The Illuminator

Dashboard

Ali Ahmad Sohrab Ashraf
Victor van Doorn

Supervised by Milos Cvetkovic and Arjen van der Meer



The Illuminator

Dashboard

Ali Ahmad Sohrab Ashraf
Victor van Doorn

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Wednesday July 1, 2020 at 10:00 AM.

Student numbers: 4391306, 4477693
Project duration: April 20, 2020 – Juni 19, 2020
Thesis committee:

Dr. M. Cvetkovic,	TU Delft, daily supervisor
Prof. dr. ir. A. van der Meer,	TU Delft, supervisor
Dr. A. Smets,	TU Delft, Assessor

TU Delft

Abstract

This report discusses the design and implementation of a subsystem required for the Illuminator, which is an energy system integration demonstrator. The purpose of the subsystem discussed in this report is visualising incoming simulated power flows, visualising the output data of various emission and finance data and also change various input parameters. This system has been realised using Dash, which is the python framework used for building the dashboard web application. A study case is provided to better understand the use of the demonstrator.

keywords: dashboard, electricity emissions, energy system visualisation, energy finance, Raspberry Pi

Preface

Before you lies the thesis "The Illuminator Dashboard". This thesis is part of the Bachelor Graduation Project of Electrical Engineering at Delft University of Technology. It details the design process of the Dashboard, one of three parts of the Energy System Integration Demonstrator project. The dashboard is the visualisation part of the demonstrator, which can either show the input results of the hardware hub or be used as an input for the hardware hub. Together with the energy data widget, where the power calculations take place, the Illuminator teaches the user about the integration of energy modules in a neighbourhood.
We would like to thank our supervisors Milos Cvetkovic and Arjen van der Meer, for their excellent guidance and support during this process. We would also like to thank Arno Smets for his helpful input to send us in the right direction. We also would like to thank our other project members: Ruben Siemensma, Jean Paul Rozestraten, Benjamin Visser and Daniel van Paassen, for the wonderful cooperation that resulted in a enjoyable and positive working environment.

Ali Ahmad Sohrab Ashraf
Victor van Doorn
Delft, June 2020

Contents

1 Introduction 5
1.1 Problem Description . . . 5
1.2 general overview . . . 6
1.3 Thesis Structure . . . 7

2 Program of Requirements 8
2.1 Requirements for the system . . . 8
2.2 Requirements for the Dashboard . . . 9

3 Photo-Voltaic Generation 10
3.1 Model Input Parameters . . . 10
3.1.1 PV panel type . . . 11
3.1.2 Number of PV panels . . . 11
3.1.3 PV panel tilt . . . 11
3.2 Model Output . . . 11

4 Wind Energy Generation 13
4.1 Input . . . 13
4.1.1 Wind turbine model . . . 13
4.1.2 Number of wind turbines . . . 14
4.2 Output . . . 14

5 Neighbourhood Power Visualisation 15
5.1 Neighbourhood Load . . . 15
5.1.1 Load input . . . 15
5.1.2 Load output . . . 15
5.2 Storage . . . 15
5.2.1 Storage input . . . 16
5.2.2 Storage Output . . . 16
5.3 Grid . . . 17
5.3.1 Grid power flow . . . 17
5.3.2 Grid electricity mix . . . 17
5.4 Stand-alone power system . . . 18

6 Life-Cycle Emission Analysis 19
6.1 PV Systems . . . 20
6.2 Wind Turbines . . . 21
6.3 EV and Hydrogen Batteries . . . 22
6.3.1 EV Battery . . . 22
6.3.2 Hydrogen Battery . . . 22
6.4 Fossil Sources . . . 23

7 Financial Analysis 25
7.1 Levelized Cost of Energy . . . 26
7.2 Payback Time . . . 27
7.3 Financial Report 2019 . . . 27

8 Implementation 29
8.1 Communication and MQTT . . . 29
8.2 Dash . . . 29

9 Case Study Results 30
9.1 The Case Study . . . 30
9.2 Power Flow Results . . . 31
9.3 Emissions Results . . . 31
9.4 Financial Results . . . 31

10 Conclusions and Future Work 32

A Cost Breakdown 34

B Case study 36
B.1 Initial input results . . . 36

C Python Code 41
C.1 Dashboard input code . . . 41
C.2 Dashboard Illuminator output code . . . 42
C.3 Dashboard emission code . . . 45
C.4 Dashboard financial code . . . 47
C.5 Dasboard MQTT python code . . . 52

Bibliography 54

1 Introduction

1.1. Problem Description

The global energy transition is considered to be one of the most urgent tasks facing humanity today. Unfortunately, understanding the integration of certain energy systems is a complex task due to the rise of new technologies who's use is unfamiliar to most. Renewable generation is becoming an increasingly dominant player in the global energy market. A significant amount of these renewable energy sources have an intermittent generation property, where the energy output is considered non-dispatchable. This means that the energy generation output relies heavily on external factors such as weather and are unable to be operated fully at any given time to meet societies fluctuating electricity demands. Most people have trouble understanding the consequences this revolution in energy generation, due to the inability of bringing the operation and integration of these large systems in perspective. Public understanding is a crucial factor in reaching the global goals in terms of the energy transition and this is the reason for develop ping the Illuminator. The Illuminator visualises the performance and integration of all essential energy system modules to give the user a better understanding in the energy transition process.

The peripherals, depicted in 1.1, can be easily disconnected or connected to the system. The Illuminator has a dashboard where the user can choose a variety of inputs regarding the system size and configuration. The dashboard will then visualise the outputs in a way that accommodates both regular people and academics. The Illuminator can also be used in a more hands on way, by using hardware inputs and experiencing certain corresponding physical outputs. The Illuminator strives to demonstrate the performance of a specific energy system (e.g. neighbourhood, street, house) in terms of finances and greenhouse gas emissions. The Illuminator is a great educational tool to help students and policy makers better understand the complexity of the energy transition. The Illuminator can also be used to show project planners the optimum energy generation configuration that suits their needs by quantifying various performance indicators. This thesis will use a case study to discuss the project results and show one of many ways the Illuminator can be used. The case study is advising a neighbourhood project planner on building a neighbourhood for 1000 households in Delft, The Netherlands. The contractor can freely change the input parameters of the Illuminator by using the dashboard and can easily connect or disconnect any system module. By doing this, the costumer will not only see the financial and emission benefits of renewable energy but also their optimum configuration and why this is the case.



Figure 1.1: The energy integration demonstrator in clip art image

1.2. general overview

The calculation and visualisation of power flows throughout a
grid is done using a demonstrator kit. In fig.1.2 an overview of the demonstrator is shown. The demonstrator consists of Raspberry Pi's depicted in the colour raspberry. Each client Pi is connected to a "Hardware hub" to which table-top models are connected visualised by the images. The server Pi however is connected to all client Pi's which is also where the visualisation happens through the "Dashboard" using a monitor. This demonstrator is able to show power production and consumption and the influence of changing inputs on these powers. This visualisation is not only done by looking at a monitor but also has a hands on use with the addition of a hardware hub. This is a component which will resemble what is going on in the grid by means of actuators and sensors. The demonstrator will then consist of 3 sub modules. Below is a small elaboration on each sub module existent in the demonstrator kit.

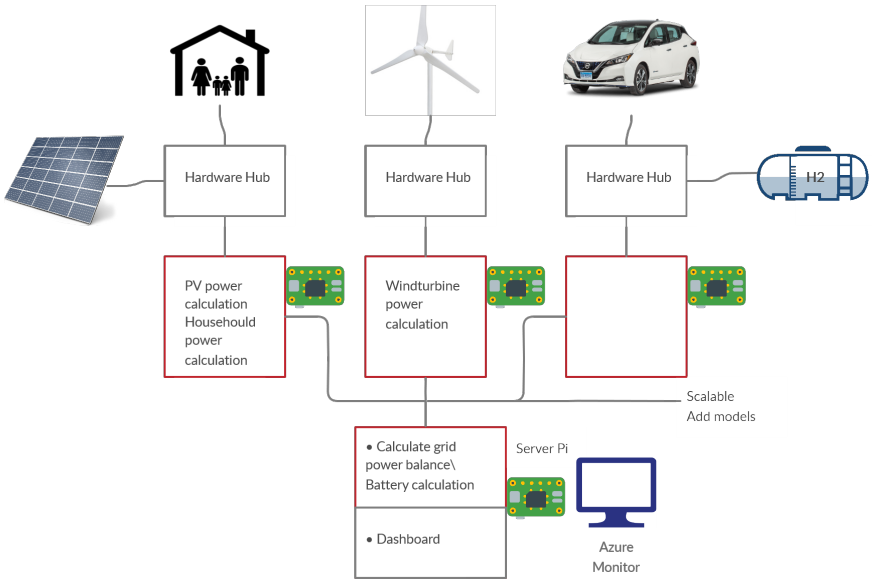

Figure 1.2: Schematic overview of the Demonstrator

Dashboard visualisation

The dashboard is where the demonstrator shows the simulated neighbourhood outputs. The dashboard is also the place where a user can change the neighbourhood configuration. The dashboard depicts several power flows over the course of a single year (2019), but is also able to zoom to an arbitrary day or month. This provides the user with more specific performance data for a shorter time line. Furthermore the dashboard uses all this received power data to make an emission analysis and a financial analysis. From these rigorous analyses the client can make decisions regarding his neighbourhood configuration. For example adding more houses, downscaling the amount of wind turbines or even completely disconnecting all hydrogen tanks.

Data Energy Widget

The Energy Data Widget is going to be the backbone of the demonstrator. It will calculate all values regarding powers and energies by means of Python. This software is implemented on Raspberry Pi's and is able to handle 2 types of inputs, coming from a dashboard or from table-top models. When using Dashboard as main input all relevant weather and consumption patterns are retrieved through an online database as well as type and amounts of components within your grid.
Besides the Dashboard the table-top models are also able to create inputs. However, they just output actual weather data created by sensors and can not change type or amount. The models should be able to operate in both modes.

Hardware hub

The Hardware Hub is a physical setup that is able to emulate information calculated by the Raspberry Pi's, using table-top model representations of energy systems. This implies that if for instance a lot of wind energy is being generated, the table-top wind turbine model will actuate that generated power by letting a DC-motor rotate. Similarly, if the households in the neighborhood are using a lot of energy then the table-top model will actuate that by showing a small lamp shine brighter than if it weren't using consuming a lot of energy.
The Hardware Hub makes it possible to interact and play with these table-top models, it is possible to change loads by rotating potentiometers, add or remove the electric vehicles by using switches and influence generated wind power by rotating the DC-motor. The Hardware Hub is also able to sense real physical information using sensors and is able to send this information back to the Raspberry Pi's. One example the solar PV table-top model, this model is able to sense the irradiance by using a physical solar PV.

1.3. Thesis Structure

This thesis starts with chapter 2 the Program of Requirements. Here the requirements for Illuminator, and the specific requirements for the dashboard will be presented. In chapter 3, 4 and 5, the PV generation, wind generation and neighbourhood power visualisation will be discussed. In these chapters, the reasons for selecting certain input choices or output methods is explained. Although the dashboard doesn't calculate the neighbourhood power values, the inputs are part of the dashboard. How certain inputs have influence on the system performance is an important factor of finding a better performing configuration. In chapter 6 and 7, the life cycle emission analysis and financial analysis methods are discussed. Because this is an electrical engineering, not a computer science project, the emission and finance analysis has been added to the dashboard. These chapters explain the reasons for showing certain performance indicators and how these are obtained. In chapter 8, the implementation of the dashboard is explained. How the Pi's communicate and how the dashboard show the correct graphs is all worked out here. With the focus being on electrical engineering, this chapter is fairly small. After this, chapter 9 will discuss the system results by working out the case study. The dashboard can be used by many different people in a number of ways. The wide range of possible users mean that the best results are subjective. One user might prefer the lowest carbon emissions while another prefers the cheapest electricity bill. Discussing the results in this way, demonstrates the practical use of the dashboard in a way that doesn't reflect on the personal preferences of the dashboard developers. The res Finally, the thesis will end with a conclusions 10 in combination with some recommendations for future work.

Some notes for reading this thesis

Throughout this thesis, the unit of kilowatt-hour (kWh) is used to refer to electrical energy. Normally the kWh a unit of work and equal to 3600 kilojoules (3.6 megajoules), with joule being the SI Base Unit for energy. The reason for using kWh instead of joule, is that kWh is commonly used as a billing unit for energy delivered to consumers by electric utilities. Also, when referring to the entire demonstrator, the term Illuminator is used. When only referring to the visualisation part of the demonstrator, the term dashboard is used. Finally, due to the wide range of users that can potentially use the Illuminator, system choices are often explained in terms of a specific user. In some cases the Illuminator will be seen as an educational demonstrator and other times it can be used as a tool for project developers to learn about potential profits.

2

Program of Requirements

This chapter will describe the requirements which the product must have and would preferably have. Below the requirements are split up for the complete system and sub module. These requirements are divided up into mandatory, necessary to function, and trade-off requirements, enriching the product.

2.1. Requirements for the system

These are the requirements the system should comply with in order for all sub modules to work together correctly and to have a full functioning demonstrator.

Mandatory requirements

- The Demonstrator must be able to calculate the power flows from real-time simulated weather data coming from the hardware hub (table-top models).
- The Demonstrator must show all power flows going into and out of the system and its individual components (for example a solar panel or wind turbine).
- A communication needs to be established between the Pi's.
- Table-top models as well as the dashboard shall be able to visualise information regarding the powers of the system.
- The table-top models must be interactive so that the user can physically influence the individual components such as the wind turbine or solar panel.
- The Dashboard must be interactive as well, so that the parameters can be changed (such as the amount of EV's or solar panels), and this leads to a change in output.

Trade-off requirements

- The resolution of the power flows from the table-top models should have an hourly resolution.
- The dashboard should be easily operable, with only a keyboard and mouse.
- The system should be easy to understand for laymen.
- The system should be built with low cost, no more than €100.
- The performance should not be too slow, preferably within seconds for a new simulation.
- The system should not be too large to make it easy to carry in for example a briefcase.
- The product should preferable also handle extreme cases, such as a hack on the wind turbine, disabling all power flow from that component within the system.
- The Demonstrator should be able to simulate a neighbourhood on in different places, otherwise the area of Delft is preferred for the case study.

2.2. Requirements for the Dashboard

The Dashboard sub-module focuses on visualising relevant neighbourhood energy integration data. Visualising power flow data and storage performance along with relevant inputs on a web-based dashboard. Also, visualising the projected future emission and financal performance by using the relevant power flow data from the energy widget sub-group. The requirements for the Dashboard are listed as follows:

Mandatory

- The dashboard should be compatible with a communication protocol that connects all the Raspberry Pi's
- The Dashboard should be able to visualise both real-time data outputs from the table-top models, or the simulated data from a year overview, using old weather data.
- The Demonstrator must function on Raspberry Pi's.
- The adequate software must be open source.
- The Demonstrator must visualise power flows of models (such as wind turbine or solar panel) received from the energy data widget
- The Dashboard needs to have variable inputs for: photo-voltaic generation, wind power generation and load consumption
- The Dashboard needs to be able to visualise the results of an emission and finance analysis
- The dashboard needs to be easy usable for all clients.

Trade-off requirements

- The performance should not be too slow, preferably within seconds for a new simulation.
- The load should be modelled as a household for the chosen case study.
- The whole system should be easily scalable, implementing more input variables should not be a problem.
- The product should preferable also handle extreme cases, such as visualising a hack on the wind turbine, disabling all output power flow from that component.
- There should also be storage options such as a hydrogen tank and/or Electric Vehicles (EV's) to make the neighbourhood more national grid independent.
- The yearly SoC of the EV's and hydrogen tanks should be visualised.
- The system's operating software is written in Python 3.7 or higher.
- The dashboard should run on a local server, meaning no internet connection is required.

3

Photo-Voltaic Generation

In this chapter the photo-voltaic module of the dashboard is introduced. Photo-voltaics (PV) is a method of generating electrical power by converting solar radiation into electricity with the use semiconductors. Generally, solar PV technology is used on a panel, which is the reason they are widely referred to as PV panels. Multiple PV cells can be connected to one another other and put inside a frame called a module. The term solar panel is often used interchangeably with solar module. The main difference being that sometimes solar panels are composed of multiple modules mounted together. Connecting multiple modules with wires forms an array, which can be scaled up or down to obtain the desired output power. PV panels generate direct current (DC) and considering most residential loads use alternating current (AC) the electricity needs to be converted by an inverter. In 2019, 97.1 GW of PV generation capacity was installed in the world, bringing the total world PV operational capacity to 586.5 GW [3]. PV generation is a significant energy source today and is expected to become an even more dominant role source in the future. When building new residential areas, local PV generation ought not to be overlooked due to its emission (chapter 6) and financial (chapter 7) benefits. This chapter explains the reasoning behind the various dashboard input parameters and the output visualisation method.

3.1. Model Input Parameters

When the Dashboard user wants to design a neighbourhood, there are many possibilities in terms of size, location, geographic, orientation etc. This means that it is of great importance for the Illuminator to be both scalable and have sufficient options for PV parameters. Presenting the user with a wide selection of choices is crucial for finding the best performing PV system configuration that suits the user's specific needs.

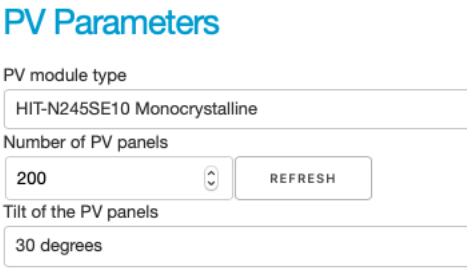

Figure 3.1: Dashboard PV parameter input

In 3.1, the various PV input parameters and method is shown. The reasoning behind these choices and how they influence the PV system is explained in the following subsections.

3.1.1. PV panel type

The material most commonly used for PV modules is silicon, which counts for almost 90% of the PV modules in the world. Fortunately, silicon is the second most abundant element in Earth's crust (after oxygen) with roughly 25% of the earths crust comprised of it. Of the silicon based PV panels, crystalline silicon (c-Si) is the most popular used. In the dashboard, the user can implement the Panasonic HIT Mono N245W(figure 3.2) monocrystalline silicon(mono-Si) module or the JAP60S01-290/SC(figure 3.3) polycrystalline silicon(multi-Si) module.



Figure 3.2: The monocrystalline silicon PV module choice



Figure 3.3: The polycrystalline silicon PV module choice

When comparing figure 3.2 and figure 3.3, there is an easy noticeable difference between the colour of the crystalline silicon PV cells. This is because the cells used in the PV modules are manufactured in a different way. Monocrystalline solar panels have solar cells made from a single silicon crystal, while polycrystalline solar panels have solar cells made from many silicon fragments melted together. Due to the different in manufacturing process, PV modules made from monocrystalline PV cells result in higher efficiencies because the electrons that generate a flow of electricity have more room to move. However due to the less complex manufacturing process, polycrystalline PV modules are cheaper generally speaking. Another difference between the crystalline silicon modules is that polycrystalline usually decays faster in terms of output power and has relative a lower output at high temperatures than monocrystalline. With all this being said, a monocrystalline PV panel isn't necessary better then a polycrystalline panel. PV performance depends on many external factors and also the type of user. Different users could prioritise different performance indicators due to their circumstances. Circumstances the Illuminator user can take into account when designing the optimal PV system for their neighbourhood.

3.1.2. Number of PV panels

The user has the ability to simulate any integer number of panels, see figure 3.1. Freedom to choose the amount of PV panels is necessary for finding the best possible output power. Increasing the amount of PV panels in the system leads to a higher output power. Decreasing the amount of PV panels lowers the output power. Every 10 panels use one inverter, which means that choosing a multiple of 10 panels is recommended in most cases.

3.1.3. PV panel tilt

The PV panel tilt and orientation, depicted in figure 3.4, have a significant influence on the PV system output. The PV panels are mounted on the side of the house roofs. The orientation of the roofs is always at the optimal angle value for PV generation at a specific angle. The user has three choices for the PV panel tilt angle which are: 30°, 35° or 40°. This is useful for finding the optimal tilt angle for a specific location and visualising the influence a sub-optimal angle has on the system. Figure 3.4, shows the difference in sun path for s summer and winter day. Due to this phenomenon the user needs to make a trade-off between the optimal summer an winter tilt.

3.2. Model Output

The PV power output is visualised on the dashboard by plotting the average power generation for each hour in a year (8760 values), see figure 3.5. This method was chosen for several reasons. First of all, with the power

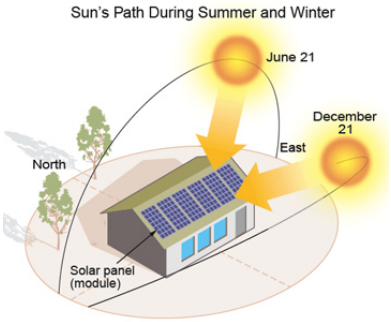

Figure 3.4: The influence of seasonal changes in sun path on the optimal panel angle

output the user can easily see the performance of the system and how certain choices effect the performance. Also, the user experiences the effect of seasonal fluctuations in PV power output. Using a scope shorter then an hour is extremely time consuming, with a years worth of data. When using a minimal scope of one hour, different resolutions for the graph can be visualized. An example of this, is the 24 hour scope graph seen in figure 3.6, this shows the user the difference in PV power output throughout the day.

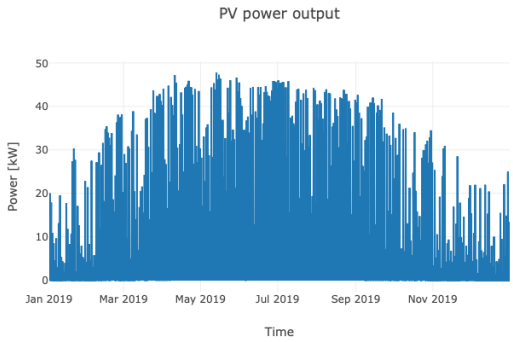

Figure 3.5: PV power output for 2019

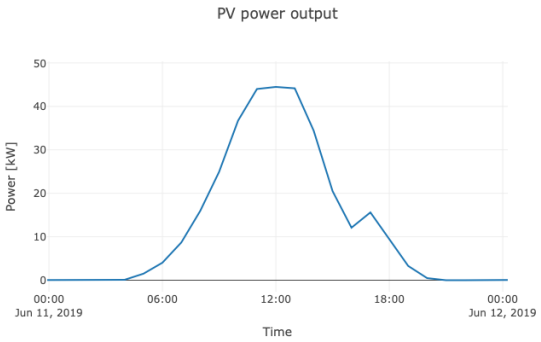

Figure 3.6: PV power for June 11, 2019

4

Wind Energy Generation

In this chapter, the wind module of the dashboard is introduced. Wind energy generation refers to the process of utilising the wind's kinetic energy. When wind moves through a wind turbine, the turbine blades rotate, turning the kinetic energy into mechanical energy. The blade rotation turns an internal shaft connected to a gearbox, which significantly increases the speed of rotation. The increased rotation is used to generate electricity by using a generator. When multiple windmills are placed in close proximity to each other, they are referred to as a wind farm, functioning as a single power source. Generally, wind energy generation is divided in three types which are: utility-scale, distributed and offshore. Utility scale has a range from 100KW and is used to deliver electricity to the power grid. Distributed wind generation has a range of below 100KW and is mostly used to power a few homes or a medium sized business. Offshore wind is when the wind turbines are located in a water body, these turbines are the largest type. Offshore wind turbines are subjected to a more consistent wind flow and have ecstatic advantages due to their remote location.

Across the world in 2019, 58.9 GW of wind power capacity was installed, bringing the total world wind operational capacity to 622.7 GW [3]. Wind power generation is the largest growing renewable energy source today already supplying 24.5% of all renewable sources. Like PV generation, wind generation has significant emission and financial benefits, which are discussed in chapters 6 and 7. This chapter, explains the reasoning behind the various input parameters and the output visualisation method implemented in the dashboard.

4.1. Input

For the same reasons as the PV module described in chapter 3, certain wind parameters should be variable. The main difference between the PV and wind modules, is the smaller freedom of choices to be made for wind generation. In figure 4, the two input choices are shown. The turbine model and the amount of wind turbines, should give the user enough freedom to find a optimal wind generation system. The reasoning behind choosing these inputs and how they influence the wind system, is explained in the following subsections.

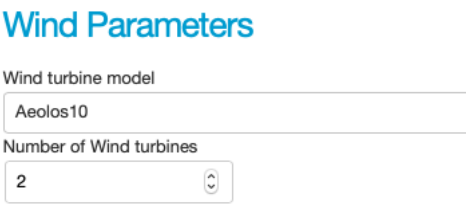

Figure 4.1: Dashboard wind parameter input

4.1.1. Wind turbine model

The most important choice regarding simulating wind generation, is the wind turbine model. The three choices for wind turbine model are shown in figures 4.2, 4.3 and 4.4. The ability to choose a different wind turbine model, makes it possible for the user to see the performance difference by visualising the power generation.



Figure 4.2: The Aeolos V - 10KW



Figure 4.3: The Hummer - 60KW



Figure 4.4: The Vestas V90 - 2MW

The Aeolos 10 wind turbine. shown in figure 4.2, is a good choice for users interested in simulating a relative small neighbourhood. The Aeolos 10 is rated at 10KW and uses vertical blades which is nice in a small sized neighbourhood of a few houses. For midsize neighbourhoods with more than 50 houses, the Hummer 60 is a better choice shown in figure 4.3. With a 60KW rated output power the turbine is still considered a distributed type wind turbine, but has the ability to power a considerable larger load size then the Aeolos 10. For simulating large size neighbourhoods of more then 300 houses, the Vestas V90 rated at 2MW, becomes a viable option.

4.1.2. Number of wind turbines

The option to choose the amount of wind turbines has two main reasons. First of all, the user needs to be able to scale the amount of wind turbines to get closer the optimum wind power output. Without an option to choose the amount of wind turbines, the user can only use 3 different wind power outputs. With this option, the user can multiple smaller wind turbines to get the desired wind power output for the neighbourhood. For example, the user wishes to simulate a 40KW rated power output and obtains this by using four Aeolos 10 turbines. This method gives more freedom for the user, this is important because the user can utilise this freedom to obtain a better performing system that suits their need.

4.2. Output

The load power output is visualised on the dashboard by plotting the average power generation for each hour in a year (8760 values), see figure 4.5. This method was chosen for several reasons. First of all, with visualising the power output, the user can quickly see the performance of the system and how certain choices effect the performance. Another reason is that the user can see both the seasonal and daily fluctuations of the chosen wind system on a specific location. Also, the ability to zoom to a smaller scope, allows the user to analyse the desired system in a different time range shown in figure 4.6.

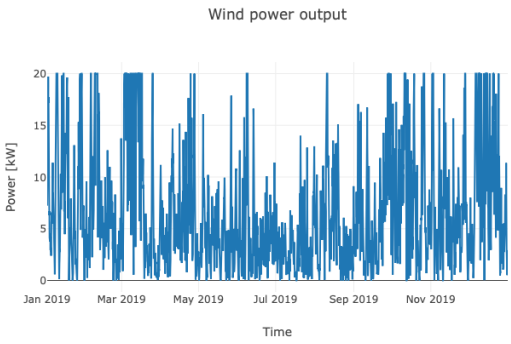

Figure 4.5: Wind power output for 2019

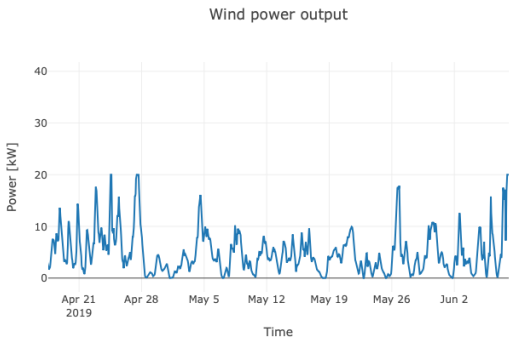

Figure 4.6: Wind power output zoomed to a smaller scope

5

Neighbourhood Power Visualisation

This chapter discusses the non-energy generating power flows visualised in the Illuminator dashboard.

5.1. Neighbourhood Load

The neighbourhood load refers to the houses connected to the Illuminator grid.

5.1.1. Load input

The dashboard load input parameters are shown in figure 5.1. The user can choose the type of energy consumers living in the neighbourhood houses and the number of homes connected to the grid.

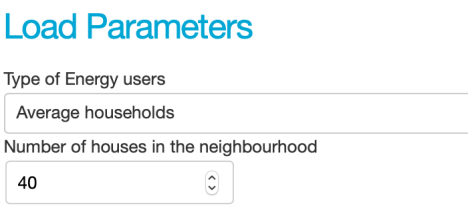

Figure 5.1: Dashboard load parameter input

Simulating the energy consumption of a house single house is a very complex process. The way the houses are build, the personal behaviour of the inhabitants, the appliances inside the house and the list goes on. The main reason for choosing these dashboard inputs, is to keep the product simple to use. How many houses are located in the neighbourhood and how energy conservative are the residents. These possibilities should give the user enough freedom to change the load input to their own preferences.

5.1.2. Load output

The load power output is visualised on the dashboard by plotting the average power generation for each hour in a year (8760 values), see figure 5.2. As can be expected, the load tends to be higher in the winter months compared to the summer months. The seasonal fluctuation are due to the colder winter needing more energy for heating then the warmer summer, also the lower temperature leads to a higher transmission loss. The zoom function, shown in figure 5.3, shows the user how load consumption fluctuates in a shorter time frame. The load consumption is lowest in the night and highest when people return from work around 18:00. A low consumption in the middle of the night can be expected due to the residents sleeping. When people come home from work they often turn on the TV, cook diner, take a shower and other energy intensive activities.

5.2. Storage

The power grid is a complex system where, at any given moment, the electricity supply and demand needs to be equal. When monitoring the power grid, constant supply adjustments need to be made for predictable changes in demand, such as the daily activity patterns. Also, unexpected changes from events like, equipment

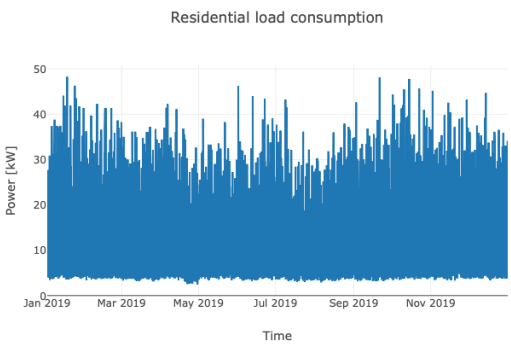

Figure 5.2

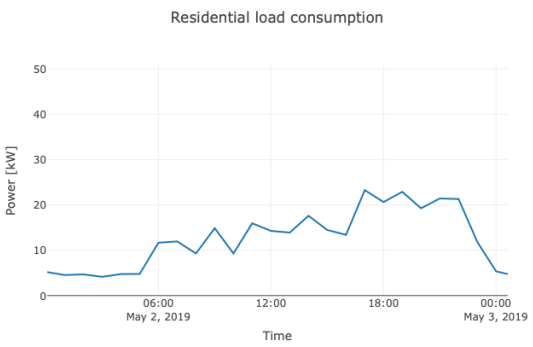

Figure 5.3

overloads, storms sudden weather changes. Energy storage plays an important role in this balancing act and helps to create a more flexible and reliable grid system. Another important role of energy storage is that it helps balancing the intermittent electricity generation from renewable sources like, wind an solar. If the wind doesn't blow or the sun doesn't shine, the power generation can not be scaled up to meet demand. By using storage, the extra generated wind power and PV power during the day, can be stored to be used at a later time. The Illuminator used two types of storage: Electric vehicle storage (EV) and hydrogen fuel tanks. The reasoning and implementation of both storage types is discussed in this section.

5.2.1. Storage input

In the Illuminator dashboard, the user can change 2 storage parameters. How many Hydrogen tanks and how many electric vehicles to use as energy storage. The Electric vehicles used for storage are Nissan leafs with a 56kWh capacity. For the hydrogen tank, the MAHYTEC 300L tank that stores about 4,2kg hydrogen at 500bar. Coupled with an electrolyzer, the generated hydrogen can be directly stored without compression resulting in a capacity of 396kWh. The reason for choosing both storage models was that both technologies are up and coming and can be interesting to implement in specific cases. The user maintains a high level of simulating freedom with both technologies available as inputs, but also the ability to choose one or neither storage technologies.

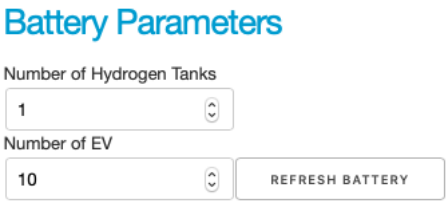

Figure 5.4: Dashboard storage parameter input

5.2.2. Storage Output

The dashboard visualises the storage performance by plotting the State of charge (SoC) for every hour in the year 2019, shown in figures 5.5 to 5.6. The state of charge (SOC) of a storage technology refers to the capacity that is currently available compared to the rated capacity. The value of the SOC varies between 0% and 100%. If the SOC is 100%, then the battery is considered to be fully charged, whereas a SOC of 0% indicates that the battery is fully discharged. In practical applications, the SOC is not allowed to go beyond 50% [27] and therefore the cell should be recharged when the SOC reaches 50%. Similarly, as a cell starts ageing, the maximum SOC starts decreasing. This means that for an aged storage unit, a 100% SOC would be equivalent to a 75%–80% SOC of a new cell.

Electric vehicle storage

Most of the time an electric car is not used being used on the road for transportation but just stands around. Fortunately, the energy stored in the car battery can be returned to the power grid. This is helpful at times of high energy demand in the grid and of no disadvantages to the car owner. In the figure 5.5, the user can see the performance of the EV storage. If the SoC of the EV storage falls to around zero in the graph to often, this is an indication that the storage element is depleted and the user might want to use more storage possibilities. On the other hand, when the SoC from the electrical vehicles is always around 80%, this is an indication that the system uses an unnecessarily expensive big storage system. It should also be noted that the EV's are also loads in the neighbourhood. The Nissan's use 4 kWh per working day, with 4 working days in the week.

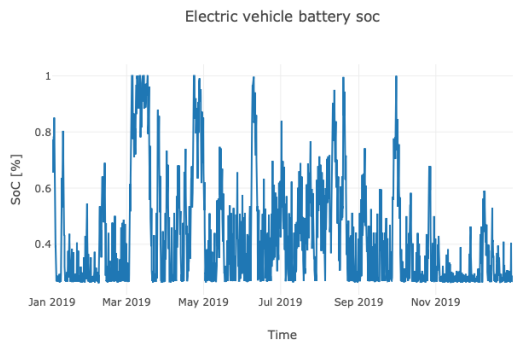

Figure 5.5: Electric vehicle SoC for 2019

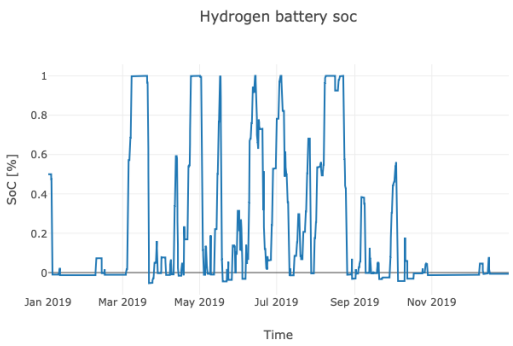

Figure 5.6: Hydrogen fuel tank SoC for 2019

Hydrogen fuel cell storage

A fuel cell is similar to a battery, in the sense that it generates electricity from an electro-chemical reaction. In the Illuminator, the fuel cell runs on the oxidation of hydrogen. The hydrogen gives the fuel cell its name, although there is no combustion involved which is a common misconception. The hydrogen storage output is analysed in the same way as the electric vehicles explained earlier. Accurately projecting the required size of storage is useful because it ensures the users doesn't overpay for his storage. It also gives a sense of scale when used as an educational tool.

5.3. Grid

The grid module of the dashboard, refers to the bidirectional electricity flow between the neighbourhood and the national power grid. When the storage elements described in section 5.2 are fully charged, the additional generated electricity is sold back to the grid. When the storage elements are depleted, the additional required electricity is supplied by the grid.

5.3.1. Grid power flow

The grid can be used as a very useful integration analysis tool. In figures 5.7 and 5.8, the bidirectional flow for a neighbourhood in 2019 is shown. First of all, the amount of negative/positive power flow indicates to what extent the neighbourhood relies on the power grid for electricity. A grid power output with only positive values shows the user that the neighbourhood is relying significantly on the national power grid for electricity. When the power flow shows both, taking from the gird and supplying power, the neighbourhood is selling when generation is abundant and buying when their own intermittent renewable energy generation cannot supply the demanded power. Exclusively negative values for the grid module, means the neighbourhood is solely selling abundant electricity to the grid. If a user is interested in a islanded grid, described in section 5.4, the power flow should show a constant zero value for the year.

5.3.2. Grid electricity mix

When taking electricity from the national grid, the generation source is not necessarily know. Generally, when signing a contract with a utility company, the generation source of is known. However, this is not always the case. For the user, it can be helpful to see where the electricity consumed originates from. This is done by visualising the source ratios in pie charts, shown in figure 5.9. When the neighbourhoods own generation is

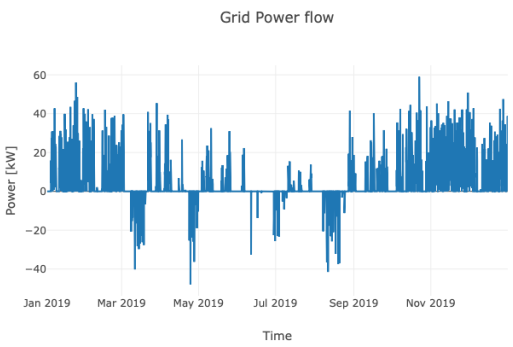

Figure 5.7: The bidirectional grid power flow 2019

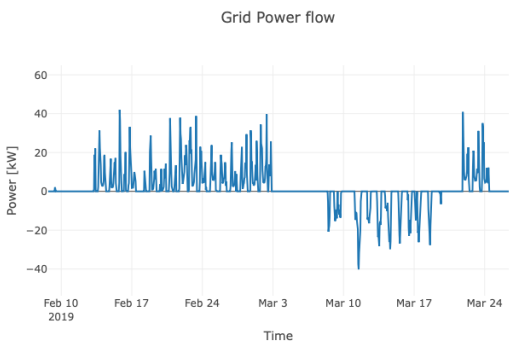

Figure 5.8: The power flow output zoomed in for May 2, 2019

insufficient and electricity is bought from the grid, the use of external sources is added to the total energy consumption. This is done dividing every KWh taken from the power grid, in the various sources that are used for supplying the dutch national with electricity. According to [15], the mix in 2019 was: 45% natural gas, 32% coal, 13% renewable sources, 4% nuclear and 3% other sources.

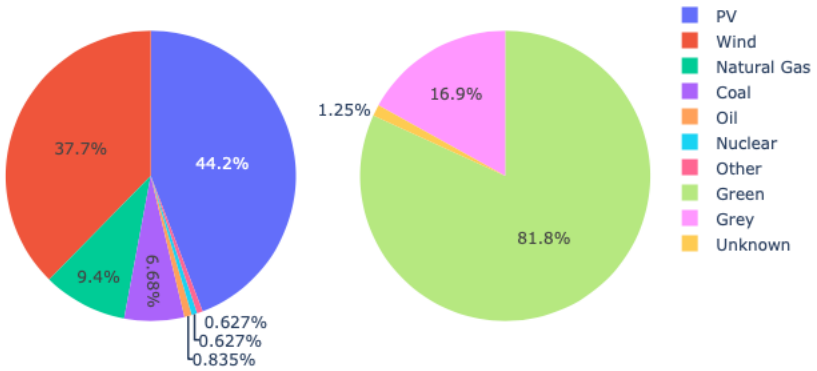

Figure 5.9: Share of total energy consumption in terms of energy source

5.4. Stand-alone power system

Some users might want to design a neighbourhood that can operate in an 'islanded' or isolated mode. This essentially means that the microgrid should be self-sufficient with it's own local generation sources without support from an external grid. Another way to explain this would be to say, the islanded microgrid must be able to generate and store enough electricity to supply the neighbourhood load demand. To obtain this goal this goal, the solar and wind generation systems need to be sized to meet the demand. If the neighbourhood system doesn't produce enough electricity at any given moment, a black-out will occur and the living standards will be sub optimal. This is visible in the dashboard when the SoC of both storage methods, shown in figures 5.5 and 5.6, goes to 0%. When this happens the grid needs to step in and it can be concluded that the neighbourhood isn't able to operate in stand-alone mode. However to realistically operate in stand-alone mode, the Soc should be more then 50% on its lowest point because the weather is known to fluctuate from year to year.

6

Life-Cycle Emission Analysis

The world is warming up due to the amount of greenhouse gasses(GHG) released in the earth's atmosphere. This could have a catastrophic effect on the world so measures are taken to reduce the GHGs to zero. This is done by reducing the emissions of electricity generation sector, because this sector is responsible for over 40% of the GHGs in the world, as of February 2020 [25]. To do this more and more renewable energy sources are employed in place of fossil fuel sources. As can be seen in figure 6.1 the GHG emissions have been reducing since the rise of renewable energy sources which consist mostly of solar and wind. This does not mean

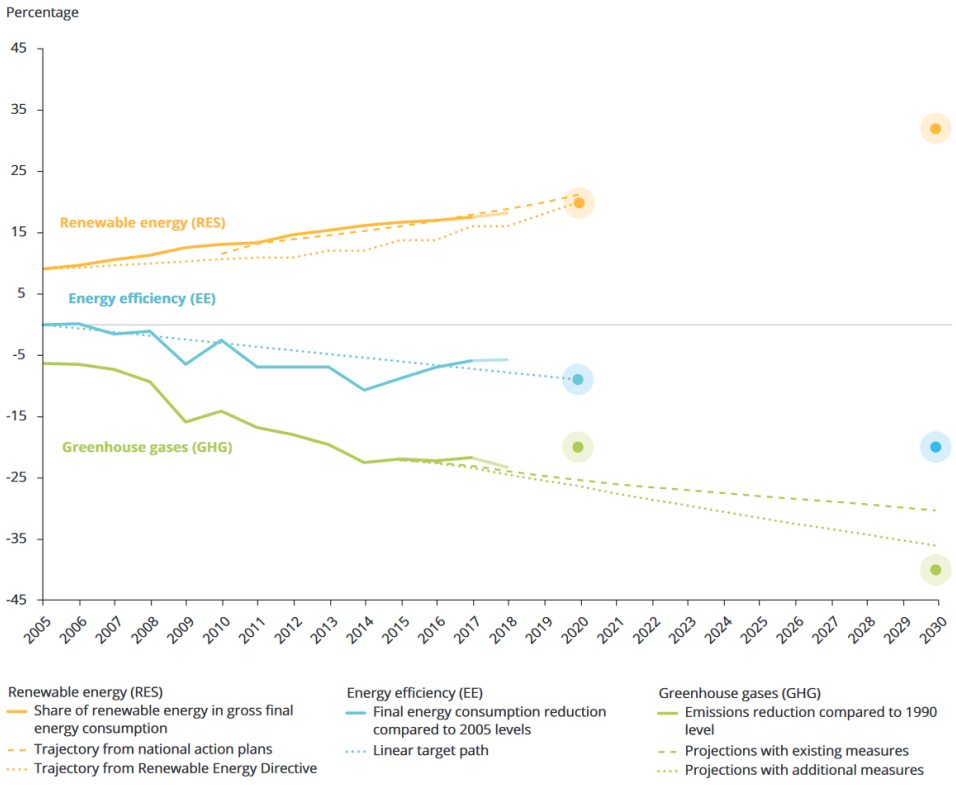

Figure 6.1: EU progress towards 2020 and 2030 targets on climate and energy[38].

that these generation sources are carbon neutral. They still have emissions of GHGs in the different part of their life cycle, for instance in the production phase fossil fuels are used to manufacture the PV modules or turbines. Furthermore the material extraction releases also a lot of GHGs.

In this chapter the life cycle emissions of the renewable energy sources are analysed and compared to the fossil emissions from which a conclusion can be made for our case study.

6.1. PV Systems

Life cycle emission of the PV system has been evaluated by doing a cross analysis of multiple studies, among which meta analyses with one having screened almost 400 studies. The focus is on the GHG which is used as measurement for calculating the Global Warming Potential(GWP). In figure 6.2 the life cycle of a PV system is given.

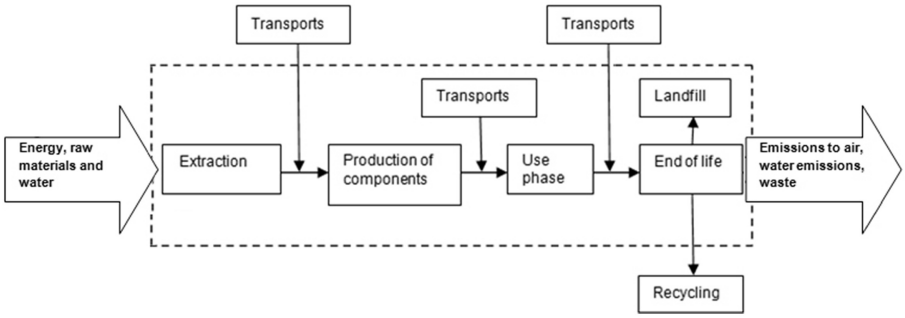

Figure 6.2: A schematic of the life cycle of a PV system[28].

In the figure the transport is also given, which also causes GHG emissions but the transport related emissions are considered insignificant according to data [34]. More than 70% of the life cycle emissions of a PV system is from the extraction and production phase[31]. The use phase itself has only a small indirect emission due to maintenance. The end of life phase has two ways it could go. The used up PV system could be purified and disposed in a landfill, this step takes up more than 40% of the total Global Warming Potential(GWP) of the system and this is of course not environmentally friendly. The better option would be to recycle the PV system, which will reduce the GWP by being used in a new PV system. This would reduce the production phase emissions which is a large part of the life cycle emissions of a PV system [39].
A few assumptions have been made before reaching the final result.

- First of, the studies agree that there is no substantial difference between the life cycle emissions of ground mounted PV systems and roof mounted PV systems, i.e. no substantial difference between crystalline silicon(c-Si) and thin film PV[30].
- Furthermore most studies disregarded the Balance of System(BoS), which is the module supports, cabling,and power conditioning. This only makes a difference of at most 4g CO_2-eq. (CO2 equivalent) per kWh[16].
- As stated earlier the transport emissions are also disregarded[34].
- Most of the studies are done in Southern Europe or a place with equivalent irradiance(1700kWh/m2/yr), which is much higher than the 1000kWh/m2/yr irradiance measured in The Netherlands. This would increase the CO2 equivalent emission per kWh[29].
- Using recycled PV systems and recycling them reduces life cycle emissions by up to 24% [39].
- As the technology and manufacturing processes become more efficient, the life cycle emissions of photovoltaic panels will continue to decrease[42].

The results of the studies can be seen in table 6.1.

Table 6.1: The results of the studies used for the dashboard, given in grams of CO2 equivalent per kWh generated by the PV system.

PV	Magrassi2019	World Nuc. Org.	Fthenakis2008	Nugent2014	Hsu2012
GHG[g CO2eq./kWh]	44.3	85	45 mono/35 poly	49.9	45
type	mono	all	mono/poly	all	mono/poly
mono-Si weight	4	1	1	2	2
poly-Si weight	0	1	2	3	4

From these results a weighted average is taken to have a value that is most accurate. The most recent study has the most importance so that is weighted highest as can be seen in the table 6.1.

Magrassi2019[28] uses the most recent data therefore it has the largest weight. It does have some differences from our chosen mono-Si PV panel option. For example, the efficiency is 14% in their analysed model and our panel has an efficiency of 19.4%. This in contrast with their 30 year lifetime, which is 5 years more than our panel, compensates the difference a bit. However there are more smaller differences which make it not the ideal study, so this is why the other studies have been included to the calculations.

The World Nuclear Association[42] uses old data and in the old data the PV life cycle emissions are comparable to fossil fuel energy generation methods. This is mostly due to PV systems not being very efficient 30 years ago. That is why they got values of over 700 g CO2-eq per kWh and got a lower weight in the calculations. The value seen in the table for World Nuc. Org. is an average of the results of studies ranging from 1993 to 2007. It should be noted that even though this meta analysis is done by the World Nuclear Association, they tried being as unbiased as possible by using only credible sources from 3rd parties like universities and industry publications which are independently verified.

Fthenakis2008[16] uses more recent data from 2004 to 2006. They tested for both polycrystalline silicon (poly-Si) and monocrystalline (mono-Si). In the dashboard both types are given as options so for both there has been calculated an weighted average. Furthermore this study takes into account the BoS but this does not make a big difference so most studies have neglected including this.

Nugent2014[31] is also a meta analysis but this time 153 studies are critically screened to find the most relevant, recent, rigorous, original, and complete life cycle assessments. This makes this study very useful the only reason this paper got a lower weight than magrassi2019 is due to it being less recent. One thing to note is that in this analysis Hsu2012 is also included, this may give Hsu2012 a hidden extra weight but this totally justified. The value shown in the table is the average value over all types of PV panels. The values found for mono-Si was a median value of 46.5 g CO2 eq. per kWh and for poly-Si a median value of 37.5 g CO2 eq. per kWh.

The reason why Hsu2012[24] has a bit higher weight is because they only analysed the life cycle emissions for mono-Si and poly-Si. This is very useful for the dashboard since these are the types used in the dashboard. Furthermore the method they used is also very rigorous. They screened 397 studies for recentness, completeness, right description for life cycle emission and more. They where left with 13 studies on crystalline silicon(c-Si) that met their minimum standards of quality, transparency, and relevance. They found for mono-Si a median value of 40 g CO2 eq. per kWh and for poly-Si a median value of 47 g CO2 eq. per kWh. The value seen in table 6.1 is the average of the 13 studies. In the calculations the mono-Si and poly-Si are calculated separately but for readability of the table the average is shown.

The formula to calculate the weighted average is,

$$\overline{W} = \frac{\Sigma_i w_i \cdot GHG_i}{\Sigma_i w_i}. \tag{6.1}$$

With w_i the weight of study i and GHG_i the greenhouse gas emission for study i. Finally in table 6.2 the final values derived from equation 6.1 and used in the dashboard are given.

Table 6.2: The weighted averages of the life cycle emissions of mono and poly crystalline silicon PV panels in gram CO_2 equivalent per kWh.

Type Of Panel	mono-Si	poly-Si
$\overline{W}$ [g CO2eq./kWh]	47.02	51.55

6.2. Wind Turbines

To find the life cycle emissions of the wind turbines to implement in the dashboard the study of Raadal et al.[33] is critical. They did a meta analysis of 63 studies and found for different wind turbine capacities different emission values. They found that the mean GHG emission decreased for decreasing turbine capacity, showing that "bigger is better". Their results is shown in table 6.3 are all based on a lifetime of 20 years for the turbines.

Even though wind turbines are almost emission free in the use phase they still have some emissions during production phase and end of life phase, but most of the emissions are from transportation of the wind turbine components. Furthermore it was found that recycling the turbine decreases the emissions of the

Table 6.3: The life cycle emissions for every range of turbine capacity[33] and the life cycle emission of our chosen Vestas V90[19], with the GHG given in gram CO2 equivalent per kWh. The lifetime of all wind turbines is estimated to be 20 years.

Wind Turbine Capacity	GHG
<100kW	47.0
100 - 500kW	20.8
500kW - 1MW	13.9
1MW - 5MW	10.4
2MW (Vestas V90)	7.2

turbine significantly[22]. The emissions due to transport are a large part of the emissions with one source claiming it to be 90% to 99% of the total GHG emissions[33]. A big discussion in the wind energy sector is weighing if onshore wind energy is better than offshore wind energy. From the perspective of life cycle emissions, almost all of the literature agrees that offshore has more environmental impact than onshore[7].

For the Vestas V90 a very specific study is done on this turbine[19]. This study is very comprehensive, for example, they individually assess all components within a wind turbine. Which is composed of around 25,000 parts. That is why for the dashboard this turbine uses a different emission value than the standard values from Raadal et al.[33].The GHG emission value can be seen in 6.3.

The other turbines, Aeolos V-10kW and The Hummer-60kW, which are given as options in the dashboard don't have a specific study done on them so for those the value from raadal et al. is used for their GHG emissions.

6.3. EV and Hydrogen Batteries

In this section the life cycle emission of EV batteries and hydrogen batteries are analysed. The EV batteries are the Electric Vehicle storage described in 5.2, the Nissan Leaf with a 56kWh battery capacity, and the hydrogen battery is the MAHYTEC 850L tank that stores about 4.2kg hydrogen at 60bar.

6.3.1. EV Battery

The EV batteries carbon emissions range from 61kg (61000g) per kWh up to 106kg (106000g) per kWh. The low end of the scale is due to the battery being produced with 100% renewable energy sources. Furthermore it is important to note that EV batteries will become "greener" in operation the more renewable energy is used to recharge the batteries[14].

A report concerning the Nissan Leaf has been used to find the life cycle emissions of the EV battery. That report used a Nissan Leaf with 30 kWh capacity, however our Leaf has a capacity of 56kWh so the carbon emission values are almost twice more. The values found are 175kg CO2 per kWh[23], thus the value for our Leaf is 327kg CO2 per kWh. This value is found by doing a cross analysis of 11 studies with 10 of them being published in the last 5 years.

However since the EVs in the neighbourhood are charged by PV systems and wind turbines, the use phase of the EV battery has no emissions. Therefore only the emissions during production are important to us. For a 59kWh battery the CO2 equivalent emission is 14,910 grams[13]. When scaled to 56 kWh this will be 14.15kg CO2eq. per battery per lifetime, with 12 years being the estimated lifetime of the EV battery[13]. Which will be 1.18kg CO2eq. per battery per year. This value is lower than what is described above, because the 61kg CO2eq/kWh value assumes the car is charged by a grid which is heavy on fossil energy sources.

6.3.2. Hydrogen Battery

There were two options for choosing long term energy storage. One was methane and the other hydrogen. The reason hydrogen won over methane is because hydrogen is superior to methane in both process cost and simplicity[5]. Since methane is produced from hydrogen, it is an unnecessary step with unnecessary costs. Furthermore the complete process efficiency of a power-to-hydrogen-to-power process is in the range 34–44%, while for power-to-methane-to-power it is only 30–38%[26].

The main parts where hydrogen battery emits GHGs is during its production and when transferring power to hydrogen. If the power is provided by non renewable sources like natural gas and coal, the life cycle emissions increase significantly[35]. However for our study case this is not the case. The hydrogen battery is injected with excess power from the wind turbines and PV systems.

The hydrogen battery has a life cycle emission of 1782g CO2 equivalent per kg hydrogen[43]. The cho-

sen hydrogen battery has a capacity of 4.2kg, so the total life cycle emission per battery will be 7485 g CO2 equivalent. Dividing this over the capacity of the hydrogen battery, which is written in 5.2.1 to be 396kWh, the carbon emission equivalent per kWh is 18.9 grams. What will be implemented in the dashboard is the total emission per year. The service life of the battery is 20 years[2], so 7485grams divided by 20 years is 374.25grams CO2eq. per battery per year.

6.4. Fossil Sources

The dutch "grey" grid power CO2 emission varies from 0.48 to 0.63 kg CO2eq/kWh[10]. The greenhouse gas emissions of every source connected to the grid are separately investigated and combined by the same percentages as they are available in the dutch grid. This is done to have a more clear and thorough emission analysis, since our own set of PV systems and wind turbines are used on the microgrid. As can be seen in figure 6.3 the carbon emissions for renewable sources are much lower than the fossil sources. If the dutch grid used more renewables the CO2 emission values for renewables would be even lower.

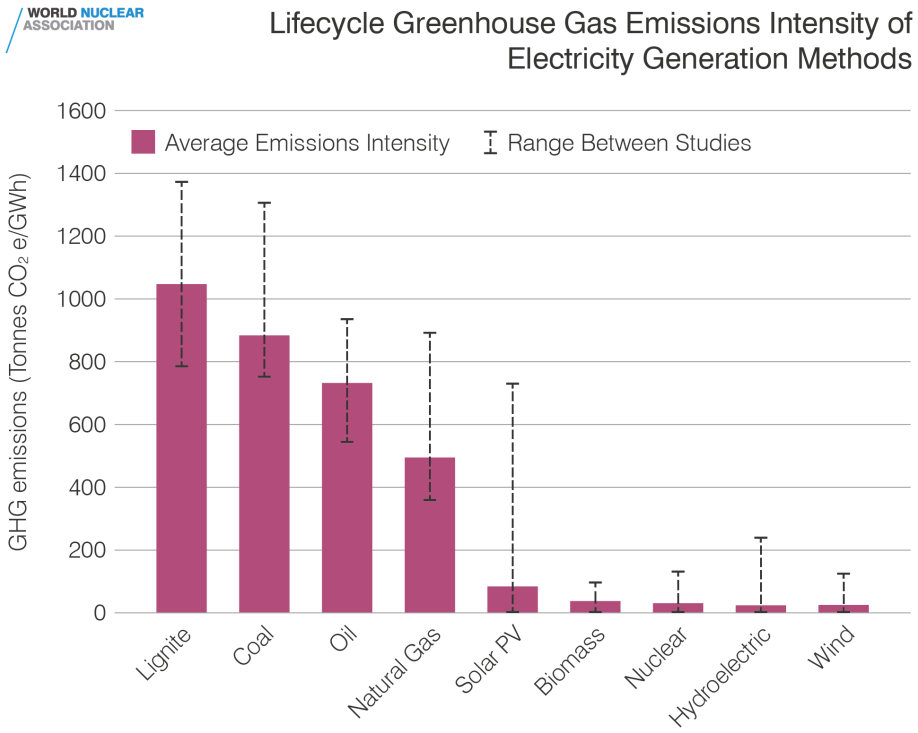

Figure 6.3: The emissions of all energy sources by the World Nuclear Organisation [42].

The mean GHG emission values of the fossil sources from figure 6.3 are given in table 6.4. The values might be a bit outdated but the difference and importance is not large enough to do a more thorough investigation.

Table 6.4: The fossil sources used in the national grid and their respective greenhouse gas emissions given in grams of CO2 equivalent per kWh[42]

Energy Source	coal	oil	natural gas	nuclear
GHG Emissions [g CO2eq/kWh]	888	733	499	29

Other GHG emissions

The CO2equivalent values are actually not all carbon dioxide gasses. It is a combination of carbon dioxide(CO2), Methane(CH4), Nitrous Oxide(N2O) and much more. These are called greenhouse gasses because they all have a certain Global Warming Potential(GWP). Even though Methane is released in far smaller values to the air, it is much more harmful to the environment. That is because its GWP is 28[21], which is 28 times the GWP of CO2, which is 1. This is also the reason why greenhouse gasses are given in CO2 equivalent values, because the GWP is normalised to CO2.

Therefore it is of much importance to keep track of the Methanes and Nitrous Oxides released into the air. They are much more harmful than Carbon Dioxide. Nitrous Oxide even has a GWP of 265[21], which means its 265 times as harmful as CO2. In table 6.5 the individual greenhouse gasses are shown in percentages of the total emissions per energy source.

Table 6.5: The greenhouse gasses are shown individually in percentages of the total CO2equivalent for each source given in the previous sections[12][4][36][11][40][17][20].

Greenhouse Gasses	coal	oil	natural gas	PV	Nuclear	Wind <100kWh	Wind >1MW
Carbon Dioxide	85.5%	90.1%	93.8%	83.6%	65.1%	94.1%	81.7%
Methane	6.3%	9.1%	0.23%	11.2%	0.07%	0.02%	1.39%
Nitrous Oxide	2.0%	0.74%	0.11%	0.25%	0.33%	0.05%	1.71%
Rest	6.2%	0.06%	5.91%	4.95%	34.6%	5.83%	15.2%

Final Result

In figure 6.4 all previous sections are implemented in the dashboard. The sum of all the emissions of all sources is given above the graph with a parallel to how many trees are needed to absorb that amount of carbon for a 100 years. The total emission of every source is splitted into the individual greenhouse gasses. As is easily seen from the graph, carbon dioxide is a large part of the greenhouse gas emissions.

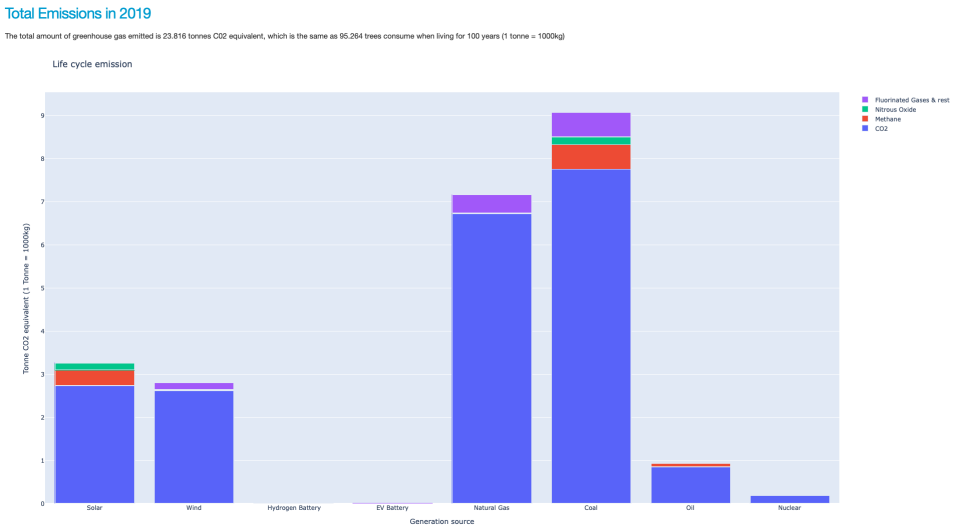

Figure 6.4: The greenhouse gas emissions for all energy sources on the grid as shown on the Dashboard. The total emission is also given on top of the graph, with a comparison of how many trees absorb that amount of carbon in 100 years.

7

Financial Analysis

In 2019, global clean energy investments totalled an impressive $363.3bn [6]. However clean energy investment is still relatively small compared to the $3325.4bn invested in oil and gas exploration [41]. In figure 7.1, the new investments in clean energy between 2006-2019 are shown. This figure shows that yearly clean energy investments have more than tripled from $120.1bn to $363.3bn. Maintaining this growth pattern is crucial for a future world with zero carbon emissions from electricity generation.

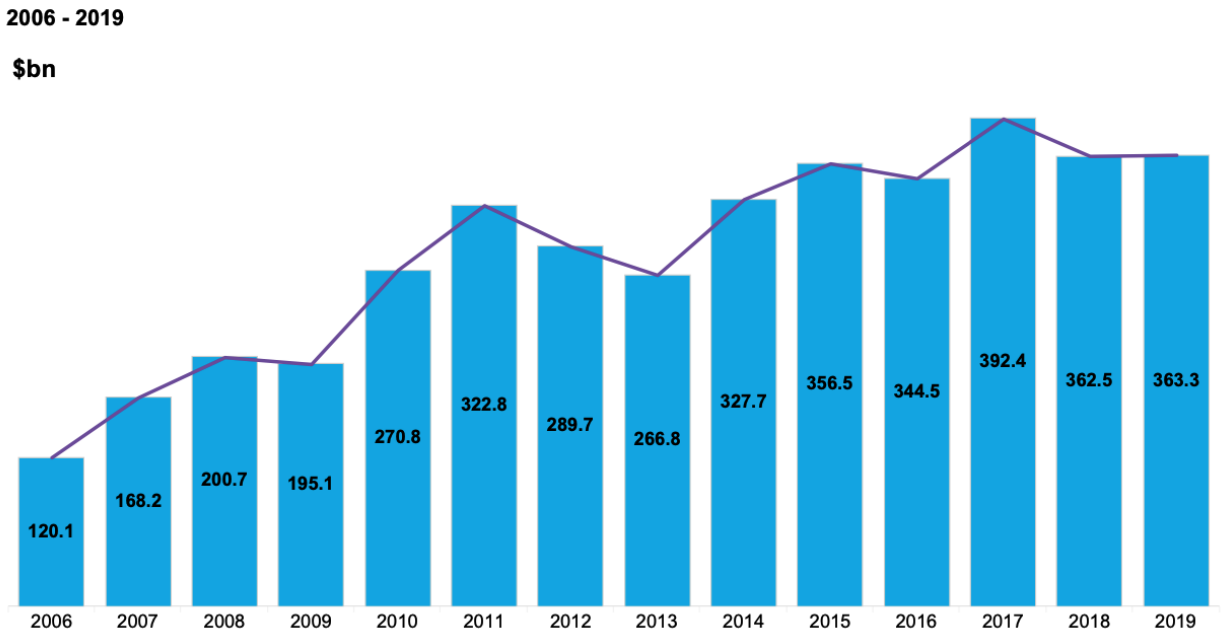

Figure 7.1: Global new investment in clean energy 2006-2019

If there are insufficient new investments in the renewable energy industry, there will be no additional research progress and the installed clean energy capacity will not grow. Investors are interested in profit and fortunately clean energy is highly competitive, cost effective and expected to become even more financially attractive in the near future. With the use of various financial performance indicators, investors can determine if a project is worth investing in. The Illuminator dashboard shows three important financial performance indicators which are: the levelized cost of energy (section 7.1), expected payback time (section 7.2) and a yearly cost savings analysis from 2019 data (section 7.3). In this chapter, the chosen financial performance are discussed along with the reason for using them on the dashboard.

7.1. Levelized Cost of Energy

LCOE, or levelized cost of energy, is the ratio between the total costs and the total electricity produced in a lifetime for a energy source. The LCOE is a useful indicator because it allows energy generation sources with different lifetimes and different capital costs to be compared with a single performance indicator. The LCOE is based on the predictions of both cost and electricity yield, which means there are two major downsides to consider. First, dispatchability, which is the ability of a generation plant to increase or decrease its energy production quickly as demand swings.

Secondly, the extent to which the availability profile matches or conflicts with the market demand profile. The Illuminator has a maximum scope of 1 year, while the PV panels have a expected lifetime of 25 years and the wind turbines are expected to be in service for 20 years. To present reliable data which can be used for multiple neighbourhood implementations, various assumptions have to be made. In equation 7.1, the formula for calculating the LCOE is given. The I_0 in equation 7.1, refers to the initial investment cost made for an energy generation source. A_t is used for the total annual cost in a year t. The E_t is the Annual energy yield and r = the discount rate. The discount rate is essentially used to discount the future value against today's. This is done because, otherwise someone could invest his funds today, get a return, and then invest a larger amount tomorrow. Due to the returns being energy, the Illuminator discounts future production – accomplishing the same goal.

$$LCOE = \frac{I_0 + \sum_{t=1}^{N} \frac{A_t}{(1+i)^t}}{\sum_{t=1}^{N} \frac{E_t}{(1+i)^t}} \tag{7.1}$$

In the tables A.1, A.2 and A.3, a breakdown of the obtained cost values is given. The O&M cost in the tables refer to the operation and maintenance cost. The cost price of the PV panels and inverter in table A.1 were acquired from internet wholesalers to get the most realistic price. The other cost data is taken from the National Renewable Energy Laboratory (NREL) [18]. The NREL is a national laboratory of the U.S. department of Energy, which means that the prices were given in dollar ($). Also, a number of costs such as permitting and labour are different to that of the Dutch market. After considering the potential influence of this difference in various costs, it was chosen to use the NREL acquired data without conversion. The reasoning behind this was primarily that, the Illuminator is a demonstrator and the potential difference in demographic costs is small enough to keep the Illuminator information still relevant. The currency issue was simply solved by taking the dollar euro currency exchange rate at June 1, 2020. In figure 7.2, the PV size categories are shown. The Illuminator is designed to simulate a neighbourhood, so only the residential and commercial categories were implemented in the calculations. The inverter is chosen so that 10 panels use the same inverter, meaning 20 panels use 2 inverters and 21 panels use 3 inverters.

Sector Category	Description	Size Range
Residential PV	Residential rooftop systems	3 kW – 10 kW
Commercial PV	Commercial rooftop systems, ballasted racking	10 kW – 2 MW
Utility-Scale PV	Ground-mounted systems, fixed-tilt and one-axis tracker	> 2 MW

Figure 7.2: PV size categories for different scale PV systems.

The investment costs for the "Vestas V90" and 'Hummer 60" in table A.2, are derived from the international journal of energy and environment [37]. The report shows the results of 12 different studies into capital cost. Due to the limited information available from the turbine producers, the average values of the studies no older then 10 year old where taken. This resulted in a total initial cost of €1228.00 per kW rated output power for the Utility size Vestas V90 and €1781.00 for the Hummer 60. For the "Aeolos 10kW" wind turbine the current prices are easily available from various turbine wholesalers. The O&M costs where not specified and are known to be inconsistent. According to [32] a yearly O&M cost of 2% investment cost is reasonable, so this was used for all three turbines. Also, it should be noted that the price for a small scale wind turbine is more general when buying a 10kW rated compared to ordering a utility scale 2MW turbine. This is the reason why a projected cost was used for the Hummer 60 and Vestas V90 while taking the current wholesale price of the Aeolos 10.

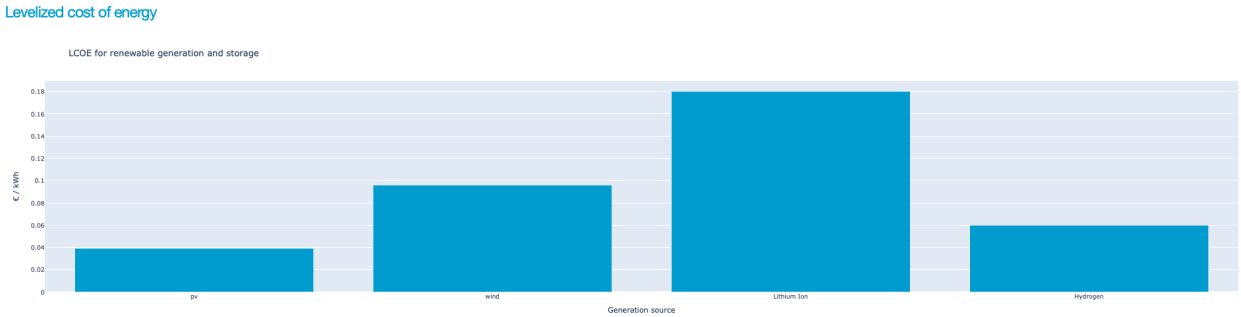

Figure 7.3: LCOE output in the dashboard for Pv, wind , EV and hydrogen battery.

The costs regarding storage can be found in table A.3. The Nissan leaf+ lifespan, power and capacity where taken from an ev-database [1]. For the Nissan leaf+ 56kWh lithium ion battery capital costs, the car producer charges around €12,000.00 which includes 3 hours of labour cost.

The expected cycles for the Nissan where obtained by dividing the 100,000km warranty by the 325km radius, which means 308 cycles. The hydrogen tank lifespan, cycles and capacity parameters were taken from the producer Mahytec [2]. The other values where not specified so the values from the National Renewable Laboratory of the U.S. Department of Energy where taken [9]. The capital cost of the fuel cell is the sum of the electrolyzer and the fuel tank. The projected OM costs for both storage technologies where taken to be 50€/kW.

For an input of 200 monocrystalline panels at a 30° tilt, a LCOE of 0.067 €/kWh is obtained. For two small scale Aeolos 10kW wind turbines, a LCOE of 0.104 €/kWh is calculated. For 1 hydrogen tank and 10 EV's a LCOE of respectivly 0.18 €/kWh and 0.06 €/kWh is obtained. These results are shown in figure 7.3. For the renewable generation source LCOE's, an increase in size corresponds to a decrease in LCOE. One of the reasons for this is the cost benefits a larger scale generation plant benefits from.

7.2. Payback Time

The projected payback time is a useful financial performance indicator for deciding whether to implement a solar module or wind turbine. The formula to calculate the payback time is shown in equation 7.2. The projected payback time is a projection for the period when an investment breaks even and the asset has essentially bought itself back. The decision was made to define the inverter replacement cost as initial cost although, the present value after a 10 year use was taken. The cash flow is the sum of the money saved by the renewable energy sources and the profit made in case electricity is sold back to the national grid. Also, the total investment costs are shown, this tells the client not only when the initial investment is earned back, but also how much he should invest to obtain the financial benefits discussed in section 7.3. The Illuminator works with a flat rate buy price of €0.22 per KWh and a flat rate selling price of €0.07 per KWh. In 2019, Dutch Minister Wiebes announced plans to cancel the "salderings regeling", which is a policy that allowed private generated electricity to be sold back to the grid without extra taxes. This resulted in a method where customers only pay for the electricity they have used minus the electricity they have supplied to the grid. The reason for the implementing the "salderings regeling" was to give people an incentive to buy PV panels. The Illuminator will not implement this policy change, pretending that the "salderings regeling" will remain the same. The reason for this is the uncertainty of the actual policy implementation timeline, having been delayed before.

$$\textbf{Payback Time} = \frac{\text{Initial Investment}}{\text{Cashflow}} \tag{7.2}$$

7.3. Financial Report 2019

The final financial performance visualisation, is a cost savings table for the year 2019 shown in figure 7.4. A table that shows the monthly savings in 2019 helps the user understand both the seasonal fluctuations in renewable generation and the financial impacts. Also, by showing the difference in electricity cost, the user is confronted with the financial benefits from implementing the chosen system. The analysis is presented in a way that makes it easy to see the financial system performance vary on a monthly basis and at the same time

the total estimated cost savings in 2019. The financial report over 2019 shows the users output for each month and total 2019 in: the average daily PV generation in kWh/day, the average wind generation in kWh/day, the daily average load in kWh/day, the average energy saved in kWh/day and the daily average energy cost with using renewables and without using renewables.

payback time [years]

The payback time is expected to be 11.71 years with a total investment cost of €235396.01

Month	PV output (KWh) / day	Wind output (KWh) / day	Grid output (KWh) / day	Electricity Saved (KWh) / day	Electricity bill old (€)	Electricity bill new (€)
January	54.5KWh	205.4KWh	177.7KWh	259.9KWh	2564.39€	1211.99€
February	121.7KWh	146.5KWh	127.6KWh	268.2KWh	2240.11€	870.49€
March	166.6KWh	236.5KWh	-2.9KWh	403.1KWh	2237.72€	-19.57€
April	259.0KWh	132.9KWh	4.9KWh	392.0KWh	2124.33€	33.35€
May	281.7KWh	102.9KWh	31.3KWh	384.6KWh	2305.70€	213.70€
June	296.8KWh	106.4KWh	3.1KWh	403.2KWh	2132.32€	20.90€
July	294.4KWh	79.5KWh	8.8KWh	373.9KWh	2083.54€	59.92€
August	275.4KWh	125.1KWh	-24.9KWh	400.5KWh	2143.81€	-169.72€
September	212.5KWh	147.3KWh	53.8KWh	359.8KWh	2179.18€	366.89€
October	123.3KWh	181.8KWh	128.4KWh	305.1KWh	2529.88€	875.98€
November	57.5KWh	138.8KWh	234.1KWh	196.3KWh	2424.84€	1596.63€
December	43.1KWh	221.1KWh	180.6KWh	264.1KWh	2576.04€	1231.98€
Total 2019	2186.6KWh	1824.5KWh	922.8KWh	4011.1KWh	27544.26€	6293.81€

Figure 7.4: Financial report 2019 table

$$E_{saved} = E_{pv} + E_{wind} \tag{7.3}$$

$$C_{old} = E_{load}.P_b \tag{7.4}$$

$$C_{new} = E_{+grid}.P_b + E_{-grid}.P_s \tag{7.5}$$

The values in the "Electricity Saved" column are found with equation 7.3, with E_{pv} used for the daily average PV energy generation in a month. The E_{wind} is used for the daily average generated wind energy in the same month. In equations 7.4 and 7.5 the method for calculating the electricity costs is shown. The electricity price for the neighbourhood is given by C_{old}, while C_{new} is used to indicate the cost with the use of the renewable generation assets in the neighbourhood. The P_b and P_s are used to indicate the fixed buy and selling rate when using the grid. For both the payback time and the 2019 savings analysis a fixed buy and sell rate is used. From the data table, the client in this specific case learns that the implemented renewable generation sources lead to a electricity bill of 6,293.81€ instead of 27,544.26€ which is a reduction of 21,250.45€. In combination with the initial investment cost, the client should have a good indication if the chosen renewable inputs are economically feasible.

8

Implementation

The implementation of the dashboard is discussed in this chapter. The two main software components needed for the dashboard are MQTT discussed in 8.1 and Dash, discussed in 8.2. The MQTT refers to the protocol used for communication between the raspberry pi's. Dash is the python framework used for building the web application shown on the Illuminator dashboard. All python code for the dashboard is given in appendix C.

8.1. Communication and MQTT

For the communication between the clients and the server, the MQTT protocol will be used. The MQTT (Message Queuing Telemetry Transport) protocol has two network entities, a server and a number of clients. The server is also called an MQTT broker, to which all messages will be sent from the clients. This broker ensures that all clients, that are subscribed to a certain "topic", receive their messages. Any already existing broker on the internet could have been used for exchanging these messages. However, creating a broker on the server Pi has the advantage that no internet communication is required, as well as the lower latency. Here, the MQTT will be using the TCP [8] protocol for the communication to and from the broker. The python code used for the MQTT protocol is shown in appendix C.5.

8.2. Dash

The Illuminator dashboard has been developed using Dash. Dash is a productive Python framework for building web applications. Dash is ideal for building data visualisation apps with highly custom user interfaces in pure Python. Dash applications are rendered in the web browser.Since Dash apps are viewed in the web browser, Dash is inherently cross-platform and mobile ready. Dash is an open source library, released under the permissive MIT license.

Illuminator dashboard

The python code used for the dashboard inputs is given in appendix C.1, which is basically where the user changes the neighbourhood parameters which invokes the callbacks to calculate the new data. For the dashboard to visualise the incoming energy widget power flows and storage SoC's, the python code in appendix C.2 was used. The graphs are refreshed when an input change in the dashboard has occurred. The power flows are graphed using plotly, which is an interactive, open-source plotting library for python. Different to the power flow data, the emissions data is calculated by the dashboard. The code used to obtain this data is depicted in appendix C.3. Same as the emissions, the financial data is calculated by the dashboard. The neighbourhood financial analysis code is given in appendix C.4. The financial performance indicator dashboard outputs are recalculated when an arbitrary power flow change is noticed. This is done by checking if the new incoming power flow value is the same as the old one. This leads to the finance output only changing when a parameters changes that influences the finance indicator.

9

Case Study Results

In this chapter, the results of the Illuminator dashboard will be discussed. The possibilities for using the Illuminator are vast and users won't necessarily agree on what the most important result for a specific neighbourhood is. For example, one user might want to have the lowest possible carbon emissions for his neighbourhood. Another user might prefer the cheapest energy cost or most profit. To discuss the results of the dashboard, the case study described below is used.

9.1. The Case Study

The case study is advising a residential project developer, who is going to build a neighbourhood for 1000 households in Delft, The Netherlands. The developer in the case-study has a budget of €4,000,000 euros to invest in renewable energy sources in the neighbourhood. His reason for this is a curiosity in renewable energy sources and also the cost savings. However, like always in business, they do not want to pay extra to implement renewables over the already present grey energy from the national grid, which is still very heavily influenced by fossil energy sources.

Inputs for case study

The developer has several preferences regarding the neighbourhood inputs in terms of aesthetics and size. These preferences, in combination with several assumptions are shown in the list below.

- The neighbourhood will consist of 1000 houses, that can be considered as energy saving homes.
- For every household an electric vehicle can serve as a battery. This is symbolic for a Nissan Leaf+. The developer expects (300-500) houses to have interest in using a Nissan Leaf+. The reason for this is that this neighbourhood focuses on sustainability, so the expected residents are aware of the importance in sustainability. The client starts with 500 EV's, but is prepared to lower this to 300 if financially more attractive.
- The developer chooses to start with 10 Hummer 60kW turbines,for financial and aesthetic reasons. The Aeolos 10 has a low output so many need to be used to obtain a significant output. The Vestas V90 isn't considered aesthetically appealing by the client so the Hummer 60 is chosen.
- The developer chooses 1 or 2 PV panels per roof but is still uncertain about which type to use or at what angle to build the roofs at which they lie. The polycrystalline PV modules are generally cheaper, so the client starts with 1500 panels. The initial panel tilt is set at a 30° because the client doesn't know the ideal tilt yet.
- For a long term storage option, hydrogen fuel cells can be added to the grid. There is only one option for this and that is the medium sized hydrogen tanks which can be used in parallel to provide a larger capacity. The client also believes some residents might use hydrogen power cars in the future, so the initial number of hydrogen tanks is set at 10.

The resulting dashboard inputs are shown in B.1.

Outputs of the case study

The initial inputs of the case study result in the dashboard output shown in Appendix B.1, the resulting power flows are depicted in figure B.2 and the resulting storage output is depicted in B.3. Furthermore, by analysing the resulting power flows various projected emission data and financial indicators are calculated. The results of the emissions analysis are shown in B.5 and the results of the financial analysis are shown in B.6. Also, the corresponding electricity mix is is depicted in B.4 The resulting dashboard data will be analysed in the next 3 sections.

9.2. Power Flow Results

The resulting power flows in B.2 tell the client several things regarding the generation performance compared to the load consumption. First thing that is noticed here is the seasonal fluctuations in power flows, which are quiet sever in this location. The PV power output is lowest in times where consumption is highest. If the client had been interested in a stand-alone grid, he could immediately see that he should either significantly increase PV generation or use more of the steady wind power generation. This is due to the peaks around 900kW consumption in January with a maximum combined PV and wind power of 600kW. However in this case, the client isn't interested in a islanded-grid, meaning that the chosen inputs perform sufficient. There is one parameter left to decide on which is the PV panel tilt. The resulting PV output for the three choices are shown in figures B.7 to B.9, from these values it is derived an angle of 35° gives the highest yearly PV generation output. Because of this, the client chooses a panel tilt of 35°.

The corresponding grid power flow and storage SoC dashboard results are presented in B.3. From these outputs, the client can analyse the performance of the storage modules and if the initial storage size is correct. Also, the grid power flow indicates to what extend the client is taking from the grid or supplying the grid with the neighbourhoods excess electricity. Due to the dashboard storage outputs the client knows that the initial storage options are not too large. The way the client deduces this from the SoC, is by observing that both storage options are never fully charged. To test if the storage options are too large, the client uses the minimum 300 Nissan Leafs. The storage output when using only 300 EV's, is depicted in figure B.10. The new dashboard storage outputs indicate a more suitable Nissan Leaf+ fleet. The neighbourhood storage system SoC's are higher but still not fully charged to often. in sections 9.3 and 9.4, more reasons for using 300 Leafs are given.

9.3. Emissions Results

In figure B.11, the neighbourhood emissions for the new inputs are depicted. By analysing these new emission performance outputs, the client learns that the new inputs resulted in a total emission reduction of 273.3 tonne CO2. This value is obtained by comparing the old emission output of 570.068 tonnes in B.5 by the new 296.761 tonnes CO2 equivalent output from B.11.

Furthermore the client easily sees that the hydrogen batteries contribute very little to the emissions, so the client could increase the amount of hydrogen batteries to reduce the use of fossil sources and increase the amount of renewable sources. The amount of PV panels can still be increased more without reaching above the amount of carbon emissions of fossil sources.

9.4. Financial Results

The new neighbourhood parameter inputs have significant financial benefits, as can be seen in figure B.12. By changing the the panel tilt and number of EV's, the client reduces his yearly cost with an estimated €108,099.32. This value is obtained by substituting the old electricity cost output of €172,980.00, by the new €64,880.68. It should be noted that this is mainly due to the Nissan Leafs also being used as loads that require energy to drive to work and back. This could be charged directly to the residents. However, the payback period reduces from 12.53 years to 7.69 years, primarily because the total initial investment cost is reduced from €3.626.149,72 to €3,014,749.72. The client learns that a reduction in the EV's leads to a more realistic payback model.

10

Conclusions and Future Work

Conclusions

In the demonstrator a lot of choices are given to optimise for the users ideal neighbourhood. From the dashboard the user can easily see what to change for their purposes.

The choices for choosing mono and poly crystalline PV panels in our demonstrator is because these are the most produced panels in the industry. Mono-Si has better efficiency but is more expensive and the poly-Si is the other way around, less efficient but also less expensive. This gives our client choices to choose from to optimise for their needs. For the wind generation the problem is more aesthetic. Large wind turbines in a neighbourhood is not very visually pleasing. Smaller turbines are easy to hide from plain view, however, they are have larger life cycle emissions and the LCOE is lower than the larger turbines. Furthermore choices are given for different loads in the dashboard. For households the choice is between average households and energy saving households. The project developer can estimate what kind of resident will live in the neighbourhood and make an appropriate choice. Options for different storages are also given. The options are using EV batteries from electric cars as quick storage and hydrogen batteries as long term storage. Moreover the grid is also a load in the Illuminator, because excess energy can be put into the grid which will reduce the electricity costs.

The life cycle emission analysis gives more dept to the demonstrator. It shows the sustainability of the neighbourhood. It does this by comparing all the life cycle emissions of the renewable and fossil energy sources. Furthermore it also depicts the methane and nitrous oxide values from each source to show the importance of methane and nitrous oxide emissions. The total CO2eq. emissions are also given for the chosen time frame to have a better understanding of how much emissions are emitted. To better understand the large number provided from the total emission analysis, an example is given by comparing the total emission to the amount of trees that have to live for 100 years to absorb everything.

An financial analysis is also provided to give insight to the financial part of the neighbourhood. This is done by calculating Levelized Cost of Energy(LCOE) and comparing those, by calculating the payback time of the renewable energies and providing a full financial report of a given time frame, in this report the time frame is the whole year of 2019.

Finally the study case is fully explained to show how the demonstrator can be used to design a neighbourhood without building anything at all. This saves a lot of time for the client by providing everything in one system. All the information is easily depicted by graphs in the dashboard.

Future Work

In future work we would like to look closer at the irradiance for PV systems. Most literature focuses on a irradiance of 1700kWh/m2/yr, but the irradiance of The Netherlands is on average 1000kWh/m2/yr. This would give emission values closer to reality. Furthermore the time frame used in this report is the year 2019. Due to limited time this was not implemented. The dashboard and energy widget can quickly give a different time frame but unfortunately this was not tested due to the hardware hub group not being able to make a physical module because of the COVID-19 quarantine. For all renewable sources more options can be provided, however due to the time constraints we were not able to add more. However this is also quickly

implemented since all the literature needed is already included in the bibliography. This can be combined with implementing more graphs to easily compare two different graphs of the same energy source.

A

Cost Breakdown

Model	Timespan (years)	Cost per panel ex-btw	Yearly decay
Panasonic HIT N245 W Mono	25	€ 164,46	0,36%
JAP60S01-290/SC Poly	25	€ 89,45	0,64%

BOS cost (balance of system)	General	Residential	Commerical
Inverter Enphase IQ7+			
Inverter cost per 10 panels	€ 505,74		
Lifespan (years)	15		
Other BOS per Watt DC		€ 0,30	€ 0,25

Installation costs per Watt DC	Residential	Commercial
labour cost	10,00%	10,00%
Permitting, inspection and interconnection (PII)	€ 0,05	€ 0,09
Overhead (general & administrative)	€ 0,60	
Total EPC and other overhead		13,00%
Sales & marketing (customer acquisition)	€ 0,70	
Profit % on installation costs	17,00%	7%

O&M cost per KW per year	Residential	Commercial
Module cleaning & vegetation management	€ 0,72	€ 2,44
System inpsection & monitoring	€ 4,50	€ 4,97
Component parts replacement	€ 4,12	€ 0,84
Module replacement	€ 0,74	€ 0,74
Operations administration	€ 2,35	€ 2,33
TOTAL euro/kw/year	€ 12,43	€ 11,32

Table A.1: PV cost values

Model	Timespan (years)	Output [kW]	Total initial	O&M cost per year
Vestas V90	20	20000	€ 2,456,000.00	€ 49120.00
Hummer 60	20	60	€ 106,860.00	€ 2137.20
Aeolos10	20	10	€ 23,313.40	€ 466.27

Table A.2: Wind turbine cost parameters

Type	Lifespan	Cycles	Power	Capacity	Capital cost	O&M Cost
Nissan leaf+	5 years	308	160kW	56kWh	€ 45840.00	€ 600 / year
hydrogen tank	10 years	5000	98kW	396kWh	€ 78048	€ 4800/ year

Table A.3: Storage cost data

B

Case study

B.1. Initial input results

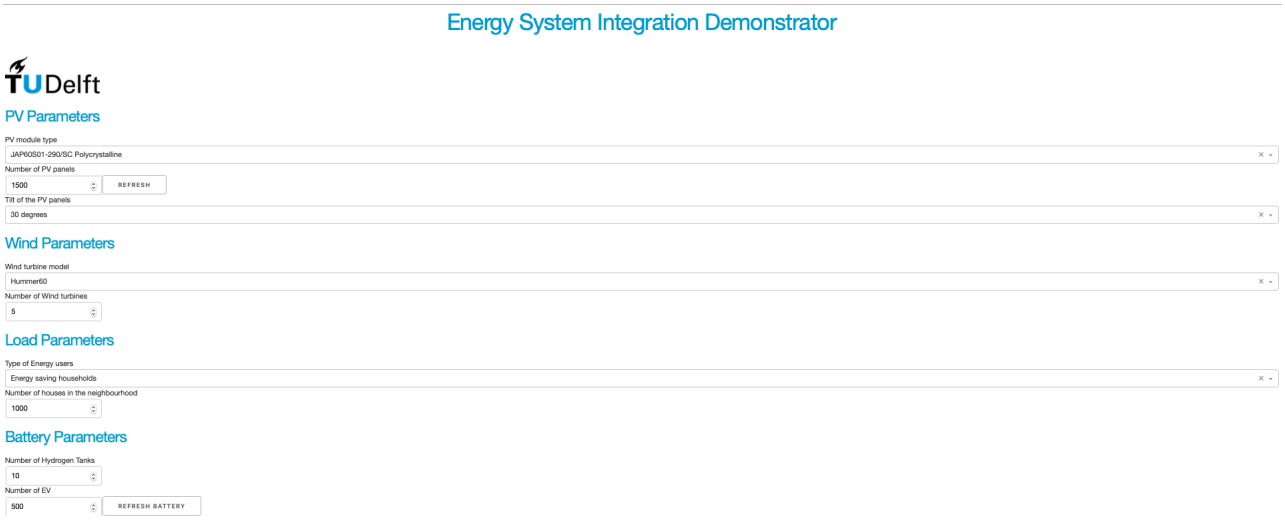

Figure B.1: The starting inputs of the cas

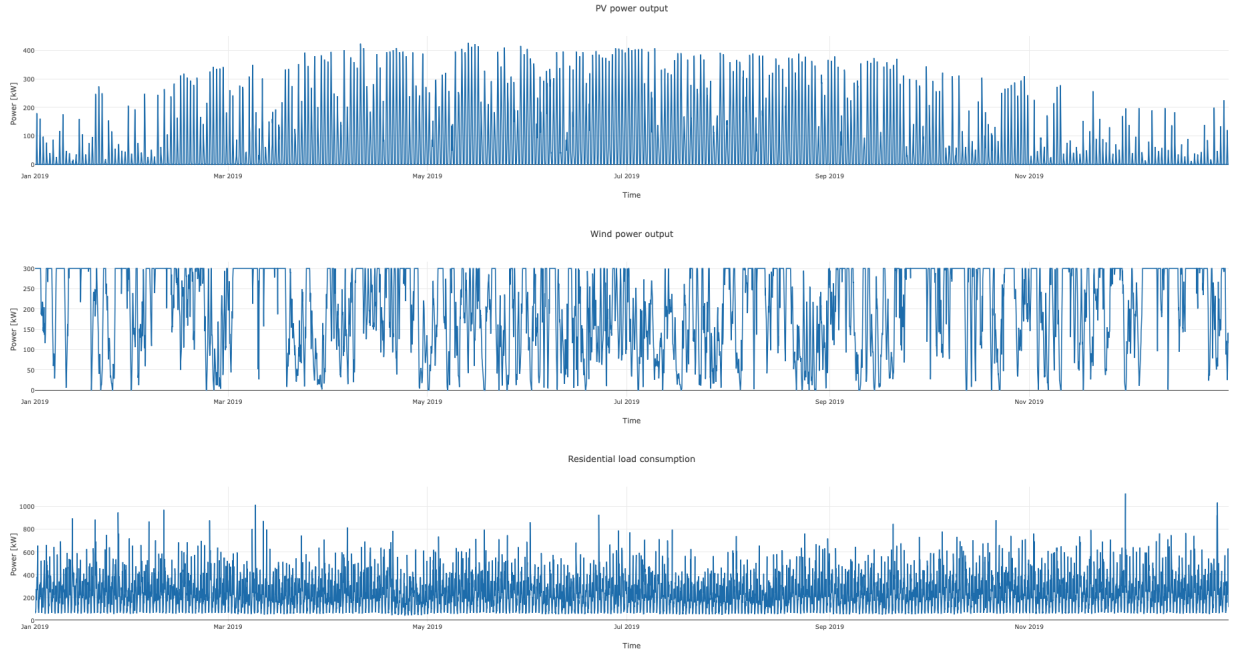

Figure B.2

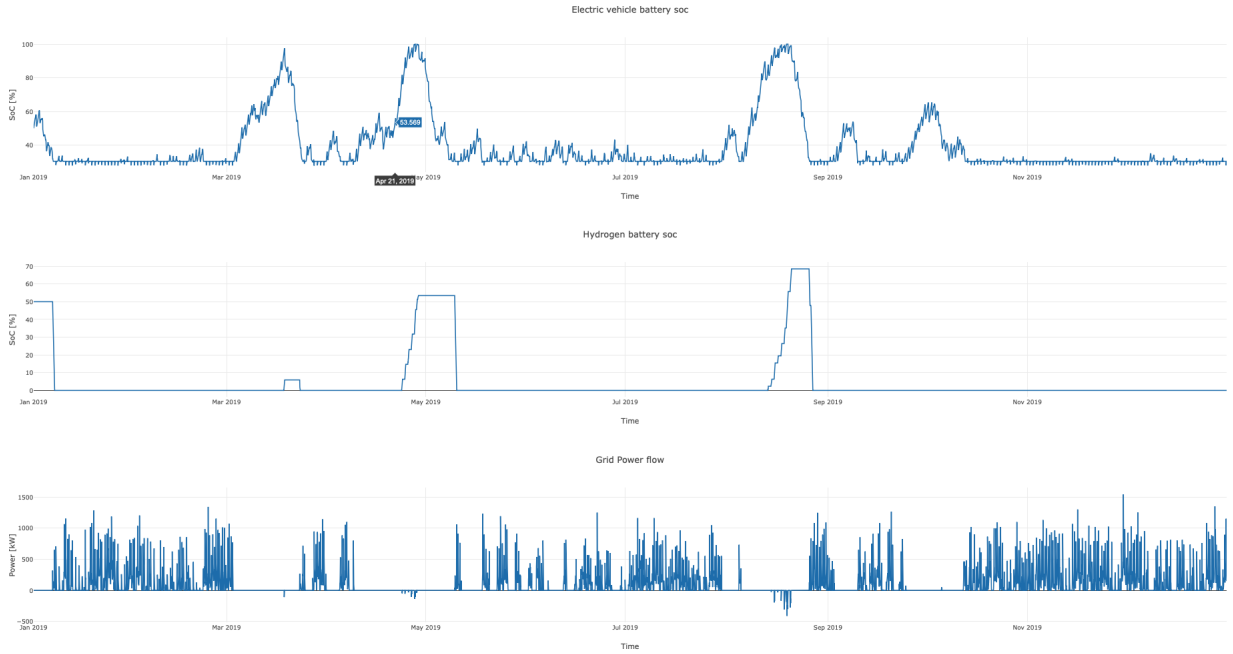

Figure B.3



Figure B.4

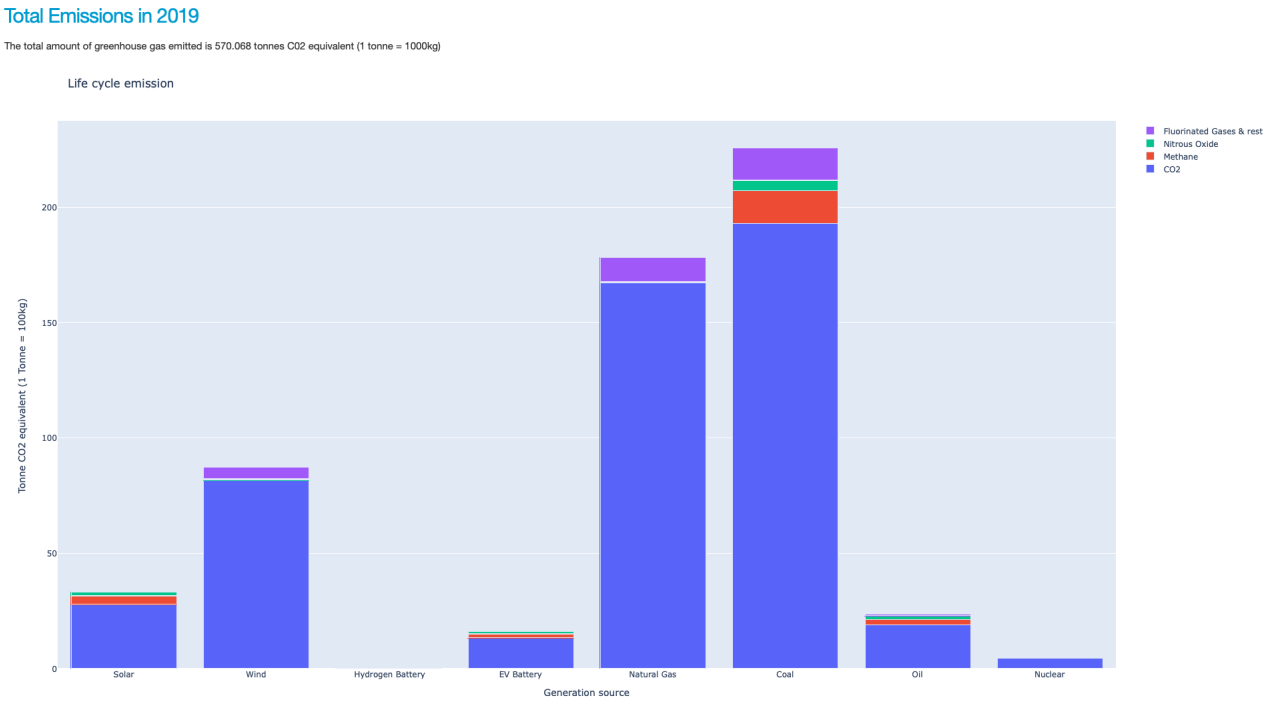

Figure B.5

payback time [years]

The payback time is expected to be 12.53 years with a total investment cost of €3626149.72

Month	PV output (KWh) / day	Wind output (KWh) / day	Grid output (KWh) / day	Electricity Saved (KWh) / day	Electricity bill old (€)	Electricity bill new (€)
January	491.6KWh	5514.7KWh	2967.0KWh	6006.2KWh	41441.48€	20234.97€
February	1092.8KWh	4673.9KWh	2669.4KWh	5766.8KWh	36801.07€	18205.44€
March	1495.0KWh	5212.0KWh	1185.6KWh	6707.0KWh	37729.70€	8085.67€
April	2316.5KWh	4911.5KWh	522.0KWh	7228.1KWh	36241.10€	3560.30€
May	2516.8KWh	4234.4KWh	1376.1KWh	6751.2KWh	40464.85€	9384.77€
June	2636.3KWh	4004.2KWh	1560.4KWh	6640.5KWh	37347.96€	10641.78€
July	2611.7KWh	3450.2KWh	3099.8KWh	6061.9KWh	37260.15€	21140.34€
August	2441.4KWh	4488.7KWh	863.1KWh	6930.2KWh	37928.04€	5886.05€
September	1890.6KWh	4810.4KWh	1502.3KWh	6701.0KWh	36446.46€	10245.43€
October	1101.8KWh	5459.2KWh	2208.2KWh	6561.0KWh	41692.54€	15059.85€
November	516.7KWh	4786.3KWh	4101.4KWh	5303.0KWh	39817.31€	27971.52€
December	387.5KWh	5792.5KWh	3318.7KWh	6180.0KWh	41110.06€	22633.57€
Total 2019	19498.7KWh	57354.2KWh	25374.9KWh	76853.0KWh	464312.98€	173057.11€

Figure B.6

Month	PV output (KWh) / day
January	491.6KWh
February	1092.8KWh
March	1495.0KWh
April	2316.5KWh
May	2516.8KWh
June	2636.3KWh
July	2611.7KWh
August	2441.4KWh
September	1890.6KWh
October	1101.8KWh
November	516.7KWh
December	387.5KWh
Total 2019	19498.7KWh

Figure B.7: 30° panel tilt PV output

Month	PV output (KWh) / day
January	511.4KWh
February	1138.7KWh
March	1520.6KWh
April	2328.7KWh
May	2491.5KWh
June	2597.6KWh
July	2576.7KWh
August	2438.8KWh
September	1920.4KWh
October	1134.8KWh
November	534.5KWh
December	404.0KWh
Total 2019	19597.6KWh

Figure B.8: 35° panel tilt PV output

Month	PV output (KWh) / day
January	528.8KWh
February	1177.7KWh
March	1538.0KWh
April	2328.0KWh
May	2453.4KWh
June	2545.6KWh
July	2528.7KWh
August	2424.3KWh
September	1939.7KWh
October	1161.6KWh
November	549.7KWh
December	418.3KWh
Total 2019	19593.9KWh

Figure B.9: 40° panel tilt PV output

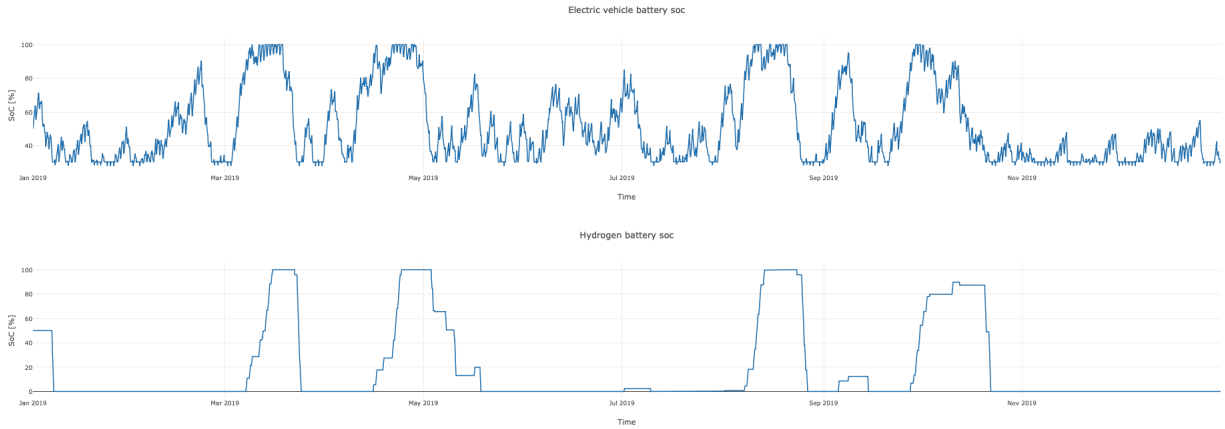

Figure B.10: Storage when using 300 EV's

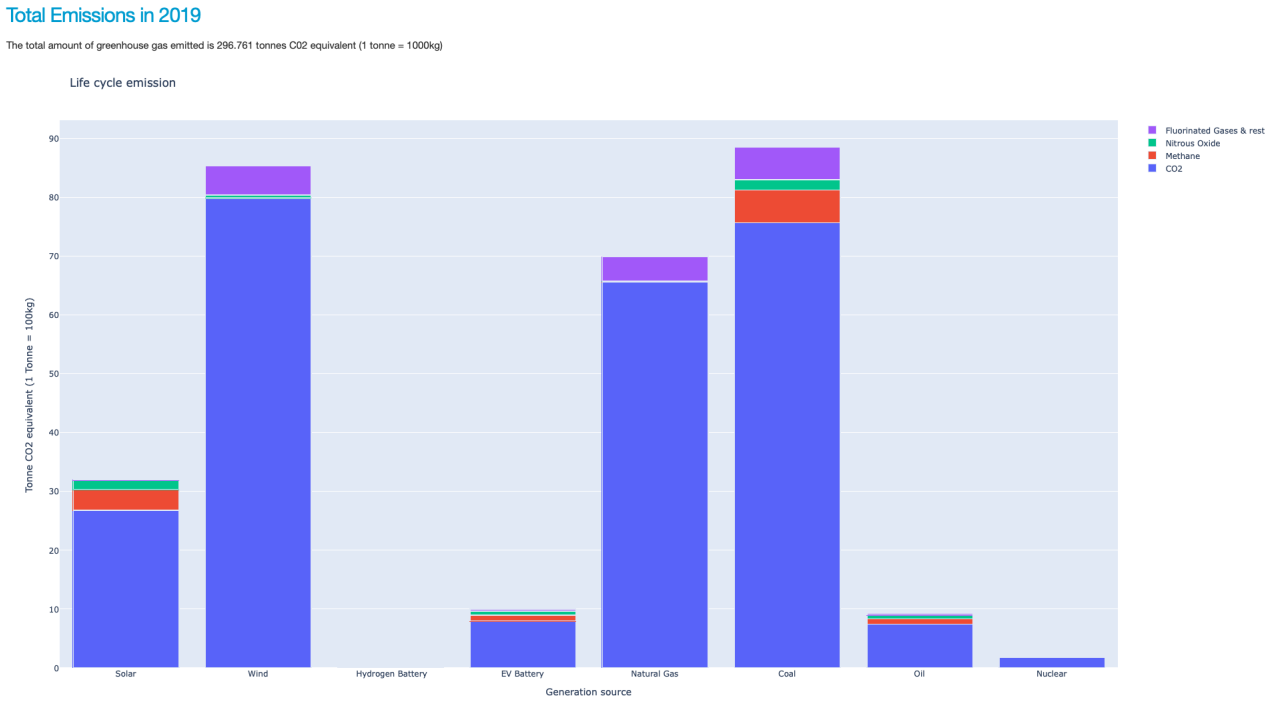

Figure B.11: The dashboard emission output for the new inputs

payback time [years]

The payback time is expected to be 7.69 years with a total investment cost of €3014749.72

Month	PV output (KWh) / day	Wind output (KWh) / day	Grid output (KWh) / day	Electricity Saved (KWh) / day	Electricity bill old (€)	Electricity bill new (€)
January	511.4KWh	5514.7KWh	1432.4KWh	6026.1KWh	41441.48€	9769.10€
February	1138.7KWh	4673.9KWh	821.4KWh	5812.6KWh	36801.07€	5601.42€
March	1520.6KWh	5212.0KWh	359.0KWh	6732.6KWh	37729.70€	4556.58€
April	2328.7KWh	4911.5KWh	-507.9KWh	7240.2KWh	36241.10€	-86.08€
May	2491.5KWh	4234.4KWh	444.5KWh	6725.9KWh	40464.85€	3031.16€
June	2597.6KWh	4004.2KWh	136.2KWh	6601.8KWh	37347.96€	928.40€
July	2576.7KWh	3450.2KWh	959.7KWh	6027.0KWh	37260.15€	6545.22€
August	2438.8KWh	4488.7KWh	-27.9KWh	6927.6KWh	37928.04€	3487.14€
September	1920.4KWh	4810.4KWh	333.7KWh	6730.8KWh	36446.46€	2740.12€
October	1134.8KWh	5459.2KWh	497.7KWh	6594.0KWh	41692.54€	3624.56€
November	534.5KWh	4786.3KWh	2201.1KWh	5320.9KWh	39817.31€	15011.70€
December	404.0KWh	5792.5KWh	1418.0KWh	6196.5KWh	41110.06€	9670.32€
Total 2019	19597.6KWh	57354.2KWh	8067.8KWh	76951.9KWh	464312.98€	64880.68€

Figure B.12: The dashboard financial performance indicators output for the new inputs

C

Python Code

C.1. Dashboard input code

Listing C.1: Dasboard python code

```
#------------------dash layout---------------------------------------------------------------

app.layout = html.Div(children=[
    html.H1(
        children='Energy System Integration Demonstrator',
        style={
            'textAlign': 'center',
            'color': colors['text']
        }
    ),
    html.Img(src='{}'.format(test_base64)),
    html.H3('PV Parameters',
            style={'color': colors['text']}),
    html.Div('PV module type'),
    dcc.Dropdown(
        id='dropdownpvtype',
        options=[
            {'label': 'HIT-N245SE10 Monocrystalline', 'value': 'HIT-N245SE10'},
            {'label': 'JAP60S01-290/SC Polycrystalline', 'value': 'JAP60S01-290/SC'},
        ],
        value='HIT-N245SE10'
    ),
    html.Div('Number of PV panels'),
    dcc.Input(id='input', value=200, type='number'),
    html.Button('Refresh', id='button', n_clicks=0),
    html.Div('Tilt of the PV panels'),
    dcc.Dropdown(
        id='dropdown',
        options=[
            {'label': '30 degrees', 'value': 30},
            {'label': '35 degrees', 'value': 35},
            {'label': '40 degrees', 'value': 40}
        ],
        value=30
    ),
    html.H3('Wind Parameters',
            style={'color': colors['text']}),
    html.Div('Wind turbine model'),
    dcc.Dropdown(
        id='dropdownwind',
        options=[
            {'label': 'Aeolos10', 'value': 'Aeolos10'},
            {'label': 'Hummer60', 'value': 'Hummer60'},
            {'label': 'Vestas V90 2MW', 'value': 'V90-2MW'},
        ],
        value='Aeolos10'
```

```
    ),
    html.Div('Number␣of␣Wind␣turbines'),
    dcc.Input(id='inputwind', value=2, type='number'),
    html.H3('Load␣Parameters',
            style={'color': colors['text']}),
    html.Div('Type␣of␣Energy␣users'),
    dcc.Dropdown(
        id='dropdownhousehold',
        options=[
            {'label': 'Energy␣saving␣households', 'value': 'saving'},
            {'label': 'Average␣households', 'value': 'average'},
        ],
        value='average'
    ),
    html.Div('Number␣of␣houses␣in␣the␣neighbourhood'),
    dcc.Input(id='input␣load', value=40, type='number'),
    html.H3('Battery␣Parameters',
            style={'color': colors['text']}),
    html.Div('Number␣of␣Hydrogen␣Tanks'),
    dcc.Input(id='input␣H', value=1, type='number'),
    html.Div('Number␣of␣EV'),
    dcc.Input(id='input␣EV', value=10, type='number'),
    html.Button('Refresh␣battery', id='button_bat', n_clicks=0),



    html.Div(children = [
    dcc.Graph(id='pvpower', ),
    dcc.Graph(id='windpower',),
    dcc.Graph(id='loadpower', ),
    dcc.Graph(id='EVpower', ),
    dcc.Graph(id='Hydrogen', ),
    dcc.Graph(id='gridpower', ),
    html.H3('Electricty␣mix␣in␣2019',
            style={'color': colors['text']}),
    dcc.Graph(id='piechart', ),
    #dcc.Graph(id='piechartshare', animate=False,)
    html.H3('Total␣Emissions␣in␣2019',
            style={'color': colors['text']}),
    html.Div(id='emtot'),
    dcc.Graph(id='emission',),
    html.H3('Levelized␣cost␣of␣energy',
            style={'color': colors['text']}),
    dcc.Graph(id='lcoe',),
    html.H3('payback␣time␣[years]',
            style={'color': colors['text']}),
    html.Div(id='pay'),
    html.Div(id='table')
    ],id='output-all'),

    dcc.Interval(
        id='interval-component',
        interval=4*1000,#in miliseconds
        n_intervals=0
    ),
    # empty dummy div
    html.Div(id='hidden-div', style={'display':'none'})

])
```

C.2. Dashboard Illuminator output code

Listing C.2: Dasboard python code

```
#------------------pv figure-------------------------------------------------------------------
PV_cache = 0

@app.callback(Output('pvpower', 'figure'),
```

```
            [Input('interval-component', 'n_intervals')])
def update_graph_solar(n):
    global PV_cache
    if PV_cache != dh['power_solar']:
        PV_cache = dh['power_solar']

        figure = {
            'data': [
                {'x': dh['time'], 'y': dh['power_solar'], 'type': 'line', 'name': 'PV'}
            ],'marker' : { 'color' :'#00A6D6' },
            'layout': {
                'title': 'PV power output',
                'xaxis': {
                    'title': 'Time'
                },
                'yaxis': {
                    'title': 'Power [kW]'
                }
            }
        }
    else:
        raise PreventUpdate
    return figure

#------------------Wind figure-----------------------------------------------------------------------
wind_cache = 0

@app.callback(Output('windpower', 'figure'),
              [Input('interval-component', 'n_intervals')])
def update_graph_wind(n):
    global wind_cache
    if wind_cache != dh['power_wind']:
        wind_cache = dh['power_wind']
        figure = {
            'data': [
                {'x': dh['time'], 'y': dh['power_wind'], 'type': 'line', 'name': 'Wind'}
            ],
            'layout': {
                'title': 'Wind power output',
                'xaxis': {
                    'title': 'Time'
                },
                'yaxis': {
                    'title': 'Power [kW]'
                }
            }
        }
    else:
        raise PreventUpdate
    return figure


#------------------load figure-----------------------------------------------------------------------
load_cache = 0

@app.callback(Output('loadpower', 'figure'),
              [Input('interval-component', 'n_intervals')])
def update_graph_live_load(n):
    global load_cache
    if load_cache != dh['power_load']:
        load_cache = dh['power_load']
        figure = {
            'data': [
                {'x': dh['time'], 'y': dh['power_load'], 'type': 'line', 'name': 'load',}
            ],'marker' : { 'color' :'#00A6D6' },
            'layout': {
                'title': 'Residential load consumption',
                'xaxis': {
                    'title': 'Time'
                },
                'yaxis': {
```

```
                    'title': 'Power [kW]'
                }
            }
        }
    else:
        raise PreventUpdate
    return figure

#-------------------Battery-----------------------------------------------------------------------
EV_cache = 0

@app.callback(Output('EVpower', 'figure'),
              [Input('interval-component', 'n_intervals')])
def update_graph_ev(n):
    global EV_cache
    if EV_cache != dh['EV_SoC']:
        EV_cache = dh['EV_SoC']
        figure = {
            'data': [
                {'x': dh['time'], 'y': dh['EV_SoC'], 'type': 'line', 'name': 'Battery'}
            ],'marker' : { 'color' :'#00A6D6' },
            'layout': {
                'title': 'Electric vehicle battery soc',
                'xaxis': {
                    'title': 'Time'
                },
                'yaxis': {
                    'title': 'SoC [%]'
                }
            }
        }
    else:
        raise PreventUpdate
    return figure


#-------------------Battery-----------------------------------------------------------------------
hydro_cache = 0

@app.callback(Output('Hydrogen', 'figure'),
              [Input('interval-component', 'n_intervals')])
def update_graph_hydrogen(n):
    global hydro_cache
    if hydro_cache != dh['H_SoC']:
        hydro_cache = dh['H_SoC']
        figure = {
            'data': [
                {'x': dh['time'], 'y': dh['H_SoC'], 'type': 'line', 'name': 'Hydrotank'}
            ],'marker' : { 'color' :'#00A6D6' },
            'layout': {
                'title': 'Hydrogen battery soc',
                'xaxis': {
                    'title': 'Time'
                },
                'yaxis': {
                    'title': 'SoC [%]'
                }
            }
        }
    else:
        raise PreventUpdate
    return figure


#-------------------grid--------------------------------------------------------------------------
grid_cache = 0

@app.callback(Output('gridpower', 'figure'),
              [Input('interval-component', 'n_intervals')])
def update_graph_live_load(n):
    global grid_cache
```

```
    if grid_cache != dh['power_grid']:
        grid_cache = dh['power_grid']
        figure = {
            'data': [
                {'x': dh['time'], 'y': dh['power_grid'], 'type': 'line', 'name': 'grid'}
            ],
            'layout': {
                'title': 'Grid Power flow',
                'xaxis': {
                    'title': 'Time'
                },
                'yaxis': {
                    'title': 'Power [kW]'
                }
            }
        }
    else:
        raise PreventUpdate
    return figure
```

C.3. Dashboard emission code

Listing C.3: Dasboard python code

```
#------------------pie chart-----------------------------------------------------------------
pie_cache = 0

@app.callback(Output('piechart', 'figure'),
              [Input('interval-component', 'n_intervals')])
def update_graph_live_pie(n):

    global pie_cache

    tot_net = sumPositiveInts(dh['power_grid'])
    tot_pv = sum(dh['power_solar']) + tot_net * 0.05
    tot_wind = sum(dh['power_wind']) + tot_net* 0.08
    tot_gas = tot_net * 0.45
    tot_coal = tot_net * 0.32
    tot_oil = tot_net * 0.04
    tot_nuclear = tot_net * 0.03
    tot_other = tot_net * 0.03
    green = tot_pv + tot_wind
    grey = tot_gas + tot_oil + tot_coal
    other = tot_nuclear + tot_other
    share = [tot_pv, tot_wind, tot_gas, tot_coal, tot_oil, tot_nuclear, tot_other]
    share_2 = [green, grey, other]

    fig = make_subplots(rows=1, cols=2, specs=[[{"type": "pie"}, {"type": "pie"}]])

    fig.add_trace(go.Pie(
        values=share,
        labels=labels,
        domain=dict(x=[0, 0.5]),
        name="Share of Energy source"),
        row=1, col=1)

    fig.add_trace(go.Pie(
        values=share_2,
        labels=["Green", "Grey", "Unknown",
                ],
        domain=dict(x=[0.5, 1.0]),
        name="Green vs grey"),
        row=1, col=2)
    global pie_cache
    if pie_cache != fig:
        pie_cache = fig
    else:
        raise PreventUpdate
```

```
    return fig


#-------------------emissions figure-------------------------------------------------------------
em_cache = 0

@app.callback(Output('emission', 'figure'),
              [Input('interval-component', 'n_intervals')],state=[State('dropdownpvtype', 'value'),
                                                                   State('input', 'value'),
                                                                   State('dropdownwind', 'value'),
                                                                   State('inputwind', 'value'),
                                                                   State('input␣H', 'value'),
                                                                   State('input␣EV', 'value'),
                                                                  ],)
def update_graph_live_emission(n,type,number,typewind,numberwind,H,EV):
    global em_cache

    WIND_EMISSIONS = {
        'Aeolos10': {'P_rated': 10000, 'GHG': 0.000047,},
        'Hummer60': {'P_rated': 60000, 'GHG': 0.000047,},
        'Vestas␣V90␣2MW': {'P_rated': 2000000, 'GHG': 0.0000093,}
    }
    PV_EMISSIONS = {
        'Mono': {'watt_peak': 245, 'GHG': 0.00004702,},
        'Poly': {'watt_peak': 295, 'GHG': 0.00005155,}
    }
    if type == 'HIT-N245SE10':
        dpv = PV_EMISSIONS['Mono']
    else:
        dpv = PV_EMISSIONS['Poly']
    if typewind == 'Aeolos10':
        dwe= WIND_EMISSIONS['Aeolos10']
    elif typewind == 'V90-2MW':
        dwe = WIND_EMISSIONS['Vestas␣V90␣2MW']
    elif typewind == 'Hummer60':
        dwe = WIND_EMISSIONS['Hummer60']

    ghgwind=dwe['GHG']
    ghgpv = dpv['GHG']
    tot_net = sumPositiveInts(dh['power_grid'])
    tot_pv = sum(dh['power_solar']) + tot_net * 0.05
    pv_carbon = round(tot_pv*ghgpv,4)
    tot_wind = sum(dh['power_wind']) + tot_net* 0.08
    wind_carbon = round(tot_wind * ghgwind,4)
    tot_h=H *0.007485
    tot_ev=EV*0.0327
    tot_gas = (tot_net * 0.45)*0.000499
    tot_coal = (tot_net * 0.32)*0.000888
    tot_oil = (tot_net * 0.04)*0.000733
    tot_nuclear = (tot_net * 0.03)*0.00029
    tot_other = tot_net * 0.03
    sources = ['Solar', 'Wind', '␣Hydrogen␣Battery', 'EV␣Battery','Natural␣Gas', 'Coal', 'Oil', 'Nuclear']

    fig = go.Figure(data=[
        go.Bar(name='CO2', x=sources, y=[pv_carbon*0.836, wind_carbon*0.941, tot_h*0.81, tot_ev*0.81, tot_gas*0.9375, tot_co
        go.Bar(name='Methane', x=sources, y=[pv_carbon*0.112, wind_carbon*0.002, tot_h*0.1, tot_ev*0.1, tot_gas*0.0023, tot_
        go.Bar(name='Nitrous␣Oxide', x=sources, y=[pv_carbon * 0.0495, wind_carbon * 0.005, tot_h * 0.07, tot_ev * 0.07, tot
        go.Bar(name='Fluorinated␣Gases␣&␣rest', x=sources, y=[pv_carbon * 0.0025, wind_carbon * 0.0583, tot_h * 0.03, tot_ev
    ])
    fig.update_layout(
        title="Life␣cycle␣emission␣",
        xaxis_title="Generation␣source",
        yaxis_title="Tonne␣CO2␣equivalent␣(1␣Tonne␣=␣100kg)",
        height=1000, barmode='stack'
    )
    global em_cache
    if em_cache != fig:
        em_cache = fig
    else:
        raise PreventUpdate
```

```
    return fig


emt_cache = 0

@app.callback(Output('emtot', 'children'),
              [Input('interval-component', 'n_intervals')],state=[State('dropdownpvtype', 'value'),
                                                                  State('input', 'value'),
                                                                  State('dropdownwind', 'value'),
                                                                  State('inputwind', 'value'),
                                                                  State('input_H', 'value'),
                                                                  State('input_EV', 'value'),
                                                                 ],)
def update_graph_live_emission(n,type,number,typewind,numberwind,H,EV):
    global emt_cache

    WIND_EMISSIONS = {
        'Aeolos10': {'P_rated': 10000, 'GHG': 0.000047,},
        'Hummer60': {'P_rated': 60000, 'GHG': 0.000047,},
        'Vestas_V90_2MW': {'P_rated': 2000000, 'GHG': 0.0000093,}
    }
    PV_EMISSIONS = {
        'Mono': {'watt_peak': 245, 'GHG': 0.00004702,},
        'Poly': {'watt_peak': 295, 'GHG': 0.00005155,}
    }
    if type == 'HIT-N245SE10':
        dpv = PV_EMISSIONS['Mono']
    else:
        dpv = PV_EMISSIONS['Poly']
    if typewind == 'Aeolos10':
        dwe= WIND_EMISSIONS['Aeolos10']
    elif typewind == 'V90-2MW':
        dwe = WIND_EMISSIONS['Vestas_V90_2MW']
    elif typewind == 'Hummer60':
        dwe = WIND_EMISSIONS['Hummer60']

    ghgwind=dwe['GHG']
    ghgpv = dpv['GHG']
    tot_net = sumPositiveInts(dh['power_grid'])
    tot_pv = sum(dh['power_solar']) + tot_net * 0.05
    pv_carbon = round(tot_pv*ghgpv,4)
    tot_wind = sum(dh['power_wind']) + tot_net* 0.08
    wind_carbon = round(tot_wind * ghgwind,4)
    tot_h=H *0.007485
    tot_ev=EV*0.0327
    tot_gas = (tot_net * 0.45)*0.000499
    tot_coal = (tot_net * 0.32)*0.000888
    tot_oil = (tot_net * 0.04)*0.000733
    tot_nuclear = (tot_net * 0.03)*0.00029
    tot_other = tot_net * 0.03
    tot_em=round(pv_carbon+wind_carbon +tot_h +tot_ev +tot_gas+tot_coal+tot_oil+tot_nuclear,3)


    global emt_cache
    if emt_cache != tot_em:
        emt_cache = tot_em
    else:
        raise PreventUpdate
    return 'The total amount of greenhouse gas emitted is {} tonnes C02 equivalent (1 tonne = 1000kg)'.format(tot_em)
```

C.4. Dashboard financial code

Listing C.4: Dasboard python code

```
#--------------------lcoe-------------------------------------------------------------------
lcoe_cache = 0

@app.callback(Output('lcoe', 'figure'),
              [Input('interval-component', 'n_intervals',)],state=[State('dropdownpvtype', 'value'),
```

```
                                                    State('input', 'value'),
                                                    State('dropdownwind', 'value'),
                                                    State('inputwind', 'value'),
                                                    State('input␣H', 'value'),
                                                    State('input␣EV', 'value'),],)
def update_graph_live_lcoe(n,type,number,typewind,numberwind,H,EV):
    global lcoe_cache
    WIND_FINANCE = {
        'Aeolos10': {'P_rated': 10000, 'investment_cost': 23313.40, 'OM␣per␣year': 466.27},
        'Hummer60': {'P_rated': 60000, 'investment_cost': 106860.00, 'OM␣per␣year': 2137.20},
        'Vestas␣V90␣2MW': {'P_rated': 2000000, 'investment_cost': 2456000, 'OM␣per␣year': 49120}
    }
    PV_FINANCE = {
        'Mono␣residential': {'watt_peak': 245, 'price': 164.46, 'yearly_decay': 0.0939, 'bos': 1.65, 'OM': 12.43},
        'Mono␣commercial': {'watt_peak': 245, 'price': 164.46, 'yearly_decay': 0.0939, 'bos': 0.35, 'OM': 11.32},
        'Poly␣residential': {'watt_peak': 295, 'price': 89.45, 'yearly_decay': 0.0964, 'bos': 1.65, 'OM': 12.43},
        'Poly␣commercial': {'watt_peak': 295, 'price': 89.45, 'yearly_decay': 0.0964, 'bos': 0.35, 'OM': 11.32}
    }
    discountRate = 0.09  # Nine percent per annum
    total_solar_power = dh['power_solar']
    if type == 'HIT−N245SE10':
        output_pv = number* 245
        if output_pv < 10000:
            dpv = PV_FINANCE['Mono␣residential']
        else:
            dpv = PV_FINANCE['Mono␣commercial']
    elif type == 'JAP60S01−290/SC':
        output_pv = number * 295
        if output_pv < 10000:
            dpv = PV_FINANCE['Poly␣residential']
        else:
            dpv = PV_FINANCE['Poly␣commercial']
    inverter_cost = roundup(number) * 0.1 * 505.74
    Initialcost_pv = (number * dpv['price'] + (inverter_cost + dpv['watt_peak'] * dpv['bos'])/0.9)
    cashflows = [dpv['OM']] * 14
    cashflows.extend([dpv['OM'] + inverter_cost])
    cashflows.extend(
        [dpv['OM'], dpv['OM'], dpv['OM'], dpv['OM'], dpv['OM'], dpv['OM'], dpv['OM'], dpv['OM'], dpv['OM'],
         dpv['OM']])
    yearlyyield = [sum(dh['power_solar'])]*25
    netpresent = npf.npv(discountRate, cashflows)
    totalyield = npf.npv(dpv['yearly_decay'], yearlyyield)
    lcoe_pv = ((Initialcost_pv + netpresent) / totalyield).round(3)

    if typewind == 'Aeolos10':
        dw= WIND_FINANCE['Aeolos10']
    elif typewind == 'V90−2MW':
        dw = WIND_FINANCE['Vestas␣V90␣2MW']
    elif typewind == 'Hummer60':
        dw = WIND_FINANCE['Hummer60']
    turbine_cost= numberwind * dw['investment_cost']
    discountrate_decay_wind = 0.112
    cashflowwind = [dw['OM␣per␣year']] * 20
    yearlyyieldwind = [sum(dh['power_wind'])]*20
    netpresentwind = npf.npv(discountRate, cashflowwind)
    totalyieldwind = npf.npv(discountrate_decay_wind, yearlyyieldwind)
    lcoe_w = ((turbine_cost + netpresentwind) / totalyieldwind).round(3)
    bat_cost = EV * (57 + 3000)
    bat_cap = EV*56*308
    bat_cost_h = H * (78048 + 48000)
    bat_cap_h = H * 396 * 5000
    lcoe_sev = round(((bat_cost) / bat_cap),2)
    lcoe_seh = round(((bat_cost_h) / bat_cap_h), 2)
    fig = go.Figure(data=[
        go.Bar(name='LCOE', x=['pv', 'wind', 'Lithium␣Ion', 'Hydrogen'], y=[lcoe_pv, lcoe_w, lcoe_sev, lcoe_seh], marker_col
    ])
    fig.update_layout(
        title="LCOE␣for␣renewable␣generation␣and␣storage",
        height=600,
        xaxis_title="Generation␣source",
        yaxis_title="euro␣/␣kWh",
```

```
    )

    # fig = make_subplots(rows=1, cols=2, specs=[[{"type": "bar"}, {"type": "bar"}]])
    # fig.add_trace(go.Bar(
    #     y=[lcoe_pv,lcoe_w],
    #     x=['pv','wind',],
    #     domain=dict(x=[0, 0.5]),
    #     name="LCOE energy"),
    #     row=1, col=1)
    #
    # fig.add_trace(go.Bar(
    #     y=[lcoe_s],
    #     x=['storage'],
    #     domain=dict(x=[0.5, 1.0]),
    #     name="LCOE storage"),
    #     row=1, col=2)
    # fig.update_layout(
    #     title="Levelized cost of energy",
    #     xaxis_title="Generation source",
    #     yaxis_title="euro/kWh",
    # )

    global lcoe_cache
    if lcoe_cache != fig:
        lcoe_cache = fig
    else:
        raise PreventUpdate
    return fig


#------------------paybacktime-----------------------------------------------------------------------
pay_cache = 0

@app.callback(Output('pay', 'children'),
              [Input('interval-component', 'n_intervals',)],state=[State('dropdownpvtype', 'value'),
                                                                   State('input', 'value'),
                                                                   State('dropdownwind', 'value'),
                                                                   State('inputwind', 'value'),
                                                                   State('input␣H', 'value'),
                                                                   State('input␣EV', 'value'),
                                                                   ],)
def update_graph_live_pay(n,type,number,typewind,numberwind,H,EV):
    global pay_cache
    WIND_FINANCE = {
        'Aeolos10': {'P_rated': 10000, 'investment_cost': 23313.40, 'OM␣per␣year': 466.27},
        'Hummer60': {'P_rated': 60000, 'investment_cost': 106860.00, 'OM␣per␣year': 2137.20},
        'Vestas␣V90␣2MW': {'P_rated': 2000000, 'investment_cost': 2456000, 'OM␣per␣year': 49120}
    }
    PV_FINANCE = {
        'Mono␣residential': {'watt_peak': 245, 'price': 164.46, 'yearly_decay': 0.0939, 'bos': 1.65, 'OM': 12.43},
        'Mono␣commercial': {'watt_peak': 245, 'price': 164.46, 'yearly_decay': 0.0939, 'bos': 0.35, 'OM': 11.32},
        'Poly␣residential': {'watt_peak': 295, 'price': 89.45, 'yearly_decay': 0.0964, 'bos': 1.65, 'OM': 12.43},
        'Poly␣commercial': {'watt_peak': 295, 'price': 89.45, 'yearly_decay': 0.0964, 'bos': 0.35, 'OM': 11.32}
    }
    discountRate = 0.09  # Nine percent per annum
    if type == 'HIT-N245SE10':
        output_pv = number* 245
        if output_pv < 10000:
            dpv = PV_FINANCE['Mono␣residential']
        else:
            dpv = PV_FINANCE['Mono␣commercial']
    elif type == 'JAP60S01-290/SC':
        output_pv = number * 295
        if output_pv < 10000:
            dpv = PV_FINANCE['Poly␣residential']
        else:
            dpv = PV_FINANCE['Poly␣commercial']
    if typewind == 'Aeolos10':
        dw= WIND_FINANCE['Aeolos10']
    elif typewind == 'V90-2MW':
        dw = WIND_FINANCE['Vestas␣V90␣2MW']
```

```
    elif typewind == 'Hummer60':
        dw = WIND_FINANCE['Hummer60']
    inverter_cost = roundup(number) * 0.2 * 505.74
    Initialcost_pv = (number * dpv['price'] + (inverter_cost + dpv['watt_peak'] * dpv['bos'])/0.9)
    Initialcost_wind= (numberwind * dw['investment_cost'])
    bat_cost_h = H * (78048 + 48000)
    bat_cost = EV * (57 + 3000)
    total_investment= Initialcost_wind + Initialcost_pv +bat_cost_h + bat_cost
    pos_net = sumPositiveInts(dh['power_grid'])
    neg_net = sumNegativeInts(dh['power_grid'])
    yearly_savings = (sum(dh['power_load'])*0.22 - pos_net*0.22 + neg_net*0.06)
    paytime= total_investment/(0.1+yearly_savings)
    if paytime > 0:
        paytimes=round(paytime,2)
    else:
        paytimes= '(not reached)'
    total_investments=round(total_investment,2)
    global pay_cache
    if pay_cache != paytime:
        pay_cache = paytime
    else:
        raise PreventUpdate
    return 'The payback time is expected to be {} years with a total investment cost of euro{} '.format(paytimes,total_invest
#------------------Table-------------------------------------------------------
table_cache = 0

@app.callback(Output('table', 'children'),
              [Input('interval-component', 'n_intervals')])
def update_graph_live_table(n):
    global wind_cache
    pv_o = dh['power_solar']
    w_o = dh['power_wind']
    g_o = dh['power_grid']
    l_o = dh['power_load']

    df_typing_formatting = pd.DataFrame(OrderedDict([
        ('Month', ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'Novemb
        ('PV', [sum(pv_o[:744]) / 31, sum(pv_o[745:1416]) / 31, sum(pv_o[1417:2161]) / 31, sum(pv_o[2161:2881]) / 31,
                sum(pv_o[2881:3625]) / 31, sum(pv_o[3625:4345]) / 31, sum(pv_o[4345:5089]) / 31, sum(pv_o[5089:5833]) / 31,
                sum(pv_o[5833:6553]) / 31, sum(pv_o[6553:7297]) / 31, sum(pv_o[7297:8017]) / 31, sum(pv_o[8017:8761]) / 31,
                sum(pv_o) / 31]),
        ('Wind', [sum(w_o[:744]) / 31, sum(w_o[745:1416]) / 31, sum(w_o[1417:2161]) / 31, sum(w_o[2161:2881]) / 31,
                sum(w_o[2881:3625]) / 31, sum(w_o[3625:4345]) / 31, sum(w_o[4345:5089]) / 31, sum(w_o[5089:5833]) / 31,
                sum(w_o[5833:6553]) / 31, sum(w_o[6553:7297]) / 31, sum(w_o[7297:8017]) / 31, sum(w_o[8017:8761]) / 31,
                sum(w_o) / 31]),
        ('Grid', [sum(g_o[:744]) / 31, sum(g_o[745:1416]) / 31, sum(g_o[1417:2161]) / 31, sum(g_o[2161:2881]) / 31,
                sum(g_o[2881:3625]) / 31, sum(g_o[3625:4345]) / 31, sum(g_o[4345:5089]) / 31, sum(g_o[5089:5833]) / 31,
                sum(g_o[5833:6553]) / 31, sum(g_o[6553:7297]) / 31, sum(g_o[7297:8017]) / 31, sum(g_o[8017:8761]) / 31,
                sum(g_o) / 31]),
        ('Electricity Saved', [(sum(w_o[:744]) + sum(pv_o[:744])) / 31, (sum(w_o[745:1416]) + sum(pv_o[745:1416])) / 31,
                               (sum(w_o[1417:2161]) + sum(pv_o[1417:2161])) / 31, (sum(w_o[2161:2881]) + sum(pv_o[2161:2881
                               (sum(w_o[2881:3625]) + sum(pv_o[2881:3625])) / 31, (sum(w_o[3625:4345]) + sum(pv_o[3625:4345
                               (sum(w_o[4345:5089]) + sum(pv_o[4345:5089])) / 31, (sum(w_o[5089:5833]) + sum(pv_o[5089:5833
                               (sum(w_o[5833:6553]) + sum(pv_o[5833:6553])) / 31, (sum(w_o[6553:7297]) + sum(pv_o[6553:7297
                               (sum(w_o[7297:8017]) + sum(pv_o[7297:8017])) / 31, (sum(w_o[8017:8761]) + sum(pv_o[8017:8761
                               (sum(w_o) + sum(pv_o)) / 31]),
        ('Without', [sum(l_o[:744])*0.22, sum(l_o[745:1416])*0.22, sum(l_o[1417:2161])*0.22, sum(l_o[2161:2881])*0.22,
                sum(l_o[2881:3625])*0.22, sum(l_o[3625:4345])*0.22, sum(l_o[4345:5089])*0.22, sum(l_o[5089:5833])*0.22,
                sum(l_o[5833:6553])*0.22, sum(l_o[6553:7297])*0.22, sum(l_o[7297:8017])*0.22, sum(l_o[8017:8761])*0.22,
                sum(l_o)*0.22]),
        ('With',
         [(sum(g_o[:744]) * 0.22), sum(g_o[745:1416]) * 0.22, sum(g_o[1417:2161]) * 0.22, sum(g_o[2161:2881]) * 0.22,
          sum(g_o[2881:3625]) * 0.22, sum(g_o[3625:4345]) * 0.22, sum(g_o[4345:5089]) * 0.22,
          sum(g_o[5089:5833]) * 0.22,
          sum(g_o[5833:6553]) * 0.22, sum(g_o[6553:7297]) * 0.22, sum(g_o[7297:8017]) * 0.22,
          sum(g_o[8017:8761]) * 0.22,
          sum(g_o) * 0.22]),
    ]))
    table = dash_table.DataTable(
        id='typing_formatting_1',
        data=df_typing_formatting.to_dict('rows'),
```

```
columns=[{
    'id': 'Month',
    'name': 'Month',
    'type': 'text'
},

    {
    'id': 'PV',
    'name': 'PV output (KWh) / day',
    'type': 'numeric',
    'format': Format(
        precision=1,
        scheme=Scheme.fixed,
        symbol=Symbol.yes,
        symbol_suffix=u'KWh')
}, {
    'id': 'Wind',
    'name': 'Wind output (KWh) / day',
    'type': 'numeric',
    'format': Format(
        precision=1,
        scheme=Scheme.fixed,
        symbol=Symbol.yes,
        symbol_suffix=u'KWh')
}, {
    'id': 'Grid',
    'name': 'Grid output (KWh) / day',
    'type': 'numeric',
    'format': Format(
        precision=1,
        scheme=Scheme.fixed,
        symbol=Symbol.yes,
        symbol_suffix=u'KWh')

}, {
    'id': 'Electricity Saved',
    'name': 'Electricity Saved (KWh) / day',
    'type': 'numeric',
    'format': Format(
        precision=1,
        scheme=Scheme.fixed,
        symbol=Symbol.yes,
        symbol_suffix=u'KWh')

}, {
    'id': 'Without',
    'name': 'Electricity bill old (euro)',
    'type': 'numeric',
    'format': Format(
        precision=2,
        scheme=Scheme.fixed,
        symbol=Symbol.yes,
        symbol_suffix=u'euro'
    ),
}, {
    'id': 'With',
    'name': 'Electricity bill new (euro)',
    'type': 'numeric',
    'format': Format(
        precision=2,
        scheme=Scheme.fixed,
        symbol=Symbol.yes,
        symbol_suffix=u'euro'
    )
},
],style_header={'backgroundColor': '#00A6D6'},
style_cell={
    'font_family': 'normal',
    'font_size': '26px',
    'text_align': 'center'
},
```

```
        editable=True
    )



    global table_cache
    if table_cache !=table:
        table_cache = table
    else:
        raise PreventUpdate
    return table
```

C.5. Dasboard MQTT python code

Listing C.5: Dasboard python code

```
#-----------------------MQTT-----------------------------------------------------------------
wind = 0
load = 0
PV = 0
def connect_and_run_dash(client, number_bat):

    @app.callback(
        Output('hidden-div','load'),
        [Input('button', 'n_clicks')],
        state=[State('input␣load', 'value'), State('dropdownhousehold', 'value')],
    )
    def update_output_load(n_clicks,loadvalue, loadtype):
        global load,start
        if load != [loadvalue,loadtype]:
            load = [loadvalue,loadtype]
            data = {'N_load': loadvalue}
            data.update({'load_type': loadtype})
            client.publish("load", json.dumps(data))
            print( 'load␣request␣at'  +str(time.time()-start))
            start = time.time()
        return 0

    # @app.callback([
    #     Output('pvpower', 'animate'),
    #     Output('windpower', 'animate'),
    #     Output('loadpower', 'animate'),
    #     Output('EVpower', 'animate'),
    #     Output('Hydrogen', 'animate'),
    #     Output('gridpower', 'animate'),
    # ],
    #     [Input('interval-component', 'n_intervals')])
    # def update_graph(n):
    #     print(n)
    #     if n % 20 == 10:
    #         animate = False
    #     else:
    #         animate = True
    #     animate = [animate] * 6  # 8 because 8 outputs are required
    #     return animate

    @app.callback(
        Output('hidden-div','PV'),
        [Input('button', 'n_clicks')],
        state=[State('dropdownpvtype', 'value'), State('input', 'value'), State('dropdown', 'value'),],
    )
    def update_output_solar(n_clicks, paneltype, panelvalue, tiltvalue):
        global PV,start
        if PV != [paneltype, panelvalue, tiltvalue]:
            PV = [paneltype, panelvalue, tiltvalue]
            data = ({'pv_type': paneltype})
            data.update({'N_solar': panelvalue})
            data.update({'tilt_panel': tiltvalue})
```

```
            client.publish("solar", json.dumps(data))
            print('solar request at' +str(time.time()-start))
            start = time.time()
        return 0


@app.callback(
    Output('hidden-div', 'Turbine'),
    [Input('button', 'n_clicks')],
    [State('dropdownwind', 'value'), State('inputwind', 'value')],  # add amount of windmills
)
def update_output_wind(n_clicks, turbinetype, windvalue):
    global wind,start
    if wind != [turbinetype,windvalue]:
        wind = [turbinetype,windvalue]
        data = ({'turbine_type': turbinetype})
        data.update({'N_wind': windvalue})
        client.publish("wind", json.dumps(data))
        print('wind request at' +str(time.time()-start))
        start = time.time()
    return 0


@app.callback(
   Output('hidden-div','battery'),
   [Input('button_bat', 'n_clicks')],
   [State('input EV', 'value'), State('input H', 'value')],
)
def update_output_bat(n_clicks,evvalue, hydrogen):
   global start
   print('bat request at' +str(time.time()-start))
   start = time.time()
   #hydrogen = 1 or hydrogen = 0
   number_bat.setValue(evvalue, hydrogen)
   return 0
```

Bibliography

[1] Nissan leaf e+. URL https://ev-database.nl/auto/1144/Nissan-Leaf-eplus.

[2] Mahytec datasheet. URL http://www.mahytec.com/wp-content/uploads/2019/07/CL-DS7-Data-Sheet_500bar-200L-300L.pdf.

[3] The International Renewable Energy Agency. Renewable capacity statistics 2020, 2020. URL https://www.irena.org/-/media/Files/IRENA/Agency/Publication/2020/Mar/IRENA_RE_Capacity_Statistics_2020.pdf.

[4] Anders Arvesen and Edgar G. Hertwich. Assessing the life cycle environmental impacts of wind power: A review of present knowledge and research needs. *Renewable and Sustainable Energy Reviews*, 16(8): 5994 – 6006, 2012. ISSN 1364-0321. doi: https://doi.org/10.1016/j.rser.2012.06.023. URL http://www.sciencedirect.com/science/article/pii/S1364032112004169.

[5] Ovidiu Mihai Balan, Mihaela-Ramona Buga, Aymeric Brunot, Adrian Badea, and Daniel Froelich. Technical and economic evaluation of power-to-gas in link with a 50mw wind park. *Journal of Energy Storage*, 8:111 – 118, 2016. ISSN 2352-152X. doi: https://doi.org/10.1016/j.est.2016.10.002. URL http://www.sciencedirect.com/science/article/pii/S2352152X16301840.

[6] Bloomberg NEF (BNEF). Clean energy investment trends 2019, 2020.

[7] Alexandra Bonou, Alexis Laurent, and Stig I. Olsen. Life cycle assessment of onshore and offshore wind energy-from theory to application. *Applied Energy*, 180:327–337, oct 2016. ISSN 03062619. doi: 10.1016/j.apenergy.2016.07.058.

[8] Vinton Cerf, Yogen Dalal, and Carl Sunshine. Specification of internet transmission control program, 1974. URL https://tools.ietf.org/html/rfc675.

[9] title = lifecyle Cost Analysis of Hydrogen Versus Other Technologies for Electrical Energy Storage year = 2009 D. Steward, G. Saur, M. Penev, and T. Ramsden, institution = National Renewable Energy Laboratory. Technical report, USA.

[10] CE Delft. Environmental profile of dutch power suppliers, 2004. URL https://www.cedelft.eu/en/publications/183/environmental-profile-of-dutch-power-suppliers.

[11] Roberto Dones, Thomas Heck, Mireille Faist Emmenegger, and Niels Jungbluth. Life cycle inventories for the nuclear and natural gas energy systems, and examples of uncertainty analysis. *International Journal of Life Cycle Assessment*, 10(1):10–23, 2005. ISSN 09483349. doi: 10.1065/lca2004.12.181.2. URL http://dx.doi.org/10.1065/lca2004.12.181.2.

[12] Roberto Dones, Christian Bauer, Rita Bolliger, Bastian Burger, Thomas Heck, Alexander Röder, Paul Scherrer Institut, Mireille Faist Emmenegger, Rolf Frischknecht, Niels Jungbluth, and Matthias Tuchschmid. Life Cycle Inventories of Energy Systems : Results for Current Systems in Switzerland and other UCTE Countries. Ecoinvent report No. 5. Paul Scherrer Institut Villigen, Swiss Centre for Life Cycle Inventories. *Ecoinvent report*, (5), 2007.

[13] Linda Ager-Wick Ellingsen, Bhawna Singh, and Anders Hammer Strømman. The size and range effect: lifecycle greenhouse gas emissions of electric vehicles. *Environmental Research Letters*, 11(5):054010, may 2016. doi: 10.1088/1748-9326/11/5/054010. URL https://doi.org/10.1088%2F1748-9326%2F11%2F5%2F054010.

[14] Erik Emilsson and Lisbeth Dahllöf. Lithium-Ion Vehicle Battery Production - Status 2019 on Energy Use, CO2 Emissions, Use of Metals, Products Environmental Footprint, and Recycling. Technical Report C444, IVL, 2019. URL www.ivl.se.

[15] Energie beheer Nederland. Energie in nederland 2019. Technical report, Centraal bureau voor statistiek (CBS), Netherlands, 2019.

[16] Vasilis M. Fthenakis, Hyung Chul Kim, and Erik Alsema. Emissions from Photovoltaic Life Cycles. *Environmental Science & Technology*, 42(6):2168–2174, mar 2008. ISSN 0013-936X. doi: 10.1021/es071763q. URL https://pubs.acs.org/doi/10.1021/es071763q.

[17] Yinyin Fu, Xin Liu, and Zengwei Yuan. Life-cycle assessment of multi-crystalline photovoltaic (pv) systems in china. *Journal of Cleaner Production*, 86:180 – 190, 2015. ISSN 0959-6526. doi: https://doi.org/10.1016/j.jclepro.2014.07.057. URL http://www.sciencedirect.com/science/article/pii/S0959652614007859.

[18] Fu, Ran and Feldman, David and Margolis, Robert. U.S. Solar Photovoltaic System Cost Benchmark: Q1 2018. Technical report, National Renewable Energy Laboratory, Lafayette, 2018.

[19] Peter Garrett and Klaus Rønde. Life cycle assessment of wind power: Comprehensive results from a state-of-the-art approach. *International Journal of Life Cycle Assessment*, 18(1):37–48, jan 2013. ISSN 09483349. doi: 10.1007/s11367-012-0445-4. URL http://link.springer.com/10.1007/s11367-012-0445-4.

[20] Saïcha Gerbinet, Sandra Belboom, and Angélique Léonard. Life cycle analysis (lca) of photovoltaic panels: A review. *Renewable and Sustainable Energy Reviews*, 38:747 – 753, 2014. ISSN 1364-0321. doi: https://doi.org/10.1016/j.rser.2014.07.043. URL http://www.sciencedirect.com/science/article/pii/S136403211400495X.

[21] Greenhouse Gas Protocol. Global Warming Potential Values. Technical Report 1995, 2015. URL www.ipcc.ch.

[22] Karl R Haapala and Preedanood Prempreeda. Comparative life cycle assessment of 2.0 MW wind turbines. *International Journal of Sustainable Manufacturing*, 3(2):170–185, 2014. ISSN 17427231. doi: 10.1504/IJSM.2014.062496.

[23] Dale Hall and Nic Lutsey. Effects of battery manufacturing on electric vehicle life-cycle greenhouse gas emissions. *Environmental Research Letters*, 2018. doi: 10.1088/1748-9326/11/5/054010. URL https://theicct.org/sites/default/files/publications/EV-life-cycle-GHG_ICCT-Briefing_09022018_vF.pdf.

[24] David D. Hsu, Patrick O'Donoughue, Vasilis Fthenakis, Garvin A. Heath, Hyung Chul Kim, Pamala Sawyer, Jun Ki Choi, and Damon E. Turney. Life Cycle Greenhouse Gas Emissions of Crystalline Silicon Photovoltaic Electricity Generation: Systematic Review and Harmonization. *Journal of Industrial Ecology*, 16(SUPPL.1):S122–S135, apr 2012. ISSN 10881980. doi: 10.1111/j.1530-9290.2011.00439.x. URL http://doi.wiley.com/10.1111/j.1530-9290.2011.00439.x.

[25] Iea. Global co2 emissions in 2019 – analysis, Feb 2020. URL https://www.iea.org/articles/global-co2-emissions-in-2019.

[26] M. Jentsch and M. Sterner. Energiewirtschaftliche und ökologische Bewertung eines Windgas-Angebotes. Technical report, Fraunhofer IWES, 2011.

[27] Angel Kirchev. Battery Management and Battery Diagnostics. *Electrochemical Energy Storage for Renewable Sources and Grid Balancing*, 138:411–435, 2015. doi: https://doi.org/10.1016/B978-0-444-62616-5.00020-6. URL https://www.sciencedirect.com/science/article/pii/B9780444626165000206.

[28] Fabio Magrassi, Elena Rocco, Stefano Barberis, Michela Gallo, and Adriana Del Borghi. Hybrid solar power system versus photovoltaic plant: A comparative analysis through a life cycle approach. *Renewable Energy*, 130:290 – 304, 2019. ISSN 0960-1481. doi: https://doi.org/10.1016/j.renene.2018.06.072. URL http://www.sciencedirect.com/science/article/pii/S0960148118307249.

[29] A. Meijer, M. A. J. Huijbregts, J. J. Schermer, and L. Reijnders. Life-cycle assessment of photovoltaic modules: Comparison of mc-si, ingap and ingap/mc-si solar modules. *Progress in Photovoltaics: Research and Applications*, 11(4):275–287, 2003. doi: 10.1002/pip.489. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/pip.489.

[30] Y. Moumouni and M. Gdeisat. Sizing an off-grid photovoltaic system. In *2018 Advances in Science and Engineering Technology International Conferences (ASET)*, pages 1–4, 2018.

[31] Daniel Nugent and Benjamin K. Sovacool. Assessing the lifecycle greenhouse gas emissions from solar pv and wind energy: A critical meta-survey. *Energy Policy*, 65:229 – 244, 2014. ISSN 0301-4215. doi: https://doi.org/10.1016/j.enpol.2013.10.048. URL http://www.sciencedirect.com/science/article/pii/S0301421513010719.

[32] Mihail Predescu. Economic evaluation of small wind turbines and hybrid systems for residential use. *Renew. Energy Environ. Sustain. 1 33 (2016)*, (1), 2016. ISSN 0013-936X. doi: 10.1051/rees/2016023.

[33] Hanne Lerche Raadal, Luc Gagnon, Ingunn Saur Modahl, and Ole Jørgen Hanssen. Life cycle greenhouse gas (GHG) emissions from the generation of wind and hydro power. *Renewable and Sustainable Energy Reviews*, 15(7):3417–3422, sep 2011. ISSN 13640321. doi: 10.1016/j.rser.2011.05.001. URL https://linkinghub.elsevier.com/retrieve/pii/S1364032111001924.

[34] N.H. Reich, E.A. Alsema, W.G.J.H.M. van Sark, W.C. Turkenburg, and W.C. Sinke. Greenhouse gas emissions associated with photovoltaic electricity from crystalline silicon modules under various energy supply options. *Progress in Photovoltaics: Research and Applications*, 19(5):603–613, aug 2011. ISSN 10627995. doi: 10.1002/pip.1066.

[35] International Renewable Energy Agency. *GLOBAL RENEWABLES OUTLOOK 2050 ENERGY TRANSFORMATION E D I T I O N : 2 0 2 0*. International Renewable Energy Agency, 2020. ISBN 978-92-9260-238-3. URL www.irena.org.

[36] Qingbin Song, Zhishi Wang, Jinhui Li, Huabo Duan, Danfeng Yu, and Gang Liu. Comparative life cycle ghg emissions from local electricity generation using heavy oil, natural gas, and msw incineration in macau. *Renewable and Sustainable Energy Reviews*, 81:2450 – 2459, 2018. ISSN 1364-0321. doi: https://doi.org/10.1016/j.rser.2017.06.051. URL http://www.sciencedirect.com/science/article/pii/S1364032117309991.

[37] Wagner Sousa de Oliveira, Antonio Fernandes, and Joaquim Gouveia. Economic metrics for wind energy projects. *International Journal of Energy and Environment (IJEE)*, 2:1013–1038, 01 2011.

[38] Melanie Sporer, Suzanne Dael, Sabine Gores, Hannah Förster, Christian Nissen, and Mihai Tomescu. Trends and projections in Europe 2019. Technical Report 15, European Environment Agency(EEA), Copenhagen, 2019.

[39] Michela Vellini, Marco Gambini, and Valentina Prattella. Environmental impacts of PV technology throughout the life cycle: Importance of the end-of-life management for Si-panels and CdTe-panels. *Energy*, 138:1099–1111, nov 2017. ISSN 03605442. doi: 10.1016/j.energy.2017.07.031. URL https://linkinghub.elsevier.com/retrieve/pii/S0360544217312069.

[40] Michael Whitaker, Garvin A. Heath, Patrick O'Donoughue, and Martin Vorum. Life cycle greenhouse gas emissions of coal-fired electricity generation. *Journal of Industrial Ecology*, 16(s1):S53–S72, 2012. doi: 10.1111/j.1530-9290.2012.00465.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1530-9290.2012.00465.x.

[41] IBIS World. Industry market research, reports, and statistics, 2020. URL https://www.ibisworld.com/global/industry-trends/biggest-industries-by-revenue/.

[42] World Nuclear Association (WNA). Comparison of Lifecycle Greenhouse Gas Emissions of Various Electricity Generation Sources. http://world-nuclear.org/uploadedFiles/org/reference/pdf/comparison_of_lifecycle.pdf. *World Nuclear Association*, page 12, 2011.

[43] Guangling Zhao and Allan Schrøder Pedersen. Life Cycle Assessment of Hydrogen Production and Consumption in an Isolated Territory. In *Procedia CIRP*, volume 69, pages 529–533, 2018. doi: 10.1016/j.procir.2017.11.100. URL www.sciencedirect.com.