Water saving and leakage protection system

IoT project

Aleksandar Calovic	aleksandar.calovic@stud.hslu.ch
Stephan Stofer	stephan.stofer@hslu.ch
Nico Bosshard	nico.bosshard@stud.hslu.ch
Victor Kozlov	victor.kozlov@stud.hslu.ch

HSLU, Informatik Department, Switzerland

1 Abstract

The group decided to pick a project that is focused on an environmental issue, as it is getting a more and more important topic every day. Water is a finite resource and needs to be conserved for as long as possible. This is the reason why we chose to focus the project on this issue. The system contains two implementations. One for the toilet flush and the other for a bathtub. It tracks leakages in these areas and alerts the user, whenever one is occurring. As a result, we achieved to create an IoT system with 3 sensors, TCP communication and UI. The customer can be warned in time, if he has some water leakage in his house and see the sensor data in an UI chart.

2 Introduction

While dripping taps and leaky flushing systems are perceived as a nuisance and eliminated in private households, they often go unnoticed in large toilet facilities (e.g. office buildings). Leakages with permanent water loss often occur especially with very hard water or with sanitary appliances with old seals. In connection with the search for a meaningful task for the module IoT, we came up with the idea that we could develop a monitoring system for leaks in sanitary appliances. Since we are dealing with many different sanitary appliances in a household or building, the networking of the monitoring sensors also plays an important role.

The aim of this work is to implement a monitoring system on a server that graphically displays the data from the leak sensors of the various sanitary appliances and triggers an alarm in the event of a dripping tap or a leaky toilet flush.

3 Related Work

In order to find related projects, the students had to search for already existing solutions. The result were a few technologies that aim to take on similar problems:

1. According to the article “Evaluating Physical and Fiscal Water Leakage in Water Distribution System” (Bhagat, et al., 2019) there are certain methods to detect water leaks. This article also provides some mathematical knowledge for detection of water leakages and serves as a great introduction into the topic. The author also claims that there is a possibility to lessen the water leakage without implementing IOT-Systems, but to predict them with the help of artificial intelligence. With this method the pumping schedule could be optimized to solve the problem at hand.

2. The paper “Underground Pipeline Water Leakage Monitoring Based on IOT” (Badawi, 2019) describes how to build a leakage protection system based on Arduino with a flow sensor, but gives no information about the particular data exchange using TCP/IP protocol.
3. “IoT Leak Detection System for Building Hydronic Pipes” (Urbonaviciusa & Saeed, 2019) is an article that described a similar solution to the previous, but also provided the project with insight into a common TCP/IP communication for a water leakage IOT-System. It describes the communication between microcontroller and the data server, the structure of the server interface and a simple GUI design approach.

4 System Design and Implementation

4.1 System Overview

The system consists of two major components:

- An ESP32 with two piezo sensors and a humidity sensor
- A Raspberry Pi Zero serving as Backend-Server and GUI

This diagram shows all the components that are involved. It should give a rough overview of all the modules used in this project.

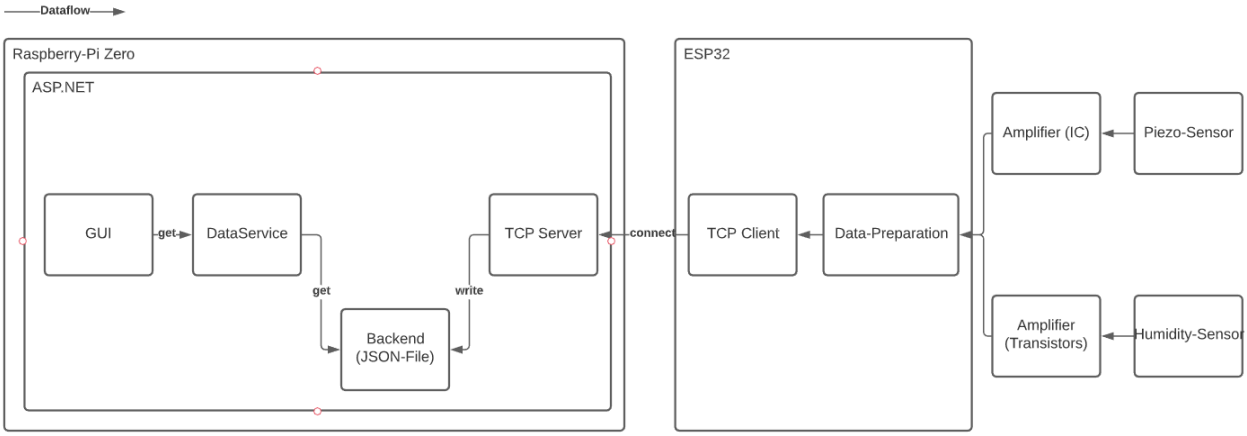


Figure 1 system overview

A small program on the ESP32 reads the data on the sensors regularly. The data gets processed to a valid JSON Object and transmitted over a TCP socket to the Backend Server.

4.2 System Architecture

Figure 1 on page 2 shows the overall system architecture.

4.3 Software Architecture Layers & Modules

The architecture consists of three layers. The main application, running on the ESP32, reads data on ADC-pins and process the measurements in a JSON object. In turn the data will be transmitted to the Raspberry Pi Zero, acting as a backend server. The data will be processed and can be viewed in a graphical user interface.

The ESP32 collects data in a single process regularly. The data is will be processed to a JSON-structure. When data must be sent, several Interfaces must be turned on. For example, the network Interface, a RTOS Event loop to handle connection issues and the WiFi module on the ESP32. After data input took from the sensors, the initialization will be done in the TCP Client, where as soon as the WiFi is available a TCP socket is created and a connection to the server is established. Following, the data can be transferred through this connection.

The receiver, a TCP Server, bind the TCP socket and read the data. The data are appended to an existing JSON which is stored server side. Before the data gets persisted, a control mechanism checks if the measurements correspond to standard values. For that the second derivative is used whether the measured values exceed a predefined threshold.

Finally, the measurements can be viewed in graphical user interface using diagrams. The latest data will be read from the JSON document stored on the webserver.

4.4 System Implementation / Functional Software Architecture

4.4.1 Circuit

There are multiple ESP32 Microcontrollers that are connected to different types of sensors using their ADC ports. Three types of sensors were used for the final system, all of which are using different signal amplification circuits.

Piezo vibration sensor to detect water flow inside a pipe:

Piezo elements are mostly used for speakers, though for this project they will be used for the opposite effect. Vibration slightly deforms the piezo element which leads to an extremely low amount of electricity being generated. This weak signal can be amplified using the OPA1612AID, which is a sound amplification integrated circuit. The integrated circuit only comes inside an SOIC-8 socket, so an adapter had to be used to convert it to a DIP-8. With the two amplification circuits on the OPA1612AID, there can be two piezo sensors used without any analog multiplexer, which should be sufficient for the system.

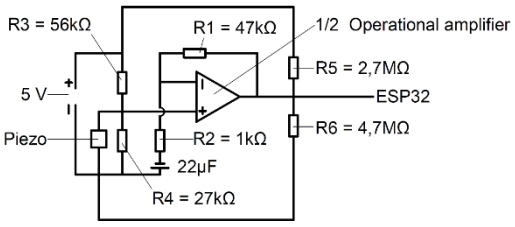


Figure 2circuit diagram

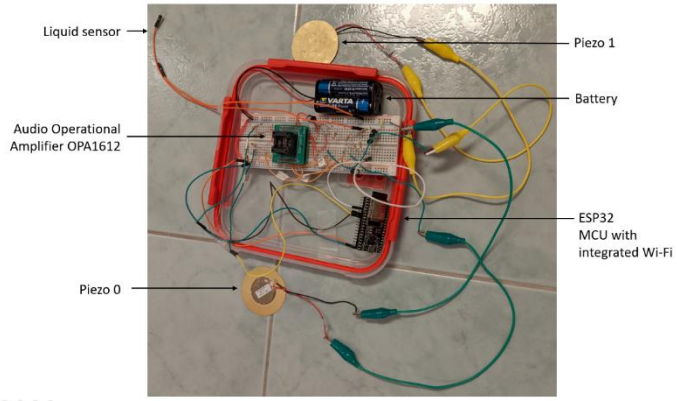


Figure 3circuit picture

Designing a circuit that is optimized to amplify the vibration caused by flowing water was very challenging. It took quite a lot of experimentation in order to come up with a great design. The final circuit consisted of a 27 to 56 kΩ voltage splitter, where the Piezo is parallel to the 27 kΩ resistor. The point between these resistors is the + input of the amplification integrated circuit. For the – input the system makes use of a positive feedback over a 47 kΩ resistor and a 1 kΩ serial to a 22 µF Capacitor going to ground. The circuit was copied twice in order to make use of every integrated amplification circuit. To supply the whole circuit with 5V for optimal performance, the output must be voltage split to not exceed the 3.3V maximum on the ADC ports of the ESP32 micro controller. Resistors of 2.7 to 4.7 MΩ were used to achieve that. The circuit was tweaked until the output signal was of sufficient quality.

Liquid sensor to detect water reaching that sensor:

In order to detect leakages having a liquid sensor is important. The sensor's internal resistor decreases whenever water reaches it and stays low until it dries up again. The resistors were quite high, so they had to be amplified as well. This was achieved by cascading three transistors.

Open wire water resistor sensor:

In some situation waiting for the water sensor to dry, after it got wet, is not an option. Instead, it should only have low resistance if there is water surrounding it, for example when the tank storing the flush water is getting filled too high. For this case, open wires must be used. Tab water has a way higher conductivity then air, so by placing two open wires, the same three transistor cascading circuits as the liquid sensor can be used in order to measure the resistor between the two open wires.

High voltage Open wire water resistor sensor:

Before we built the open wire resistor sensor with normal voltage, we used 9V containing two wire spools transformer on the secondary end and a self-interrupting circuit to create high frequency pulsating DC current to drive the transformer. That way we got a very high voltage of multiple hundred volts on the other side. With such high voltage the water resistance gets near zero and no amplification is needed.

The ESP32 measures all sensors every 5 minutes and then puts itself into deep sleep for the next 5 minutes. For water sensors the ESP32 will measure 1000 times the ADC connected to the water sensors. It will then calculate the average which it then puts inside the JSON. For piezo it will just write all 1000 measurement inside a JSON array. The JSON data is created using low level C++ operations and so there was no need to use a slow resource greedy JSON library.

The memory address to the completed JSON String is then passed to the TCP client. The TCP client enables and initializes all the networking hardware required to open a WLAN connection. After it successfully connects to the WLAN network a new TCP connection to the data collection server gets established. Once all JSON data is sent the TCP connection gets closed. Then the ESP32 disconnects from WLAN and turns of all its networking hardware to save energy.

After everything is done the ESP32 goes into deep sleep while waiting 5 minutes for the next measurement in order to save battery power.

4.4.2 ESP32 Data Flow

The *adc_reader* contains the main-method and is reading data on three different adc-channels. The data is passed to the *tcp_client* through the *send_data()* method. In there, the network interface gets started and an event loop is created. Then the *connect_to_wifi()* function is called, which establishes a connection to the given access point in the idf.py *menuconfig* setup. After that the *send_data()* method calls the *tcp_client_task()* where a socket is created, and the data transmits to the server. If the data has been transmitted, the socket shut down and gets closed. Before returning to the *adc_reader* the WiFi will be disconnected and the event loop gets deleted.

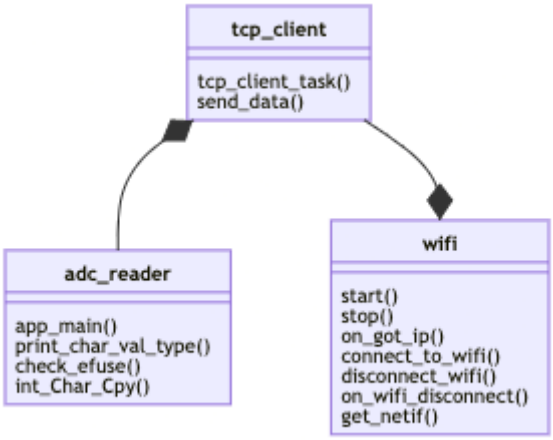


Figure 4 tcp client class diagram

4.4.3 ESP32 Power management

Because our ESP32 is battery powered saving energy is essential for a user friendly IoT system. In order to achieve this, we put the ESP32 into deep sleep during the 5-minute break between two measurements. During deep sleep the ESP32's power consumption goes from about 260 mA to just 10 µA. During deep sleep only the ULP Coprocessor and the RTC is enabled. The reason we need the RTC is in order wake up the ESP32 after 5 minutes again. Booth the CPU and most of the RAM and all peripherals are completely powered of during deep sleep. In addition to deep sleep, we also disabled

the quite power greedy brown out detection as we don't care about being vulnerable against hardware glitching attacks as there is no sensible data stored in our system.

4.4.4 ASP.NET

The UI Application with using the ASP:NET Core technology. The UI backend contains the TCP Listener and consist of 4 Layers: view, business logic, model and data access.

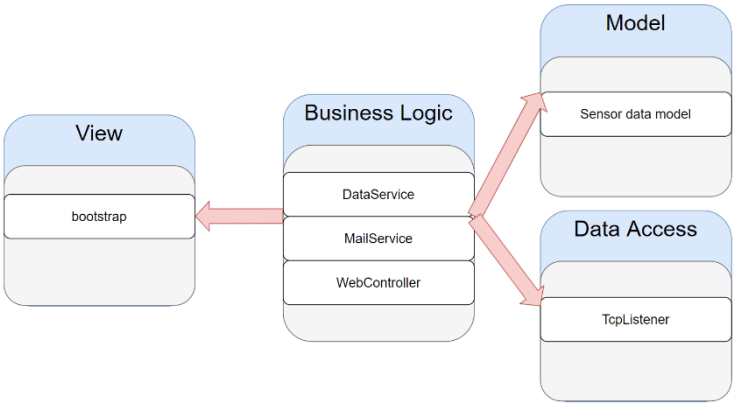


Figure 5 Frontend layer overview

The TCP Listener gets and saves the data sent from ESP32 Controller in a json file. In the business logic we have two services. One of the is the MailService, which sends the alert email to the customer and the DataSerevcie, which reads and parses the sensor data. The view renders six charts with the sensor data.

The web application consists of 3 types of charts. The long-time chart to see around the past month how the situation changed in long term. The sort time chart that shows around the last day and finally the chart that shows the second derivation of the piezo measurements.

Why the second derivation of the piezo signal? The signal we care about is modulated on top of a sinus wave caused by the circuit's oscillation, so we must use the second derivation in order to get any useful data.

The web panel can be visited using any web browser. This makes it fully cross platform compatible.

If the server detects any kind of abnormality the user will be notified using a fully automated alert email.

5 Evaluation/Experiments/Results/Discussion

5.1 Evaluation

5.1.1 Sensors

In a first step, we evaluated our use cases. We need to be able to measure water flow inside a pipe from outside in order to not damage the pipe. We need to be able to measure the level of water present in a flush thank in order to detect water levels above normal or even overflows. We also need to be able to measure small dops of water leaking to the drain. After thinking about it we came up with the following candidates of sensors: Piezos for measuring the vibration of flowing water inside a pipe from outside, high voltage resistance sensors to detect the present of water and a humidity sensor to detect small amounts of water.

First, we evaluated the Piezos. After trying out all different types of Piezos we figured out that the large cheremic ones without a casing resulted in the best measurements.

After trying out the high voltage resistance sensor we decided against using it as it required a lot of energy and it was inconvenient to measure ampere using and ADC input as voltage reduction over a resistor would have to be measured instead. It turned out that just using three cascaded transistor amplified open wire is more than enough to detect the presence of water, which is easier to build, uses less energy and is easier to measure.

The humidity sensor was perfect and can be amplified using the same 3 cascaded transistor circuit as used for the open wire water detector. Unfortunately, we only had one of those and they no longer seem to by obtainable.

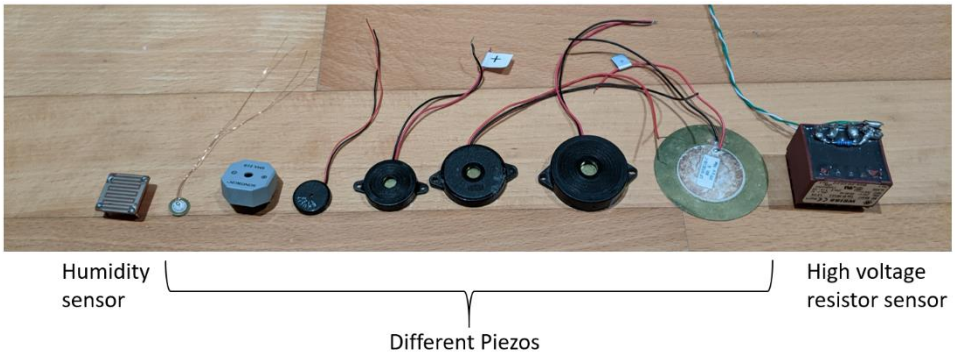


Figure 6 sensors picture

5.1.2 Circuit

There are multiple components that vastly change how the Piezo amplification circuit works. For the integrated circuit we first tried a common LM358. It turned out that the output signal isn't from sufficient quality. After comparing all sound amplification integrated circuits, we found OPA1612AID. It has a noise of only 1.1nV and a total harmonic distortion of 0.000015%, which is something no other integrated circuit can offer. The OPA1612AID was necessary for this project to get the most precise measurements possible. From the amplification circuit itself the amplification ratio using the positive feedback loop resistor value was the part that required the most tuning as the output signal had to fluctuating in a certain range. We also had to tune the voltage divider ratio of the input, the and the capacitor in order to get a good signal.

5.1.3 Piezo result interpretation

After investigating the amplified piezo signal using an Oscilloscope (PicoScope 2204A) we concluded that we have a sinus wave on top of which the actual signal is modulated. In order to extract the desired signal, we decided to calculate the second derivation of the signal.

5.2 Results

After the correct calibration of the measurements, it's well recognizable when a disruption occurs. If the duration of such a higher amplitude is longer than usual then the problem of the permanent leakage is detected. In the chart below we tested a flush with a tank that filled too much and made the liquid sensor wet, which caused the higher amplitude. The overflowing water took the emergency drain and flowed down the toilet. The same results also occur on water dropping from a tap on a humidity sensor. Despite it being a different sensor, its circuit is identical and produces the same results but detects dropping water. If there is a permanent waterflow (due to a untight flush) the piezo sensor will detect the permanent noise and vibration of the flowing water and also cause a high amplitude.

Toilet flush:

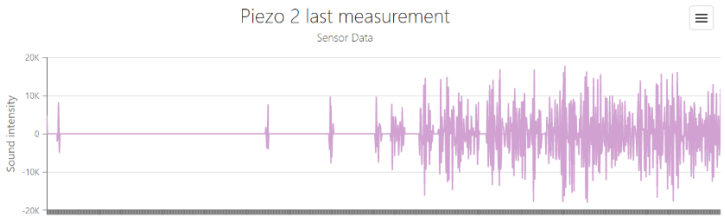


Figure 7 Piezo, toilet flush

Toilet overflow sensor:

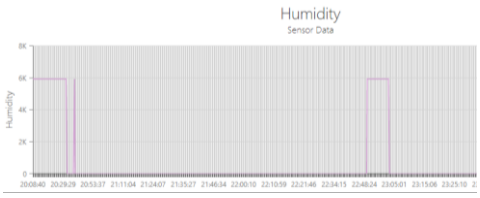
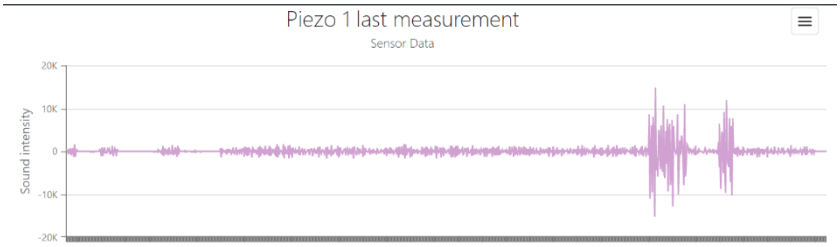


Figure 8: Humidity sensor, overflow

Leaking Tap:



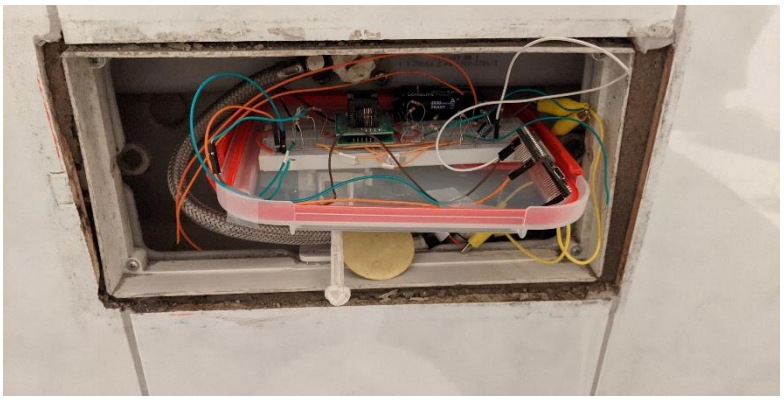


Figure 9 circuit toilet flush

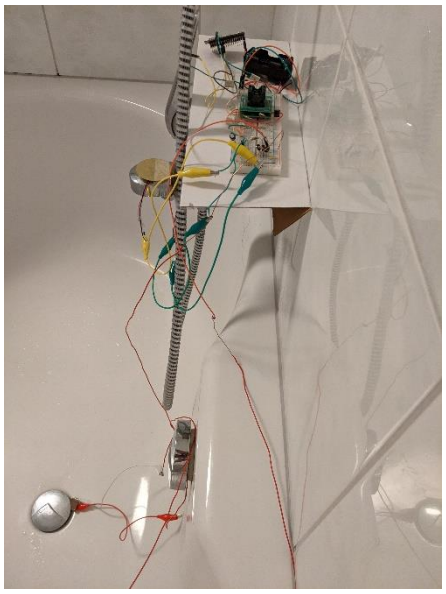


Figure 10 circuit bathtub

Key	Description	Value	Alarm value
water_0	Indicates if the humidity sensor is wet	Integer	4000
piezo_0	Indicates if there is an unnecessary water flow	Integer Array	1000
time	The time, on the second, when the TCP Listener received the data from the ESP32.	23:43:46 Format: {hh:mm:ss}	-
date	The date when the TCP Listener received the data from the ESP32.	16.12.2020 Format: {dd:mm:yyyy}	-

6 Application(s)

The application can detect water leakages inside of a water flush tank. It can also detect dripping water taps. All this information is displayed in a web application created in ASP.NET, which shows precise charts of the leakage issues. Monitoring of heating and air-condition is already common in facility management. Our invention is an additional tool in order to save water and money. In a big facility, such as schools and office buildings, this monitoring can reduce the wasted water to a huge amount. In the application the data is displayed as follows:

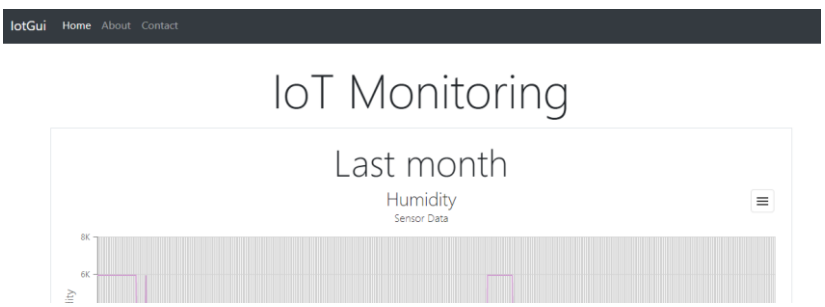


Figure 7 IoT GUI

7 Conclusion

With the ever so slightly decreasing resource of water, the students took on the challenge to find a solution for wasted water in leakages. The application can detect leaks in certain areas and brings the world closer to solving this massive issue. By creating fully functional circuits to measure the needed data, it is possible to create a system of sensors and use them in big facilities. The state-of-the-art graphical user interface displays all the necessary data in an easy-to-understand way.

One of the hardest decisions we had to take during the project was to completely revamp our Front-End as we came to a bottleneck, where we didn't see a clear solution. We moved our project away from an angular implementation and decided to use ASP.NET instead. Ultimately, we still managed to pull through and achieved in our eyes a satisfying result.

Nevertheless, there are certain parts that need improvement and would have been implemented if the time was available. The Front-End could have a little bit more settings for the charts implemented. Also, the backend could have been changed from a JSON-File to a proper database. The TCP Connection lacks in error handling and verification of data, which is mostly sufficient for this project, but could be considered for further development.

8 Contributions / Acknowledgments

Nico focused mainly on planning and building the electronical circuits to amplify the signal of the sensors and wrote the code to measure the sensors and convert the data into a JSON string. As he was the most experienced in IoT matters, he also supported the members throughout their work. Stephan implemented the entire ESP32 application, which include the ADC-reader to access the pins, the TCP-Client that fragments the data into TCP packets and established a WLAN connection for the ESP32. He also created a first draft of the final presentation. Aleksandar and Victor worked together on the Front-End implementation, starting with angular and revamping the project in ASP.NET. They also shared the workload for creating a fully functional TCP Server in python and transferred the logic over to ASP.NET. Victor was also responsible for the project management and scheduling our meetings. Aleksandar and Victor focused on writing the general part of the final report as well.

9 Major Milestones & Deliverables

Project management is an essential part of our work. In this project we defined the roles of each team member, created work packages and assigned all of the to the team members. So we defined the milestones and created the time schedule.

9.1 Team roles

TEAM	Name	Role
TEAM AVSN	Nico	Electronic Engineer, Software Engineer
	Stephan	Software Engineer
	Aleksandar	Software Engineer
	Victor	Software Engineer, Project Manager

9.2 Project work packages

Following packages were defined for our project:

Work package number	Description
WP 1.1	Electrical circuit for piezo sensor
WP 1.2	Electrical circuit for humidity sensor
WP 1.3	Bind esp32 with the sensors

WP 1.4	Implement and configure wifi connection
WP 1.5	Create a ADC-Reader and TCP Client
WP 1.6	Create a server (TCP/IP receiver)
WP 1.7	Create Server interface
WP 1.8	Create GUI for monitoring/Alarm tool
WP 1.9	Testing
WP2.1	Set up project management
WP2.2	Initiate and split documentation
WP2.3	Set up team communication environment

As next we assigned the work packages to the team members:

Team member	Project work package
Nico	WP1.1 WP1.2 WP1.3 WP1.6 WP1.7 WP1.9
Stephan	WP1.4 WP1.5 WP1.7 WP1.9
Aleksandar	WP1.6 WP1.7 WP1.8 WP1.9
Victor	WP1.7 WP1.9 WP2.1 WP2.2 WP2.3

9.3 Project Planning

To visualize our planning, we have created a rough schedule overview of our project:

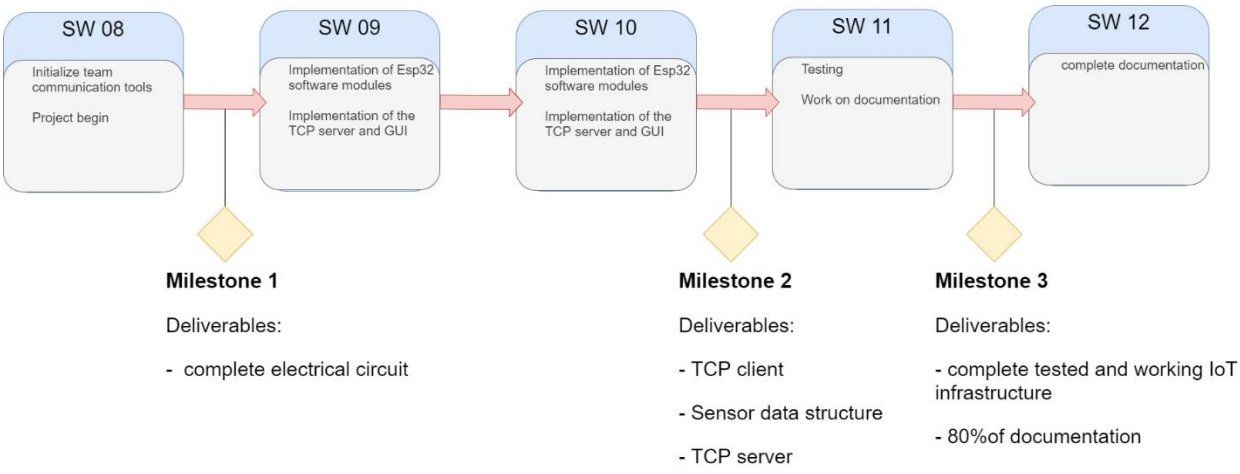


Figure 8 project management schedule

9.4 Milestones & Deliverables

M1: Electrical circuit

M2: Data collector

M3: Data representation

9.5 Time schedule

A time schedule was created in order to schedule the work packages for all team members.

IoT

ID	Tasks	done	V	[SW] date	8	9	10	11	12
					Schedule				
P1	Initialisierungsphase								
WP 1.1	Electrical circuit for piezo sensor		NB	SW09					
WP 1.2	Electrical circuit for humidity sensor		NB	SW09					
WP 1.3	Bind esp32 with the sensors		NB	SW10					
WP 1.4	Implement and configure wifi connection		SS	SW12					
M1	Milestone 1		all			⌂			
WP 1.5	Create a TCP Client		SS	SW12					
WP 1.6	Create a server (TCP/IP receiver)		VK, AC	SW11					
WP 1.7	Create Server interface		VK, AC	SW12					
WP 1.8	Create GUI for monitoring/Alarm tool		VK, AC	SW12					
M2	Milestone 2						⌂		
WP 1.9	Testing		all	SW12					
WP2.1	Set up project management		VK	SW11					
WP2.2	Initiate and split documentation		VK	SW12					
WP2.3	Set up team communication environment		VK	SW09					
M3	Milestone 3								⌂

Legend	
ID	Ident-Nummer
done	if work package is done
V	Responsible
VK	Victor Kozlov
AC	Aleksandar Calovic
NB	Nico Bosshard
SS	Stephan Stofer
alle	alle
⌂	Milestone
	Duration
	Done

Figure 9 Time schedule

10 References / Biography

Badawi, W. A. (2019, 09 25). *ijmsr.org.* (I. J. Research, Ed.) Retrieved 11 19, 2019, from ijmsr.org: http://www.ijmsr.org/ojs/index.php/ijmsr/article/view/206/187

Bhagat, S. K., Tiyasha, Welde, W., Tesfaye, O., Tung, T. M., Al-Ansari, N., . . . Yaseen, Z. M. (2019, 10 08). Evaluating Physical and Fiscal Water Leakage in Water Distribution System. Ho Chi Minh City, Vietnam. doi:10.3390/w11102091

Urbonaviciusa, A., & Saeed, N. (2019 , 09 08). *repository.uwl.ac.uk.* Retrieved from repository.uwl.ac.uk: http://repository.uwl.ac.uk/id/eprint/6371/1/Saeed_and_Urbonaviciusa_IJEM_2019_IoT_leak_detection_system_for_building_hydronic_pipes.pdf

Vinoj1, J., & Gavaskar2, D. S. (2018, 08). *academia.edu.* (E. a. International Journal of Scientific Research in Computer Science, Ed.) Retrieved 11 19, 2020, from academia.edu: http://ijsrcseit.com/paper/CSEIT1836156.pdf

11 List of Figures