Smart Home API

January 2023

Contents

1 Introduction

Ideally, the smart mirror should be connected to the Edison servers in order to monitor all the consumption of the home. This would allow for real-time monitoring of energy usage and provide insights into how to reduce consumption and save on energy costs. By connecting to the Edison servers, the smart mirror would have access to data on electricity information on renewable energy sources such as the user's solar panel.

For simplicity in the presentation stage, an artificial API has been used. This API has been created with real values for home consumption, allowing for a demonstration of the smart mirror's capabilities without the need for a physical connection to the Edison servers. This approach would still provide a good representation of how the smart mirror would function in a real-world setting and would allow for testing and refining of the user interface.

2 Simulated House

This section focuses on the structure of the simulated house. The house is equipped with photovoltaic panels that can convert sunlight into electrical and a battery capable to store the energy. Photovoltaic panels convert energy in a weather-dependent manner according to the following formula:

$$energy = number_panels \cdot energy_panel \cdot k \tag{1}$$

The variable k depends on the weather condition ($k = 1$ for sunny day, $k = 0.3$ for cloudy day, $k = 0.1$ for rainy day) while $energy_panel$ is the maximum energy a panel can collect in an hour. The simulated house has 18 panels (with a maximum production capacity of $0.35kWh$ each) and a battery with a capacity of $10kW$.

In order to make the API as realistic as possible, the following appliances consumption were chosen:

Appliance	**Consumption**
Air conditioner	3.00kWh
Dehumidifier	0.07kWh
Cooker	1.04kWh
Dishwasher	1.20kWh
Dryer	3.50kWh
Boiler	24.00kWh
Oven	2.30kWh
Washing machine	1.30kWh

Due to the modular code, it's easy to modify the home structure or to add new appliances just by modifying the `home.js` file inside the model folder.

3 Weather

The weather can be:

- Sunny
- Cloudy
- Rainy

The weather is initialized randomly and it changes during simulation using a fuzzy logic function.

4 Simulate

One of the main functionalities of the API is to simulate one hour of consumption switching on/off the house appliances. During simulation, the API collects the energy from photovoltaic panels and tries first to use those energy for the appliances that are switched on and secondly, if the energy is still available, it starts charging the battery up to the maximum battery capacity. It is considered *clean energy* the one coming from batteries or photovoltaic panels, while we define *dirty energy* the energy from any other source.

The API not only collects the amount of *clean* and *dirty energy* during the simulation but also considers the full history of the mirror.

The API outputs a total grade and a current simulation grade which are indicators of the user's clean consumption. The grades are computed according to the following formulas:

$$CurrentGrade = \frac{clean_{energy}}{clean_{energy} + dirty_{energy}} \tag{2}$$

$$TotalGrade = \frac{1}{N}\sum_{i=0}^{N} CurrentGrade_i \tag{3}$$

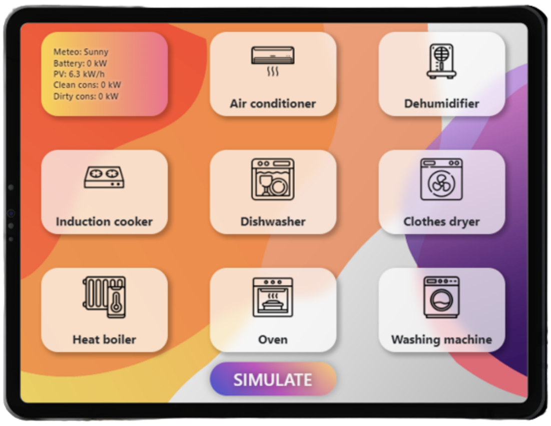
5 Smart Forest interaction

In order to simplify the interaction with the API and to display some relevant data (weather, consumption...), a tablet is added to the mirror to display some relevant data such as weather and consumption.

It is also possible to switch household appliances on and off by clicking on the relative button on the tablet.

When the user wants information regarding a specific household appliance, he/she asks Flora which automatically makes the right call to the API which provides data.

Total grade and current grade indicators are used to update trees experience and to increment the number of leaves each time the simulation button is clicked.



Meteo: Sunny
Battery: 0 kW
PV: 6.3 kW/h
Clean cons: 0 kW
Dirty cons: 0 kW

Figure 1: Tablet

6 API structure

The API has been developed with JavaScript. It is divided into 2 main parts:

- Model
- Router

The model folder contains different classes where the functionalities are implemented and data are stored. At each class is associated a Router class which manage the interaction of the relative endpoints.

6.1 API endpoints

HOME ROUTER

- The API returns all the information about the house.
- `https://smart-home-api-2j4i.onrender.com/home`
- The API simulates one hour of consumption.
- `https://smart-home-api-2j4i.onrender.com/home/simulate`
- The API resets all the data.
- `https://smart-home-api-2j4i.onrender.com/home/reset`

APPLIANCES ROUTER

- The API returns all the information about the appliances in the house.
- `https://smart-home-api-2j4i.onrender.com/appliances`
- The API returns all the information about the appliance with the name passed as parameter.
- `https://smart-home-api-2j4i.onrender.com/appliances/name=:name`

- The API turns on the appliance with the name passed as parameter.
- `https://smart-home-api-2j4i.onrender.com/appliances/turnOn/name=:name`
- The API turns off the appliance with the name passed as parameter.
- `https://smart-home-api-2j4i.onrender.com/appliances/turnOff/name=:name`
- The API returns the appliance which is consuming the most at the moment of the call.
- `https://smart-home-api-2j4i.onrender.com/appliances/mostConsuming`

BATTERY ROUTER

- The API returns the state of the batteries.
- `https://smart-home-api-2j4i.onrender.com/batteries`
- The API sets to zero the capacity of the batteries.
- `https://smart-home-api-2j4i.onrender.com/batteries/reset`

PHOTOVOLTAIC PANELS ROUTER

- The API returns the energy collected by the photovoltaic panels (in kWh).
- `https://smart-home-api-2j4i.onrender.com/panels/output`

WEATHER ROUTER

- The API returns the current weather.
- `https://smart-home-api-2j4i.onrender.com/meteo`
- The API simulates the weather using basic fuzzy logic.
- `https://smart-home-api-2j4i.onrender.com/meteo/change`
- The API changes the weather.
- `https://smart-home-api-2j4i.onrender.com/meteo/sunny`
- `https://smart-home-api-2j4i.onrender.com/meteo/cloudy`
- `https://smart-home-api-2j4i.onrender.com/meteo/rainy`