



# Mauto Gateway Configuration Page Manual

Introduction .....	2
Accessing the Configuration Page.....	2
Layout.....	2
Pages .....	3
System Info .....	4
Access Point.....	4
Connection:.....	4
MQTT.....	5
HTTP .....	6
Addresses .....	6

## Introduction

The Mauto Gateway is capable of communicating from any industrial device with rs485 Modbus capability, and connecting through Ethernet or Wi-Fi to the internet to send data to the Mauto cloud platform through HTTP or MQTT (or any 3<sup>rd</sup> party platform through MQTT). To configure the parameters used by the device, the configuration page should be accessed.

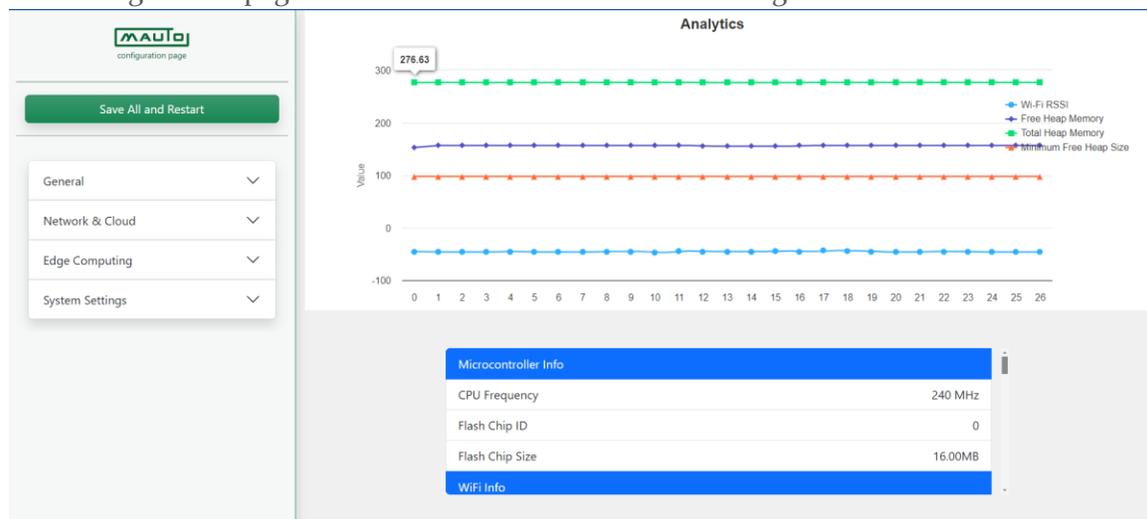
## Accessing the Configuration Page

The configuration page can be accessed through one of the two following methods:

1. Pressing the restart button for more than 1s, till the power led blinks, then connecting to the Wi-Fi access point coming from the gateway, then accessing the following IP address from any browser: <http://192.168.4.1>, or using the hostname <http://mauto.local> .
2. After the gateway is connected to a local area network, the configuration page can be accessed using the IP that the gateway has received. For example, <http://192.168.1.100>. The hostname <http://mauto.local> can also be used if the device is connected to a Wi-Fi local network on devices supporting MDNS.

## Layout

The configuration page that will be accessible is the following:

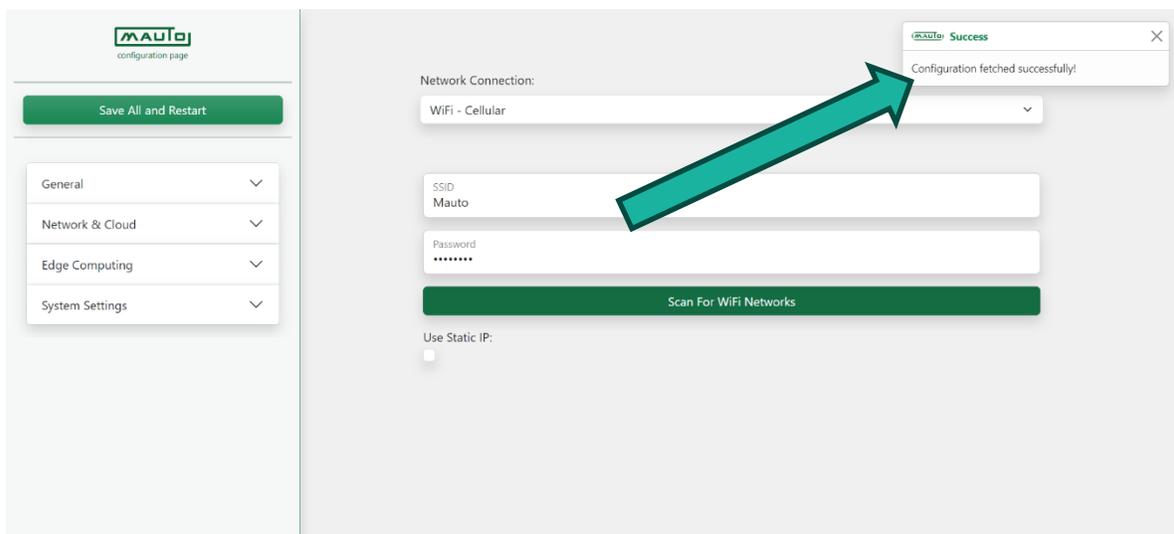


The page is divided into the following sections:

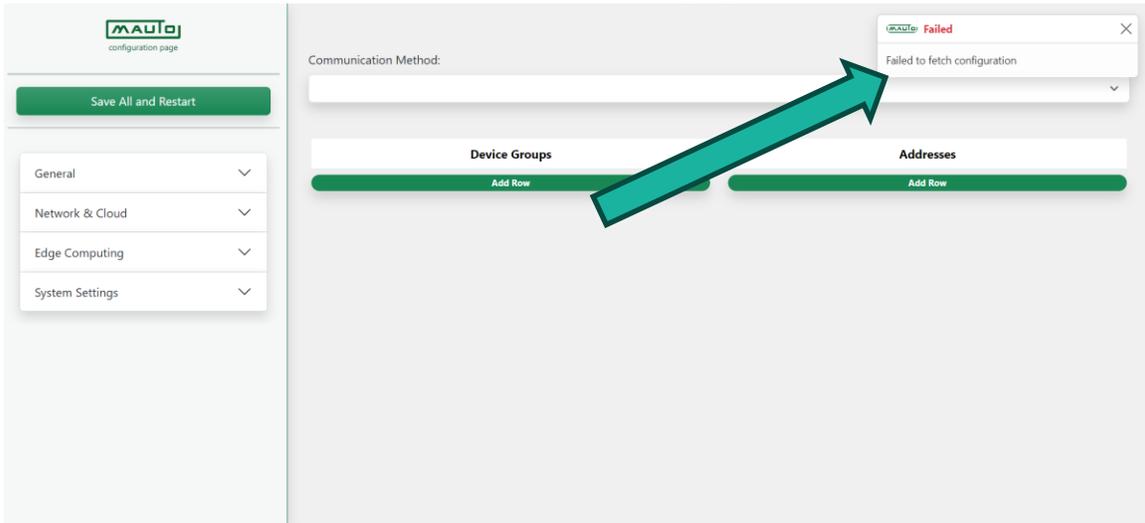
- **General:** Configure and monitor general parameters
  - **System Info:** Monitor General parameters such as free memory and connectivity.
  - **Access Point:** Configure the Wi-Fi access point SSID and Password.
- **Network & Cloud:**
  - **Connection:** Setup Wi-Fi/Ethernet Connectivity
  - **MQTT:** Configure MQTT parameters.
  - **HTTP:** Configure HTTP parameters.
- **Edge Computing:** Define how data will be collected and sent
  - **Addresses:** Set up the connected device, and sensor parameters
- **System Settings:**
  - **Debug:** Set up monitoring through MQTT
  - **Update:** Perform updates such as configuration import/export and firmware upgrade

## OPENING OF CONFIG PAGE

When the configuration page is loaded, the page will fetch for the current configuration file inside the gateway. If the configuration fetch is successful, a successful notification will be displayed:



If the fetch was unsuccessful, a failed error will be displayed:



## PAGES

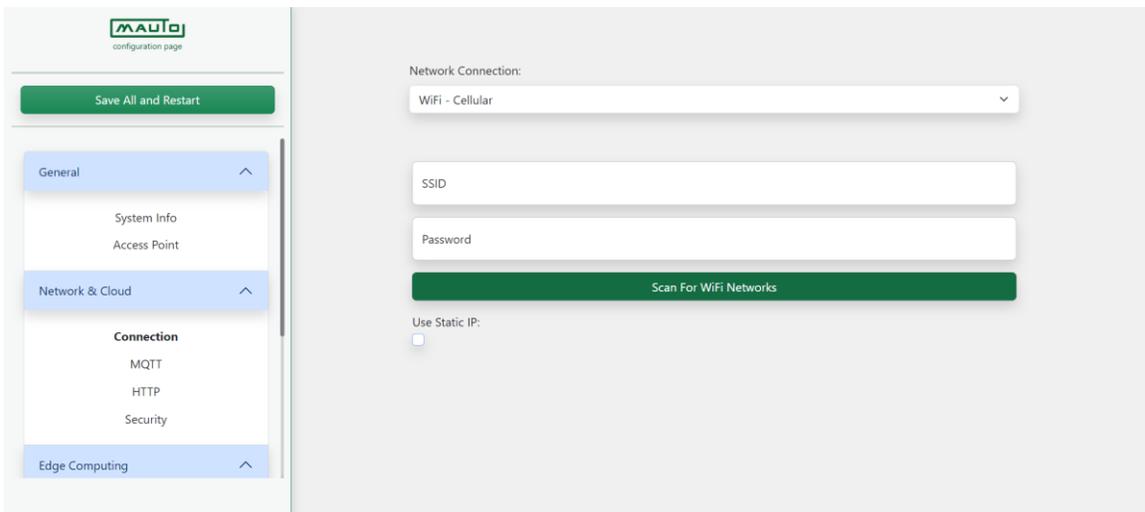
### System Info

This page will contain an *analytics* graph showing the Wi-Fi connectivity and memory progression. Below it, there will be other information of the gateway.

### Access Point

Configuration of the SSID and Password of the Access Point of the Gateway

### Connection:

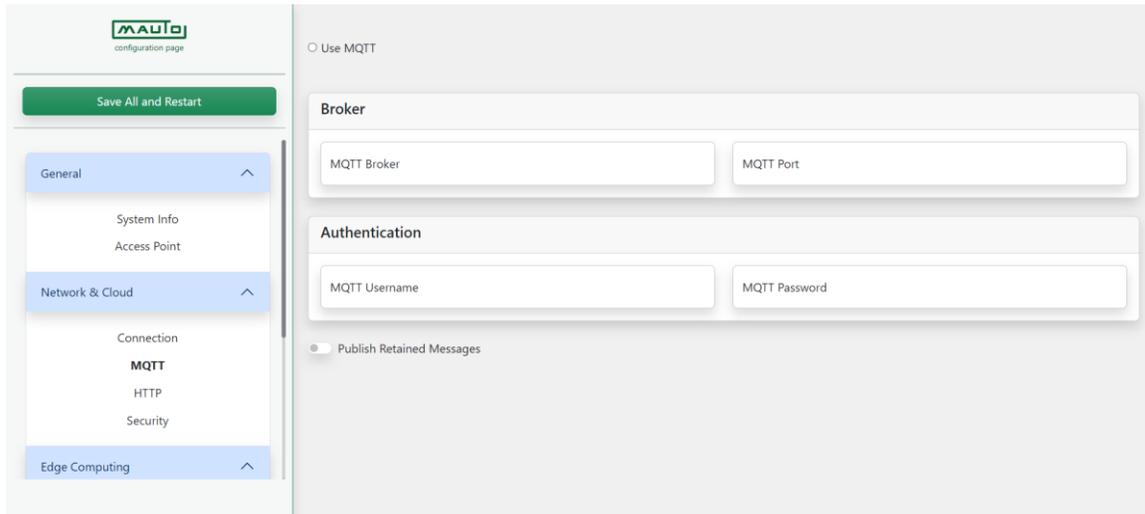


Two connection options are available:

- WiFi – Cellular: Connect to WiFi router or WiFi enabled cellular module
- Ethernet: Connect to router via ethernet

In the Wi-Fi – Cellular section, it is required to input the SSID and Password of the Wi-Fi network. Use the [Scan for Wi-Fi Networks](#) button to automatically scan for available Wi-Fi networks. If scan was successful, a success notification will appear, if the scan was unsuccessful, a failed notification will appear.

## MQTT



The screenshot shows the MQTT configuration interface. On the left is a navigation sidebar with a 'MAUI configuration page' header and a 'Save All and Restart' button. The sidebar has three main sections: 'General' (containing 'System Info' and 'Access Point'), 'Network & Cloud' (containing 'Connection', 'MQTT', 'HTTP', and 'Security'), and 'Edge Computing'. The 'MQTT' option is selected. The main content area has a radio button for 'Use MQTT' which is currently unselected. Below this are two sections: 'Broker' with input fields for 'MQTT Broker' and 'MQTT Port', and 'Authentication' with input fields for 'MQTT Username' and 'MQTT Password'. At the bottom of the main area is a toggle switch for 'Publish Retained Messages' which is currently turned off.

Click on [Use MQTT](#) to use MQTT for data communication. The required parameters to input are:

- **MQTT Broker:** address of the MQTT Broker, *eg: broker.hivemq.com*
- **MQTT Port:** port of the MQTT connection, *eg: 1883 (default mqtt port)*
- **MQTT Username and MQTT Password:** for authentication, optional parameters.
- **Publish Retained Messages:** if the switch is on, all mqtt communicated sensors will be sent as retained messages.

## HTTP

The screenshot shows the MAUTO configuration page for HTTP settings. The left sidebar contains a navigation menu with categories: General, Network & Cloud, and Edge Computing. The main content area is titled "Use HTTP" and includes a "Server" section with input fields for "HTTP Server" and "HTTP Port". Below that is an "Authorization" section with tabs for "Basic" and "Bearer". The "Bearer" tab is selected, showing input fields for "Username", "Email", and "Password". A "Save All and Restart" button is located at the top left of the configuration area.

Click on **Use HTTP** to use HTTP/S for data communication. The required parameters to input are:

- **HTTP Server:** address of the HTTP server. *eg: iotapi.mautoiot.com*
- **HTTP Port:** port of the MQTT connection, *eg: 80 (HTTP), 443 (HTTPS)*
- **In Authorization -> Bearer Section:** For authorization with the mauto server, input the *username, email, and password* provided by mauto.

## Addresses

The screenshot shows the MAUTO configuration page for "Addresses". The left sidebar contains a navigation menu with categories: Edge Computing, Addresses, and System Settings. The main content area shows a "Communication Method" dropdown menu set to "Modbus RS485". Below this are two tables: "Device Groups" and "Addresses". Each table has a header row with a number, a text input field, and icons for edit, copy, and delete. The "Addresses" table has a row with the number "1" and the text "New Address". Both tables have an "Add Row" button at the bottom. A "Save All and Restart" button is located at the top left of the configuration area.

In this page, all the connected devices will be setup. **Communication Method** box will contain the available protocols to communicate with (*modbus rs485*).

## Editing options



Edit



Duplicate



Delete

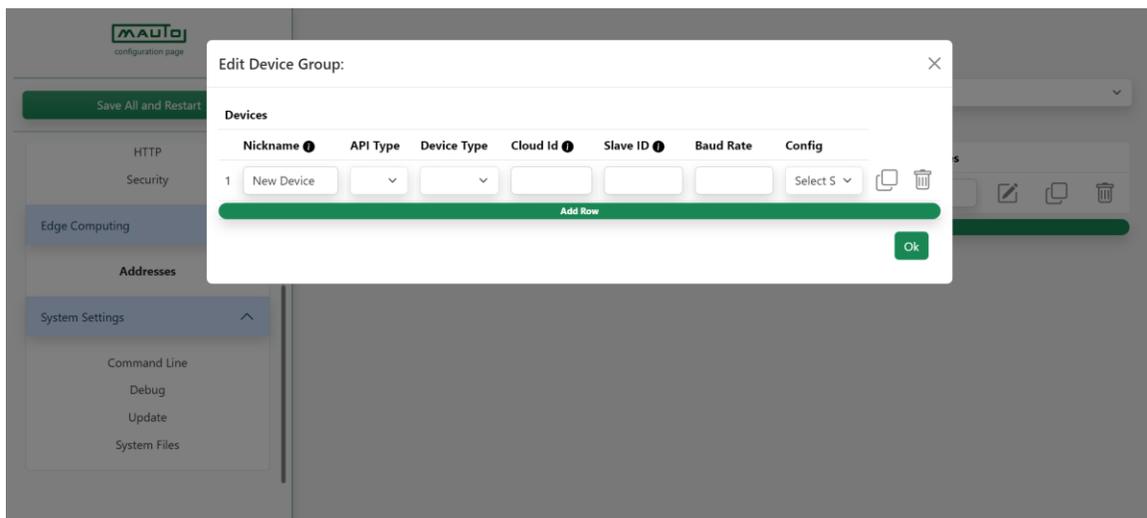
## Device Groups

A *device group* will contain multiple devices each with its communication parameters. All devices within a *device group* will share the same sensor (register) table. To edit the sensor table for a device group, click on its selector . A nickname can also be assigned to each device group. By clicking on the **edit** button, individual device groups can be edited.

## Addresses

Will contain the sensor list for the selected *device group*. A nickname can be assigned for each sensor. Clicking on the **edit** button will bring the parameters to edit each sensor.

## Device Group Edit



The above dialog will appear when clicking on the [edit](#) button for each device group. The following parameters are required for input:

- **Nickname:** nickname of each device
- **API Type:** options are: *meter*, *generator*. (use this in case HTTP is chosen or if alarms are needed for a generator).
- **Device Type:** used for generator Alarms.
- **Cloud ID:** Id for each device
- **Slave ID:** Modbus slave id of device
- **Baud Rate:** rs485 baud rate
- **Config:** define the data configuration, options are in the form ABC, where:
  - **A:** Data Bits
  - **B:** Parity, options are: *Even (E)*, *Odd (O)*, *None (N)*, *Mark (M)*, *Space (S)*
  - **C:** Stop Bits

Click on [Add Row](#) to add a new device.

## Sensor Edit

The image displays two screenshots of the MAUTO configuration page, specifically the 'Sensor Edit' dialog box for a new address. The background shows the 'Addresses' table with a 'New Address' row and an 'Add Row' button.

**Top Screenshot (Initial Form):**

- Topic:** Input field
- Cloud ID:** Input field
- Data Type:** Signed Int 64
- Function Code:** 4 - Read Input Registers
- Register Address:** Input field
- Multiplier Value:** Input field
- Offset Value:** Input field
- Equation:**  $new\ value = old\ value * Multiplier\ Value + Offset\ Value$
- Extract Bit:**  Extract Bit 1
- Extracted Bit Number:** Input field
- Synchronize:**  Synchronize 1

**Bottom Screenshot (Synchronize Section):**

- Synchronize:**  Synchronize 1
- Link To Address:** Dropdown menu
- Translation Table:**

Edge Value	Cloud Value
<input type="text"/>	<input type="text"/>

**Delete** (button) **Add Row** (button)
- Edit API:**  Edit API
- Configure Body Template:**

```
{
  } +
```
- Ok** (button)

Input fields are:

- **Topic:** input this field if MQTT is chosen
- **CloudID:** ID for sensor
- **Data Type:** Options are:
  - Unsigned Int 16 bit
  - Signed Int 16 bit
  - Unsigned Int 32 bit
  - Signed Int 32 bit
  - Float 32 bit
  - Unsigned Int 64 bit
  - Signed Int 64 bit

- Custom: input with it the number of registers (1 register = 16bit)

Data Type Custom (String) ▼	Num. of Registers
--------------------------------	-------------------

- **Function Code:** Options Are:
  - 1 - Read Coil Status
  - 2 - Read Input Status
  - 3 - Read Holding Registers
  - 4 - Read Input Registers
  - 15 - Write Multiple Coils
  - 16 - Write Multiple Registers
- **Register Address:** the modbus register address
- **Multiplier and Offset Values:** manipulate the read or written value such that:
  - If Read (FC = 1,2,3,4):  $CloudValue = ModbusValue * Multiplier + Offset$
  - If Write (FC = 15,16):  $ModbusValue = CloudValue * Multiplier + Offset$
- **Extract Bit:** Available only for read operations (FC = 1,2,3,4). Extract a specific bit from the register.
- **Synchronize:** If checked, the gateway will save the sent value in the RAM, then will only send the next value if it is changed. If linked to another sensor using the [Link To Address](#) box, both sensors will act as one sensor. For example, if a controller mode reading and writing sensors are saved as different sensors and linked to each other, if the reading sensor value is changed from *OFF* to *MAN*, the writing sensor will remember that the last value was *MAN*, not *OFF*.
- **Translation Table:** Convert an edge value to cloud values. For example, in the following table, if the collected value was 0, *OFF* will be sent, if collected value was 1, *ON* will be sent, if neither, the collected value will be sent as it is:

**Translation Table** ⓘ

Edge Value	Cloud Value	
0	OFF	Delete
1	ON	Delete
Add Row		

- **Edit API section – Configure Body Template (for MQTT only):**

### Configure Body Template

```
{
  key:  ~ -
} +
```

```
{
  "key": "value"
}
```

Fill to send the body of a sensor in a certain JSON format. Click on **+** to add a key-value pair, **~** to erase the value of a key-value pair, and **-** to delete a key-value pair.

**IMPORTANT NOTE:** When filling the *topic* fields or the *body templates*, placeholders can be added to be replaced by the firmware when performing the data communication:

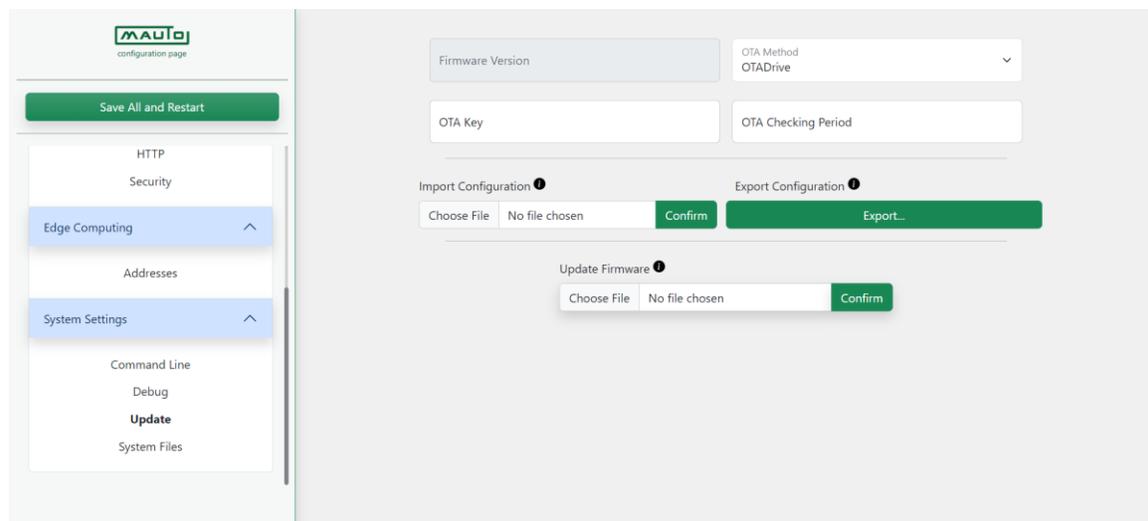
Placeholder	Replaced By
<b>\$deviceId</b>	CloudID of associated devices.
<b>\$addressId</b>	CloudID of sensor
<b>\$value</b>	Collected Value From Slave Device
<b>\$timeStamp</b>	Curent Device Time of Format: YY-MM-DDTHH:MM:SS

## Debug

In this section, choose to send debug information through MQTT. Input the following fields:

- **MQTT Broker**
- **MQTT Port**
- **MQTT Username and Password** (optional)
- **Serial MQTT Topic:** debug information will be sent to this topic
- **Data MQTT Topic:** send commands to the gateway through this topic. Available commands are: *restart* for restarting the gateway and *params* to relay the content of the current saved configuration file to the selected Serial MQTT topic.

## Update



The screenshot shows the MAUIO configuration page. On the left is a sidebar with a 'Save All and Restart' button and a menu with categories: HTTP, Security, Edge Computing (expanded), Addresses, System Settings (expanded), Command Line, Debug, Update (highlighted), and System Files. The main content area is titled 'Update' and contains three sections: 1. 'Firmware Version' (input field) and 'OTA Method' (dropdown menu set to 'OTADrive'). 2. 'OTA Key' (input field) and 'OTA Checking Period' (input field). 3. 'Import Configuration' section with a 'Choose File' button, 'No file chosen' text, and a 'Confirm' button. 4. 'Export Configuration' section with a green 'Export...' button. 5. 'Update Firmware' section with a 'Choose File' button, 'No file chosen' text, and a 'Confirm' button.

In this section you can configure and perform updates to the device. The **OTA Method** box will have the options: *OTADrive*, *Mauto*. If *Mauto* is chosen, OTA updates are performed using the Mauto proprietary server. If *OTADrive* is chosen, the OTADrive third party service is used to do the OTA updates. The *OTADrive* option requires an **OTA Key** provided by the service. For both methods, the default checking period is every 1hr if no input is provided, else the checking is done in the period provided by the **OTA Checking Period** field (input in seconds).

In the import configuration section, import a .json file existing in the local machine, and click on **Confirm** to do the actual import operation.

Use the **Export** button to export the current .json configuration file.

In the update firmware section, choose a .bin firmware file, and click on **Confirm** to perform a local firmware update.

## Saving The Configuration to Gateway

After filling all the needed fields press on the **Save All and Restart** button, then **Yes** to save the configuration.

