RASv.1 User Manual

Eficent Business and IT Consulting Services S.L.

This user guide serves as a reference for the technical description, initialization, functioning and frequently asked questions about the RAS v.1.

Contents

List of Figures

1 Introduction

The **RASv.1** (or RFID Attendance System version 1) is a time/punch clock machine fully integrated with OpenERP, conceived as a open-software and open hardware solution to the employee attendance registration problem, and designed by **Eficent Business and IT Consulting Services S.L.**

All the different elements that conform the RASv.1 device are available online:

- The software can be found at the **docker-rfid-hr-attendance** repository.
- The PCB design is located at the **efficent-rfid-attendance-pcb-design** repository.
- The case 3D design is available at the **efficent-rfid-attendance-case-design** repository.

2 Technical description

2.1 Main components

A brief description of the most important elements that conform the working loop of the RASv.1 is developed in this section.

i. *NodeMCU*

The NodeMCU [8] is an open-source firmware and development kit widely spread for its use at IoT (Internet of Things) projects. This kind of device presents some advantages of utilization as the possibility of coding at a well-known integrated development environment as the Arduino IDE; the presence of the ESP8266 WiFi MCU, allowing an easy connection to the WiFi network; or the integration of several communication protocols and signal generation processes, as GPIOs, PWM, IIC...

Its main functionality in the working loop consists of reading the RFID cards IDs (using ii.) and sending these data to another node (iv.) through MQTT [10] (iii.), to finally receive information about the result of the employee identification and entrance/leaving registration and transmit it to the user by visual (colour LEDs) and resounding (buzzer) feedback.

It is included as one of the components of the PCB (vi.).

ii. *RFID-RC522 Reader*

The MFRC-522 [7] RFID cards reader is a cheap RFID module specifically designed for its use in electronics projects. It employs a SPI communication protocol to send the data to the NodeMCU (i.), who receives it through its GPIO pins (some

concrete pins are designed for their possible use as ports in a SPI communication).

iii. *MQTT*

The Message Queuing Telemetry Transport protocol, also known as MQTT [10], is one of the preferred IoT publish-subscribe lightweight messaging methods, and the one selected for the wireless communication between the NodeMCU (i.) and other node (iv.).

iv. *Python MQTT Client*

This MQTT (iii.) node behaves as a link between the NodeMCU (i.) and Odoo. It was designed to receive the data from the other MQTT client by reading to some subscribed topics, and making a request to Odoo so that the MRP tool registers the entrance/departure of the employee, and also sends the feedback information of the result of the operation back to the NodeMCU.

v. *3D printed case*

The 3D case was designed using FreeCAD [4], a widely used open-source 3D design program, converted to GCODE using the Cura Lulzbot6 software, and printed using the Lulzbot Mini at first, and then the Lulzbot Taz 6. It is formed of three parts: the bottom case, which hass the different holes prepared for screwing the PCB to it, and the case to the wall; and the top case that is divided in two subparts (in next sections it is explained with detail).

vi. *Printed Circuit Board*

The PCB works as a union element, in which all the electronic components are soldered. It has been designed using Kicad [6], an open-source PCB designing software.

2.2 Detailed description

Once the different elements have been introduced, the work process of this internship must be explained, so that the different development steps become clear. The different tasks will be explained in a chronological way:

- PCB design: the hardware consists of:
  - NodeMCU: the "brain" of the electronic set (receiving information from the RFID reader and giving feedback using the LEDs, producing the PWM signals for the buzzer...)

- YWRobot Bredboard [9]: converts the electrical energy that arrives to it from the power supply from 9V to 5V, adapting it to the NodeMCU input voltage. It also feeds the buzzer control circuit.
- MFRC-522: provides the RFID ID to the NodeMCU.
- LEDs: supply feedback information of the result of the operation, as well as of possible errors, to the user (the colors code is explained in Section ***.
- Buzzer control circuit: conformed by a variable resistor, a transistor and a buzzer as main components, the circuit allows to control the buzzer volume by changing the potentiometer resistance.

The schematic designed circuit can be found in Figure 1, as well as the PCB final design can be found at Figure 2 (as a diagram and the final result).

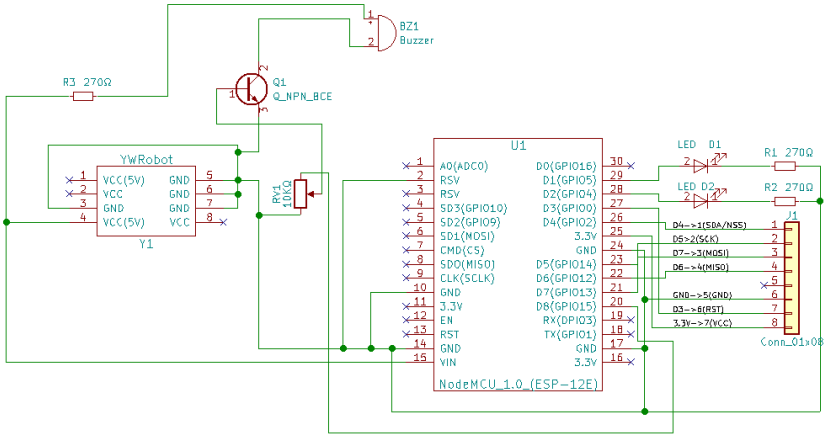


Figure 1: PCB schematic diagram

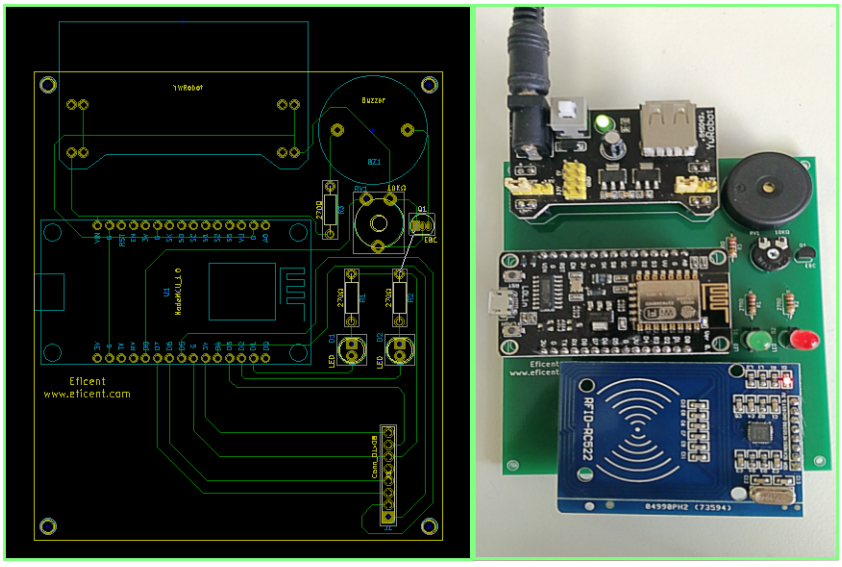
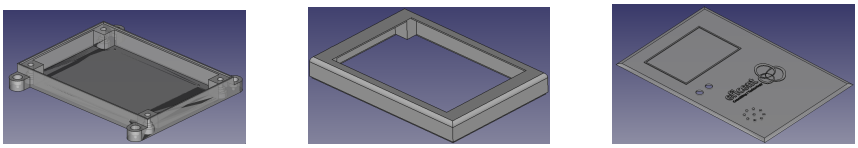


Figure 2: PCB design

- 3D case design: it is divided into three different pieces for a easier 3D printing process. The bottom one is presented in Figure 3a, while the top one is divided into two subparts, being the bottom one in Figure 3b and the top subpart in Figure 3c.



(a) Bottom part (no subparts)

(b) Top - Bottom subpart

(c) Top - Top subpart

Figure 3: 3D design different parts

The printed and assembled product can be seen in Figure 4.

Figure 4: Actual finished product

- Software: once the hardware part of the project has been covered, the software part is explained. There were mainly three kinds of coding involved:
  - Docker [3]: this platform allows the user to set multiple containers (they are kind of services packed in a storing function, so that only the necessary re-

sources for the task are introduced into it, saving a lot of space and computer capabilities) to solve different problems with distinct sets of requirements. For our project, it was employed to run at the same time the MQTT broker alongside the Python client.

– Arduino-based code: the NodeMCU code run several tasks as get information from the subscribed topics, read the RFID cards, encrypt and code the RFID ID before sending it and publish this data (as well as different functions related with the WiFi connection, configuration parameters file reading and writing...).

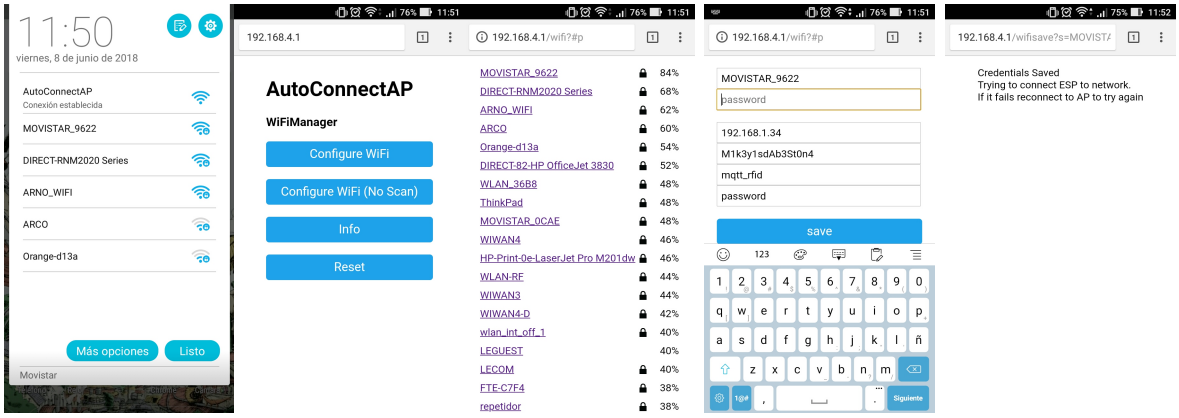
  This custom security methods include encoding (Base64 [2]), encryption (AES [1]) and authentication (HMAC-SHA256 [5]) functions.

– Python: apart from the code of the Odoo module, the Python client was modified from the original one that the company had developed previously to my arrival, so that I included the decryption, authentication, and decoding methods complementary to the aforementioned ones.

3 Configuration

The steps to correctly configure the RASv.1 once it has been just received are:

1. Plug on the power supply to the RASv.1 voltage entrance.
2. After a few seconds, using a smart device as smartphones or tablets to a WiFi network (it is an AP point) named that just appeared. It does not have a password (see Figure 5a).



(a) (b) (c) (d) (e)

Figure 5: WiFi configuration steps

3. Once connected, by trying to connect to any website, it will redirect you to the configuration portal. Tap on *Configure WiFi* (see Figure 5b).
4. From the several WiFi networks appearing (see Figure 5c), select yours and you will go to the screen seen at Figure 5d. Introduce the Wifi password in the *password* field. The rest of the parameters are, in appearing order:
    - Odoo host: IP address of the server where the Docker containers are running. Change the one appearing by yours.
    - Key: digits and letters string used for the encryption and authentication operations. DO NOT CHANGE IT UNLESS THE DOCKER ENVIRONMENTAL VARIABLE **KEY** IS ALSO CHANGED IN THE SAME WAY.
    - MQTT user: username for the NodeMCU to connect to the MQTT broker. DO NOT CHANGE IT UNLESS YOU INTRODUCE A NEW USER:PASSWORD PAIR AT THE **passwd** FILE AT THE DOCKER MOSQUITTO CONTAINER.
    - MQTT password: password for the NodeMCU to connect to the MQTT broker. DO NOT CHANGE IT UNLESS YOU INTRODUCE A NEW USER:PASSWORD PAIR AT THE **passwd** FILE AT THE DOCKER MOSQUITTO CONTAINER.
5. Tap on *Save* and the parameters will be sent to the NodeMCU (see Figure 5e).
6. After a few seconds, there are two possible behaviours of the device:
    - The RASv.1 is properly configured and it can start functioning. In this case, through the LEDs, a blinking blue light can be appreciated.
    - The RASv.1 has not been properly configured (one or more parameters are badly set) or the MQTT broker is not functioning (the Docker containers need to be running). For both options, the red LED will start blinking every second. If the MQTT broker is up, to reset the settings and be able of entering again the configuration portal, leave the device turned on for approximately 5 minutes. When the red LED stops blinking, go to the AP point again and repeat the steps.
7. Connect the RFID reader to the computer

(a) Connection of the RFID reader

(b) RFID reader

Figure 6: RFID reader usage and detail

8. At this point, you are ready to configure the attendances. To associate ID cards to the employees, log into Odoo and go to the employee form view.

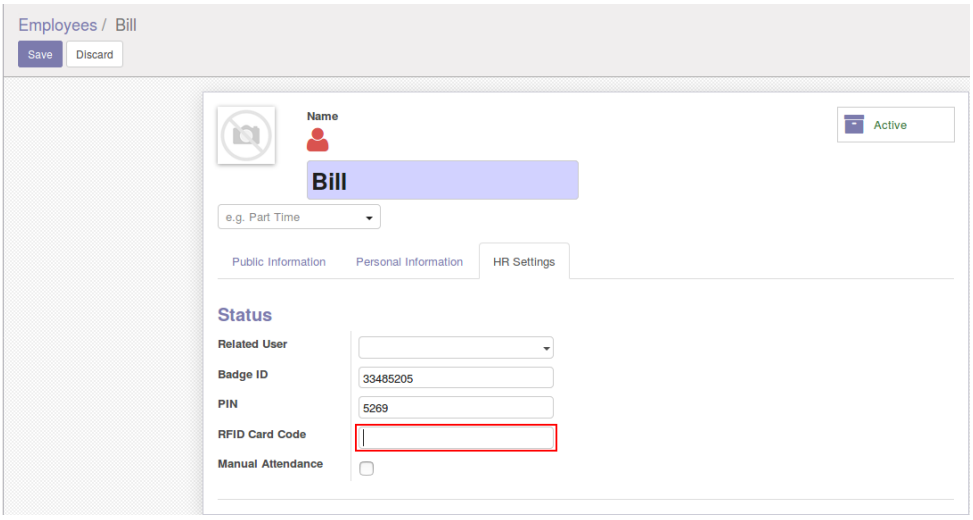


Figure 7: Odoo employee form view (empty *RFID Card Code field*)

Edit and put the cursor in the field RFID Card Code.

9. Pass the card/key over the RFID reader, so that its ID appears in the *RFID Card Code* field of the employee form view, and save the form.

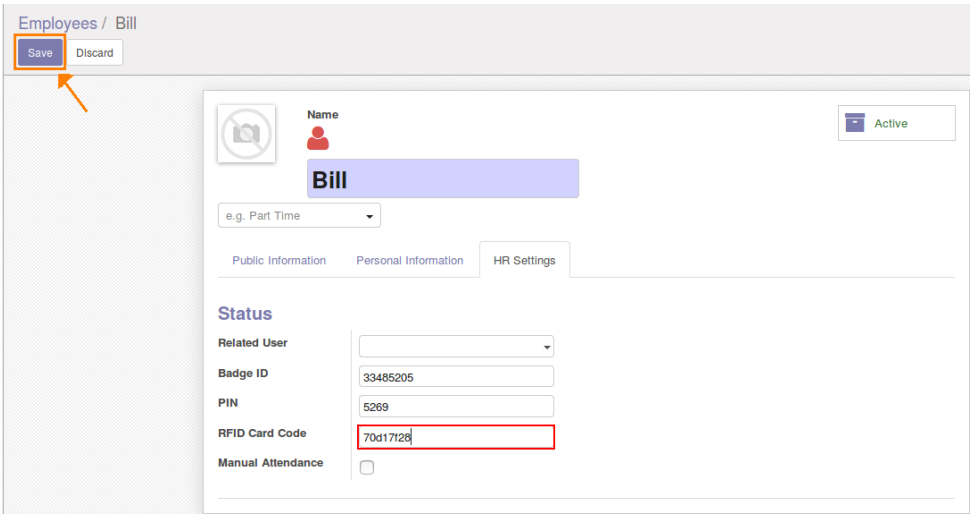


Figure 8: Odoo employee form view (filled *RFID Card Code field*)

10. Once the RFID ID has been associated to the employee, the card/key can be given to him/her (remember that this code must be unique for each employee). If at this moment the card/key is passed over the RASv.1, there are three possible behaviours.

    - The green LED blinks once while a sound is played. If the sound goes from a lower to higher tone, it is a *Check In*, but if it is the opposite way, a *Check Out* has occurred.

- The red LED blinks once, and a tone can be heared (in this case it is continuous, that is, the sound does not change while being played). It means that the RFID card passed over the RASv.1 is not registered at the system. If you used the card/key you just registered, enter Odoo again to look for any misspelling in the RFID ID.
- Finally, if the red LED blinks twice, and there is not sound, it means that the RASv.1 is not able to connect to Odoo. This can mean a problem with Odoo, so check that everything is properly configured and it is working fine. If Odoo works properly and the problem persists, write us to carry out a diagnosis.

At Odoo, by default the current day's attendances are shown, presenting the check in time and the check out time.

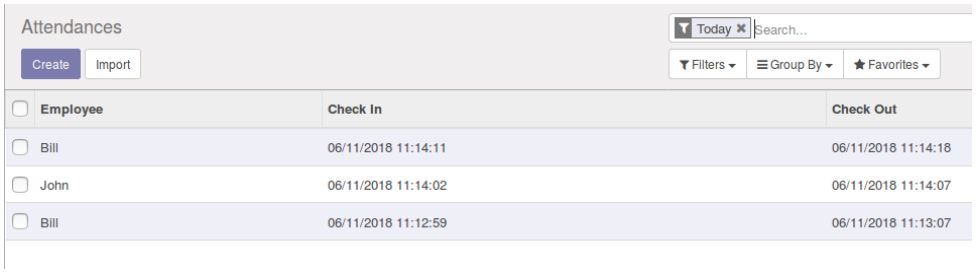


Figure 9: Odoo attendances sheet

4 Frequently Asked Questions

- ***I went through all the steps in the manual, but when passing the card/key, there is not blinking neither sound. What is happening?***

  The most common reason for this problem is malfunction in the MFRC522, possibly for the movement during the travel. To solve it, open the RASv.1 by unscrewing the big screws at the bottom of the device. Carefully extract the MFRC522 and put it again where it was located, tightening it. Close again the RASv.1 by screwing again the bottom screws.

- ***The LEDs blink correctly but the sound is not played, or vice versa. Is there any problem?***

  If you are facing this situation, it probably means that there is a problem with the LED/LEDs/buzzer or even the PCB. For problems with the LEDs, they can be easily changed by opening the RASv.1. In the case of the buzzer or the PCB, it would be a more difficult problem to solve. You can deliver the malfunctioning device to us, and we will immediatly deliver a new one to you.

- ***How can I update or modify the code in my RASv.1?***

The NodeMCU can be flashed using the Arduino IDE or PlatformIO. The codes are in the repositories indicated in the introduction.

References

[1] AES description. https://es.wikipedia.org/wiki/Advanced_Encryption_Standard.

[2] Base64 description. https://es.wikipedia.org/wiki/Base64.

[3] Docker Official Site. https://www.docker.com/.

[4] FreeCAD Official Site. https://www.freecadweb.org/.

[5] HMAC description. https://es.wikipedia.org/wiki/HMAC.

[6] Kicad Official Site. http://kicad-pcb.org/.

[7] MFRC522 Datasheet. https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf.

[8] NodeMCU Official Site. http://nodemcu.com/index_en.html.

[9] YWRobot Bredboard description. https://www.petervis.com/Raspberry_PI/Breadboard_Power_Supply/YwRobot_Bread.

[10] Gastón C. Hillar. *MQTT Essentials - A Lightweight IoT Protocol*. Packt, Livery Place, 35 Livery Street, Birmingham, B3 2PB, UK., 4 2017. The preferred IoT publish-subscribe lightweight messaging protocol.