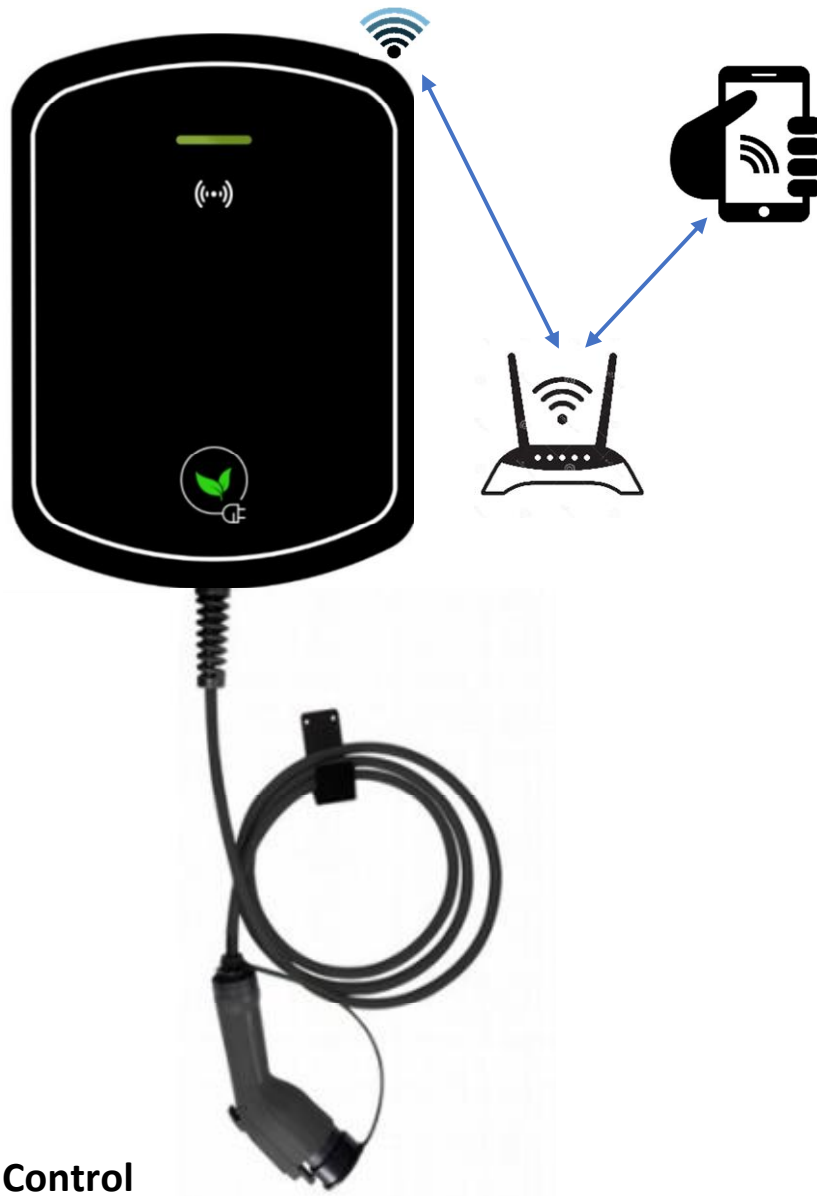


## Mobile-App development specification for visualisation and parametrisation of a Wallbox for E-Mobility



### Document Control

#### Revision History

Title	Mobile-App development specification for visualisation and parameterize of a Wallbox for E-Mobility
Project Name	Wallbox
Document Revision	00.03
Date	04/05/2022
Author	Joergens Michael, Sandfort Christoph

## Inhalt

<b>Document Control</b> .....	1
1. Introduction.....	3
2. Overview of the Hardware .....	4
3. Software specification .....	6
3.1 Structure of REST-API interface:.....	6
3.1.1 Read status parameter .....	6
3.1.2 Write and change Parameter: .....	7
3.1.2.1 set charging current .....	8
3.1.2.2 free charging.....	9
3.1.2.3 switch on RFID learning mode.....	9
3.1.2.4 Change Password .....	9
3.1.2.5 Wi-fi configuration .....	9
3.1.2.6 LAN configuration.....	10
3.1.2.7 Show charging history .....	10
3.2. Parameter calculations.....	10
3.2.1 Costs of Charing session .....	10
3.2.2 total cost of current delivered through wallbox .....	11
3.2.3 average power delivered to the car .....	12
3.2.4 cost savings compared to a combustion engine for actual charging session .....	13
3.2.5 total cost savings compared to a combustion engine .....	14
3.2.6 calculation of savings of CO2.....	14
3.2.7 total savings CO2 .....	17
3.2.8 Assign a Name to a RFID-UID.....	18
4. Design .....	20
4.1. Main-Page/Dashboard/startscreen.....	20
4.2. Calculated Values .....	21
4.3. Settings .....	22
4.4. System information .....	24
4.5. Help and FAQ.....	25
4.6. Initial start Screen.....	26
4.7 Things to clarify .....	27
<b>Revision Changes</b> .....	28

## 1. Introduction

To reduce the GHG emission and reduce climate change, governments of many countries (e.g. Germany) has set a goal of increase sales of Electrical vehicles (EV) by 2030. As the increase of EV share, the increase deployment of charging stations is necessary. Currently, the fully functional charging stations are far less than adequate for the EV charging. To be part of this market and to help distribute charging points for electrical cars the Saventor GmbH wants to distribute wallboxes for EV. To be smart and have the possibility to interact with the wallbox it is necessary to control the wallbox by a mobile-app. Therefore, this document is meant to define the generic software functionality for the mobile-app.

At first the hardware will be described to roughly understand what it is about.

The second part of the specification will describe in detail the interface of the Wallbox and how to get access to the parameters.

The third part of the specification will describe a few design ideas and the minimum content of the visualization.

## 2. Overview of the Hardware

Basically, the wallbox is an Interface between the power grid and the electric vehicle (EV). The wallbox supplies the car with electrical power. To do so it is necessary to have an intelligent charging controller which communicates with the car to handle all the charging important things like charging current, maximum power, failure management in case of critical electrical errors, authentication and much more charging relevant stuff. Pictorial it is the brain of the wallbox which includes all the intelligence.

Within the Saventor Wallbox a charging controller CC613 from Bender GmbH is used. This controller can be parameterized to obtain the desired loading behavior and it is possible to read out many internal values like charging current.

The charging controller can be accessed by different interfaces. One interface is a Wi-Fi-connection with an internal USB Wi-Fi adapter. The charging controller and therefore the wallbox can be connected to the local Wi-Fi.

The following picture shows the charging controller which is installed within the wallbox.



Figure 1: Charging controller

To avoid unauthorized charging, the wallbox has an internal RFID-Reader integrated. The owner of the wallbox is able to authorize himself with his own RFID-Chip. By default two RFID-Chips will be delivered with the wallbox to the customer to authorize and start the charging process. The RFID-UID of each RFID-Chip is stored in a list (so called Whitelist). To authorize and start the charging process (when car is connected to the wallbox) the RFID-Chip needs to be hold in front of the RFID-Sign:



Figure 2: RFID-AuthORIZATION

The charging cable to connect a car to the wallbox is a fixed mounted Type2 charging cable. Like the picture shows below.



*Figure 3: Charging cable*

An internal failure detection device controls if there is a dangerous current flowing throughout the wallbox or the car. It is placed within the wallbox and directly connected to the charging controller. The device is called RCMB, and it delivers the actual failure current and the information if the failure current exceeds a critical current.



*Figure 4: RCMB Device*

### 3. Software specification

The software should be programmed for Android and Apple.

To get access to the parameter of the charging controller a REST-API interface is implemented within the charging controller. As described above the controller is connected to the local Wi-Fi with the USB to Wi-Fi adapter.

The REST-API interface describes the access to the parameters of the charging controller within the local Wi-Fi.

The App should use the REST-API interface to handle the parameter. The interface is described in detail below:

#### 3.1 Structure of REST-API interface:

The following chapter will describe in detail the REST-API interface. The first chapter will show how to read out status parameters. The second chapter will show how to set and change parameters.

##### 3.1.1 Read status parameter

The REST-API allows to get detailed information about the state of the charging controller. With the following URL, it is possible to observe the output of different parameters.

The structure of the URL is like:

`http://IP-Address-of-charging-controller/rest/parameter_name`

For Example, `http://192.168.178.55/rest/conn_state` will deliver the information of the connection state of the wallbox. The following feedback is returned: `no_vehicle_connected`

```
C:\>curl -get 192.168.178.55/rest/conn_state
no_vehicle_connected
```

The following table shows which parameters should be read out and what the return value is.

parameter_name	Key values or example	Comment
conn_state	no_vehicle_connected vehicle_connected_type2 vehicle_charging_type2 vehicle_connector_error	conn_state shows the connection status of the wallbox. Is a car connected and charging started or not and is there a critical error?
auth_state	not_authorized_for_charging wait_for_auth authorized_for_charging auth_timeout	
auth_uid	e.g: 046e453af012ca3	Shows which RFID-UID (which RFID-Chip) is currently authenticated to the wallbox.
time_since_charging_start	e.g.: 2127	Time since start of charging in seconds. Should be calculated in h:mm:ss
meter_wh	e.g.: 426416	Total Energy delivered with the Wallbox. Should be displayed in kWh (meter_wh/1000)
power_w	e.g.: 3600	Actual charging power. Should be displayed in kW (power_w/1000)

transaction_wh	e.g.: 13500	Energy delivered to the Car. Should be displayed in kWh (transaction_wh/1000)
type2_state	a, b, c, d, e	See Table 2: Charging state for detail information
sig_current	e.g: 16	Current that is communicated to the car. It is the maximum charging current which is at the moment available from the wallbox.
free_charging	On, off	If free charging is enabled it is possible to charge a car without RFID authentication. free_charging shows if free charging is turned on or off.
current_a	e.g: 14.0,14.5,14.1	Actual output current of the wallbox delivered to the car on each phase
energy_man_current	e.g: 12	???
ambient_temp	e.g: +25.30	Estimated value of the ambient temperature of the charging controller (internal temperature of wallbox)
firmware_ver	e.g: 5.20.4-13148	Returns the value of the firmware Version.
cc_serial_n	e.g: 1703512493	Returns the Charging controller serial number
con_cycles_type2	e.g: 1450	Returns the number of cycles a car was connected to the wallbox.
max_current	e.g: 16	Returns the value of the maximum current the wallbox can deliver to a car.
rcmb_state	Triggered, okay	Shows if the internal DC-Failure current detection is okay or a failure was detected (it was triggered)
rcmb_max	e.g: 0.0, 0.0	Maximum residual current
rcmb_values	e.g: 0.0, 0.0	Actual residual current
Errors		Display errors as text string.

Table 1: Read Parameter REST-API

Vehicle State	Description
State A	Not connected
State B	EV connected. Ready to charge
State C	EV charging
State D	EV charging, ventilation required
State E	Error

Table 2: Charging state

### 3.1.2 Write and change Parameter:

The REST-API Interface can change and write Parameter values. The following chapters will describe the values of these parameters and how to get access to.

The structure of the URL is like:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data
"OperatorCurrentLimit_vehicleif=10&SUBMITTYPE=0d" --user operator:yellow_zone
http://192.168.178.55/operator/operator
```

The variables within the URL are:

#### Parameter name

The name of the Parameter. A list of the possible parameters are shown below.

#### Parameter value

Value of the parameter. The possible values of each parameter is shown in the list below.

#### Submitttype

SUBMITTYPE value="0d" Save parameter after write

SUBMITTYPE value="1d" Save parameter after write and restart charging controller

SUBMITTYPE value="2d" Set the value to default and restart the charging controller

#### Username

The username is operator and will not be changed

#### Password

The default password is "yellow\_zone". If a password change option is required, this parameter needs to be variable.

#### IP-Address

IP-Address of the wallbox. Need to be variable to be able to get access in any local area network. The IP-Address must be typed in within the App at the very first beginning opening the app. Therefore, an initial start screen must be opened to enter the IP-Address of the wallbox (see chapter 4.6.).

For authorization, please use the following Information:

Autorisierung: „Basic Auth“

Username: „operator“ see above like described

Password: „yellow\_zone“ see above like described

Content Type: „application/json“

parameter_name	Parameter value	Comment
OperatorCurrentLimit_vehicleif	Range: 6 to 16 Submitttype: 0d	Charging current the wallbox will deliver to the car. The parameter range for current is 6A to 16A. <a href="#">See 3.1.2.1 set charging current</a>
FreeCharging_vehicleif	Range: on/off Submitttype: 0d	Switch free charging on or off. <a href="#">See 3.1.2.2 free charging</a>
FLLFillMode_fll	Range: 1 = on, 0 = off Submitttype: 0d	Switch on or off the local whitelist learning mode. <a href="#">See 3.1.2.3 switch on RFID learning mode</a>

Table 3: Write Parameter REST-API

#### 3.1.2.1 set charging current

The maximum charging current which will be deliver to the car from the wallbox can be set during charging. The parameter to handle this is *OperatorCurrentLimit\_vehicleif* and the value of the charging current will be set in range from 6A to 16A. For Example the access to this parameter (where a value of 13A is submitted) is:



Set charging current to 13A:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data  
"OperatorCurrentLimit_vehicleif=13&SUBMITTYPE=0d" --user operator:yellow_zone  
http://192.168.178.55/operator/operator
```

*3.1.2.2 free charging*

The wallbox can be set to free charging if for example a car should be charged without RFID authentication. The parameter can set free charging on or off. The access to this parameter is:

Switch free charging on:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data  
"FreeCharging_vehicleif=on&SUBMITTYPE=0d" --user operator:yellow_zone  
http://192.168.178.55/operator/operator
```

Switch free charging off:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data  
"FreeCharging_vehicleif=off&SUBMITTYPE=0d" --user operator:yellow_zone  
http://192.168.178.55/operator/operator
```

*3.1.2.3 switch on RFID learning mode*

To add more RFID-chips to the RFID local whitelist a parameter can set the local whitelist to learning mode. In learning mode every RFID chip swiped over the RFID reader will be added to the local whitelist. If no RFID chip are swiped for 5 minutes the learning mode will be disabled automatically.

Switch on local whitelist learning mode:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data  
"FLLFillMode_fll=1&SUBMITTYPE=0d" --user operator:yellow_zone  
http://192.168.178.55/operator/operator
```

switch off local whitelist learning mode:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data  
"FLLFillMode_fll=0&SUBMITTYPE=0d" --user operator:yellow_zone  
http://192.168.178.55/operator/operator
```

*3.1.2.4 Change Password*

Not needed for version one

*3.1.2.5 Wi-fi configuration*

Not needed for version one

Wi-fi switch-on:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded"  
--data "Enabled_wlan=1&SUBMITTYPE=0d" --user operator:yellow_zone  
http://192.168.178.55/operator/operator
```

Wi-fi switch-off:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded"
--data "Enabled_wlan=0&SUBMITTYPE=1d" --user operator:yellow_zone
http://192.168.178.55/operator/operator
```

#### Wi-fi change SSID:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data
"WLANSSID_wlan=NameOfWLAN&SUBMITTYPE=0d" --user operator:yellow_zone
http://192.168.178.55/operator/operator
```

#### Wi-fi change password:

```
curl -X POST -H "content-type: application/x-www-form-urlencoded" --data
"WLANPassword_wlan=123&SUBMITTYPE=0d" --user operator:yellow_zone
http://192.168.178.55/operator/operator
```

### 3.1.2.6 LAN configuration

Not needed for revision 1

### 3.1.2.7 Show charging history

Not needed for revision 1

## 3.2. Parameter calculations

Further information for the customer can be generated by calculating interesting values based on internal parameter values of the charging controller. The parameters which need to be calculated are shown in the following chapters.

### 3.2.1 Costs of Charging session

This value should display the customer how much the actual cost of the charging session is. This value can be calculated based on the parameter *transaction\_wh*. The formula to calculate the costs is:

$$\text{cost\_of\_charging\_session} = \frac{\text{transaction\_wh}}{1000} * \text{average\_costs\_electricity}$$

The *transaction\_wh* can be read out via REST-API Interface like described in chapter 3.1.1.

The *average\_costs\_electricity* depends on the costs for electricity the customer must pay to the electricity supplier and the costs of a possible photovoltaic system which can be part of the electricity mix the car will be charged with. For the calculation of the *costs\_of\_charging\_session* and therefore the needed *average\_costs\_electricity*, a few variables need to be created. The parameters should have the following units and value ranges:

Name of Parameter	Type	Min value	Default value	Max value	Increment
cost_of_charging_session	Read only	0.00	-	655.35	0.01 [€]
share_of_photovoltaics_in_charging_electricity	Read/write	0.0	0.0	100.0	0.1 [%]
photovoltaics_costs	Read/write	0.00	0.0	655.35	0.01 [€/kWh]
electricity_costs	Read write	0.00	0.35	655.35	0.01 [€]
average_costs_electricity	Read only	Will be calculated			0.01 [€/kWh]

Table 4: parameter for Costs of charging session

The calculation of the *average\_electricity\_costs*, which is necessary for calculation of the costs for a charging session like described, is shown below:

$$\begin{aligned} \text{average\_costs\_electricity} &= \frac{\text{Photovoltaics\_costs} * \text{share\_of\_photovoltaics\_in\_charging\_electricity}}{100} \\ &+ \frac{\text{electricity\_costs} * (100 - \text{share\_of\_photovoltaics\_in\_charging\_electricity})}{100} \end{aligned}$$

For example:

*Photovoltaics\_costs* = 0.06 €/kWh

*electricity\_costs* = 0.35 €/kWh

*share\_of\_photovoltaics\_in\_electricity* = 30%

$$\text{average\_costs\_electricity} = \frac{0.06 \frac{\text{€}}{\text{kWh}} * 30\%}{100} + \frac{0.35 \frac{\text{€}}{\text{kWh}} * (100 - 30\%)}{100} = 0.26 \frac{\text{€}}{\text{kWh}}$$

The *average\_costs\_electricity* needs to be calculated once the app has been started or a parameter change was made to the *photovoltaics\_costs* or *share\_of\_photovoltaics\_in\_charging\_electricity*

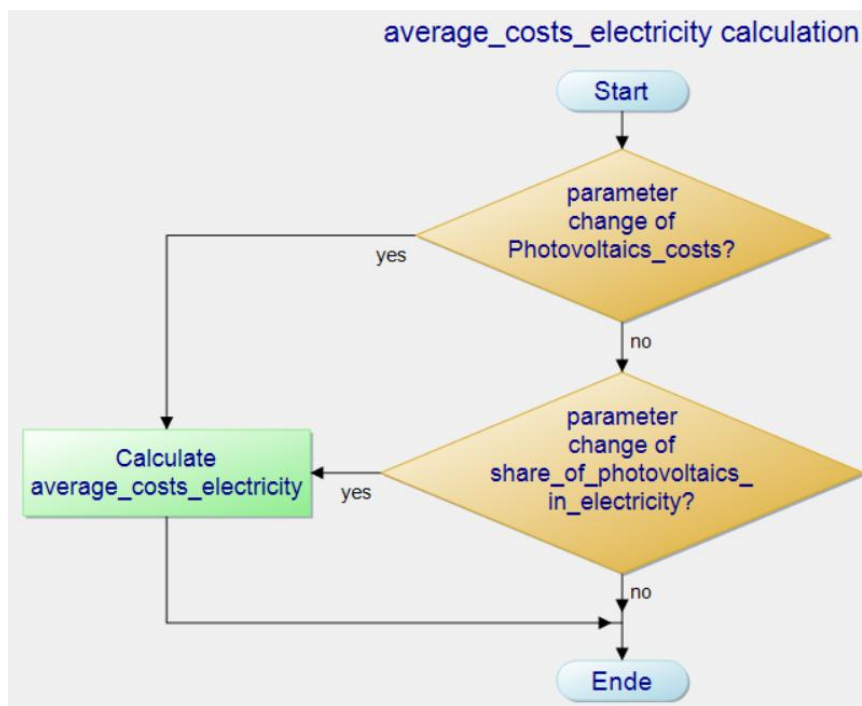


Figure 5: calculation average\_costs\_electricity

The *average\_costs\_electricity* parameter will be needed for other functions.

The calculation interval for the *cost\_of\_charging\_session* value will be coordinated with the developer of the App. Aim of the calculation interval should be a value which will be shown to the customer as smooth as possible with low resource usage of the mobile phone.

### 3.2.2 total cost of current delivered through wallbox

This value should display the whole costs of electrical current charged with the wallbox. This value can be calculated based on the parameter *meter\_wh*. The formula to calculate the total costs is:

$$total\_costs\_of\_charged\_current = \frac{meter\_wh}{1000} * average\_costs\_electricity$$

The *meter\_wh* can be read out via REST-API Interface like described. The parameters should have the following units and value ranges:

Name of Parameter	Type	Min value	Default value	Max value	Increment
Total_cost_of_charged_current.	Read only	0.0	-	1000000.0	0.1€
average_costs_electricity	See 3.2.1				

Table 5: Parameter for total cost of current delivered through wallbox

The calculation interval for the *total\_costs\_of\_charged\_current* value will be coordinated with the developer of the App. Aim of the calculation interval should be a value which will be shown to the customer as smooth as possible with low resource usage of the mobile phone

### 3.2.3 average power delivered to the car

Not needed for version one

This parameter calculates the average power delivered to the car over all past charging cycles. This value can be calculated based on the parameter *meter\_wh* and *con\_cycles\_type2*. This value should be calculated when the app is opened and only if the *conn\_state* has the status *vehicle\_charging\_type2*. The formula to calculate the average power delivered to the car is:

$$average\_power = \frac{meter\_wh}{1000 * cycles\_type2}$$

The *meter\_wh* and *cycles\_type2* can be read out via REST-API Interface like described in chapter 3.1.1. The parameters should have the following units and value ranges:

Name of Parameter	Type	Min value	Default value	Max value	Increment
average_power	Read only	0.0	-	6553.5	0.1 kWh

Table 6: Parameter for average power delivered to the car

The following program flow chart illustrates the calculation of the average power delivery. Care must be taken not to divide by zero (divide when *con\_cycles\_type2* = 0 must be avoided).

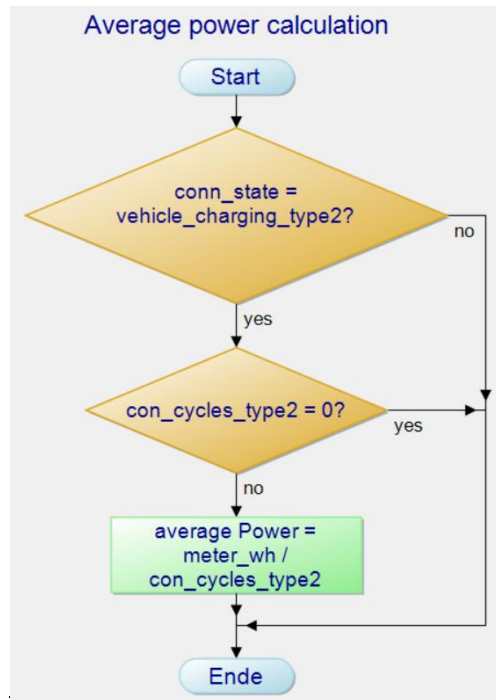


Figure 6: flow charg average power calculation

The calculation interval for the *average\_power* value will be coordinated with the developer of the App. Aim of the calculation interval should be a value which will be shown to the customer as smooth as possible with low resource usage of the mobile phone

### 3.2.4 cost savings compared to a combustion engine for actual charging session

This calculated value should display the cost savings between a normal combustion engine fueled with gasoline and the electric car charged with electrical energy. This value can be calculated based on the parameter *transaction\_wh* and a few parameters that need to be created and filled with variables. The following variables need to be created:

Name of Parameter	Type	Min value	Default value	Max value	Increment
gasoline_costs	Read/write	0.00	1.50	655.35	0.01 [€/l]
power_consumption	Read/write	0.0	15.0	6553.5	0.1 [kWh/100km]
gasoline_consumption	Read/write	0.0	6.5	6553.5	0.1 [l/100km]
costs_savings	Read only	-655.35	-	655.35	0.01 [€/kWh]
power_charged_distance	Read only	Will be calculated			0.01 [km]
amount_of_gasoline	Read only	Will be calculated			0.01 [l] (liter)
average_costs_electricity	See 3.2.1				

Table 7: Parameter for cost savings actual charging session

The calculation behind the cost savings is shown below:

1. Calculation of actual *power\_charged\_distance*:

This value calculates the distance the car can drive with the charged electricity.

$$power\_charged\_distance = \frac{transaction\_wh/1000}{power\_consumption} * 100 = \frac{transaction\_wh}{power\_consumption} * 10$$

2. Amount of gasoline:

This value calculates the amount of gasoline that would be necessary to drive the distance, calculated in point 1, with gasoline.

$$amount\_of\_gasoline = \frac{power\_charged\_distance * gasoline\_consumption}{100}$$

3. Equivalent Costs of gasoline:

This value calculates the costs of the amount of gasoline, calculated in point 2. It will be the equivalent costs of gasoline needed for the distance charged within point 1 of this calculation.

$$equivalent\_costs\_of\_gasoline = amount\_of\_gasoline * gasoline\_costs$$

4. Costs for electricity (always calculated in 3.2.1):

This value calculates the costs for the electricity charged to the vehicle.

$$Cost\_of\_charging\_session = \frac{transaction\_wh}{1000} * average\_costs\_electricity$$

5. Cost savings:

This value is the result of the calculation and will show the cost savings between the charged electricity and the equivalent costs for gasoline.

$$cost\_savings = Equivalent\_costs\_of\_gasoline - cost\_of\_charging\_session$$

This calculation could be done when transaction\_wh/1000 increases by one like the flow chart below shows. Regarding of the resource usage of the mobile phone the calculation interval could be faster. This need to be clarified with the developer of the App.

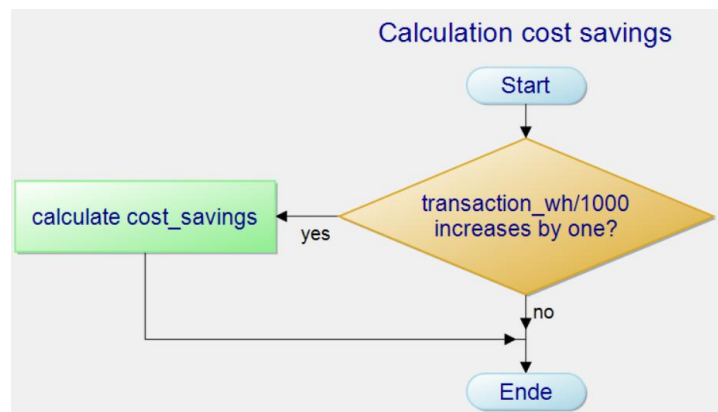


Figure 7: calculation cost savings

### 3.2.5 total cost savings compared to a combustion engine

Not needed for version one.

### 3.2.6 calculation of savings of CO2

This calculated value should display the CO2 savings between a normal combustion engine fueled with gasoline and the electric car charged with electrical energy. This value can be calculated based

on the parameter *transaction\_wh* and a few parameters that need to be created and filled with variables. The following variables need to be created:

Name of Parameter	Type	Min value	Default value	Max value	Increment
gasoline_CO2	Read/write	0.00	2370	65535	1 [g CO2/l]
diesel_CO2	Read/write	0.0	2650	65535	1 [g CO2/l]
electricity_CO2	Read/write	0.0	366	65535	1 [g CO2/kWh]
photovoltaics_CO2	Read/write	0.0	50	65535	1 [g CO2/kWh]
CO2_savings	Read only	Will be calculated			0.1 [Kg CO2/kWh]
choose_gasoline_diesel	Read/write	0	0	1	1
average_CO2_electricity	Read only	Will be calculated			1 [g CO2/kWh]
power_charged_distance	See 3.2.4				
amount_of_gasoline	See 3.2.4				
power_consumption	See 3.2.4				
share_of_photovoltaics_in_charging_electricity	See 3.2.1				
equivalent_CO2_of_gasoline	Read only	Will be calculated			0.1 [Kg CO2/kWh]

Table 8: Parameter for savings CO2

The calculation behind the CO2 savings is shown below:

1. Calculation of average CO2 in electricity

The *average\_CO2\_electricity* depended on the CO2 for electricity which can be adopted for the electricity from the grid and the CO2 emission caused in the production of a possible photovoltaic system and could be part of the power mix the car will be charged with.

*average\_CO2\_electricity*

$$= \frac{\text{photovoltaics\_CO2} * \text{share\_of\_photovoltaics\_in\_charging\_electricity}}{100} + \frac{\text{electricity\_CO2} * (100 - \text{share\_of\_photovoltaics\_in\_charging\_electricity})}{100}$$

For example:

*photovoltaics\_CO2* = 50 gCO2/kWh

*electricity\_CO2* = 366 gCO2/kWh

*share\_of\_photovoltaics\_in\_electricity* = 30%

$$\text{average\_CO2\_electricity} = \frac{50g * 30\%}{100} + \frac{366g * (100 - 30\%)}{100} = 271 \frac{g \text{ CO2}}{kWh}$$

The *average\_CO2\_electricity* needs to be calculated once the app has been started or a parameter change was made to the *photovoltaics\_CO2* or *share\_of\_photovoltaics\_in\_charging\_electricity*.

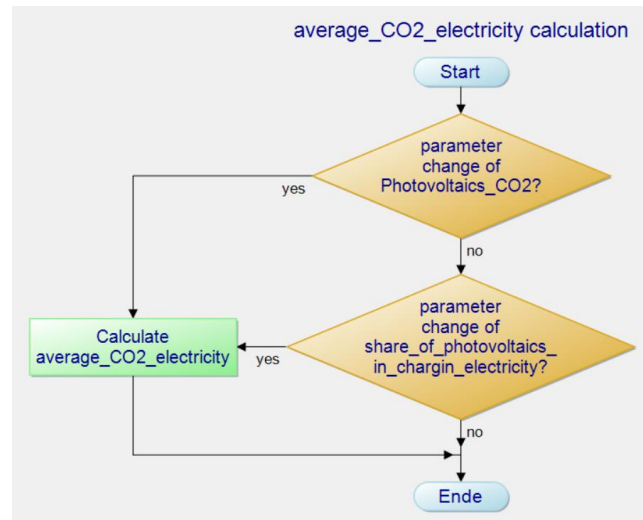


Figure 8: Calculation average\_CO2\_electricity calculation

2. Equivalent gCO2 of gasoline:

This value calculates the CO2 emission of the *amount\_of\_gasoline*, calculated in chapter 3.2.4. It will be the equivalent CO2 emission of gasoline needed for the distance charged with electrical power calculated in 3.2.4. Because gasoline and diesel have different emissions at first it must be chosen which kind of fuel to compare with.

If the variable *choose\_gasoline\_diesel* = 0 than gasoline is selected and the calculation is:

$$equivalent\_CO2\_of\_gasoline = \frac{amount\_of\_gasoline * gasoline\_CO2}{1000}$$

If the variable *choose\_gasoline\_diesel* = 1 than diesel is selected and the calculation is:

$$equivalent\_CO2\_of\_gasoline = \frac{amount\_of\_gasoline * diesel\_CO2}{1000}$$

3. CO2 for electricity:

This value calculates the costs for the electricity charged to the vehicle.

$$CO2\_for\_electricity = \frac{transaction\_wh}{1000} * \frac{average\_CO2\_electricity}{1000}$$

4. CO2 savings:

This value is the result of the calculation and will show the CO2 savings between the charged electricity and the equivalent CO2 for gasoline.

$$CO2\_savings = Equivalent\_CO2\_of\_gasoline - CO2\_for\_electricity$$

This calculation needs to be done when *transaction\_wh/1000* increases by one like the flow chart below shows. Regarding of the resource usage of the mobile phone the calculation interval could be faster. This need to be clarified with the developer of the App



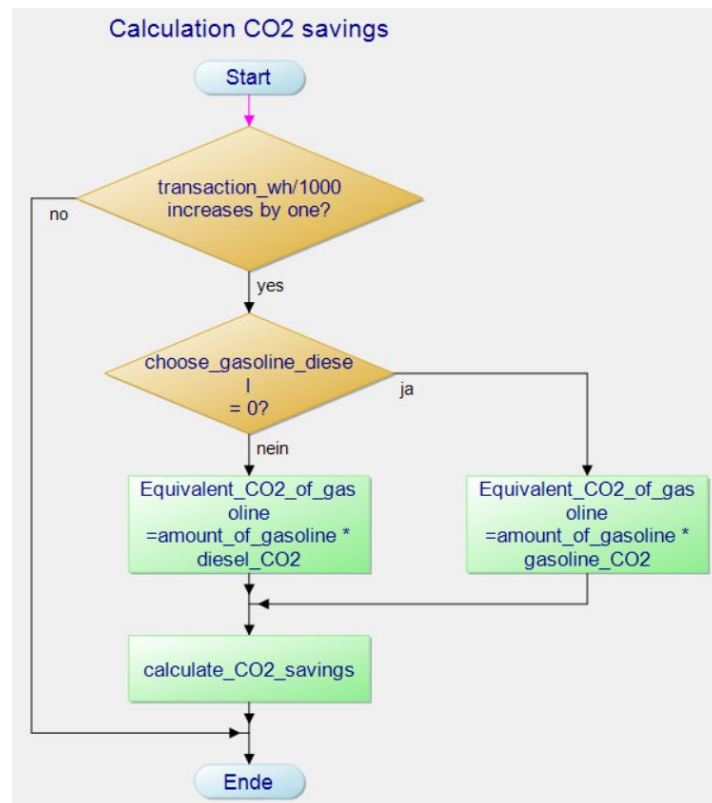


Figure 9: Calculation CO2 savings

### 3.2.7 total savings CO2

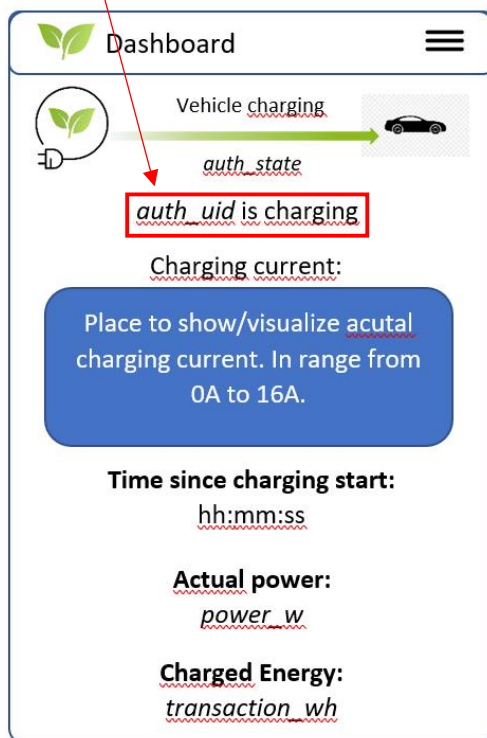
Not needed for version one

### 3.2.8 Assign a Name to a RFID-UID

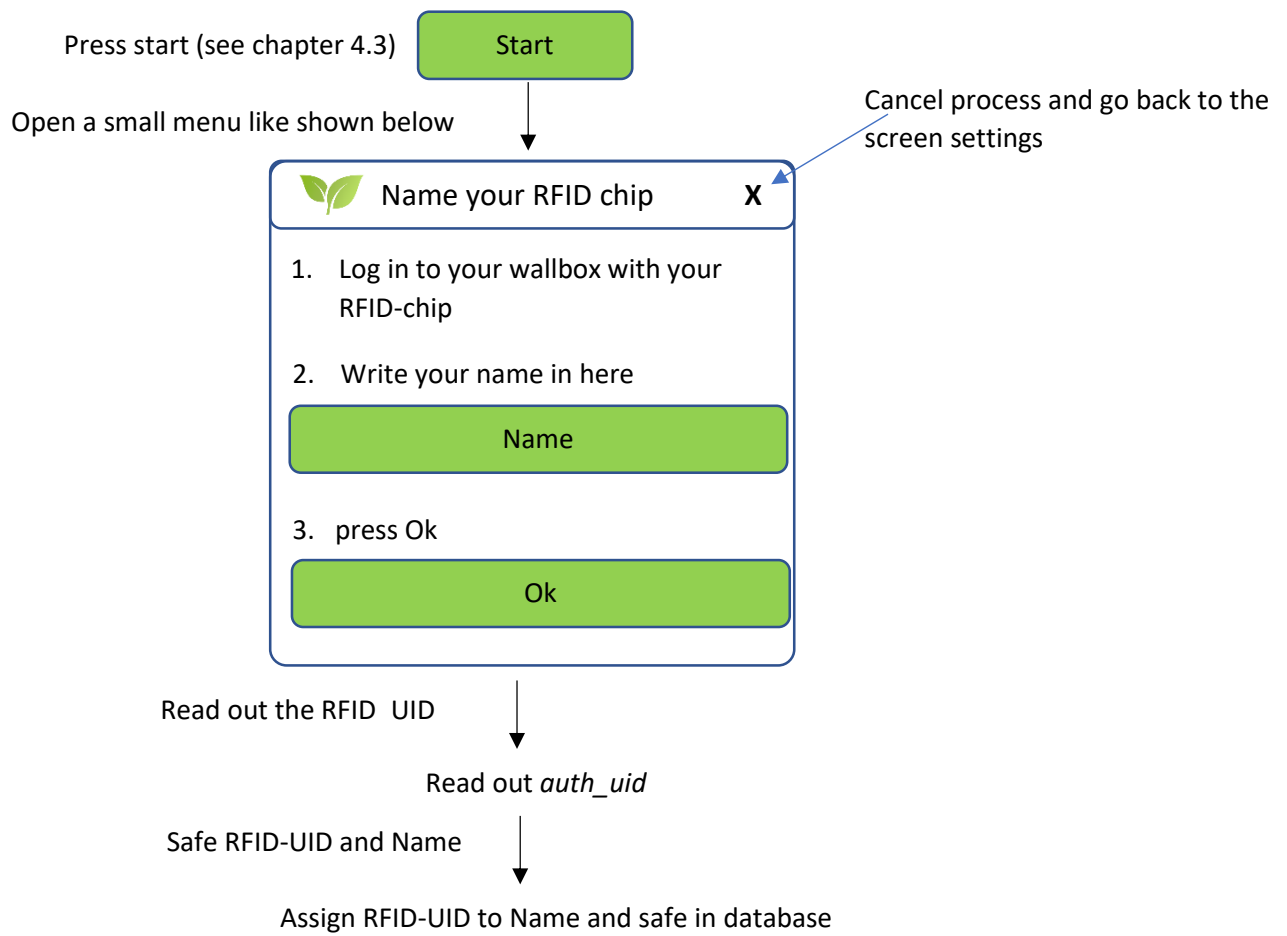
If the authorization to the wallbox is done with the RFID-Chip the currently authorized RFID-UID can be read out via REST-API using the *auth\_uid* parameter. Because a long hexadecimal value is not comfortable to read and to see directly in the app who is currently authorized and charging a car it should be implemented a function to assign a Name to a RFID-UID.

To choose this option a start button to start the procedure could be on the settings screen (see 4.3. Settings). To assign a name to a RFID-UID the following steps need to be done:

1. Enter a name which should be assigned to an RFID-Chip
2. Read in RFID-Chip
  - a. Display a text in the app “pleas swipe now your RFID-Chip in front of your Wallbox and press ok to confirm”
  - b. Read out the actual RFID-UID by read out the parameter *auth\_uid*.
3. Safe the Name and the RFID-UID in the app (kind of a small database)
4. This Name should now be displayed in the app (see 4.1. Main-Page/Dashboard/startscreen)



After a Name was assigned to an RFID-Chip the Name instead of the *auth\_uid* should be displayed on the startscreen if the RFID-Chip was swiped in front of the wallbox. Because one wallbox could have more than one RFID-Chip to authorize with, a kind of a small database should be existing where the names assigned to the RFID-Chips. Each RFID-Chip/RFID-UID must only be assigned to one name. The following steps will show the process in detail. The design shown below is only an example to show the technical background, the design should be state of the art, smart and modern.



## 4. Design

the following chapter will show roughly an overview of the design. This will only show which parameters needs to be displayed and how the screen of the app could be structured. A modern, smart, and nice design should be aimed.

### 4.1. Main-Page/Dashboard/startscreen

Within the start screen a kind of a dashboard should be realized where the following parameters should be displayed:

Connection state (*conn\_state* from REST-API)

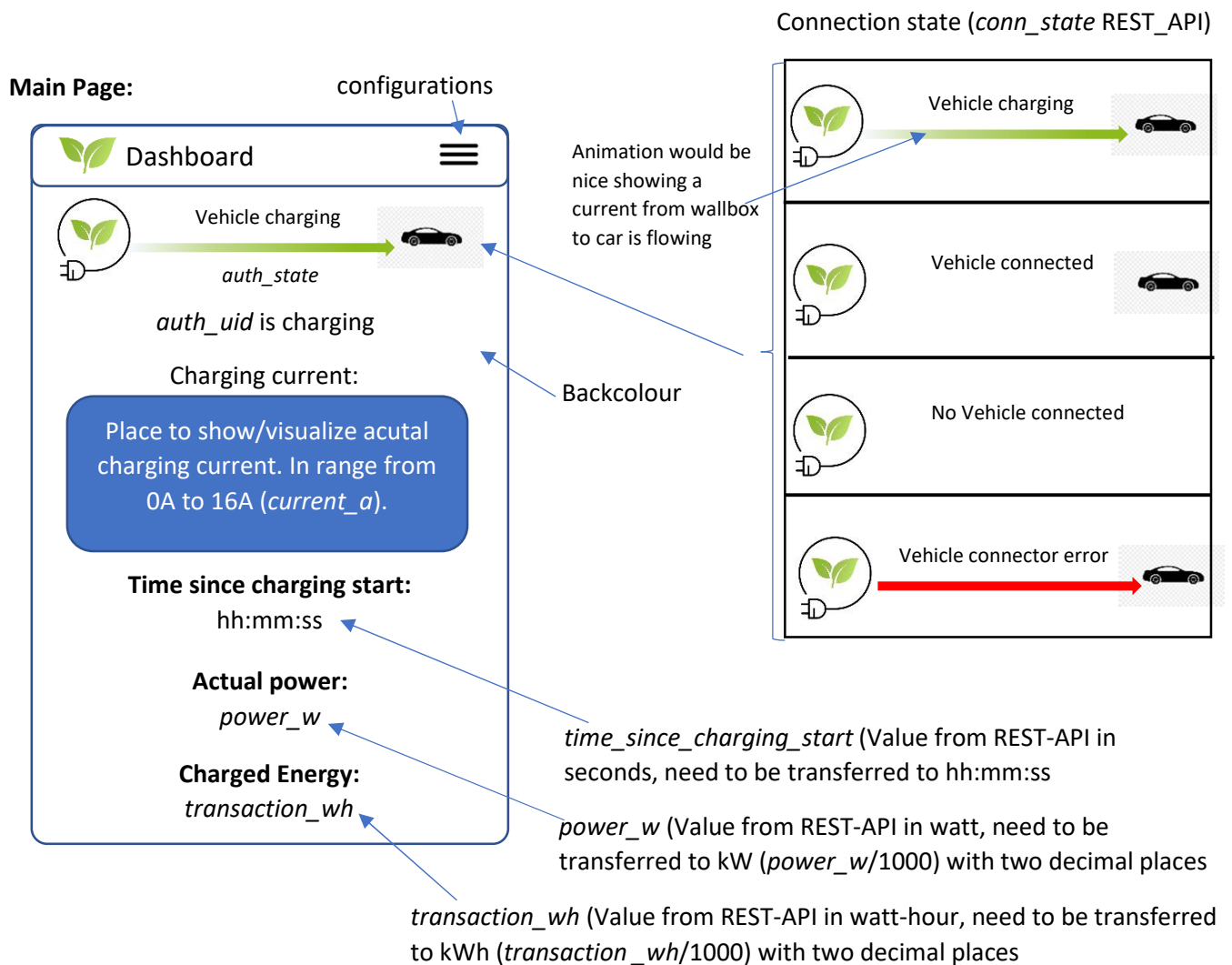
Authentication state (*auth\_state* from REST-API)

Actual charging current (*current\_a* from REST-API)

Time since charging start (*time\_since\_charging\_start* from REST-API)

Actual charging power (*power\_w* from REST-API)

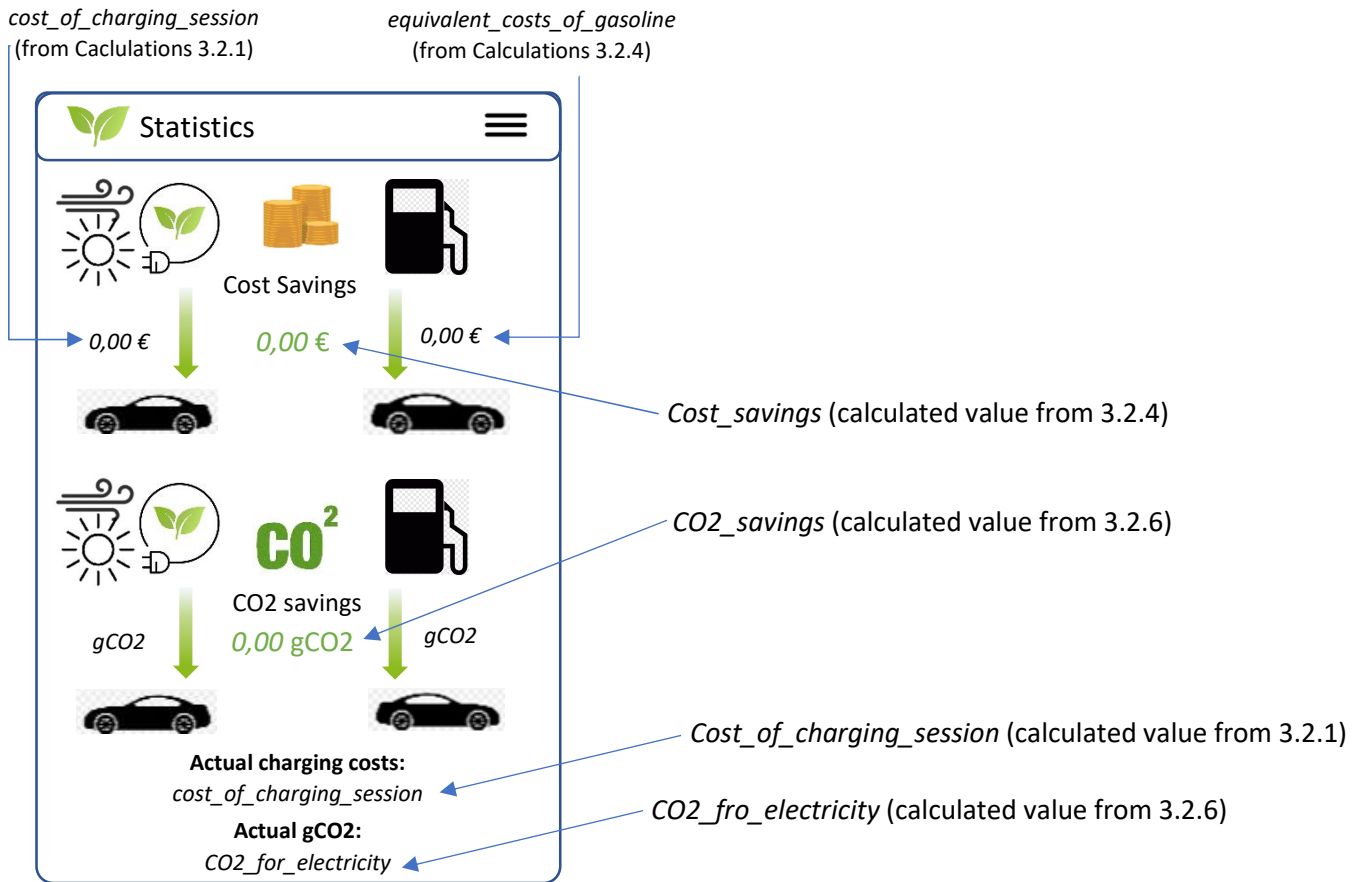
Charged Energy (*transaction\_wh* from REST-API)



The parameter *current\_a* from REST-API will deliver 3 currents because three phases from grid will charge the car. A smat visualization should visualize the 3 currents (in a range from 0 Ampere to 16 Ampere) and the average of these 3 currents should be displayed as a value.

## 4.2. Calculated Values

This screen should display all the calculated values from chapter 3.2.



### 4.3. Settings

This screen can be opened by touching the  symbol. The screen must be scrollable because many parameters must be set.



The screenshot shows the 'Settings' screen of the Wallbox app. The screen has a green header with a leaf icon and the title 'Settings'. A hamburger menu icon is in the top right corner. The settings are listed in a scrollable list with green input fields. Annotations with arrows point to specific fields:

- Charging current:** Xx [A]. Annotation: "This current must not be greater than the maximum value that can be queried in parameter *max\_current* (see 3.1.1 Table 1). If a value greater than *max\_current* will be typed in here, a warning should be displayed: "the maximum current is *max\_current*" and the value should than be limited to *max\_current*."
- Free Charging:** On/Off. Annotation: "Including a safety query when switching the button to *on*: "do you really want to activate the free charging mode? If you choose yes, no authentication (RFID) will be required to start the charging session" This should be confirmed with (yes/no)."
- Whitelist learning mode:** On/Off
- Share of Photovoltaic:** 0.0%
- Photovoltaic costs:** 0.00 [€/kWh]
- electricity costs:** 0.35 [€/kWh]
- Average electricity costs:** x.xx [€/kWh]. Annotation: "Value calculated from *electricity\_costs* and *photovoltaic\_costs* and *share\_of\_photovoltaics\_in\_charging\_electricity*. This value cannot be modified. It is just the information from the calculation (see 3.2.1)"
- gasoline costs:** 1.50 [€/l]
- Power consumption:** 15.0 [kWh/100km]
- Gasoline consumption:** 6.5 [l/100km]
- CO2 Gasoline:** 2370 [gCO2/l]
- CO2 diesel:** 2650 [gCO2/l]
- CO2 electricity:** 366 [gCO2/kWh]
- CO2 Photovoltaics:** 50 [gCO2/kWh]
- Average CO2 electricity:** x.xx [gCO2/kWh]. Annotation: "Value calculated from *electricity\_CO2* and *photovoltaic\_CO2* and *share\_of\_photovoltaics\_in\_charging\_electricity*. This value cannot be modified. It is just the information from the calculation (see 3.2.6)"
- Chose Combustion:** Gasoline/diesel. Annotation: "This is to select whether the calculation (calculation of savings of CO2 chapter 3.2.6) should be made with Gasoline or diesel."
- Assign Name to RFID:** Start. Annotation: "Opens a small menue to handle the process assigne a name to a RFID-UID (see 3.2.8)"
- IP-Address Wallbox:** 192.168.178.111. Annotation: "Type in IP-Address for REST-API interface (see chapter 3.1)"
- Language:** Deutsch/English

The value of Charging current must be communicated to the wallbox via REST-API within the parameter *OperatorCurrentLimit\_vehicleif* (see table 3) (default value is 16A).

The free charging mode must be communicated to the wallbox via REST-API within the parameter *FreeCharging\_vehicleif* (see table 3) (default value is off).

The whitelist learning mode for new RFID-UIDs must be communicated to the wallbox via REST-API within the parameter *FLLFillMode\_fll* (see table 3) (default value is off).

Share of Photovoltaic is the value that must be written to the variable *share\_of\_photovoltaics\_in\_charging\_electricity* (see 3.2.1 table 4) (default value is 0.00).

Photovoltaic costs is the value that must be written to the variable *photovoltaic\_costs* (see 3.2.1 table 4) (default value is 0.00).

electricity costs is the value that must be written to the variable *electricity\_costs* (see 3.2.1 table 4) (default value is 0.35).

gasoline costs is the value that must be written to the variable *gasoline\_costs* (see 3.2.4 table 7) (default value is 1.50)

power consumption is the value that must be written to the variable *power\_consumption* (see 3.2.4 table 7) (default value is 15.0).

gasoline consumption is the value that must be written to the variable *gasoline\_consumption* (see 3.2.4. table 7) (default value is 6.5)


CO2 gasoline is the value that must be written to the variable *gasoline\_CO2* (see 3.2.6 table 8) (default value is 2370).

CO2 diesel is the value that must be written to the variable *diesel\_CO2* (see 3.2.6. table 8) (default value is 2650)

CO2 electricity is the value that must be written to the variable *electricity\_CO2* (see 3.2.6 table 8) (default value is 366)

Choose combustion is the value that will choose which calculation will be done within the chapter 3.2.6 calculation step 2 (Equivalent gCO2 of gasoline) (default value should be 0).

#### 4.4. System information


System

**Firmware version:**  
*Firmware\_ver*

**RCMB State:**  
*rcmb\_state*

**Errors:**  
*errors*

**Estimated ambient temp:**  
*ambient\_temp*

**Charging state:**  
*type2\_state*

**Maximum charging current Wallbox:**  
*max\_current*

**Actual RFID-UID authorized:**  
*auth\_uid*

**Actual residual current:**  
*rcmb\_values*

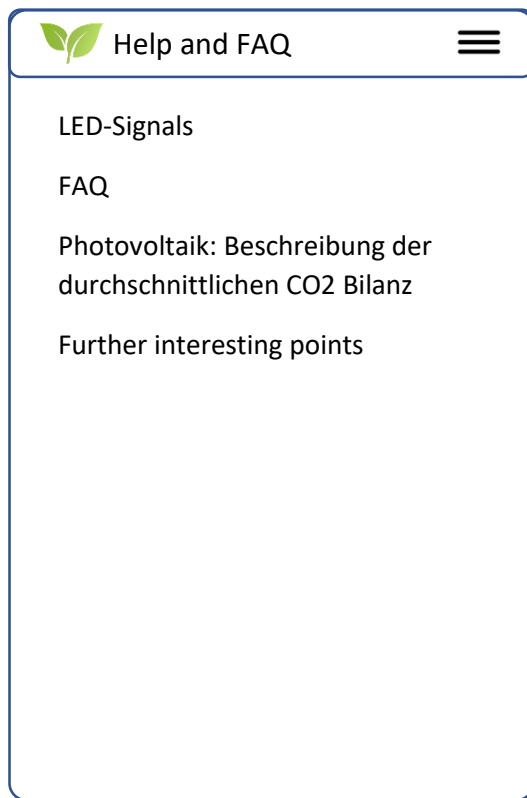
**Maximum residual current:**  
*rcmb\_max*

*type2\_state* will deliver back a letter A, B, C, D or E as a result. This need to be transferred to clear text. Below the letters are assigned to the clear text:

- A Not connected
- B EV connected. Ready to charge
- C EV charging
- D EV charging, ventilation required
- E Error

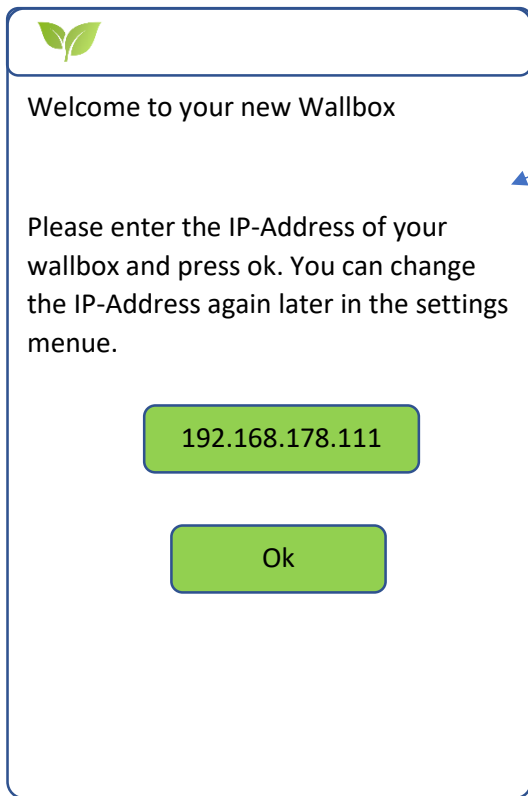


#### 4.5. Help and FAQ



#### 4.6. Initial start Screen

At the very first time the App will be opened the IP-Address of the Wallbox must be entered. This is necessary to have the actual IP-Address of the wallbox for the REST-API interface. The IP-Address must be saved within the app to not re-enter the address every time



The image shows a wireframe of the initial start screen of the Wallbox app. It features a green leaf icon in the top left corner. Below the icon, the text "Welcome to your new Wallbox" is displayed. Further down, a paragraph of text reads: "Please enter the IP-Address of your wallbox and press ok. You can change the IP-Address again later in the settings menu." Below this text, there is a green rounded rectangular button containing the IP address "192.168.178.111". At the bottom, there is another green rounded rectangular button labeled "Ok".

Kind of animation? Kind of a welcome screen for the very first time?

## 4.7 Things to clarify

~~Possible to choose two languages German and English~~

Help and FAQ: Embed texts later? Ongoing process possible?

Max current dynamical adaptation? (In Design 4.1) (*max\_current*)

Grafik Design Symbol

Maybe a variable password for REST-API interface

~~General Modbus TCP interface possible?~~

## Revision Changes

Revision	Changes	Date
Rev 00.01	Basic document	30.03.2022
Rev 00.02	<ul style="list-style-type: none"> <li>Add missing parameter photovoltaics_CO2 in Table 8: Parameter for savings CO2</li> <li>Add missing parameter photovoltaics_CO2 in chapter 4.3 settings.</li> </ul>	20.04.2022 20.04.2022
Rev 00.03	<ul style="list-style-type: none"> <li>Add some more Information about REST API (Content Type and authorization)</li> </ul>	04.05.2022