

UNIVERSIDAD POLITÉCNICA DE MADRID

ETSI y Sistemas de Telecomunicación y

ETSI y Sistemas Informáticos

Máster Universitario en Internet of Things



MASTER'S THESIS

“BIOMETRIC WORK ATTENDANCE MANAGEMENT AND LOGGING
WITH A BLOCKCHAIN SYSTEM”

JEIDI PADRON NÚÑEZ

JULY 2021



POLITÉCNICA



Máster Universitario en Internet of Things

Master in Internet of Things

Master's Thesis		
Title	Biometric Work Attendance Management and Logging With A Blockchain System	
Author	Jeidi Padron Nuñez	Signature
Tutor	Sergio Arevalo Viñuales	Signature
Co-Tutor	Maria Isabel Muñoz Fernández	Signature
Director		Signature
Board of examiners		
Presidente/ President		
Secretario/ Secretary		
Vocal		
Date		
Grade		

Secretary

Acknowledgments

This project would not have been possible without the support of many people. Many thanks to my daughter Sabrina and my husband Pedro for encouraging me to follow my dreams. With their support and love, together we share this journey.

A special thanks to Dr. Sergio Arevalo Viñuales and Dr. Maria Isabel Muñoz Fernandez, my tutoring professors. Their lessons at Distributed System influenced me to continue investigating the uses of distributed technologies in the IoT world. They offered support, guidance, and dedication during the project development.

To all professors and students at UPM MIoT class, who have been my support during all lessons, projects, exercises, and presentations, I will always appreciate all we have done.

Measure what can be measured and make measurable what cannot be measured.

Galileo Galilei

Table of content

Table of content	i
List of figures	iv
List of tables	v
Summary	vii
1 Introduction	1
2 State of the art	3
2.1. Legal considerations	4
2.2. IoT Platform	5
2.2.1. Biometric Sensor	5
2.2.2. OLED Display module	7
2.2.3. NodeMCU Development kit	8
2.2.4. IoT Edge computing	9
2.3. Decentralized Application	11
2.3.1. Ethereum and blockchain	12
2.3.2. Ethereum accounts	14
2.3.3. Smart contract	14
2.3.4. IPFS	15
3 Design and Configuration	19
3.1. Architecture design	21
3.2. Use case	21
3.3. IoT device configuration	22
3.3.1. Hardware connection	22
3.3.2. Configure NodeMCU Lolin V3	23
3.3.3. Configure OLED display.	24
3.3.4. JM-101 Fingerprint Scanner Sensor	25
3.3.5. Configure IoT Device	27
3.4. Edge node configuration (centralized)	29

3.5.	Smart contract code and deploy	32
3.5.1.	Install MetaMask	32
3.5.2.	Get funds to deploy Smart Contract	34
3.5.3.	Create and deploy Smart contract	35
3.6.	Decentralized application configuration	39
3.6.1.	HTML file structure	40
3.6.2.	CSS file structure	43
3.6.3.	JS file structure	44
4	Deployment and Test Results	48
4.1.	Connect IoT device to network	49
4.2.	Open IoT Edge Web page	49
4.3.	Connect DApp to Blockchain	54
5	Budget	57
5.1.	bugest and time schedule.	58
6	Conclusions	61
6.1.	General conclusions.	62
6.2.	Future works	62
7	References	65
7.1.	References	66
7.2.	Repositories	69
8	ANNEX	71
	Annex A - Glossary	i
	Annex B - IoT Device Arduino Code	i
	Annex C - Change Bitmap to array	i
	Annex D - Smart contract - timinglog.sol	i
	Annex E - How to deploy a Smart contract	i
	Annex F: DApp: index.html	i
	Annex G: DApp - metamask.css	i
	Annex H: DApp - contract.js	i

List of figures

Figure 1.	Attendance system using biometric sensor	5
Figure 2.	Fingerprint sensor model JM-101	6
Figure 3.	AZ Delivery 0.96 inch I2C OLED Display	7
Figure 4.	Development kit NodeMCU general scheme	9
Figure 5.	IoT Edge development	10
Figure 6.	Decentralized application interfaces	12
Figure 7.	Blockchain of three blocks	12
Figure 8.	HTTP and IPFS approach	17
Figure 9.	System architecture	21
Figure 10.	Use Case diagram	22
Figure 11.	Circuit Diagram for IoT Fingerprint Attendance System	23
Figure 12.	SFG windows program to test fingerprint sensor	25
Figure 13.	IoT Edge Sequence Diagram	29
Figure 14.	Import database to server	32
Figure 15.	Installation of MetaMask	33
Figure 16.	Request ETH from faucet	35
Figure 17.	Dapp "Employee Attendance Management"	40
Figure 18.	Turn on IoT Device and connect to network	49
Figure 19.	run Apache and MySQL server on XAMPP program	50
Figure 20.	Device working mode	50
Figure 21.	List of all employees in a company, with	51
Figure 22.	New fingerprint added	52
Figure 23.	Export daily attendance report to excel	53

List of tables

Table. I.	Development kit comparation.	8
Table. II.	HTTP and IPFS comparation	16
Table. III.	NodeMCU pinout connection.	23
Table. IV.	DApp variables name and function	41

Summary

The maximum legal working hours in Spain are generally 9 hours per day (maximum 40 hours per week), with a maximum of 80 overtime hours per year. Since May of 2019, the Law 08/2019 requires companies to keep a daily record of the working hours, including the exact hour of entry and exit of each worker, regardless of whether overtime was performed or not. This register should be available and accessible for consultation to avoid possible manipulations.

The project objective is to develop an Internet of Things end-to-end prototype for generating, collecting, storing, and getting reports about employee attendance time. This biometric attendance system uses NodeMCU Arduino board with fingerprint sensor, for generate data and send instant collected information to Edge node using a Wi-Fi wireless communication. At the edge node, the data is stored inside traditional SQL database. The daily report is generated using a traditional web page.

Finally, to keep the daily report immutable, and auditable it was developed a decentralized application executed on web browser that calls a storage peer-to-peer InterPlanetary File System (IPFS), and smart contracts from Ethereum blockchain.

The solution has been tested and confirm there is not possible to modify the daily record thanks the blockchain properties.

This project has a wide application in school, college, business organization, offices, etc., where marking of attendance is required accurately with time. Thus, by using the fingerprint sensor, the system will become more secure for the users.

Keywords: IoT, NodeMCU, blockchain, Ethereum, Decentralized Applications, Smart Contract, IPFS, Infura, Remix.

1

Introduction

The regulation in Spain dated March 8th, 2019, (Royal Decree Law 8/2019) [1] specified it is mandatory for all companies to keep a "daily record contains the specific start and end time of work for every employee, during four (4) years and accessible for the employees, legal representative and work inspector". The aim of this regulation is to end with a lot of extra working time, unjustified employee absence, and fraud labor contract [2].

In this Thesis project, the innovation consists to develop an IoT smart tool to generate a daily attendance report, avoid data modification, keeping this data history immutable, auditable and available for specific people. This project used quantitative experimental methods, to collect and analyze the data stored:

- Build a biometric attendance system prototype, using IoT device with a fingerprint sensor, to generate the data [3]. Each time any employee enters or exits to office or factory, they must "sign the assistance", that means, the system validates the data using ID biometric authentication. If the data is valid, then it will be stored their clock-in and clock-out time.
- Send this data using Wi-Fi connection to a web service that use traditional database acting as edge computing. At the end of every day, it can be generated a single report containing all employees working time.
- Develop Decentralized Applications (DApps) to store the daily report inside distributed nodes Inter Planetary File System (IPFS) [4] using Ethereum network. The file hash will be stored in a Blockchain system [5]. The storing and recovery of hash file will be done using a blockchain smart contracts [6].
- Decentralized Applications (or DApps) are applications that interacts directly with a blockchain with a nodes with full replication on every peer in an untrusted peer-to-peer network. Blockchain transactions are sent to and processed by "random" nodes via Proof of Work or Proof of Stake.

This document includes State of the art that contain all theoretical concepts about smart contracts, blockchain and IoT device characteristics. Design and configuration section include network topology, interfaces, communications protocols, use cases diagram and brief explanation for coding programmed. Deployment and test part showed the results obtained. Finally, the document ends with budget and conclusion.

2 State of the art

2.1. Legal considerations

The main objective of this project is to keep track of employees' working hours by daily recording their clock-in and clock-out time. These records are traditionally used by companies to know how long have been working an employee at the company.

The attendance system will benefit both worker and company, because its offer a legal report to have justification for any legal issue regarding absenteeism and overtime. Spain Royal Decree-Law 8/2019, of 8 March specified it is mandatory to have attendance record with the following legal considerations [1]:

- The company must guarantee "the daily record of working time", including of the specific start and end times of each employee.
- Such record will be organized and documented through collective consultation or company agreement.
- The records must be kept by the company during four (4) years.
- The records must be available to workers, the statutory body of worker representatives and the Labor and Social Security Inspectorate.
- Non-compliance with the obligation to record working time is classified as a serious infringement, with penalties ranging from 626 euros to 6,250 euros.
- It is mandatory since May 12th, 2019.

Therefore, we can extract the following insights:

- The registry must be daily. It is not valid a weekly or monthly report
- The registry must include the start time and end time and it is not necessary to record work breaks. For example, if the employee has time schedule from 9:00 to 14:00 and from 16:00 to 19:00 it must record the entrance at 09:00 and exit at 19:00
- If a company do not register the breaks, it will assume that the record is only for effective work.
- The law does not specify how should be done the registry (paper, excel data), so in principle, it will be valid any system that was reliable, objective and that cannot be modified or altered by the company

2.2. IoT Platform

2.2.1. Biometric Sensor

Attendance systems are used to track when employees start and stop work. It enables an employer to monitor their employees working hours, late arrivals, early departures, time taken on breaks, overtime and absenteeism.



Figure 1. Attendance system using biometric sensor

There are five main different technologies used to keep track of attendance:

- Manual System: This system is still used in many companies, record the entrance and exit of employee in a sheet of paper, and later use an excel sheet to manually count the working time. Disadvantage: data can be modified, damage or lost.
- Mechanical clock: the first employee time clock was invented in 1888. Its purpose was to record the time an employee entered and left the factory. The employee put a paper card and it were stamp the time. This clock machines were used for about 100 years. Disadvantage: employees can register on behalf another employee and company need to count manually the working hours.
- Automated systems using RFID card: In the mid 1990's the first UHF reader was invented. The employee receives a plastic card, that contains their information like department, section, name and ID number. The employee just put or swipe the card close to reader, the system automatically registers the attendance and saves the data in the system. Disadvantage: employees can register on behalf another employee using another card.

- Automated systems using Biometric characteristics: The system uses biometric data in the form of the iris, voice, fingerprint, or face recognition. Disadvantage: high cost and false positive.
- Mobile phone Application systems: the latest technology use a mobile APP, based on Android or iOs operating system allowing to register the time for employee working at home or any remote location. In 2020, about 77% of smart phones include biometric capacitive fingerprint reader [7]. The majority of facial or voice recognition processes performed on smartphones are software-based rather than hardware-based. Disadvantage: if someone take the mobile phone, it can delete the biometric identification and replace for new data.

In this project it was selected the biometric attendance system, using a very cheap sensor that can be connected to any IoT device, compatible with Adafruit Arduino library [8]. It has been selected the JM-101 optical fingerprint module [9].

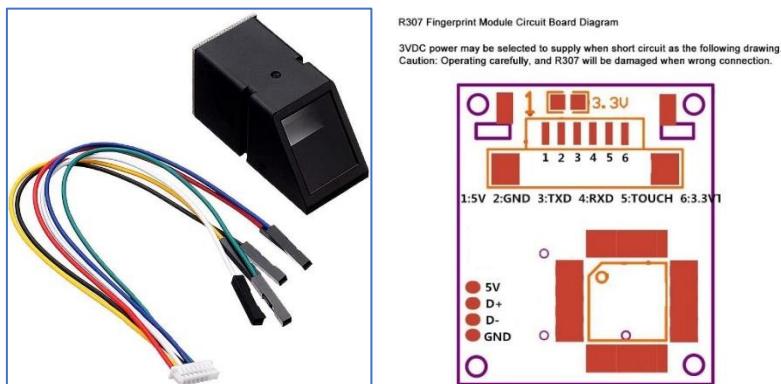


Figure 2. Fingerprint sensor model JM-101

The Fingerprint module can be directly interfaced with any microcontroller as well as Arduino Board. This optical biometric fingerprint reader with great features and can be embedded into a variety of end products like access control system, attendance system, safety deposit box, car door locking system.

Features:

1. Integrated image collecting and algorithm chip together, with functions such as fingerprint input, image processing, feature extraction, template generation, template storage, fingerprint comparison (1:1) and fingerprint search (1:N)
2. Low power consumption, low cost, small size, excellent performance

3. Professional optical technology, precise module manufacturing techniques
4. Good image processing capabilities can successfully capture image up to resolution 500 dpi
5. Provide TTL UART interface for direct connections to microcontroller UART or to PC through MAX232 / USB-Serial adapter

For detailed information, please read JM-101B User Manual [10]

2.2.2. OLED Display module

The project includes also a Organic Light Emitting Diode (OLED) display module, used to visualize the system status, like if Wi-Fi signal is available, or the name of employee that is enter or exit. OLED is a self light-emitting technology composed of a thin, multi-layered organic film placed between an anode and cathode. In contrast to LCD technology, OLED does not require a backlight.

For this project, it is used 0.96 inch blue OLED display module from AZ Delivery. The package includes display board, display, and 4 pin male header pre-soldered to board.

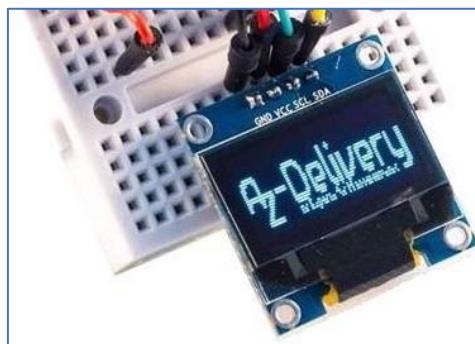


Figure 3. AZ Delivery 0.96 inch I2C OLED Display

Features:

1. Chip set: SSD1306, Arduino compatible
2. Resolution: 128 x 64 pixels
3. Operating voltage: 3.3V to 5V . 4. Low power consumption: 0,04W
4. Pin coating: VCC, GND, SCL, SDA. Control: I2C / TWI

For detailed information please read Quick Start Guide OLED Display [11].

2.2.3. NodeMCU Development kit

Development kit is an open-source electronics platform based on easy-to-use hardware and software that helps to prototype IoT projects. There are many developments kit available in the market. The following table show a comparation for the most common development kit:

Type	B-L072Z-LRWAN1	Raspberry Pi Pico	Arduino nano 33 IoT	Arduino nano	NodeMCU V3
CPU	ARM Cortex-M0+	Dual-core RP2040	ARM Cortex-M0+ SAMD21	Single core ATmega 328	ESP8266
Clock	@ 32 MHz	@ 133 MHz	@ 48MHz	@ 16MHz	@ 80MHz
Flash memory	192 Kbytes	2 MB	32 Kb	1 K	4 MB (32 MBit)
Data memory	20 KBytes	264 KB	256Kb	32 Kb	64 KB
number of pins	52 pins Lora 32 GPIO	26 GPIO	22 GPIO	22 GPIO	16 GPIO
USB port	yes	yes	yes	yes	yes
wireless communication	LoRa WAN	Bluetooth	Bluetooth, Wi-Fi	none	Wi-Fi
Ultra Low power	yes	yes	yes	yes	yes
cryptography	no	no	ATECC608A (TLS)	no	SHA256; AES 128 and 256
Power output	3,3V and 5V	3,3, v	3,3 V, 5V if solder	3,3V and 5V	3,3 V
Price	45 €	7 €	20 €	20 €	15 €
Embedded sensors	no	no	Accelerometer and gyroscope	no	no
Program IDLE	Keil (C++)	micro Python	Arduino	Arduino	Arduino / LUA
Reset button	yes	external	yes	yes	yes

Table. I. Development kit comparation.

After analyzing previous table, it was decided to use a development kit with Wi-Fi as wireless communication: Arduino nano 33 or NodeMCU. Arduino nano 33 IoT is a new 2020 product, with few references of uses. Then, it was selected NodeMCU development kit [12], which is stable product, low price, low power consumption, many libraries available (Arduino), USB port available and the use of Wi-Fi as communication interface.

NodeMCU is a development kit open source based on ESP8266, integrates GPIO, PWM, IIC, 1-Wire and ADC assembled all in one piece.

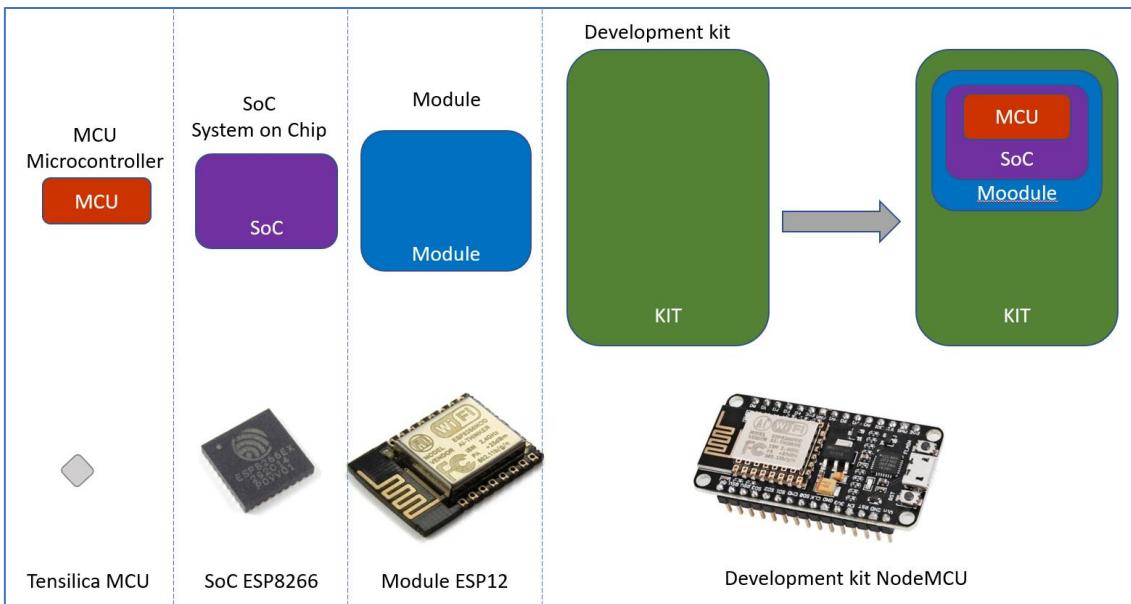


Figure 4. Development kit NodeMCU general scheme

Features:

1. Micro Controller Unit (MCU) Tensilica L106 32 bits.
2. System on Chip (SoC): ESP8266 is a highly integrated chip. It offers a complete and self-contained Wi-Fi networking solution. Include: MCU, WiFi 2,4 GHz.
3. Module ESP12: this module includes the access to pins plus flash memory.
4. Development kit: NodeMCU allow integration with MCU, SoC and Module, with USB port, power pins and LEDs board

NodeMCU, has powerful on-board processing and storage capabilities. It allows to be integrated with sensors and other specific applied devices through its GPIOs add more components to build IoT prototype.

2.2.4. IoT Edge computing

IoT devices typically have limited data processing and storage capabilities, then substantial data processing needs to occur outside the device. Edge computing allows IoT deployments to perform data processing closer to the end device. This results in lower latency and improved efficiencies in data transport. The location of the edge has various possibilities and will differ according to the use case. As this project is just a Proof-of-concept, the Edge computing is done inside a laptop with the function of data aggregation, employee validation and generate one daily report

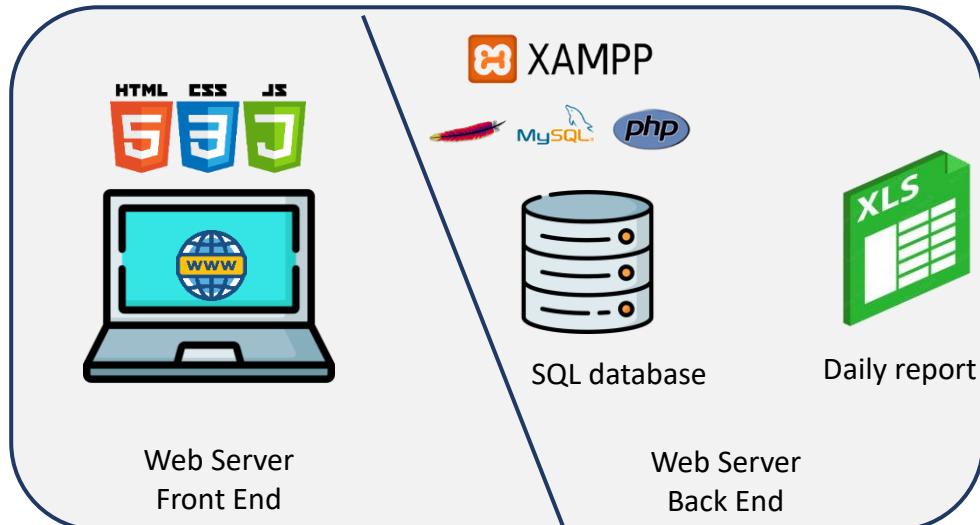


Figure 5. IoT Edge development

The previous figure showed a schema of IoT Edge node, as a web server. Note: icons were available at Flaticon web page [13]

IoT Edge Features:

1. Real time data processing
2. Data visualization
3. Basic Analytic
4. Employee authentication
5. Obtain Daily report with time attendance for all employees.

The front end is made up of many different languages and libraries:

- Cascading Style Sheets (CSS) allows to apply styles to web pages and enables to do this independent at any view of each web page.
- Hypertext Markup Language (HTML) give the structure of the website. It can be used to integrate formatting attributes and layout specifications into the content of these Web pages.
- JavaScript is a runtime language for web browsers. The JavaScript will loaded and can perform actions and make decisions.

Backend is the server-side of the website. It is the portion of software that does not come in direct contact with the users. It stores and arranges data:

- Cross-platform, Apache, MySQL, PHP and Perl, (XAMPP) is an open-source package Apache distribution used for to build a local web server.

- Hypertext Preprocessor (PHP) is a scripting language especially suited to web development. It is considered a backend scripting language.
- Node.js is an open-source and cross-platform runtime environment for executing JavaScript code outside a browser. NodeJS is not a framework, and it's not a programming language. We often use Node.js for building back-end services like Web App
- Database is a collection of data, in our project, the information about employees time logs. Using Structured Query Language (SQL) lets the user access the databases and generate a report in excel format (table with rows and columns).

2.3. Decentralized Application

According Ethereum organization a decentralized application (short for DApp) "is an application built on a decentralized network that combines a smart contract and a frontend user interface." [14]. DApp are also known as serverless web3. The main characteristics of DApp are:

- **Decentralized** means they are independent, and no one can control them as a group.
- **Deterministic** means they perform the same function irrespective of the environment they are executed.
- **Turing complete**, which means given the required resources, the DApp can perform any action.
- **Isolated**, which means they executed in a virtual environment known as Ethereum Virtual Machine so that if the smart contract happens to have a bug, it won't hamper the normal functioning of the blockchain network.

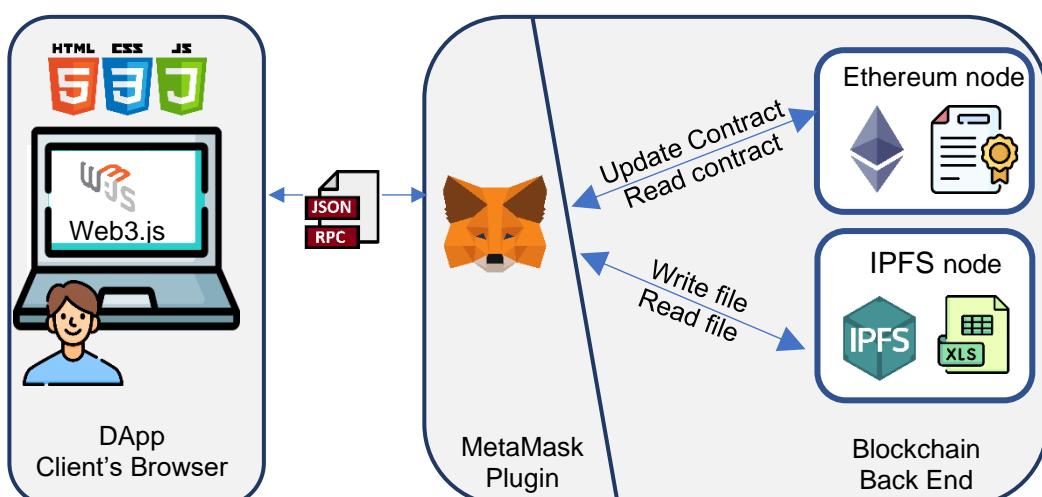


Figure 6. Decentralized application interfaces

MetaMask is a web extension, which allows to manage Ethereum private keys via web browser. By doing so, it serves as a wallet for Ether and ERC20 (Ethereum Requests for Comments) tokens and allows to run Ethereum Decentralized Apps (DApps) right in your browser without running a full Ethereum node [20]. That means the execution of smart contracts if they are deployed in the active account of the wallet in use.

Once DApps are deployed on the Ethereum network nobody can't change them. DApps can be decentralized because they are controlled by the logic written into the contract, not an individual or a company. DApp code can't be taken down. And anyone can use the DApp's features. Even if the team behind the DApp disbanded you could still use it. Once on Ethereum, it stays there.

2.3.1. Ethereum and blockchain

A blockchain is a chain of blocks that contains information. Each block contains some data, the hash of the block itself, and the hash of the previous block.

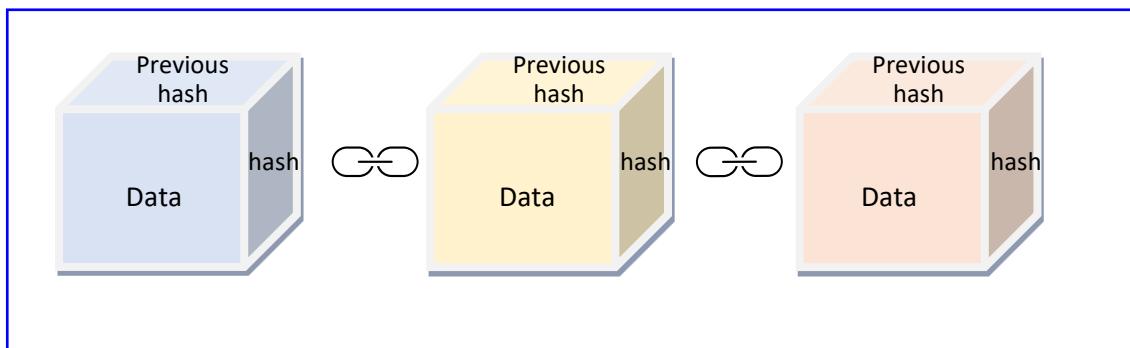


Figure 7. Blockchain of three blocks

The *hash* is a mathematical algorithm used to map data (of arbitrary size) to data of fixed size. Ethereum uses the Keccak-256 cryptographic hash function, where always is returned a value of 256 bits. It is a unidirectional function, since given a data is possible to calculate the hash, but given a hash it is impossible to get the data value.

Ethereum is an open source decentralized infrastructure that executes smart contracts. It uses a blockchain to synchronize and store the system's state changes.

Ropsten is a test blockchain where Ether doesn't have any real monetary value. It's a free blockchain used for development. It is needed it because developers don't want to spend real money when testing and designing any DApp. [15]

"**Miners**" (network nodes) solve a complex math algorithm to add a new block, finally obtaining the *hash* creating a "proof of work". This job take time, consume computer resources and energy. Since each Ethereum transaction requires computational resources to execute, each transaction requires a fee. The miner who performs the calculation and solve the puzzle first obtain economical reward in a virtual coin named Ether (ETH) with the lowest denomination named Wei ($1 \text{ ETH} = 1 \times 10^{18} \text{ wei}$).

Gas (Ethereum) refers to the unit that measures the amount of computational effort required to execute specific operations on the Ethereum network. There are two parts that determine the cost of your transaction: gas limit and gas price.

- **Gas limit** is the maximum amount of gas the originator is willing to buy for this transaction. The Gas limit can not be zero as every transaction need a gas. If gas is insufficient, the transaction will not finish, and there is not refund
- **Gas price** is the amount (in wei) that the originator is willing to pay for each unit of gas. The minimum value that gasPrice can be zero, which means a fee-free transaction.

To calculate how much will cost a transaction, create a new block, or execute a contract, just multiply the amount of gas limit by the gas price:

$$\text{total cost} = \text{gas limit} \times \text{gas price}$$

$$\begin{aligned}\text{Example} &= 21.000 \text{ gas} \times 140 \text{ GWei/u} \\ &= 2,1 \times 10^4 \text{ gas} \times 1,4 \times 10^{11} \text{ wei/u} = 2,94 \times 10^{15} \text{ wei} \\ &= 0,00294 \times 10^{18} \text{ wei} = 0,00294 \text{ ETH}\end{aligned}$$

1 ether cost 1.593,41 euros [16] (March 2021), the transaction will cost 4,68 €. Read transactions do not modify the state, that means, it does not cost any gas.

Blockchain is intrinsically a very secure technology: if data changed, it is also need to modify all blocks. To modify a block take times (as July 2021, between 10 and 20 seconds) [17], the hacker need to paid to miners, the blockchain is replicated on several nodes and reverse engineering to know the data from hash is impossible. If one block is

changed by someone, then the blockchain will alerting the network to the fact that one block is fake.

2.3.2. Ethereum accounts

To access from real world to Ethereum network it is needed to create an account. For access Ethereum it is needed address and private key for authentication These values are not stored in any database. Only the account owner knows it.

Ethereum wallets are software applications to interact with Ethereum account. The wallet lets users to read balance, send transactions, and connect to any decentralized applications. Most of the wallets also allows to generate an Ethereum account.

Ethereum has two (2) types of accounts or addresses which are:

- Accounts that hold Ethers (ETH): This account is called "Externally Owned Accounts (EOAs)". You can make an ETH transfer or payment by signing transactions. This account is controlled by anyone with the private keys.
- Accounts that hold ETH and smart contracts: This account is for smart contracts deployed to the network This account is controlled by code.

2.3.3. Smart contract

Smart contract are self-executing pieces of software code that run on a blockchain and contains an agreement between two entities. Smart contract automatically "run" or execute when a relevant set of terms are met. As such, these "contracts" can automatically verify and perform a transaction between different parties.

Smart contracts are a type of Ethereum account. This means they have a balance, and they can send transactions over the network. However, they're not controlled by a user, instead they are deployed to the network and run as programmed. After a smart contract has been coded and uploaded, it will sit in this account and wait to be activated. User accounts can then interact with a smart contract by submitting transactions that execute a function defined on the smart contract. Smart contracts can define rules, like a regular contract, and automatically enforce them via the code.

Perhaps the best metaphor for a smart contract is a vending machine, as described by cryptographer Nick Szabo in his blog page [18]: To get a snack, check inputs and produce output, the logic programmed into the vending machine can be as following:

1	money + snack selection = snack dispensed
---	---

Smart contract characteristics are described in *devteam* blog [19]:

- It must have an open-source code.
- The code is stored in a blockchain.
- The code can't be altered after it's deployed.
- The execution is automatic.
- The code and execution results are visible in a decentralized blockchain.
- The execution results are irreversible.

Web3 is the interface where the smart contract interacts with the user. it will use the same structure that traditional web pages we already explained on previous section:

- html: structure
- css: style
- JavaScript: advance code for many functions

To run the JavaScript program is needed NodeJS, an open-source runtime environment for executing JavaScript code outside a browser.

2.3.4. IPFS

In the 2014, Juan Benet wrote a paper explaining InterPlanetary File System (IPFS) as "peer-to-peer distributed file system that seeks to connect all computing devices with the same system of files" [4]. In simples' words, IPFS is a distributed system for storing and accessing files. These files can be website, application, and data.

HTTP	IPFS
It uses a centralized client server approach	It uses a decentralized peer to peer approach
Data is requested using the address on which data is hosted	Data is requested using the cryptographic hash of that data.

HTTP	IPFS
Data cannot be accessed if the server is down or fails or any link gets broken.	Data is copied to multiple nodes, hence it can be accessed whenever needed.
The bandwidth provided is low, as multiple clients request from a single server at the same time.	Bandwidth is high, as data is requested from the closest peer who has the copy of that data.
One has to set up a hosting server or pay for one, in order to make content public available.	Uploading content on the IPFS network does not require a host server, every node hosts the data on the network.
HTTP is well established as an industry standard.	IPFS is relatively newer and is not yet as popular as HTTP.
HTTP support is inbuilt on almost all machines.	To run IPFS you need to access it using the HTTP to IPFS portal or manually setup up an IPFS node on your machine.
HTTP is used by almost everyone to access the web.	There is a shortage of IPFS nodes due to its low popularity.

Table. II. HTTP and IPFS comparation

IPFS is like HiperText Transfer Protocol (HTTP), used in world wide web to store and access files. The main difference is, in HTTP, when you request a resource, it is needed to know the *domain name* or *IP address* of the server that is hosting the file, while IPFS need to know the hash associated to that content [21]. The following table shows the differences between these protocols. Every piece of content that uses the IPFS protocol has a content identifier, or CID, that is its hash.

IPFS has five layers:

- naming — a self-certifying PKI interplanetary naming system (IPNS)
- MerkleDAG — data structure format (thin waist)
- exchange — block transport and replication
- routing — locating peers and objects using DHT
- network — establishing connections between peers

A distributed hash table (DHT) is a distributed system designed to map values to keys. It is used to support routing and discover content and peers on the network

JS IPFS is the implementation of IPFS protocol in JavaScript. It can run on any browser, inside a service or web worker, browser extensions, Electron and in Node.js.

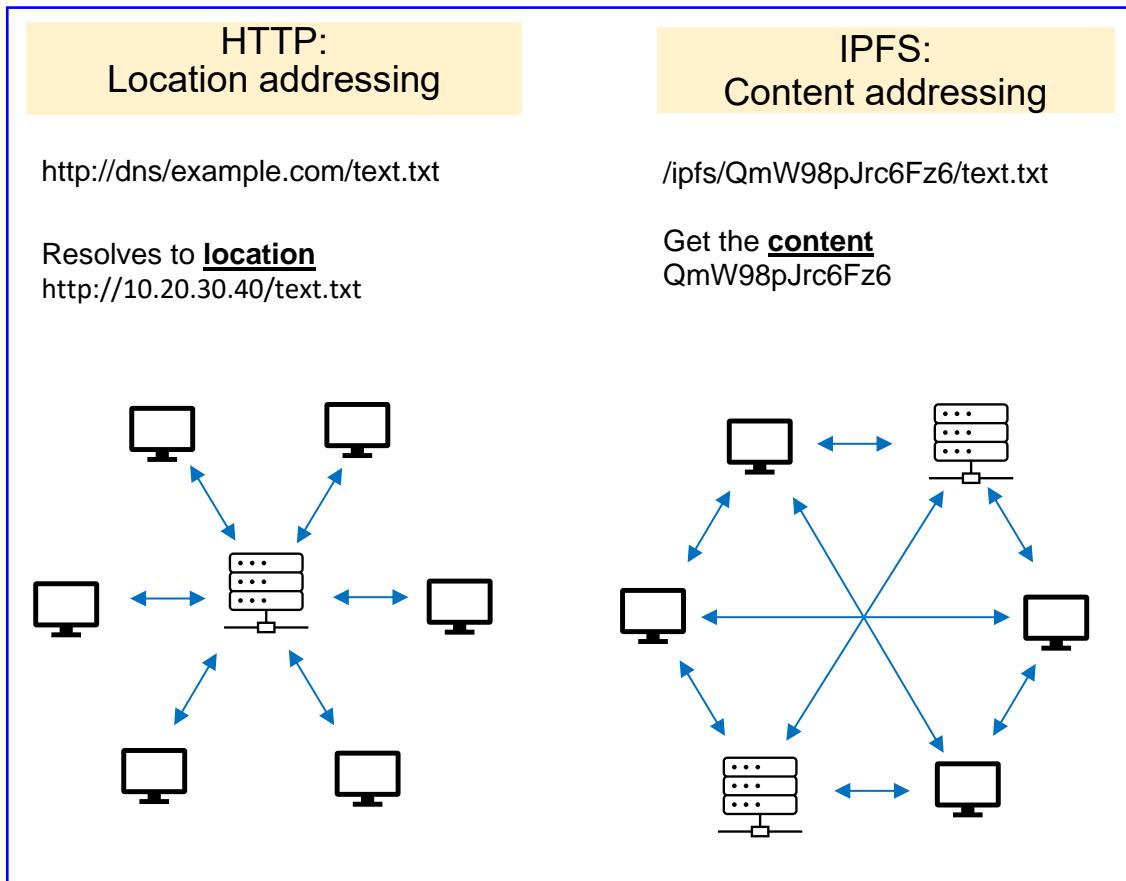


Figure 8. HTTP and IPFS approach

3 Design and Configuration

3.1. Architecture design

For this project it was implemented a complete end to end Proof of Concept. The final goal is to obtain the daily report for employee attendance. The data is generated in IoT device with fingerprint sensor, then, process all data into a single excel file inside the IoT Edge node. Finally, store the logs in a blockchain. This logs, thanks to blockchain characteristics, can not be modified, and people that read the file can trust in it.

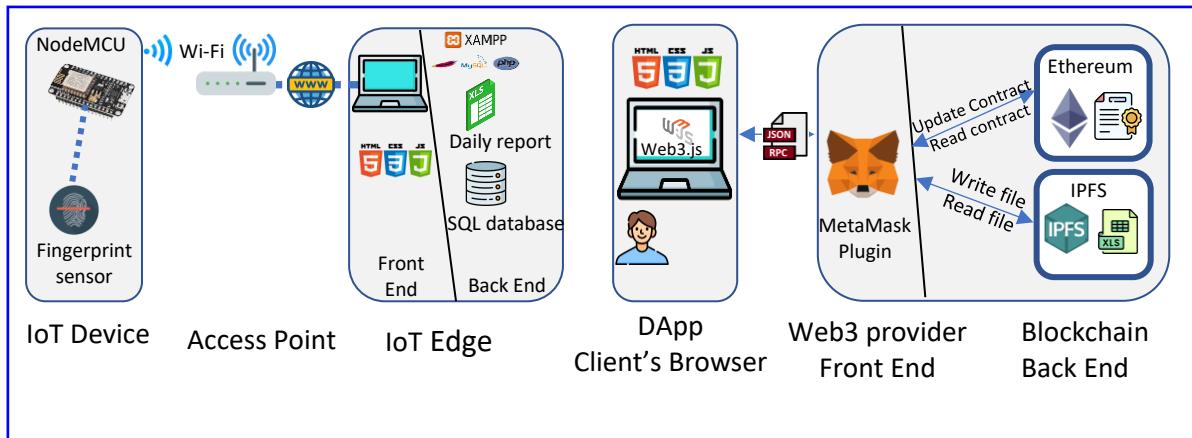


Figure 9. System architecture

3.2. Use case

The system will be used as attendance system, with three main actors:

- Employees: are responsible to use IoT device every time they start and finish to work, using fingerprint sensor. Also, any employee can access the daily report, to verify the attendance time, late arrival, early exit, or overtime period.
- Labor inspector: can get daily report from blockchain.
- Company responsible: it the person responsible for obtaining every day the daily log stored in IoT Edge inside traditional SQL database. As soon as the daily log is extracted, the company is responsible to upload the report in the blockchain network.

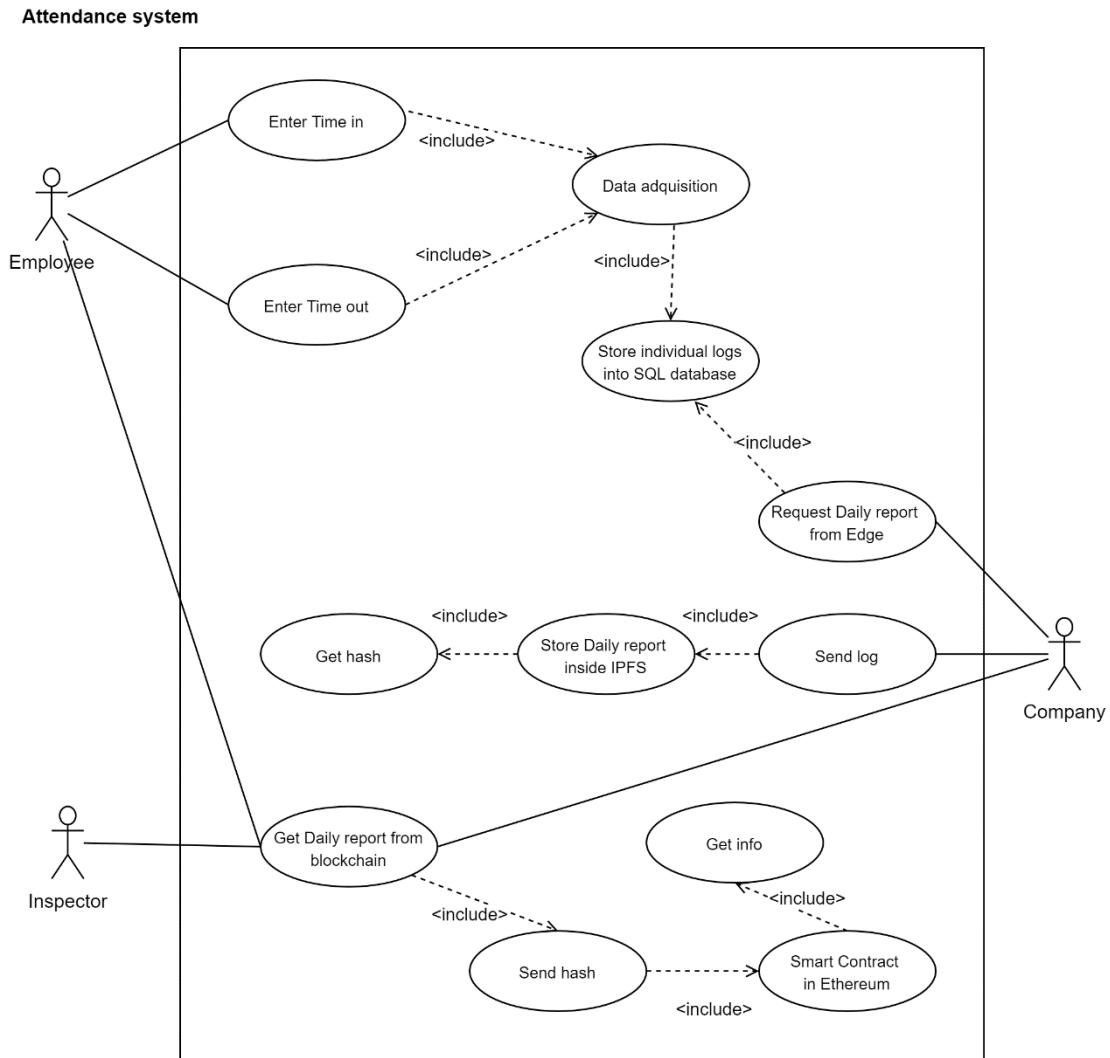


Figure 10. Use Case diagram

3.3. IoT device configuration

3.3.1. Hardware connection

In the following figure is showed how an OLED Display & Fingerprint Sensor are connected to NodeMCU. The I2C pins of OLED Display, i.e SDA & SCL are connected to NodeMCU D2 & D1 pins respectively. Similarly, fingerprint sensor is connected to UART pins D5 & D6. In this project, the colors for all cabling are specified in the NodeMCU pinout connection table. So connect it by finding appropriate color wires else the module won't be detected by NodeMCU.

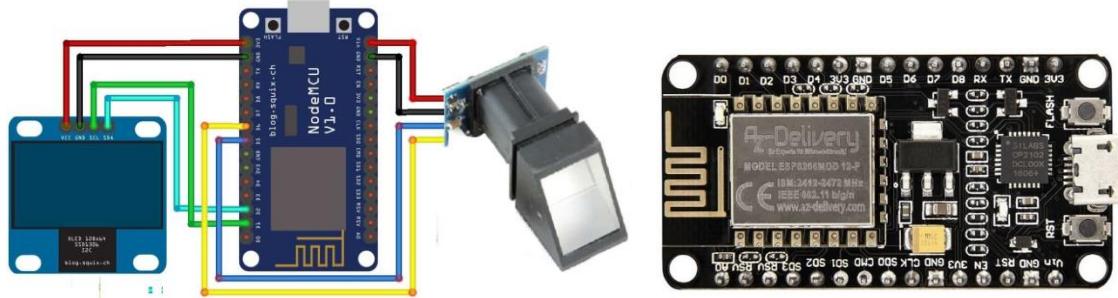


Figure 11. Circuit Diagram for IoT Fingerprint Attendance System

Device	Pin description	NodeMCU pin number	Color cable
Fingerprint sensor	Vin 3-5.5V	3V3	Red
	GND	GND	Black
	Serial TX	D7	Yellow
	Serial RX	D8	Dark blue
OLED screen	Vin 3-5 V	3V3	Red
	GND	GND	Black
	SCL i2c	D3	Green
	SDA i2c	D2	Light blue

Table. III. NodeMCU pinout connection.

3.3.2. Configure NodeMCU Lolin V3

The configuration was done using NodeMCU user Manual provide by AZ delivery [12].

Steps to configure:

- 1) Download the Arduino IDE from web [23]. In this project was used version 1.8.13
- 2) Download CH3408 drive, to recognize USB port [24] and connect NodeMCU
- 3) Set up your Arduino IDE as: Go to File > Preferences. Copy the URL below:
http://arduino.esp8266.com/stable/package_esp8266com_index.json to get the ESP board manager extensions
 Placing the http:// link before the URL lets the Arduino IDE use it, otherwise it gives you a protocol error.
- 4) Go to Tools > Board > Board Manager> Type "esp8266" and download the Community esp8266 and install.
- 5) Set up your chip as:

- Tools -> Board -> NodeMCU 1.0 (ESP-12E Module)
 - Tools -> Flash Size -> 4M (3M SPIFFS)
 - Tools -> CPU Frequency -> 80 Mhz
 - Tools -> Upload Speed -> 921600
 - Tools-->Port--> (whatever it is, in our case was COM5)
- 6) Download and run the 32 bit / 64 bit flasher exe (ESP8266Flasher.exe) at Github [25] (Search for nodemcu flasher at Github)
- 7) In Arduino IDE, look for the Blink program. Load, compile and upload.
- 8) Go to FILE> EXAMPLES> ESP8266> BLINK, it will start blinking.

If leds start to blink, the setup is finished

3.3.3. Configure OLED display.

The configuration was done using OLED display User Manual provide by Adafruit library, SSD1306 model [26]. Steps to configure:

- 1) Download external library Adafruit_SSD1306 and Adafruit_GFX
- 2) Sketch > Include library > Manage libraries
- 3) After installing the Adafruit_SSD1306 and Adafruit_GFX library, restart the Arduino IDE.
- 4) You should now be able to access the sample code by navigating through menus in this order:
- 5) File > example > Adafruit_SSD1306 > ssd1306_128x64_i2c
- 6) Change program to be compatible with NodeMCU and 128x64 resolution

```
/* file: ssd1306_128x64_i2c.ino - OLED connection settings */
#define SCREEN_WIDTH 128          // OLED display width, in pixels
#define SCREEN_HEIGHT 64           // OLED display height, in pixels

//define display object
#define OLED_RESET    0           // Reset pin
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#define SCREEN_ADDRESS 0x3D        // 0x3D for 128x64 and 0x3C for 128x32
```

- 7) compile and upload.

If worked, you will see a moving line at the screen. The setup is finished

3.3.4. JM-101 Fingerprint Scanner Sensor

The sensor is relatively easy to use with any TTL capable microcontroller or system. Although this is not the exact same sensor, it is compatible with the Adafruit tutorial [8]. The first step is to test the fingerprint stand alone, using the board as a "bridge" to USB port and configure from Windows PC using free Windows based software. In this way, NodeMCU do not interact with the sensor. And later, connect to NodeMCU.

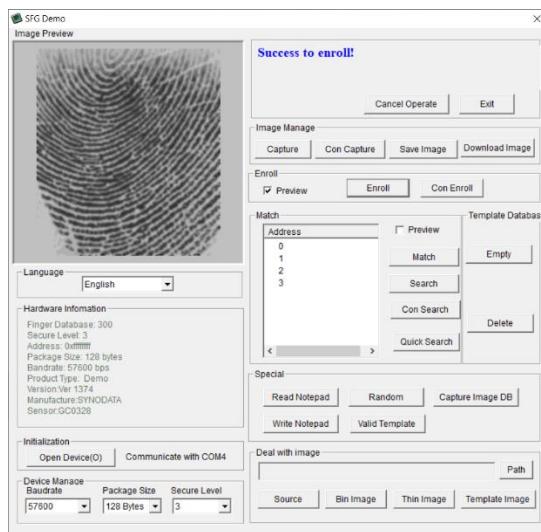
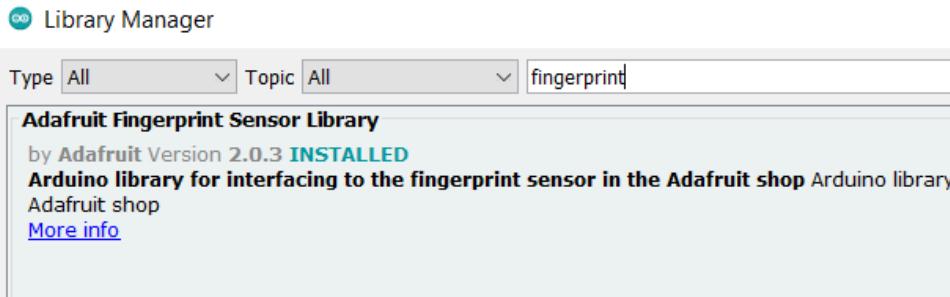
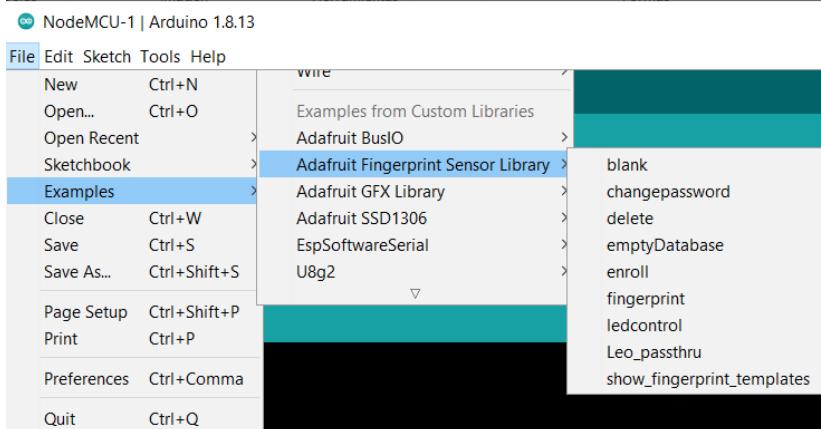


Figure 12. SFG windows program to test fingerprint sensor

- 1) Connect fingerprint cables to UART NodeMCU (TX and RX)
- 2) Load empty sketch to bypass NodeMCU
- 3) Download DOS program SFGDemoV2.0 from Adafruit web page [27]
- 4) Start up the SFGDemo software and click Open Device from the bottom left corner. Select the COM port used by the Arduino. If you get an error when you Open Device, check your wiring, try swapping the RX and TX wires on the sensor. Also, close Arduino program as the communication is only bypass
- 5) Enrol your first fingerprint
- 6) Now, you are sure the sensor is working, it is time to connect to NodeMCU. Reconnect cables to D7 and D8 pins
- 7) Download external Adafruit Fingerprint Sensor Library
Sketch > Include library > Manage libraries



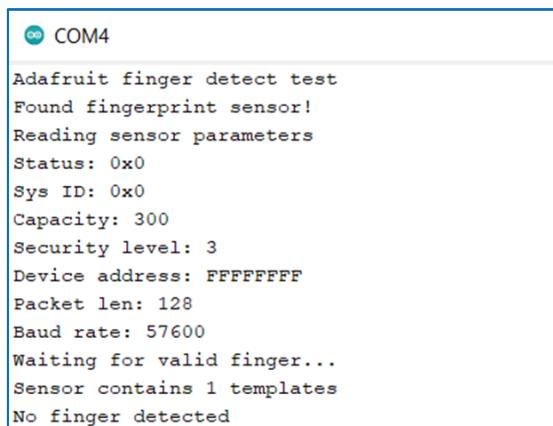
- 8) After installing the Adafruit Fingerprint Sensor Library, restart the Arduino
- 9) You should now be able to access the sample code by navigating through menus in this order: File > example > Adafruit Fingerprint Sensor Library > enroll



- 10) Change program to be compatible with NodeMCU

```
// * file: enroll.ino - Fingerprint connection settings */
#include <Adafruit_Fingerprint.h>
#define Finger_Rx 14 //D5
#define Finger_Tx 12 //D6
SimpleTimer timer;
SoftwareSerial mySerial(Finger_Rx, Finger_Tx);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
```

- 11) Compile and upload. If worked, you will see on the console the message about fingerprint detection. The setup is finished



3.3.5. Configure IoT Device

On previous steps, we have tested all components stand alone to guarantee proper cabling connections and performance of the device. The NodeMCU use only a single *.ino code for all components: NodeMCU, fingerprint sensor, messages displaye on OLED screen and Wi-Fi connection to IoT Edge Node.

For this project, there are available many codes in GitHub, and the following code was used as a library and modified according to our project:

<https://github.com/InfinityWorldHI/Biometric-attendance-system-V1.0>

Steps to configure:

- 1) Configure Wifi credentials

```
#include <WiFiClient.h>
#include <ESP8266WiFi.h>
const char *ssid = "Jeidi_AP";                                //Enter your WiFi values
const char *password = "12345678";                            //Enter your WiFi password
```

- 2) Initialize devices and services. Loop to check if any fingerprint is available

```
void setup() {
    Serial.begin(115200);
    connectToWiFi();
    finger.begin(57600);
}

void loop() {
    CheckFingerprint();           //Check the sensor if there is a fingerprint.
    delay(10);
}

void CheckFingerprint(){        // Get the Fingerprint ID from the Scanner
    FingerID = getFingerprintID();
    DisplayFingerprintID();
}

int getFingerprintID() {
    uint8_t p = finger.getImage();
    switch (p) {
        case FINGERPRINT_OK:          //Serial.println("Image taken");
            break;
        case FINGERPRINT_NOFINGER:    //Serial.println("No finger detected");
            return 0;
        case FINGERPRINT_IMAGEFAIL:   //Serial.println("Imaging error");
            break;
    }
}
```

```
    return -2;
}
                                // OK success!
p = finger.image2Tz();
switch (p) {
    case FINGERPRINT_OK:           //Serial.println("Image converted");
}
                                // OK converted!
p = finger.fingerFastSearch();
if (p == FINGERPRINT_OK) {      //Serial.println("Found a print match!");
} else if (p == FINGERPRINT_PACKETRECIEVEERR) { //Serial.println("Comm error");
    return -2;
} else if (p == FINGERPRINT_NOTFOUND) { //Serial.println("not find a match");
    return -1;
}
// found a match!
//Serial.print("Found ID #"); Serial.print(finger.fingerID);
//Serial.print(" with confidence of "); Serial.println(finger.confidence);
return finger.fingerID;
}
```

3) Add the communication with IoT Edge, that is, send and receive data

```
#include <ESP8266WebServer.h>
#include <ESP8266HTTPClient.h>

const char* device_token  = "6b07332bc7f1c0b9";           // IoT device ID
String getData, Link;
int FingerID = 0, t1, t2;                                // Fingerprint ID

void SendFingerprintID(int finger){                      //Sending Fingerprint ID
    if(WiFi.isConnected()){
        HTTPClient http;                                //Declare object HTTPClient
        //GET Data
        getData = "?FingerID=" + String(finger) + "&device_token=" + device_token;
        // Add the Fingerprint ID to the Post array in order to send it
        //GET methode
        Link = URL + getData;
        http.begin(Link); //initiate HTTP request //Specify content-type header

        int httpCode = http.GET();                      //Send the request
        String payload = http.getString();              //Get the response payload

        Serial.println(httpCode);                      //Print HTTP return code
        Serial.println(payload);                      //Print request response payload
        Serial.println(finger);                        //Print fingerprint ID

        if (payload.substring(0, 5) == "login") {
            String user_name = payload.substring(5);
            display.print(F("Welcome"));
            display.print(user_name);                  // Employee enter to work
        }
    }
}
```

```

    }
    else if (payload.substring(0, 6) == "logout") {
        String user_name = payload.substring(6);
        display.print(F("Good Bye"));
        display.print(user_name);                                // Employee ends to work
    }
    delay(10);
    http.end();                                              //Close connection
}
}

```

To see complete configuration file see Annex B: IoT Device Arduino Code and Annex C: change Bitmap to array.

3.4. Edge node configuration (centralized)

The main function of the edge node is to aggregate input values over each time interval and calculate a single output value. The input is obtained from IoT device when the employee register the attendance, and the output is a daily excel report.

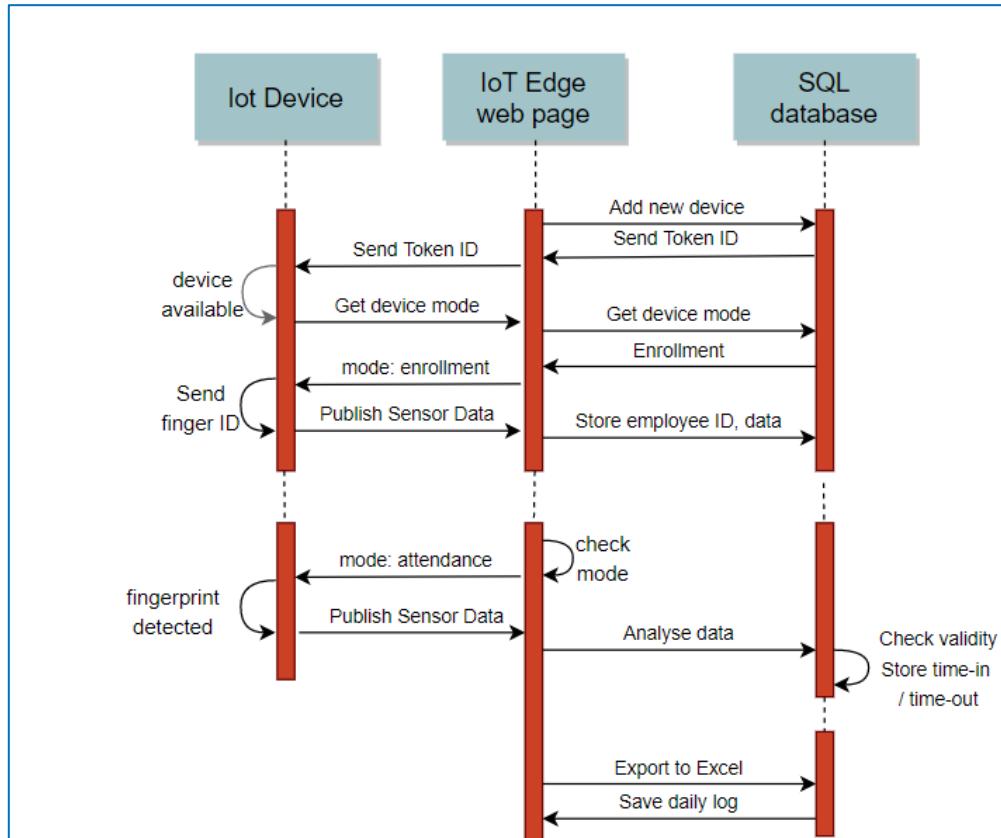
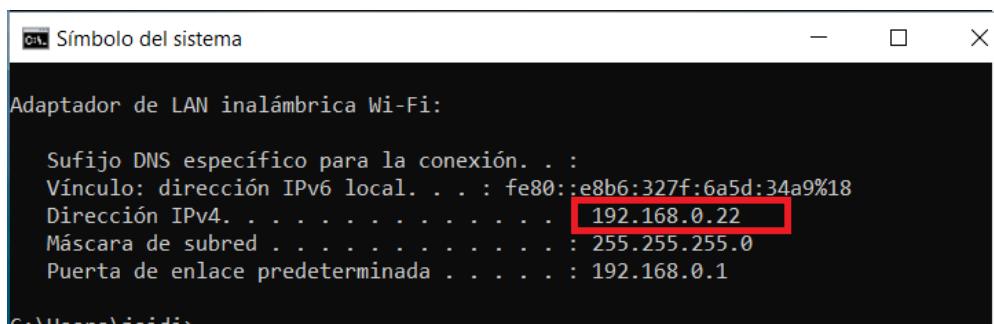


Figure 13. IoT Edge Sequence Diagram

Everyday can be generated a single Daily report containing all time attendance for all employees. If assume a small company with 25 employees, every day will be generated four (4) attendance logs per employee (morning time in, morning time out, afternoon time in, afternoon time out), total 100 logs, per day, 500 logs by week and 26.000 logs by year. Using IoT Edge computing node, instead of handle 26.000 logs a year, it can keep one register per day, that means, 52 registry a year.

To create an IoT Edge server, the following code was used as a template and modified according to our project: <https://github.com/InfinityWorldHI/Biometric-attendance-system-V1.0>. Steps to configure:

- 1) Open CMD window and check the server IP (IP config command), and write it on Arduino code (because the IoT device need to communicate with IoT Edge)



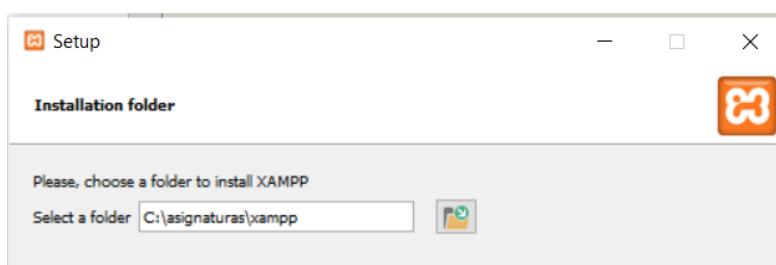
```
C:\ Símbolo del sistema
Adaptador de LAN inalámbrica Wi-Fi:

Sufijo DNS específico para la conexión. . . :
Vínculo: dirección IPv6 local. . . : fe80::e8b6:327f:6a5d:34a9%18
Dirección IPv4. . . . . : 192.168.0.22
Máscara de subred . . . . . : 255.255.255.0
Puerta de enlace predeterminada . . . . : 192.168.0.1

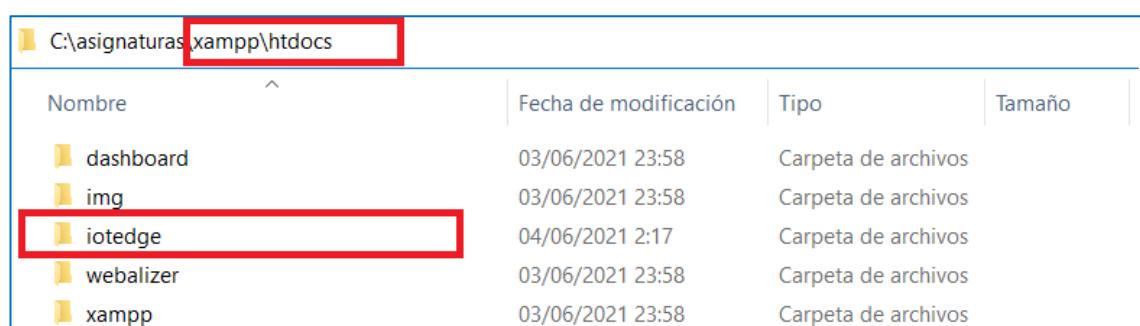
C:\Users\jeidi>
```

```
String URL = "http://192.168.0.22/iotedge/getdata.php"
```

- 2) Install a local server xampp server [28]:



- 3) Go to GitHub [40]. Copy folder "biometricattendance" into C:\...\xampp\htdocs folder and rename it to " iotedge "



Nombre	Fecha de modificación	Tipo	Tamaño
dashboard	03/06/2021 23:58	Carpeta de archivos	
img	03/06/2021 23:58	Carpeta de archivos	
iotedge	04/06/2021 2:17	Carpeta de archivos	
webalizer	03/06/2021 23:58	Carpeta de archivos	
xampp	03/06/2021 23:58	Carpeta de archivos	

4) Add new database to server.

- run xampp -> start apache -> start MySQL services (click start button)

The screenshot shows the XAMPP Control Panel interface. It lists two services: Apache and MySQL. Apache is running with PID 32624 and Port 80, 443. MySQL is running with PID 16012 and Port 3306. There are 'Stop' buttons next to each service entry.

Below the control panel, there are two bullet points:

- open firefox brower -> open <http://localhost/phpmyadmin/>
- click on new -> dabatabse -> create

The screenshot shows the phpMyAdmin interface for managing databases. On the left, a sidebar shows existing databases: New, information_schema, iotedge, iotedgeweb, and mysql. The main area is titled 'Databases' and shows a 'Create database' form. The database name 'iotedge' is entered, and the character set 'utf8mb4_general_ci' is selected. A 'Create' button is visible at the bottom right of the form.

5) Import database to server

- import database
- click on import, browse
- select file " iotedgeedb.sql", click on go button

The screenshot shows the phpMyAdmin 'Import' page for the 'iotedge' database. The title bar indicates the current location is 'localhost / 127.0.0.1 / iotedge'. The main area is titled 'Importing into the database "iotedge"' and contains a 'File to import:' section. It includes instructions for compressed files and a 'Browse...' button where the file 'iotedge.sql' is selected. Other options like 'Character set of the file:' are also visible.

6) check the database is complete.

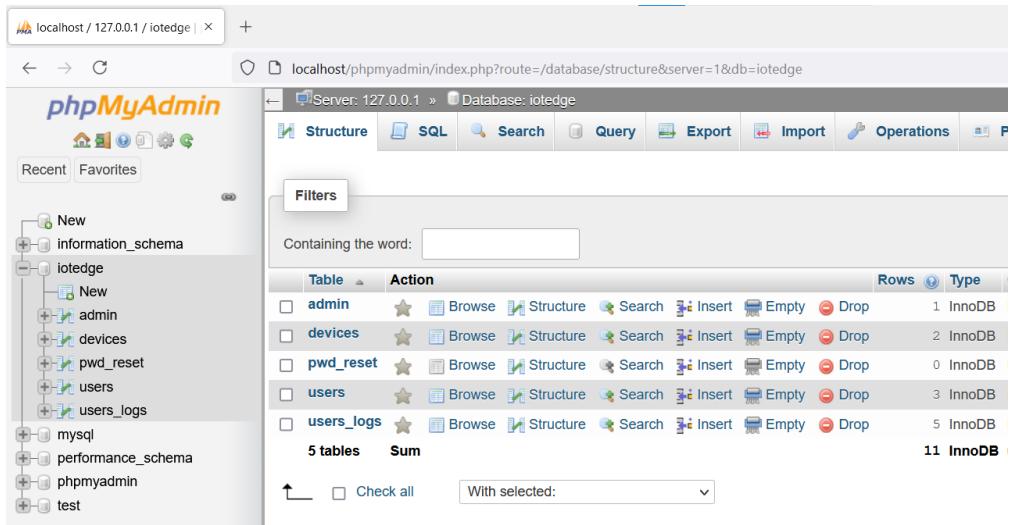


Figure 14. Import database to server

- 7) Open a browser and login to <http://localhost/iotedge/index.php> to start.

E-mail: admin@gmail.com
password: 123

- 8) Add device: devices > New Device -> Create
- Device Name: NodeMCU1
 - Device Location: Madrid_building1
- 9) Get device token and write in on Arduino NodeMCU code
- Device Name: NodeMCU1
 - Dev.UID: ea16b938f7e73a9d (generated by IoT web server)
- 10) Load NodeMCU.ino program into Arduino board
- 11) IoT system configuration is finished. The system is ready to:
- enroll employee fingerprint into database details.
 - get employees time-in and time-out
 - generate excel file with employee attendance daily record

3.5. Smart contract code and deploy

3.5.1. Install MetaMask

MetaMask is a web extension, that runs in many browsers (Chrome, Firefox, Opera or Brave) and it is free to download, which allows to manage Ethereum private keys via web browser. By doing so, it serves as a wallet for Ether and allows to run Ethereum Decentralized Apps (DApps). With MetaMask, private keys are only stored on your

device. With this identity, you can perform different actions in both centralized (web resource controlled by a central authority) and decentralized networks, using the website as an access point and/or as a graphical user interface.

In this project we will use the tutorial about create a simple DApp from MetaMask webpage [20]. Installation of the MetaMask Wallet:

- 1) Open official MetaMask [29] page in Firefox browser and Click on "Install MetaMask for Firefox" button

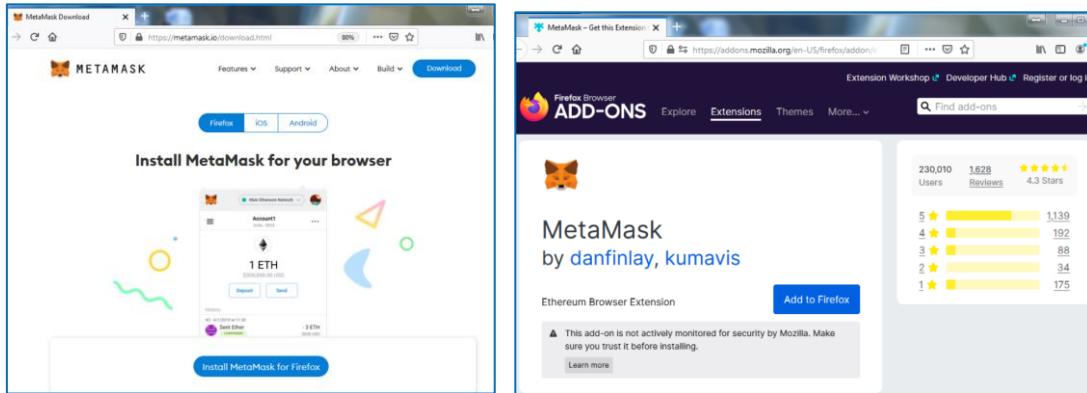
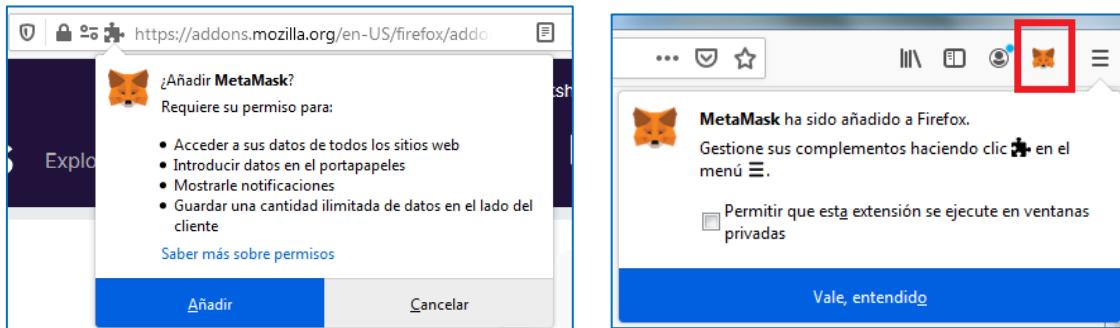
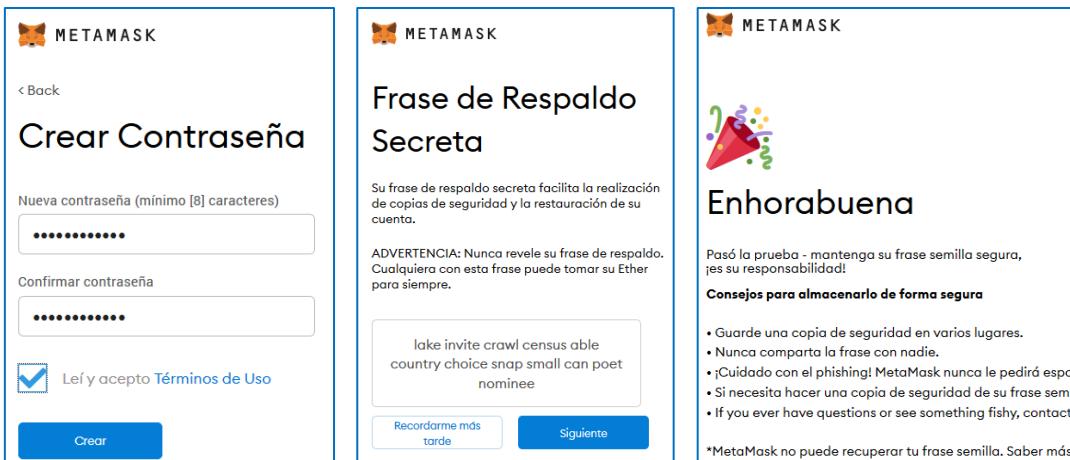


Figure 15. Installation of MetaMask

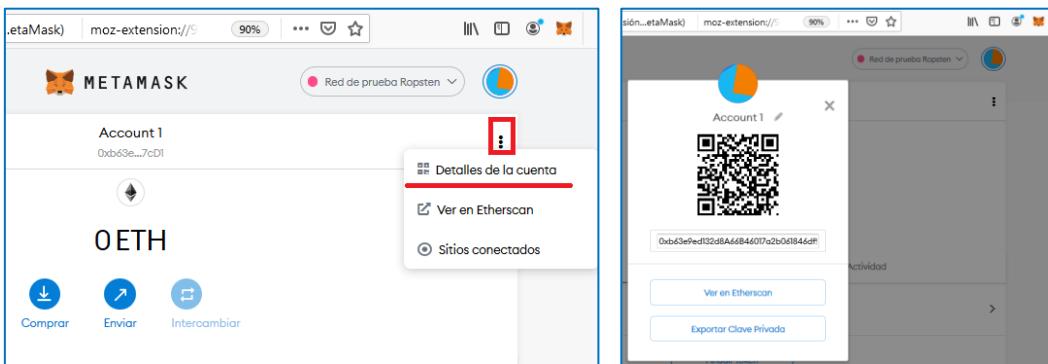
- 2) The installation will open Mozilla Firefox Addons and extension. Click on Add to Firefox. Approve all permissions (read, store data, show, etc). Click on Add. If extension was added, it is possible to see a confirmation message, and see the MetaMask logo in the tools bar.



- 3) The system will ask to use existing account or create a new one. Click on Create Wallet. Write MetaMask password. Read and accept terms and conditions. Click on Create. The system will show a seed phrase, with 12 words, to be used in case forgot the password. It is important to store the seed phrase in a safe place because it is the only way to recover the account. If the user can not remember the password, and can not remember the seed phrase, it will be impossible to access the account. Finally, it will be show a congratulations message



- 4) See the account number: click on three dots to display account details



3.5.2. Get funds to deploy Smart Contract

Anyone can write a smart contract and deploy it to the network. They just need to learn how to code in a smart contract language and have enough ETH to deploy the contract. Deploying a smart contract is technically a transaction, so there is needed to pay for Gas in the same way that it is needed to pay gas for a simple ETH transfer. However, Gas costs for contract deployment are far higher: A standard ETH transfer requires a gas limit of 21.000 units of gas, while a contract deployment need about 100.000 units of gas, depend on contract complexity.

For deploy smart contract on the main network real ether costs money and handling it requires a bit more experience. For this reason, it will be used the Ropsten Ethereum, a testing network that runs the same protocol as Ethereum does. There are a few ways of getting Ropsten Eths: mining with your computer, asking someone who mines on their computer or requesting from a "faucet". Faucet is a pool of Ropsten Ethereum from where you require rETHs and it sends them to you automatically, straight to your address at no cost

Request ETH from faucet:

- 1) Open faucet page [30] using the same web browser with MetaMask, and click on MetaMask extension
- 2) Select Ropsten network
- 3) click on "request 1 ether from faucet" button and wait some seconds (the transactions need time to be executed)
- 4) Verified the account contains 1 ETH and show the transaction ID, just for reference

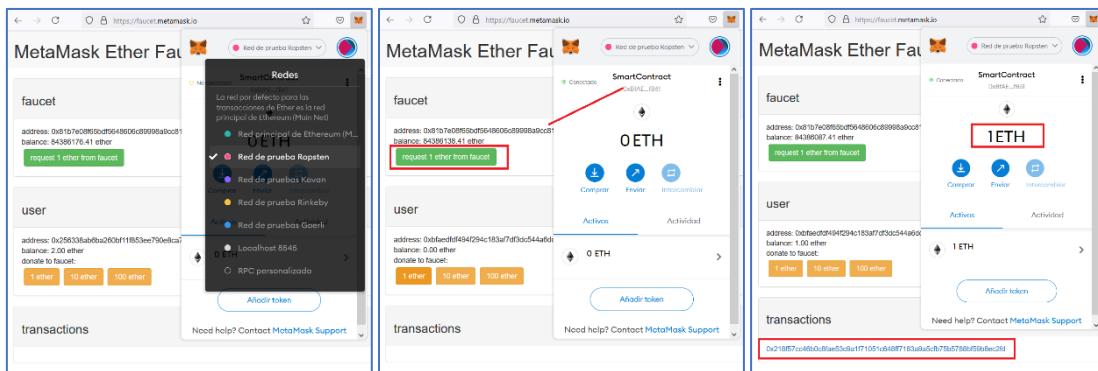


Figure 16. Request ETH from faucet

3.5.3. Create and deploy Smart contract

Ethereum has developer-friendly languages for writing smart contracts like Solidity [31] and Vyper [32]. In this project the smart contract was writing as a text file, named "*timinglog.sol*" using Visual Code. and compiled in Remix IDE (Integrated Development Environment) [33] using Solidity language. This web application can run on Windows or Unix environment, and it is used to write, debug, and deploy Ethereum Smart Contracts.

Steps to create a contract named *timinglog.sol*:

- 1) Install and open Visual Studio Code (VSC) [34] to create the file. Add state variables: solidity version and "contract" name, for example *timinglog*

```
pragma solidity ^0.6.1;
pragma experimental ABIEncoderV2; //experimental not needed for v0.8
contract timinglog {
    uint constant DAY_IN_SECONDS = 86400;
    uint constant YEAR_IN_SECONDS = 31536000;
    uint constant HOUR_IN_SECONDS = 3600;
    uint constant MINUTE_IN_SECONDS = 60;
    ...
}
```

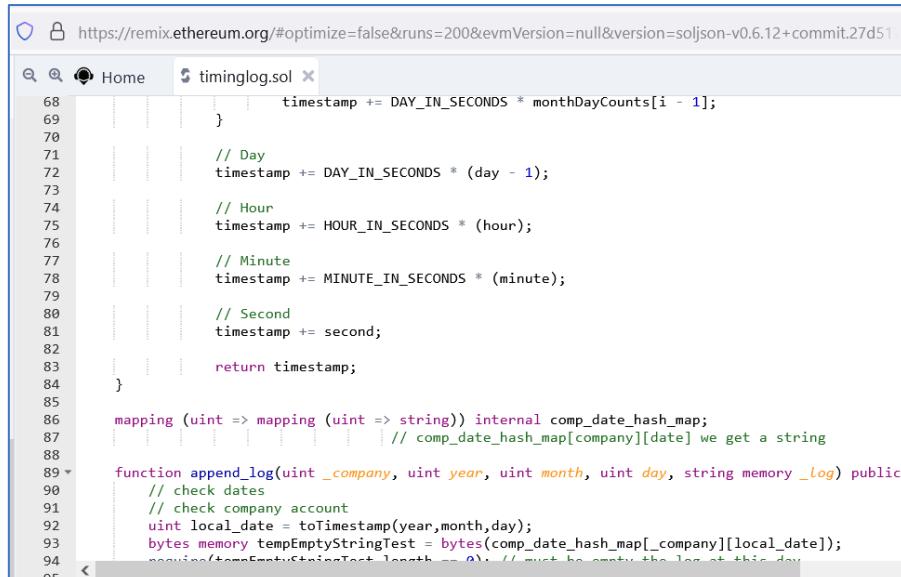
- 2) Define functions. In Ethereum is important to distinguish between a write function and read function, because write function will modified blockchain and read will not. For this reason, write function create a hash, mining blocks, and read only search specific hash

```
function append_log(uint _company, uint year, uint month, uint day, string memory _log) public returns (bool OK) {
    // check dates
    // check company account
    uint local_date = toTimestamp(year,month,day);
    bytes memory tempEmptyStringTest = bytes(comp_date_hash_map[_company][local_date]);
    require(tempEmptyStringTest.length == 0); // must be empty the log at this day
    comp_date_hash_map[_company][local_date] = _log;
    return true;
}
```

```
function get_log(uint _company, uint year, uint month, uint day) view public returns (string memory _log) {
    uint local_date = toTimestamp(year,month,day);
    bytes memory tempEmptyStringTest= bytes(comp_date_hash_map[_company][local_date]);
    require(tempEmptyStringTest.length != 0); // must have data the log at this day
    _log = comp_date_hash_map[_company][local_date];
    return _log;
}
```

To see the complete file, please see the Annex D: Smart contract - timinglog.sol

- 3) Upload Solidity code. Visit Remix [33] and create a new file. Click on icon panel, file explores icon (two sheets icon). Click in the option you will open the smart contract: New File, New Folder, Publish to GitHub or Load Local File.



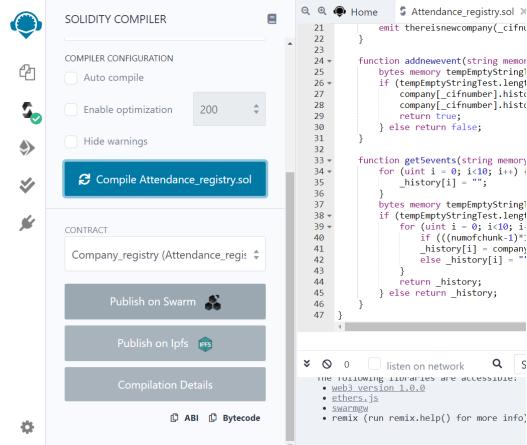
The screenshot shows the Remix IDE interface with the URL <https://remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.6.12+commit.27d51>. The code editor displays the Solidity code for the timinglog contract. The code includes a timestamp calculation loop and two functions: append_log and get_log. The append_log function checks dates and company account, converts them to a timestamp, and stores it in the comp_date_hash_map. The get_log function retrieves the stored log for a given date.

```
timestamp += DAY_IN_SECONDS * monthDayCounts[i - 1];
// Day
timestamp += DAY_IN_SECONDS * (day - 1);
// Hour
timestamp += HOUR_IN_SECONDS * (hour);
// Minute
timestamp += MINUTE_IN_SECONDS * (minute);
// Second
timestamp += second;
return timestamp;
}

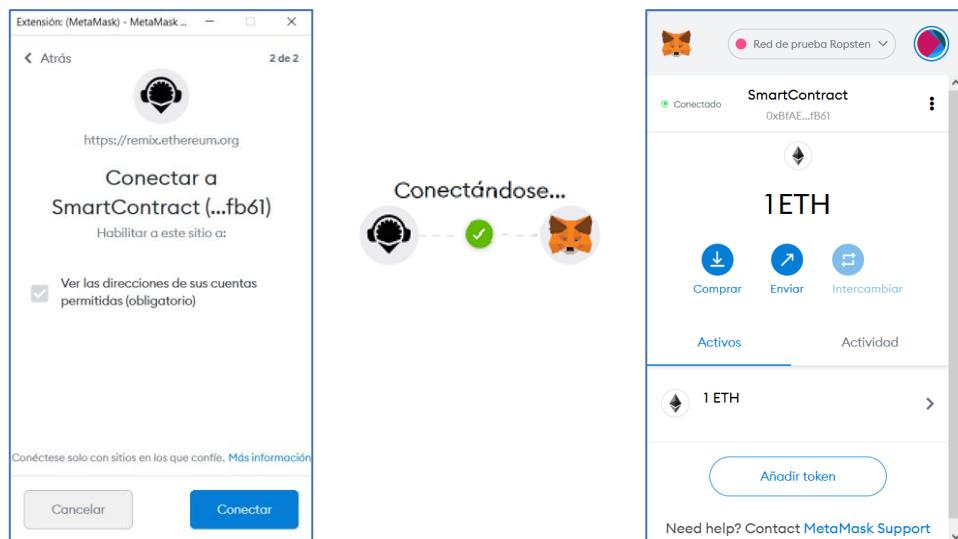
mapping (uint => mapping (uint => string)) internal comp_date_hash_map;
// comp_date_hash_map[company][date] we get a string

function append_log(uint _company, uint year, uint month, uint day, string memory _log) public {
    // check dates
    // check company account
    uint local_date = toTimestamp(year,month,day);
    bytes memory tempEmptyStringTest = bytes(comp_date_hash_map[_company][local_date]);
    require(tempEmptyStringTest.length == 0); // must be empty the log at this day
    comp_date_hash_map[_company][local_date] = _log;
}
```

- 4) Click on "Solidity compiler" icon, click on "compile timinglog.sol" button. If compile is successful, it show a button for "compilation details" and a green tick on solidity button. Store ABI Code, click on ABI and store in safe place

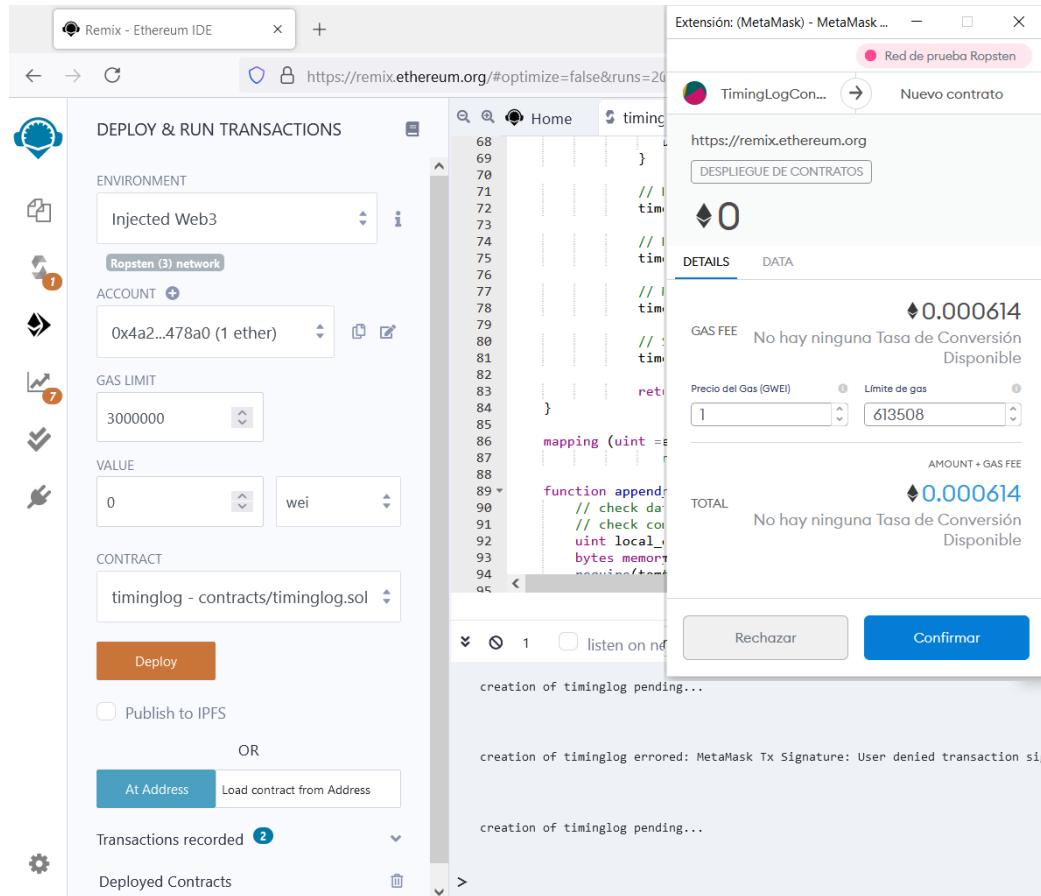


- 5) Deploy contract. Click on Ethereum logo, select Injected Web3 environment.. MetaMask plugin will be ask for password. Enter your credentials. MetaMask will ask permit to connect to Remix. Select which account will be connected to Remix, and what network is used (ropsten). To check, only click on MetaMask plugin logo

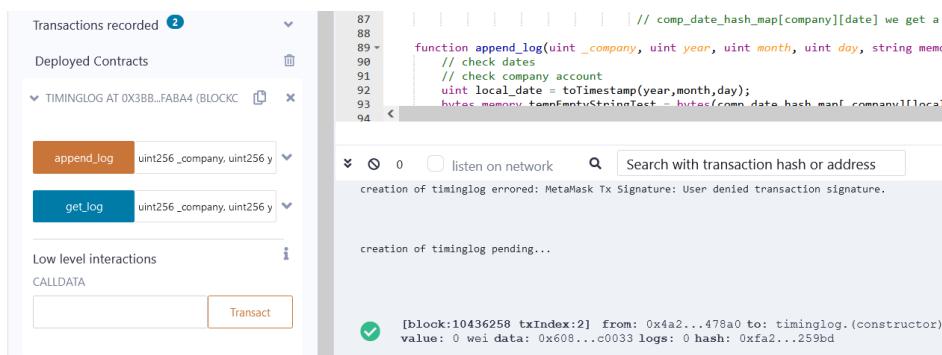


- 6) Click on "Deploy" button at Remix page. MetaMask will ask for Gas fee (1GWEI) and Gas Limit (613508), and show the price, in ETH. This is the price to deploy the contact. Click on "Confirm" button. MetaMask will request to miners to work with this contract. If there is not enough gas, the contract will not be deployed, but the computation cost will be charged to the account. Wait some seconds, as the process need time, until see the confirmation message

2. STATE OF THE ART



- 7) Once the blockchain is created, Remix will notify the transaction was done. To see details of transaction, see debug panel and click on hash. Also save contract address.



Contract: 0x3bb37250390a96f6dc99fedd6386359c03cfaba4

Transactions details available at:

<https://ropsten.etherscan.io/tx/0xfa2da61a1918ecdfdc22e78411cd92aca00e44a75b1ef4519e802c7e4c2259bd>

For a detailed explanation, see Annex E: How to deploy a Smart contract.

3.6. Decentralized application configuration

Node.js lets developers to write command line tools and to produce dynamic web page content before the page is sent to the user's web browser. Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language. Node has its own pre-installed package management, known as Node Package Manager (npm).

To create DApp, it was followed the instructions on MetaMask tutorial stored in:

<https://github.com/BboyAkers/simple-dapp-tutorial> that give a basic template for anyone that want to develop a decentralized application.

Steps to configure:

- 1) Download Node.js program from web [35]. At the time this project was developed, version was 16.3.0, that includes npm 7.15.1
- 2) Browse the location where you have saved the zip file and double-click it to launch. Click Run, click Next to follow the wizard, finally click on Finish
- 3) Verified installation. Open cmd command prompt and write the command:

```
> node --version
> npm -v
```

```
C:\Windows\System32\cmd.exe
C:\Users\jeidi>node --version
v16.3.0
C:\Users\jeidi>npm -v
7.18.1
C:\Users\jeidi>
```

- 4) Clone GitHub repository [41]. For this project it is needed only the "start" folder. Click on Download ZIP file
- 5) Go to start folder. Rename folder to "DAppTimeLog". Open this folder and check the following files are available:

```
-- index.html
-- contract.js
-- metamask.css
```

3.6.1. HTML file structure

Based on existing index.html file, it will be used as a template and modified according to our project. To edit the file, it can be used Visual Studio Code [34]. The file is divided by sections. The first step is to design the layout to connect it to smart contract, DApp need three sections: Basic (Connect to Ethereum), store attendance report (upload daily logs, that means to add new block in blockchain) and read attendance report (read daily logs, that means get IPFS file)

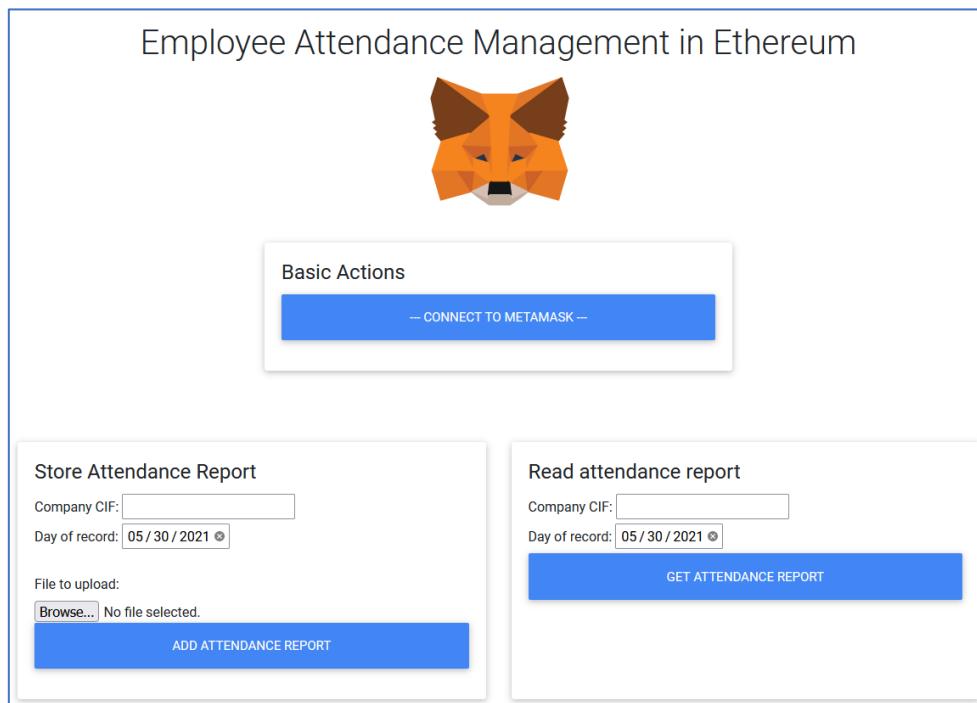


Figure 17. Dapp "Employee Attendance Management"

The web3 page can interact with the user with actions button, text input and date selection as specified in the following table. It is important to maintain the same id name inside the *index.html* file and inside the *contract.js* file, otherwise, the program can stop.

	action	type	id
button	Connect to MetaMask	listener on click	id="connectButton"
output	show text	show text	id="accounts"
button	Add daily log to blockchain	listener on click	id="append_log"
input	CIF company that upload	text	id="company_cif_write"
input	Date of upload logs	date	id="date_write"

	action	type	id
input	open directory to select file	file	id="daily_log_file"
button	Get daily log from blockchain	listener on click	id="get_log"
input	CIF company required logs	text	id="company_cif_read"
input	Date of required logs	date	id= "date_read"

Table. IV. DApp variables name and function

- 1) Update header, update the name of the web page to " DAppTimeLog " and check there is a reference to use "metamask.css"

```
<meta charset="UTF-8">
<title>DAppTimeLog</title>
<link href="metamask.css" rel="stylesheet">
```

- 2) Inside body, for first section change the name of section to " Basic Actions" and keep the code related to connect to MetaMask.

```
<!-- Part 1 Setting up Basic Actions and Status-->
<div class="card-body">
    <h4 class="card-title"> Basic Actions </h4>
    <button class="btn btn-primary btn-lg btn-block mb-3"
        id="connectButton" >
        Connect to MetaMask
    </button>
</div>
<div class="col-xl-4 col-lg-6 col-md-12 col-sm-12 col-12">
    <p class="info-text alert alert-success">
        Ether Accounts: <span id="accounts"></span>
    </p>
</div>
```

- 3) For second section, change the name to " Store Attendance Report", add the field for select date and upload file button with daily employee logs

```
<!-- Append Log => add attendance registries-->
<div class="card-body">
    <h4 class="card-title">
        Store Attendance Report
    </h4>
    <label>Company CIF: </label>
```

```
<input type="text" id="company_cif">
</input><br>
<label for="type_event">Day of record: </label>

<input type="date" id="registry_date" value="2021-05-30" </input><br>

<label>
    <br>File to upload: <br>
</label><br>

<input type="file" id="file_event"></input><br>
<button class="btn btn-primary btn-lg btn-block mb-3" id="addEvent">
    Add Attendance Report
</button>
</div>
```

- 4) For third section, change the name to " Read Attendance Report", add the field for select date and button to obtain report.

```
<!-- Get log => Reading attendance registries -->
<div class="card-body">
    <h4 class="card-title">
        Read Attendance Report
    </h4>
    <label>Company CIF: </label>
    <input type="text" id="company_cif">
    </input><br>
    <label>Day of record: </label>
    <input type="date" id="registry_date" value="2021-05-30" </input><br>

    <button class="btn btn-primary btn-lg btn-block mb-3" id="getFile">
        Get Attendance Report
    </button>
</div>
</section>
```

- 5) Add external scripts to html file. A script is a small piece of program that can add interactivity to website. For example, a script could generate a pop-up alert box message, or provide a dropdown menu. it is a good idea to keep that function in a single script file and then include this file in all the HTML pages. You can include a style sheet file into HTML document using <script> element.

The DApp example contains two scripts: "MetaMask-onboarding.js" used to check if MetaMask is installed, and "contract.js" used to configure all functions declared

in smart contract. Scripts with the defer attribute will prevent loaded event until the script has loaded and finished evaluating.

```
<script src="node_modules/@metamask/onboarding/dist/metamask-onboarding.bundle.js"
defer></script>

<script src="contract.js" defer></script>
```

- 6) Add IPFS script, as a technology to store data in decentralized servers. The JS-IPFS HTTP client [36] is a smaller library, implemented in JavaScript. This client library implements the IPFS Core API enabling applications to change between an embedded js-ipfs node and any remote IPFS node without having to change the code. To understand the IPFS implementation in JavaScript please see Alberto Lassa tutorial in youtube [37] in Spanish language.

```
<script src="https://cdn.jsdelivr.net/npm/ipfs-http-client/dist/index.min.js">
</script>
```

- 7) Add Ethereum script implementation [38] that creates JavaScript objects from any smart contract ABI, and connect to Ethereum nodes over JSON-RPC, INFURA, Etherscan, Alchemy or MetaMask to do transactions.

```
<script src="https://cdn.ethers.io/lib/ethers-5.0.umd.min.js" type="text/javascript">
</script>
```

To see the complete config file please see Annex E: DApp - index.html

3.6.2. CSS file structure

Cascading Style Sheets (css) CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts. It allows one to adapt the presentation to different types of devices, such as large screens, small screens, or printers. CSS is independent of HTML. The updated version on E2E Test DApp showed a image with a fox, just to remind this DApp need to interact with Ethereum. After several test, the MetaMask logo changed size according the length of title: short title showed small fox and large title showed a large fox. for this reason, it was modified the size of the image.

```
#mm-logo {  
    width: 20%;  
    justify-content: center;  
    align-items: center;  
}
```

To see the complete config file please see Annex F: DApp - metamask.css

3.6.3. JS file structure

DApp MetaMask tutorial give a very basic *contract.js* file, based mainly on a contract for digital coins (make deposit, execute withdraw). Using the concepts introduce by Alberto Lasa on his youtube channel, it will be create the JS code.

Steps:

- 1) In JavaScript, arrays and objects assigned to *const* variables. The first step is to define all variables, buttons and text defined on DApp web page layout.

```
// Address for web server (nodejs) that contains all packages  
const forwarderOrigin = 'http://localhost:9010';  
  
//Instance of the file to connect Metamask  
const onboarding = new MetamaskOnboarding({ forwarderOrigin });  
  
// Store Attendance Report Section  
const company_cif_write      = document.getElementById('company_cif_write');  
const date_write              = document.getElementById('date_write');  
const addReportButton        = document.getElementById('addReport');  
const company_cif_read       = document.getElementById('company_cif_read');  
const date_read               = document.getElementById('date_read');  
const getReportButton         = document.getElementById('getReport');
```

- 2) Save the Contract Account Address and ABI code, obtained during the Smart Contract deployment

```
// Smart contract where is deployed TIMELOG contract  
const ContractAccountAddress = '0x3Bb37250390a96f6dc99fEDD6386359C03CFABA4';  
  
const ABI = [  
    {
```

```

    "inputs": [ _company, date . . . ],
    "name": "append_log",
    "outputs": [ OK],
    "stateMutability": "nonpayable",
    "type": "function"
},
{
    "inputs": [ _company, date . . . ],
    "name": "get_log",
    "outputs": [ _log],
    "stateMutability": "view",
    "type": "function"
}
]

```

- 3) Check if the browser has MetaMask extension installed. Connect MetaMask account, since write on blockchain need to paid to miners for computation work. The function "onClickConnect" is asynchronous, and uses "await" instruction to indicate it cannot continue until the user select Ethereum account.

```

const initialize = () => {
    //Check if the browser has the Metamask extension installed
    const isMetaMaskInstalled = () => {
        const { ethereum } = window;
        return Boolean(ethereum && ethereum.isMetaMask);
    };

    //Show new MetaMask windows to select account to be connected
    const onClickConnect = async () => {
        try {
            await ethereum.request({ method: 'eth_requestAccounts' });
            onboardButton.disabled = true;
        }
    };

    //Check if MetaMask is installed
    const MetamaskClientCheck = async () => {
        if (!isMetaMaskInstalled()) {
            onboardButton.innerText = 'Click here to install MetaMask!';
            onboardButton.onclick = onClickInstall;
            onboardButton.disabled = false;
        }
    };
}

```

- 4) Convert the structure and content of the HTML document into an object model that can be used by programs, using Document Object Model (DOM). First instantiated

Ethers provider and define signer as authorized to do transactions that modified blockchain

```
// Load web page
window.addEventListener('DOMContentLoaded', initialize);

const provider = new ethers.providers.Web3Provider(window.ethereum);
const signer = provider.getSigner();
```

- 5) Interact with smart contract. The function "*myContract*" is defined a contract funcion of ABI and provider. The method "*connect*" returns a new instance of the Contract, but connected to providerOrSigner. By passing in a Provider, this will return a downgraded Contract which only has read-only access (i.e. constant calls). By passing in a Signer. this will return a Contract which will act on behalf of that signer. Now, the object is ready to deploy transactions.

```
const myContract = new ethers.Contract(ContractAccountAddress, ABI, provider);
const myContractWithSignature = myContract.connect(signer);
```

- 6) Declare IPFS server. Use the metadata information available in HTML, defined with id=daily_log_file and store in a new variable named "fileData". Define "ipfs" as a IPFS server node located in INFURA [39], instead to deploy an IPFS node by ourselves. Check *ipfs-http-client* is defined in the script of HTML file.

```
var fileData = '';

const fileSelector = document.getElementById('daily_log_file');
fileSelector.addEventListener('change', (evento) => {
    fileData = evento.target.files[0];
    console.log(fileData);
});

const ipfs = window.IpfsHttpClient({ host: 'ipfs.infura.io', port: 5001 })
```

- 7) Add attendance report to IPFS server (function: write blockchain). First are collected the attributes declared: "*company_cif_write*", "*date_write*". It is created an auxiliar variable named "fileData" followed by "await" instruction. The auxiliar variable will store the hash code used to store the file in the IPFS.

```
addEventButton.onclick = async () => {
    try {
        let company = company_cif_write.value;
```

```

let date_write = date_write.value;
let result = await ipfs.add(fileData);
let _url = "http://ipfs.io/ipfs/" + result.cid.string;
// Upload the event to Ethereum
console.log(await myContractWithSignature.addnewevent(company, date_write, _url));
} catch (error) {
    console.error(error);
}
}
}

```

- 8) Get attendance report from IPFS (function: read blockchain). This function will be launched when the user click on "Get Attendance Report" button. It will appear a icon, and as the user click on it, it will be redirected to the corresponding IPFS stored file.

```

getReportButton.onclick = async () => {
    try {
        let company = company_cif_read.value;
        let date_read = date_read.value;
        let result = await myContract.append_log(company, date_read);
        //Se muestran en una pequeña tabla con su respectivo hipervínculo a la dirección en IPFS donde se encuentren alojados
        //En JS se puede "inyectar" código HTML en cualquier elemento teniendo únicamente su id
        document.getElementById("contenedor").innerHTML = '<table>' +
            + '<tr><td> Primer archivo: </td><td><a href=' + result[1] + '">' + result[0] + '</a>' +
            + '</table>';
    }
}

```

To see complete code please see Annex G: DApp - contract.js

- 9) Execute and run all programs. Open cdm terminal. To install all dependencies needed for the project, run "npm install" command. Once the installation is finished, it will be created a new folder named "node_modules" with several files inside. Also install the package to interact with Smart contract and IPFS

```

> npm install
> npm install --save ethers
> npm install ipfs-core

```

To check the procedure to fix errors, please see Annex H: DApp - debugging code in VSC (Visual Studio Code)

4 Deployment and Test Results

4.1. Connect IoT device to network

In this step, it is needed to power on the NodeMCU. It can be used USB cable and connected to AC power, or it can be used a power bank.

The LED screen shows the following:

- Adafruit logo
- Wi-Fi identification (Vodafone E35B)
- connected successful
- fingerprint indicating waiting
- Welcome message, with the name of the employee that enter in the company
- Good By message, with the name of the employee is leaving the company.



Figure 18. Turn on IoT Device and connect to network

4.2. Open IoT Edge Web page

The IoT Edge is the main interface between the system administrator in the company and the IoT device in charge of capture the fingerprint of all employees. The IoT Edge server is running on a computer, as localhost. To start the service:

- execute XAMPP program. In windows: start - programs - XAMPP - run,
- click on START button for modules: Apache and MySQL
- Then, open any browser, like Firefox, Chrome or Explorer and open the server at <http://127.0.0.1/iotedge/index.php>.
- Login credentials are:

E-mail: admin@gmail.com

password: 123

4. DEPLOYMENT AND TEST RESULTS



Figure 19. run Apache and MySQL server on XAMPP program

Devices

The Attendance device can work in two modes: Enrollment, to capture employee fingerprint, and store in the system, only one time, and by default, Attendance mode: to capture employee fingerprint every time they enter or leave the company. To select the device mode, click on "Devices" tab and then, click on Enrollment or Attendance.

The screenshot shows the IoT Edge web server interface. The 'Devices' tab is highlighted with a red box in the top navigation bar. Below the header, it says 'ADD A NEW DEVICE/UPDATE/REMOVE/ENABLE/DISABLE'. The main content area shows a table titled 'Your Devices:' with two entries:

Dev.Name	Dev.Location	Dev.UID	Dev.Date	Dev.Mode	Dev.Config
NodeMCU2	SimulatedDevice	70f09240d6544562	2021-06-07	Enrollment Attendance	
NodeMCU1	Madrid_Building1	ea16b938f7e73a9d	2021-06-07	Enrollment Attendance	

At the bottom left, it shows the URL: 127.0.0.1/iotedge/devices.php

Figure 20. Device working mode

Users

Under the "Users" tab is showed the total of employees in the company. Check company employees list

IoT Edge web server with database

Users
Manage Users
Users Log
Devices
Admin
Log Out

HERE ARE ALL THE USERS

ID NAME	SOCIAL SECURITY NUMBER	GENDER	FINGER ID	DATE	WORK PLACE
13 Galileo Galilei	311041269532	Male	50	2021-06-10	Madrid_Building1
12 Cristobal Colon	351069869699	Male	23	2021-06-10	Madrid_Building1
11 Maria Garcia	381079839655	Female	30	2021-06-07	Madrid_Building1

Figure 21. List of all employees in a company, with

Manage Users

The first action consists to create the database with the employee information. This task is executed only one time, using the IoT device in enroll mode. The fingerprint sensor is only capable to store finger ID and fingerprint image. Thanks to IoT Edge, this information can be enriched with: Name, Social Security number, department, and date of enroll. To add new employee data:

- Set the device on "Enrollment" mode under "Devices" tab
- Click "Manage Users" tab
- Add the location of fingerprint sensor (Building Name)
- Enter Fingerprint ID
- The device will receive the Finger ID, the led will show an scanning image and the user can put the finger, two times. The scanner will compare the pictures, and confirm image correctly stored



- After storing the image, the device will send a confirmation to Edge node and the status will change to "added" to indicate successful enrollment

- Click on Finger ID and fill User Information: name, Social Security Number and gender. Click "Add". The system will show the message: A new User has been added.

FINGER .ID	NAME	GENDER	SOCIAL-SEC NUMBER.	DATE	WORK PLACE	FINGERPRINT STATUS
50	Galileo Galilei	Male	311041269532	2021-06-10	Madrid_Building1	Added
23	Cristobal Colon	Male	351069869699	2021-06-10	Madrid_Building1	Added
30	Maria Garcia	Female	381079839655	2021-06-07	Madrid_Building1	Added

Figure 22. New fingerprint added

Users Log

User Log tab shows the attendance report. It can be filtered by employee, by day or by location. Once the system administrator selects a filter, the result can be exported, in excel format. To obtain the daily report:

- Click "Users Log" tab
- select a day, and select Time-in, between 12:00 am and 11:59 pm
- Click on filter and review the data. Filter is mandatory, because excel file will save the report according "from date"
- Click on Export button
- A file will be generated with a name corresponding to selected date,

ID	NAME	EMPLOYEE NUMBER	FINGERPRINT ID	DEVICE DEP	DATE	TIME IN	TIME OUT
5	Galileo Galilei	311041269532	30	Madrid_Building1	2021-06-10	14:11:44	14:11:50
4	Maria Garcia	381079839655	50	Madrid_Building1	2021-06-10	13:42:45	13:42:50
3	Cristobal Colon	351069869699	23	Madrid_Building1	2021-06-10	13:16:39	13:42:54
2	Maria Garcia	351069869699	50	Madrid_Building1	2021-06-10	14:09:32	14:14:56

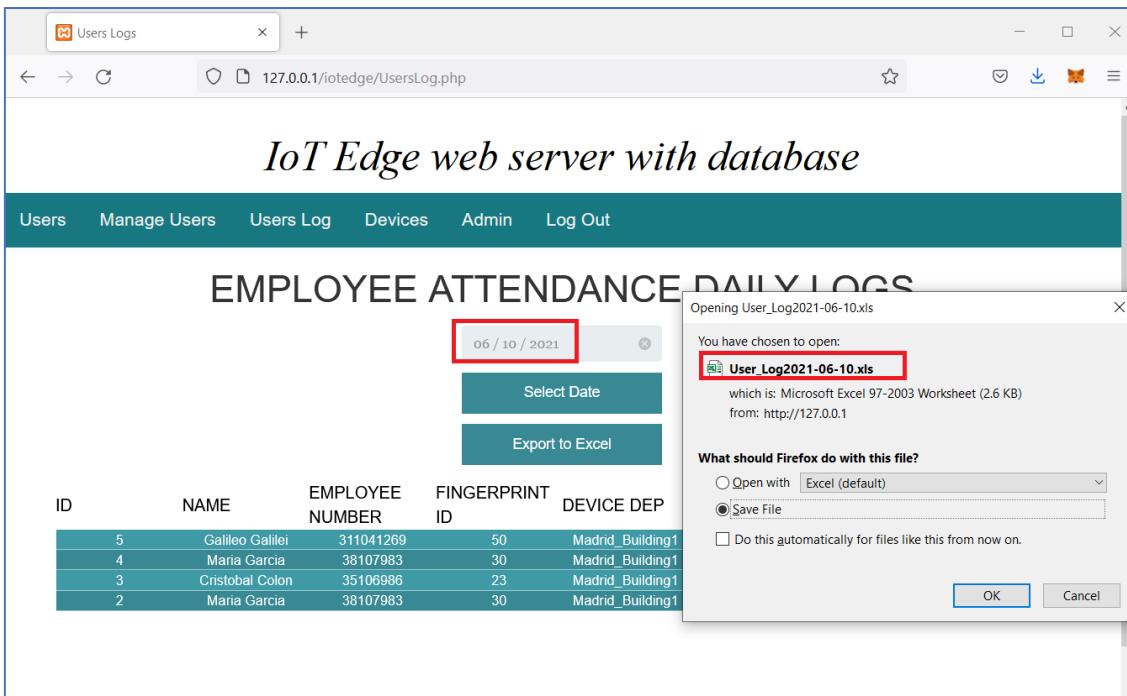


Figure 23. Export daily attendance report to excel

- see file on download folder. Rename file, or modified data is possible at this stage.

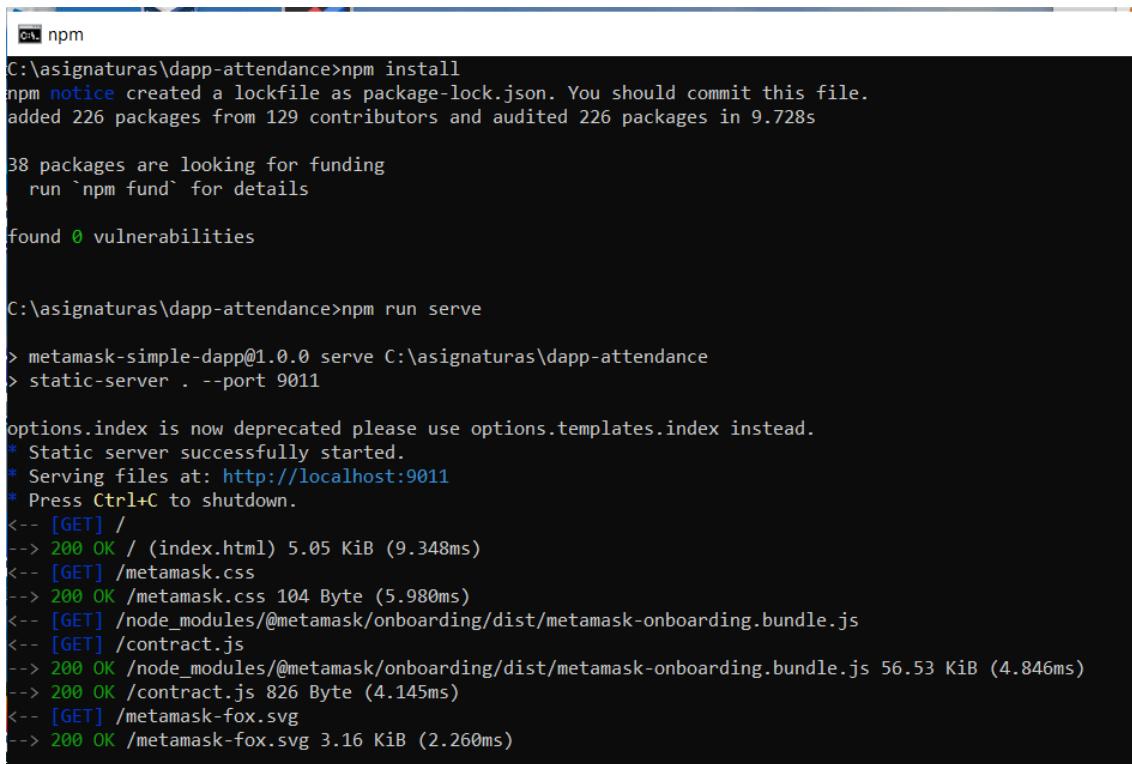
User_Log2021-06-10-1.xls - Solo lectura - Excel								
JEIDI PADRON NUÑEZ								
Archivo Inicio Insertar Disposición de página Fórmulas Datos Revisar Vista Ayuda								
Autoguardado								
A1	B	C	D	E	F	G	H	
1	ID	Name	Employee Number	Fingerprint ID	Device Dep	Date log	Time In	Time Out
2	5	Galileo Galilei	311041269	50	Madrid_Building1	10/06/2021	14:11:44	14:11:50
3	4	Maria Garcia	38107983	30	Madrid_Building1	10/06/2021	13:42:45	13:42:50
4	3	Cristobal Colon	35106986	23	Madrid_Building1	10/06/2021	13:16:39	13:42:54
5	2	Maria Garcia	38107983	30	Madrid_Building1	10/06/2021	14:09:32	14:14:56
6								
7								
User_Log2021-06-10-1								
Listo								

4.3. Connect DApp to Blockchain

The Distribute Application DApp allows the system administrator of a company to store the daily attendance report and store it in a blockchain, according the instructions give by the Smart contract. In case of problems, check Annex I: DApp - Debugging code in VSC (Visual Studio Code). Initialize start the service:

- 1) Start DApp service. Open a CMD console in the same directory where is the code and write the command:

```
C:\asignaturas\dAppTimeLog> npm run serve
```



```
C:\asignaturas\dapp-attendance>npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
added 226 packages from 129 contributors and audited 226 packages in 9.728s

38 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\asignaturas\dapp-attendance>npm run serve

> metamask-simple-dapp@1.0.0 serve C:\asignaturas\dapp-attendance
> static-server . --port 9011

options.index is now deprecated please use options.templates.index instead.
* Static server successfully started.
* Serving files at: http://localhost:9011
* Press Ctrl+C to shutdown.

<-- [GET] /
--> 200 OK / (index.html) 5.05 KiB (9.348ms)
<-- [GET] /metamask.css
--> 200 OK /metamask.css 104 Byte (5.980ms)
<-- [GET] /node_modules/@metamask/onboarding/dist/metamask-onboarding.bundle.js
<-- [GET] /contract.js
--> 200 OK /node_modules/@metamask/onboarding/dist/metamask-onboarding.bundle.js 56.53 KiB (4.846ms)
--> 200 OK /contract.js 826 Byte (4.145ms)
<-- [GET] /metamask-fox.svg
--> 200 OK /metamask-fox.svg 3.16 KiB (2.260ms)
```

- 2) Then, open any browser, like Firefox, Chrome or Explorer and open the server at <http://localhost:9011>
- 3) In case the browser do not has the MetaMask extension, DApp will show a message: "Click here to install MetaMask". A new window will be open, to add MetaMask extension in the browser. Click on "Add to Chrome". The Dapp will show the message "Onboarding in progress" until the extension is installed.
- 4) Follow the instructions indicated in the chapter 3 of this document, about MetaMask installation.

- 5) Once browser has MetaMask extension installed, the DApp will show a message: "Connect"

Send daily log to blockchain

The company is responsible for upload the daily report. In case of no working day, is still possible to upload previous report. To send the report, just fill de company CIF (only numbers), the date of report, click on "Browse..." to select the file and click on "Add Attendance Report"

Get daily log from blockchain

5

Budget

5.1. Budget and time schedule.

Estas recomendaciones en cuanto a la estructura del Trabajo Fin de Máster tienen por objetivo dar bla bla

Tasks The prototype will be carried out in different phases:

Select and buy IoT platform, IDE development: Arduino IoT board, fingerprint sensor and LED screen for real data generation. Arduino already has a library for fingerprint sensors.

Configure and test the Arduino board, check and test Wi-Fi connection, configure LED screen

Configure and test the fingerprint sensor and connect it to Arduino

Develop a traditional web page (client-server topology) to keep a data history of enter / exit time of employees in a company inside a traditional database. Export daily report

Develop IPFS server for daily data storing, write a smart contact using solidity language. The upload of daily record is done manually by authorized people

Configure a Distributed app (dApp) in Javascript [6] to take the daily record and store it using blockchain network

Generate Smart contract with the daily information, using dApp, Ethereum blockchain and IPFS peer-to-peer technologies.

Test end to end system and make adjust (if any)

Write final report and conclusion

Prepare presentation

Materials to be used: Arduino board, fingerprint sensor, LED screen, Ethereum blockchain network, student laptop

Bill of Materials

The following are the components required to make **IoT Based Biometric Fingerprint Attendance System**. All the components were purchased from Amazon Spain. The purchase links are given below.

N.	COMPONENTS	DESCRIPTION	QTY	PRICE	amazon
1	NodeMCU	ESP8266-12E Board	1	7,79€	https://amzn.to/2Q7b5TG
2	Fingerprint Sensor	R307 Module	1	16.89 €	https://amzn.to/3muwQZO
3	OLED Display	0.96" SSD1306 I2C OLED Display	1	6,79€	https://amzn.to/3d19h7C
4	Connecting Wires	Jumper Wires	10	-	
5	Breadboard	Proto Distribution Shield Connecting Blocks	2	7,00 €	https://amzn.to/3aoWZnX
6	Arduino	Donation for Arduino IDE	1	3€	
7	Web server	Laptop Lenovo	1	600 €	
8	Developer	Engineer	1	10.000 €	Annual gross salary = 24K Payment for 5 month = 10K

APÉNDICE B. PRESUPUESTO ECONÓMICO

111

Actividad	Número de horas	Coste
Comprensión de la documentación	70	1.260€
Formación con el simulador <i>Wireless InSite</i>	110	1.980€
Estudios previos realizados con el simulador	200	3.600 €
Simulaciones óptimas	25	450€
Implementación del postprocesado en Matlab	50	900€
Formación con Python	40	720€
Implementación del postprocesado en Python	55	990€
Redacción de la memoria	200	3.600€
Total	750	13.500€

Tabla B.3: Coste de mano de obra

Time-schedule: the work uses 40 hours a week, total eight weeks (320 hours)

5. BUDGET

	W1 May 17-21	W2 May 24-28	W3 Jun 1-4	W4 Jun 7-11	W5 Jun 14-18	W6 Jun 21-25	W7 Jun 28-31	W8 Jul 5-9	W9 Jul 12-16
Task 1: buy material, configure Arduino, Wifi									
Task 2: configure Led screen and Wi-Fi									
Task 3: configure fingerprint sensor									
Task 4: develop wep page (client server)									
Task 5: develop IPFS server, write smart contract									
Task 6: develop dApps with javascript									
Task 7: generate smart contract, Ethereum									
Task 8: testing and correct (if any)									
Task 9: write final report									
Task 10: prepare presentation									

6

Conclusions

6.1. General conclusions.

Estas recomendaciones en cuanto a la estructura del Trabajo Fin de Máster tienen por objetivo dar bla bla

((PARAFRASEAR)) Blockchain technology has multiple disadvantages that have decreased its adoption rate. An often-overlooked challenge is that the technology is initially difficult to understand and adopt. Getting people to use blockchain applications is a difficult task, which brings disadvantages as people believe it's an unnecessary precaution for a network.

This allows for a fully trustworthy system that we can trust will not be manipulated. There is no more "middleman" or centralized authority that holds all of the information. Every node on the network holds a copy of the ledger which allows for confirmations, for validity, and for a truly trustless system to survive. Blockchains are fairly simple to bootstrap once they are implemented. Furthermore, it is a new way to envision technology and the next frontier of Internet.

6.2. Future works

f 2. Cloud-based attendance monitoring facility for remote working employee

It is an essential feature for monitoring the attendance of employees who prefer remote working facilities. With a flexible cloud-based time tracking feature, employers can easily track the working hours of remotely working employees from anywhere. They can also enable GPS tracking for location monitoring.

f 3. Integration with payroll processing

For paycheck generation, you need to first gather some details like employee attendance, leaves taken and overtime payment, etc. The employees who work extra hours or overtime usually receive extra pay. But, doing this task manually is difficult and time-consuming. To simplify the task, integration with payroll processing is needed. Employers can easily maintain an accurate record of employee salary, bonus, and other details needed for paycheck generation. The tracked time details of employees are recorded into the system that is used for the automatic generation of paychecks.

6. CONCLUSIONS

f

7 References

7.1. References

- [1] Royal Decret- Law 8/2019 on urgent measures of social protection and the fight against job insecurity in the workday (online)
<https://www.boe.es/boe/dias/2019/03/12/pdfs/BOE-A-2019-3481.pdf>
- [2] Ministerio del Trabajo y Economia Social (mites). Frequently asked questions about working day registry. (online)
<https://www.mites.gob.es/ficheros/ministerio/GuiaRegistroJornada.pdf>
- [3] Biometric attendance system - International Research Journal of Engineering and Technology (IRJET) (online) <https://www.irjet.net/archives/V6/i4/IRJET-V6I4866.pdf>
- [4] Benet, Juan: "IPFS - Content Addressed, Versioned, P2P File System" (arxiv.org)
<https://arxiv.org/pdf/1407.3561.pdf>
- [5] Andreas M. Antonopoulos, Gavin Wood (2019) Ethereumbook: Mastering Ethereum, (online)
<https://github.com/ethereumbook/ethereumbook/blob/develop/book.asciidoc>
- [6] Ethereum.org - Anatomy of smart contracts | (online)
<https://ethereum.org/en/developers/docs/smart-contracts/anatomy/>
- [7] PaymentsSource - Smartphone design is the next challenge for biometrics
<https://www.paymentssource.com/payments/opinion/smartphone-design-is-the-next-challenge-for-biometrics>
- [8] Adafruit Optical Fingerprint Sensor (online) <https://learn.adafruit.com/adafruit-optical-fingerprint-sensor>
- [9] Adafruit Learning System - optical fingerprint sensor (online) <https://cdn-learn.adafruit.com/downloads/pdf/adafruit-optical-fingerprint-sensor.pdf>
- [10] JM-101 Optical Fingerprint Module User manual (online)
<https://3v3.com.ua/data/files/JM-101-OPTICAL-FINGERPRINT-MODULE-USER-MANUAL-V1.8A.PDF>
- [11] AZ Delivery - OLED display 1.3 inch User Manual (online) <https://www.az-delivery.de/a/downloads/-/e68ccc911afe63a6/403e1e3183023714>

- [12] AZ Delivery - NodeMCU Lua Lolin V3 Module ESP8266 ESP-12F WIFI Development Board, User Manual (online) <https://www.az-delivery.de/a/downloads/-/52395f57d55a704f/3abbabde78085841>
- [13] Flaticon.com - Free Vector Icons and Stickers - PNG, SVG, EPS, PSD and CSS (online) <https://www.flaticon.es/autores/prosymbols>
- [14] ethereum.org - Introduction to decentralized application (online) <https://ethereum.org/en/developers/docs/dapps/>
- [15] Merunas Grincalaitis "Ultimate Guide to Convert a Web App To a Decentralized App Dapp". Ethereum Developers in Medium blog page. March 2018 (online) <https://medium.com/ethereum-developers/ultimate-guide-to-convert-a-web-app-to-a-decentralized-app-dapp-f6112a079509>
- [16] CoinMarketCap - Ethereum price today, ETH live marketcap, chart, and info (online) <https://coinmarketcap.com/currencies/ethereum/>
- [17] Quora - What is the blocktime of Ethereum? (online) <https://www.quora.com/What-is-the-blocktime-of-Ethereum>
- [18] Nick Szabo blog page (online) <http://unenumerated.blogspot.com/>
- [19] DevTeam.Space - How to Deploy Contract Ethereum? (online) <https://www.devteam.space/blog/how-to-deploy-smart-contract-on-ethereum/>
- [20] MetaMask Create A Simple Dapp (online) <https://docs.metamask.io/guide/create-dapp.html>
- [21] GeeksforGeeks: Difference between HTTP and IPFS (2019) (online) <https://www.geeksforgeeks.org/difference-between-http-and-ipfs/>
- [22] ichi.pro - Cree su primer Dapp con Web3.js (online) <https://ichi.pro/es/cree-su-primer-dapp-con-web3.js-169818826304097>
- [23] Arduino - Software Download (online) <https://www.arduino.cc/en/software>
- [24] Nanjing Qinheng Microelectronics - Windows USB controller CH341SER (online) http://www.wch.cn/download/CH341SER_ZIP.html
- [25] GitHub - nodemcu flasher Win64 Release at master (online) <https://github.com/nodemcu/nodemcu-flasher/tree/master/Win64/Release>

- [26] Adafruit Learning System: Monochrome OLED Breakouts (online) <https://learn.adafruit.com/monochrome-oled-breakouts/arduino-library-and-examples>
- [27] Adafruit Learning System: Optical Fingerprint Sensor - Downloads <https://learn.adafruit.com/adafruit-optical-fingerprint-sensor/downloads>
- [28] Apachefriends.org - XAMPP Installers and Downloads (online) <https://www.apachefriends.org/index.html>
- [29] Install MetaMask for your browser. Download for browser (Chrome, Firefox, Brave, Edge), iOS and Android. (online). <https://metamask.io/download.html>
- [30] Test Ether Faucet (metamask.io) (online). <https://faucet.metamask.io/>
- [31] soliditylang.org - Contracts , Solidity 0.8.8 documentation (online) <https://docs.soliditylang.org/en/latest/contracts.html>
- [32] Vyper — Vyper documentation <https://vyper.readthedocs.io/en/latest/>
- [33] Remix - Ethereum IDE (online) <https://remix.ethereum.org>
- [34] Visual Studio Code - Code Editing. Redefined. Download for Windows, macOS, Linux (online) <https://code.visualstudio.com/>
- [35] Node.js JavaScript runtime. Download for Windows (x64) (online) <https://nodejs.org/en/>
- [36] IPFS Documentation - JS-IPFS (online) <https://docs.ipfs.io/reference/js/api/>
- [37] Alberto Lasa - Youtube channel - IPFS - Subir y Descargar Archivos, Regular File API (online) <https://www.youtube.com/watch?v=J8YGaKD0wd8>
- [38] The Ethers Project (npmjs.com) ethers - npm (online) <https://www.npmjs.com/package/ethers>
- [39] Infura IPFS API - Scalable and distributed IPFS storage infrastructure for your application (online) <https://infura.io/product/ipfs>

- [] Decentralized Applications Architecture: Back End, Security and Design Patterns (freecodecamp.org) <https://www.freecodecamp.org/news/how-to-design-a-secure-backend-for-your-decentralized-application-9541b5d8bddb/>

7.2. Repositories

- [40] GitHub InfinityWorldHI - Biometric-attendance-system-V1.0 (online)
<https://github.com/InfinityWorldHI/Biometric-attendance-system-V1.0>
- [41] GitHub - BboyAkers/simple-dapp-tutorial: A tutorial based on the MetaMask E2E Test Dapp(decentralized app) (online) <https://github.com/BboyAkers/simple-dapp-tutorial>

8 ANNEX

Annex A - Glossary

Term	Definition	Source
Blockchain	A decentralized public ledger that records all transactions within a given network.	
Consensus algorithm	A consensus algorithm allows nodes on the network to trust that a given piece of data is valid and that it has been synchronized with all other nodes.	
DAGs	Directed acyclic graphs.	

jfyj

DApp	Decentralized Application
DLT	Distributed Ledger Technology
DOM	Document Object Model ()
EVM	Ethereum Virtual Machine
ERC	Ethereum Requests for Comments
ETH	Ether (virtual coin, cryptocurrency)
GPIO	general-purpose input/output
Infura	It is a set of tools for anyone to create an application that connects to the Ethereum blockchain. It interacts with the Ethereum blockchain and runs nodes on behalf of its users.
HTML	HyperText Markup Language
HTTP	HiperText Transfer Protocol ()
MCU	Micro Controller Unit
npm	Node Package Manager
OLED	"organic light emitting diode."
P2P	
PoC	Proof of Concept ()
rETH	Ropsten ETH (test ETH, no value)
RFID	(radio frequency identification

UMD	Universal Module Definition
UHF	
USD	United States Dollar (currency of the USA)
VSC	Visual Studio Code

1 Annex B - IoT Device Arduino Code

```

2 MSc in Internet of Things
3 Master Thesis Project
4 Biometric work attendance management and logging with a blockchain system
5 Author: Jeidi Padron Nuñez
6 Date: July 2021
7 version 1.0
8
9 -----
10 This code was based on code created by Electronics Tech youtube channel for
11 the Biometric attendance project with NodeMCU, available at
12 https://github.com/InfinityWorldHI/Biometric-attendance-system-V1.0
13
14 Test code provided by vendor
15 Adafruit      - Fingerprint model JM-101B
16 AZ delivery - OLED I2C 128 x 64 pixel display for Arduino and Raspberry Pi
17 AZ delivery - NodeMCU V3 Lolin
18 https://www.az-delivery.de/en
19
20 -----
21
22
23 /////////////////
24 // LIBRARIES
25 /////////////////
26
27 #include <SPI.h>
28 #include <Wire.h>
29 #include <WiFiClient.h>
30 #include <ESP8266WiFi.h>
31 #include <SoftwareSerial.h>
32 #include <ESP8266WebServer.h>
33 #include <ESP8266HTTPClient.h>
34 #include <Adafruit_GFX.h>           //https://github.com/adafruit/Adafruit-GFX-Library
35 #include <Adafruit_SSD1306.h>
36 //https://github.com/adafruit/Adafruit\_SSD1306
37 #include <Adafruit_Fingerprint.h>      //https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library
38 #include <SimpleTimer.h>
39 //https://github.com/jfturcot/SimpleTimer
40
41
42
43
44 /////////////////
45 // Fingerprint scanner Pins
46 /////////////////
47 #define Finger_Rx 14                  //D5

```

```

48 #define Finger_Tx 12           //D6
49 SimpleTimer timer;
50 SoftwareSerial mySerial(Finger_Rx, Finger_Tx);
51 Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
52
53
54 ///////////////////////////////////////////////////////////////////
55 //    SSD1306 display - I2C Pins (22 SCL, 21 SDA)
56 ///////////////////////////////////////////////////////////////////
57 #define SCREEN_WIDTH 128          // OLED display width, in pixels
58 #define SCREEN_HEIGHT 64           // OLED display height, in pixels
59 #define OLED_RESET 0              // Reset pin # (or -1 if sharing
60 Arduino reset pin)
61 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
62
63
64 ///////////////////////////////////////////////////////////////////
65 //    Declaration of Wi-Fi credentials.
66 ///////////////////////////////////////////////////////////////////
67 const char *ssid = "Jeidi_AP";           //Enter your WiFi values
68 const char *password = "12345678";
69
70
71 ///////////////////////////////////////////////////////////////////
72 //    Declaration of IoT Device ID
73 ///////////////////////////////////////////////////////////////////
74 const char* device_token = "6b07332bc7f1c0b9";
75
76
77 ///////////////////////////////////////////////////////////////////
78 //    Declaration of IoT Edge Web Server and Database.
79 ///////////////////////////////////////////////////////////////////
80 String getData, Link;
81 String URL = "http://192.168.0.22/iotedge/getdata.php"; //computer IP or the server
82 domain and folder
83 int FingerID = 0, t1, t2;                      // The Fingerprint ID from the
84 scanner
85 bool device_Mode = false;                     // Default Mode Enrollment
86 bool firstConnect = false;
87 uint8_t id;
88 unsigned long previousMillis = 0;
89
90
91 //    Biometric Icons
92 ///////////////////////////////////////////////////////////////////
93 #define Wifi_start_width 54
94 #define Wifi_start_height 49
95 const uint8_t PROGMEM Wifi_start_bits[] = {
96     0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

```

```

97 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
98 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
99 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
100 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
101 ,0x00,0x00,0x1f,0xf0,0x00,0x00,0x00
102 ,0x00,0x03,0xff,0xff,0x80,0x00,0x00
103 ,0x00,0x1f,0xf0,0x1f,0xf0,0x00,0x00
104 ,0x00,0x7e,0x00,0x00,0xfc,0x00,0x00
105 ,0x01,0xf0,0x00,0x00,0x1f,0x00,0x00
106 ,0x03,0xc0,0x00,0x00,0x07,0xc0,0x00
107 ,0x0f,0x00,0x00,0x00,0x01,0xe0,0x00
108 ,0x1c,0x00,0x00,0x00,0x00,0x70,0x00
109 ,0x38,0x00,0x07,0xc0,0x00,0x38,0x00
110 ,0x70,0x00,0xff,0xfe,0x00,0x1e,0x00
111 ,0xe0,0x03,0xfc,0x7f,0xc0,0x0e,0x00
112 ,0x00,0x1f,0x80,0x03,0xf0,0x00,0x00
113 ,0x00,0x3c,0x00,0x00,0x78,0x00,0x00
114 ,0x00,0xf0,0x00,0x00,0x1c,0x00,0x00
115 ,0x01,0xe0,0x00,0x00,0x0c,0x00,0x00
116 ,0x03,0x80,0x00,0x00,0x00,0x00,0x00
117 ,0x03,0x00,0x00,0x00,0x00,0x00,0x00
118 ,0x00,0x00,0x3f,0xf8,0x07,0x1e,0x00
119 ,0x00,0x00,0xff,0xfe,0x1f,0xbf,0x80
120 ,0x00,0x03,0xe0,0x04,0x7f,0xff,0xc0
121 ,0x00,0x07,0x80,0x00,0xff,0xff,0xe0
122 ,0x00,0x0e,0x00,0x00,0xff,0xff,0xe0
123 ,0x00,0x0c,0x00,0x00,0x7f,0xff,0xc0
124 ,0x00,0x00,0x00,0x00,0xfe,0x07,0xe0
125 ,0x00,0x00,0x00,0x03,0xf8,0x03,0xf8
126 ,0x00,0x00,0x07,0xe7,0xf9,0xf1,0xfc
127 ,0x00,0x00,0x1f,0xe7,0xf1,0xf9,0xfc
128 ,0x00,0x00,0x1f,0xe7,0xf3,0xf9,0xfc
129 ,0x00,0x00,0x3f,0xe7,0xf3,0xf9,0xfc
130 ,0x00,0x00,0x3f,0xe7,0xf1,0xf1,0xfc
131 ,0x00,0x00,0x3f,0xe3,0xf8,0xe3,0xfc
132 ,0x00,0x00,0x3f,0xf3,0xfc,0x07,0xf8
133 ,0x00,0x00,0x1f,0xf0,0x7f,0x0f,0xc0
134 ,0x00,0x00,0x0f,0xe0,0x7f,0xff,0xe0
135 ,0x00,0x00,0x07,0xc0,0xff,0xff,0xe0
136 ,0x00,0x00,0x00,0x00,0x7f,0xff,0xe0
137 ,0x00,0x00,0x00,0x00,0x3f,0xff,0x80
138 ,0x00,0x00,0x00,0x00,0x0f,0xbf,0x00
139 ,0x00,0x00,0x00,0x00,0x03,0x18,0x00
140 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
141 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
142 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
143 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
144 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00
145 };

```

```

146 #define Wifi_connected_width 63
147 #define Wifi_connected_height 49
148 const uint8_t PROGMEM Wifi_connected_bits[] = {
149   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
150 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
151 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
152 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
153 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
154 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
155 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
156 ,0x00,0x00,0x03,0xff,0xff,0x80,0x00,0x00
157 ,0x00,0x00,0x3f,0xff,0xff,0xf8,0x00,0x00
158 ,0x00,0x01,0xff,0xff,0xff,0x00,0x00,0x00
159 ,0x00,0x0f,0xff,0xff,0xff,0xe0,0x00
160 ,0x00,0x3f,0xff,0xc0,0x07,0xff,0xf8,0x00
161 ,0x00,0xff,0xf8,0x00,0x00,0x3f,0xfe,0x00
162 ,0x03,0xff,0x80,0x00,0x00,0x03,0xff,0x80
163 ,0x07,0xfe,0x00,0x00,0x00,0x00,0xff,0xc0
164 ,0x1f,0xf8,0x00,0x00,0x00,0x00,0x3f,0xf0
165 ,0x3f,0xe0,0x01,0xff,0xff,0x00,0x0f,0xf8
166 ,0x7f,0x80,0x0f,0xff,0xe0,0x03,0xfc
167 ,0xff,0x00,0x7f,0xff,0xff,0xfc,0x01,0xfe
168 ,0xfc,0x01,0xff,0xff,0xff,0x00,0x7e
169 ,0x78,0x07,0xff,0xc0,0x07,0xff,0xc0,0x3c
170 ,0x00,0x0f,0xfc,0x00,0x00,0x7f,0xe0,0x00
171 ,0x00,0x1f,0xf0,0x00,0x00,0x1f,0xf0,0x00
172 ,0x00,0x3f,0xc0,0x00,0x00,0x07,0xf8,0x00
173 ,0x00,0x7f,0x00,0x01,0x00,0xfc,0x00
174 ,0x00,0x7e,0x00,0x7f,0xfc,0x00,0xfc,0x00
175 ,0x00,0x3c,0x03,0xff,0xff,0x80,0x78,0x00
176 ,0x00,0x00,0x07,0xff,0xff,0xc0,0x00,0x00
177 ,0x00,0x00,0x1f,0xff,0xff,0xf0,0x00,0x00
178 ,0x00,0x00,0x3f,0xf0,0x1f,0xf8,0x00,0x00
179 ,0x00,0x00,0x3f,0x80,0x03,0xf8,0x00,0x00
180 ,0x00,0x00,0x3f,0x00,0x01,0xf8,0x00,0x00
181 ,0x00,0x00,0x1c,0x00,0x00,0x70,0x00,0x00
182 ,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00
183 ,0x00,0x00,0x00,0x0f,0xe0,0x00,0x00,0x00
184 ,0x00,0x00,0x00,0x1f,0xf0,0x00,0x00,0x00
185 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00
186 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00
187 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00
188 ,0x00,0x00,0x00,0x3f,0xf8,0x00,0x00,0x00
189 ,0x00,0x00,0x00,0x1f,0xf0,0x00,0x00,0x00
190 ,0x00,0x00,0x00,0x0f,0xe0,0x00,0x00,0x00
191 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
192 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
193 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
194 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

```

```

195 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
196 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
197 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
198 };
199 #define FinPr_start_width 64
200 #define FinPr_start_height 64
201 const uint8_t PROGMEM FinPr_start_bits[] = {
202     0x00,0x00,0x00,0x1f,0xe0,0x00,0x00,0x00
203 ,0x00,0x00,0x01,0xff,0xfe,0x00,0x00,0x00
204 ,0x00,0x00,0x03,0xff,0xff,0x80,0x00,0x00
205 ,0x00,0x00,0x0f,0xc0,0x0f,0xe0,0x00,0x00
206 ,0x00,0x00,0x1f,0x00,0x01,0xf8,0x00,0x00
207 ,0x00,0x00,0x3c,0x00,0x00,0x7c,0x00,0x00
208 ,0x00,0x00,0x78,0x00,0x00,0x3e,0x00,0x00
209 ,0x00,0x00,0xf0,0x3f,0xf8,0x0f,0x00,0x00
210 ,0x00,0x01,0xe0,0xff,0xfe,0x07,0x80,0x00
211 ,0x00,0x03,0xc3,0xff,0xff,0x03,0x80,0x00
212 ,0x00,0x03,0x87,0xc0,0x07,0xc3,0xc0,0x00
213 ,0x00,0x07,0x0f,0x00,0x03,0xe1,0xc0,0x00
214 ,0x00,0x0f,0xe0,0x00,0x00,0xe0,0xe0,0x00
215 ,0x00,0x0e,0x1c,0x00,0x00,0xf0,0xe0,0x00
216 ,0x00,0x0c,0x3c,0x1f,0xe0,0x70,0xe0,0x00
217 ,0x00,0x00,0x38,0x3f,0xf0,0x38,0x70,0x00
218 ,0x00,0x00,0x78,0x78,0xf8,0x38,0x70,0x00
219 ,0x00,0x00,0x70,0x70,0x3c,0x18,0x70,0x00
220 ,0x00,0x00,0xe0,0xe0,0x1e,0x1c,0x70,0x00
221 ,0x00,0x03,0xe1,0xe0,0x0e,0x1c,0x70,0x00
222 ,0x00,0x0f,0xc1,0xc3,0x0e,0x1c,0x70,0x00
223 ,0x00,0x3f,0x03,0xc3,0x8e,0x1c,0x70,0x00
224 ,0x00,0x3e,0x03,0x87,0x0e,0x1c,0x70,0x00
225 ,0x00,0x30,0x07,0x07,0x0e,0x18,0xe0,0x00
226 ,0x00,0x00,0xe0,0xe0,0x0e,0x38,0xe0,0x00
227 ,0x00,0x00,0x3e,0x1e,0x1e,0x38,0xe0,0x00
228 ,0x00,0x00,0xf8,0x1c,0x1c,0x38,0xe0,0x00
229 ,0x00,0x03,0xf0,0x38,0x3c,0x38,0xe0,0x00
230 ,0x00,0x3f,0xc0,0xf8,0x78,0x38,0xe0,0x00
231 ,0x00,0x7f,0x01,0xf0,0x70,0x38,0xf0,0x00
232 ,0x00,0x78,0x03,0xe0,0xe0,0x38,0x70,0x00
233 ,0x00,0x00,0x0f,0x81,0xe0,0x38,0x7c,0x00
234 ,0x00,0x00,0x3f,0x03,0xc0,0x38,0x3e,0x00
235 ,0x00,0x00,0xfc,0x0f,0x80,0x38,0x1e,0x00
236 ,0x00,0x07,0xf0,0x1f,0x1c,0x1c,0x04,0x00
237 ,0x00,0x3f,0xc0,0x3e,0x3f,0x1e,0x00,0x00
238 ,0x00,0x7f,0x00,0xf8,0x7f,0x0f,0x00,0x00
239 ,0x00,0x38,0x01,0xf0,0xf7,0x07,0xc0,0x00
240 ,0x00,0x00,0x07,0xe1,0xe3,0x83,0xf8,0x00
241 ,0x00,0x00,0x3f,0x87,0xc3,0xc0,0xfc,0x00
242 ,0x00,0x01,0xfe,0x0f,0x81,0xe0,0x3c,0x00
243 ,0x00,0x0f,0xf8,0x1f,0x00,0xf0,0x00,0x00

```

```

244 ,0x00,0x1f,0xc0,0x7c,0x00,0x7c,0x00,0x00
245 ,0x00,0x1e,0x01,0xff8,0x00,0x3f,0x00,0x00
246 ,0x00,0x00,0x07,0xe0,0x78,0x0f,0xc0,0x00
247 ,0x00,0x00,0x3f,0x81,0xfe,0x07,0xf0,0x00
248 ,0x00,0x01,0xfe,0x07,0xff,0x01,0xf0,0x00
249 ,0x00,0x07,0xf8,0x0f,0x87,0x80,0x30,0x00
250 ,0x00,0x07,0xc0,0x3f,0x03,0xe0,0x00,0x00
251 ,0x00,0x06,0x00,0xfc,0x01,0xf8,0x00,0x00
252 ,0x00,0x00,0x03,0xf0,0x00,0x7e,0x00,0x00
253 ,0x00,0x00,0x0f,0xc0,0x00,0x3f,0x80,0x00
254 ,0x00,0x00,0x7f,0x00,0xf8,0x0f,0x80,0x00
255 ,0x00,0x00,0xfc,0x03,0xfe,0x01,0x80,0x00
256 ,0x00,0x00,0xf0,0x1f,0xff,0x80,0x00,0x00
257 ,0x00,0x00,0x00,0x7f,0x07,0xe0,0x00,0x00
258 ,0x00,0x00,0x00,0xfc,0x03,0xf8,0x00,0x00
259 ,0x00,0x00,0x03,0xf0,0x00,0x78,0x00,0x00
260 ,0x00,0x00,0x0f,0xc0,0x00,0x18,0x00,0x00
261 ,0x00,0x00,0x0f,0x01,0xf8,0x00,0x00,0x00
262 ,0x00,0x00,0x00,0x07,0xfe,0x00,0x00,0x00
263 ,0x00,0x00,0x00,0x1f,0xfe,0x00,0x00,0x00
264 ,0x00,0x00,0x00,0x1e,0x0e,0x00,0x00,0x00
265 ,0x00,0x00,0x00,0x18,0x00,0x00,0x00,0x00
266 };
267 //-----
268 #define FinPr_valid_width 64
269 #define FinPr_valid_height 64
270 const uint8_t PROGMEM FinPr_valid_bits[] = {
271   0x00,0x00,0x03,0xfe,0x00,0x00,0x00,0x00
272 ,0x00,0x00,0x1f,0xff,0xe0,0x00,0x00,0x00
273 ,0x00,0x00,0x7f,0xff,0xf8,0x00,0x00,0x00
274 ,0x00,0x00,0xfc,0x00,0xfe,0x00,0x00,0x00
275 ,0x00,0x03,0xe0,0x00,0x1f,0x00,0x00,0x00
276 ,0x00,0x07,0xc0,0x00,0x07,0x80,0x00,0x00
277 ,0x00,0x0f,0x80,0x00,0x03,0xe0,0x00,0x00
278 ,0x00,0x0e,0x03,0xff,0x01,0xe0,0x00,0x00
279 ,0x00,0x1c,0x1f,0xff,0xe0,0xf0,0x00,0x00
280 ,0x00,0x3c,0x3f,0xff,0xf0,0x78,0x00,0x00
281 ,0x00,0x78,0x7c,0x00,0xf8,0x3c,0x00,0x00
282 ,0x00,0x70,0xf0,0x00,0x3c,0x1c,0x00,0x00
283 ,0x00,0xe1,0xe0,0x00,0x1e,0x1c,0x00,0x00
284 ,0x00,0xe1,0xc0,0x00,0x0f,0x0e,0x00,0x00
285 ,0x00,0xc3,0x81,0xfc,0x07,0x0e,0x00,0x00
286 ,0x00,0x03,0x83,0xff,0x07,0x8e,0x00,0x00
287 ,0x00,0x07,0x07,0x8f,0x83,0x87,0x00,0x00
288 ,0x00,0x0f,0x0f,0x03,0xc3,0x87,0x00,0x00
289 ,0x00,0x1e,0x0e,0x01,0xc3,0x87,0x00,0x00
290 ,0x00,0x3c,0x1c,0x00,0xe1,0x87,0x00,0x00
291 ,0x00,0xf8,0x1c,0x30,0xe1,0x87,0x00,0x00
292 ,0x07,0xf0,0x38,0x70,0xe1,0x86,0x00,0x00

```

```

293 ,0x07,0xc0,0x78,0x70,0xe3,0x8e,0x00,0x00
294 ,0x02,0x00,0xf0,0xf0,0xe3,0x8e,0x00,0x00
295 ,0x00,0x01,0xe0,0xe0,0xe3,0x8e,0x00,0x00
296 ,0x00,0x03,0xc1,0xe1,0xc3,0x8e,0x00,0x00
297 ,0x00,0x0f,0x83,0xc3,0xc3,0x8e,0x00,0x00
298 ,0x00,0x7f,0x07,0x83,0x83,0x0e,0x00,0x00
299 ,0x07,0xfc,0x0f,0x07,0x83,0x0e,0x00,0x00
300 ,0x07,0xf0,0x1e,0x0f,0x03,0x0e,0x00,0x00
301 ,0x07,0x80,0x7c,0x1e,0x03,0x07,0x00,0x00
302 ,0x00,0x00,0xf8,0x3c,0x03,0x87,0x80,0x00
303 ,0x00,0x03,0xf0,0x78,0x03,0x83,0xc0,0x00
304 ,0x00,0x1f,0xc0,0xf0,0x02,0x00,0x00,0x00
305 ,0x00,0xff,0x01,0xe1,0xc0,0x0c,0x00,0x00
306 ,0x07,0xfc,0x03,0xc3,0xe1,0xff,0xc0,0x00
307 ,0x07,0xe0,0x0f,0x87,0xc7,0xff,0xf0,0x00
308 ,0x07,0x00,0x3f,0x0f,0x0f,0xff,0xfc,0x00
309 ,0x00,0x00,0x7c,0x3e,0x3f,0xff,0xfe,0x00
310 ,0x00,0x03,0xf8,0x7c,0x3f,0xff,0xff,0x00
311 ,0x00,0x1f,0xe0,0xf0,0x7f,0xff,0xff,0x80
312 ,0x00,0xff,0x83,0xe0,0xff,0xff,0xff,0x80
313 ,0x01,0xfc,0x07,0xc1,0xff,0xff,0xe3,0xc0
314 ,0x01,0xe0,0x1f,0x01,0xff,0xff,0xc3,0xc0
315 ,0x00,0x00,0xfe,0x01,0xff,0xff,0x87,0xe0
316 ,0x00,0x03,0xf8,0x13,0xff,0xff,0x0f,0xe0
317 ,0x00,0x1f,0xe0,0x73,0xff,0xfe,0x1f,0xe0
318 ,0x00,0x7f,0x81,0xf3,0xff,0xfc,0x1f,0xe0
319 ,0x00,0xfc,0x03,0xe3,0xef,0xf8,0x3f,0xe0
320 ,0x00,0x60,0x0f,0xc3,0xc7,0xf0,0x7f,0xe0
321 ,0x00,0x00,0x3f,0x03,0xc3,0xe0,0xff,0xe0
322 ,0x00,0x00,0xfc,0x03,0xc1,0xc1,0xff,0xe0
323 ,0x00,0x07,0xf0,0x13,0xe0,0x83,0xff,0xe0
324 ,0x00,0x0f,0xc0,0x7b,0xf8,0x07,0xff,0xe0
325 ,0x00,0x0f,0x01,0xf9,0xfc,0x0f,0xff,0xc0
326 ,0x00,0x00,0x07,0xf1,0xfe,0x1f,0xff,0xc0
327 ,0x00,0x00,0x1f,0xc0,0xff,0x3f,0xff,0x80
328 ,0x00,0x00,0x7e,0x00,0xff,0xff,0xff,0x80
329 ,0x00,0x00,0xfc,0x00,0x7f,0xff,0xff,0x00
330 ,0x00,0x00,0xf0,0x1f,0x3f,0xff,0xfe,0x00
331 ,0x00,0x00,0x00,0x7f,0x1f,0xff,0xfc,0x00
332 ,0x00,0x00,0x01,0xff,0x8f,0xff,0xf8,0x00
333 ,0x00,0x00,0x03,0xe0,0xe3,0xff,0xe0,0x00
334 ,0x00,0x00,0x01,0x80,0x00,0x7f,0x00,0x00
335 };
336 //-----
337 #define FinPr_invalid_width 64
338 #define FinPr_invalid_height 64
339 const uint8_t PROGMEM FinPr_invalid_bits[] = {
340     0x00,0x00,0x03,0xfe,0x00,0x00,0x00,0x00
341     ,0x00,0x00,0x1f,0xff,0xe0,0x00,0x00,0x00

```

```

342 ,0x00,0x00,0x7f,0xff,0xf8,0x00,0x00,0x00
343 ,0x00,0x00,0xfc,0x00,0xfe,0x00,0x00,0x00
344 ,0x00,0x03,0xe0,0x00,0x1f,0x00,0x00,0x00
345 ,0x00,0x07,0xc0,0x00,0x07,0x80,0x00,0x00
346 ,0x00,0x0f,0x80,0x00,0x03,0xe0,0x00,0x00
347 ,0x00,0x0e,0x03,0xff,0x01,0xe0,0x00,0x00
348 ,0x00,0x1c,0x1f,0xff,0xe0,0xf0,0x00,0x00
349 ,0x00,0x3c,0x3f,0xff,0xf0,0x78,0x00,0x00
350 ,0x00,0x78,0x7c,0x00,0xf8,0x3c,0x00,0x00
351 ,0x00,0x70,0xf0,0x00,0x3c,0x1c,0x00,0x00
352 ,0x00,0xe1,0xe0,0x00,0x1e,0x1c,0x00,0x00
353 ,0x00,0xe1,0xc0,0x00,0x0f,0xe0,0x00,0x00
354 ,0x00,0xc3,0x81,0xfc,0x07,0x0e,0x00,0x00
355 ,0x00,0x03,0x83,0xff,0x07,0x8e,0x00,0x00
356 ,0x00,0x07,0x07,0x8f,0x83,0x87,0x00,0x00
357 ,0x00,0x0f,0x0f,0x03,0xc3,0x87,0x00,0x00
358 ,0x00,0x1e,0xe0,0x01,0xc3,0x87,0x00,0x00
359 ,0x00,0x3c,0x1c,0x00,0xe1,0x87,0x00,0x00
360 ,0x00,0xf8,0x1c,0x30,0xe1,0x87,0x00,0x00
361 ,0x07,0xf0,0x38,0x70,0xe1,0x86,0x00,0x00
362 ,0x07,0xc0,0x78,0x70,0xe3,0x8e,0x00,0x00
363 ,0x02,0x00,0xf0,0xf0,0xe3,0x8e,0x00,0x00
364 ,0x00,0x01,0xe0,0xe0,0xe3,0x8e,0x00,0x00
365 ,0x00,0x03,0xc1,0xe1,0xc3,0x8e,0x00,0x00
366 ,0x00,0x0f,0x83,0xc3,0xc3,0x8e,0x00,0x00
367 ,0x00,0x7f,0x07,0x83,0x83,0xe0,0x00,0x00
368 ,0x07,0xfc,0x0f,0x07,0x83,0xe0,0x00,0x00
369 ,0x07,0xf0,0x1e,0x0f,0x03,0xe0,0x00,0x00
370 ,0x07,0x80,0x7c,0x1e,0x03,0x07,0x00,0x00
371 ,0x00,0x00,0xf8,0x3c,0x03,0x87,0x80,0x00
372 ,0x00,0x03,0xf0,0x78,0x03,0x83,0xc0,0x00
373 ,0x00,0x1f,0xc0,0xf0,0x02,0x00,0x00,0x00
374 ,0x00,0xff,0x01,0xe1,0xc0,0x00,0x00,0x00
375 ,0x07,0xfc,0x03,0xc3,0xe1,0xff,0xc0,0x00
376 ,0x07,0xe0,0x0f,0x87,0xc7,0xff,0xf0,0x00
377 ,0x07,0x00,0x3f,0x0f,0x0f,0xff,0xf8,0x00
378 ,0x00,0x00,0x7c,0x3e,0x1f,0xff,0xfe,0x00
379 ,0x00,0x03,0xf8,0x7c,0x3f,0xff,0xff,0x00
380 ,0x00,0x1f,0xe0,0xf0,0x7f,0xff,0xff,0x00
381 ,0x00,0xff,0x83,0xe0,0xfe,0xff,0xb0,0x80
382 ,0x01,0xfc,0x07,0xc0,0xfc,0x7f,0x1f,0xc0
383 ,0x01,0xe0,0x1f,0x01,0xf8,0x3e,0x0f,0xc0
384 ,0x00,0x00,0xfe,0x01,0xf8,0x1c,0x07,0xe0
385 ,0x00,0x03,0xf8,0x13,0xf8,0x00,0x0f,0xe0
386 ,0x00,0x1f,0xe0,0x73,0xfc,0x00,0x1f,0xe0
387 ,0x00,0x7f,0x81,0xf3,0xfe,0x00,0x3f,0xe0
388 ,0x00,0xfc,0x03,0xe3,0xff,0x00,0x7f,0xe0
389 ,0x00,0x60,0x0f,0xc3,0xff,0x80,0xff,0xe0
390 ,0x00,0x00,0x3f,0x03,0xff,0x00,0x7f,0xe0

```

```

391 ,0x00,0x00,0xfc,0x03,0xfe,0x00,0x3f,0xe0
392 ,0x00,0x07,0xf0,0x13,0xfc,0x00,0x1f,0xe0
393 ,0x00,0x0f,0xc0,0x79,0xf8,0x08,0x0f,0xe0
394 ,0x00,0x0f,0x01,0xf9,0xf8,0x1c,0x0f,0xc0
395 ,0x00,0x00,0x07,0xf1,0xfc,0x3e,0x1f,0xc0
396 ,0x00,0x00,0x1f,0xc0,0xfe,0x7f,0x3f,0x80
397 ,0x00,0x00,0x7e,0x00,0xff,0xff,0xff,0x80
398 ,0x00,0x00,0xfc,0x00,0x7f,0xff,0xff,0x00
399 ,0x00,0x00,0xf0,0x1f,0x3f,0xff,0xfe,0x00
400 ,0x00,0x00,0x00,0x7f,0x1f,0xff,0xfc,0x00
401 ,0x00,0x00,0x01,0xff,0x8f,0xff,0xf8,0x00
402 ,0x00,0x00,0x03,0xe0,0xe3,0xff,0xe0,0x00
403 ,0x00,0x00,0x01,0x80,0x00,0x7f,0x00,0x00
404 };
405 //-----
406 #define FinPr_failed_width 64
407 #define FinPr_failed_height 64
408 const uint8_t PROGMEM FinPr_failed_bits[] = {
409 0x00,0x00,0x3f,0xe0,0x00,0x00,0x00,0x00
410 ,0x00,0x01,0xff,0xfe,0x00,0x00,0x00,0x00
411 ,0x00,0x0f,0xc0,0x1f,0x80,0x00,0x00,0x00
412 ,0x00,0x1e,0x00,0x03,0xc0,0x00,0x00,0x00
413 ,0x00,0x78,0x00,0x00,0xf0,0x00,0x00,0x00
414 ,0x00,0xe0,0x00,0x00,0x38,0x00,0x00,0x00
415 ,0x01,0xc0,0x00,0x00,0x1c,0x00,0x00,0x00
416 ,0x03,0x80,0x00,0x00,0x0e,0x00,0x00,0x00
417 ,0x07,0x00,0x7f,0xe0,0x07,0x00,0x00,0x00
418 ,0x06,0x01,0xff,0xf8,0x03,0x00,0x00,0x00
419 ,0x0c,0x03,0xc0,0x3c,0x03,0x80,0x00,0x00
420 ,0x1c,0x0f,0x00,0x0e,0x01,0x80,0x00,0x00
421 ,0x18,0x0c,0x00,0x03,0x00,0xc0,0x00,0x00
422 ,0x18,0x18,0x00,0x01,0x80,0xc0,0x00,0x00
423 ,0x30,0x38,0x00,0x01,0xc0,0xe0,0x00,0x00
424 ,0x30,0x30,0x0f,0x00,0xc0,0x60,0x00,0x00
425 ,0x30,0x30,0x3f,0xc0,0xe0,0x60,0x00,0x00
426 ,0x70,0x60,0x78,0xe0,0x60,0x60,0x00,0x00
427 ,0x60,0x60,0x60,0x60,0x60,0x70,0x00,0x00
428 ,0x60,0x60,0x60,0x60,0x60,0x30,0x00,0x00
429 ,0x60,0x60,0x60,0x60,0x30,0x30,0x00,0x00
430 ,0x60,0x60,0x60,0x30,0x30,0x20,0x00,0x00
431 ,0x60,0x60,0x60,0x30,0x30,0x01,0xe0,0x00
432 ,0x60,0x60,0x60,0x30,0x30,0x0f,0xfc,0x00
433 ,0x60,0x60,0x60,0x30,0x30,0x3f,0xff,0x00
434 ,0x60,0x60,0x60,0x30,0x18,0x78,0x03,0x80
435 ,0x60,0x60,0x60,0x30,0x1c,0x60,0x01,0x80
436 ,0x60,0x60,0x30,0x38,0x0c,0xc0,0x00,0xc0
437 ,0x00,0x60,0x30,0x18,0x00,0xc0,0x00,0xc0
438 ,0x00,0x60,0x30,0x18,0x00,0xc0,0x00,0xc0
439 ,0x00,0xe0,0x30,0x0c,0x01,0xc0,0x00,0xe0

```

```

440 ,0x00,0xc0,0x18,0x0e,0x01,0xc0,0x00,0xe0
441 ,0x60,0xc0,0x18,0x07,0x01,0xc0,0x00,0xe0
442 ,0x01,0xc0,0x1c,0x03,0x81,0xc0,0x00,0xe0
443 ,0x01,0x80,0x0c,0x01,0xc1,0xc0,0x00,0xe0
444 ,0x03,0x80,0x0e,0x00,0xf1,0xc0,0x00,0xe0
445 ,0x0f,0x00,0x06,0x00,0x01,0xc0,0x00,0xe0
446 ,0x3e,0x01,0x03,0x00,0x01,0xc0,0x00,0xe0
447 ,0x30,0x03,0x83,0x80,0x1f,0xff,0xff,0xfe
448 ,0x00,0x03,0x81,0xc0,0x3f,0xff,0xff,0xff
449 ,0x00,0x07,0xc0,0xe0,0x30,0x00,0x00,0x03
450 ,0x00,0x0e,0xc0,0x78,0x30,0x00,0x00,0x03
451 ,0x00,0x3c,0x60,0x1e,0x30,0x00,0x00,0x03
452 ,0x00,0x78,0x70,0x0f,0x30,0x00,0x00,0x03
453 ,0x03,0xe0,0x38,0x03,0x30,0x00,0x00,0x03
454 ,0x07,0x80,0x1c,0x00,0x30,0x00,0x00,0x03
455 ,0xc0,0x00,0x0f,0x00,0x30,0x00,0x00,0x03
456 ,0xc0,0x00,0x03,0x80,0x30,0x01,0xe0,0x03
457 ,0x00,0x18,0x01,0xe0,0x30,0x03,0xf0,0x03
458 ,0x00,0x18,0x00,0x7c,0x30,0x07,0x38,0x03
459 ,0x00,0x0c,0x00,0x1f,0x30,0x06,0x18,0x03
460 ,0x18,0x0e,0x00,0x07,0x30,0x06,0x18,0x03
461 ,0x0c,0x07,0x80,0x00,0x30,0x07,0x38,0x03
462 ,0x0e,0x03,0xc0,0x00,0x30,0x03,0x30,0x03
463 ,0x07,0x00,0xf0,0x00,0x30,0x03,0x30,0x03
464 ,0x03,0x00,0x7e,0x00,0x30,0x03,0x30,0x03
465 ,0x01,0x80,0x1f,0xc0,0x30,0x03,0x30,0x03
466 ,0x01,0xc0,0x03,0xe1,0x30,0x07,0xf8,0x03
467 ,0x00,0xf0,0x00,0x01,0x30,0x03,0xf0,0x03
468 ,0x00,0x38,0x00,0x00,0x30,0x00,0x00,0x03
469 ,0x00,0x1e,0x00,0x00,0x30,0x00,0x00,0x03
470 ,0x00,0x07,0xc0,0x00,0x30,0x00,0x00,0x03
471 ,0x00,0x01,0xff,0x80,0x3f,0xff,0xff,0xff
472 ,0x00,0x00,0x3f,0x80,0x1f,0xff,0xff,0xfe
473 };
474 //-----
475 #define FinPr_scan_width 64
476 #define FinPr_scan_height 64
477 const uint8_t PROGMEM FinPr_scan_bits[] = {
478   0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
479 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
480 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
481 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
482 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
483 ,0x00,0x00,0x00,0x1f,0xf8,0x00,0x00,0x00
484 ,0x00,0x00,0x00,0x7f,0xff,0x00,0x00,0x00
485 ,0x00,0x00,0x01,0xfc,0x7f,0xc0,0x00,0x00
486 ,0x00,0x00,0x03,0xc0,0x03,0xe0,0x00,0x00,0x00
487 ,0x00,0x00,0x07,0x80,0x00,0xf0,0x00,0x00,0x00
488 ,0x00,0x00,0xe0,0x00,0x00,0x3c,0x00,0x00,0x00

```

```

489 ,0x00,0x00,0x1c,0x1f,0xfc,0x1c,0x00,0x00
490 ,0x00,0x00,0x38,0x7f,0xfe,0x0e,0x00,0x00
491 ,0x00,0x00,0x78,0xf8,0x0f,0x87,0x00,0x00
492 ,0x00,0x00,0x71,0xe0,0x03,0xc7,0x00,0x00
493 ,0x00,0x00,0xe3,0x80,0x01,0xc3,0x80,0x00
494 ,0x00,0x00,0xc3,0x83,0xc0,0xe3,0x80,0x00
495 ,0x00,0x00,0xc7,0x0f,0xf0,0x71,0x80,0x00
496 ,0x00,0x00,0x06,0x1f,0xf8,0x71,0xc0,0x00
497 ,0x00,0x00,0xe0,0x1c,0x3c,0x31,0xc0,0x00
498 ,0x00,0x00,0x1c,0x38,0x1c,0x31,0xc0,0x00
499 ,0x00,0x00,0x38,0x70,0x0e,0x39,0xc0,0x00
500 ,0x00,0x01,0xf0,0x71,0x8e,0x39,0xc0,0x00
501 ,0x00,0x03,0xe0,0xe1,0x86,0x31,0xc0,0x00
502 ,0x00,0x03,0x81,0xe3,0x8e,0x31,0x80,0x00
503 ,0x00,0x00,0x03,0xc3,0x8e,0x33,0x80,0x00
504 ,0x00,0x00,0x07,0x87,0x0c,0x73,0x80,0x00
505 ,0x00,0x00,0x1f,0x0e,0x1c,0x73,0x80,0x00
506 ,0x7f,0xff,0xff,0xff,0xff,0xff,0xff,0xfe
507 ,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff
508 ,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff
509 ,0x7f,0xff,0xff,0xff,0xff,0xff,0xff,0xfe
510 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
511 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
512 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
513 ,0x00,0x03,0xf0,0x1e,0x3e,0x1c,0x00,0x00
514 ,0x00,0x03,0x80,0x7c,0x77,0x0f,0x00,0x00
515 ,0x00,0x00,0x01,0xf0,0xe3,0x07,0xc0,0x00
516 ,0x00,0x00,0x07,0xe3,0xc3,0x81,0xf0,0x00
517 ,0x00,0x00,0x3f,0x87,0x81,0xc0,0x60,0x00
518 ,0x00,0x01,0xfc,0x1f,0x00,0xf0,0x00,0x00
519 ,0x00,0x01,0xe0,0x3c,0x00,0x7c,0x00,0x00
520 ,0x00,0x00,0x00,0xf8,0x78,0x1f,0x00,0x00
521 ,0x00,0x00,0x07,0xe0,0xfc,0x0f,0xc0,0x00
522 ,0x00,0x00,0x3f,0x83,0xef,0x03,0xc0,0x00
523 ,0x00,0x00,0xfc,0x0f,0x87,0x80,0x00,0x00
524 ,0x00,0x00,0x70,0x1f,0x03,0xe0,0x00,0x00
525 ,0x00,0x00,0x00,0x7c,0x00,0xf8,0x00,0x00
526 ,0x00,0x00,0x01,0xf0,0x00,0x3e,0x00,0x00
527 ,0x00,0x00,0x0f,0xc0,0xf8,0x0f,0x00,0x00
528 ,0x00,0x00,0x1f,0x03,0xfe,0x02,0x00,0x00
529 ,0x00,0x00,0x0c,0x0f,0x8f,0x80,0x00,0x00
530 ,0x00,0x00,0x00,0x3f,0x03,0xe0,0x00,0x00
531 ,0x00,0x00,0x00,0xf8,0x00,0xf0,0x00,0x00
532 ,0x00,0x00,0x01,0xe0,0x00,0x30,0x00,0x00
533 ,0x00,0x00,0x01,0xc0,0xf8,0x00,0x00,0x00
534 ,0x00,0x00,0x00,0x07,0xfe,0x00,0x00,0x00
535 ,0x00,0x00,0x00,0x0f,0x8e,0x00,0x00,0x00
536 ,0x00,0x00,0x00,0x06,0x00,0x00,0x00,0x00
537 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

```

```

538 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
539 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
540 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
541 ,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
542 };
543
544 /////////////////
545 //      SETUP
546 /////////////////
547 void setup() {
548     Serial.begin(115200);
549     delay(1000);
550
551 //-----initiate OLED display-----
552 // SSD1306_SWITCHCAPVCC = generate display voltage from 3.3V internally
553
554 if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3D for 128x64
555     Serial.println(F("SSD1306 allocation failed"));
556     for(;); // Don't proceed, loop forever
557 }
558
559 // Show initial display buffer contents on the screen --
560 // the library initializes this with an Adafruit splash screen.
561 // you can delet these three lines if you don't want to get the Adfruit logo appear
562 display.display();
563 delay(2000);                                // Pause for 2 seconds
564 display.clearDisplay();
565
566 //-----
567 connectToWiFi();
568 //-----
569
570 // set the data rate for the sensor serial port
571 finger.begin(57600);
572 Serial.println("\n\nAdafruit finger detect test");
573
574 if (finger.verifyPassword()) {
575     Serial.println("Found fingerprint sensor!");
576     display.clearDisplay();
577     display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
578     FinPr_valid_height, WHITE);
579     display.display();
580 } else {
581     Serial.println("Did not find fingerprint sensor :(");
582     display.clearDisplay();
583     display.drawBitmap( 32, 0, FinPr_failed_bits, FinPr_failed_width,
584     FinPr_failed_height, WHITE);
585     display.display();
586     while (1) { delay(1); }

```

```

587 }
588 //-----
589 finger.getTemplateCount();
590 Serial.print("Sensor contains "); Serial.print(finger.templateCount);
591 Serial.println(" templates");
592 Serial.println("Waiting for valid finger...");
593
594 //Timers-----
595 timer.setInterval(25000L, CheckMode);
596 t1 = timer.setInterval(10000L, ChecktoAddID);      //Set an internal timer every
597 10sec to check if there a new fingerprint in the website to add it.
598 t2 = timer.setInterval(15000L, ChecktoDeleteID);   //Set an internal timer every
599 15sec to check wheater there an ID to delete in the website.
600
601 //-----
602 CheckMode();
603
604 }
605
606 /////////////////
607 // WHILE TRUE - LOOP
608 /////////////////
609
610 void loop() {
611     timer.run();                                //Keep the timer in the loop function in order
612     to update the time as soon as possible
613     //check if there's a connection to Wi-Fi or not
614     if(!WiFi.isConnected()){
615         if (millis() - previousMillis >= 10000) {
616             previousMillis = millis();
617             connectToWiFi();                      //Retry to connect to Wi-Fi
618         }
619     }
620     CheckFingerprint();                         //Check the sensor if the there a finger.
621     delay(10);
622 }
623
624 /////////////////
625 // CHECK FINGERPRINT
626 /////////////////
627
628 void CheckFingerprint(){
629     // unsigned long previousMillisM = millis();
630     // Serial.println(previousMillisM);
631     // If there no fingerprint has been scanned return -1 or -2 if there an error or 0
632     if there nothing, The ID start form 1 to 127
633     // Get the Fingerprint ID from the Scanner
634     FingerID = getFingerprintID();
635     DisplayFingerprintID();

```

```

636 // Serial.println(millis() - previousMillisM);
637 }
638 /////////////////
639 //      Display the fingerprint ID state on the OLED
640 /////////////////
641 void DisplayFingerprintID(){
642     //Fingerprint has been detected
643     if (FingerID > 0){
644         display.clearDisplay();
645         display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
646         FinPr_valid_height, WHITE);
647         display.display();
648
649         SendFingerprintID( FingerID );           // Send the Fingerprint ID to the
650         website.
651         delay(2000);
652     }
653
654 //-----
655 //No finger detected
656 else if (FingerID == 0){
657     display.clearDisplay();
658     display.drawBitmap( 32, 0, FinPr_start_bits, FinPr_start_width,
659     FinPr_start_height, WHITE);
660     display.display();
661 }
662 //-----
663 //Didn't find a match
664 else if (FingerID == -1){
665     display.clearDisplay();
666     display.drawBitmap( 34, 0, FinPr_invalid_bits, FinPr_invalid_width,
667     FinPr_invalid_height, WHITE);
668     display.display();
669 }
670 //-----
671 //Didn't find the scanner or there an error
672 else if (FingerID == -2){
673     display.clearDisplay();
674     display.drawBitmap( 32, 0, FinPr_failed_bits, FinPr_failed_width,
675     FinPr_failed_height, WHITE);
676     display.display();
677 }
678 //-----
679 }
680 }
681 /////////////////
682 //      send the fingerprint ID to IoT Edge website
683 ///////////////
684 
```

```
685 void SendFingerprintID( int finger ){
686     Serial.println("Sending the Fingerprint ID");
687     if(WiFi.isConnected()){
688         HTTPClient http; //Declare object of class HTTPClient
689         //GET Data
690         postData = "?FingerID=" + String(finger) + "&device_token=" + device_token; // Add
691         the Fingerprint ID to the Post array in order to send it
692         //GET methode
693         Link = URL + postData;
694         http.begin(Link); //initiate HTTP request //Specify content-type header
695
696         int httpCode = http.GET(); //Send the request
697         String payload = http.getString(); //Get the response payload
698
699         Serial.println(httpCode); //Print HTTP return code
700         Serial.println(payload); //Print request response payload
701         Serial.println(finger); //Print fingerprint ID
702
703         if (payload.substring(0, 5) == "login") {
704             String user_name = payload.substring(5);
705             // Serial.println(user_name);
706
707             display.clearDisplay();
708             display.setTextSize(2); // Normal 2:2 pixel scale
709             display.setTextColor(WHITE); // Draw white text
710             display.setCursor(15,0); // Start at top-left corner
711             display.print(F("Welcome"));
712             display.setCursor(0,20);
713             display.print(user_name);
714             display.display();
715         }
716         else if (payload.substring(0, 6) == "logout") {
717             String user_name = payload.substring(6);
718             // Serial.println(user_name);
719
720             display.clearDisplay();
721             display.setTextSize(2); // Normal 2:2 pixel scale
722             display.setTextColor(WHITE); // Draw white text
723             display.setCursor(10,0); // Start at top-left corner
724             display.print(F("Good Bye"));
725             display.setCursor(0,20);
726             display.print(user_name);
727             display.display();
728         }
729         delay(10);
730         http.end(); //Close connection
731     }
732 }
733 }
```

```

734 ///////////////////////////////////////////////////////////////////
735 //      Get the Fingerprint ID
736 ///////////////////////////////////////////////////////////////////
737
738 int getFingerprintID() {
739     uint8_t p = finger.getImage();
740     switch (p) {
741         case FINGERPRINT_OK:
742             //Serial.println("Image taken");
743             break;
744         case FINGERPRINT_NOFINGER:
745             //Serial.println("No finger detected");
746             return 0;
747         case FINGERPRINT_PACKETRECIEVEERR:
748             //Serial.println("Communication error");
749             return -2;
750         case FINGERPRINT_IMAGEFAIL:
751             //Serial.println("Imaging error");
752             return -2;
753         default:
754             //Serial.println("Unknown error");
755             return -2;
756     }
757     // OK success!
758     p = finger.image2Tz();
759     switch (p) {
760         case FINGERPRINT_OK:
761             //Serial.println("Image converted");
762             break;
763         case FINGERPRINT_IMAGEMESS:
764             //Serial.println("Image too messy");
765             return -1;
766         case FINGERPRINT_PACKETRECIEVEERR:
767             //Serial.println("Communication error");
768             return -2;
769         case FINGERPRINT_FEATUREFAIL:
770             //Serial.println("Could not find fingerprint features");
771             return -2;
772         case FINGERPRINT_INVALIDIMAGE:
773             //Serial.println("Could not find fingerprint features");
774             return -2;
775         default:
776             //Serial.println("Unknown error");
777             return -2;
778     }
779     // OK converted!
780     p = finger.fingerFastSearch();
781     if (p == FINGERPRINT_OK) {
782         //Serial.println("Found a print match!");

```

```

783 } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
784     //Serial.println("Communication error");
785     return -2;
786 } else if (p == FINGERPRINT_NOTFOUND) {
787     //Serial.println("Did not find a match");
788     return -1;
789 } else {
790     //Serial.println("Unknown error");
791     return -2;
792 }
793 // found a match!
794 //Serial.print("Found ID #"); Serial.print(finger.fingerID);
795 //Serial.print(" with confidence of "); Serial.println(finger.confidence);
796
797 return finger.fingerID;
798 }
799
800
801 ///////////////////////////////////////////////////////////////////
802 //      Check if there a Fingerprint ID to delete
803 void CheckToDeleteID(){
804     Serial.println("Check to Delete ID");
805     if(WiFi.isConnected()){
806         HTTPClient http; //Declare object of class HTTPClient
807         //GET Data
808         postData = "?DeleteID=check&device_token=" + String(device_token); // Add the
809         Fingerprint ID to the Post array in order to send it
810         //GET methode
811         Link = URL + postData;
812         http.begin(Link); //initiate HTTP request,
813         //    Serial.println(Link);
814         int httpCode = http.GET(); //Send the request
815         String payload = http.getString(); //Get the response payload
816
817         if (payload.substring(0, 6) == "del-id") {
818             String del_id = payload.substring(6);
819             Serial.println(del_id);
820             http.end(); //Close connection
821             deleteFingerprint( del_id.toInt() );
822             delay(1000);
823         }
824         http.end(); //Close connection
825     }
826 }
827
828 ///////////////////////////////////////////////////////////////////
829 //      Delete Fingerprint ID
830 ///////////////////////////////////////////////////////////////////
831 uint8_t deleteFingerprint( int id ) {

```

```

832     uint8_t p = -1;
833
834     p = finger.deleteModel(id);
835
836     if (p == FINGERPRINT_OK) {
837         //Serial.println("Deleted!");
838         display.clearDisplay();
839         display.setTextSize(2);           // Normal 2:2 pixel scale
840         display.setTextColor(WHITE);      // Draw white text
841         display.setCursor(0,0);          // Start at top-left corner
842         display.print(F("Deleted!\n"));
843         display.display();
844     } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
845         //Serial.println("Communication error");
846         display.clearDisplay();
847         display.setTextSize(1);           // Normal 1:1 pixel scale
848         display.setTextColor(WHITE);      // Draw white text
849         display.setCursor(0,0);          // Start at top-left corner
850         display.print(F("Communication error!\n"));
851         display.display();
852     return p;
853 } else if (p == FINGERPRINT_BADLOCATION) {
854     //Serial.println("Could not delete in that location");
855     display.clearDisplay();
856     display.setTextSize(1);           // Normal 1:1 pixel scale
857     display.setTextColor(WHITE);      // Draw white text
858     display.setCursor(0,0);          // Start at top-left corner
859     display.print(F("Could not delete in that location!\n"));
860     display.display();
861     return p;
862 } else if (p == FINGERPRINT_FLASHERR) {
863     //Serial.println("Error writing to flash");
864     display.clearDisplay();
865     display.setTextSize(1);           // Normal 1:1 pixel scale
866     display.setTextColor(WHITE);      // Draw white text
867     display.setCursor(0,0);          // Start at top-left corner
868     display.print(F("Error writing to flash!\n"));
869     display.display();
870     return p;
871 } else {
872     //Serial.print("Unknown error: 0x"); Serial.println(p, HEX);
873     display.clearDisplay();
874     display.setTextSize(2);           // Normal 2:2 pixel scale
875     display.setTextColor(WHITE);      // Draw white text
876     display.setCursor(0,0);          // Start at top-left corner
877     display.print(F("Unknown error:\n"));
878     display.display();
879     return p;
880 }

```

```

881 }
882
883
884 ///////////////////////////////////////////////////////////////////
885 //      Check if there a Fingerprint ID to add
886 ///////////////////////////////////////////////////////////////////
887 void ChecktoAddID(){
888 //  Serial.println("Check to Add ID");
889 if(WiFi.isConnected()){
890     HTTPClient http;           //Declare object of class HTTPClient
891     //GET Data
892     getData = "?Get_Fingerid=get_id&device_token=" + String(device_token); // Add the
893     Fingerprint ID to the Post array in order to send it
894     //GET methode
895     Link = URL + getData;
896     http.begin(Link);         //initiate HTTP request,
897     //  Serial.println(Link);
898     int httpCode = http.GET(); //Send the request
899     String payload = http.getString(); //Get the response payload
900
901     if (payload.substring(0, 6) == "add-id") {
902         String add_id = payload.substring(6);
903         Serial.println(add_id);
904         id = add_id.toInt();
905         http.end();           //Close connection
906         getFingerprintEnroll();
907     }
908     http.end(); //Close connection
909 }
910 }
911
912 ///////////////////////////////////////////////////////////////////
913 //      Check the Mode
914 ///////////////////////////////////////////////////////////////////
915
916 void CheckMode(){
917     Serial.println("Check Mode");
918     if(WiFi.isConnected()){
919         HTTPClient http; //Declare object of class HTTPClient
920         //GET Data
921         getData = "?Check_mode=get_mode&device_token=" + String(device_token); // Add the
922         Fingerprint ID to the Post array in order to send it
923         //GET methode
924         Link = URL + getData;
925         http.begin(Link); //initiate HTTP request,
926         //  Serial.println(Link);
927         int httpCode = http.GET(); //Send the request
928         String payload = http.getString(); //Get the response payload
929 }
```

```

930     if (payload.substring(0, 4) == "mode") {
931         String dev_mode = payload.substring(4);
932         int devMode = dev_mode.toInt();
933         if(!firstConnect){
934             device_Mode = devMode;
935             firstConnect = true;
936         }
937         //    Serial.println(dev_mode);
938         if(device_Mode && devMode){
939             device_Mode = false;
940             timer.disable(t1);
941             timer.disable(t2);
942             Serial.println("Deivce Mode: Attandance");
943         }
944         else if(!device_Mode && !devMode){
945             device_Mode = true;
946             timer.enable(t1);
947             timer.enable(t2);
948             Serial.println("Deivce Mode: Enrollment");
949         }
950         http.end(); //Close connection
951     }
952     http.end(); //Close connection
953 }
954 // Serial.print("Number of Timers: ");
955 // Serial.println(timer.getNumTimers());
956 }
957
958 ///////////////////////////////////////////////////////////////////
959 //      Enroll a Fingerprint ID
960 ///////////////////////////////////////////////////////////////////
961 uint8_t getFingerprintEnroll() {
962     int p = -1;
963     display.clearDisplay();
964     display.drawBitmap( 34, 0, FinPr_scan_bits, FinPr_scan_width, FinPr_scan_height,
965     WHITE);
966     display.display();
967     while (p != FINGERPRINT_OK) {
968
969         p = finger.getImage();
970         switch (p) {
971             case FINGERPRINT_OK:
972                 //Serial.println("Image taken");
973                 display.clearDisplay();
974                 display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
975                 FinPr_valid_height, WHITE);
976                 display.display();
977                 break;
978             case FINGERPRINT_NOFINGER:

```

```
979     //Serial.println(".");
980     display.setTextSize(1);           // Normal 2:2 pixel scale
981     display.setTextColor(WHITE);      // Draw white text
982     display.setCursor(0,0);          // Start at top-left corner
983     display.print(F("scanning"));
984     display.display();
985     break;
986   case FINGERPRINT_PACKETRECEIVEERR:
987     display.clearDisplay();
988     display.drawBitmap( 34, 0, FinPr_invalid_bits, FinPr_invalid_width,
989     FinPr_invalid_height, WHITE);
990     display.display();
991     break;
992   case FINGERPRINT_IMAGEFAIL:
993     Serial.println("Imaging error");
994     break;
995   default:
996     Serial.println("Unknown error");
997     break;
998   }
999 }
1000
1001 // OK success!
1002 p = finger.image2Tz(1);
1003 switch (p) {
1004   case FINGERPRINT_OK:
1005     display.clearDisplay();
1006     display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
1007     FinPr_valid_height, WHITE);
1008     display.display();
1009     break;
1010   case FINGERPRINT_IMAGEMESS:
1011     display.clearDisplay();
1012     display.drawBitmap( 34, 0, FinPr_invalid_bits, FinPr_invalid_width,
1013     FinPr_invalid_height, WHITE);
1014     display.display();
1015     return p;
1016   case FINGERPRINT_PACKETRECEIVEERR:
1017     Serial.println("Communication error");
1018     return p;
1019   case FINGERPRINT_FEATUREFAIL:
1020     Serial.println("Could not find fingerprint features");
1021     return p;
1022   case FINGERPRINT_INVALIDIMAGE:
1023     Serial.println("Could not find fingerprint features");
1024     return p;
1025   default:
1026     Serial.println("Unknown error");
1027     return p;
```

```

1028 }
1029 display.clearDisplay();
1030 display.setTextSize(2); // Normal 2:2 pixel scale
1031 display.setTextColor(WHITE); // Draw white text
1032 display.setCursor(0,0); // Start at top-left corner
1033 display.print(F("Remove"));
1034 display.setCursor(0,20);
1035 display.print(F("finger"));
1036 display.display();
1037 //Serial.println("Remove finger");
1038 delay(2000);
1039 p = 0;
1040 while (p != FINGERPRINT_NOFINGER) {
1041     p = finger.getImage();
1042 }
1043 Serial.print("ID "); Serial.println(id);
1044 p = -1;
1045 display.clearDisplay();
1046 display.drawBitmap( 34, 0, FinPr_scan_bits, FinPr_scan_width, FinPr_scan_height,
1047 WHITE);
1048 display.display();
1049 while (p != FINGERPRINT_OK) {
1050     p = finger.getImage();
1051     switch (p) {
1052         case FINGERPRINT_OK:
1053             //Serial.println("Image taken");
1054             display.clearDisplay();
1055             display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
1056 FinPr_valid_height, WHITE);
1057             display.display();
1058             break;
1059         case FINGERPRINT_NOFINGER:
1060             //Serial.println(".");
1061             display.setTextSize(1); // Normal 2:2 pixel scale
1062             display.setTextColor(WHITE); // Draw white text
1063             display.setCursor(0,0); // Start at top-left corner
1064             display.print(F("scanning"));
1065             display.display();
1066             break;
1067         case FINGERPRINT_PACKETRECEIVEERR:
1068             Serial.println("Communication error");
1069             break;
1070         case FINGERPRINT_IMAGEFAIL:
1071             Serial.println("Imaging error");
1072             break;
1073         default:
1074             Serial.println("Unknown error");
1075             break;
1076     }

```

```
1077 }
1078 // OK success!
1079
1080 p = finger.image2Tz(2);
1081 switch (p) {
1082     case FINGERPRINT_OK:
1083         //Serial.println("Image converted");
1084         display.clearDisplay();
1085         display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
1086 FinPr_valid_height, WHITE);
1087         display.display();
1088         break;
1089     case FINGERPRINT_IMAGEMESS:
1090         //Serial.println("Image too messy");
1091         display.clearDisplay();
1092         display.drawBitmap( 34, 0, FinPr_invalid_bits, FinPr_invalid_width,
1093 FinPr_invalid_height, WHITE);
1094         display.display();
1095         return p;
1096     case FINGERPRINT_PACKETRECIEVEERR:
1097         Serial.println("Communication error");
1098         return p;
1099     case FINGERPRINT_FEATUREFAIL:
1100         Serial.println("Could not find fingerprint features");
1101         return p;
1102     case FINGERPRINT_INVALIDIMAGE:
1103         Serial.println("Could not find fingerprint features");
1104         return p;
1105     default:
1106         Serial.println("Unknown error");
1107         return p;
1108     }
1109 }
1110
1111 // OK converted!
1112 Serial.print("Creating model for #"); Serial.println(id);
1113
1114 p = finger.createModel();
1115 if (p == FINGERPRINT_OK) {
1116     Serial.println("Prints matched!");
1117     display.clearDisplay();
1118     display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
1119 FinPr_valid_height, WHITE);
1120     display.display();
1121 } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
1122     Serial.println("Communication error");
1123     return p;
1124 } else if (p == FINGERPRINT_ENROLLMISMATCH) {
1125     Serial.println("Fingerprints did not match");
```

```

1126     display.clearDisplay();
1127     display.drawBitmap( 34, 0, FinPr_invalid_bits, FinPr_invalid_width,
1128     FinPr_invalid_height, WHITE);
1129     display.display();
1130     return p;
1131 } else {
1132     Serial.println("Unknown error");
1133     return p;
1134 }
1135
1136 Serial.print("ID "); Serial.println(id);
1137 p = finger.storeModel(id);
1138 if (p == FINGERPRINT_OK) {
1139     Serial.println("Stored!");
1140     display.clearDisplay();
1141     display.drawBitmap( 34, 0, FinPr_valid_bits, FinPr_valid_width,
1142     FinPr_valid_height, WHITE);
1143     display.display();
1144     confirmAdding(id);
1145 } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
1146     Serial.println("Communication error");
1147     return p;
1148 } else if (p == FINGERPRINT_BADLOCATION) {
1149     Serial.println("Could not store in that location");
1150     return p;
1151 } else if (p == FINGERPRINT_FLASHERR) {
1152     Serial.println("Error writing to flash");
1153     return p;
1154 } else {
1155     Serial.println("Unknown error");
1156     return p;
1157 }
1158 }
1159
1160 ///////////////////////////////////////////////////////////////////
1161 //      Check if there a Fingerprint ID to add
1162 ///////////////////////////////////////////////////////////////////
1163 void confirmAdding(int id){
1164     Serial.println("confirm Adding");
1165     if(WiFi.status() == WL_CONNECTED){
1166         HTTPClient http;    //Declare object of class HTTPClient
1167         //GET Data
1168         postData = "?confirm_id=" + String(id) + "&device_token=" + String(device_token);
1169         // Add the Fingerprint ID to the Post array in order to send it
1170         //GET methode
1171         Link = URL + postData;
1172
1173         http.begin(Link); //initiate HTTP request,
1174     //    Serial.println(Link);

```

```

1175     int httpCode = http.GET();    //Send the request
1176     String payload = http.getString();    //Get the response payload
1177     if(httpCode == 200){
1178         display.clearDisplay();
1179         display.setTextSize(1.5);           // Normal 1:1 pixel scale
1180         display.setTextColor(WHITE);        // Draw white text
1181         display.setCursor(0,0);           // Start at top-left corner
1182         display.print(payload);
1183         display.display();
1184         Serial.println(payload);
1185         delay(2000);
1186     }
1187     else{
1188         Serial.println("Error Confirm!!!");
1189     }
1190     http.end(); //Close connection
1191 }
1192 }
1193
1194
1195 ///////////////////////////////////////////////////////////////////
1196 //      connect to the WiFi
1197 ///////////////////////////////////////////////////////////////////
1198 void connectToWiFi(){
1199     WiFi.mode(WIFI_OFF);           //Prevents reconnection issue (taking
1200     too long to connect)
1201     delay(1000);
1202     WiFi.mode(WIFI_STA);
1203     Serial.print("Connecting to ");
1204     Serial.println(ssid);
1205     WiFi.begin(ssid, password);
1206
1207     display.clearDisplay();
1208     display.setTextSize(1);          // Normal 1:1 pixel scale
1209     display.setTextColor(WHITE);        // Draw white text
1210     display.setCursor(0, 0);           // Start at top-left corner
1211     display.print(F("Connecting to \n"));
1212     display.setCursor(0, 50);
1213     display.setTextSize(2);
1214     display.print(ssid);
1215     display.drawBitmap( 73, 10, Wifi_start_bits, Wifi_start_width, Wifi_start_height,
1216     WHITE);
1217     display.display();
1218
1219
1220     uint32_t periodToConnect = 30000L;
1221     for(uint32_t StartToConnect = millis(); (millis()-StartToConnect) <
1222     periodToConnect;){
1223         if ( WiFi.status() != WL_CONNECTED ){

```

```
1224     delay(500);
1225     Serial.print(".");
1226 } else{
1227     break;
1228 }
1229 }

1230 if(WiFi.isConnected()){
1231     Serial.println("");
1232     Serial.println("Connected");
1233
1234     display.clearDisplay();
1235     display.setTextSize(2);           // Normal 1:1 pixel scale
1236     display.setTextColor(WHITE);      // Draw white text
1237     display.setCursor(8, 0);         // Start at top-left corner
1238     display.print(F("Connected \n"));
1239     display.drawBitmap( 33, 15, Wifi_connected_bits, Wifi_connected_width,
1240 Wifi_connected_height, WHITE);
1241     display.display();
1242
1243
1244     Serial.print("IP address: ");
1245     Serial.println(WiFi.localIP()); //IP address assigned to your ESP
1246 }
1247 else{
1248     Serial.println("");
1249     Serial.println("Not Connected");
1250     WiFi.mode(WIFI_OFF);
1251     delay(1000);
1252 }
1253 delay(1000);
1254 }
1255 // END OF CODE
1256
1257 //////////////////////////////////////////////////////////////////
```

Annex C - Change Bitmap to array

There are many online tools to change images into byte arrays (or your array back into an image) for use with Arduino and (monochrome) displays such as OLEDs. The web page <https://javl.github.io/image2cpp/> were used to convert the following images used in this project.

BMP simbol	ID	Use
	Wifi_start	Waiting valid Wi-fi credentials
	Wifi_connected	The IoT device is connected to Wi-fi
	FinPr_start	No finger detected
	FinPr_failed	Didn't find the scanner or there an error
	FinPr_scan	Enroll a Fingerprint
	FinPr_valid	Fingerprint has been detected
	FinPr_invalid	Didn't find a match

1 Annex D - Smart contract - timinglog.sol

```

2 // SPDX-License-Identifier: GPL-3.0
3
4 pragma solidity ^0.6.1;
5 pragma experimental ABIEncoderV2;
6
7 contract timinglog {
8
9     uint constant DAY_IN_SECONDS = 86400;
10    uint constant YEAR_IN_SECONDS = 31536000;
11    uint constant LEAP_YEAR_IN_SECONDS = 31622400;
12
13    uint constant HOUR_IN_SECONDS = 3600;
14    uint constant MINUTE_IN_SECONDS = 60;
15
16    uint16 constant ORIGIN_YEAR = 1970;
17
18    function isLeapYear(uint year) pure private returns (bool) {
19        if (year % 4 != 0) {
20            return false;
21        }
22        if (year % 100 != 0) {
23            return true;
24        }
25        if (year % 400 != 0) {
26            return false;
27        }
28        return true;
29    }
30
31    function toTimestamp(uint year, uint month, uint day) view private returns (uint timestamp) {
32        return toTimestamp(year, month, day, 0, 0, 0);
33    }
34
35
36    function toTimestamp(uint year, uint month, uint day, uint hour, uint minute,
37    uint second) view private returns (uint timestamp) {
38        uint16 i;
39
40        // Year
41        for (i = ORIGIN_YEAR; i < year; i++) {
42            if (isLeapYear(i)) {
43                timestamp += LEAP_YEAR_IN_SECONDS;
44            }
45            else {
46                timestamp += YEAR_IN_SECONDS;
47            }
48        }
49
50        // Month
51        uint8[12] memory monthDayCounts;
52        monthDayCounts[0] = 31;
53        if (isLeapYear(year)) {
54            monthDayCounts[1] = 29;
55        }
56        else {
57            monthDayCounts[1] = 28;
58        }
59        monthDayCounts[2] = 31;
60        monthDayCounts[3] = 30;
61        monthDayCounts[4] = 31;
62        monthDayCounts[5] = 30;
63        monthDayCounts[6] = 31;

```

```

64         monthDayCounts[7] = 31;
65         monthDayCounts[8] = 30;
66         monthDayCounts[9] = 31;
67         monthDayCounts[10] = 30;
68         monthDayCounts[11] = 31;
69
70         for (i = 1; i < month; i++) {
71             timestamp += DAY_IN_SECONDS * monthDayCounts[i - 1];
72         }
73
74         // Day
75         timestamp += DAY_IN_SECONDS * (day - 1);
76
77         // Hour
78         timestamp += HOUR_IN_SECONDS * (hour);
79
80         // Minute
81         timestamp += MINUTE_IN_SECONDS * (minute);
82
83         // Second
84         timestamp += second;
85
86         return timestamp;
87     }
88
89     mapping (uint => mapping (uint => string)) internal comp_date_hash_map;
90                                     // comp_date_hash_map[company][date] we get a
91     string
92
93     function append_log(uint _company, uint year, uint month, uint day, string memory
94     _log) public returns (bool OK) {
95         // check dates
96         // check company account
97         uint local_date = toTimestamp(year,month,day);
98         bytes memory tempEmptyStringTest =
99         bytes(comp_date_hash_map[_company][local_date]);
100        require(tempEmptyStringTest.length == 0); // must be empty the log at this
101 day
102        comp_date_hash_map[_company][local_date] = _log;
103        return true;
104    }
105
106    function get_log(uint _company, uint year, uint month, uint day) view public
107 returns (string memory _log) {
108        uint local_date = toTimestamp(year,month,day);
109        bytes memory tempEmptyStringTest =
110        bytes(comp_date_hash_map[_company][local_date]);
111        require(tempEmptyStringTest.length != 0); // must have data the log at this
112 day
113        _log = comp_date_hash_map[_company][local_date];
114        return _log;
115    }
116}

```

Annex E - How to deploy a Smart contract

Deploy smart contract means to have it available to users of an Ethereum network. For this project, it was used Ropsten test network. To deploy a smart contract, is send an Ethereum transaction containing the code of the compiled smart contract without specifying any recipients. This means it will need to pay a transaction fee so make sure there is available some ETH.

For the EVM to be able to run your contract it needs to be in bytecode. Compilation turns this:

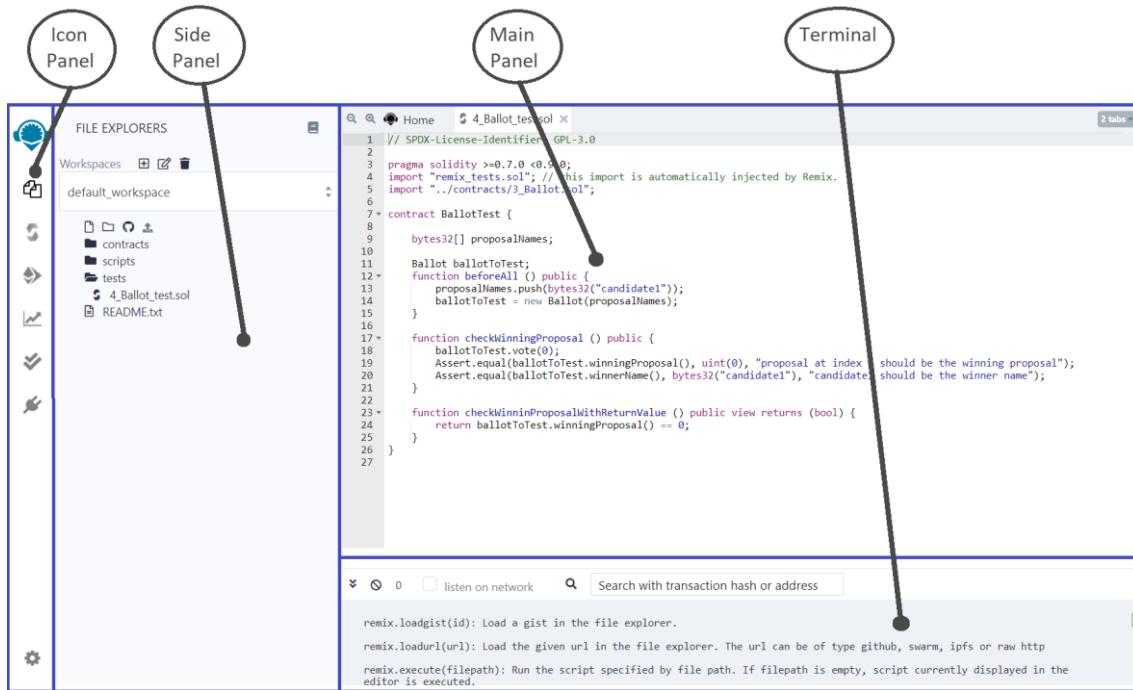
```
pragma solidity 0.4.24;

contract Greeter {
    function greet() public constant returns (string) {
        return "Hello";
    }
}
```

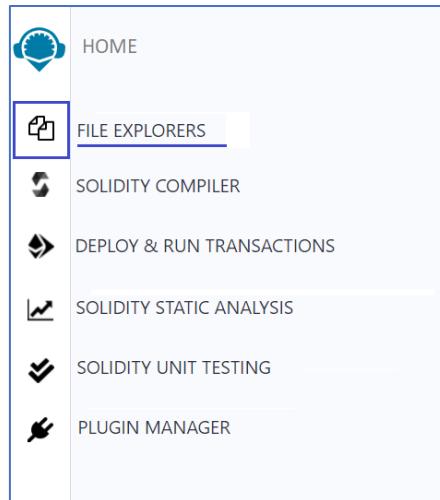
into this

Once deployed, your contract will have an Ethereum address like other accounts.

Remix (<https://remix.ethereum.org>) Remix IDE allows developing, deploying and administering smart contracts for Ethereum like blockchains. As June 2021, Remix web page is divided into four panels:



Icon panel: at the left side, it shows the menu, with different activities that can be executed, like file management, compile solidity code or deploy Ethereum contract



Side panel: in the middle, it shows the details

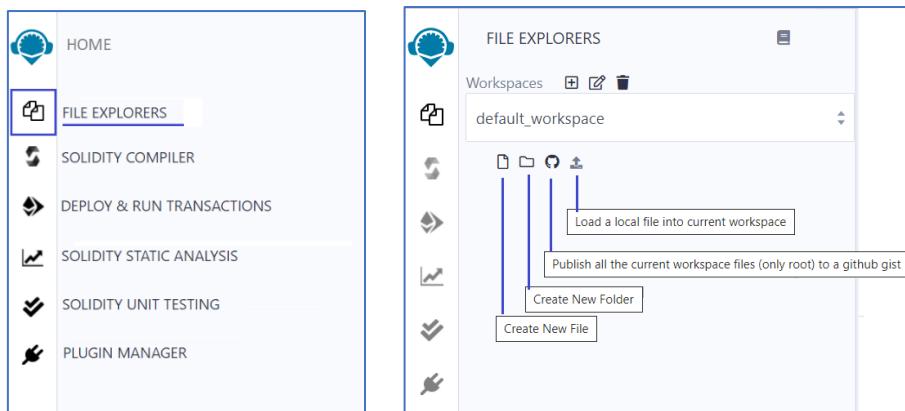
Main panel: at the right side, it is the main windows

Terminal: at the bottom side, helps to see the result

STEP 1:

Create a new file with Solidity code. Click on icon panel, file explores icon (two sheets icon).

You can organize different projects in different folder. Click in the option you will open the smart contract: New File, New Folder, Publish to GitHub or Load File.



On main panel it will be show a new tab with the contract name, and several program lines

```

68     timestamp += DAY_IN_SECONDS * monthDayCounts[i - 1];
69 }
70
71 // Day
72 timestamp += DAY_IN_SECONDS * (day - 1);
73
74 // Hour
75 timestamp += HOUR_IN_SECONDS * (hour);
76
77 // Minute
78 timestamp += MINUTE_IN_SECONDS * (minute);
79
80 // Second
81 timestamp += second;
82
83 return timestamp;
84 }

85 mapping (uint => mapping (uint => string)) internal comp_date_hash_map;
86 // comp_date_hash_map[company][date] we get a string

87 function append_log(uint _company, uint year, uint month, uint day, string memory _log) public
88 {
89     // check dates
90     // check company account
91     uint local_date = toTimestamp(year,month,day);
92     bytes memory tempEmptyStringTest = bytes(comp_date_hash_map[_company][local_date]);
93     require(tempEmptyStringTest.length == 0); // must be empty, the log at this day
94 }
```

STEP 2:

Compile the contract. Click on the compile icon on the left hand side

```

function addnewevent(string memory _cifnumber, bytes memory tempEmptyStringTest) {
    if (tempEmptyStringTest.length > 0) {
        company[_cifnumber].history.push(tempEmptyStringTest);
        emit thereisnewcompany(_cifnumber);
        return true;
    } else return false;
}

function get5events(string memory _cifnumber) {
    bytes memory tempEmptyStringTest;
    if (tempEmptyStringTest.length > 0) {
        for (uint i = 0; i<10; i++) {
            _history[i] = "";
        }
    }
    bytes memory tempEmptyStringTest;
    if (tempEmptyStringTest.length > 0) {
        for (uint i = 0; i<10; i++) {
            if (((numofchunk-1)*10) < i) {
                _history[i] = company[_cifnumber].history[i];
            } else _history[i] = "";
        }
    }
    return _history;
} else return _history;
}

```

The following libraries are accessible:

- web3 version 1.0.0
- ethers.js
- swarmpy
- remix (run remix.help() for more info)

If compile is successful, it showed a button for "compilation details" and a green tick on solidity button

The following libraries are accessible:

- web3 version 1.0.0
- ethers.js
- swarmpy
- remix (run remix.help() for more info)

Click on ABI, and save ABI code

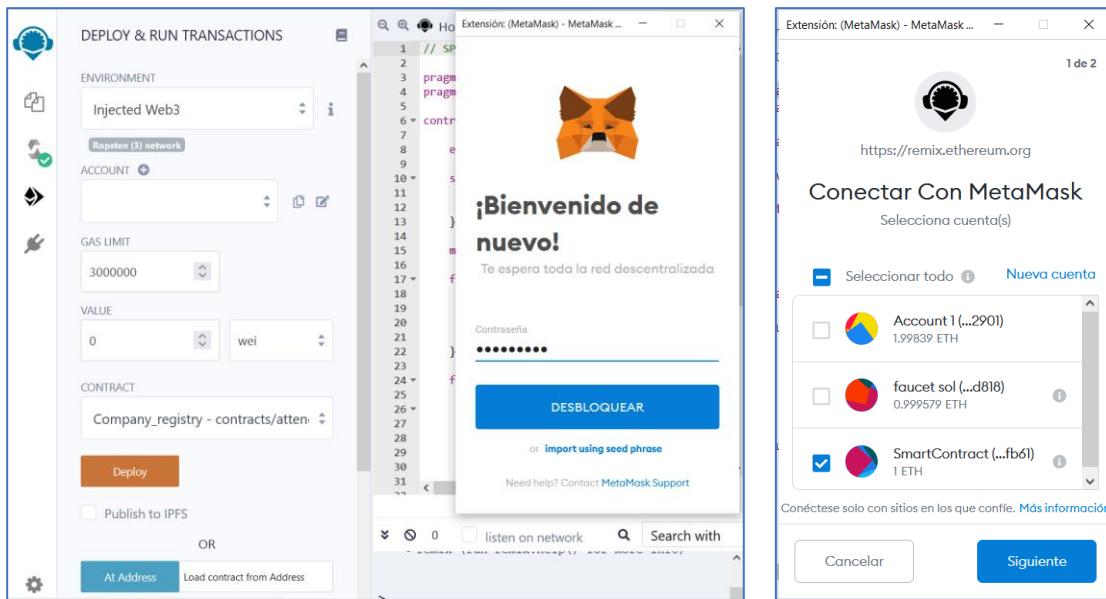
```
[  
 {  
     "inputs": [  
         {  
             "internalType": "uint256",  
             "name": "_company",  
             "type": "uint256"  
         },  
         {  
             "internalType": "uint256",  
             "name": "year",  
             "type": "uint256"  
         },  
         {  
             "internalType": "uint256",  
             "name": "month",  
             "type": "uint256"  
         },  
         {  
             "internalType": "uint256",  
             "name": "day",  
             "type": "uint256"  
         },  
         {  
             "internalType": "string",  
             "name": "_log",  
             "type": "string"  
         }  
     ],  
     "name": "append_log",  
     "outputs": [  
         {  
             "internalType": "bool",  
             "name": "OK",  
             "type": "bool"  
         }  
     ],  
     "stateMutability": "nonpayable",  
     "type": "function"  
 },  
 {  
     "inputs": [  
         {  
             "internalType": "uint256",  
             "name": "_company",  
             "type": "uint256"
```

```
        },
        {
            "internalType": "uint256",
            "name": "year",
            "type": "uint256"
        },
        {
            "internalType": "uint256",
            "name": "month",
            "type": "uint256"
        },
        {
            "internalType": "uint256",
            "name": "day",
            "type": "uint256"
        }
    ],
    "name": "get_log",
    "outputs": [
        {
            "internalType": "string",
            "name": "_log",
            "type": "string"
        }
    ],
    "stateMutability": "view",
    "type": "function"
}
]
```

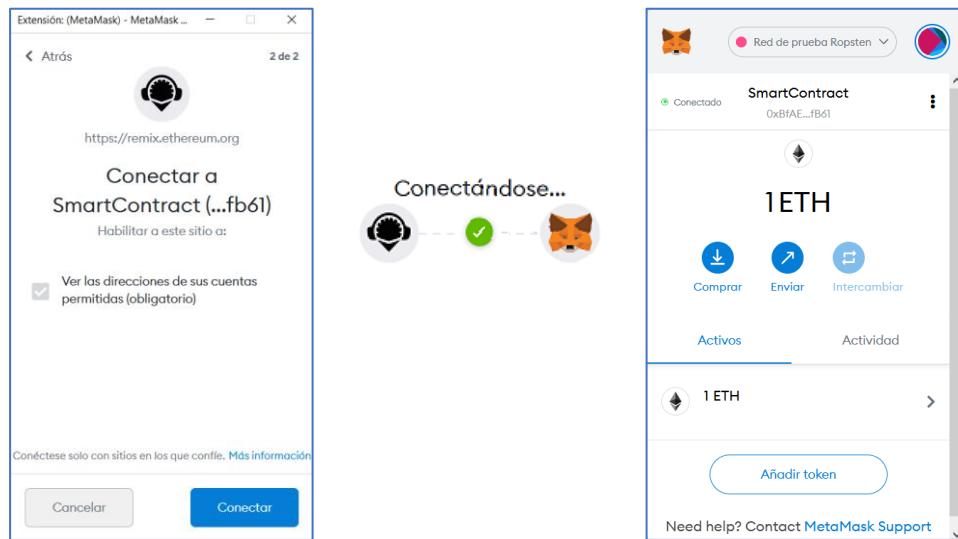
STEP 3:

Deploy contract on Ethereum. Click on Ethereum icon, select Injected Web3 environment. MetaMask plugin will be ask for password. Enter your credentials.

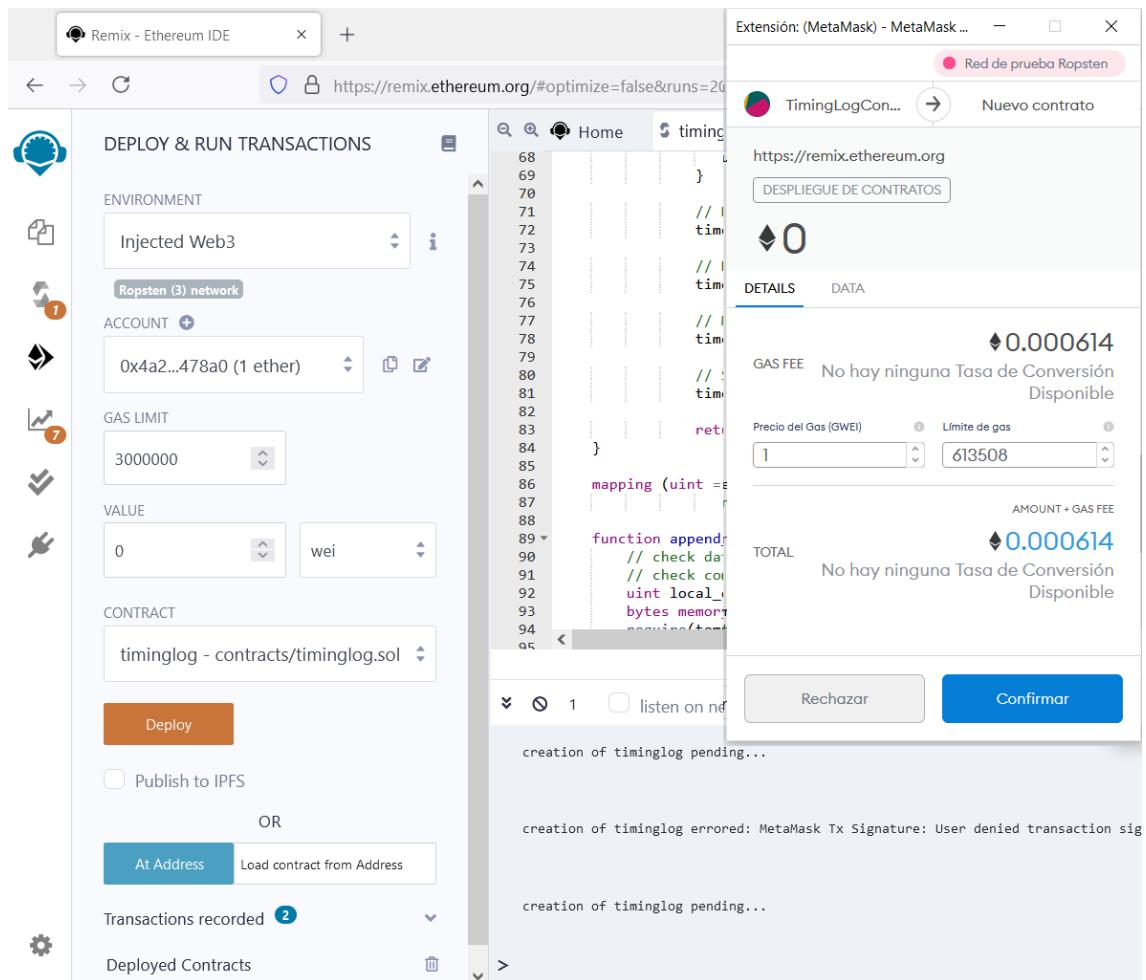
MetaMask will ask permit to connect to Remix.



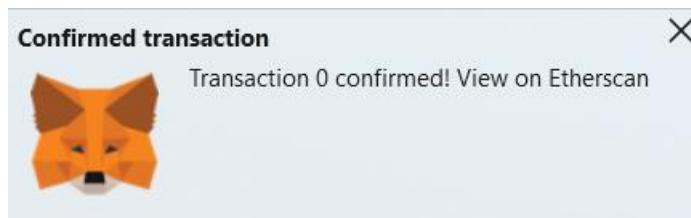
Select which account will be connected to Remix, and what network is used (ropsten). To check, only click on MetaMask plugin logo



Deploy contract by click on "Deploy" button at Remix page. MetaMask will ask for Gas fee (1GWEI) and Gas Limit (613508), and show the price, in ETH. This is the price to deploy the contact. Click on "Confirm" button. MetaMask will request to miners to work with this contract. If there is not enough gas, the contract will not be deployed, but the computation cost will be charged to the account



Wait some seconds, as the process need time, until see the confirmation message



STEP 4:

Once the blockchain is created, Remix will notify the transaction was done. To see details of transaction, see debug panel and click on hash.

Copy hash value and contract address

The screenshot shows the Truffle UI interface. On the left, there's a sidebar with sections for 'Transactions recorded' (2), 'Deployed Contracts', and a specific entry for 'TIMINGLOG AT 0x3BB...FABA4 (BLOCKC...)'. Below this are buttons for 'append_log' and 'get_log', both with parameters 'uint256 _company, uint256 y'. Under 'Low level interactions', there's a 'CALLDATA' section with a 'Transact' button. The main right-hand pane displays the Solidity source code for the TIMINGLOG contract, showing functions like 'append_log' and 'get_log'. A log message at the bottom indicates that the creation of the timinglog errored due to a user denied transaction signature. Another message shows the creation of timinglog pending. At the bottom, a green checkmark icon and text indicate a successful transaction: '[block:10436258 txIndex:2] from: 0x4a2...478a0 to: timinglog.(constructor) value: 0 wei data: 0x608...c0033 logs: 0 hash: 0xfa2...259bd'.

Contract address: 0x3bb37250390a96f6dc99fedd6386359c03cfaba4

Transactions details available at:

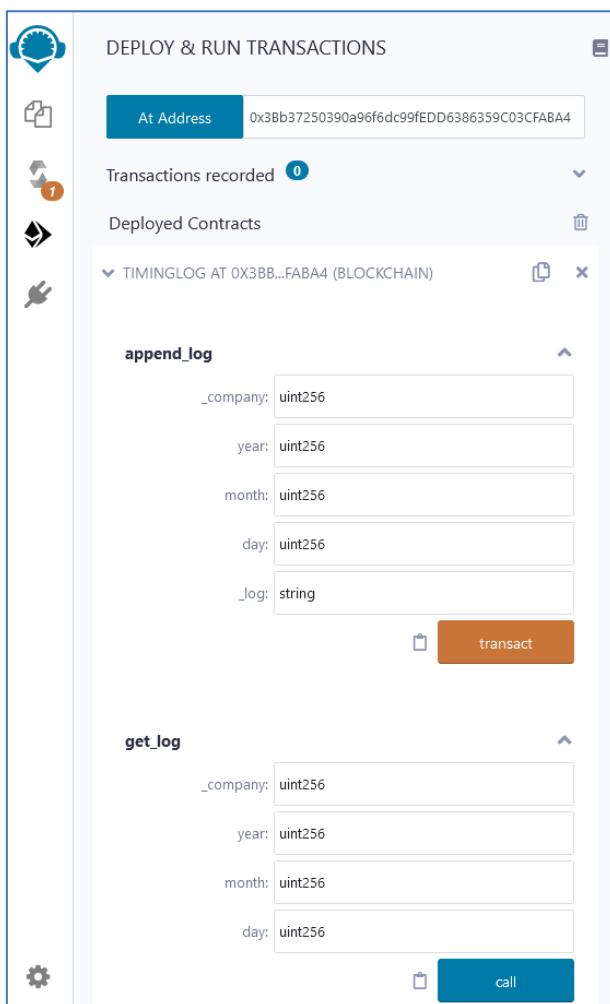
<https://ropsten.etherscan.io/tx/0xfa2da61a1918ecdfdc22e78411cd92aca00e44a75b1ef4519e802c7e4c2259bd>

The screenshot shows the Etherscan Testnet Network transaction details for the deployed contract. The transaction hash is 0xfa2da61a1918ecdfdc22e78411cd92aca00e44a75b1ef4519e802c7e4c2259bd. The status is Success, and it was included in block 10436258 with 2 Block Confirmations. The transaction occurred 47 secs ago on Jun-14-2021 12:06:24 PM +UTC. The transaction fee was 0.000613508 Ether (\$0.00) and the gas price was 0.000000001 Ether (1 Gwei).

STEP 5:

To open any deployed contract, at any other computer, there is need the solidity code, ABI code and contract address.

Open Remix. Under file tab upload solidity code. Under compile tab click on compile. Go to Deploy tab, select "Injected Web3" environment. Connect to MetaMask. Instead click on Deploy (not needed because the contract is already deployed), write the address at the section "At Address". You will see the details of Smart Contract.



1 Annex F: DApp: index.html

```

2 <html>
3
4 <head>
5   <meta charset="UTF-8" />
6   <title>DAppTimeLog</title>
7   <link rel="icon" type="image/svg" href="metamask-fox.svg" />
8
9   <link href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=
10 swap" rel="stylesheet" />
11   <link href="https://cdnjs.cloudflare.com/ajax/libs/twitter-
12 bootstrap/4.4.1/css/bootstrap.min.css" rel="stylesheet" />
13   <link href="https://cdnjs.cloudflare.com/ajax/libs/mdbootstrap/4.14.1/css/mdb.min.c
14 ss" rel="stylesheet" />
15
16   <link rel="stylesheet" href="metamask.css" />
17 </head>
18
19 <body>
20   <main class="container-fluid">
21     <header>
22       <div id="logo-container">
23         <h1 id="logo-text" class="text-center">
24           Employee Attendance Management in Ethereum
25         </h1>
26
27         
28       </div>
29     </header>
30
31     <!-- Part 1 Setting up Basic Actions and Status-->
32     <section>
33       <div class="row d-flex justify-content-center">
34         <div class="col-xl-4 col-lg-6 col-md-12 col-sm-12 col-12">
35           <div class="card">
36             <div class="card-body">
37               <h4 class="card-title">
38                 Basic Actions
39               </h4>
40
41               <button class="btn btn-primary btn-lg btn-block mb-
42 3" id="connectButton">
43                 --- Connect to MetaMask ---
44               </button>
45
46             </div>
47           </div>

```

```
48     </div>
49     </div>
50 </section> <br>
51 <!-- /Part 1 -->
52
53 <section>
54     <div class="row d-flex justify-content-center">
55
56         <!-- Append Log => add attendance registries-->
57         <div class="col-xl-4 col-lg-6 col-md-12 col-sm-12 col-12 d-flex align-items-
58 stretch">
59             <div class="card full-width">
60                 <div class="card-body">
61                     <h4 class="card-title">
62                         Store Attendance Report
63                     </h4>
64                     <label>Company CIF: </label>
65                     <input type="text" id="company_cif">
66                     </input><br>
67                     <label for="type_event">Day of record: </label>
68
69                     <input type="date" id="registry_date" value="2021-05-30" </input><br>
70
71                     <label>
72                         <br>File to upload: <br>
73                     </label><br>
74
75                     <input type="file" id="file_event"></input><br>
76                     <button class="btn btn-primary btn-lg btn-block mb-3" id="addEvent">
77                         Add Attendance Report
78                     </button>
79                 </div>
80             </div>
81         </div>
82
83         <!-- Get log => Reading attendance registries -->
84         <div class="col-xl-4 col-lg-6 col-md-12 col-sm-12 col-12 d-flex align-items-
85 stretch">
86             <div class="card full-width">
87                 <div class="card-body">
88                     <h4 class="card-title">
89                         Read attendance report
90                     </h4>
91                     <label>Company CIF: </label>
92                     <input type="text" id="company_cif">
93                     </input><br>
94                     <label>Day of record: </label>
95
96                     <input type="date" id="registry_date" value="2021-05-30" </input><br>
```

```
97
98      <button class="btn btn-primary btn-lg btn-block mb-3" id="getFile">
99          Get Attendance Report
100     </button>
101   </div>
102 </div>
103 </div>
104
105 </div>
106 </section>
107
108 <section>
109   <div id="contenedor">
110     </div>
111 </section> <br>
112
113 </main>
114
115 <script src="https://cdn.ethers.io/lib/ethers-
116 5.0.umd.min.js" type="application/javascript">
117 </script>
118 <script src="node_modules/@metamask/onboarding/dist/metamask-
119 onboarding.bundle.js" defer>
120 </script>
121 <script src="https://cdn.jsdelivr.net/npm/ipfs-http-client/dist/index.min.js">
122 </script>
123 <script src="contract.js" defer>
124 </script>
125
126 </body>
127
128 </html>
```


1 Annex G: DApp - metamask.css

```
2 section {
3   margin: 20px 0 20px 0;
4 }
5
6 .info-text {
7   font-size: 1.1em;
8 }
9
10 .full-width {
11   width: 100%;
12 }
13
14 .card {
15   margin-bottom: 20px;
16 }
17
18 /* Logo & Header */
19 header {
20   display: flex;
21   justify-content: center;
22   align-items: center;
23 }
24
25 #logo-container {
26   display: inline-flex;
27   flex-direction: column;
28   justify-content: center;
29   align-items: center;
30   margin-bottom: 20px;
31 }
32
33 #logo-text {
34   padding-top: 15px;
35   padding-bottom: 5px;
36 }
37
38 #mm-logo {
39   width: 20%;
40 }
41
42 /* Encrypt/Decrypt */
43
44 #encryptMessageInput {
45   margin-bottom: 1rem;
46 }
```


1 Annex H: DApp - contract.js

```

2  /*
3   * The `TimeLogContract` is compiled from:
4   * project folder (C:\TFM\solidity
5   * include
6   *   * TimeLogContract.sol
7   *   * ContractAccountAddress.txt
8   *   * ABI.txt
9   */
10
11 // * ES6 module: `import MetamaskOnboarding from '@metamask/onboarding'`
12 // * ES5 module: `const MetamaskOnboarding = require('@metamask/onboarding')`
13 // import MetaMaskOnboarding from '@metamask/onboarding'
14
15 // Address for web server (nodejs) that contains all packages
16 const onboard = new MetamaskOnboarding({ forwarderOrigin })
17
18 // Web Page Variables definitions
19
20 // Basic Actions Section
21 const onboardButton = document.getElementById('connectButton');
22 const accountsDiv = document.getElementById('accounts')
23
24 // Store Attendance Report Section
25 //const company_cif = document.getElementById('company_cif');
26 //const type_event = document.getElementById('type_event');
27 //const addEventButton = document.getElementById('addEvent');
28
29 // Read Attendance Section
30 //const company_cif      = document.getElementById('company_cif');
31 //const append_logButton = document.getElementById('append_log');
32
33 // Smart contract where is deployed TIMELOG contract
34 const ContractAccountAddress = '0x3Bb37250390a96f6dc99FEDD6386359C03CFABA4';
35
36 // Smart contract binary interface ABI
37 // deploy solidity contract and copy ABI result
38
39 const ABI = [
40   {
41     "inputs": [
42       {
43         "internalType": "uint256",
44         "name": "_company",
45         "type": "uint256"
46       },
47       {

```

```
48     "internalType": "uint256",
49     "name": "year",
50     "type": "uint256"
51 },
52 {
53     "internalType": "uint256",
54     "name": "month",
55     "type": "uint256"
56 },
57 {
58     "internalType": "uint256",
59     "name": "day",
60     "type": "uint256"
61 },
62 {
63     "internalType": "string",
64     "name": "_log",
65     "type": "string"
66 }
67 ],
68 "name": "append_log",
69 "outputs": [
70     {
71         "internalType": "bool",
72         "name": "OK",
73         "type": "bool"
74     }
75 ],
76 "stateMutability": "nonpayable",
77 "type": "function"
78 },
79 {
80     "inputs": [
81         {
82             "internalType": "uint256",
83             "name": "_company",
84             "type": "uint256"
85         },
86         {
87             "internalType": "uint256",
88             "name": "year",
89             "type": "uint256"
90         },
91         {
92             "internalType": "uint256",
93             "name": "month",
94             "type": "uint256"
95         },
96     {
```

```

97     "internalType": "uint256",
98     "name": "day",
99     "type": "uint256"
100   },
101 ],
102   "name": "get_log",
103   "outputs": [
104     {
105       "internalType": "string",
106       "name": "_log",
107       "type": "string"
108     }
109   ],
110   "stateMutability": "view",
111   "type": "function"
112 }
113 ]
114
115 /*
116 *****
117   Connect to MetaMask
118 *****
119 */
120
121 /*
122 const initialize = async () => {
123   //You will start here
124
125   let onboarding
126   try {
127     onboarding = new MetaMaskOnboarding({ forwarderOrigin })
128   } catch (error) {
129     console.error(error)
130   }
131
132
133 const onClickConnect = async () => {
134   try {
135     //Will open the MetaMask UI
136     // You should disable this button while the request is pending!
137     await ethereum.request({ method: 'eth_requestAccounts' });
138     onboardButton.disabled = true;
139   } catch (error) {
140     console.error(error);
141   }
142 }
143
144
145 }
```

```
146 */
147
148 const initialize = () => {
149   //You will start here
150
151   //This is to check if the brower has the Metamask extension installed
152   const isMetaMaskInstalled = () => {
153     const { ethereum } = window;
154     return Boolean(ethereum && ethereum.isMetaMask);
155   };
156
157   // Show ether account in case of Metamask is connected
158   let accounts
159
160   const isMetaMaskConnected = () => accounts && accounts.length > 0
161
162   //This will start the onboarding proccess, to MetaMask installation page
163   const onClickInstall = () => {
164     onboardButton.innerText = 'Onboarding in progress';
165     onboardButton.disabled = true;
166     onboarding.startOnboarding();
167   };
168
169   //Will open small windows: the MetaMask UI
170   // You should disable this button while the request is pending!
171   const onClickConnect = async () => {
172     try {
173       const newAccounts = await ethereum.request({
174         method: 'eth_requestAccounts',
175       })
176       handleNewAccounts(newAccounts)
177     } catch (error) {
178       console.error(error)
179     }
180   }
181
182   //Now we check to see if MetaMask is installed
183   const MetamaskClientCheck = async () => {
184     if (!isMetaMaskInstalled()) {
185       //If it isn't installed we ask the user to click to install it
186       onboardButton.innerText = 'Click here to install MetaMask!';
187       onboardButton.onclick = onClickInstall;
188       onboardButton.disabled = false;
189     } else if (isMetaMaskConnected()) {
190       //If it is installed we change our button text
191       onboardButton.innerText = 'Connected'
192       //When the button is clicked we call this function to connect the users MetaMas
193       k Wallet
194       onboardButton.disabled = true
```

```

195     if (onboarding) {
196         onboarding.stopOnboarding()
197     }
198 } else {
199     onboardButton.innerText = 'Connect'
200     onboardButton.onclick = onClickConnect
201     onboardButton.disabled = false
202 }
203 };
204
205 MetamaskClientCheck();
206
207
208
209 function handleNewAccounts (newAccounts) {
210     accounts = newAccounts
211     accountsDiv.innerHTML = accounts
212     if (isMetaMaskConnected()) {
213         initializeAccountButtons()
214     }
215     updateButtons()
216 }
217
218 };
219
220
221
222
223
224
225
226
227
228
229 window.addEventListener('DOMContentLoaded', initialize); //Lanza la inicialización se
230 gún cargue toda la página HTML
231
232 //Se instancian el provider y el "firmador" para poder trabajar con el contrato
233 //window.ethereum se refiere al wallet Metamask que es una extensión del navegador
234 const provider = new ethers.providers.Web3Provider(window.ethereum);
235 //El signer se obtiene para poder hacer métodos de alteración de estado (o transferen
236 cias)
237 const signer = provider.getSigner();
238
239 //Se instancia un objeto del contrato y se firma con signer
240 const myContract = new ethers.Contract(ContractAccountAddress, ABI, provider); //Prox
241 y del contrato en el usuario
242 const myContractWithSignature = myContract.connect(signer); //Proxy anterior con la f
243 uncionalidad de firmar

```

```
244
245 var fileData = ''; //Metainformación del archivo del usuario
246
247 //Se trata el fichero al pulsar en el elemento de explorador de archivos
248 const fileSelector = document.getElementById('file_event');
249 fileSelector.addEventListener('change', (evento) => {
250     fileData = evento.target.files[0];
251     console.log(fileData);
252 });
253
254 //Se instancia un objeto IPFS con un nodo de infura (es gratuito por ahora) y se pone
255 el puerto 5001
256 //La librería IpfsHttpClient ha sido previamente importado en el HTML
257 const ipfs = window.IpfsHttpClient({ host: 'ipfs.infura.io', port: 5001 })
258
259
```

Annex I: DApp - Debugging code in VSC

Create package.json file

A package.json file: lists the packages your project depends on specifies versions of a package that your project can use using semantic versioning rules makes your build reproducible, and therefore easier to share with other developers

In the project file, open CMD window and run "npm init" command. The program will ask about parameters and will create an updated version of package.json file

```
npm init
  package name: (dapptimelog)
  version: (1.0.0)
  description: Building DApp using MetaMask
  entry point: (contract.js)
  test command: npm run lint
  git repository:
  keywords:
  author: MetaMask
  license: (ISC) MIT
About to write to C:\asignaturas\DApTimeLog\package.json:
```

modified section, adding this lines:

```
script: {
  "deploy": "./deploy.sh",
  "lint": "eslint . --ext js,json",
  "lint:fix": "eslint . --fix --ext js,json",
  "serve": "static-server . --port 9011",
  "test": "npm run lint"
}
```

add working files

```
"files": [
  "/contract.js",
  "/index.html"
],
```

in this file are stored specific dependencies, that will be filled in the next steps

"dependencies" : Packages required by your application in production.
"devDependencies" : Packages that are only needed for local development and testing.

The difference between these two, is that devDependencies are modules which are only required during development, while dependencies are modules which are also required at runtime. To save a dependency as a devDependency on installation we need to do an npm install --save-dev , instead of just an npm install --save.

```
"name": "metamask-simple-dapp",
"version": "1.0.0",
"description": "Building a simple dapp used in MetaMask e2e tests.",
```

Update version in package.json file, information available at <https://www.npmjs.com/>

```
@metamask/onboarding": "^0.2.1",
"eth-sig-util": "^2.5.3",
"ethers": "^5.0.21",
"ipfs-core": "^0.8.0",
"ipfs-http-client": "^48.1.1",
"web3": "^1.3.0"
```

```
@metamask/onboarding": "^1.0.1",
"eth-sig-util": "^3.0.1",
"ethers": "^5.3.1",
"ipfs-core": "^0.8.0",
"ipfs-http-client": "^50.1.2",
"web3": "^1.3.6"
```

Install Lintern

ESLint is a javascript tool that checks your code for potential errors and bad code practices. It helps you enforce a code standard and style guide in your codebase. You can add ESLint in any of your JavaScript Code. It's not only limited to React Projects. You can use it with Vue.js, Node.js, or even vanilla Javascript Projects. It's pluggable and highly configurable. ESLint supports multiple plugins and extensions to extend and enhance its functionality.

[Documentation - ESLint - Pluggable JavaScript linter <https://eslint.org/docs/user-guide/formatters/>]

ESLINT is pluggable linting utility for Javascript and JSX, it help to discover possible errors.

ESLint is a linter -a tool that analyzes your code and flags potential errors. This is very helpful to avoid common mistakes that are made while you're coding (using undefined variables, syntactic errors, etc).

lintern <https://eslint.org/docs/user-guide/getting-started>

1 run npm

```
npm install eslint --save-dev
```

On new folders you might also need to create a .eslintrc configuration file.

You should then set up a configuration file, and the easiest way to do that is to use the --init flag. Note: --init assumes you have a package.json file already. If you don't, make sure to run npm init

```
$ npx eslint --init
```

it will be opened several questions:

check syntax and problems

- ✓ How would you like to use ESLint? · problems
- ✓ What type of modules does your project use? · none
- ✓ Which framework does your project use? · none
- ✓ Does your project use TypeScript? · No / Yes
- ✓ Where does your code run? · browser
- ✓ What format do you want your config file to be in? · JavaScript

Successfully created .eslintrc.js file in C:\asignaturas\DApTimeLog

Add Lintern as a extension of Visual Studio Code

VS Code is one of the top editor for development it was developed and maintain by Microsoft it help to improve productivity and also comes with many features

let try to install ESLINT extension on vs code editor

extension ESlint

view - extension

((this extension is enable globally))

succeeded

Run all programs.

Everytime npm is executed, NPM generates a 'package-lock.json' file. The package-lock.json file is a little more complex due to a trade-off between determinism and simplicity. Due to this complexity, the package-lock will generate the same node_modules folder for different npm versions. Every dependency will have an exact version number associated with it in the package-lock file.

Open cdm terminal. To install all dependencies needed fir the proyecto, run "npm install" command. Once the installation is finisehd, it will be created a new folder named "node_modules" with severals files inside

```
$ npm install
```

save ether : This package is used to make transactions and make calls to the contract deployed in Ethereum. It will be necessary to open a command console in the project directory and write:

```
$ npm install --save ethers
```

install ipfs : This package is the one that will upload to the Inter Planetary File System the file that the client uploads as a vehicle event. A command console is opened in the project and typed:

```
$ npm install ipfs-core
```

4) on console or using terminal inside VSC, run debugger to check syntax and errors

see console: get code suggestions, error checking, and warning highlights

```
npm run lint
```

Fixing error

the result for debug was as follow:

```
C:\asignaturas\DApptimeLog\contract.js
  39:7  error  'ContractAccountAddress' is assigned a value but
never used  no-unused-vars
  147:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
  148:13 error  'ethereum' is not defined    no-undef
  161:14 error  'isMetaMaskConnected' is not defined  no-undef
  167:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
  184:22 error  'ethers' is not defined       no-undef
  189:24 error  'ethers' is not defined       no-undef
  189:40 error  'contract' is not defined     no-undef
```

```
223:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
224:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
225:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
226:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
227:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
228:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
229:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
230:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
231:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
232:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
233:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
248:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
249:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
249:21 error  'chunk_read' is not defined    no-undef
250:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
251:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
252:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
253:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
254:1  error  Mixed spaces and tabs          no-mixed-spaces-and-tabs
```

3. DESIGN AND CONFIGURATION

```
255:1  error  Mixed spaces and tabs      no-mixed-spaces-and-tabs
256:1  error  Mixed spaces and tabs      no-mixed-spaces-and-tabs
257:1  error  Mixed spaces and tabs      no-mixed-spaces-and-tabs
258:1  error  Mixed spaces and tabs  no-mixed-spaces-and-tabs
259:1  error  Mixed spaces and tabs  no-mixed-spaces-and-tabs
260:1  error  Mixed spaces and tabs  no-mixed-spaces-and-tabs
```

as we already has the old version of package.json and .eslintrc.js, it can be added lines, with caution, because it can appears 500 errors instead of 500 warnings. The first errors was related to indent, because in some parts of the program it was used two spaces, in another 4 spaces and others use tab.

1) write indent conditions in .eslint

```
"rules": {
  "indent": [1, 2, {"SwitchCase": 1}]
}
```

firs numbers refers how to mark this error:

- "off" or 0 - turn the rule off
- "warn" or 1 - turn the rule on as a warning (doesn't affect exit code)
- "error" or 2 - turn the rule on as an error (exit code is 1 when triggered)

According to documentation, second '2' is for number of spaces used for indent (can be set to 4 or 6 or "tab").

2) add missing dependencies, if it is showed

errors:

ESLint couldn't find the config "@metamask/eslint-config" to extend from.

solution

```
npm install @metamask/eslint-config
```

Could not find a declaration file for module '@metamask/onboarding'.

```
npm i --save-dev @types/metamask__onboarding
```

3 do it iterative until no error appears