. Some people had to switch from their group to another one, where they were needed to help in some parts of the project. Moreover, this semester there were a few strikes, which the majority of them took place the days we had class, so the project was slowed down a little bit.

At the end, we could successfully merge all the parts together and we were able to finish our project in the right time.



Nombre	Fecha de inicio	Fecha de fin
Electronic Components research	14/09/17	19/09/17
Circuit Design	20/09/17	28/09/17
Circuit Test (Protoboard+test)	29/09/17	17/10/17
Error Corrections	18/10/17	26/10/17
Circuit Encapsulation (PCB)	27/10/17	14/11/17
PIC programmation research	14/09/17	26/09/17
PIC programmation	27/09/17	26/10/17
Ampliations	27/10/17	15/11/17
Mix with protocol team	16/11/17	22/11/17
Make program's general structure	23/11/17	7/12/17
Memory and RTC (Real Time Clock)	9/12/17	12/12/17
Create the Apache Server	14/09/17	26/09/17
Create the MySQLdb	27/09/17	6/10/17
Write the communication code between Andro...	7/10/17	17/10/17
Test the communication	18/10/17	24/10/17
Solve communication problems	25/10/17	31/10/17
Create web page	2/11/17	14/11/17
Solve web page's problems	15/11/17	29/11/17
Solve communication with Android apk	30/11/17	13/12/17
Implementation in Matlab	14/09/17	20/09/17
Test different modulation properties	21/09/17	29/09/17
Audio enhancement and latency optimization	30/09/17	3/10/17
Java programing	4/10/17	17/10/17
C programming	18/10/17	19/10/17
Implement C code into PIC	20/10/17	2/11/17
Solve problems with sampling frequency	3/11/17	17/11/17
Redo amplitude modulation code	18/11/17	5/12/17
Merge with Protocol Team	7/12/17	12/12/17
Sketch the UI	14/09/17	21/09/17
Code the UI	22/09/17	3/10/17
Add the connectivity already created by other ...	26/10/17	2/11/17
Testing apk versions	3/11/17	10/11/17
Implement OpenCV	4/11/17	11/11/17
Mesure optimal frequency in different devices	11/11/17	22/11/17
Merge with android apk	23/11/17	1/12/17
Documentation (Flash Report)	2/12/17	13/12/17
Implement code for the Flash	14/09/17	21/09/17
Try to implement the protocol function for the...	14/10/17	25/10/17
Implement the code for image processing	14/09/17	11/10/17
Try to implement the protocol to the image	14/10/17	19/10/17
Solve Camera Problems	3/11/17	7/11/17
Add connectivity with server	8/11/17	15/11/17
Implement GPS part in android apk	8/11/17	23/11/17
Implement apk's graphics	8/11/17	21/11/17
Update connectivity process with server	16/11/17	27/11/17
Join apk's parts and fix errors	28/11/17	2/12/17
Merge apk with imaging processing	3/12/17	5/12/17
Solve remaining problems and improving apk	7/12/17	12/12/17
Conceive Base Protocol	14/09/17	14/09/17
Implement Base Protocol	15/09/17	2/10/17
Implement chanel	3/10/17	3/10/17
Simulate Protocol in diferent scenarios	4/10/17	9/10/17
Improve Base Protocol	10/10/17	11/10/17
Implement Protocol Improvements	13/10/17	17/10/17
Conceive Possible Flash Modulations	4/10/17	9/10/17
Implement conceived Flash Modulations	10/10/17	11/10/17
Test Flash Modulations	13/10/17	13/10/17
Get data from OpenCV	12/11/17	19/11/17
Detecció errors	20/11/17	29/11/17
Improve Flash protocol	30/11/17	7/12/17
Improving how to send the devices' IDs	9/12/17	13/12/17

4 System Design Documentation

4.1 Protocol Team

The aim is to develop a robust and user-friendly protocol to communicate in a half-duplex way.

For the Light Communication (Device to Smartphone) we developed a custom protocol light enough for a PIC device and error-free for the user. We merged the idea of a Rolling Shutter effect in a CMOS camera and the modulation of a Smartphone camera flash to implement our final idea.

For the Sound Communication (Smartphone to Device) we have designed a light computational way to communicate to the PIC micro-controller using Amplitude Modulation and also a Hamming Code algorithm to detect and correct errors. De idea was to design this protocol to wirelessly control the micro-controller software from a Smartphone.

4.2 Platform Team

The platform team was divided in two teams: The Android Team and the DataBase Team. The DataBase team has worked in the creation of an Apache Server and a MySQL DataBase, that will save all the data that will come from the smartphone. The team has developed a webpage that analyze the data and give to the device manager charts and graphic representations. The Android Team has work in the creation of and Android App that will communicate with the device and the database.

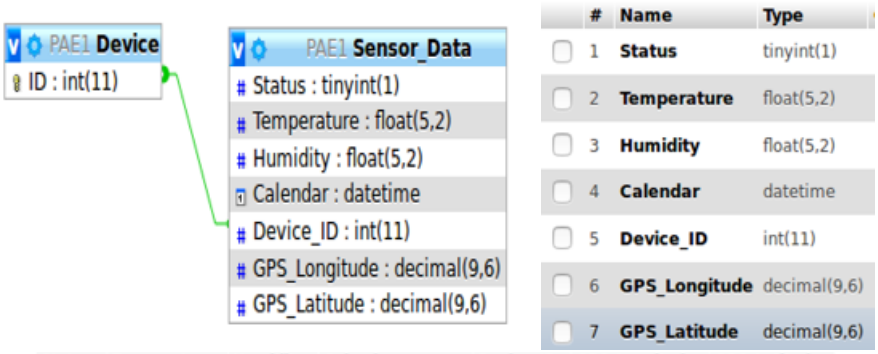
4.2.1 Database Team

The DataBase was designed to save different data from the smartphone:

- 2 float values, in our example temperature and humidity
- 1 tinyint value, in our example status of the device

- 1 datetime, that corresponds to the time that the data was taken
- 1 integer value, that correspond to the ID of the device
- 2 decimal values, for the GPS longitude and latitude values

The following schematics shows the initial database structure, the database table Sensor_Data as above described and some virtual data added to the test database:

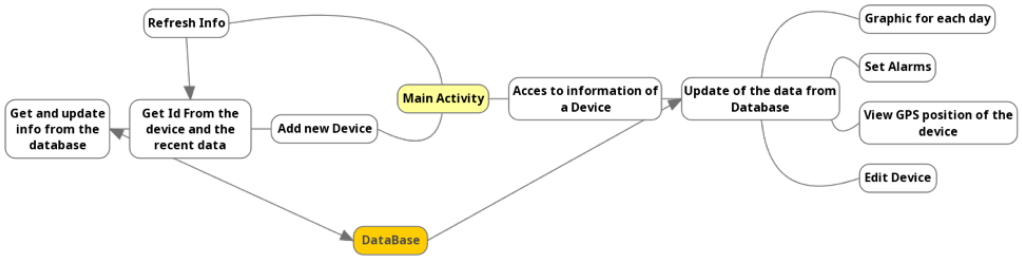


Status	Temperature	Humidity	Calendar	Device_ID	GPS_Longitude	GPS_Latitude
1	13.30	94.00	2017-11-20 12:00:00	1	4.895168	52.370216
1	13.80	93.00	2017-11-21 12:00:00	1	4.895168	52.370216
1	14.10	94.00	2017-11-22 12:00:00	1	4.895168	52.370216
1	13.10	92.00	2017-11-23 12:00:00	1	4.895168	52.370216
1	13.50	90.00	2017-11-24 12:00:00	1	4.895168	52.370216
1	13.20	89.00	2017-11-25 12:00:00	1	4.895168	52.370216
1	12.70	95.00	2017-11-26 12:00:00	1	4.895168	52.370216
1	11.70	96.00	2017-11-20 12:00:00	2	4.477733	51.924420
1	11.50	97.00	2017-11-21 12:00:00	2	4.477733	51.924420
1	11.80	97.30	2017-11-22 12:00:00	2	4.477733	51.924420
1	12.30	97.80	2017-11-23 12:00:00	2	4.477733	51.924420
1	12.10	94.80	2017-11-24 12:00:00	2	4.477733	51.924420
1	11.60	92.70	2017-11-25 12:00:00	2	4.477733	51.924420
1	11.90	93.90	2017-11-26 12:00:00	2	4.477733	51.924420
1	17.90	47.90	2017-11-20 12:00:00	3	2.173403	41.385064
1	17.40	47.30	2017-11-21 12:00:00	3	2.173403	41.385064
1	16.20	49.30	2017-11-22 12:00:00	3	2.173403	41.385064
1	16.80	51.10	2017-11-23 12:00:00	3	2.173403	41.385064
1	17.20	50.20	2017-11-24 12:00:00	3	2.173403	41.385064
1	17.00	49.40	2017-11-25 12:00:00	3	2.173403	41.385064
1	18.70	49.20	2017-11-26 12:00:00	3	2.173403	41.385064
1	13.70	78.50	2017-11-20 12:00:00	4	-0.127758	51.507351
1	13.20	76.50	2017-11-21 12:00:00	4	-0.127758	51.507351
1	13.10	77.90	2017-11-22 12:00:00	4	-0.127758	51.507351
1	13.50	79.20	2017-11-23 12:00:00	4	-0.127758	51.507351

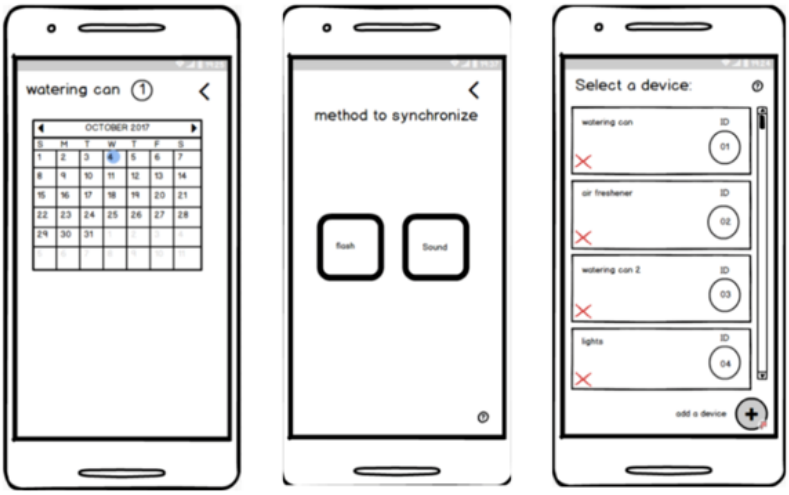
To fill the database the platform team created multiple php files, so the android application can send and get information from or to the Apache server (database). The php files are really small and make use of JSON data format to transfer the data to the database. JSON data format are known of their human readable text.

4.2.2 Android App

First of all, we discussed about which actions the application should do. This diagram shows how did we want to implement the application.



This was the first interface idea that we had. At the system implementation documentation, we will show how this first idea has evolved.



After skeching the diferent parts of our aplication we began to desin the interfaces with android studio, and meanwhile figuering out how we will implement it on the code.

Once we had the diferent parts, the proces of coding started. First without the connection to the database nor the device. With fake data.

4.3 Camera Team

The camera team was divided in two teams: the Flash Team and the Image processing Team.

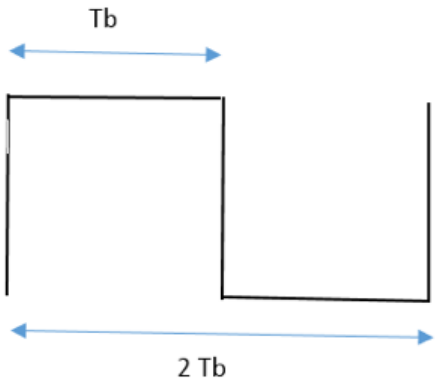
4.3.1 Flash Team

The aim is to send information to the device with light detector using flash. Flash in mobile device is controlled by Android Application. Information is going to be sent as a sequence of true and false bits (where true bit means flash on and false bit means flash off) with particular frequency which has to be understandable for the light detector. In order to receive it we had

to find dependency between theoretical time of one bit assigned in application code and real frequency measured needed to change the light from on to off and conversely. The difference between that times depends on time needed for device to apply process of switching the light on or off.

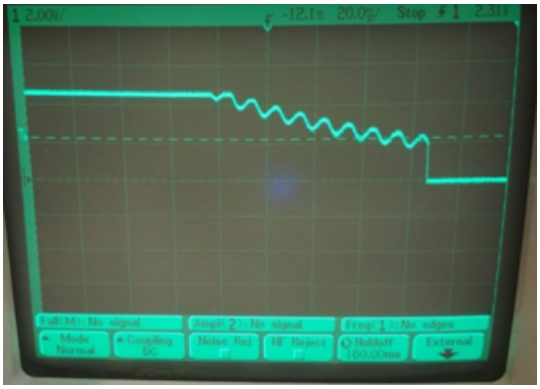
We implemented testing version of application sending one true and one false bit in a row with opportunity to change time assigned in app and we carried out tests observing output waveform and measured frequency on an oscilloscope.

Theoretical time of each bit was assigned in application in range is from 1ms to 200ms. In experiment we used two smartphones: Xiomi Redmi 4x and LG Nexus 5x



T - theoretical time of one period (assigned in the application) theoretical frequency: $f = \frac{1}{T}$
T_b - measured time of one period
Measured frequency: $fb = \frac{1}{T_b}$
Difference = measured time - theoretical time = $T_b - T$

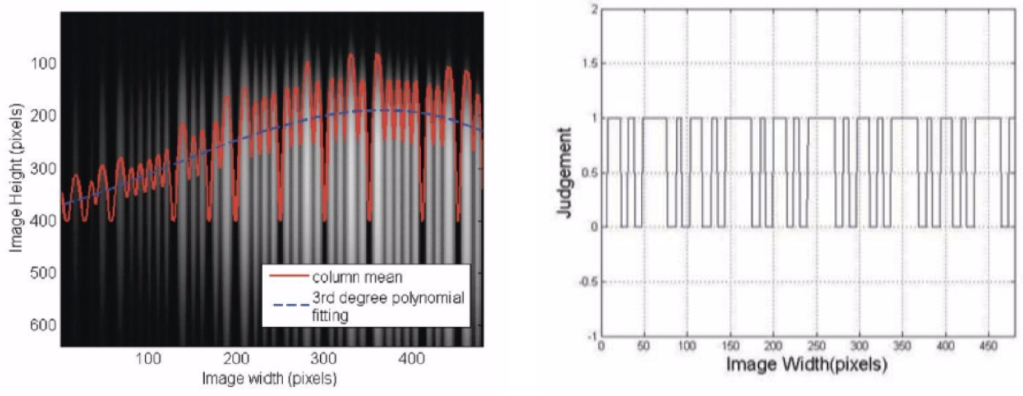
According to observing delay there is no linear correlation between time assigned in application and observing on the oscilloscope screen so to choose the optimal frequency we have to choose the one that there are the smallest difference between theoretical time and measured one and at the same time fluctuations at that frequency are getting smaller.



4.3.2 Image Processing Team

The objective of this part is to receive data from the LED of the PIC. We implemented the image processing of the LED with OpenCV for Android. At first we tested with the java functions of OpenCV but we realised that if we want to improve the code we needed to implement the code in C++.

For the implementation of the code we followed the paper "Using a CMOS Camera Sensor for Visible Light Communication" that use a polynomial regression for the threshold of the means of the rows. For a better results and reduce the variance of the light and the computational cost we only use the bottom half of the image. We made a downsampling of the results for distinguish the ones and the zeros of the result judgement graph.



4.4 Electronic Team

4.4.1 Software design

Firstly, we have done a research on the electronic components required for the project. As the initial idea was to communicate a PIC with the smartphone via light and sound (see figure below) we would need a PIC with multiple analog inputs and outputs an RTC and all the other passive components.

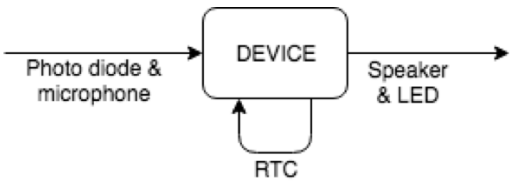


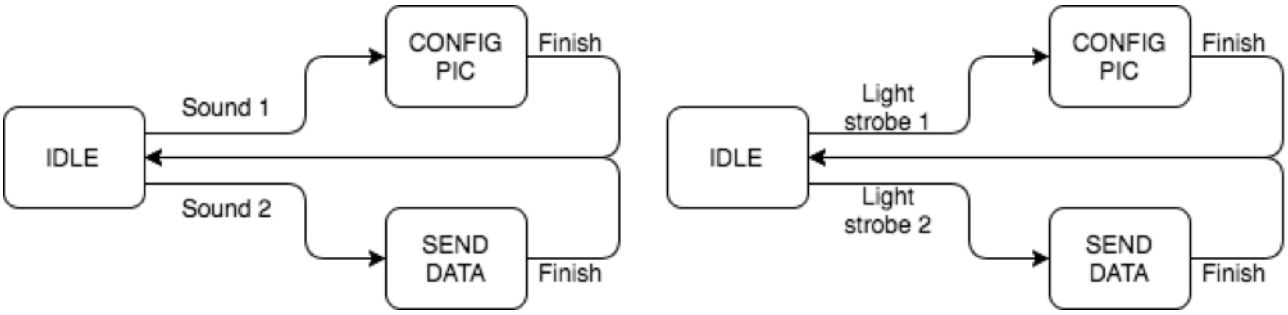
Figure 1: Inputs and outputs of the PIC, first approach.

Having in mind the requirements we chose the PIC16F19156. This PIC has the following

characteristics:

- Interrupt capability
- ADC^2
- Internal RTCC
- 28kB Flash memory
- 256 bytes of EEPROM

We designed a general state machine in order to define the global structure of our code. We came up with the following two prototypes:



Depending on which sound or light strobe we receive each pic would enter in a configuration state or a send data state. The configuration state would change the calendar parameters of the pic and the send data will send the data stored in the EEPROM.

We started developing some simple functions (e.g. a LED blink) to understand how to program the ports and pins of the PIC.

4.4.2 Hardware Design

After checking several design options and components such as LDR's, LED's, photodiodes and phototransistors; we realized that the best option in terms of performance and cost would be using a photodiode as a light receiver, and some other stages for filtering and amplifying made from passive components and a non-inverting AC-Coupled amp structure like shown below.

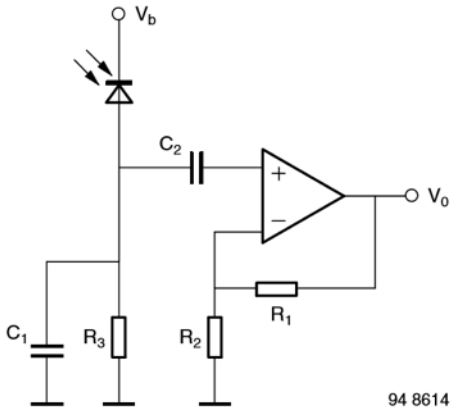


Fig. 24 - AC-Coupled Amplifier Circuit

$$V_O \approx \phi_e \times S(\lambda) \times R_3 \times \left[1 + \frac{R_1}{R_2}\right] \quad (6)$$

The circuit in figure 24 is equivalent to figure 23 with a change to AC coupling. In this case, the influence of background illumination can be separated from a modulated signal. The relation between input signal (irradiation, ϕ_e) and output voltage is given by the equation in figure 24.

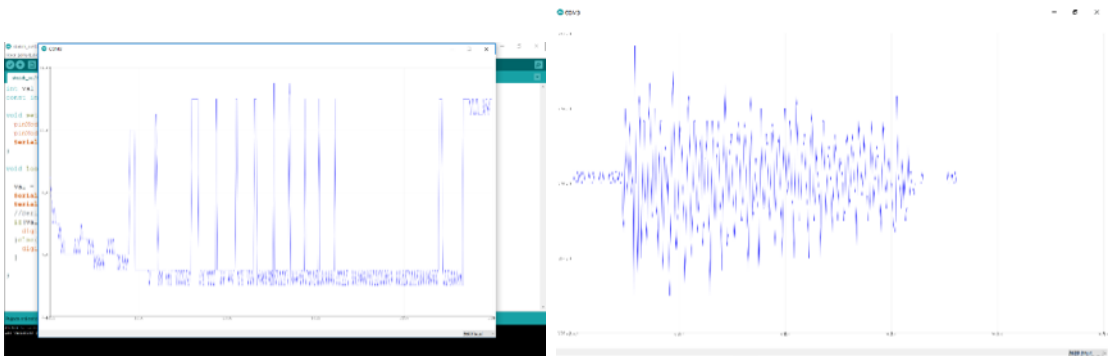


Figure 2: Photodiode response to light pulses and microphone response to voice signal

After some experiments of characterization, we could determine the optimal values of some of the components of the circuit, and therefore some relevant parameters like cutoff frequency of filters, and amplification gain.

To determine both gains, we had to check the photodiode and electret data-sheets and test with them; selecting the dynamic range that the sensors would offer, and consequently the ratio to the values needed at the input of the PIC.

The cutoff frequency of the photodiode filter, was designed in order to remove the 50 and 100 Hz interference from background lights which would get amplified without filtering it. The photodiode acts like a current source, so an RC parallel configuration is totally ideal to

get a low pass filter to remove interferences over 50Hz. The 100n capacitor, removes the DC content of the signal, adding robustness to the circuit for working on different environments, but decreasing signal amplitude to a half. As the input signal is weak, it's important to amplify/reconstruct it with the amplifier to get the desired square pulses.

For the microphone part, the two capacitors act like DC component removers. For one side, the 100n capacitor filters the DC component of the input signal, and the 1 uF one, forces the amp to not amplify at 0 frequency. The gain in DC is 1, the amp acts as a follower.

After the removal of the DC content, it's just necessary to amplify and add again the desired DC level.

These two designs are able to work in various supply voltages as they're designed for it. The output signal will be always the same. In photodiode case, square pulses between 0 and VDD; and for the microphone, audio signal between 0 and VDD centered at VDD/2. That's an important fact in order to implement the design on several projects/devices.

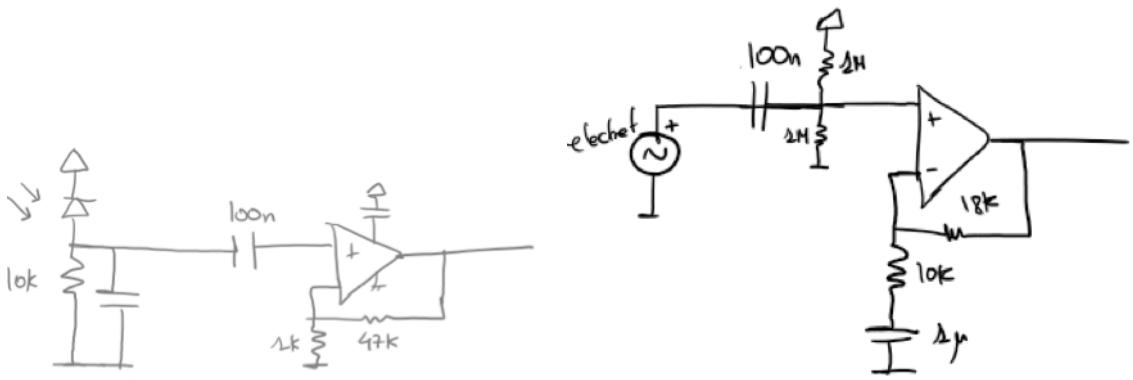


Figure 3: The next step was testing and checking that everything worked as expected.

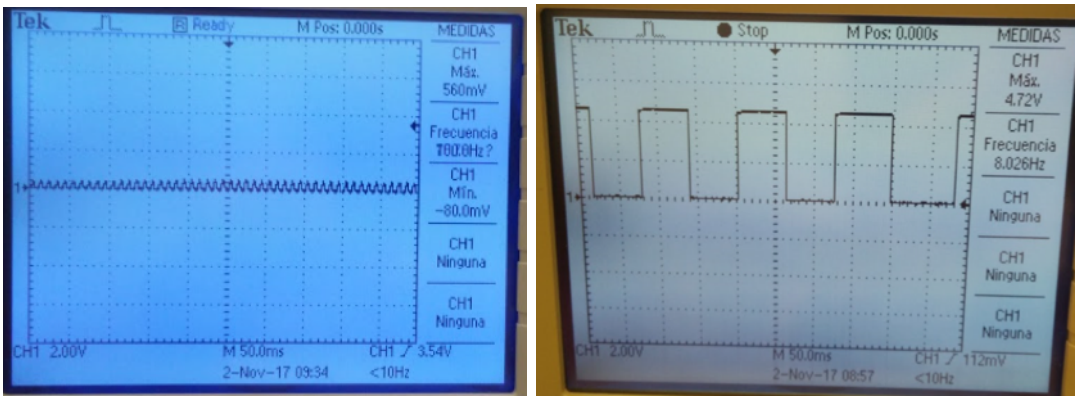


Figure 4: Interferent noise @100Hz from fluorescent light / Correct flash pulses with Vdd=5V

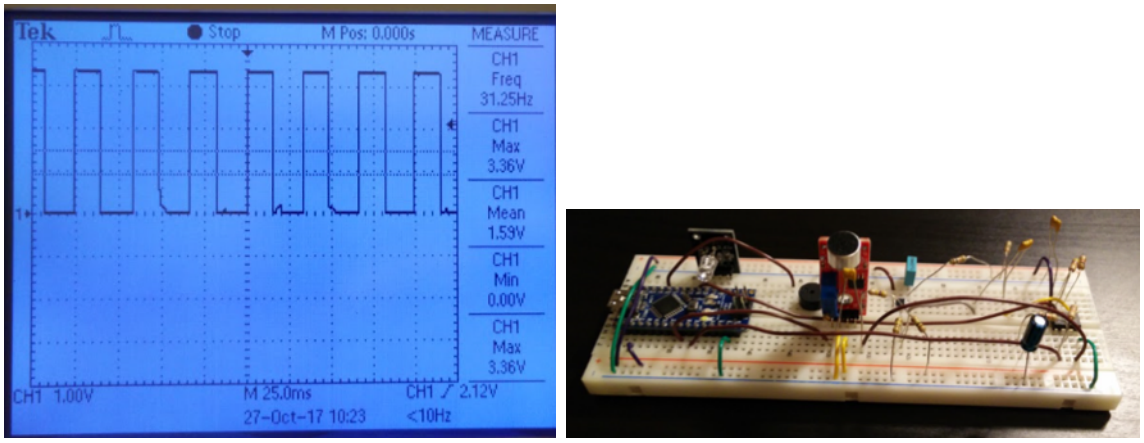


Figure 5: Correct flash pulses with Vdd=3,3V / Breadboard implementation of the circuit

It can be seen that later the VDD value from all the circuit was decreased from 5V to 3,3V in order to match PIC supply voltage and simplify the design. Some changes had to be made on the entire design. For example, changing R values to reduce amplification and changing the schematic and PCB.

The 47 kΩ resistor was finally changed to a 27 kΩ one, which reduced gain, improving robustness against interferences.

Once made that, it was the time to start designing a PCB with the GNU software: kiCAD. It was important to take care on the situation of each component due to the difficulty of routing it and the impact of the components and its free space at the time of soldering. It was also really important the fact that the photodiode had physical space in order to receive the phone signal at its best.

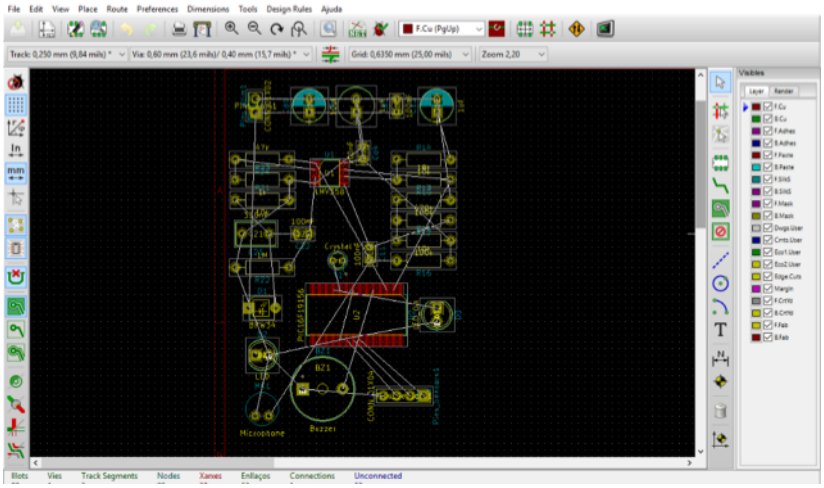


Figure 6: First component disposition on kiCAD

To protect the prototype while doing tests, a 3D printed box was designed and printed.

Figure 7: 3D Design / Final result

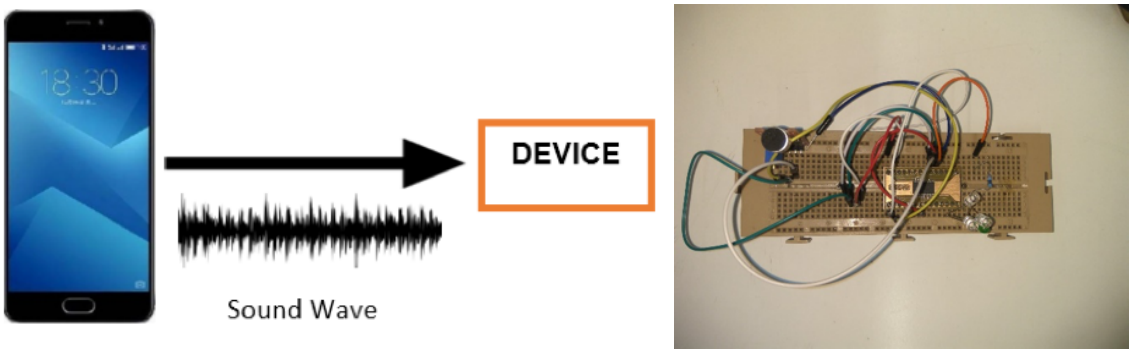
4.5 Sound Team

The aim is to use different technologies to send information using sound and to obtain a reasonable bitrate with a simple but reliable communication.

We want to encode an array of integers as an audio signal, which can be transmitted by any device (Smartphone) with a speaker. Then this signal is received by the device with its microphone if the device is within the hearing range. As the transmission takes place via audio signals, no internet connection is required to interact.

Our first purpose was to implement a quadrature phase shift key modulation (QPSK), with a carrier frequency of 1kHz and the maximum baud rate we could achieve experimenting with the PIC microcontroller. Firstly, we implemented a version in MATLAB. The MATLAB code was tested with several recordings with different properties (baud rate, sampling frequency, carrier, etc.) and it seemed to work correctly. Then, we simplified the code in order to implement it in C language without using floating point and as less operations as possible. It also worked correctly in our tests using NetBeans. After these problems, we forgot about the QPSK and we focused in a simple binary amplitude modulation. (AM). We decided to use white gaussian noise instead of a sinusoid in order to simplify the decoding process as much as possible and obtain higher bitrates.

We will have to apply experimental thresholds, phase correction and synchronization while processing.

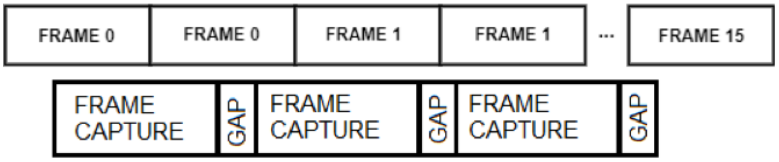

DEVICE
Sound Wave

5 System Implementation Documentation

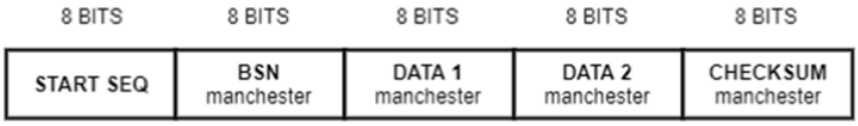
5.1 Protocol Team

5.1.1 Light Communication

The idea is to divide our data into packages with a fixed length able to later implement error detection and still transmit at a high bit rate.

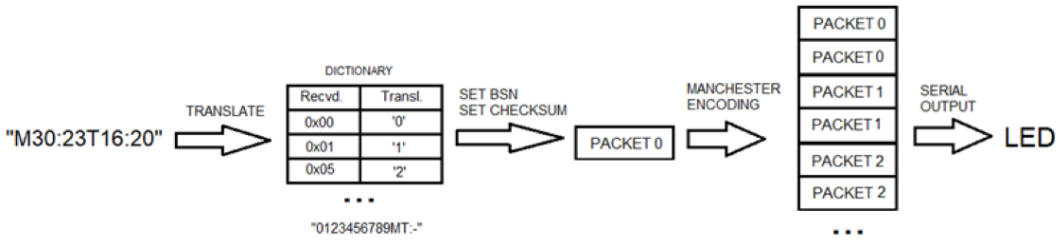


Each frame is repeated two times in order to avoid a 'gap' in the acquisition state of a cmos camera.



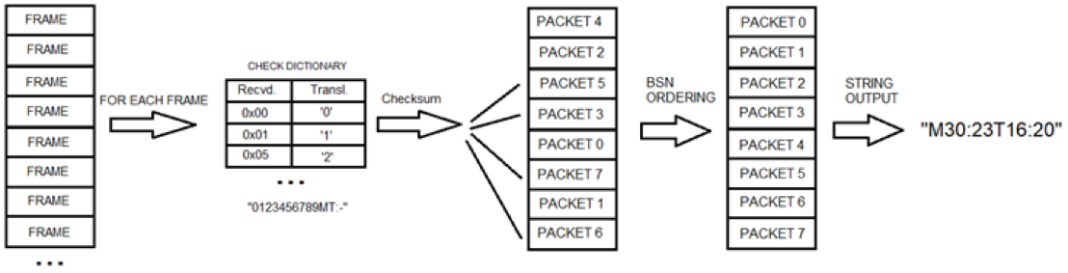
Each frame contains some control bits such as start/end sequence, parity checksum and a block sequence number in order to sort the packages once they are decoded.

Encoder Algorithm



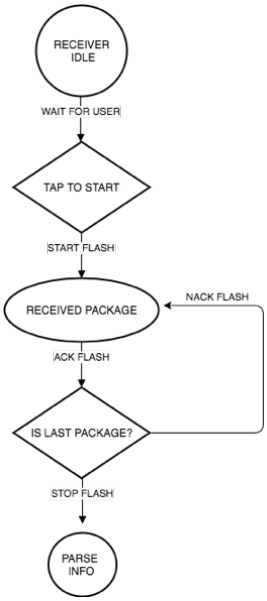
We narrowed the transmitter vocabulary to 10 number characters, 'M', 'T', ':', '-'. Since we will mostly use sensor data. Using this new dictionary we can scramble the transmitted symbols for minimum checksum error.

Decoder Algorithm



After checking the parity error of each frame, we apply the descrambler to translate the received symbols into the transmitted characters.

In order to obtain 100% error free detection, we have implemented a state machine between decoder/encoder trading off bitrate for detection errors.



5.1.2 Sound Communication

We decided to communicate to the device via sound. So we established different sound commands to send different types of data.

Generic Packet implementation

It is an amplitude modulation system communication. And we are also using a Hamming Coding (7/4) algorithm in order to detect and correct errors during the transmission.

- **Install Device Command:** 0001 - Using this command we can wirelessly program our chip to change the ID and actual timestamp.
- **Set Alarm Command:** 1100 - With this command we will change different alarms from the Real Time Clock. There is a future implementation to change GPIO states to wirelessly set inputs / outputs from the devices.

The whole code of the protocol part can be found here:

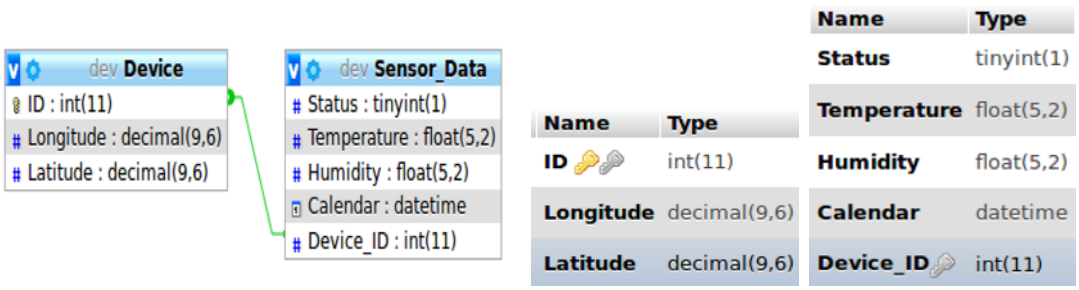
https://github.com/urisala1996/IOTtoEverything/tree/master

5.2 Platform Team

5.2.1 DataBase Team

For the implementation part of the platform team, the first action is to create a Apache server and a MySQL Database. After the creation of the Apache server and the MySQL Database, the platform team created multiple files (SQL, PHP, JSON, CSS and HTML) that directly create the database, the website with all the graphics and the needed files to exchange information between the Android application and the Apache server.

The definitive structure of the MySQL database is a slightly different than the initial one. Because the chip always be on the same place, it is unnecessary to send every time the GPS location. Because of that reason we decided to put the GPS data in the table of the Device.



5.2.2 Android App

The Android app was created as a bridge between the user and the Device, so we had to synchronize with the other teams work to keep up the specifications and the capabilities of the application.

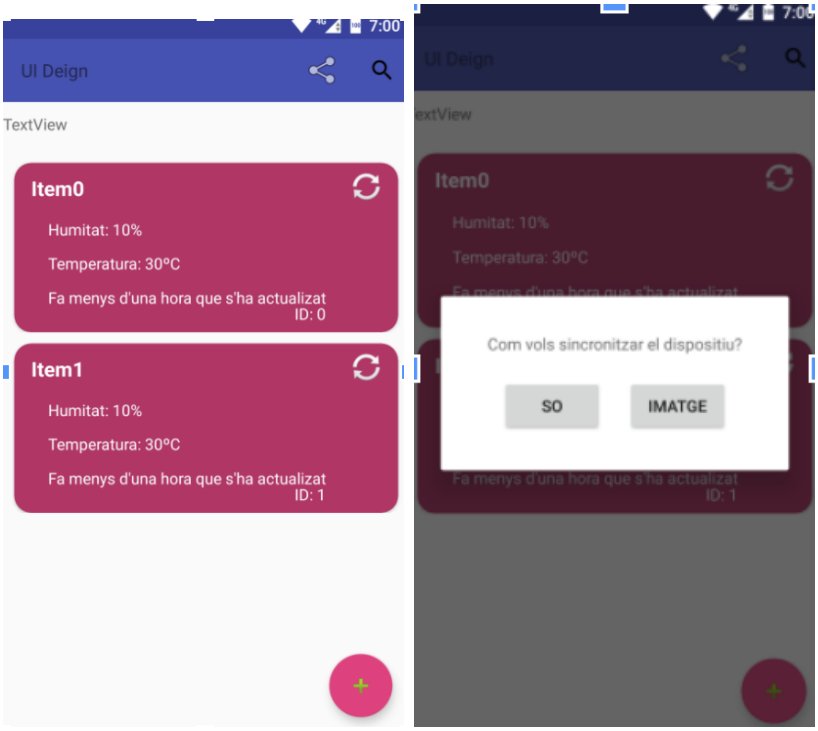
The application is composed of three main parts:

- User Interface
- Server Communications
- Device Communications

We began by the UI because the other groups as it was the beginning had not done anything.

The UI was composed of two main parts:

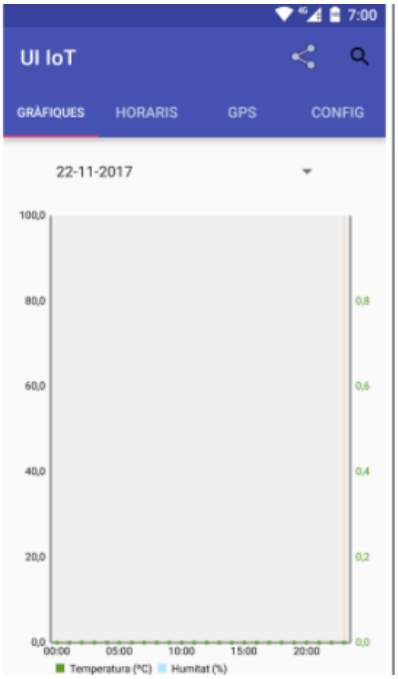
The Main Activity which has all the listed devices with the last obtained metrics and the buttons to send and receive data from the device, these actions are selected from a dialog once the user clicks on the + button.



Also, when the user changes the alarm time, the sync button appears in the device card and he can send the alarm information through sound to the Device.

When the user clicks inside the card of the Device the second part of the UI is revealed. Composed by 4 different frames we have all the controls needed to operate each Device.

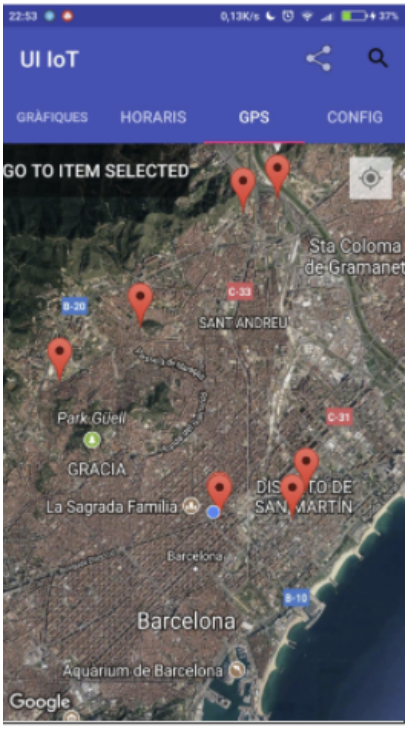
The first one is the graphs section. The user can select a date and the data that the server has of this date is showed in the graph easily.



The next tab is the alarm Tab, with which the user can configure multiple alarms, each one with each different and days of the week are wanted to create the alarm. Each alarm is added with the + button.

The third one is the map tab, where the user can be informed of the location of the Device and see where the other devices are.

The last one is the configure tab where the user can change the location of the Device to his



current one, the name and the image of the device's card.

After all of this was done, we began to collaborate with the platform team to make the connections needed to send and receive data with the server.

To fulfill this purpose the API was created. We figured out which and how we wanted to transmit the data with the server, and at the end we needed:

- Get a new id not used and create a Device
- Get all the data of a day and of a Device particularly
- Get the last data of all the devices
- Get the parameters (GPS and Alarm) of a Device
- Send the data of a Device to the data base

To make this possible code in both ends was made:

- Android

We created different AsyncTask to implement each transmission.

- Server

Several php files were developed to obtain or send data to the data base.

In the third part of the project we had to merge our application with the other teams,

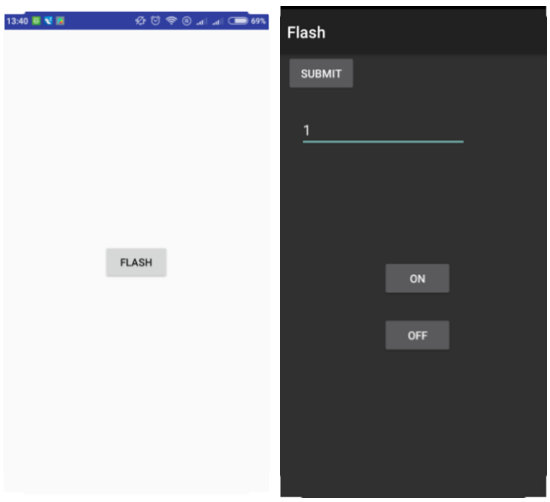
sound and image. And try the information path that we created, solving meanwhile all the compatibility problems that we had.

The main problem that we got was created by the misinformation between teams during the project, which in the end caused a lot of problems.

5.3 Camera Team

5.3.1 Flash Team

For the flash we have designed two versions of application. First and final one is for controlling the flash according to the received packages and has an ability to interpret them and to send each sequence by flash turning on/off. The other one is for testing flash modulations. This one gives possibility for the user to choose the theoretical time to set in the application in order to check the range of the frequency achieved for each the smartphone. For the code implementation, we used Android Studio environment and android.hardware.Camera library



The user needs to allow application to have an access to the device's camera

5.3.2 Image Processing Team

For the implementation of the paper we did a polynomial regression in OpenCV using a Vandermonde matrix with the row means of the image to obtain the beta coefficients in real time and multiply again with the Vandermonde matrix to obtain the polynomial regression. After all this process, we subtract the polynomial regression with the row means and we obtain the regular threshold that we wanted. Finally, with our threshold we binarize to get the value of the bit.

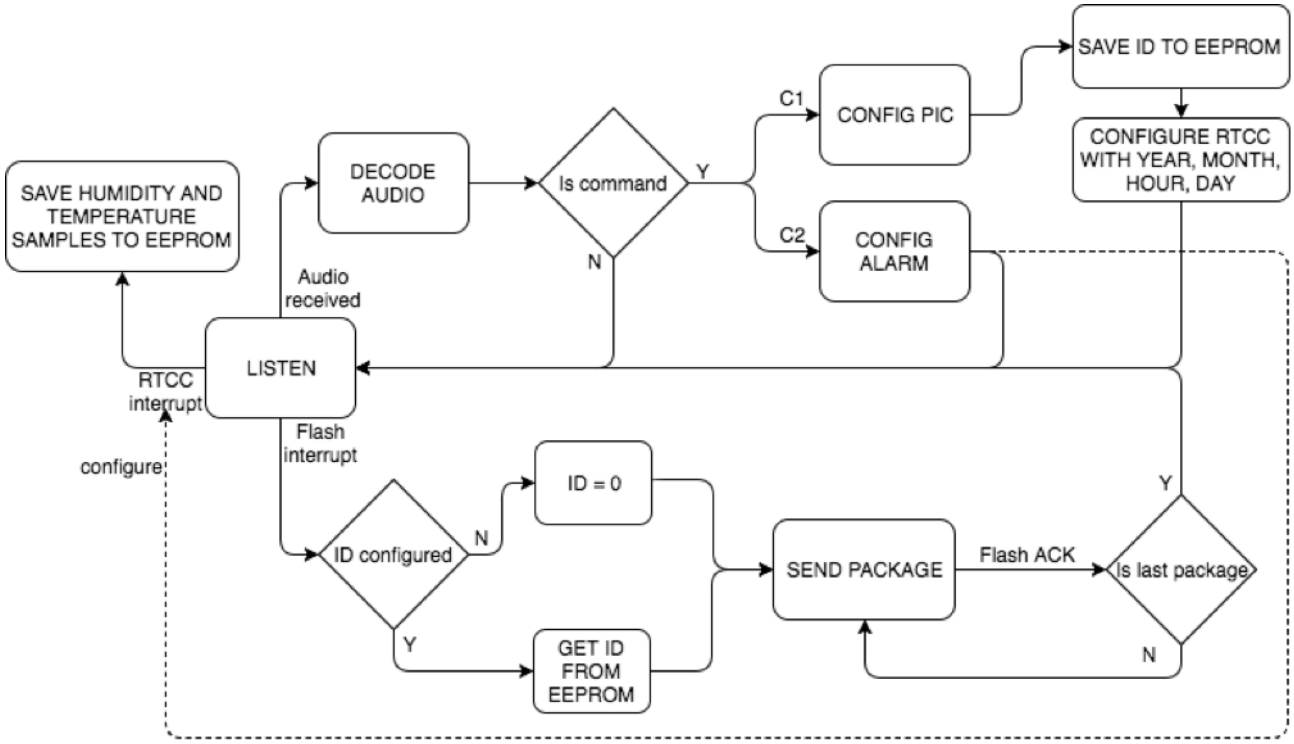
$$\hat{\vec{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\,\mathbf{X}^\mathsf{T}\vec{y}, \qquad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^m \\ 1 & x_2 & x_2^2 & \cdots & x_2^m \\ 1 & x_3 & x_3^2 & \cdots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^m \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix}$$

Figure 8: Vandermonde matrix

5.4 Electronic Team

5.4.1 Software Team

After doing some tests with our first implementation we concluded that we needed a much more complex state machine in order to accomplish our goals. This new structure consists of a main state called LISTEN



As we can see, the idle state of the PIC is the Listen state. At this state, the PIC is waiting for some sound. When the smartphone emits a sound via the speaker, a pice of code will decode the audio and check if the array received is a command or not.

The commands programmed are Config PIC, which sets a new ID for the PIC and Config Alarm, which sets de RTCC interrupt times.

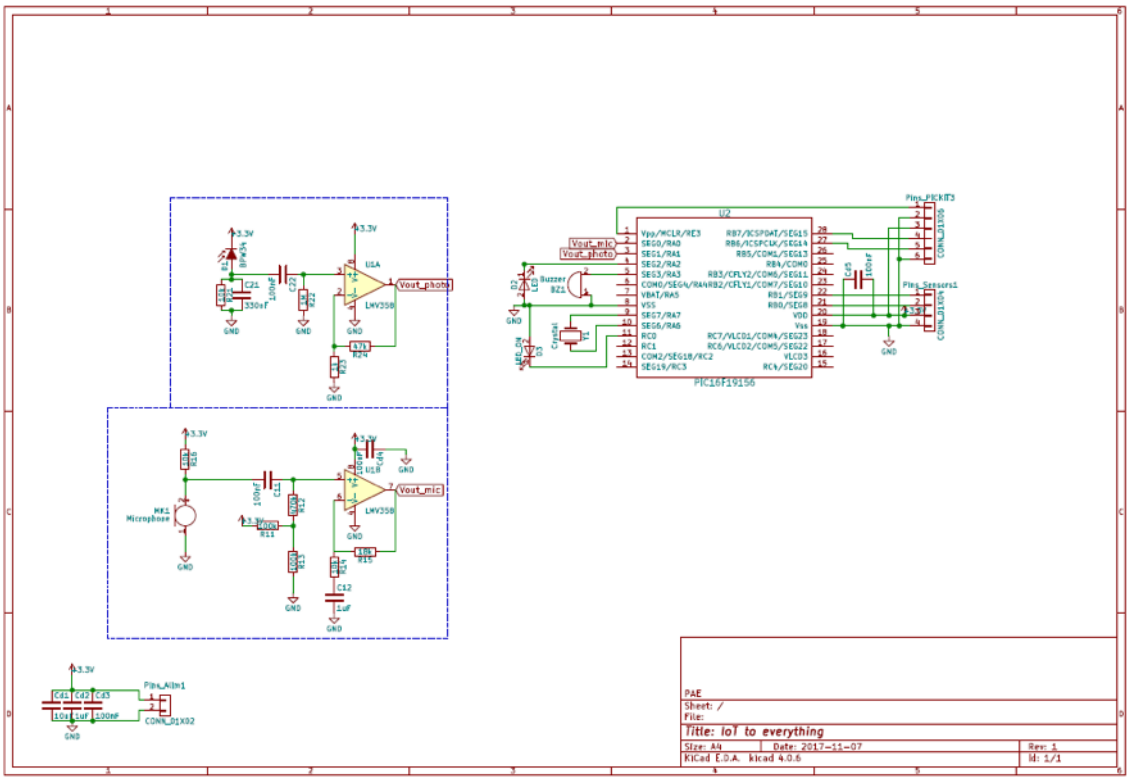
On the other side, if the photodiode detects a flash from the smartphone, the PIC will start sending all the data captured by the sensors (in our case Temperature and Humidity sensors).

If no ID is set to the PIC it will send the data with the ID '0' and then the smartphone app will tell the user to configure the pic as new: Config PIC command.

5.4.2 Hardware Team

1st Design

Final schematics / software blocks



Circuit Layout

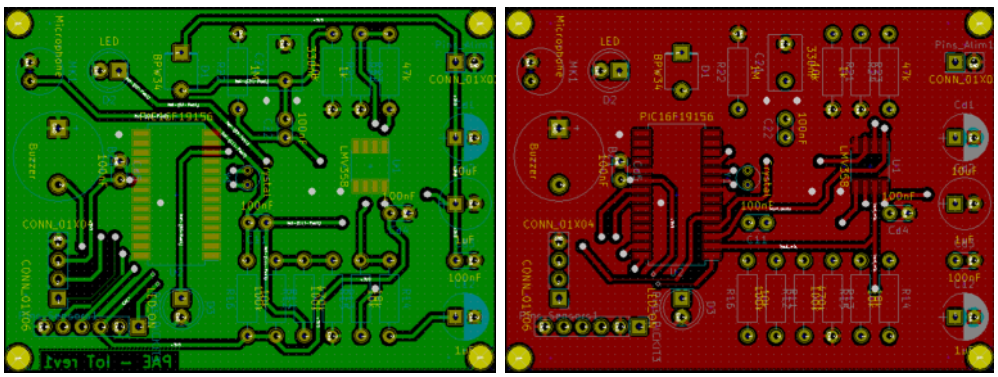


Figure 9: Bottom and Top Copper Layers of the PCB

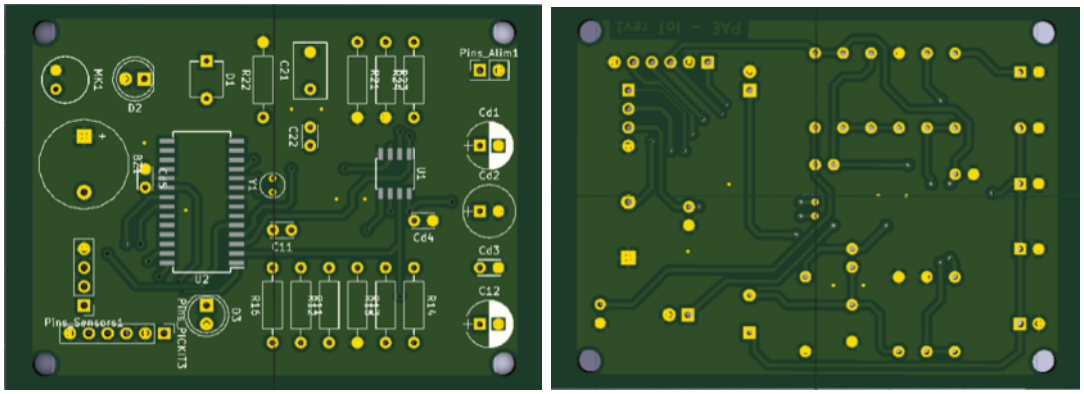


Figure 10: 3D simulation of the PCB from both ways

Final components list with values

Id	Designator	Package	Quantity	Designation
1	D2	LED_D5.0mm	1	LED
2	D1	PhotoDiode_DIL2_4.3x4.65_RM5.08	1	BPW34
3	BZ1	Buzzer_12x9.5RM7.6	1	Buzzer
4	C11,C22,Cd3,Cd4,Cd5	C_Disc_D3.0mm_W2.0mm_P2.50mm	5	100nF
5	C12	CP_Radial_D6.3mm_P2.50mm	1	1uF
6	C21	C_Disc_D7.5mm_W4.4mm_P5.00mm	1	330nF
7	Cd1	CP_Radial_D6.3mm_P2.50mm	1	10uF
8	Cd2	CP_Radial_Tantal_D7.0mm_P2.50mm	1	1uF
9	D3	LED_D5.0mm	1	LED_ON
10	MK1	Electret	1	Microphone
11	Pins_Alim1	Pin_Header_Straight_2x01_Pitch2.54mm	1	CONN_01X02
12	Pins_Sensors1	Pin_Header_Straight_1x04_Pitch2.54mm	1	CONN_01X04
13	R11,R13	R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	2	100k
14	R12	R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	470k
15	R14,R16,R21	R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	3	10k
16	R15	R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	18k
17	R22	R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	1M
18	R23	R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	1k
19	R24	R_Axial_DIN0207_L6.3mm_D2.5mm_P10.16mm_Horizontal	1	47k
20	U1	SOIC-8_3.9x4.9mm_Pitch1.27mm	1	LMV358
21	U2	SOIC-28W_7.5x18.7mm_Pitch1.27mm	1	PIC16F19156
22	Y1	Crystal_Round_d3.0mm_Vertical	1	Crystal
23	Pins_PICKIT3	Pin_Header_Straight_1x06_Pitch2.54mm	1	CONN_01X06
24	,,,	MountingHole_3.2mm_M3	4	

Circuit / device pictures

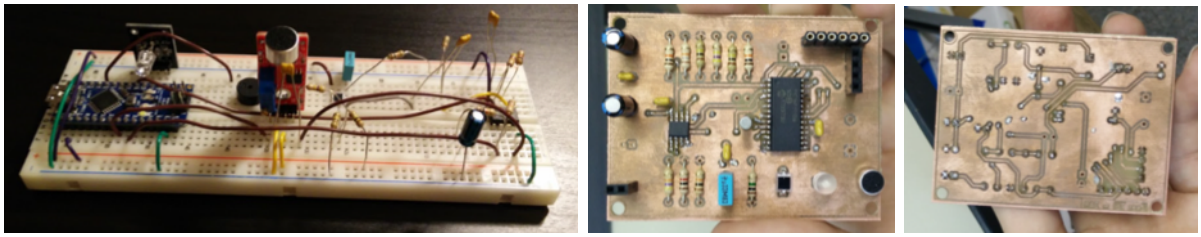


Figure 11: Breadboard implementation Final result (top) Final result (bottom)

As the soldering process was complex, a microscope was used in order to check al the pins were correctly in contact with each route.

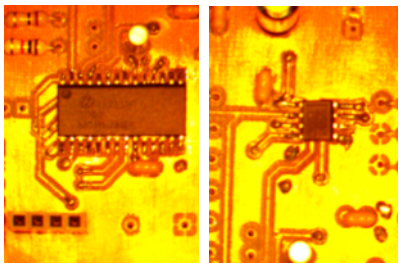


Figure 12: PIC16 MCU TLC2272 opamp

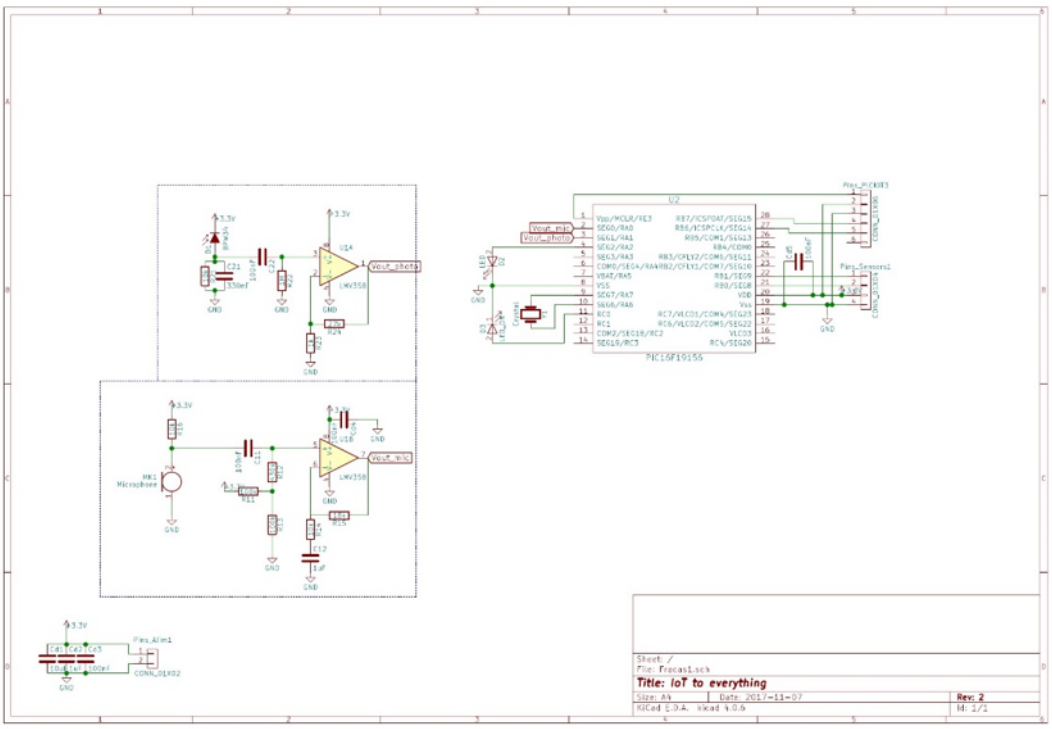
2nd Design

The aim of the second design is correcting all the problems that appeared in the testing process of the first version.

The changes that have been made are:

- Buzzer removal
- Microphone footprint change
- Pickit pins distribution change
- Values of R24 (as commented before, due to supply voltage change) and R12

Final schematics / software blocks



Circuit Layout / user interface screen captures

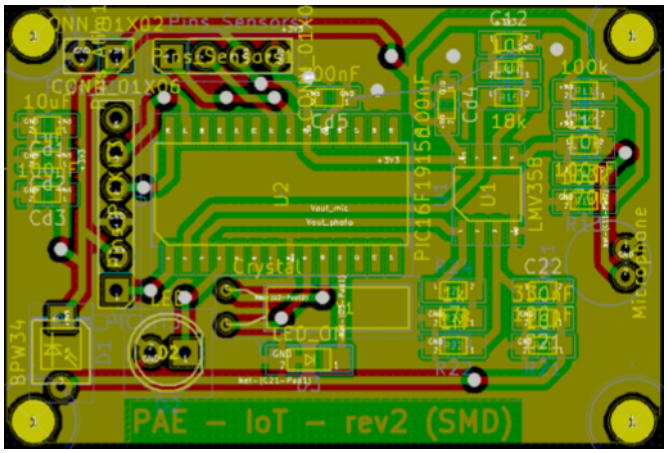


Figure 13: Bottom and Top Copper Layers of the PCB overlapped

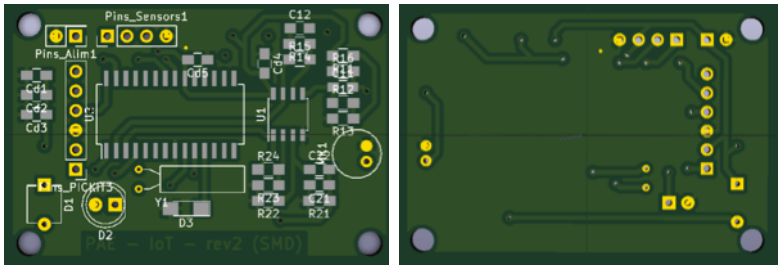


Figure 14: 3D simulation of the PCB from both ways

Final components list with values

Id	Designator	Package	Quantity	Designation
1	,,,	MountingHole_3.2mm_M3	4	
2	C11,C22,Cd3,Cd4,Cd5	C_0805_HandSoldering	5	100nF
3	C12,Cd2	C_0805_HandSoldering	2	1uF
4	C21	C_0805_HandSoldering	1	330nF
5	Cd1	C_0805_HandSoldering	1	10uF
6	D1	PhotoDiode_DIL2_4.3x4.65_RM5.08	1	BPW34
7	D2	LED_D5.0mm	1	LED
8	D3	LED_1206_HandSoldering	1	LED_ON
9	Pins_Alim1	Pin_Header_Straight_2x01_Pitch2.54mm	1	CONN_01X02
10	Pins_PICKIT3	Pin_Header_Straight_1x06_Pitch2.54mm	1	CONN_01X06
11	Pins_Sensors1	Pin_Header_Straight_1x04_Pitch2.54mm	1	CONN_01X04
12	R11,R13	R_0805_HandSoldering	2	100k
13	R12	R_0805_HandSoldering	1	430k
14	R14,R16,R21	R_0805_HandSoldering	3	10k
15	R15	R_0805_HandSoldering	1	18k
16	R22	R_0805_HandSoldering	1	1M
17	R23	R_0805_HandSoldering	1	1k
18	R24	R_0805_HandSoldering	1	27k
19	U1	SOIC-8_3.9x4.9mm_Pitch1.27mm	1	LMV358
20	U2	SOIC-28W_7.5x18.7mm_Pitch1.27mm	1	PIC16F19156
21	Y1	Crystal_AT310_d3.0mm_l10.0mm_Horizontal	1	Crystal
22	MK1	Electret	1	Microphone

Circuit / device pictures

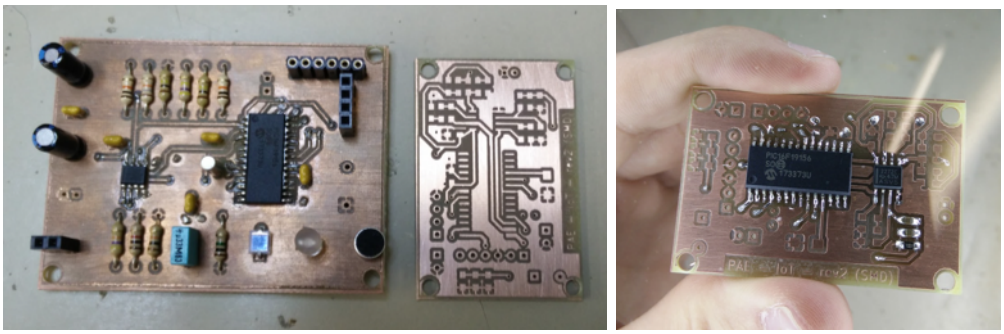


Figure 15: Size comparison between both PCB designs / PCB after 1st stage of soldering

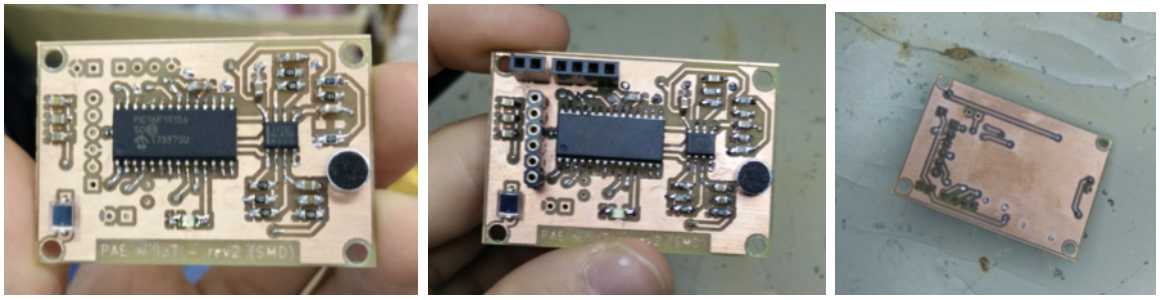


Figure 16: PCB after 2nd stage of soldering / PCB nearly finished / Bottom view of the PCB

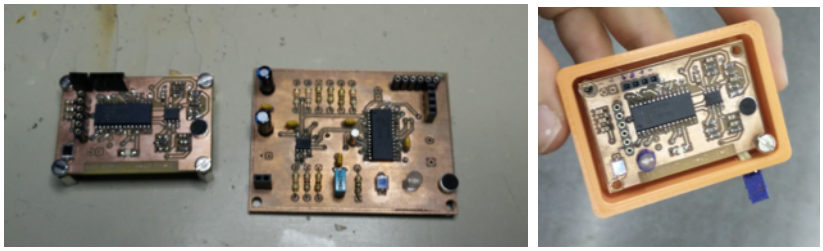


Figure 17: PCB comparison: Version 1 (right) Version 2 (left) / Final prototype

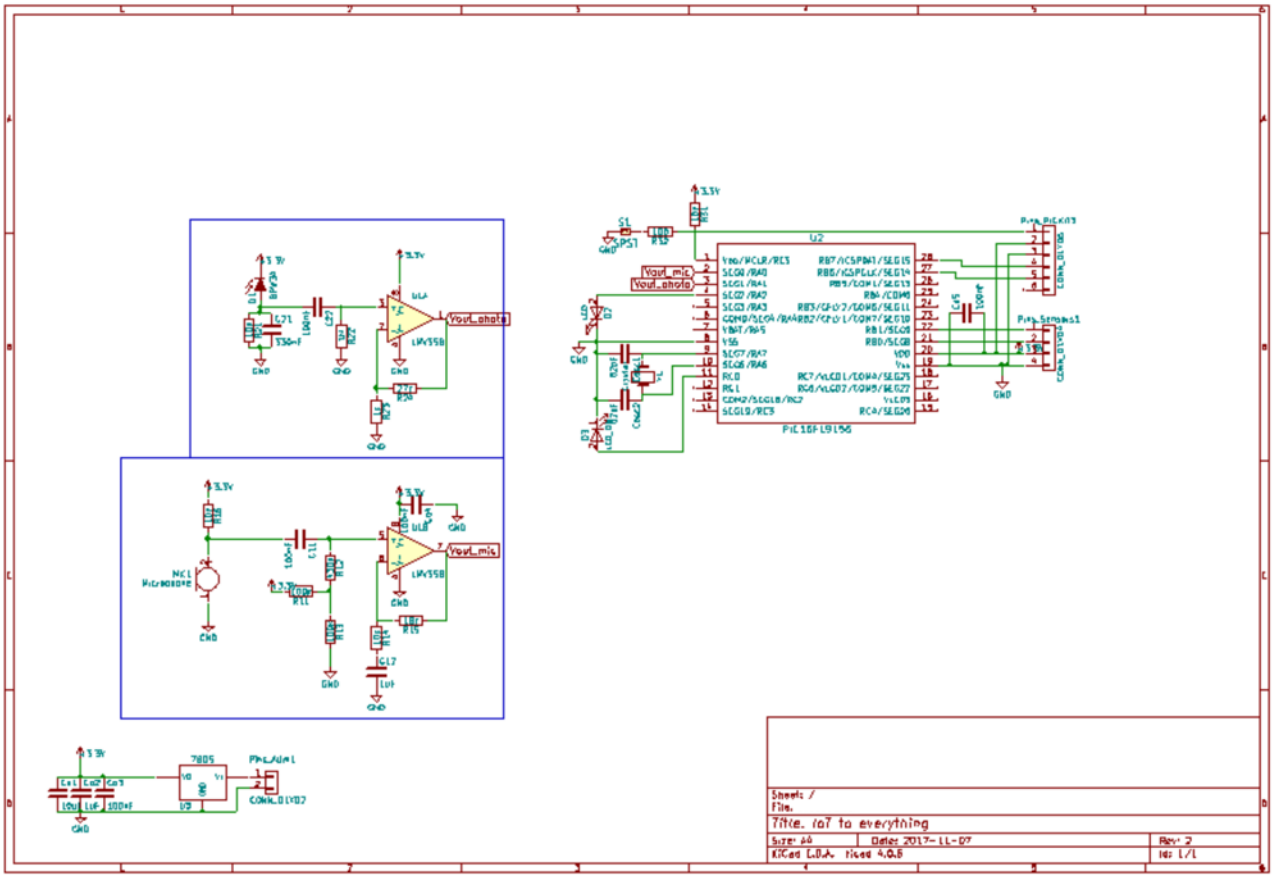
3rd Design

The aim of the third design is basically a final improvement and correction of the situation of the components in order to avoid light interferences and gain manoeuvrability while manipulating the prototype. Moreover, the design has been made with both sides of the PCB. So, the relevant components such as light receiver, microphone, led and reset switch are on top and the rest is protected by the case.

The changes that have been made are:

- Addition of a reset push button.
- Optimization of the design using both sides of the PCB.
- Addition of oscillator capacitors to stabilize the clock signal.
- Addition of a voltage regulator with gives 5VDC independently of the supply connected to the battery terminals.
- Addition of 9V battery cables and cable extensions for the PCB pins.

Final schematics / software blocks



Circuit Layout / user interface screen captures

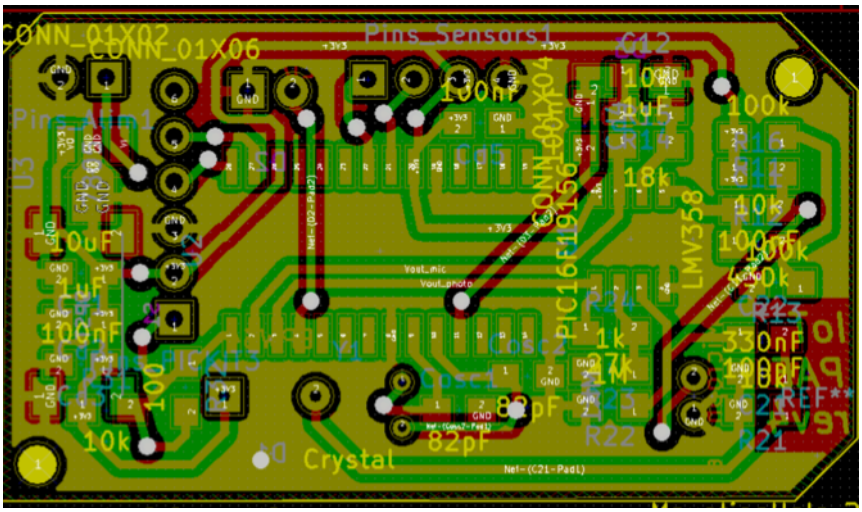


Figure 18: Bottom and Top Copper Layers of the PCB overlapped

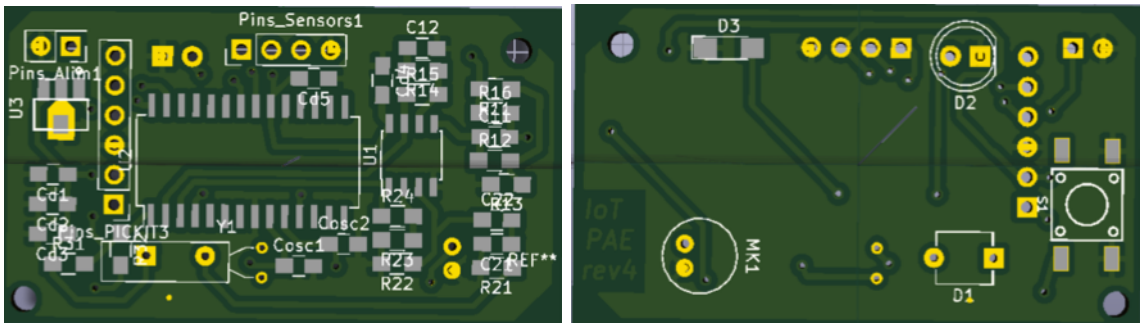


Figure 19: 3D simulation of the PCB from both ways

Final components list with values

Id	Designator	Package	Quantity	Designation
1	REF**	MountingHole_2.2mm_M2	2	MountingHole_ 2.2mm _M2
2	C11,C22,Cd3,Cd4,Cd5	C_0805_HandSoldering	5	100nF
3	C12,Cd2	C_0805_HandSoldering	2	1uF
4	C21	C_0805_HandSoldering	1	330nF
5	Cd1	C_0805_HandSoldering	1	10uF
6	D1	PhotoDiode_DIL2_4.3x4.65_RM5.08	1	BPW34
7	D2	LED_D5.0mm	1	LED
8	D3	LED_1206_HandSoldering	1	LED_ON
9	Pins_Alim1	Pin_Header_Straight_2x01_Pitch2.54mm	1	CONN_01X02
10	Pins_PICKIT3	Pin_Header_Straight_1x06_Pitch2.54mm	1	CONN_01X06
11	Pins_Sensors1	Pin_Header_Straight_1x04_Pitch2.54mm	1	CONN_01X04
12	R11,R13	R_0805_HandSoldering	2	100k
13	R12	R_0805_HandSoldering	1	430k
14	R14,R16,R21,R31	R_0805_HandSoldering	4	10k
15	R15	R_0805_HandSoldering	1	18k
16	R22	R_0805_HandSoldering	1	1M
17	R23	R_0805_HandSoldering	1	1k
18	R24	R_0805_HandSoldering	1	27k
19	U1	SOIC-8_3.9x4.9mm_Pitch1.27mm	1	TLC2272
20	U2	SOIC-28W_7.5x18.7mm_Pitch1.27mm	1	PIC16F19156
21	Y1	Crystal_AT310_d3.0mm_l10.0mm_Horizontal	1	Crystal
22	MK1	Electret	1	Microphone
23	Cosc1,Cosc2	C_0805_HandSoldering	2	82pF
24	R32	R_0805_HandSoldering	1	100
25	S1	SPST-4-PIN_6mm_sm	1	SPST
26	U3	SOT89-pins-reg	1	7805

Circuit / device pictures

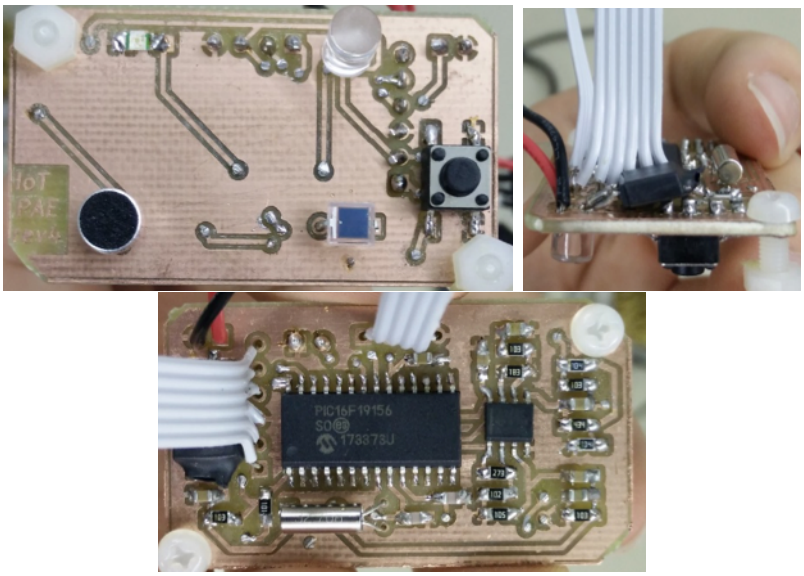
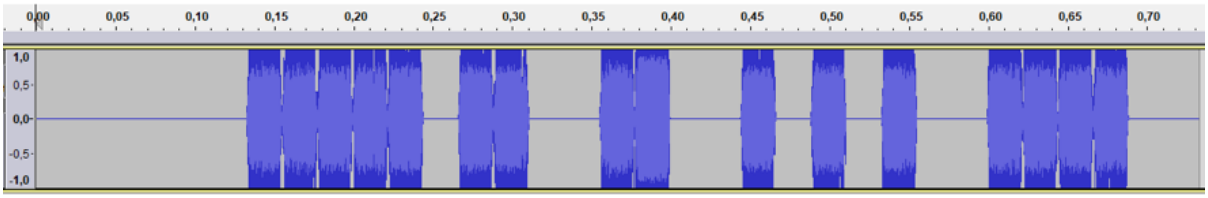


Figure 20: Finished PCB seen from different points of view (top, side and bottom respectively)

5.5 Sound Team

We decided to use white gaussian noise instead of a sinusoid in order to simplify the decoding process as much as possible and obtain higher bitrates.

Our signal is white noise modulated in amplitude with a binary function. As it can be seen in this screenshot, the final transmitted signal from a Smartphone fits the specifications of our AM signal:

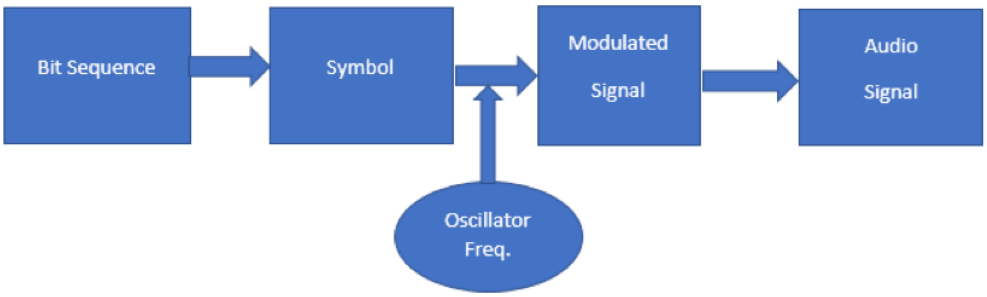


To decode the data without error we have added a synchronism sequence repeated every 42 bits (6 packets). This sequence is known a priori by the receiver.

Modulator

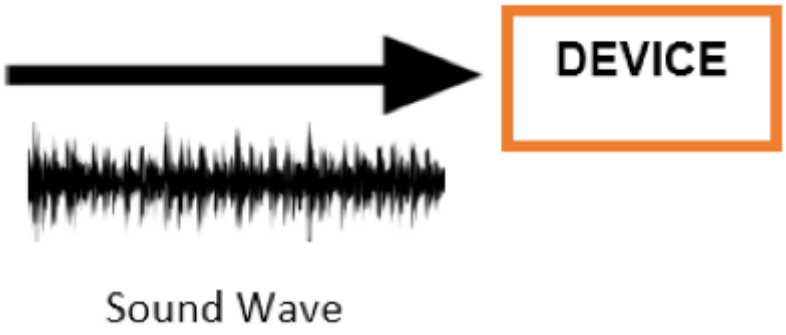
	SMARTPHONE
Testing Software	MATLAB, Java
Programming language	Java
TX / RX	TX - Speaker
Modulation	AM – Noise generator
Fs	44100 Hz
N symbol	2
BitRate Max	45 bps

The code developed in this section for the modulator is integrated in the Android application.



Demodulator

	DEVICE
Testing Software	MATLAB, MPLAB, Oscilloscope
Programming language	C
TX / RX	RX - Microphone
Modulation	AM – Noise detector
Fs	8000 Hz
N symbol	2
BitRate	45 bps



The demodulation consisted in a state machine of 3 different states: Listen, Sync and Read. The Listen state consisted in waiting until a strong audio is received in the microphone (a high level). The Sync state synchronizes the timing in order to read the data correctly, it waits until the signal goes from a high level to a low level. The Read state simply decodes the data using a fixed threshold.

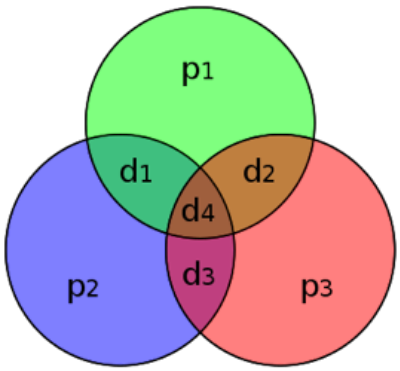
To recover the data, we need to follow the inverse process and transform the signal into information (symbol/byte/bit). We are implementing it in C as it must be run in the microchip's side.

To decide if we are receiving data or just noise, we are fixing different thresholds like:

- Signed short listen_threshold= 500
- Signed short read_threshold= 410

In order to test and confirm that we are obtaining the data back in the demodulator, we are using the PickIt, microchip and LEDs to check visually if it decodes the appropriate symbols.

We have implemented a FEC (Forward error correction with a Hamming Code (7,4) based in the following table. This way, incorporating 3 bits of redundancy for every nibble, the decoder has a correction capacity of 1 bit for every package of 4 bits.

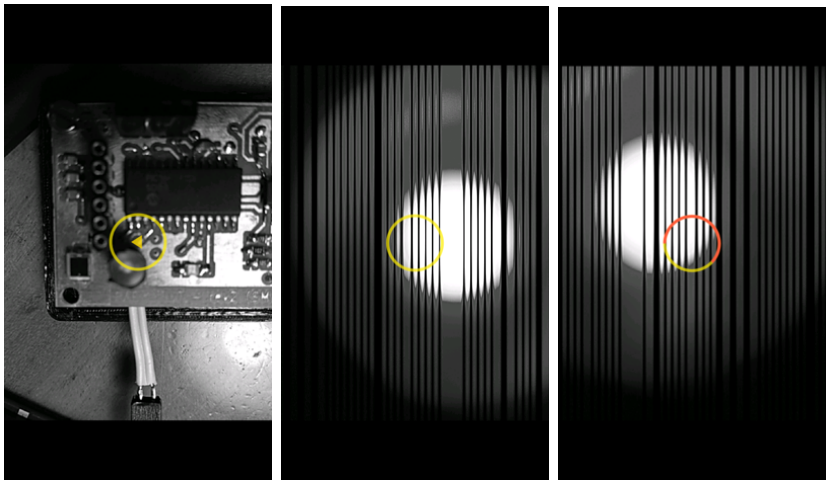


Bit #	1	2	3	4	5	6	7
Transmitted bit	p_1	p_2	d_1	p_3	d_2	d_3	d_4
p_1	Yes	No	Yes	No	Yes	No	Yes
p_2	No	Yes	Yes	No	No	Yes	Yes
p_3	No	No	No	Yes	Yes	Yes	Yes

6 System Characterization

6.1 Protocol Team

6.1.1 Light Communication



These three snapshots capture the three states of the light communication, IDLE (Waiting for the user to Tap), Start decoding, In Progress to Decode a frame.

One of the main objectives we wanted to accomplish is to develop a fully automatic and user-friendly interface, and so we did. The user only has to start the communication, after that, the flash modulated 'acknowledge'/'non-acknowledge' packages will control the state of the data transfer between device and smartphone. The application will return to its original state with the received data from the device.

6.1.2 Sound Communication

The communication between smartphone - device relies on sound commands followed by data to communicate. We have developed two different commands:

- **Install Device:**

  Used once to set an ID and the actual timestamp to the device.

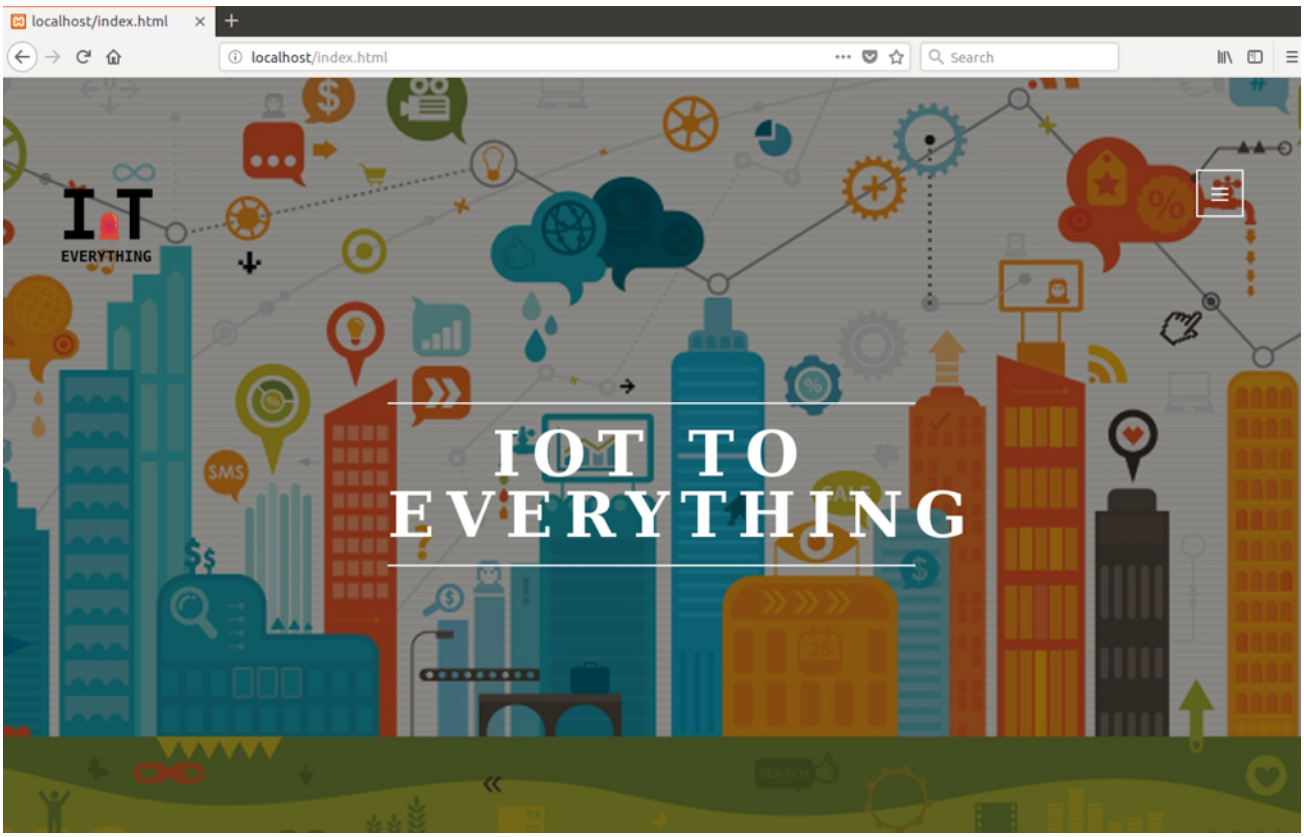
- **Set Alarm:**

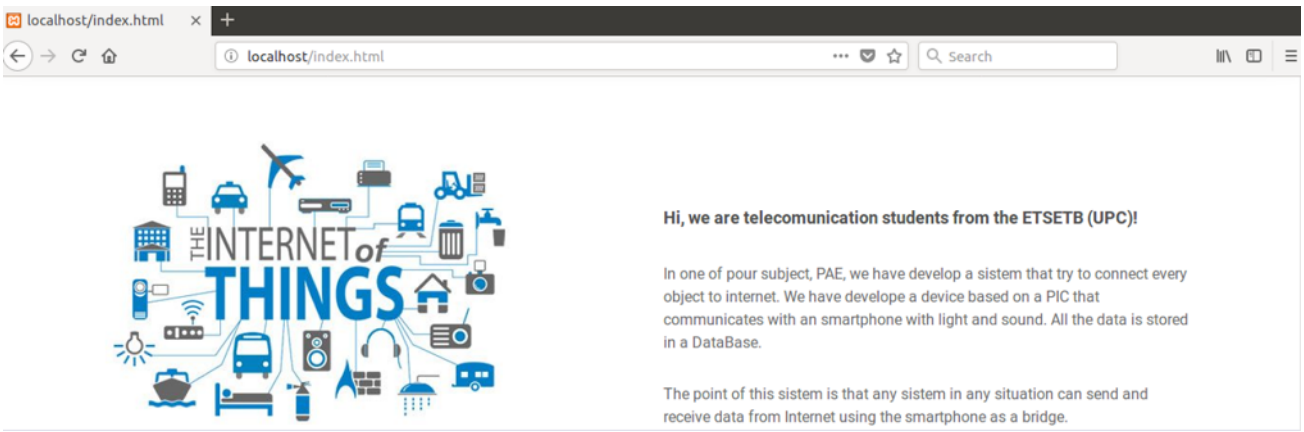
  Used every time it is necessary to reset the alarm to another time period.

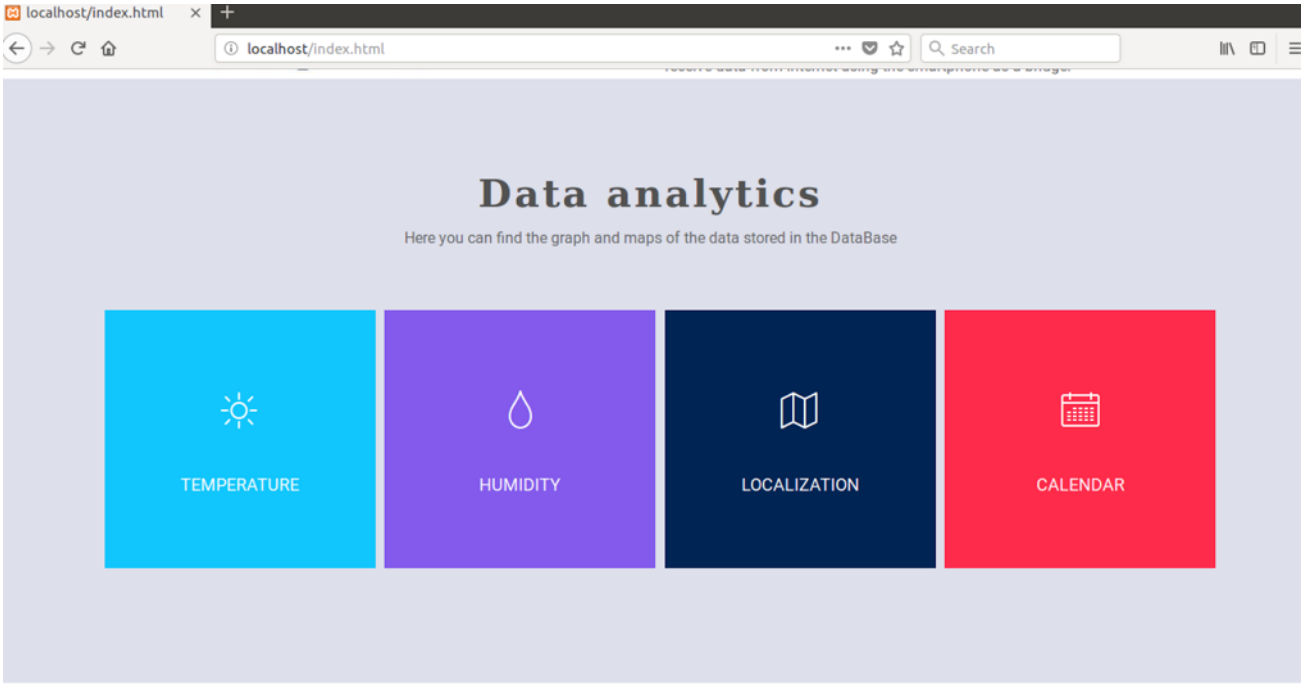
6.2 Platform Team

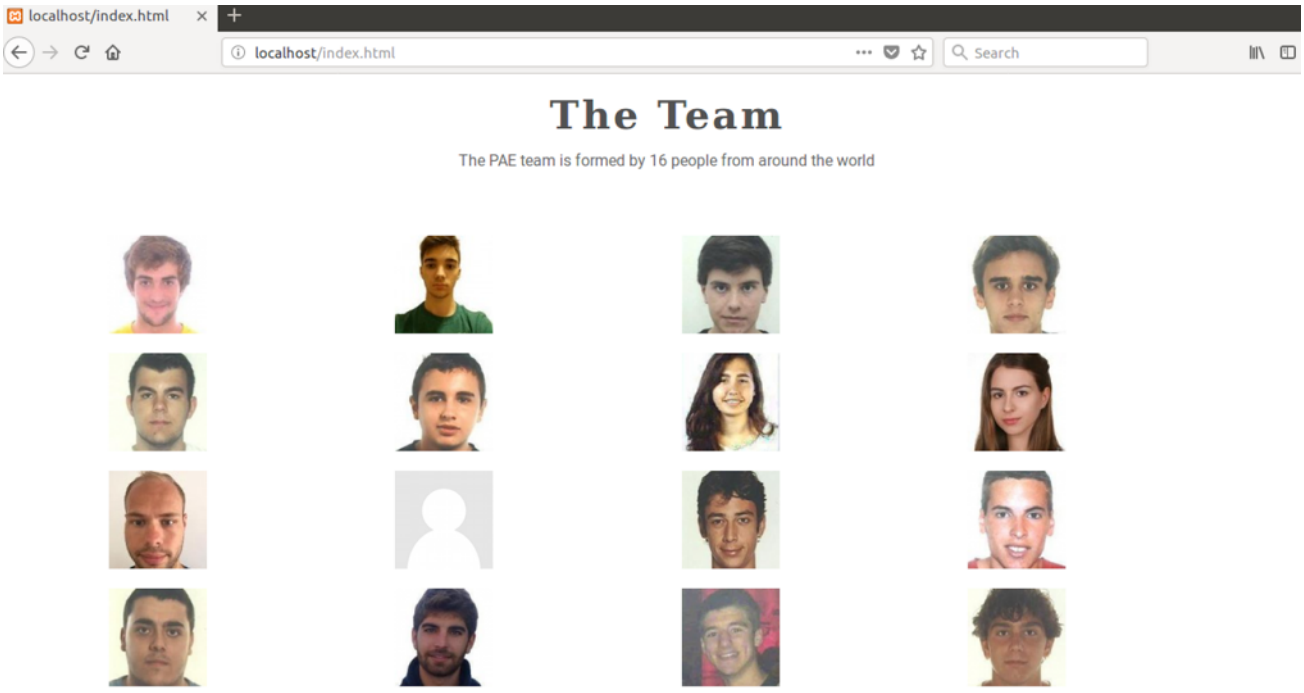
6.2.1 Database Team:

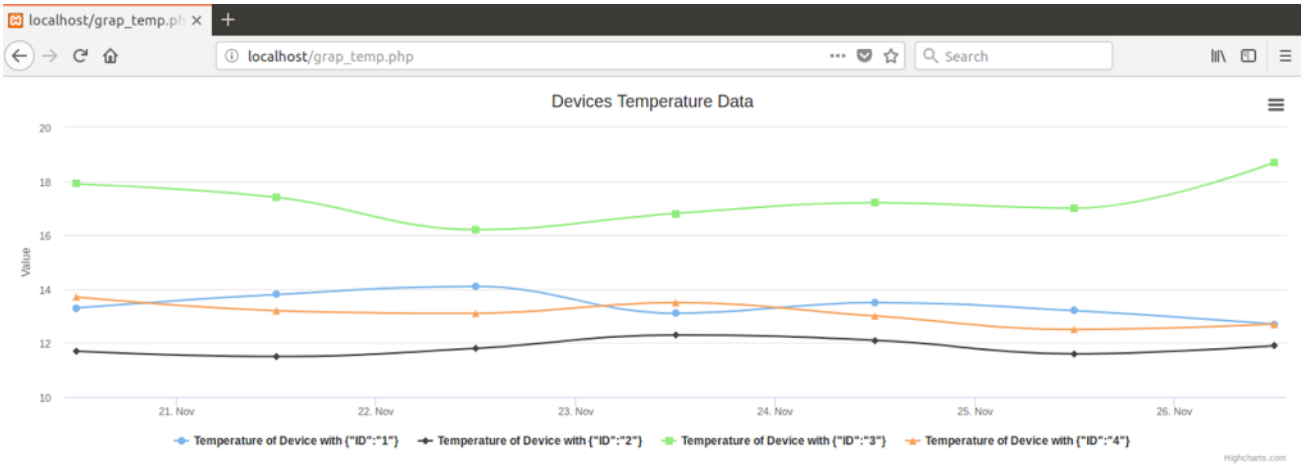
The user will only see the user interface, that will be the android application and the website. The website automatically analyze the data from the database and show manager charts and graphic representations.



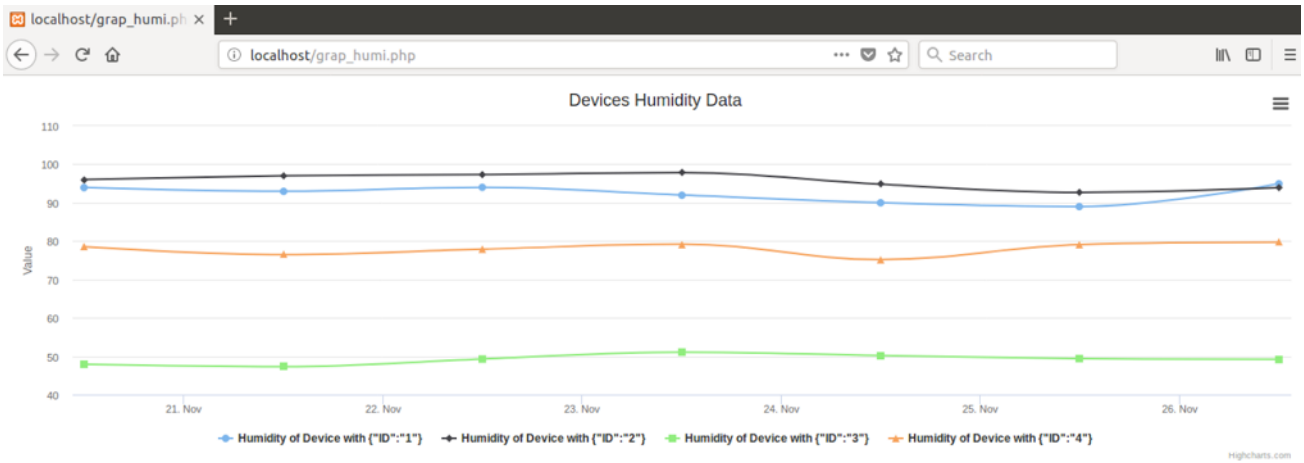



localhost/index.html
localhost/index.html
Search
Data analytics
Here you can find the graph and maps of the data stored in the DataBase
TEMPERATURE
HUMIDITY
LOCALIZATION
CALENDAR

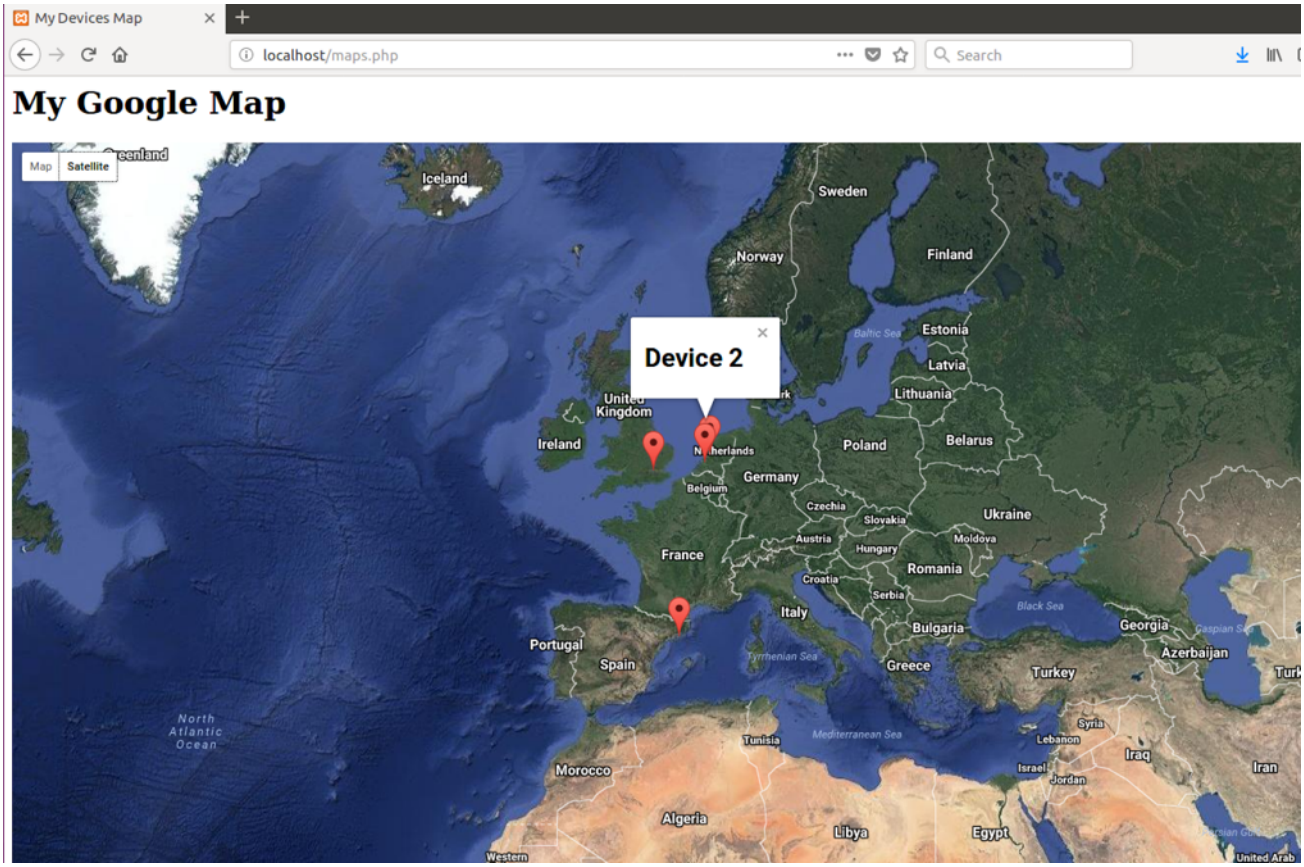

localhost/index.html
localhost/index.html
Search
The Team
The PAE team is formed by 16 people from around the world



Go Back



Go Back

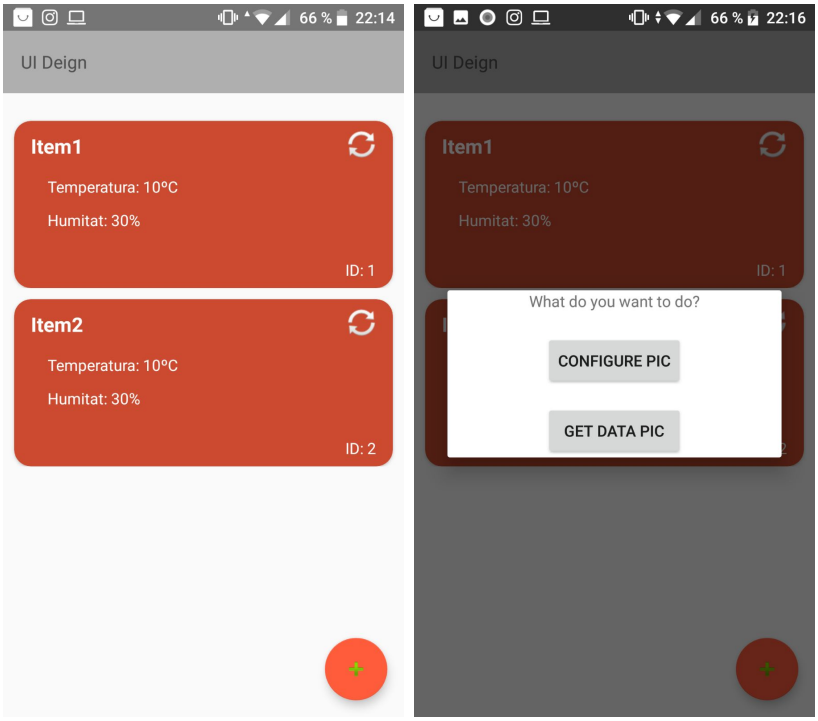


All the code can be found in the GitHub page of the web:

https://github.com/sebastienkanj/PAE_IoT_web

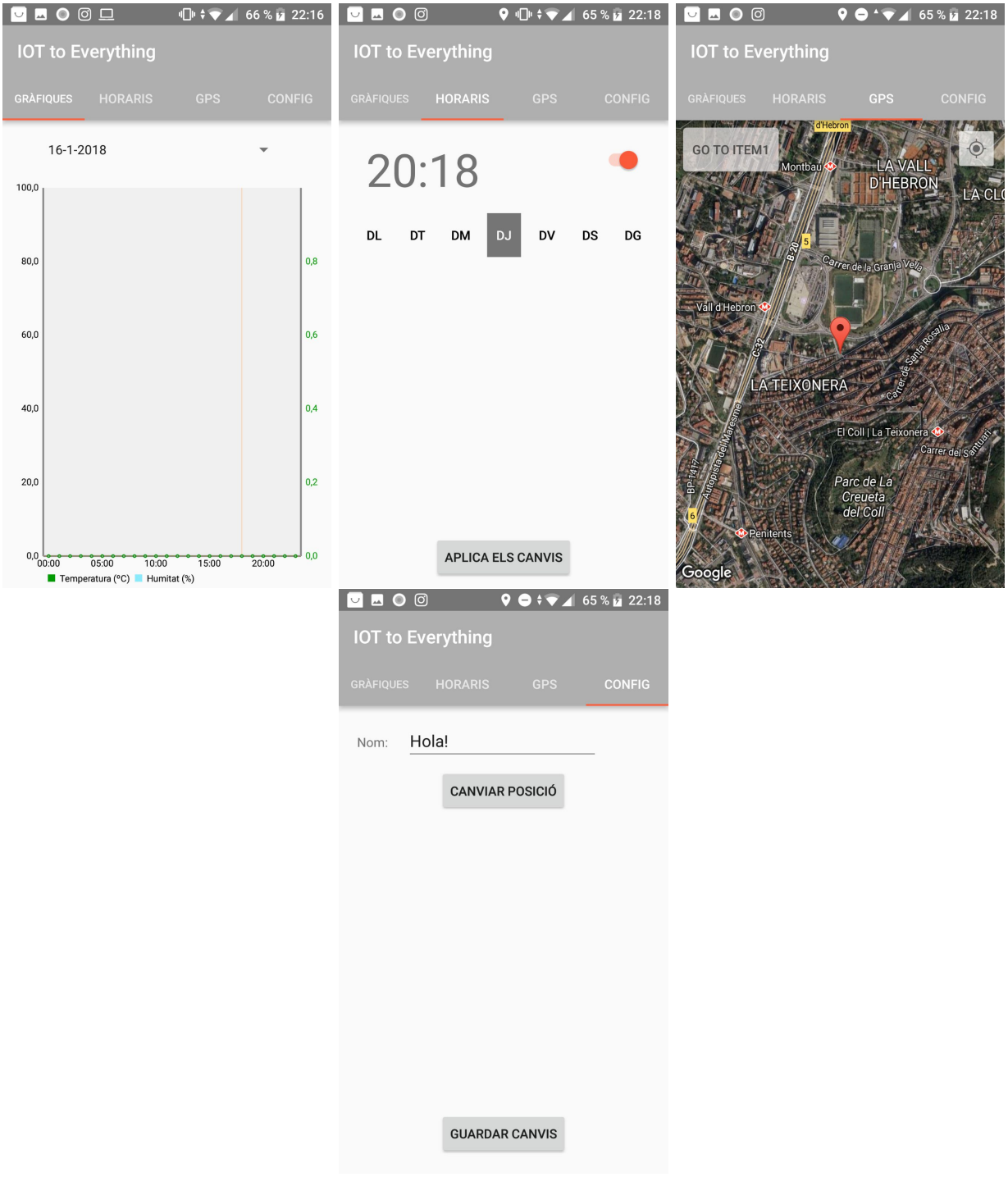
6.2.2 Android App

The final app is very similar to what we planned, but we had to cut out some functionalities due to the time and some problems that with had merging the last part, which was the communication with the device.



The main change of the principal activity is how we change the data with the device, only with two buttons. One to configure a new device that has not been initialized, and the other one to obtain the data of an initialized device.

In the second part, we can see little changes



The main difference is found in the time tab. Due to the limitations of the device is not possible to configure different alarms, only one.

In the map section a button is added to guide the user to where the device is located.

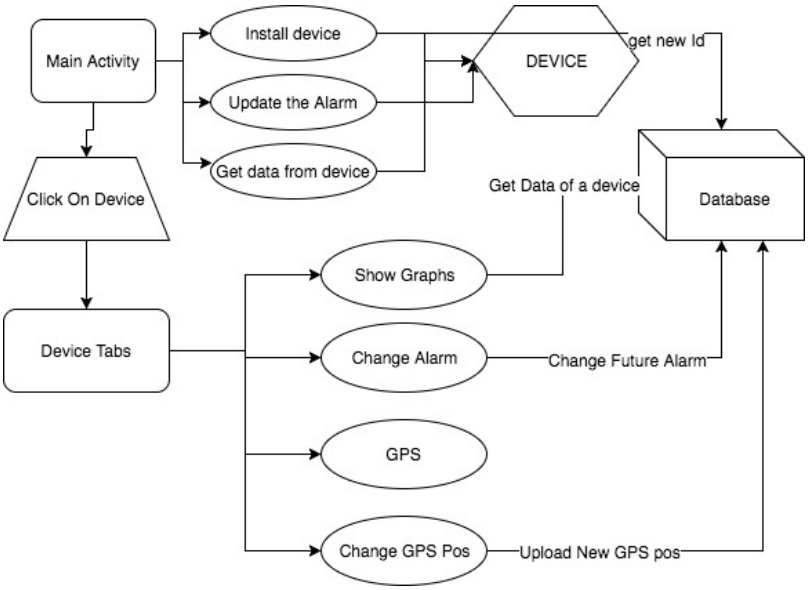
The configuration section has only the ability of changing the location of the device to the current one of the user.

About the API, all remains the same except of the part that obtained the data of a day and

a device. We had to change it to get all the data of a device.

The information of the communication can be found in the other parts of the project.

To help to understand how the application works, you have below a graphic with all the functionalities:



All the code can be found in the GitHub page of the android app:

www.github.com/guillemlla/iottoeverything

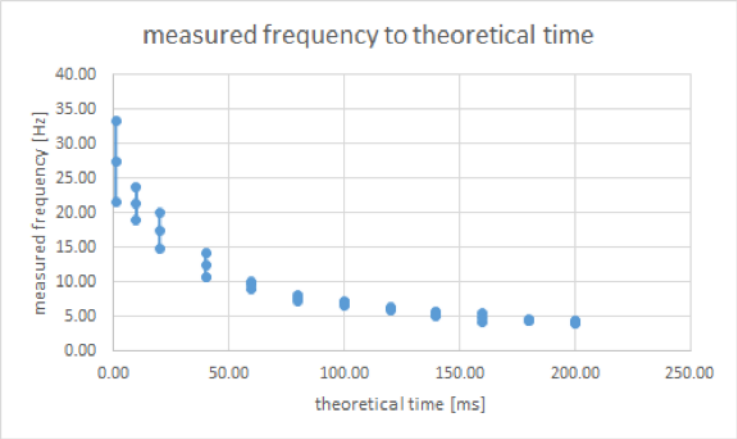
6.3 Camera Team

6.3.1 Flash Team

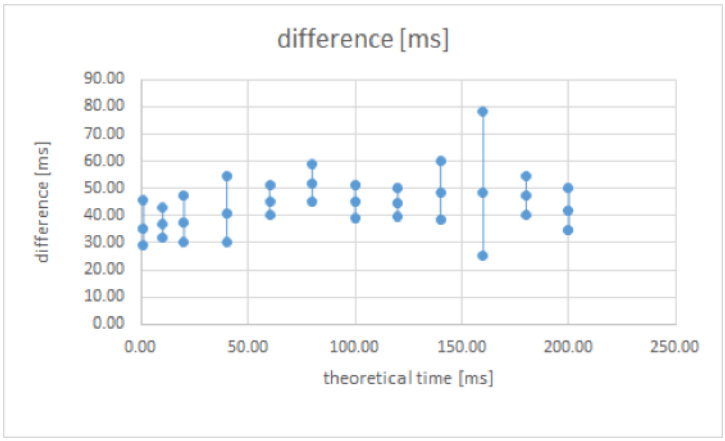
We did tests to check dependence between theoretical time of one bit assigned in application and real frequency measured needed to change the light from on to off and conversely.

We managed to achieve the range of frequency from 4.00 to 33.40 Hz.

It was also observed that for the high frequency the difference between maximum and minimum frequency is significant. With the frequency decreasing, this difference is also getting less significant.



Moreover, we also observed some delay, time that smartphone needs to turn flash on/off. It is difference between theoretical time assigned in applications and time calculated from measured frequency (t=1/f) Results are presented below:



According to our research, the best suggested frequency to apply for the device is for theoretical time 120ms (assigned in application) which is real frequency 6.07 Hz. It is because for that frequency difference between theoretical time and measured is one the lowest and at the same time the minimum and maximum measured frequencies are similar to each other, so fluctuations should not disturb communication between application and device.

6.3.2 Image Processing

The final implementation of image processing has been done in Android Studio with OpenCV Library. The app has two working states, one for wait and other for starting reading the data. The following images shows this two states:

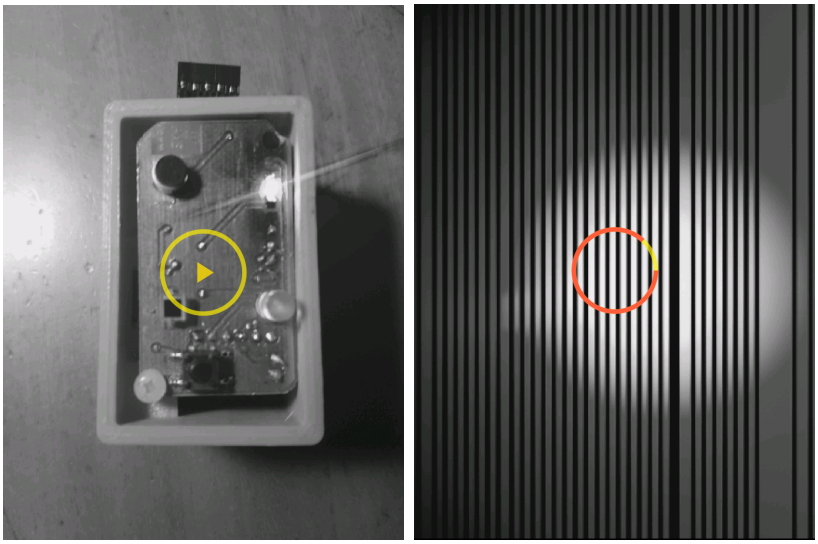


Figure 21: Wait state / Reading state

With this implementation we achieved a bitrate of 1.2 kbps.

As a conclusion of our part, we can say that the image processing in OpenCV have some errors due to variance of light around the entire image. However, we fixed that errors using a forward error correction codes. This team has work with a new way of communication, so we had some problems on finding information.

6.4 Electronic Team

6.4.1 Software Team

Finally, we have completed the state machine and the figures below show the different states of the PIC:

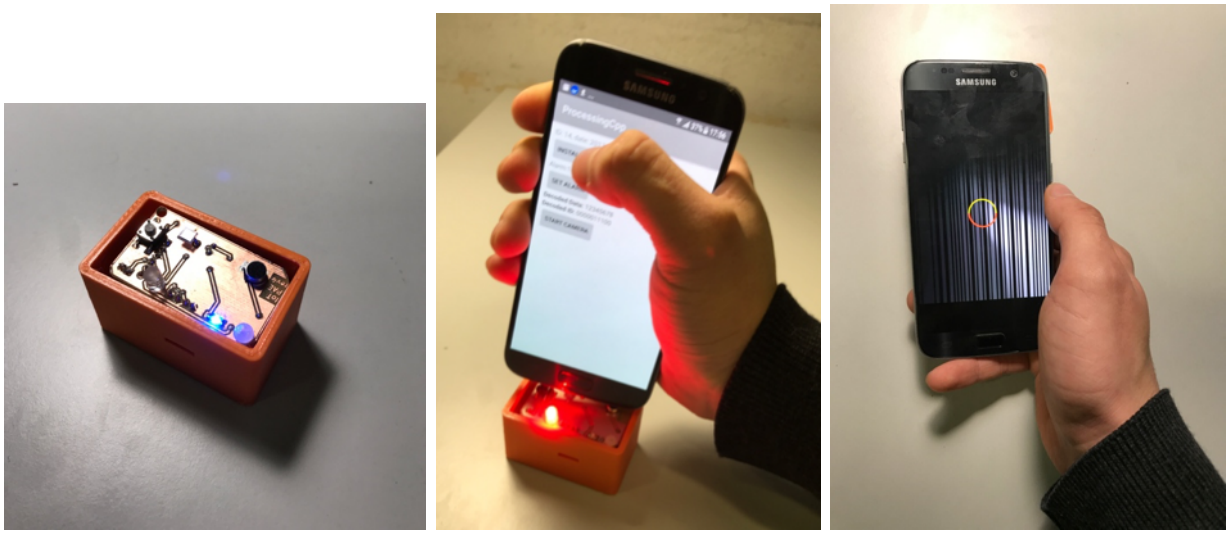


Figure 22: Listen state / Config PIC state / Send data state

On the other side, we have been working on the internal RTCC precision, however the precision is not the highest one. More or less we achieve an error of 4 minutes per hour. The error is caused by the working of two oscillators simultaneously. The PICs CPU is working at 32 MHz and the RTCC must work at 31-32 kHz, conclusively we needed an Internal High-Frequency Oscillator and a Secondary Oscillator to work at the same time.

All the code can be found in the GitHub page of the electronic team:

https://github.com/gferrate/pae

6.4.2 Hardware Team

To check the correct behavior of the PCB prototype sensing part, we measured the signals that came to the input pins of the PIC as well as the supply voltage filtering.

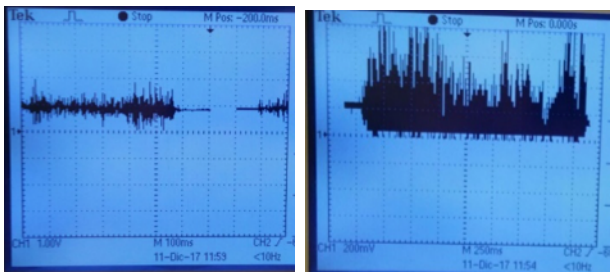


Figure 23: Perfectly adapted audio at the PIC input pin with VDD/2 DC voltage / High gain audio input which saturates

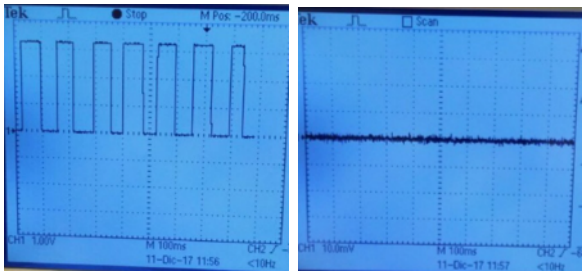


Figure 24: Pulses at the PIC input pin due to a phone flash / Supply voltage in AC mode after being filtered

As a final test, the supply voltage coming from the output of the regulator and the input signals that enter to the PIC were checked from the 3rd prototype shown below:

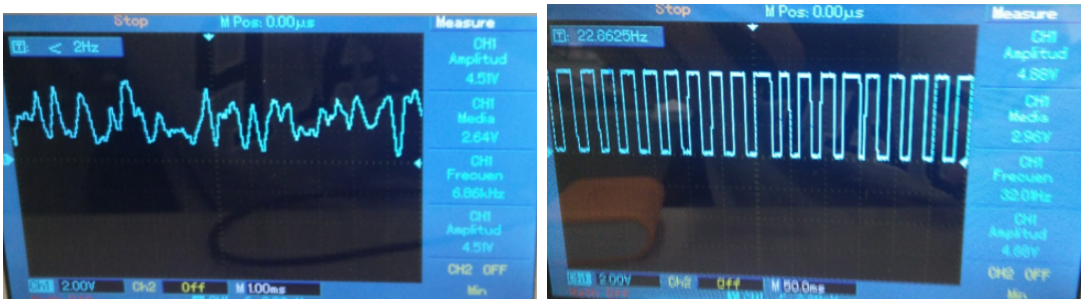


Figure 25: Perfectly adapted audio at the PIC input pin with VDD/2 DC voltage / Pulses at the PIC input pin due to a phone flash

Figure 26: Supply voltage from the 7805 regulator

6.5 Sound Team

Transmitting data over sound has advantages like not being dependent on standard communication systems or the fact that it is a widely extended technology existing in many devices. The main inconveniences are the low bit rates that are reached and the noisy environments.

When we first decided to use a QPSK modulation, most of the problems appeared when we imported our code to the PIC micro-controller. We had lots of trouble configuring the

ADC controller, we could configure it but the sampling frequency we achieved was not very consistent and had a little jitter. For this reason, the phase couldn't synchronize and the overall modulation couldn't work.

We noticed that, because of the inconsistency of the sampling frequency and some rounding errors, the demodulation desynchronized when we attempted to send large bit streams. To solve this problem, we changed the modulator in order to resynchronize the demodulator after a fixed number of bits.

After these problems, we forgot about the QPSK and we focused in a simple binary amplitude modulation. (AM). We decided to use white gaussian noise instead of a sinusoid in order to simplify the decoding process as much as possible and obtain higher bitrates. The maximum bitrate we could achieve was 50 bps, but we stepped it down to 45 bps to make sure that it would work perfectly. After this little change, we could send as much information as we wanted without fear.

The demodulation process has been checked with the PIC, using the 3 state machine, and experimental thresholds

7 Costs

Our project is a device composed of a micro-controller and a series of sensors that interact with a smartphone connected to the internet. This device can be part of another larger and complex system, so we do not dedicate ourselves to marketing it. In this case, it is not necessary to make a liquidity plan since our project is part of a more general project that is where these types of studies are carried out. It would not make sense to start talking about loans, stock, depreciation, materials, machinery, advertising, revenue ... when we basically dedicated ourselves to programming and testing to verify the operation of our device, and not to sell it.

7.1 Activity based costing

Activities	**People**	**Hours/Person**	**Total hours**	**Cost**	
Research	2	5	10	80 €	16%
Research tecnic information	12	45	540	4.320 €	
Design	4	45	180	1.440 €	
Programmation	7	175	1225	9.800 €	37%
Create	2	30	60	480 €	
Test	8	115	920	7.360 €	27.6%
Mix all the project parts	5	55	275	2.200 €	8%
Writting documentation	12	10	120	960 €	
			3330	26.640 €	
Other costs					
Local rent				3.000 €	
Computers				7.500 €	
National insurance				7.992 €	
Total				45.132 €	

With this table of activity based costing we can draw a series of conclusions: first of all we can see that there is a large initial investment both of time and consequently of money in

the search of information (16.5% of the time), and we should consider whether to allocate first to many people in the search or if it would be better for some workers to join the work when they already had this information and had been processed. Another way in which the company would save money would already be to provide reliable sources of information, or a large part of the information necessary to start working directly. It would also be a great benefit if workers could devote their time to tasks much more important than the search for information. The second conclusion that we can take is that programming is the main part of the project (37%) because possibly you should be given more resources both at the personnel level, in order to reduce the number of hours, as well as computers so that each worker can use one and spend more hours to program. We also see that testing the circuit and programs, correcting the errors and then joining all parts of the project take up more than one third of the time, a logical and normal thing. We make a great investment here since it is essential to meet the needs that all parties work correctly without errors first individually and then all together, and this will lead to greater benefit.

7.2 Components list of the prototype

Products	Price 1 Product	Quantity	Price 1	Price 1000+ Products	Price +1000	Price 5000+ Products	Price +5000	Price +10000 Products	Price +10000
LED	0.0659	1	0.0659	0.0596	0.048	0.0346	0.0346	0.048	0.048
BPW34	0.822	1	0.822	0.38	0.296	0.33	0.33	0.296	0.296
Buzzer	0.356	1	0.356	0.198	0.187	0.191	0.191	0.187	0.187
100nF	0.07	5	0.35	0.062	0.048	0.055	0.275	0.048	0.24
1uF	0.0507	1	0.0507	0.0219	0.0154	0.017	0.017	0.0154	0.0154
330nF	0.0536	1	0.0536	0.0232	0.0162	0.0179	0.0179	0.0162	0.0162
10uF	0.376	1	0.376	0.0903	0.0614	0.0765	0.0765	0.0614	0.0614
1uF	0.0452	1	0.0452	0.0196	0.0137	0.0151	0.0151	0.0137	0.0137
LED_ON	0.045	1	0.045	0.027	0.0257	0.0259	0.0259	0.0257	0.0257
Microphone	0.63	1	0.63	0.3	0.238	0.256	0.256	0.238	0.238
CONN_01X02	0.086	1	0.086	0.0623	0.0375	0.0427	0.0427	0.0375	0.0375
100k	0.0529	2	0.1058	0.0169	0.0045	0.0094	0.0188	0.0045	0.009
470k	0.0802	1	0.0802	0.0194	0.0099	0.0122	0.0122	0.0099	0.0099
10k	0.0525	3	0.1575	0.0168	0.007	0.0093	0.0279	0.007	0.021
18k	0.118	1	0.118	0.0217	0.0143	0.0146	0.0146	0.0143	0.0143
1M	0.23	1	0.23	0.041	0.0234	0.0284	0.0284	0.0234	0.0234
1k	0.12	1	0.12	0.022	0.0139	0.0148	0.0148	0.0139	0.0139
47k	0.225	1	0.225	0.04	0.0131	0.021	0.021	0.0131	0.0131
LMV358	0.46	1	0.46	0.19	0.128	0.152	0.152	0.128	0.128
PIC16LF19156	1.35	1	1.35	1.12	1.12	1.12	1.12	1.12	1.12
Crystal	0.187	1	0.187	0.146	0.073	0.094	0.094	0.073	0.073
		Unit price	5,9139		3,1762		2,6914		2,6045

From these tables, we can reach a logical conclusion as it is more convenient to buy the components to make the maximum number of units possible. Taking the total 2.6045 €that costs each unit in the case of making 10000, we see that we do not adjust to the target of the marked price that was 2 €. Surely, we could adjust more to the price or even to fulfill it if

we used a simpler PIC, since we chose a good quality to ensure that it supported the entire load and in the end, it proved not to be so necessary. On the other hand, we could also save with a cheaper silicon photodiode to reduce the total price.

8 Sustainability Analysis

	PPP	Lifetime	Risks
Environmental	Design consumption	Ecological footprint	Environmental risks
	8/10	15/20	-3/-20
Economical	Budget	Viability plan	Economical risks
	8/10	20/20	-5/-20
Social	Personal impact	Social impact	Social risks
	10/10	17/20	-0/-20
Sustainability	26/30	52/60	-8/-20

8.1 Environmental

- Design consumption (8/10): This project doesn't have a huge environmental impact since we only use a few small electronic components, explained in costs of the project, and the principal work it's done in software, even though devices like batteries or the microphone have a great environmental cost despite being small. If we would do the Project again, we could use only the electronic final components, so we would do this with less resources.
- Ecological footprint (15/20): We can say that this device decreases the environmental footprint, we focused on improving the comfort when interacting with objects, but also, we proposed a way to save water. Since we obtain data on temperature and humidity, we can water in a much more efficient way. The only resource that we are going to consume during the lifetime of the device are the device and mobile batteries.
- Risks (-3/-20): One thing that could increase the ecological footprint after the dismantling is a bad recycling of the device elements. We must be very careful at the time of the dismantling, since these devices, although they are small, are very dangerous if are left outdoors without recycling. Another risk would be the waste of water caused by a malfunction of the device due to the deterioration of any component.

8.2 Economical

- Budget (8/10): As this is explained in cost analysis, there is a low material costs but some decent human resources. We can reduce the material costs by buying more components and also reduce the human resources by working with a more efficient way. We had bigger costs than the expected caused by the use of a different PIC, improving the quality that it offers us
- Viability plan (20/20): The device hardware doesn't need a huge economical cost during the useful time, only the cost of a component that we are forced to replace. But we will have some costs due to the salary of people who will work to solve customers' problems or people improving the software of the application. We will also have some costs with the dismantling, which as we have said before, we will have to be very careful.
- Economical risks (-5/-20): We think that our product will not have viability problems unless people will stop taking advantage of it caused by a drastic change of conditions. Even though in this case we could upgrade our product to satisfy people. A problem that we could have is that suddenly a company better known than us, bring to the market a better product or equal to ours. As we are not a well-known company, it would be a complication.

8.3 Social

- Personal impact (10/10): Carrying out this project has been one of the things we will benefit most from this degree. Teamwork, determination and a lot of experience. So, the personal impact has been so huge. It is very likely that companies look for these kinds of attributes.
- Social impact (17/20): We believe that this product will improve the comfort of all those who buy it as well as saving water as mentioned in the economic section. On the other hand, if our product were to be successful, there would be a sector of society that would be unemployed, for example, some gardeners. Although we will also create jobs, for example, on data analysis or software programmers.
- Social risks (-0/-20): There are not scenarios that can be socially harmful for a segment of the population when the dismantling. Our product does not create any kind of dependency to the users. If a person decides to stop using our product, he could return to the same situation as he had before without any problems.

9 Conclusions

After working these months with this project, we can say that we didn't hit the initial planning, since it was unrealistic for the time we had. Also, in certain aspects about the design and implementation, we have wanted to design the difficult version before having the easiest one perfectly implemented and working. An example of this would be that we implemented the sound using the sound phase, possibly if we worked first on sound amplitude, we would have avoided some problems. We have seen that one way to do an accurate design is to have a simple prototype that works and then to improve it.

On the other hand, not all the problems have been due to the initial Planning, we also had certain limitations due to the components used or the lack of experience that we still have. These problems caused us that certain goals we initially had, have not been achieved.

We have to differentiate these unfulfilled goals in two. The first is those problems that we could solve in a short term and the second one, those that would make us rethink the whole system.

The first objective not achieved is to have a device cost of less than 2 euros, this is due to the fact that we used a more expensive PIC than the one we initially supposed to use. This is an example of a resolution that would make us rethink part of the system if we decide to achieve that goal and use a cheaper PIC.

Another problem is that the PIC's RTC only allows us to program an alarm one time per day. This is an example of the limitation that the Android group has had mixing the work of each team. We could improve with a short-term upgrade, having multiple alarms at the times we want.

Most of the limitations that have been taken, have been due to limitations of the hardware used, such as the LED, which has given us a lower speed and more errors than expected. This would be another problem that would involve a reconsideration of the System if we would want to solve it.

10 Reflection Document

In general, the organizers have guided us well during the performance warning us of the different problems that we would confront, and thus saving us a lot of time. The only thing they could have been done better is to define the tasks better at the beginning, so we could be able to assign people knowing our qualities.

One of the things we could have been done better as a team is to communicate more often and more accurately between the different subgroups. We worked pretty good individually or in pairs, but this lack of communication has made us difficult to merge the work of the different groups. If we had communicated better, we would all start the work from the same point and we could avoid these problems.

Another thing that we could have been done better is to work harder since the first day. We performed hard the last month, but we wasted a lot of time working inefficiently the first weeks.

Even though we could work better improving this explained points, we should be proud of having completed a project with a group of 16 people, we were used to carry on projects in groups of two or four people. Therefore, we have improved our teamwork and in some cases, our leadership.