Project Documentation IC216 Internet of Everything

Authors information

Authors:	Cyrill Weber	Florian Löw	Silvio Brändle
E-Mails:	cyrill.weber@noseryoung.com	florian.loew@noseryoung.com	silvio.braendle@noseryoung.com

Table of Contents

Table of Figures

1 Plan

During this project, we will be developing, testing, and documenting a system to remotely check the elevator presence and call the elevator with the intention of making the descent more efficient while eliminating waiting time.

1.1 Project Components

1. Elevator Integration Modules: ESP32 that reads the elevator’s presence and presses the call button.
   ⇨ Flo
2. User Interface Module: Device that displays the presence, with a button to call the elevator.
   ⇨ Cyrill
3. Broker Server: Interconnects the hardware elements.
   ⇨ Silvio

1.2 Elevator Integration Modules

These Modules consist of a light sensor to check if either of the elevators are present and a servo actuator to press the elevator call button.

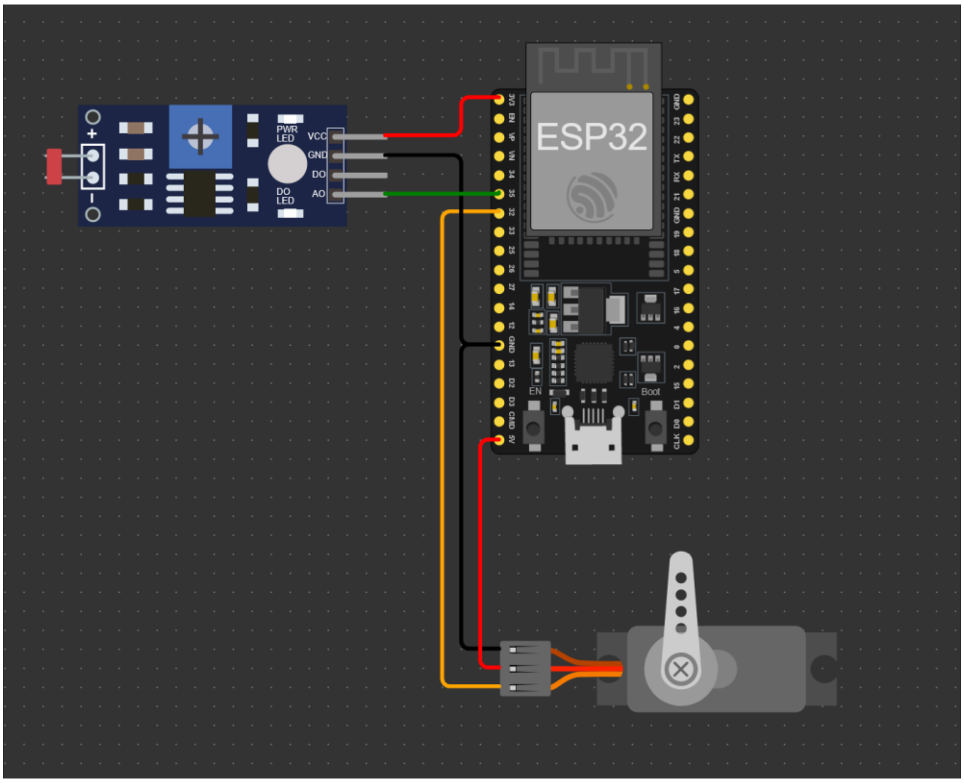


Figure 1: Wokwi Diagram of Integration Module

1.3 User Interface Module

The User Interface Module can be used to remotely check if an elevator is present and call it if necessary.

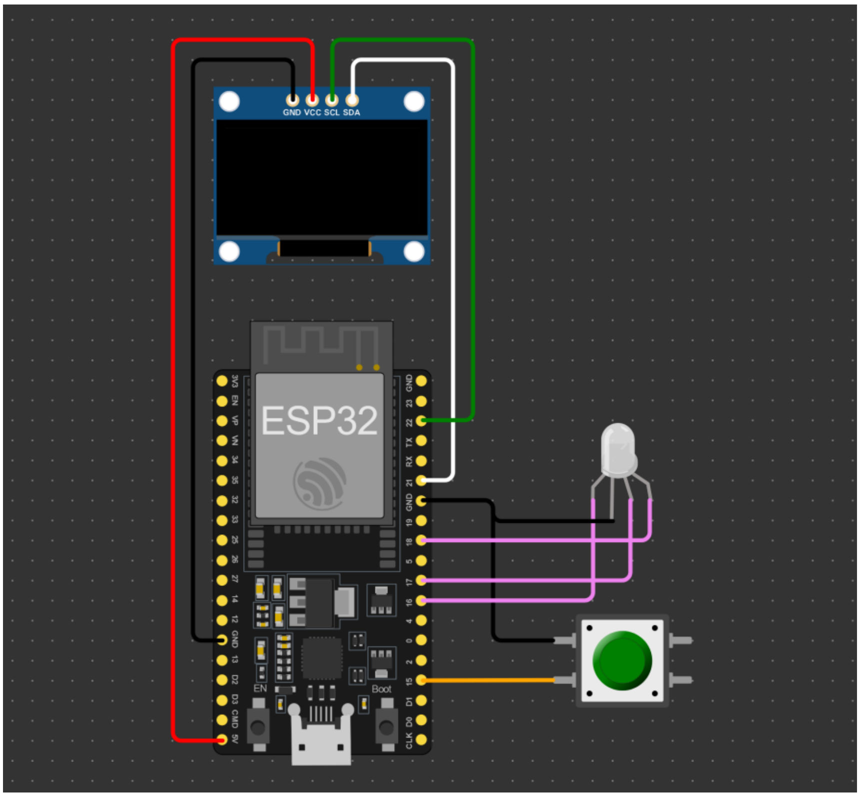


Figure 2: Wokwi Diagram of Interface Module

1.3.1 Concept

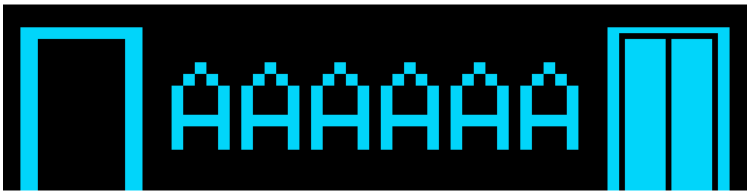


Table 1: Interface Display Concept

One of the outputs of the device is an OLED display which is used for displaying the state of the elevators as shown in **Error! Reference source not found.**. The LED is red when both elevators aren't present. When they have been called it gets yellow and when one of the elevators is there, it gets green.

A simple button is used as an input for calling the elevator.

1.4 Broker Server

The Broker Server is responsible for interconnecting the two Elevator Integration Modules with the User Interfviewace Module and offer a web interface to check and call the elevator from any device.

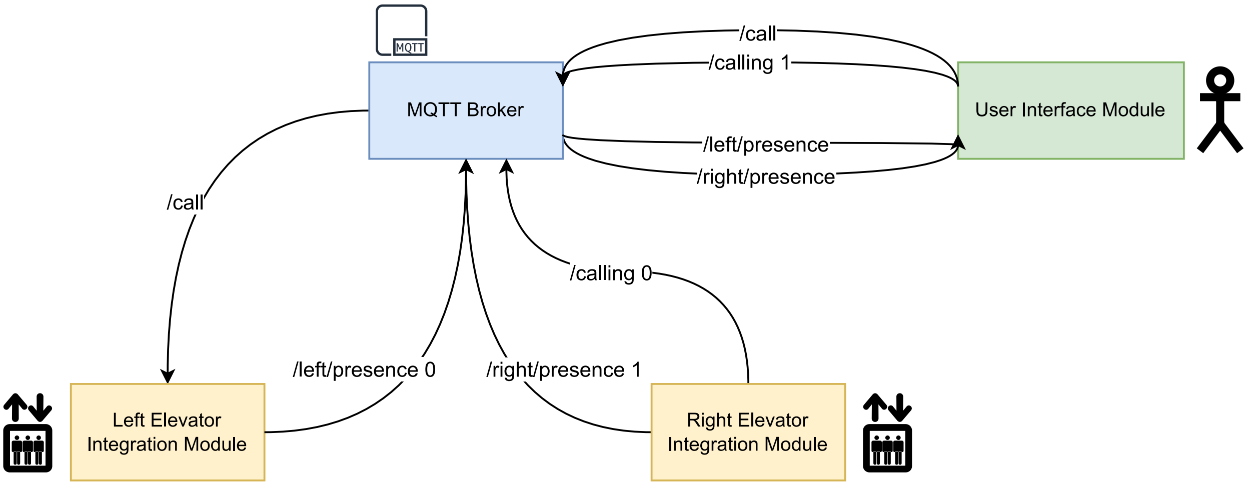


Figure 3: Data Flow Diagram

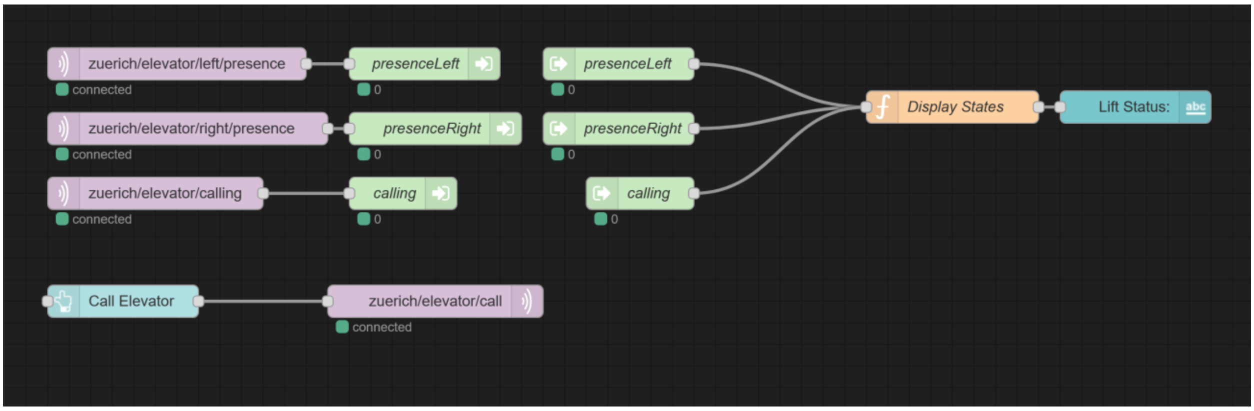


Figure 4: Node-Red Flow

1.4.1 Topics

All stateful topics are updated once the value changes. For eventful topics, a message is sent only once per event.

Topic			Description	Values
zuerich/elevator/	[left|right]/	presence	Describes if the elevator is present for the side	1|0
	calling		Describes if the elevator has been called	1|0
	call		Event for calling an elevator	–

Table 2: MQTT Topic Concept

2 Testing

To make sure all functions work as expected, this table allows us to check each of the requirements and edge cases.

Testcase-No.	1
Testcase-Description	Interface Module test
Requirements-No.	1
Test-Environment	Elevator room
Functionality	Testing of the Calling Button, the Display of the action, the Display of the Elevator, the Light
Testing Date	22.12.23
Tester	Cyrill Weber
Test-Steps:	

Nr.	**Action**	**Expected Outcome**	**Real Outcome**	**Successful**	**Comment**
1	Press the call button on the Interface Module	The display should show the text “Coming”, the button at the elevator should be pressed, and the interface light should be yellow.	As expected	Yes	
2	Wait until the elevator has arrived	The display should show the text “Here”, display an open elevator, and the light should be green.	As expected	Yes	
3	Send the elevator to another floor	The display should show the text “Gone”, the display should show a closed elevator, and the light should be red.	As expected	Yes	
4	Both elevators have arrived	The display should show the text “Here”, show two open elevators, and the light should be green.	As expected	Yes	
5	Call the elevator when one of them is already there	The display should show the text “Here” and on the Display there should be two open Elevators, the Light should be green, and the Elevator that is already there should open	The elevator opens, but it is set to the “Calling” status and not to “Here” and the light is yellow	No	

3 Display States

Here is a Demo of what the display can show:

3.1 Elevator is gone

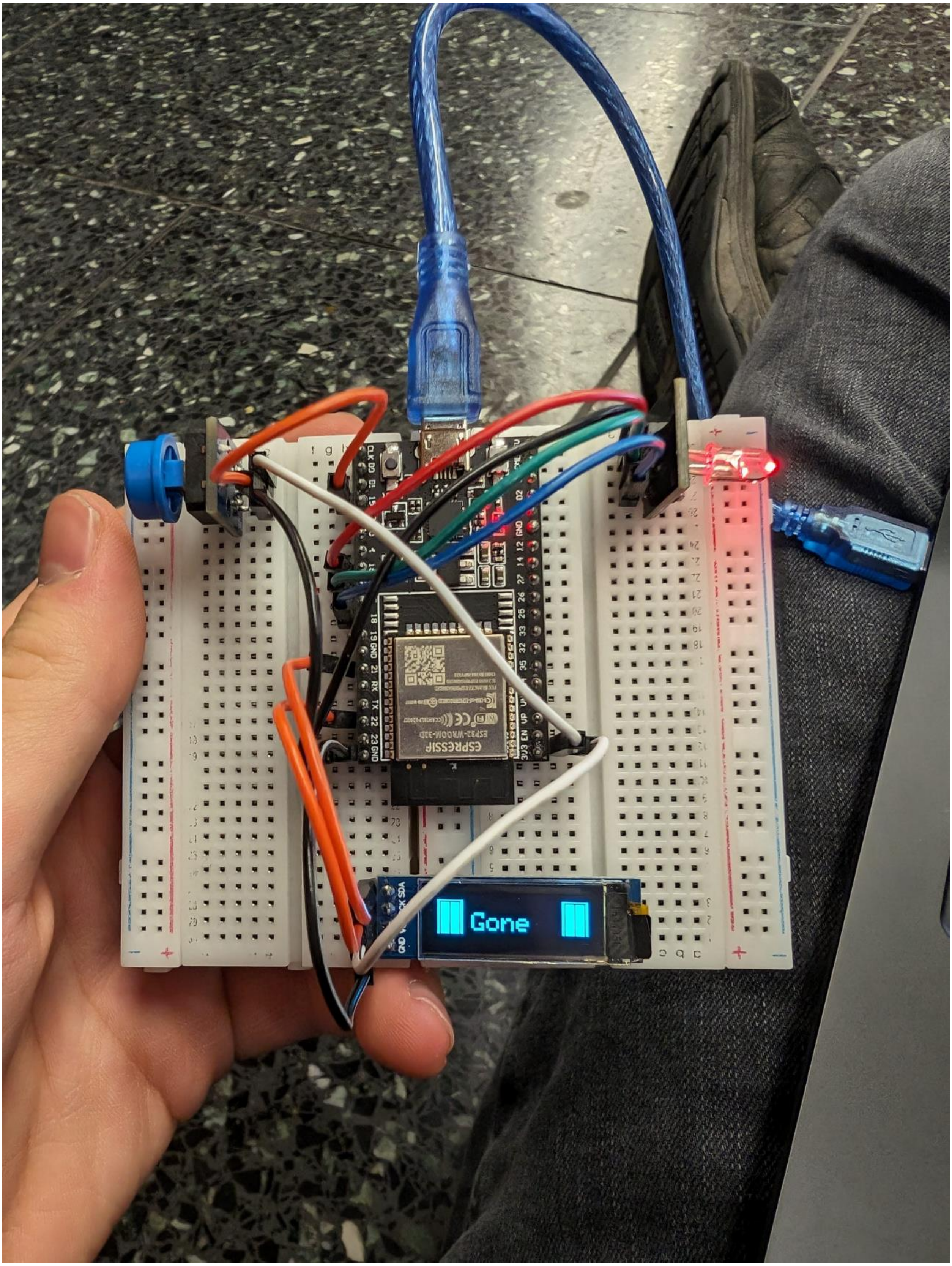


Figure 5: Interface Module indicating Gone

3.2 Elevator is coming

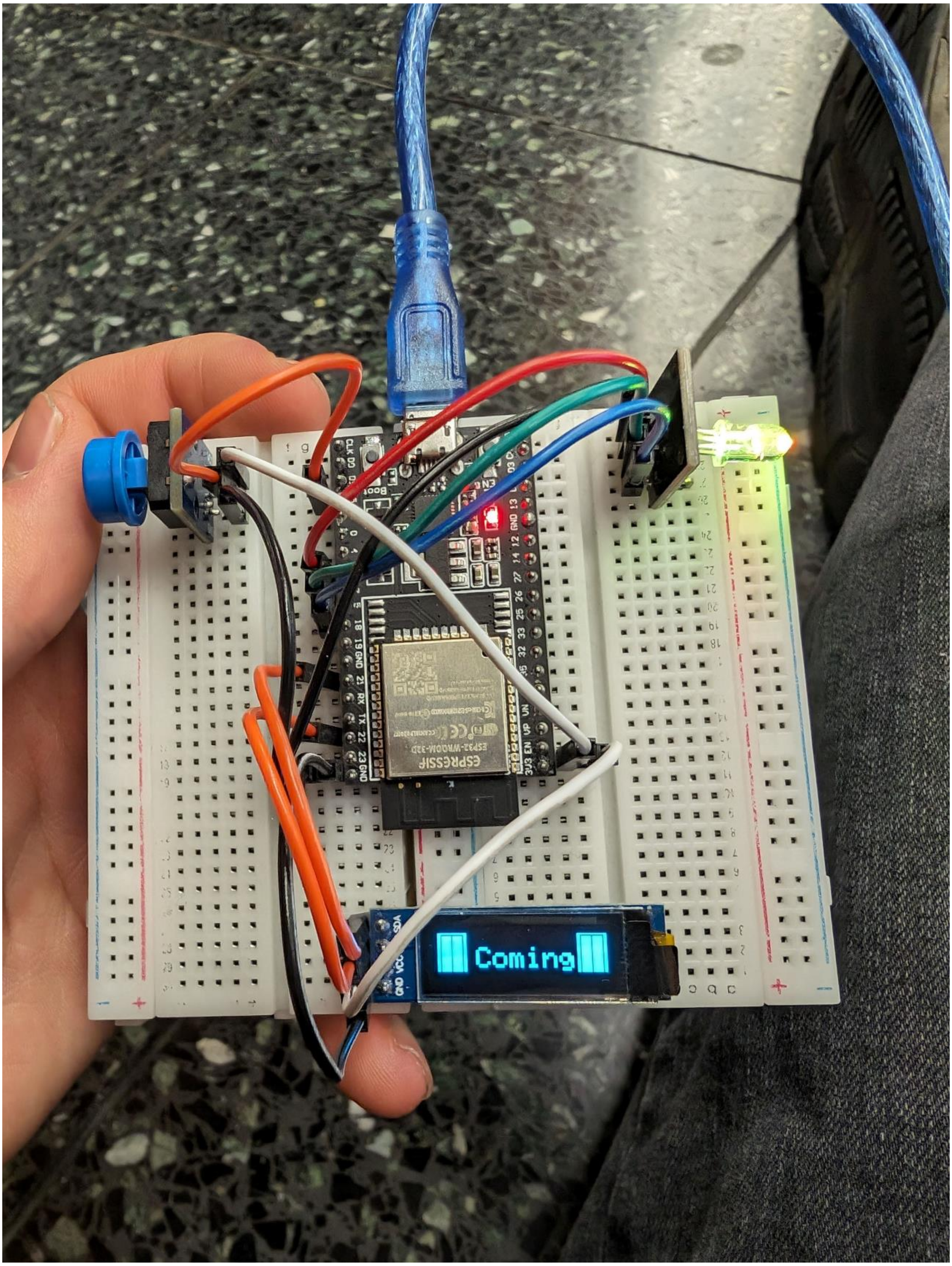


Figure 6: Interface Module indicating Coming

3.3 Elevator is here

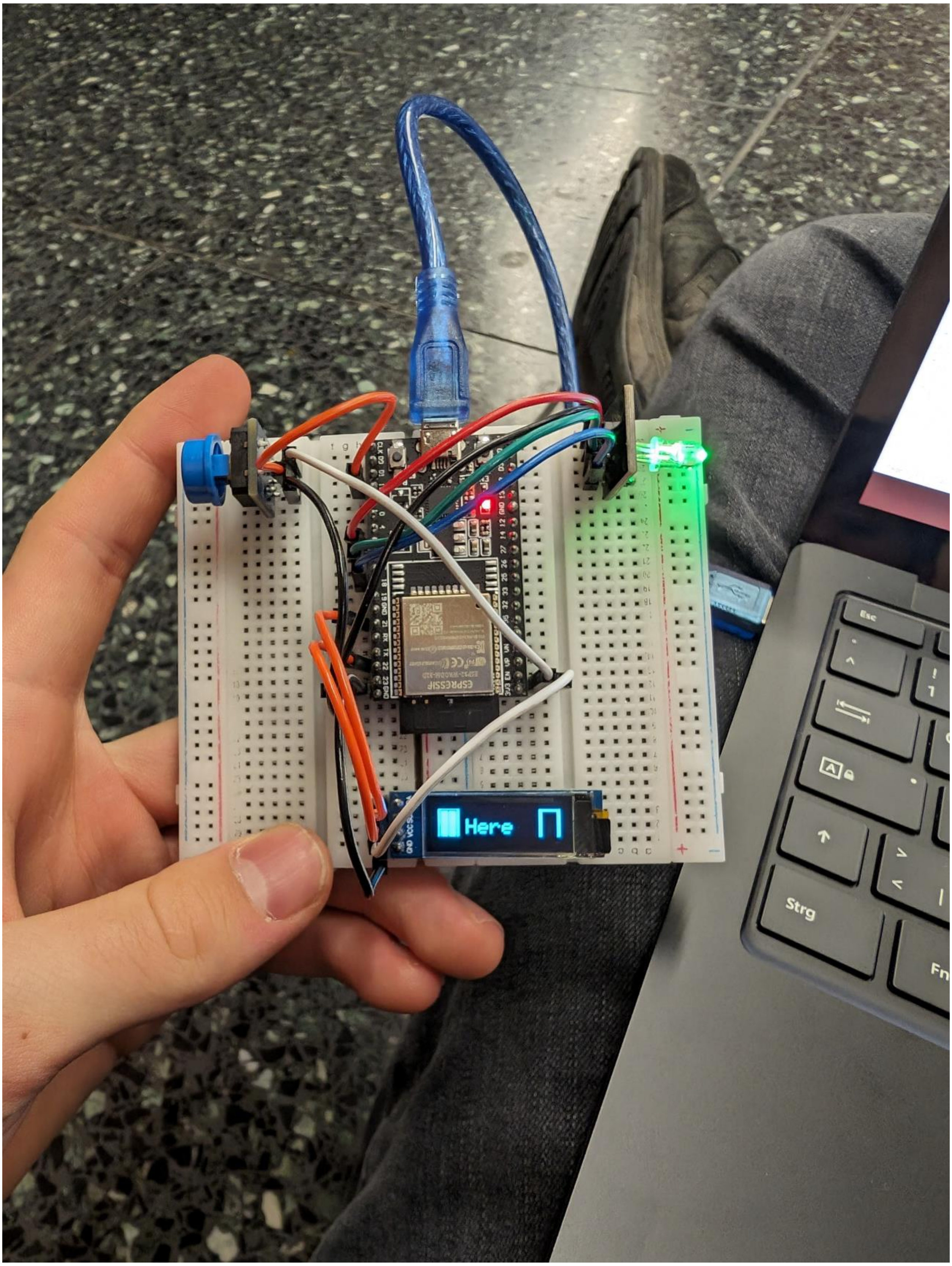


Figure 7: Interface Module indicating Here

4 Security

An essential part of IoT development is ensuring the security of the systems. Of course this project was built with security in mind. However, we weren’t able to guarantee full security because of limited resources and the rapid pace of development. The following table demonstrates the security considerations we had in mind:

Measure		Current Implementation	Possible Improvements
Use complex passwords	MQTT	The MQTT server is not password protected	Add password protection
	Wi-Fi	The Wi-Fi network is not password protected	Add password protection
	Node-RED	The Node-RED server uses a default password defined by Noser Young	Change the password and only use it for one thing.
	ESP32 WiFiManager	The Wi-Fi network for configuring the ESP32 doesn’t require a password	Define a password for the configuration portal.
Encrypt MQTT traffic		The traffic from and to the MQTT server are not encrypted.	Enable SSL encryption on the devices as well as on the MQTT server itself.
Use a separate network for the IoT devices		We didn’t use a separate network.	Set up a separate network.
Use up-to-date software	MQTT	Mosquitto V2.0.14 (newest V2.0.18)	Update the software
	Node-RED	V3.1.0 (newest V3.1.3)	Update the software
	PlatformIO	V6.1.11 (up to date)	–

Table 3: Project Security Considerations

5 Setup

5.1 Bill of Materials

- 27x Jumper Cables
- 3x ESP32
- 3x Micro-USB cable
- 3x Breadboards
- 2x Servomotors
- 2x Light sensors
- 1x Push button
- 1x SSD1306 OLED display
- 1x RGB LED

5.2 Configure Code

Before we begin with the setup guide, we will assume that you have a working installation of PlatformIO present on your system.

Here's a link to the PlatformIO installation guide:

https://platformio.org/install/ide?install=vscode

We also assume that you have already cloned our code from this GitHub repository:

https://github.com/Floew-NY/BLJ2023_uek216_team2_fahrstuhl

In our repository under src/config.h there is a configuration file which controls the following:

- Pin configurations
- Integration vs Interface Module
  - For Integration: Manual vs Automatic mode

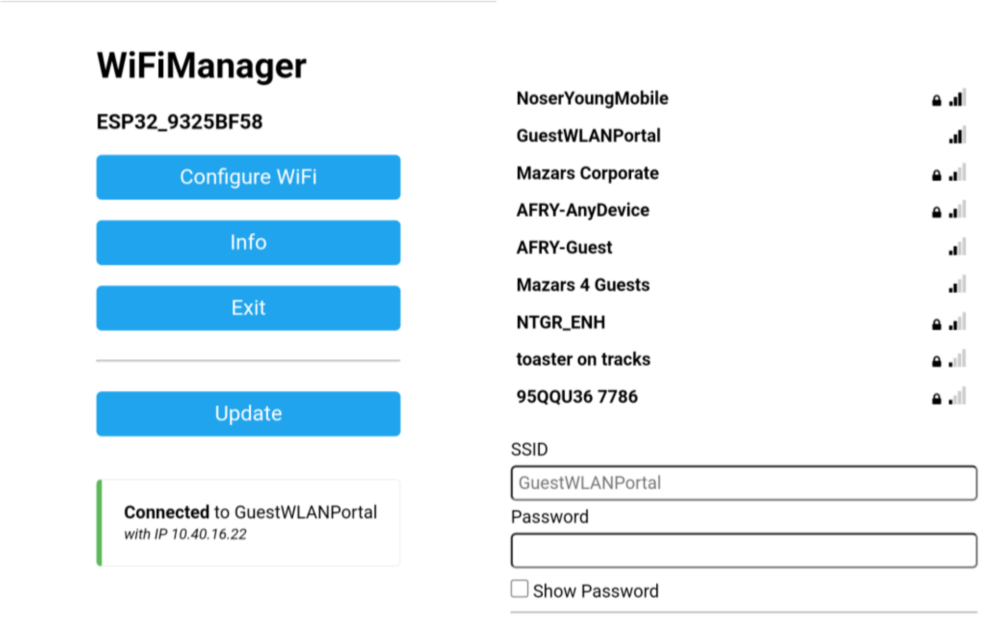
You will want to change the configuration before you flash any one of the ESPs.

5.3 Flashing

Now all you need to do is connect the ESPs one by one (potentially update configs) and hit upload in PlatformIO. Make sure to use the correct firmwares for the different devices.

5.4 On Device Configuration

Once the ESP has been flashed, press the boot button on the ESP and connect to the Wi-Fi, it's created using your phone. This should open a website like this. Hit configure and setup the internet.



WiFiManager

ESP32_9325BF58

Connected to GuestWLANPortal
with IP 10.40.16.22

NoserYoungMobile
GuestWLANPortal
Mazars Corporate
AFRY-AnyDevice
AFRY-Guest
Mazars 4 Guests
NTGR_ENH
toaster on tracks
95QQU36 7786

SSID: GuestWLANPortal

Password:

□ Show Password

Figure 8: WiFiManager Interface

5.5 Integration Module Specific

5.5.1 Wiring

For the Integration Module specifically, connect the components on a breadboard as shown in Figure 1.

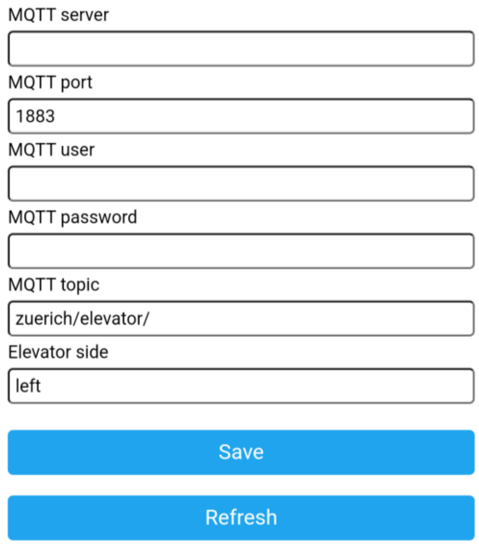
5.5.2 Configure Parameters

Configure these Parameters the same on all devices:

- MQTT server
- MQTT port
- MQTT topic

On each Integration Module, you have defined either left or right for the corresponding elevator.

Make sure you only have one left and one right elevator.



MQTT server

MQTT port: 1883

MQTT user

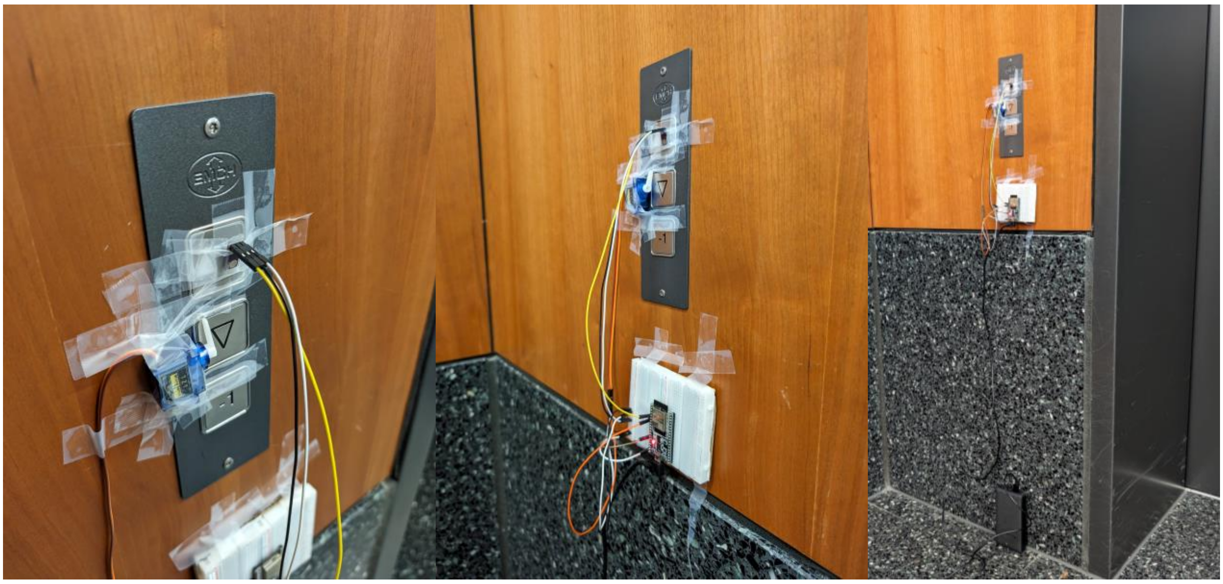
MQTT password

MQTT topic: zuerich/elevator/

Elevator side: left

Figure 9: WiFiManager Custom Parameters

5.6 Mounting



5.7 Interface Module Specific

5.7.1 Wiring

For the Interface Module specifically, connect the components on a breadboard as shown in Figure 2.

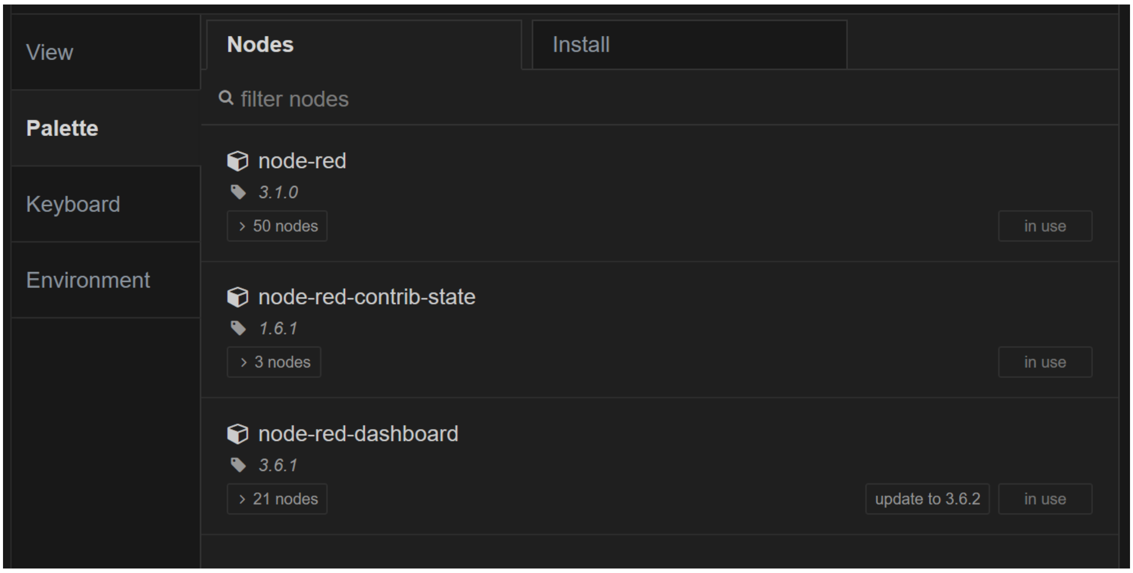
5.7.2 Configure Parameters

Do the same as in the Integration module, the device specific parameters can be left empty.

5.8 Node Red

5.8.1 Plugins

To make sure all nodes are imported correctly, you first need to install the following NPM packages to your Node-RED palette:



node-red
3.1.0
› 50 nodes — in use

node-red-contrib-state
1.6.1
› 3 nodes — in use

node-red-dashboard
3.6.1
› 21 nodes — update to 3.6.2 — in use

Figure 10: Node-RED Node Palette

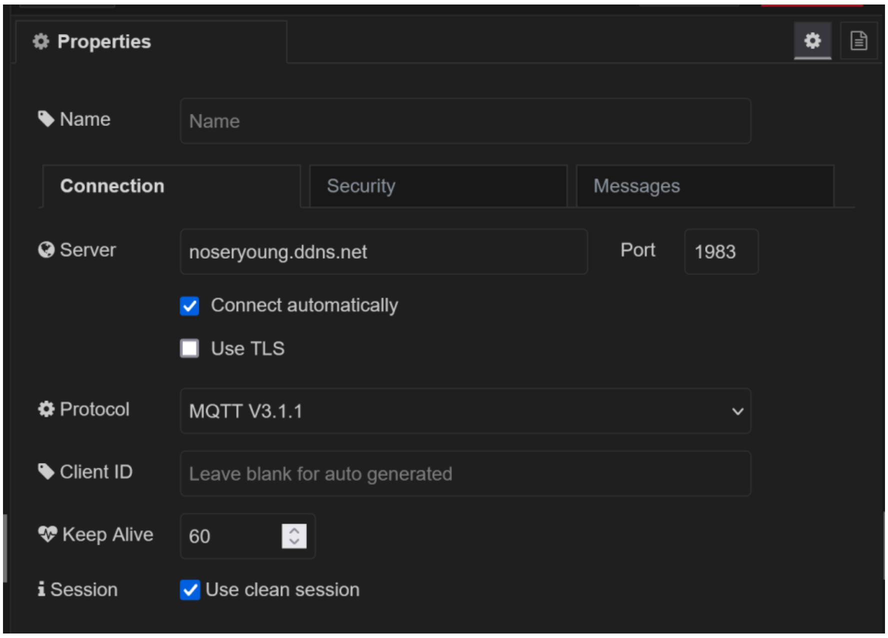
5.8.2 Importing

To import our preconfigured interface, use the “Import” button from the dropdown on the top right.

The file you want to import is “Node-Red.json” from our Project repository’s root.

5.8.3 Changing Host

To change the MQTT server host in Node-RED, you can navigate to the config tab and double click on the mqtt-broker node. In there you can configure host, port and credentials.



Name: Name

Server: noseryoung.ddns.net Port: 1983

☑ Connect automatically

☐ Use TLS

Protocol: MQTT V3.1.1

Client ID: Leave blank for auto generated

Keep Alive: 60

Session: ☑ Use clean session

Figure 11: Node-RED mqtt-broker Node Properties