**Diese Arbeit wurde vorgelegt am**
**Lehr- und Forschungsgebiet Theorie der hybriden Systeme**

# Simulation, planning and visualization of a photovoltaic system on residential houses

Masterarbeit
Informatik

April 2023


| | |
|---|---|
| Vorgelegt von<br>Presented by | Md Al Mamun<br>Matrikelnummer: 408146<br>md.al.mamun@rwth-aachen.de |
| Erstprüfer<br>First examiner | Prof. Dr. rer. nat. Erika Ábrahám<br>Lehr- und Forschungsgebiet: Theorie der hybriden Systeme<br>RWTH Aachen University |
| Zweitprüfer<br>Second examiner | Prof. Dr. rer. nat. Thomas Noll<br>Lehr- und Forschungsgebiet: Software Modellierung und Verifikation<br>RWTH Aachen University |
| Betreuer<br>Supervisor | Dr. rer. nat. Pascal Richter<br>Lehr- und Forschungsgebiet: Theorie der hybriden Systeme<br>RWTH Aachen University |

# Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. rer. nat. Pascal Richter, for his support and guidance throughout this thesis. I would also like to thank Prof. Dr. rer. nat. Erika Ábrahám and Prof. Dr. rer. nat. Thomas Noll for accepting and taking the time to read and correct this thesis. I am also grateful to my family, friends and colleagues, who have offered support and encouragement during the writing of this paper.

# Contents

# 1 Introduction

Climate change is an undeniable issue that our generation is currently facing, and its consequences are being constantly felt by plenty of peoples worldwide, whether in the form of floods, earthquakes, storms, wildfires, global warming, and other natural disasters. These drastic changes to our environment are primarily caused by the exponential increase in heavy industrialization behaviors over the last century, which has resulted in the production of excessive amounts of greenhouse gases like carbon dioxide. The primary sources of these gases are the burning of fossil fuels such as coal, oil, and natural gas, which serve as the driving force behind human industry. The problem that comes with the increase of concentration of greenhouse gases in the earth's atmosphere is the trapping of heat-producing the greenhouse effect. This isn't an easy problem to deal with, especially with deforestation which makes it less possible to decrease such concentration of greenhouse gases from our atmosphere.

A large percentage of the produced greenhouse gases come from electricity which is what powers almost everything in our daily lives, and life can simply not be imagined without its existence these days. Electricity is produced with the burning of fossil fuels, however, in the last few decades, scientific progress has been made in discovering new, cleaner ways of producing electricity.

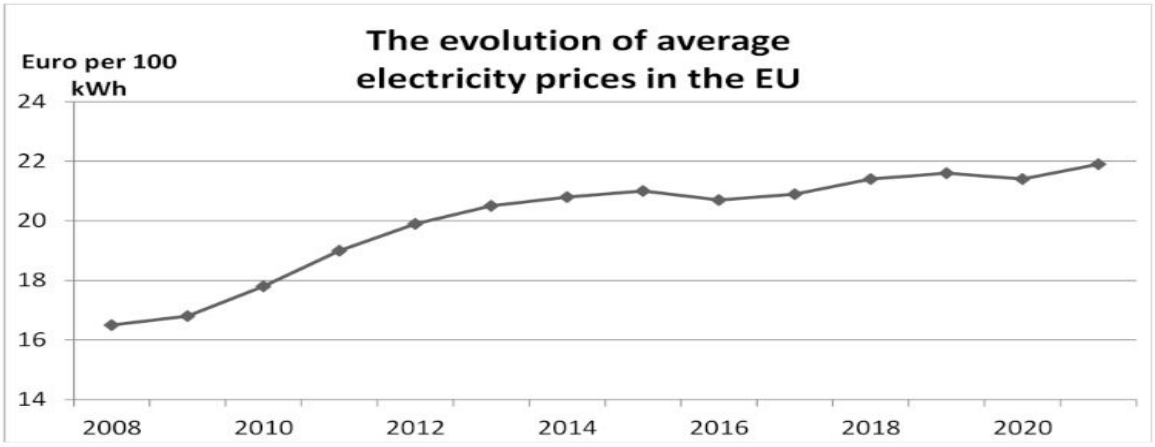


Figure 1: The evolution of average electricity prices per 100 KWh between 2008 and 2021 in the European Union [17].

To reduce the reliance on fossil fuels, which not only are limited in quantity but also very polluting and is the reason behind the disasters, we had to shift towards renewable sources of energy and increase the reliance on them. Solar energy investment alone has grown by roughly four-fold since 2018,leading

the way with 44% of global investment in renewable energy[1].Photovoltaic systems convert energy from the sun directly into electricity.These cells typically consist of thin wafers or strips of semiconductor material that produce a small electric current when exposed to sunlight. Multiple cells can be combined to create modules, which can then be wired together to form an array of any size. In recent years, there has been a substantial growth in the global capacity of PV systems, reaching an impressive 707.5 GWp by 2020. This capacity represents the theoretical production under optimal conditions, while the actual electricity production worldwide amounts to 135 GWp. Particularly, the European Union has witnessed significant advancements in solar energy adoption and expansion in recent times [35, 21, 8]. Germany holds the prominent position as the leading player in the European photovoltaic sector, boasting a total connected capacity exceeding 50 GW. Italy follows closely behind with 21 GW, and the United Kingdom with 13.3 GW. However, Poland is positioned within the second tier of the ranking, surpassing the Baltic States in terms of connected capacity [19].

Figure 2: Photovoltaic solar panel on roof of house [33].

To further encourage installing photovoltaic systems, as an outcome of this thesis an intuitive hybrid application has been developed where user can simulate and construct a PV system. It is important to note that the simulation output heavily relies on meteorological data such as solar irradiation and ambient temperature, as well as cadastral data such as roof direction, tilt, and area.

[1]Global investment in renewables capacity was $495 billion in 2022,up 17% from the year prior:https://about.bnef.com/blog/global-low-carbon-energy-technology-investment-surges-past-1-trillion-for-the-first-time/

Therefore in section 2 all simulation constraints has been discussed such that it can be easily realized during backend integration.

## 1.1 Related work

PV systems can be classified as either on-grid or off-grid. Off-grid systems consist of a PV module or array that generates direct current (DC), DC-DC converters (batteries) to store the DC power, and a DC-AC converter (power inverter) to convert the DC into alternating current (AC) for powering household appliances (electrical load). Conversely, on-grid systems employ power transformers instead of batteries and utilize net metering to connect to the local utility grid. In the case of surplus PV-generated power beyond the immediate demand, the excess energy is exported back to the grid. When there is a power outage, power from the utility grid is imported. The scale of the system must be specified early in the planning and design process. System performance modelling tools and computer simulations may help designers and system integrators forecast energy output and analyze potential configurations[6]. There is wide variety of software available to help with the analysis, simulation, and design of PV systems. The vast majority of systems include estimating the amount of solar radiation while also taking into consideration the characteristics and positioning of the PV system.The authors did a thorough analysis of the available literature on PV simulation tools in their research and reported on 37 tools that are frequently utilised for modelling and analysing single building systems to national energy systems [9]. The findings of this research demonstrate the wide range of available tools for PV simulation and analysis. When compared to the more traditional methods of sizing, such as manual computation, the planning and simulation software may speed up the whole design process.

Based on the form and purpose of software, PV system tools can be classified into five categories: analysis and planning, simulation, economic evaluation, site analysis and solar radiation map tools [3]. Simulation tools are computer software designed to predict the performance of power systems, using meteorological algorithms and/or databases. These tools are used to simulate, analyze, monitor, and visualize power system performance, but not optimize the system. INSEL is one such simulation tool developed by the University of Oldenburg, Germany [20]. It can simulate time series solar irradiance, PV power plants, and solar cooling and heating systems, and also detects faults in the system. The software provides PV component databases and meteorological databases from 2000 locations worldwide. TRNSYS, developed by the University of Wisconsin, Madison, USA, can simulate not only PV systems, but also other cogeneration systems such as wind turbines, fuel cells, and batteries [34]. It is flexible, allowing users to modify the mathematical model in its library. Both INSEL and

TRNSYS offer demo versions and require input data such as meteorological data and component design by users.Planning and analysis tools help users in organising, designing, sizing, optimizing sources, and accurately characterizing proposed systems. PVsyst [4],Pvsolpremium [1], SolarPro [27], F-CHART [22] and System Advisor Model (SAM) [18] are the most commonly used photovoltaic analysis and planning tools. In particular PVsyst is the most used solar project development software in the United States [6]. PV economic evaluation tools offer the capability to conduct economic analysis for user-designed systems. By inputting relevant cost and financial parameters, users can assess the feasibility of the proposed system or optimize the net benefit of electricity consumption. Running the analysis allows for the minimization of overall project costs while meeting load demands and adhering to project constraints. Homerpro can provide system optimization and technology options according to cost and energy resources availability [15]. PVWatts is an automated calculator software that provides rapid estimates for energy production and potential cost savings in grid-connected photovoltaic (PV) systems. It serves as a simplified version of PVForm and can be conveniently accessed either through the System Advisor Model (SAM) or via online platforms on the Internet [11].This tool only calculates for crystalline-silicon PV modules [29]. SAM [18] and RETScreen Expert [16] not only provides comprehensive analysis of PV system performance but also gives a detail financial analysis and optimization report. Solar radiation maps offer users a straightforward visual representation of solar resources across different locations worldwide. To fulfill this purpose, two online software programs, namely PVGIS and SolarGIS, are available. These programs utilize maps and satellite imagery to provide valuable geographical information. One of these programs, Photovoltaic Geographical Information System (PVGIS), not only estimates solar radiation but also offers solar radiation maps and calculates the annual energy generation potential for both grid-connected and standalone photovoltaic (PV) systems. SolarGIS is suitable for a wide range of applications, from small residential systems to large-scale solar power plants. Its outputs include detailed solar resource reports, energy yield forecasts, and financial analysis reports. It is available as a subscription-based service with different pricing options based on the level of access and the number of locations. Multiple literature review has been carried out on different PV simulation tools [42, 46, 24, 43]. Among all papers very less papers address in detail about some of the most important photovoltaic tools therefore in section 3 a comparative analysis of five major software has been presented.

## 1.2 Contribution

Currently available free PV planning and simulation software such as PVGIS and PVWatts provide important simulation information to help users make informed decisions before installing a PV system. However, they do not offer complete step-by-step guidelines for constructing a PV system. On the other hand, proprietary software like PV*SOL premium, PVsyst, and RETScreen Expert offer advanced functionalities, but they are developed for specific target user groups, such as PV system designers and engineers. Additionally, these premium software packages are not free to use, which means that house owners may not be able to afford them. Therefore, we carefully designed an intuitive user interface to cater to people of different age groups. Additionally, we have ensured that users are able to simulate and construct a photovoltaic system independently by following just a few straightforward steps.

## 1.3 Outline

Section 2 provides an overview of independent variables that needs to be realized to simulate a PV systems. In Section 3, a comparative analysis is presented of five major software packages used for analysis, quick estimations and calculations relevant to photovoltaic electricity production. These software packages include Photovoltaic Geographical Information System (PVGIS), PVWatts, PVsyst, PV*SOL premium, and RETScreen. The section concludes with a summary explaining how the proposed app solution resolves the existing problems for stakeholders such as house owners. Section 4 outlines the concept of prototypes and the UX laws that have been followed in order to guarantee the best user interface in the context of human-computer interaction. It begins with clarifying the basic nature of prototyping and the classification of prototypes based on prototype fidelity is presented. Further, a brief description of implemented UX laws has been discussed during prototype analysis and development. Section 5 present the technology stack which can be utilized in the construction of the software. This includes a comprehensive analysis of the programming languages, frameworks, libraries, and databases that can be used in the development process. Through this discussion, we aim to provide readers with a detailed understanding of the technological infrastructure that forms the backbone of our proposed app solution. section 6 present the outcome of the app implementation, including detailed descriptions of the main pages and their functionalities, along with corresponding screenshots. Finally section 7 provides a summary and discussed potential areas for further work.

# 2 Simulation of a photovoltaic system

The result of a photovoltaic system is affected by a variety of factors, some of which are independent of the PV system and others are dependent on it. The dependent variables are mainly electrical specifications provided by the manufacturer, such as Maximum Power Point (MPP). The MPP is the point at which a solar panel generates the maximum amount of power, and it is dependent on the electrical specifications of the panel, such as its voltage and current ratings.

The independent variables on the other hand, include meteorological data such as solar irradiance and the orientation of the PV panels with respect to the sun. Solar irradiance refers to the amount of solar energy that reaches a particular area, and it varies depending on the location, time of day, and weather conditions. The orientation of the panels with respect to the sun also affects the amount of incident radiation received, as the panels will receive more radiation when they are facing directly towards the sun.

Total energy generated by a photovoltaic system over the course of a year can be calculated using the following formula:

$$E_{total} = \sum (G(t)) \cdot A_{panel} \cdot \eta_{\cos} \cdot \eta_{sys} \cdot \Delta t \tag{1}$$

Where: $E_{total}$ is the total energy in a year. $\sum$ represents the summation over all the time steps in a year (e.g. hourly, half-hourly, etc.). $G(t)$ is the solar irradiation at time $t$. $A_{panel}$ is the solar panel area (in units of area, e.g. $m^2$). $\eta_{\cos}$ is the cosine efficiency factor, which takes into account the angle between the solar panel surface and the incoming sunlight. $\eta_{sys}$ is the overall system efficiency factor, which takes into account all the losses and inefficiencies in the photovoltaic system. Lastly, $\Delta t$ is the time step (in units of time, e.g. hours).

$G(t)$ can be obtained from meteorological data such as sunshine duration, cloud cover, and temperature. $\eta_{\cos}$ is the angle of incidence radiation on the solar panel which can be deribred from the SUN position data. Cadastral data helps to determining the orientation and inclination of the building's rooftop, which can impact the amount of solar radiation that falls on the solar panels. $\eta_{sys}$ helps to caluclate efficiency of the entire PV system, including losses due to wiring, inverter inefficiencies, shading, and other factors. It is typically expressed as a percentage, and can vary depending on the specific components of the PV system. $\eta_{sys}$ falls into dependent category which is mainly electrical specifications provided by the manufacturer, such as Maximum Power Point (MPP), panel voltage and current ratings,etc. To calculate photovoltaic total energy in a year independent variables (Meteorological data, Sun position and Cadastral data) needs to be realized. Therefore, each independent variables realization techniques will be discussed in the below sub-sections.

## 2.1 Meteorological data

In Germany, Deutscher Wetterdienst (DWD) provides virous meteorological datasets that can be used for solar simulation, such as air temperature, air pressure, wind speed, Direct normal irradiance (DNI), Golal horizontal irradiance (GHI), Direct horizontal irradiance, Diffuse horizontal irradiance, Solar azimuth.

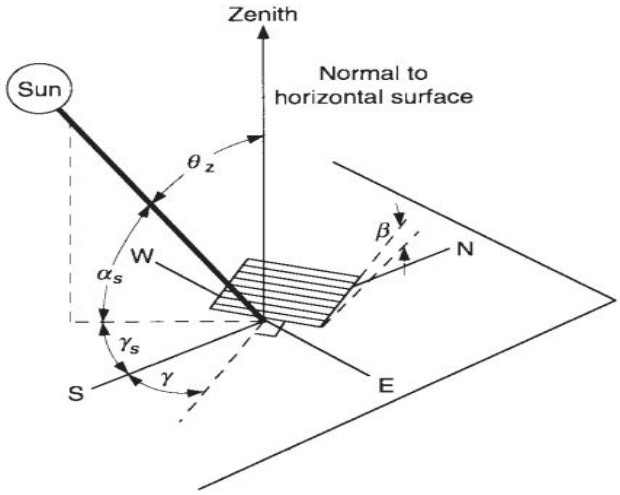


Figure 3: The angle of incidence, zenith angle, tilt angle, and azimuth angle for a tilted surface [13].

Direct normal irradiance (DNI) measures the amount of solar radiation received by a surface oriented perpendicular to the sun's rays, while global horizontal irradiance (GHI) measures the total amount of solar radiation received by a horizontal surface, including both direct and diffuse irradiance. Direct horizontal irradiance (DHI) is the amount of solar radiation received by a surface that is horizontal to the ground and facing the sun. On the other hand, Diffuse horizontal irradiance (DIF) is the amount of solar radiation received by a horizontal surface that is not directly facing the sun but has been scattered or reflected by the atmosphere.

To calculate the maximum amount of irradiance DNI is particularly important because it represents the most direct and concentrated form of solar radiation. PV systems with tracking panels can track the sun's movement throughout the day and adjust their angle to maximize the amount of DNI received. On the other hand, Direct horizontal irradiance and diffuse horizontal irradiance are also important because they represent the overall amount of incident radiation that is available at a particular location and can be used to estimate the total amount of solar energy that a PV system would receive. Global horizontal irradiance is

the sum of direct and diffuse horizontal irradiance which has been measured by DWD dataset.

Solar azimuth is a measure of the angle between true south and the point on the horizon where the sun appears to rise. These parameters can be used to estimate the amount of solar energy that a PV system would receive at a particular location, and thus optimize the system's performance.

## 2.2 Sun position

In order to optimize the performance of photovoltaic (PV) systems, it is necessary to accurately determine the position of the sun in the sky. The position of the sun is a key factor in determining the amount of solar radiation that reaches the PV panels, which in turn affects the efficiency and output of the PV system. By precisely calculating the sun position and the resulting solar radiation, it is possible to ensure that the design and operation of the PV system are optimized, leading to maximum efficiency and output. Therefore, accurate determination of the sun position is critical for the effective design, modeling, and analysis of PV systems. The position of the sun can be calculated using mathematical formulas based on the location, date, and time. There are several different formulas that can be used, but one commonly used method is called the "Solar Position Algorithm" (SPA) [39].

In the Solar Position Algorithm (SPA), the first step is to calculate the Julian Day (JD) of the year. The JD is a continuous count of days and fractions of a day, starting from noon Universal Time (UT) on January 1, 4713 BC (Julian calendar). JD is calculated for the date and time of interest using the following equation:

$$JD = 367n - 7 \cdot \frac{(n + int(\frac{m+9}{12}))}{4} + int(275 \cdot \frac{m}{9}) + d + 1721013.5 + \frac{UT}{24} \quad (2)$$

Where: $n$ is the number of years since $2000$ (e.g., for $2023$, $n = 30$), $m$ is the month $(1 - 12)$, $d$ is the day of the month $(1 - 31)$, $UT$ is the Universal Time in hours. $1721013.5$ represents the Julian date for noon Universal Time on $January\ 1,\ 4713\ BC$ in the Julian calendar. It is used as an offset to ensure that the Julian date calculation starts from this historical date, which is commonly used as the reference date for astronomical calculations.

After calculating the $JD$ the next step is to calculate the number of days that have elapsed since $J2000$. This value is denoted as "n" and is calculated using the following formula:

$$n = JD - 2451545. \quad (3)$$

The constant value $2451545.0$ represents the Julian Day for the $J2000\ epoch$, which is defined as noon on $January\ 1,\ 2000$ in the Gregorian calendar. Subtracting this value from the $JD$ yields the number of days that have elapsed since the $J2000\ epoch$.

Next step is to calculate the mean solar noon time for a given location. This can be done using the following formula:

$$Tsn = 12.0 - (\frac{longitude}{15.0}) \tag{4}$$

where $Tsn$ is the mean solar noon time in hours, and $longitude$ is the longitude of the location in degrees. The formula calculates the time difference between the solar noon at the location's longitude and the solar noon at the standard meridian (which is $15$ degrees for each time zone). The resulting value is then subtracted from $12.0$ (noon) to get the mean solar noon time for the location. The result is expressed in hours.This calculation is necessary to adjust for the local time of solar noon, which varies with longitude. By accurately determining the mean solar noon time for the location, it is possible to account for the time zone offset and ensure that the solar position calculations are accurate.

The next step in SPA algorithm is calculating the solar declination angle for a given date. The solar declination angle is the angular distance between the equatorial plane and the line connecting the center of the sun and the center of the earth. It is calculated using the following formula:

$$\sin\ \delta = \sin\epsilon\ \sin\ \epsilon_0 + \cos\epsilon\ \cos\ \epsilon_0 \cos\ h \tag{5}$$

where $\delta$ is the solar declination angle in radians, $\epsilon$ is the obliquity of the ecliptic (the angle between the ecliptic plane and the equatorial plane), $\epsilon_0$ is the obliquity of the ecliptic at $J2000$, and $h$ is the hour angle of the sun.

The hour angle h is calculated using the formula:

$$h = (LST\ -\ 12) \cdot 15 \tag{6}$$

where $LST$ is the local solar time in hours. The local solar time can be calculated using the longitude of the location, the equation of time (which is a function of the date), and the time zone offset.The solar declination angle is an important parameter for determining the position of the sun in the sky, and it varies throughout the year due to the tilt of the earth's axis and its orbit around the sun.

After calculating the declination next step in SPA algorithm is calculating the solar hour angle for a given time and location. The solar hour angle refers to the angular deviation of the sun's position, either to the east or west, from the local meridian caused by the Earth's rotation. This value can be calculated using the following formula:

$$\cos\ \omega = \frac{(\sin\ \theta \sin\ \Phi - \ \sin\ \delta)}{(\cos\ \theta \cos\ \Phi)} \tag{7}$$

where $\omega$ is the solar hour angle in radians, $\theta$ is the solar altitude angle (the angle between the horizon and the center of the sun), $\Phi$ is the latitude of the location, and $\delta$ is the solar declination angle.And the solar altitude angle can be calculated using the following formula:

$$\sin\ \theta = \sin\ \Phi \sin\ \delta + \ \cos\ \Phi \cos\ \delta \cos(HA) \tag{8}$$

where $HA$ is the hour angle of the sun, which has been described in 7.By accurately calculating the solar hour angle, it is possible to determine the position of the sun relative to a PV panel and calculate the amount of solar radiation that reaches it.

The next step is calculating the extraterrestrial radiation for a given date. The extraterrestrial radiation is the amount of solar radiation that would be received at the top of the earth's atmosphere if it were not for atmospheric absorption and scattering. The extraterrestrial radiation can be calculated using the following formula:

$$I_0 = Isc\ \cos\theta_z \tag{9}$$

where $Isc$ is the solar constant (the amount of solar radiation that reaches the top of the earth's atmosphere on a surface perpendicular to the sun's rays), and $\theta_z$ is the solar zenith angle (the angle between the zenith and the center of the sun). The solar zenith angle can be calculated using the formula:

$$\cos\theta z = \sin\Phi \sin\delta + \cos\Phi \cos\delta \cos(HA) \tag{10}$$

where $\phi$ is the latitude of the location,$\delta$ is the solar declination angle, and $HA$ is the hour angle of the sun, which can be calculated as described in 7.

And the final step is calculating the air mass for a given location and time. The air mass is a measure of the amount of air that sunlight passes through before reaching a PV panel, and it affects the amount of energy that the panel can produce. The air mass can be calculated using the following formula:

$$AM = \frac{1}{\cos(\theta z)} \tag{11}$$

where $\theta_z$ is the solar zenith angle, which can be calculated as described in 10.

The Solar Position Algorithm involves several steps to accurately calculate the position of the sun in the sky and the resulting solar radiation that reaches a PV panel. By determining the sun's position and the amount of solar radiation that reaches the panel, it is possible to optimize the design and performance of a PV system and ensure that it operates at maximum efficiency. The key steps in the algorithm include calculating the Julian Day, the number of days since $J2000$, the mean solar noon, the hour angle, the solar altitude angle, the solar hour angle, the extraterrestrial radiation, and the air mass. These parameters are all interconnected and are used to calculate the position of the sun in the sky and the amount of solar radiation that reaches a PV panel.

## 2.3 Cadastral data

When considering the investment of installing rooftop-mounted solar panels, stakeholders need to know the orientation and inclination of the building's rooftop, as it significantly impacts the irradiation received by the solar panels. To achieve this, the user's address input is automatically used to calculate these angles. The land cadaster data for North Rhine-Westphalia (LANUV) is typically provided in a geographic information system (GIS) format, such as shapefiles or geodatabases, which contain several different layers of information about the land use, buildings, and infrastructure in the state.

The data contains various fields and attributes that can be useful for PV system simulation, including:

- Geometry information (polygon or point) for each building
- Building area
- Building height
- Building usage (residential, commercial, etc.)
- Roof type and area for buildings
- Address and location information (latitude, longitude) for building

This dataset offers an advantage in identifying and eliminating obstructions such as chimneys, and cutting shadows from rooftop surfaces, which may otherwise reduce the efficiency of the PV panels shows in Data licence Germany attribution Version 2.0 [2].These fields can be used to determine the location, size, and

[2]Data licence Germany attribution Version 2.0 - `https://www.opengeodata.nrw.de/produkte/umwelt_klima/klima/solarkataster/photovoltaik/`

orientation of buildings and rooftops, which are important factors in determining the potential for solar energy generation from rooftop-mounted PV systems. Therefore, having access to a wide range of cadastral data,can be more effective and useful for a larger number of users, as they can provide more accurate and detailed information about potential PV system installations.

## 2.4 Incident radiation

Before reaching the surface of a PV panel, a significant percentage of the solar beams scatter and disperse through the atmosphere, resulting in two types of solar radiation on Earth: direct radiation and diffuse radiation. The quantity of solar radiation that comes directly from the sun in a straight line is referred to as direct solar radiation, whereas diffuse radiation is the amount of energy that comes from dispersed sunlight in the atmosphere.

$$IR_{global},\ H = IR_{direct},\ H + IR_{diffuse},\ H \tag{12}$$

PV panels are often not installed horizontally but rather on a tilted rooftop with a certain orientation. As a result, the angle of incidence $\cos\theta_{incidence}$ coming from the PV panel can be estimated using the following formula.

$$\cos\theta_{incidence} = \cos\varphi_0 \cdot \sin\beta \cdot \cos(\gamma_0 - \gamma) + \sin\varphi_0 \cdot \cos\beta \tag{13}$$

where $\beta$ is the panel's inclination (tilt), $\gamma$ is its azimuth, North is the origin, and West is positive. After determining the angle of incidence $\cos\theta_{incidence}$, the solar direct radiation on the bent PV panel can be determined geometrically is as follows.

$$IR_{direct},\ T = IR_{direct}, H \cdot \frac{\cos\theta_{incidence}}{\sin(\gamma_0)} \tag{14}$$

And the diffuse radiation formula is.

$$IR_{diffuse},\ T = IR_{diffuse}, H \cdot \frac{\cos\beta}{2} \tag{15}$$

As the direct and diffuse beams hit the surface, some of them reflect and reach the PV panels. Reflected radiation can be calculated as follows

$$IR_{reflected},\ T = r_{albedo} \cdot IR_{global},\ H \cdot \frac{1 - \cos\beta}{2} \tag{16}$$

where $r_{albedo}$ is the surface reflectance. The value of $r_{albedo}$ varies according on the substance. Snow, for example, has an $80$% reflectivity whereas new asphalt has a $4$% reflection.

Therefore, the global incident irradiance IR on the PV surface is as follows:

$$IR = IR_{direct},\ T + IR_{diffuse},\ T + IR_{reflected},\ T \tag{17}$$

## 2.5 Financial model

Stakeholders attach significant importance to reducing the expenditure on electricity when designing a PV system. Thus, the main factors considered here are the primary factors that influence the overall costs of solar energy generation and storage. These factors include the buying and selling rates from the grid, the production costs associated with solar energy, and the expenses related to energy storage. Section 2.5.2 presents a detailed calculation of the costs involved in producing solar energy, while Section 2.5.3 explores the costs associated with storing energy, taking into account factors such as degradation and lifetime. In Section 2.5.1, we combine these costs to determine the total expenses and compare them with the costs incurred by the local grid.

### 2.5.1 Levelized Cost of Electricity

The levelized cost of electricity (LCOE) is the entire cost of energy, regardless of whether it is imported from the grid or generated on-site [47].

$$C_{LCOE} = \frac{C_{import} - C_{feed-in} + C_{BESS} + C_{PV}}{E_{load}} \tag{18}$$

The annual costs incurred for importing electricity from the grid are denoted as $C_{import}$, while the annual revenues generated from exporting electricity to the grid are denoted as $C_{feed-in}$. The annual costs of energy storage are represented by $C_{BESS}$, and the annual costs of producing solar energy are represented by $C_{PV}$.

### 2.5.2 Levelized Cost of Solar Energy

The levelized cost of solar energy is defined as the ratio of the entire cost of the system to its lifetime production [25].

$$C_{LCOE,\,PV} = \frac{Total\ Life\ Cycle\ Cost}{Total\ Lifetime\ Energy\ Production\ (kWh)} \tag{19}$$

The value of the output energy of the system decreases over time. This decrease is measured by the discount rate $r_{dr,\,PV,\,real}$. If the inflation rate $r_{inflation}$ is considered, the nominal discount rate $r_{dr,\,PV}$ can be given as [45]

$$r_{dr,\,PV} = (1 + r_{dr,\,PV,\,real}) \cdot (1 + r_{inflation})^{-1} \tag{20}$$

Accounting for these is critical to determining the true cost of energy. In order to account for this in terms of output energy value, the yearly output energy is divided by the capital recovery factor (CRF), which is given as [45]

$$r_{CRF,\,PV} = \frac{r_{dr,\,PV} \cdot (1 + r_{dr,\,PV})^{N_{PV,\,years}}}{(1 + r_{dr,\,PV})^{N_{PV,\,years}\,-1}} \tag{21}$$

The cost of the PV system can be divided into equity $C_{equity,\,PV}$ which is paid now, and follow-on investments for operation and maintenance $C_{O\&M}$, which are paid yearly [45].

$$C_{system} = C_{equity,\,PV} + C_{O\&M,\,NPV} \tag{22}$$

Calculating the net present value (NPV) of the O&M costs is necessary, as the future value of money may vary. The NPV coefficient is derived from [45]

$$\alpha N_{PV} = \sum_{n=1}^{N_{PV,\,years}} \frac{(1 + r_{inflation})^{n-1}}{(1 + r_{dr,\,PV})^{n}} \tag{23}$$

and equity is described here as

$$C_{equity,\,PV} = C_{module} \cdot N_{modules} + C_{additional} \tag{24}$$

where $C_{module}$ is the cost of one module and $N_{modules}$ are the number of purchased modules and $N_{additional}$ are additional costs that includes installation and inverter costs, and installation fees.As reported in [37], the current discount rate for PV is 6-9%. The discount rate could be as much as 2-3% lower over the next decade, and could fall by a further 1-2% by 2040. A discount rate for a PV system $r_{dr,\,PV,\,real}$ at 5% is used to be in line with the best discount rate for PV systems in Kenya [36] and the lifetime of a PV system $N_{PV,\,years}$ to be 25 years. Now by putting it together Equation (19) becomes

$$C_{equity,\,PV} = \frac{C_{equity,\,PV} + C_{O\&M\cdot\,\alpha NPV}}{E_{AC,\,annual}} \cdot r_{CRF,\,PV} \tag{25}$$

### 2.5.3 Levelized Cost of Stored Energy

The levelized cost of stored energy is the ratio of the battery's lifetime expenses to its lifetime output and is provided by [7].

$$C_{LCOE,\,BESS} = \frac{C_{BESS} \cdot C_{nom}}{E_{BESS,\,lifetime}} \tag{26}$$

# 3 Analysis of existing photovoltaic softwares

Photovoltaic system manufacturing and installation are currently experiencing remarkable growth as one of the world's fastest-growing industries. In general, both construction companies and homeowners invest in photovoltaic analysis and simulation software to assess the feasibility of installing such systems. While some free simulation software options are available on the internet, they often fail to meet users' expectations, compelling them to purchase proprietary software that can cost thousands of euros. Consequently, many house owners

and freelance engineers do not have the budget to purchase expensive computation tools, and so, in developing countries, free software is commonly used [14]. To evaluate the performance and economic potential of solar systems, streamline the design process, and optimise the utilisation of renewable energy resources, numerous simulation software tools have been developed. This section provides a comparison of five prominent software programmes widely used for rapid estimates and computations in solar power production: Photovoltaic Geographical Information System (PVGIS), PVWatts, PVsyst, PV*SOL premium, and RETScreen.

## 3.1 PVGIS

PVGIS [3] is a photovoltaic simulation tool developed by the Joint Research Centre of the European Commission available in five languages with English as the primary. It has undergone several updates and improvements since its initial release in 2001, with the latest version in 2021 being PVGIS 5. The user interface of PVGIS is primarily map-based, with a map of Europe as the default view is clearly visible on the left half of the page when the app is first opened. Users can easily zoom in and out of the map to select the location of their PV system, or they can search for a specific location using a search bar located at the bottom of the screen. One significant limitation of this programme is that it is not applicable to nations outside of Europe and Africa. There are maps available based on solar radiation levels and a temperature database. The data interface for the computations is provided in a plain grey environment on the right side. Once a location has been selected, users can choose from several different simulation options, including PV potential, solar radiation, and climate data. The PV potential options like performance of grid-connected PV, off-grid PV provides a quick estimate of the energy yield of a PV system in the selected location, while the solar radiation and climate data options provide more detailed data on solar radiation and climate conditions in the area.

Users can also select the characteristics of their PV system, including module type, tilt angle, and orientation. PVGIS provides default values for these parameters based on the location selected, but users can adjust these values to better match their specific system. In addition to the map-based interface, PVGIS also provides a detailed output report that includes information on the energy yield of the PV system, as well as information on the system's performance. The output report includes graphs and tables that make it easy for users to understand the performance of their PV system under different conditions. While PVGIS offers comprehensive data spanning a considerable timeframe, it does not offer

[3] PVGIS - `https://re.jrc.ec.europa.eu/pvg_tools/en/tools.html`

financial forecasts regarding the energy output of the photovoltaic system. An advantage of PVGIS is its portability, enabling users to access it conveniently on various devices, including smartphones, without the need for installation.

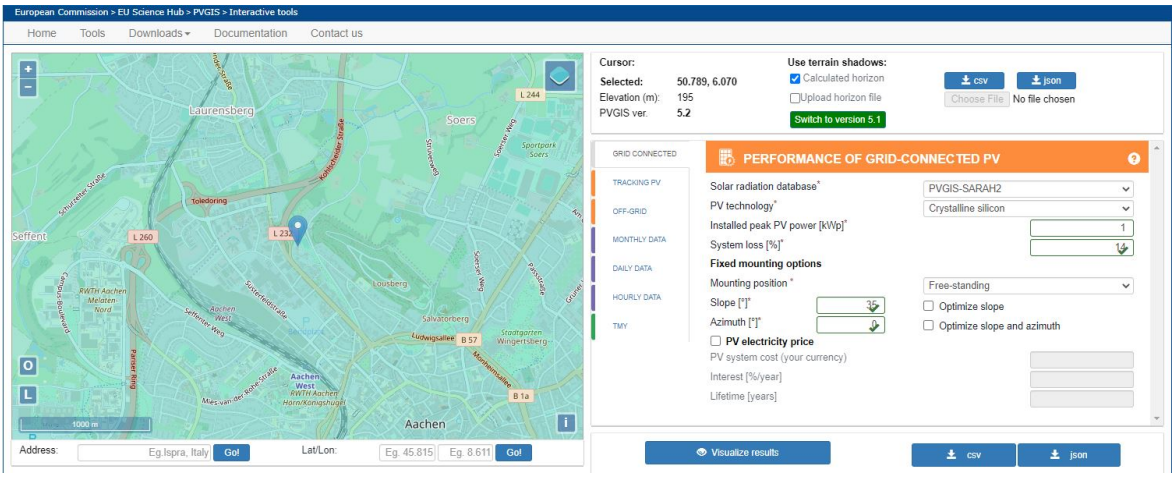


Figure 4: PVGIS interface [38].

## 3.2 PVWatts

PVWatts[4] is a web-based solar energy calculator developed by the National Renewable Energy Laboratory (NREL) in the United States. It has been in development since the early 2000s, and has gone through several versions and updates over the years. The first version of PVWatts, PVWatts v.1, was released in 2004. In 2016, NREL released PVWatts v.5, which added further improvements and enhancements to the tool. The latest version of the software provides users with a user-friendly tutorial that simplifies the process of estimating the energy production of a photovoltaic installation. On the main page, there is a prominently displayed 'Get Started' box where users can input either their postal code or the name of a city. Upon entering the address, the tool opens a new window tab with Google Maps, allowing users to precisely outline the area of the photovoltaic system by utilising the zoom feature. The tool offers two options for determining the desired capacity: users can either directly input the desired capacity or utilise the drawing tool, which enables them to select a preferred photovoltaic efficiency for accurate calculations. Both of these options are readily available to the user. Once the drawing is completed, the total capacity of the system can be determined by summing up the cumulative values.

To calculate the direct current (DC) energy for each hour, the PV system's DC rating and incident solar radiation are utilized and adjusted for the temperature

[4]PVWatts - `https://pvwatts.nrel.gov/pvwatts.php`

of the PV cells. Subsequently, the alternating current (AC) energy for each hour is determined by multiplying the DC energy with the overall DC-to-AC derate factor and accounting for inverter efficiency based on the load. The AC energy values for each hour are then aggregated to obtain monthly and annual AC energy production figures. While default system parameters are employed, users are provided with the flexibility to modify the following parameters: DC system size, module type, PV array configuration (fixed or tracking), system losses, PV array tilt angle, and PV array azimuth angle. Moreover, advanced parameters can be adjusted at the bottom of the form to enhance the accuracy of the estimates.

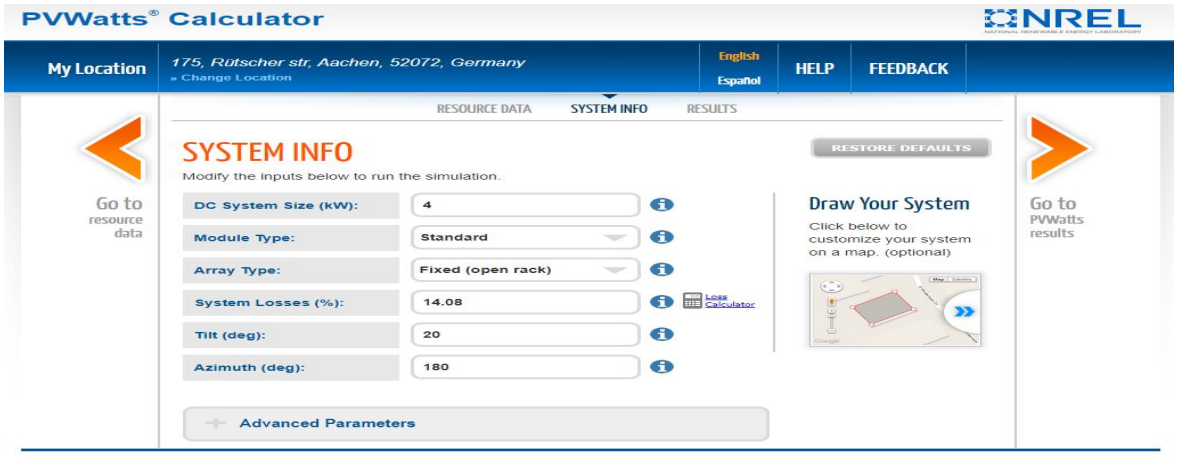


Figure 5: PVWatts interface [32].

## 3.3 PVsyst

PVsyst is a desktop software application designed to simulate the performance of grid-connected or stand-alone photovoltaic (PV) system. The software encompasses a wide range of functionalities, incorporating extensive databases of meteorological data and PV system components. In addition, PVsyst offers various general solar energy tools to aid in system analysis. PVsystis developed by Swiss physicist Andre Mermoud and electrical engineer Michel Villoz [5]. it has undergone several updates and improvements since its initial release PVsyst 1.0 in 1997, with the latest version in 2022 being PVsyst 7.3.The user interface is organized into several tabs, each of which represents a different stage in the simulation process. For example, the first tab is the "Site" tab, where users can input the location and coordinates of their PV system project. The next tab is the "Setup" tab, where users can input details such as the module type, inverter type, and system configuration.

[5]PVsyst software evaluation - `https://www.pvsyst.com/software-evaluation/`

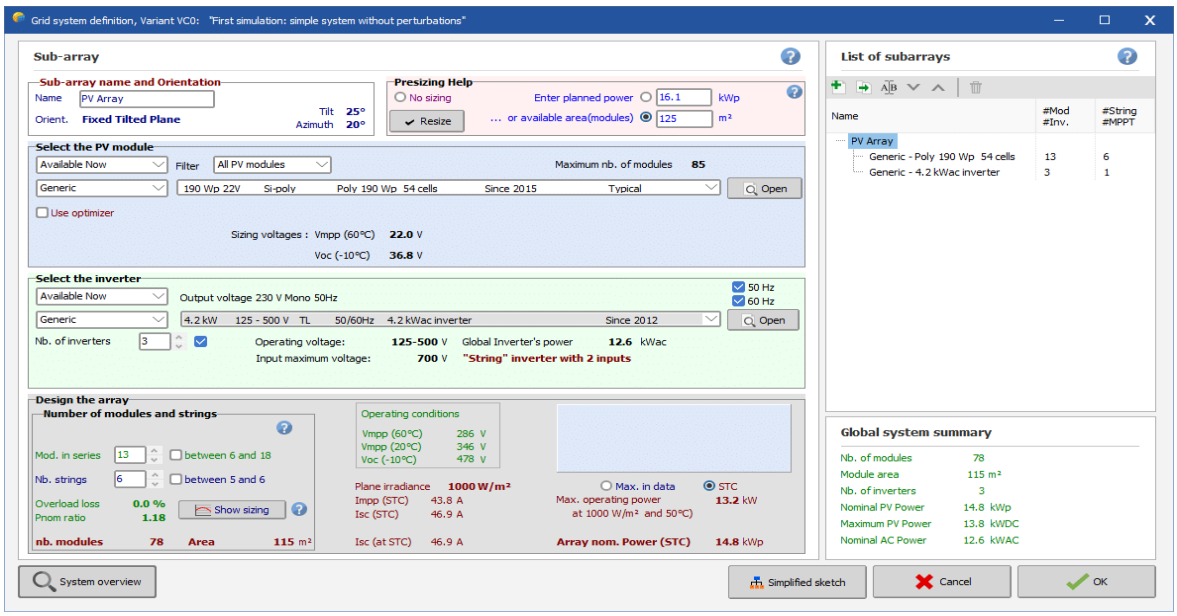


Figure 6: PVSyst system design board [2].

The interface includes options for selecting input parameters, viewing and customizing simulation results, and performing financial analysis. The user interface also provides options for specifying simulation settings, such as the simulation period, time step, irradiance model, and system losses. The software provides a 3D CAD modeling environment that considers surrounding obstacles. However, it lacks the ability to import complex 3D models, requiring users to create models using simple blocks provided. This makes the modeling process time-consuming and less accurate. Users can input weather and shading data, and the software accounts for various losses such as partial shading effects, module mismatches, wiring losses, inverter losses, and ambient temperature variations. Simulation settings allow for input customization such as plane orientation, system components, PV array configuration, and battery pack. Reports can be generated for each simulation run, including parameters and results. The software also supports detailed economic evaluations based on real component prices, additional costs, and investment conditions. PVsyst is a complex software tool with many advanced features, and it may take some time for users to become familiar with the interface and simulation model. This software is geared to the needs of architects, engineers, and researchers. Initially users may need to invest time in learning how to use the software effectively, which could be a barrier for some users. PVsyst program screen cannot be maximized therefore can be tedious to see all parameters if using a small monitor. As PVsyst is a proprietary software tool and can be quite expensive, which may limit its accessibility for smaller organizations or individuals. The cost may also be a factor in deciding whether to use PVsyst or alternative software tools. PVsyst offers shared access

to the software through their online platform, PVsyst Cloud, which allows multiple users to access the software and collaborate on projects without having to purchase individual licenses.

## 3.4 PV*SOL premium

PV*SOL premium is German software developed by Valentine Software [6] for dynamic simulation program with 3D visualization and detailed shading analysis of photovoltaic systems. The user interface is designed to be intuitive and user-friendly, with a tab-based layout that guides users through the simulation process. Each tab represents a different stage in the simulation process, such as Project, 3D Design, Shading, Simulation, and Results tabs. The interface includes options for selecting input parameters, viewing and customizing simulation results, and performing financial analysis.There are options for specifying simulation settings, such as the simulation period, time step, irradiance model, and system losses. Users can also specify weather data, module and inverter types, and shading objects for their location.The software offers a 3D design feature for visualizing buildings and shading objects, allowing users to rotate the land area along with the objects. It enables convenient drag-and-drop functionality for shading objects and provides animated sun-path visualization. Additionally, the software includes shade frequency distribution analysis and detailed shade analysis for each individual module. Users have the flexibility to specify parameters such as the number of covered areas, PV modules, inverters, PV generator output, orientation, azimuth, inclination, and installation type. The software simulates the performance of the PV system based on the input parameters and simulation settings. The simulation model includes detailed calculations of factors such as module temperature, DC and AC power losses, shading losses, and system losses. The simulation reports also include graphs, tables, and other visual aids that make it easy for users to understand the performance of their PV system under different conditions. One potential drawback of PVSOL*premium's software is that it may be overwhelming for new users, as it includes a wide range of input parameters and simulation settings that may require some time to learn and understand. Additionally, the cost of the software may be a limiting factor for some users, as PVSOL*premium is a proprietary software tool and can be quite expensive.

[6] PV*SOL premium - `https://valentin-software.com/produkte/pvsol-premium/`

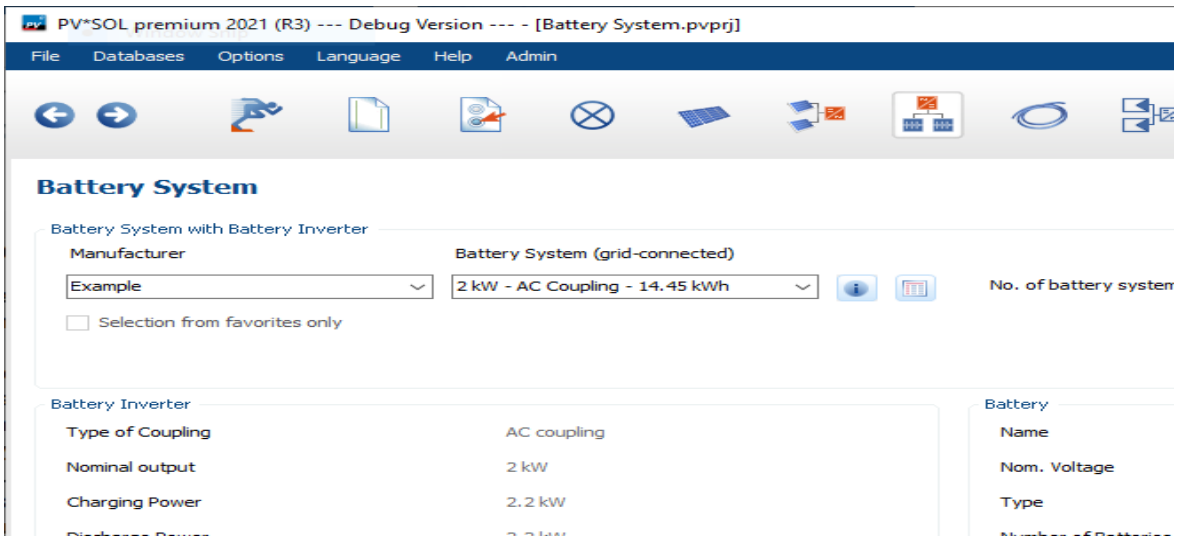


Figure 7: PV*SOL premium: Setup with battery system [1].

## 3.5 RETScreen Expert

The RETScreen[7] software, a decision support tool, has been collaboratively developed by government, industry, and academia. Initially created in 1996 by the Natural Resources Canada (NRCan) CANMET Energy Technology Centre for the analysis of renewable energy technologies, it has evolved to encompass a comprehensive set of features. The software offers multilingual support, along with databases for products, projects, hydrology, and climate. Additionally, it provides a detailed user manual and a college/university-level training course based on case studies, including an engineering e-textbook.

[7] RETScreen Expert - https://natural-resources.canada.ca/maps-tools-and-publications/tools/modelling-tools/retscreen/7465

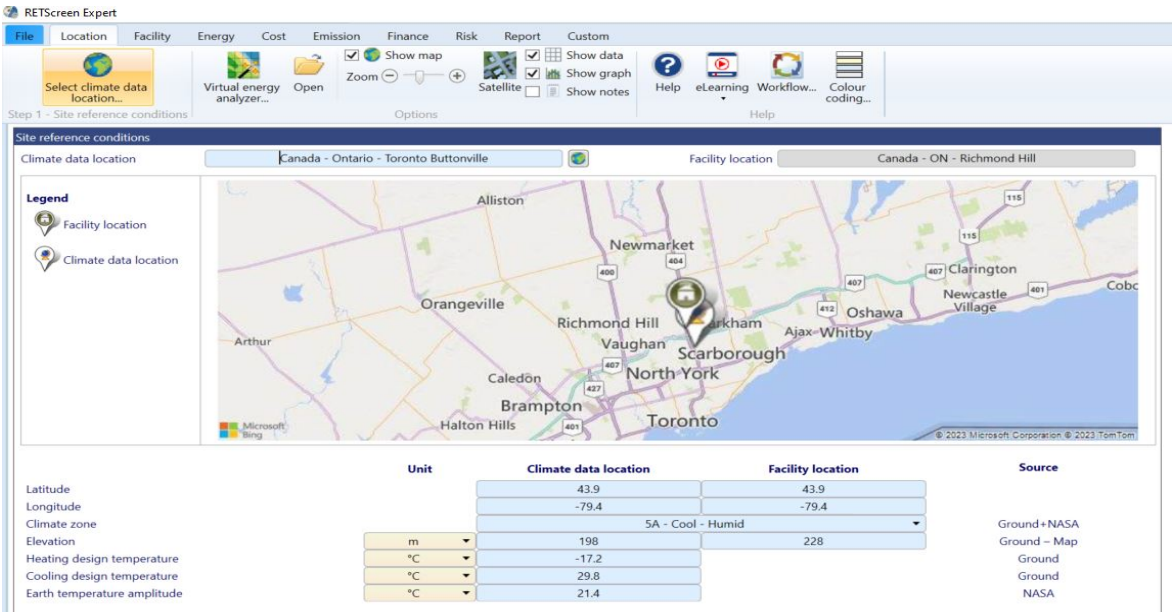


Figure 8: Location: Self-Simulation in RETScreen Expert [16].

The first step is to define the scope of the project, including the location of the system, the size of the system. The next step is to collect data on the project site, including irradiance data, temperature data, and shading data. RETScreen includes a built-in database of weather data for thousands of locations around the world, which can be used to obtain the necessary data. Once the data has been collected, the next step is to specify the components of the PV system. This includes the PV module type, inverter type, and system configuration. Once the inputs have been specified, RETScreen can be used to simulate the performance of the PV system. This includes calculating the energy output of the system and the financial metrics associated with the project. Based on results of simulation, the PV system can be optimized to improve its performance and financial viability. This may involve adjusting the system configuration or selecting different components. The full functionality of RETScreen Expert (including the ability to save, print and export files) is available in Professional mode by purchasing a renewable 12-month subscription, currently priced at CAD 869 per subscribing.

| Software | Advantages | Disadvantages | Target group | Price |
|---|---|---|---|---|
| PVGIS (Web app) | - Provides detailed data and analysis for the European region, including data on irradiance, temperature, and air temperature. | - Not applicable for countries outside of Europe and Africa.<br>- Does not indicate any financial projections of the energy produced by the photovoltaic installation. | Researchers, PV system designers. | Free to use. |
| PVWatts (Web app) | - Simple and easy to use, with a user-friendly interface. | - Limited input parameters and analysis options compared to other software tools. | Homeowners, small business owners, and other non-experts. | Free to use. |
| PVsyst (Desktop app) | - Comprehensive analysis tools, with detailed electrical and financial analysis capabilities.<br>- Offers advanced features such as shading analysis and simulation of complex system configurations. | - PVsyst is a powerful tool with many advanced features, but it can also have a steep learning curve for new users. It may require some training or experience to use the software effectively.<br>- Users may need to ensure that their system meets the minimum requirements to run the software effectively. | PV system designers, installers, and engineers | 611€ / year |
| PV*SOL premium (Desktop app) | - Detailed shading analysis and optimization of system components, with advanced visualization tools. Offers advanced features such as 3D shading visualization and integrated thermal analysis. | - it may be overwhelming for new users, as it includes a wide range of input parameters and simulation settings that may require some time to learn and understand. | PV system designers, installers, and engineers | 1295€ / year |
| RETScreen Expert (Desktop app) | - Provides a comprehensive analysis of PV system performance, including financial analysis and optimization. Offers advanced features such as detailed cost analysis and support for multiple languages. | - RETScreen Expert focuses mainly on financial analysis and optimization, rather than providing comprehensive electrical analysis or system design tools. This may make it less suitable for certain types of PV system projects.<br>- Less visually appealing than other software tools. | PV system designers, financial analysts, and project managers | 869 CAD / year |

Figure 9: Main features of various software of photovoltaic systems.

All the PV software tools discussed above, including PVGIS, PVWatts, PVsyst, PVSOL Premium, and RETScreen, have been developed to cater to different tar-

get groups such as house owners, professional PV system designers, engineers, researchers, and others. Commercial software tools like PVsyst, PVSOL Premium, and RETScreen are mostly used by stakeholders from photovoltaic construction companies, enabling them to perform energy simulation, financial analysis, and a wide range of other operations to construct photovoltaic systems. On the other hand, house owners have the option to use free PV simulation software such as PVGIS and PVWatts to obtain important information and make informed decisions before installing a PV system.

Currently available PV simulation software only provides some necessary PV simulation information, rather than guiding the house owner throughout the whole process, starting with PV energy simulation, helping in decision making by providing different reports such as energy consumption, energy production, and financial reports, and assisting in constructing a PV system with the help of qualified craftsman. Our proposed solution targets mainly house owners and small business owners who want to install photovoltaic systems on their property. We have carefully designed the user interface to cater to people of different age groups, and the app user needs to follow five simple steps to simulate and construct a PV system. We have ensured that each step is designed carefully to provide a smooth user experience, see Section 6.

# 4 User experience design

User experience design is the process of increasing user happiness with a product by improving its usability, accessibility, and happiness during an interaction [23]. In this section, we investigate on how we approached creating an attractive user experience design from beginning to end to help users navigate the information being provided, interactive visualizations, and interaction possibilities. A variety of aspects must be considered in order to get the best possible user experience, and these considerations must always be made at the system analysis and design phase. These factors add up to what is known as the “Mobile Equation” [30] which involves considering the various device types, operating systems, and screen sizes that all must be addressed. In today’s software industry, achieving the right user experience design means achieving effective communication of information to the users that belong to our target group.

To analyse desired solutions and convey them to users and other stakeholders early on, these solutions have to be represented in the form of scenarios, mockups, simulations, interactive prototypes, or beta-versions of products. Throughout the user interface design, we choose the process of prototyping by incorporating stakeholder feedback over several iterations and following user experience laws and standards.

## 4.1 Designing prototypes

According to Design council (2015) prototypes help finding unanticipated problems with creative ideas and gives insight into how the design will be used before a finished version is created [12]. The representations mainly serve two purposes. Firstly, they could serve as proof of concept, determining if things are acceptable and whether they can be anticipated to satisfy the needs they want to address. Secondly, they act as a channel of communication between the designer, users, and other stakeholders. Therefore, it is certain that prototyping plays an essential role in the process of developing excellent user experiences. In the early stages of the design process a simple prototype can be used to test underlying principles, while in late stages a more accurate prototype is required to refine details in form and function. With several iterations, coupled with evaluations and incorporated feedback from the potential end-users, a presentable and accurate product could eventually be developed and launched.

Prototypes however don't have to look like the final product, they can have different levels of detail, different fidelity. The fidelity of a prototype refers to how it conveys the look and feel of the final product. In other word, fidelity refers to the degree of precision, attention to detail, and the functionality of a prototype. Fidelity can varies in the areas of Visual Design, Content, and Interactivity. Two types of prototypes can be distinguished [44]. One is called Low Fidelity (Lo-Fi) prototypes and the other one is called High Fidelity (Hi-Fi) prototypes. The initial type of prototype does not aim to appear like or operate like the planned end-product but rather is used to test early design ideas or partial solutions among the users and stakeholders. These prototypes do not strive to be realistic in appearance or operation. The latter type acts as a simulation of the final product and is used to illustrate how the product functions without the need for a full implementation.

### 4.1.1 Low fidelity prototypes

Low fidelity prototyping is a quick and easy method for turning high-level design concepts into observable and testable artifacts. Whether UX designers use paper or digital wireframes, low-fidelity prototyping is the starting point for testing design concepts and user workflows. For example, the design of a mobile app screen may be a mix of text and boxes that provide users with a better grasp of the screen's structure, content, and function. Low fidelity design enables the product team to test design concepts and user processes prior to committing to a certain methodology. Low fidelity design can be created both on paper or digitally. During software development, team members usually utilise low-fidelity designs to explore numerous ideas within a limited time-frame. They generate low-fidelity designs in real-time during brainstorming sessions, enabling them to

prototype user flows and observe how users would navigate through a product or tool. This approach provides a comprehensive understanding of the clarity and logic of the chosen navigation scheme.They are a simple and efficient way of communicating an idea or a concept early in the process without putting in too much effort. According to Rettig [40], the use of paper-based prototyping makes it possible to present a system while it is still in an early stage of development and to test a far larger number of concepts than is feasible when using high-fidelity prototypes. The creation of low-fidelity prototypes may be accomplished on a budget using methods such as sketches, storyboards, and paper-based prototypes. Designers are not constrained by technology limits, do not need to spend time on programming, and can nevertheless recognise and fix issues with usability at an early point in the design process [26].

### 4.1.2 High fidelity prototypes

High-fidelity prototypes, often computer-based, facilitate realistic user interactions such as clicking or using hand gestures in the case of apps. As design ideas mature and early solutions are assessed and chosen, representations of the intended system shift from exploratory to less speculative. High-fidelity prototypes closely resemble and function like the expected end product, ranging from mock-ups to fully working products. Unlike low-fidelity prototypes, high-fidelity prototypes allow test participants to interact with the prototype as if it were the final product. While high-fidelity prototypes are valuable for refining details through usability tests, they require significant time and effort to create and modify. Furthermore, their resemblance to a finished product may limit constructive feedback from test participants. As a result, high-fidelity prototypes are more suitable for usability tests in the later stages of the development process than in the early stages. Therefore, high-fidelity prototypes serve as a definitive reference for the engineering team responsible for coding the design.

For our prototype development, a low fidelity approach was the starting point through several paper prototype sketches. Besides paper sketches, we have also used online tool called draw.io[8] to realize the possible use case, activity diagram from a high-level point of view at the initial stage. Low fidelity protypes are time efficient, inexpensive, which helped to gain initial insights and clarifications on what feature we wanted to implement by running multiple variations and iterations. After finalizing the features from high level point of view the second step was to implementing a high-fidelity prototype which will be digital and describe each feature details with proper visualization and interaction. To implement a high-fidelity prototype, we used a prototyping tool called Figma[9]. Figma's pro-

[8]Draw.io - `https://www.draw.io/`
[9]Figma - `https://www.figma.com/`

totyping features allows to create interactive flows that explore how a user may interact with the application. Moreover, Figma is a web-based team collaborative tool which also helps to get feedback from end users during the development phase. After performing several iterations our wireframe was thoroughly tested by the end user. As a result, our final product (which has been developed and the implementation details will be discussed in later section) matches the prototype. Which emphasizes the importance of the activity of prototyping to reach a successful design decision.

## 4.2 Laws and practices

Throughout the design process and prototyping several UX laws have been followed in order to guarantee the best user interface outcomes, which are essential for effectively representing the results of visualization. They have been implemented from the start on, during the prototyping phase which has been discussed in the previous section, and also in the actual frontend designing and coding. These laws achieve several UX goals such as usability, learnability, memorability, efficiency of use and many others. For a product to be usable, these dimensions have to be considered [31].

The following are some of the most significant laws that can be easily identified from a first impression on the app:

1. Fitt's Law[10]: The time to acquire a target is a function of the distance to and size of the target. This law emphasizes on how important it is to have large target to minimize higher error rates, due to speed-accuracy trade-off. In user experience (UX) and user interface (UI) design, Fitt's law is extensively used. Our objective is to reduce the time required to acquire targets, such as clicking on buttons, pressing on tabs, and to do that, targets must be large enough to precisely select them and have space between them if numerous targets are nearby. We have applied this concept throughout the whole user interface (UI) design. For example, this law had an impact on the practice of making interactive buttons big enough to be easily acquired and have enough spaced from its surroundings to avoid confusion and to minimize the choosing process time, see Figure 10.

[10] Fitt's Law - `https://lawsofux.com/fittss-law/`

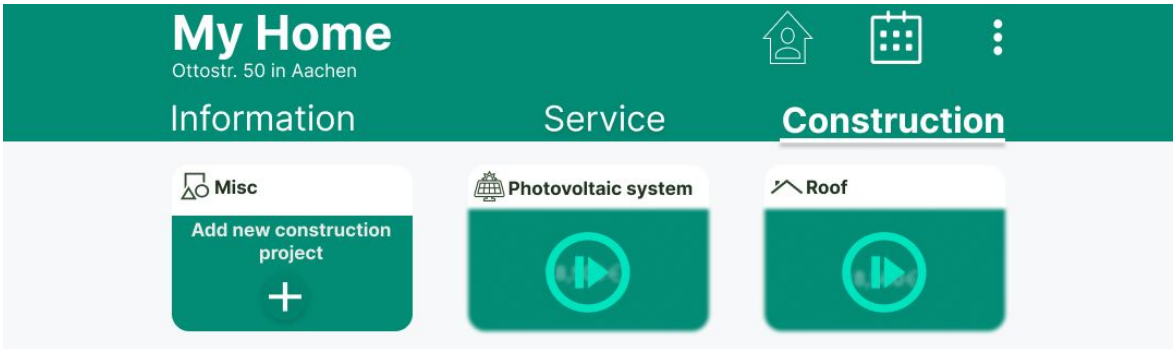


Figure 10: Screenshot highlight the strategic position of icons in order to comply with the Fitt's Law.

2. Hick's Law[11]: The time it takes to make a decision increases with the number and complexity of choices. In other words, Hick's Law states that a person will take longer to make a decision the more options they are given. Because, when app visitors are presented with too many options, the risk of information saturation / overload increases. This almost likely will have an impact on how quickly they leave and end their user experience. However, when building big lists (like a list of UX design themes or list of language selection options) Hick's Law is less significant, but it is essential when designing small lists (such as a navigation menu, or action buttons). Therefore, throughout the user interface design we have carefully applied the concept of Hick's Law to avoid overwhelming users with too many choices in order to keeping them engaged.To simplify the decision-making process during photovoltaic simulation and construction, we implemented a streamlined workflow that consists of several steps, see Figure 23.

   To further simplify the user interface, we incorporated dropdown options for each step of the process. For instance, if the user selects step-3 the app will automatically guide the user to complete the construction process without displaying any irrelevant options from other steps, see Figure 11. This approach minimizes the number of options presented to the user, which is in line with Hick's Law, as it decreases the time required to make a decision.

[11] Hick's Law - `https://lawsofux.com/hicks-law/`

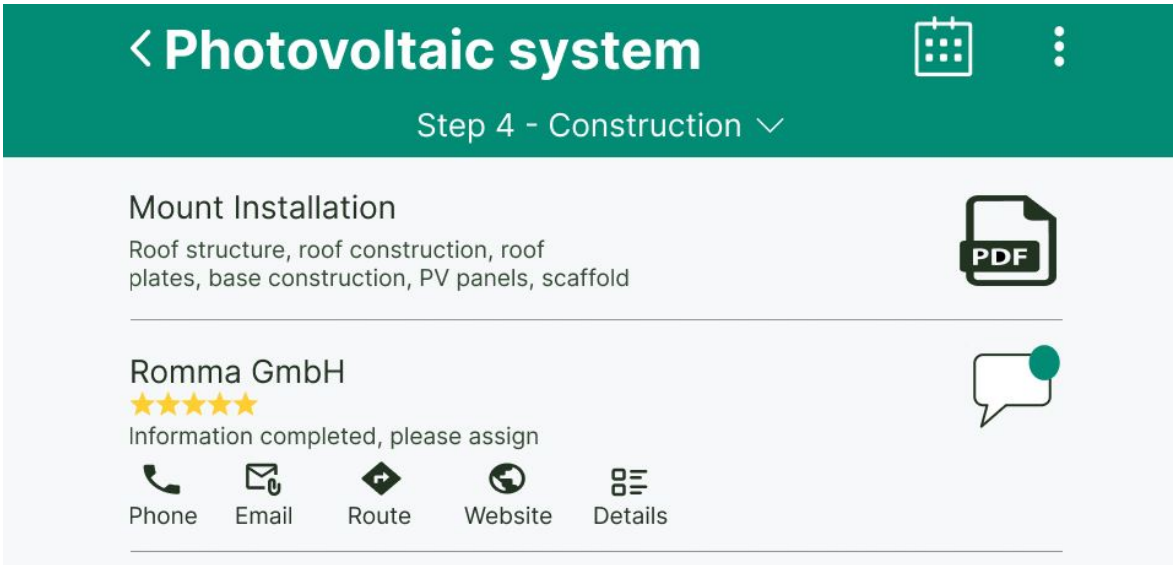


Figure 11: Screenshot shows how complex system has been design with possible minimum action options in order to comply with the Hick's Law. Moreover, to avoid further confusion and to support Hick's Law our apps guide user to perform actions step by step by disabling unnecessary action items based on user target.

3. Jakob's Law[12]: Users spend most of their time on other sites. This means that users prefer your site to work the same way as all the other sites they already know. In general, user can learn how an app works if they have accumulated experience using multiple apps. They anticipate that your app will operate in the same way when they visit it. If you try something new and unconventional on your app, your users could become lost and confused. By making use of pre-existing mental models, we may design user interfaces that let users concentrate on their goal, rather than learning new models. Mental models set an universal standards where user expect the app and its elements to function in a certain way [41]. In other word, a mental model is a way of thinking and believing that there is a common ground of how things are intended to operate. Applying a common, user-friendly design language will reduce disagreement. In our case, it was especially crucial to handle this issue because the target group of the app is a wide range of people from different ages and backgrounds. Therefore, the implementation has been carefully developed in accordance with current and accepted standards to be understandable and self-explanatory. This means that all icons used for example are once that aren't particularly new or necessarily different, rather icons that can be seen in almost all apps / websites

[12]Jakob's Law - `https://lawsofux.com/jakobs-law/`

(For instance: The save icon, which is a well-known floppy disk that can be seen in text-editing software and on many other apps, immediately denotes saving the current input's state).

4. Miller's Law[13]: The average person can only keep 7 (plus or minus 2) items in their working memory. This is due to the fact that our brain's capacity to comprehend and recall information decreases when it is presented with more information that it can handle at the time. This concept refers to the fact of cognitive load, which means the amount of information that our working memory can hold at a time [5]. Therefore, while working on user interface analysis and design we make sure not to put too much information at any point in our design, to avoid information overload which may frustrates users and ruins their experience. We have structured app content in a way that each page has a maximum of 6 elements on average 2, and when needed pages were divided into subtabs that could be easily accessed from the app header, see Figure 12. This process of grouping information is known as chunking [10]. This way user has enough elements in their working memory so that they have their full attention on what really matters.

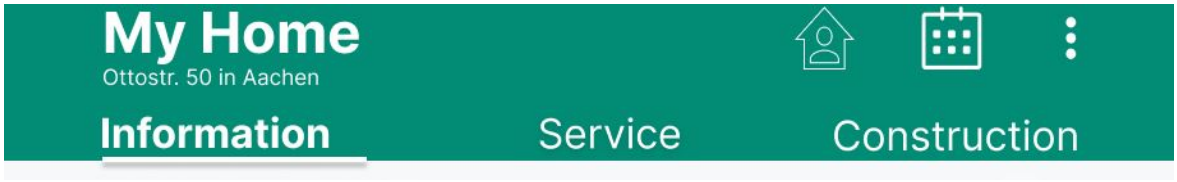


Figure 12: Screenshot demonstrating the app's structural design aimed at enhancing usability by reducing the amount of simultaneously displayed content.

5. Tesler's Law[14]: Also known as The Law of Conservation of Complexity, states that for any system there is a certain amount of complexity which cannot be reduced, and therefore, it must be dealt with either by the application or by the user. For instance, in our application we have integrated user registration and login functionality therefore, registered user can logged in from any other devices and start working where the user left off. Too restore previous activity into new device user must provide valid login credentials only then user will be redirected to the app with all it's functionality.

   In compliance with Tesler's Law, the app offers users multiple login options, including traditional email and password login and streamlined "Sign

[13]Miller's Law - `https://lawsofux.com/millers-law/`
[14]Tesler's Law - `https://lawsofux.com/teslers-law/`

in with Apple" and "Sign in with Google" options. This approach minimizes the overall complexity of the login process and provides users with a choice that suits their needs and preferences. By allowing users to create an account and log in with their email and password, the app offers a familiar and conventional login method that many users may be comfortable with. This option can be particularly useful for users who do not have an Apple or Google account or who prefer not to link their account with the app to their existing accounts.In addition, "Sign in with Apple" and "Sign in with Google" options provide users with a convenient and streamlined login method that requires minimal effort. For users who are already logged in to their Apple or Google account on their device, these options allow them to log in to the app quickly and easily, without the need to remember a separate username and password. Overall, while it may not be possible to completely eliminate the need for users to take specific steps to log in, the app login process still adheres to Tesler's Law by making the process as straightforward, streamlined, and user-friendly as possible. By minimizing the cognitive load and reducing complexity, that is intuitive and easy to use, resulting in a better user experience.

6. Von Restorff Effect[15]: The Von Restorff effect, also known as The Isolation Effect, predicts that when multiple similar objects are present, the one that differs from the rest is most likely to be remembered. The German pediatrician and psychiatrist Hedwig von Restorff, after whom the theory is called, discovered it through research. Since then, designers across various disciplines have adopted it as a guiding principle. In general, human memory tend to remember the outlying thing when presented with a list of similar items. This can be easily detected for example in photovoltaic simulation parameter roop type page list of available roof types is showing and only the selected option will be highlighted with visual elements such as background color, see figure 13.

[15]Von Restorff Effect - `https://lawsofux.com/von-restorff-effect/`

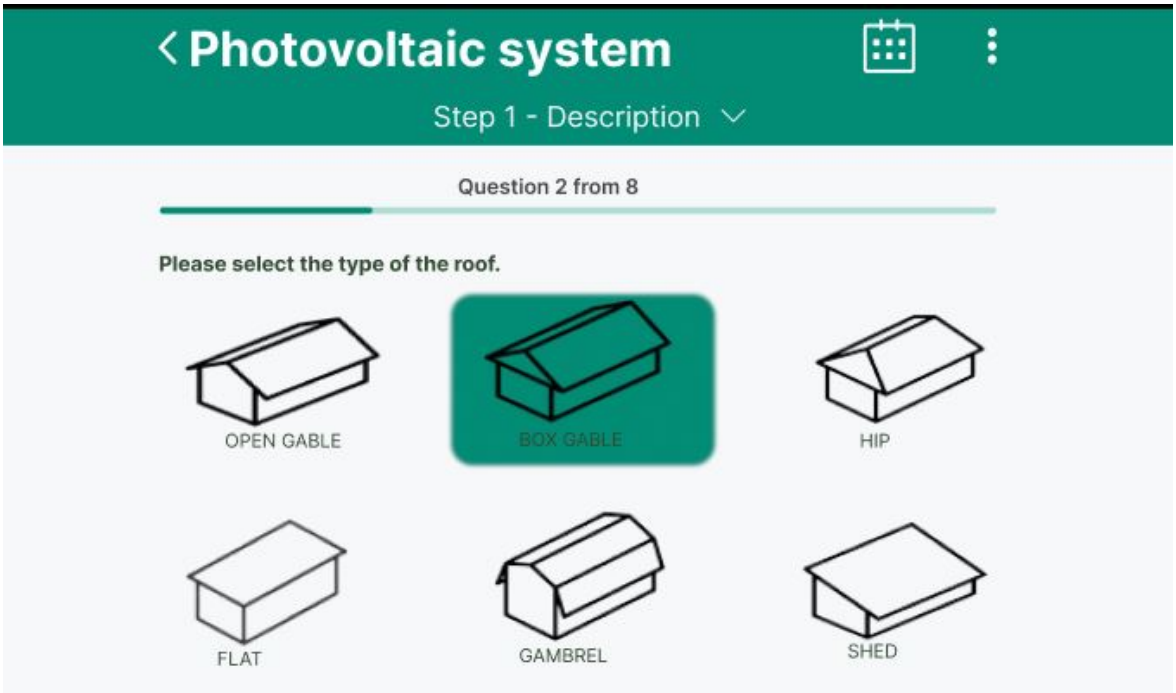


Figure 13: Screenshot shows how one item has been highlighted from a list of similar items.

7. Zeigarnik Effect[16]: People remember uncompleted or interrupted tasks better than completed tasks. In the 1920s, Bluma Wulfovna Zeigarnik conducted a research on memory in which she compared memory in relation to incomplete and complete tasks. She found that incomplete tasks are easier to remember than successful once and this is now known as the Zeigarnik effect. According to the Zeigarnik Effect, when a work is left undone, it causes mental stress and remains top-of-mind in our memories. The only thing that will relieve this tension, is closure brought on by completing the task. Zeigarnik Effect is a brilliant technique designers use to make users do certain things they wouldn't do otherwise. In the app this can easily noticed where the user's input is necessary such as photovoltaic simulation parameter setting page. This is essential and vital to fulfill the app's purpose. For instance, the use of progress bars, percentage which inform users of how close they are to complete a task by saying you have two more steps left,see Figure 14.

[16]Zeigarnik Effect - `https://lawsofux.com/zeigarnik-effect/`

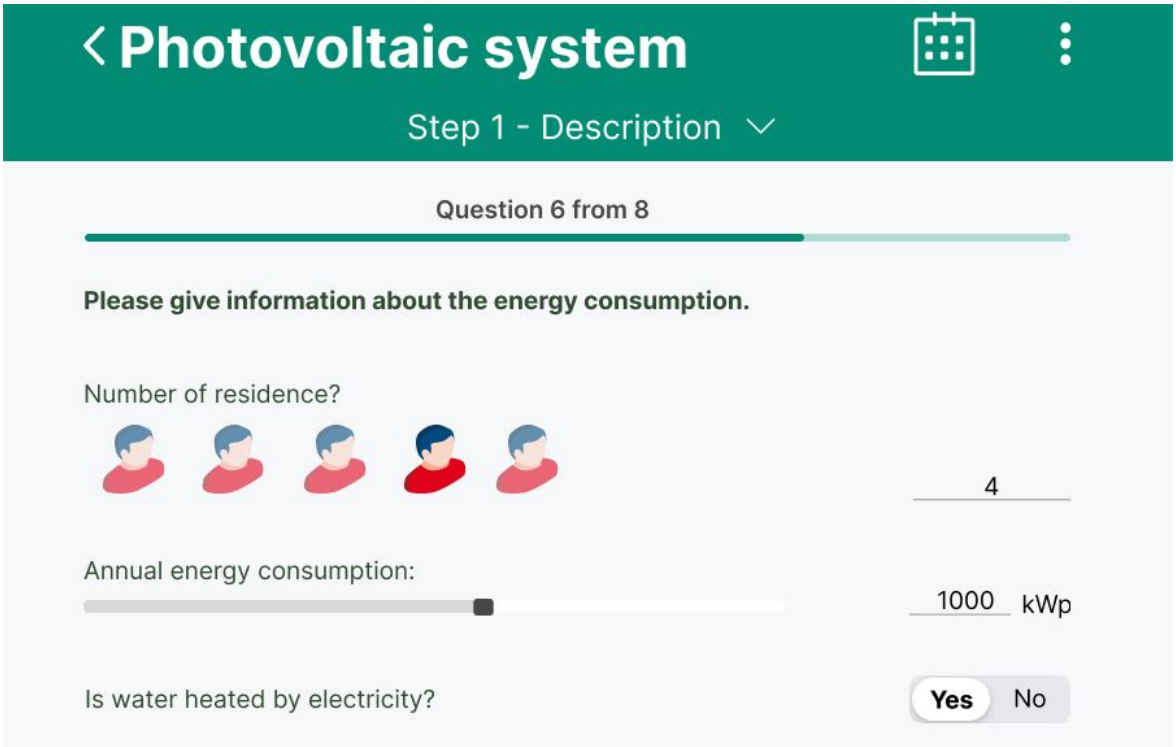


Figure 14: Screenshot shows the progress by showing the current state and final steps to complete photovoltaic simulation.

8. Law of Common Region[17]: Elements tend to be perceived into groups if they are sharing an area with a clear defined boundary. This law clearly visible across the entire app, where all contents that are related to each other are grouped together in cards or tabs. The tendency of items positioned inside a contour or on an area to be caught by it, in the sense that they are viewed as belonging to it and maybe even linked to it, rather than floating in front with no discernible means of support, plays a significant role in grouping by common region.

[17]Law of Common Region - `https://lawsofux.com/law-of-common-region/`

# 5 Software construction

The main goal of software construction is to create a high-quality product that meets the needs and expectations of the end-users that is sustainable and would support future interactions. This requires a deep understanding of the users requirements and preferences, which should be taken into consideration in every step of the development process. In our app implementation, emphasis was placed on maintaining high-quality and readable code throughout the entire application life cycle, while ensuring a smooth handover to future developers through careful documentation. The technology choices made during the development process were thoroughly researched and will be discussed in this thesis, accompanied by justifications for why certain options were preferred over others.

## 5.1 Goals

The goals of software construction must be centered around usability, and software developers must strive to create a product that fulfills the different goals of the user. This requires a deep understanding of the user's needs, preferences, and expectations. By prioritizing usability in the software construction process, developers can create a product that is not only functional but also easy to use, making it more likely to be adopted and successful in the market. Below, we discuss some of the main concept that required to flow during software construction:

1. **Simplicity**: Simplicity refers to the principle of designing and developing software systems that are easy to understand, use, and maintain. When constructing it was crucial to design the app that user-centric and meets the user's needs and preferences. To achieve this we followed a set of principles, such as designing a simple UI that is easy to navigate and understand, providing only the necessary functionality to achieve the user's goals, using a clean and easy-to-understand visual design, providing intuitive navigation to enable users to move seamlessly between different sections of the app. The choices made to achieve such simplicity has been discussed more in further detail in the User Experience design section of this paper.

2. **Accessibility**: The target group for the result of this work is a wide array of users, that encompasses different backgrounds, ages, as well as different mediums of access.It was critical to ensure that all results were easily accessible on different devices with varying screen sizes and resolutions, while maintaining the same level of transparency and consistency across all

user interfaces. The increasing number of operating systems and screen sizes, ranging from mobile phones to large desktop screens, along with varying manufacturers and their display settings presented a significant challenge. To overcome this challenge, we utilized state-of-the-art modern technologies, including responsive frameworks and single code-bases, that can adapt to the varying screen sizes and display settings. This ensured that all users could access the software with ease, regardless of the device or platform they were using.

3. **Maintainability**: Maintainability is one of the key goals of software construction. It refers to the ability of software to undergo modifications, updates, and enhancements without introducing errors or breaking the system. In other words, maintainability is the ease with which software can be maintained and improved over time, which is a critical factor in reducing the cost of software development and ensuring the longevity of software products. This goal is achieved by adhering to best practices, employing efficient coding techniques,using tools, frameworks and the latest standards to ensure that the code-base can be easily understood and updated by multiple developers in the future. In addition, technologies with well-documented resources and large, active communities were chosen to ensure that any issues that do arise can be addressed and resolved in a timely and effective manner, thus ensuring the overall maintainability of the App.

4. **Compatibility**: Compatibility in software construction refers to the ability of different components or systems to work together seamlessly. As this work is merely a frontend visualization, In the case of integrating complex algorithms and computations with the frontend visualization, it is important to ensure that the different components used in the existing and future development are compatible with the app implementation. This includes ensuring that the new code adheres to the same coding standards and best practices as the existing code, as well as ensuring that there are no conflicts or compatibility issues between the different technologies used. Additionally, it is important to ensure that the data structures used are compatible and consistent between the different components, allowing for easy communication and transfer of data between them. By ensuring compatibility between the different components, future development and integration can be done smoothly and without major issues, ultimately leading to a more maintainable and robust software system. In our application development, we used modular design by dividing the different components and building blocks into separate modules, each with its own specific function. This approach allows for greater flexibility and scalability, as well as

easier maintenance and future updates. For offline computations different API endpoints we used JSON as the primary data interchange medium as it is easy to parse and generate using built-in libraries in most programming languages. This makes it very convenient for developers to work with and integrate into their frontend and backend development.

5. **Reusability**: Reusability is an important goal in software construction, especially in large projects where code can easily become messy and difficult to manage. The concept of reusability involves the use of previously coded components, either custom-developed or integrated from open-source libraries, to minimize code length and duplication, support modularity, and improve code readability. This approach also saves time and effort by enabling developers to focus on more important aspects of the project rather than repeating certain tasks. To achieve reusability, we used techniques such as creating libraries of reusable code, designing modular components that can be easily integrated into other parts of the codebase, and utilizing open-source resources whenever possible. By implementing these strategies, generally developers can improve the efficiency and quality of their work, and ensure that their codebase remains organized and manageable even as the project continues to grow and evolve over time.

## 5.2 Architecture

In this section, we discuss the architecture of the entire app and how the different flows of data behave and which services interact with each other, see Figure 15.

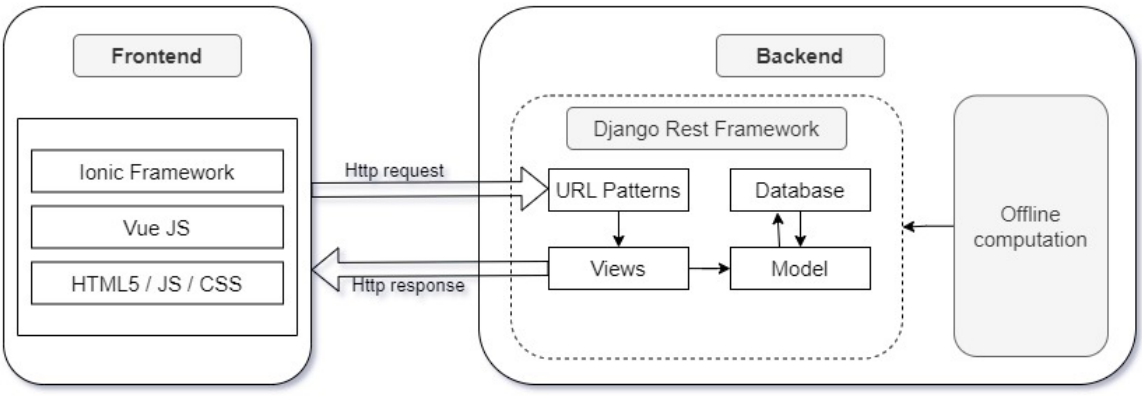


Figure 15: Diagram showing the data flow between the various abstract layers.

In our application architecture there are two main abstract layers: Frontend and Backend.

1. **Frontend**: The frontend of the app is responsible for handling the user interface and user interaction aspects of the app, providing a seamless experience for users. The frontend interacts with the backend of the app, which is developed using Python and the Django REST framework. Through rest API services, the frontend is able to communicate with the backend, enabling it to receive and store data to perform calculations based on user inputs. The frontend has been developed with a user-centered approach, with a focus on usability and ensuring that the app meets the different goals of the user. This has been achieved through the use of responsive design principles and best practices, resulting in a highly efficient and dynamic workflow, rapid and high-quality results, and a clean and readable codebase.

2. **Backend**: The backend layer is responsible for handling the heavy lifting of the app, which includes data processing, computation, and storage. In our application, we have used Python for the offline calculation, which means that the app can function even without an internet connection. The outcomes of these calculations are stored in JSON files that are later stored in the database managed by the backend, which is then accessed by the frontend to display the calculations meaningfully to end-users. Python is a popular programming language that is well-suited for data processing and analysis. We also used the Django REST framework to communicate with the frontend through REST API services. Django is a web framework that follows the Model-View-Controller (MVC) architectural pattern, which helps in organizing the codebase and separating the concerns. REST stands for Representational State Transfer, which is an architectural style for building web services. RESTful APIs are designed to be scalable, easy to maintain, and platform-independent. With Django REST framework, we can create a scalable and robust API that can handle various types of requests and responses. This allows us to easily integrate the backend with the frontend, as the API provides a standardized interface for communication between the two parts. Additionally, the use of RESTful API services allows for easier testing and debugging, as well as simplifies the process of adding new features or modifying existing ones.

Below we will further explain each level in detail, how it's operating behind the scenes, which main tasks does it have, as well as the data, flows and the endpoints it communicates with.

Starting with the frontend, this layer handles all the visual representations of the work. In our hybrid app development this has been done through pre-built and custom-built reusable Vue components that are placed and grouped alongside the relevant data to display certain views for end-users. The frontend level

also provides users with all the navigation capabilities to access the different pages of the app, this is handled using Vue's highly renowned routing library.

Aside from the styling and design of the app as a whole, the app mainly revolves around the data that are displayed within these design elements. And these data come from mainly two sources: a local JSON file and the Backend endpoints. The local JSON file includes all the hardcoded content such as titles, headers, texts, button texts, etc.Using a local JSON file to store hardcoded content can offer several benefits for software construction. First, it separates the content from the code, making it easier to maintain and update the content without affecting the code. This makes it simpler for content creators to update or change content without having to involve developers. Second, it can improve the scalability of the application as it allows for easy updates to the content without having to redeploy the entire application. Third, it can help in localization efforts as it enables easy translation of the content into different languages. Finally, it can reduce load times by allowing the content to be preloaded and cached, resulting in a smoother and faster user experience.

Below is an excerpt from the main JSON file that was just briefly explained:

```
{
    "en": {
      "label": "English",
      "headers": { ... },
      "splash": {
        "app_title": "Efficient Buildings",
        "loadingText": "Loading current information"
      },
      "onboarding": { ... },
      "login_signup": { ... },
      "new_property": { ... },
      "dashboard": { ... },
      "settings": { ... },
      "CraftsmanInfo": { ... },
      "myDashboard": { ... },
      "myHouse": { ... },
      "myInfo": { ... },
      "about": { ... },
      "craftsmen_management": { ... },
      "newPV": { ... }
    },
    "de": { ... }
}
```

The second source is from the backend REST API through the different endpoints it offers, these data are the essence of the visualizations provided on the app, they are the final results of the algorithms of the level of the offline calculations.

Below is an excerpt from the Django url configuration that interacts through frontend route setting:

```
from django.db import router
from django.urls import path
from django.urls.conf import include
from . import views
from rest_framework import routers
router = routers.SimpleRouter()
urlpatterns = [
    path('onboarding-splashscreen/', views.onboardingSplashscreen),
    path('onboarding-pages/', views.onboardingPages),
    path('new-property/', views.newproperty),
    path('settings/', views.Settings),
    path('user-login/', views.userLogin),
    path('user-registration/', views.userRegistration),
    path('dashboard', views.dashboard),
    path('generate-pv-simulation',
    views.generatePVSimulation),
    path('update-pv-simulation/<int:pk>/',
    views.updatePVSimulation),
    path('pv-simulation-report',
    views.pvFinancialReport),
    path('generate-list-of-tasks',
    views.generateListOfTasks),
    path('send-invitation', views.sendInvitationToCraftsman),
    path('assign-task-to-craftsman',
    views.assignTaskToCraftsman),
    path('update-task/<int:pk>/', views.updateTaskToCraftsman),
    path('delete-task/<int:pk>/', views.deleteTaskToCraftsman),
    path('chat-history', views.chatHistoryWithCraftsman),
    path('give-rating-to-craftsman',
    views.giveRatingToCraftsman),
    path('', include(router.urls)),
]
```

## 5.3 Technologies used

Below, we provide an overview of the technology stack employed in the development process. We also provide justifications and arguments for the selection of specific technologies and the benefits they offer. The subsections cover topics such as hybrid application development using Ionic and Capacitor, frontend technologies utilizing VueJS, data and state management through Pinia, and styling libraries including Tailwind. By carefully choosing and integrating these technologies, the app was designed to be interactive, sustainable, and easily maintainable in the long term.

### 5.3.1 Hybrid application (Ionic + Capacitor)

Hybrid applications are essentially web apps that use a lightweight native application container that enables it to use native platform capabilities and device hardware (such as camera, accelerometer, GPS, calendar, push notifications, etc.) that a web application cannot, which will come in handy for the visualization of our work. For example, suggesting a ZIP code using the device GPS or during house setup gives users options to use their device camera to upload a house profile picture. Hybrid applications are a blend of both native and web applications, and in order to give hybrid applications cross-platform capabilities, they use widely used web technologies such as HTML, CSS, and JavaScript, see Figure 18. A hybrid application is device independent and can be run on different operating systems like Android and iOS or a device browser as a Progressive Web App (PWA). Another aspect is the capability to provide a consistent and similar user experience across platforms, even when a user switches between several devices or browsers. Mainly, the native application container uses its embedded browser to publish the app rather than being published within the user's browsers, and the whole operation is executed in the backend across platforms which is essentially invisible to the user. For example, the iOS operating system uses the WKWebView element to display hybrid applications, whereas on Android, they use the WebView element.

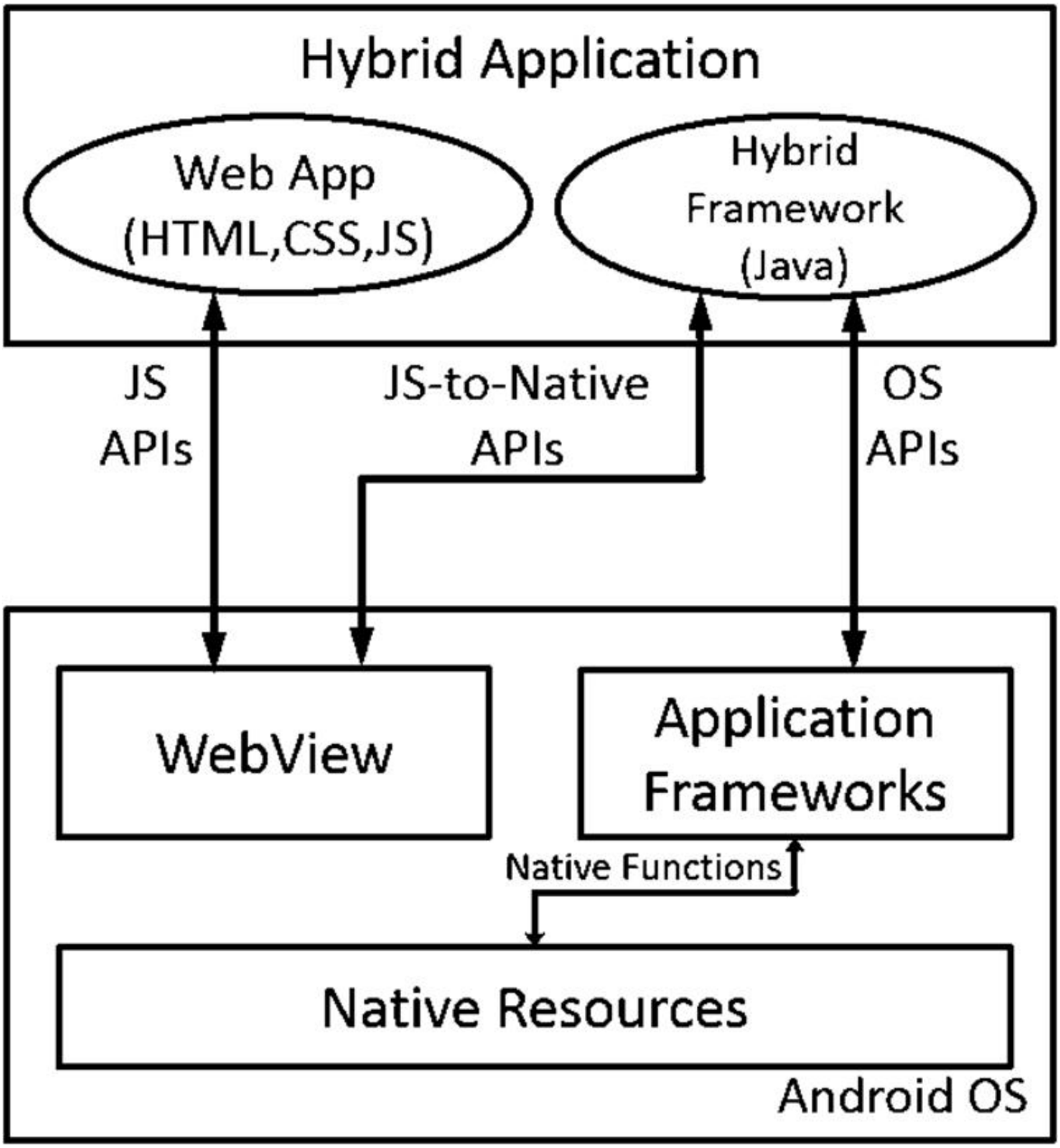


Figure 16: Hybrid application architecture [28].

There are many frameworks like React native[18] ,Ionic, Flutter[19], Xamarin[20] that supports hybrid applications development. For our work, we choose Ionic [21] to build a hybrid application. Ionic is an open-source software development kit (SDK) for creating hybrid mobile apps, and it can be used with modern frontend

[18]React Native - `https://reactnative.dev/`
[19]Flutter - `https://flutter.dev/`
[20]Xamarin - `https://github.com/xamarin`
[21]Ionic - Cross platform mobile app development - `https://ionicframework.com/`

frameworks like Angular, React, and Vue (more on that in the next section) that support building user interface design. It has a robust framework that aids in building Progressive Web Apps. Ionic was released to the market in 2013 and since has been used to build more than 5 million apps. The Ionic framework offers a vibrant community that is always expanding, making it simpler for new developers to begin creating their first applications.

Then, using a program like Ionic's Capacitor [22] or Apache Cordova [23] (also known as PhoneGap), application code is injected inside a native application wrapper. In order to load your web application, these solutions produce a native shell application that is just the platform's WebView component. This enables you to produce legitimate native programs that can be uploaded to each of the platform's app stores for further usage.

In our case, we use Capacitor. Capacitor has a plugin system that allows to extend beyond the limitations of the browser and obtain full native device access. Hardware features, such as the camera, touchID, push notifications, instant updates, GPS, and offline data sync, for instance. Capacitor achieves this by compiling web technologies such as HTML, CSS, and JavaScript into native code. In addition, ionic accelerate app development by wrapping the Capacitor build tool for distribution to several platforms. This compatibility makes the creation of multi and cross-platform apps easier. Because native code for each platform differs, creating apps for many platforms has historically been challenging and time-consuming. For example, developers have to create separate versions of an app for Android and iOS.

The advantages of using hybrid applications include the ability to fully utilize web-based services regardless of whether the device is connected to the internet or not, in contrast to regular web applications that require internet. Unlike traditional mobile app development, which requires separate codebases for each operating system, they can operate on a variety of platforms by creating and maintaining only one codebase. Ionic is therefore, platform independent. This fits in exceptionally well with the limited human resources and time that we have available for this task. Compared to the creation of native apps, this results in shorter development and construction times. When compared to creating two apps for two separate platforms, this also results in significant cost savings in general. Additionally, it is easier to issue patches and updates, which fits our requirements well, as frequent modifications to the language preference and craftsman selection option are required to increase the app's reliability. These updates do not need to get approval from app store. Whenever there are changes

[22] Capacitor - build cross platform apps with the web - `https://capacitorjs.com/`
[23] Apache Cordova - `https://cordova.apache.org/`

the necessary changes will be fetched and shown immediately. Therefore, all the time the users are benefited from the latest version of the app with the newest updates. In addition, hybrid applications allow the use of the local storage on the device, which will be covered in more detail in a later section, and how this helps in gaining database functionalities of a web app while being offline.

The drawbacks include potential differences in the app's functionality or appearance on two distinct platforms or devices due to focusing on one platform during development and testing. This might lead to unresolved issues that might show on an untested platform. A measure to counter this disadvantage is to increase the testing of the application on a wide range of devices to ensure proper and smooth operation. Additionally, hybrid apps add an extra layer between the source code and the target platform. Therefore, hybrid apps may perform slightly slower than native or web versions of the same app where the source code interacts directly with the platform. In general, hybrid apps have issues rendering high quality 3D graphics. To resolve this issue, one might need to integrate third party tools to enhance the 3D graphics experience, which may need higher efforts compared to a native approach. However, in our apps we rarely need high quality graphics representation.

### 5.3.2 Front-end technologies (VueJS)

A pre-written program structure on top of which you may create is known as a software framework. A framework is meant to help you build applications quicker by addressing common development problems and often starts you out with Boilerplate code (covering methods that are reused by most apps) and a directory structure (often following a design philosophy).

Frontend frameworks are responsible for building the part of a web application that is in direct contact with the end user, in contrast to the back-end where it is more about the behind-the-scenes functionalities. In another word, A front-end framework refers to a software platform or tool used for the visualization of back-end instructions in the user interface. The idea behind front-end frameworks is bundled code developed by other developers that can be used in apps to speed up and ease the development process, like any other framework in software engineering.

Using front-end frameworks eases the workflow for developers, speeds up development and improves the robustness and stability of the application being built. Building entire applications with just JavaScript, commonly known as VanillaJS, can be very tedious and recourse inefficient because there will be a lot of repetitiveness and duplication. Front-end frameworks offer the strength of

reusability and dynamism, as well as robust data management, which is the case for our work.

For our work, we use VueJS [24], which is an open-source lightweight front-end JavaScript framework based on MVVM mode [48] in web application to build user interfaces and single-page applications.

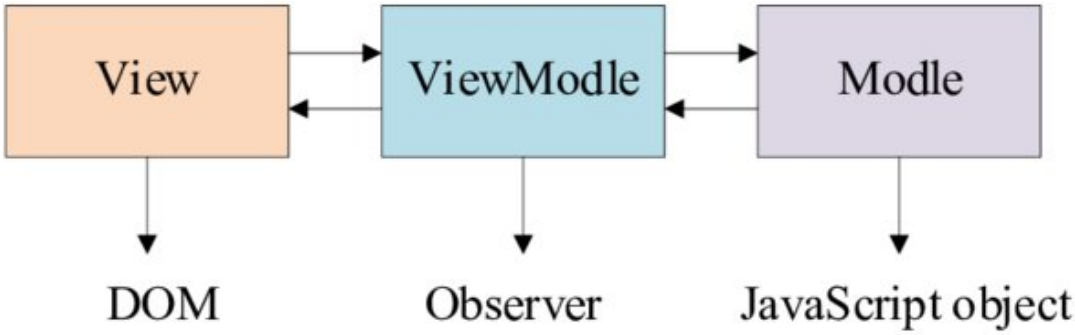


Figure 17: MVVM model architecture diagram.

Another key idea in Vue is the component system, which extends HTML elements to encapsulate code, make them reusable across contexts, and render them as much as necessary with dynamic content (see Figure 19).With the aid of this straightforward method, Vue instances can be declaratively reused and composed in a manner like to that of Web Components without the requirement for the newest browsers or heavy poly fills. Therefore, a highly decoupled and maintainable codebase is produced by dividing an application into smaller components. In our app an example to explain this concept could be “Description” component which has been used in different occasions for instance we have integrated this feature on “Report a bug”, “Assign task to craftsman modal”, and “Task assigned privately modal” page. The component in this case is an empty “Text area” html tag that has placeholders for the different data, and the component is duplicated and submitted with the respective data for a certain user. Such information is exchanged between components using Vue’s props (Props are used to pass data from the parent components down to the child components – it follows a unidirectional flow.) and dynamically displayed in the HTML declarations of the components using JSX, a JavaScript grammar extension that permits HTML coexistence with JavaScript code. This leads us to Vue’s HTML-based template syntax, which utilizes the virtual DOM render capabilities of Vue to enable binding the rendered DOM (a programming interface for online pages) to the component’s data. With the use of a virtual Document Object Model, Vue can render web page elements in-memory before updating the browser, improving efficiency, and allowing for dynamic data changes. In another word, VueJS

[24]Vue is a JavaScript framework for building user interfaces - `https://vuejs.org/`

automatically tracks JavaScript state changes and efficiently updates the DOM when changes happen.

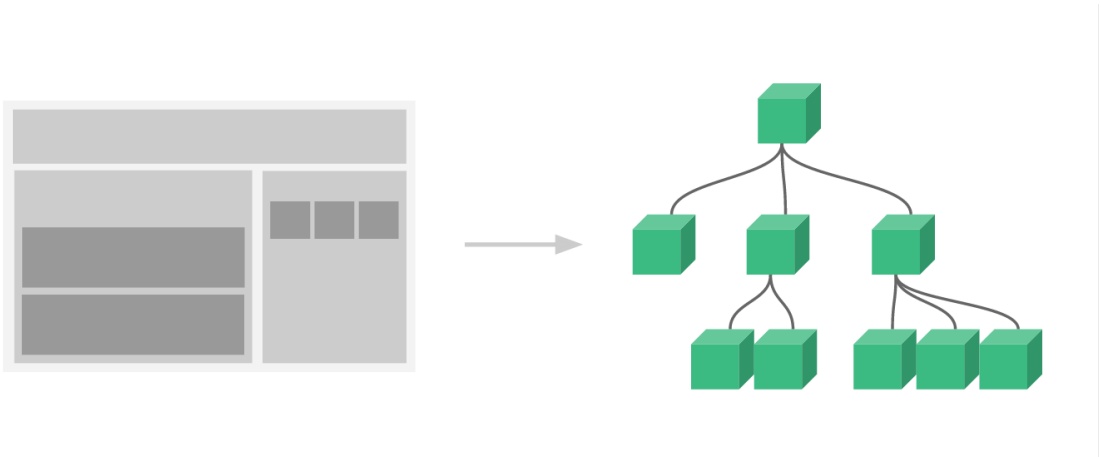

Figure 18: Apps in VueJS consist of different components within each other.

AngularJS [25] and ReactJS [26] are two notable VueJS substitutes. According to the number of stars on GitHub, VueJS has the fastest-rising developer popularity. This is advantageous when utilizing frameworks since the architecture and performance of the framework are always being developed and improved. In addition to the large community that present great support through contributions of reusable components for example. In addition to the large community that provides excellent assistance, for instance through donations of reusable components. In comparison to its competitors, Vue is also far more configurable and allows combining UI and behavior of components from within a script. Moreover, Unlike its peers, there is no need to be forced to follow the framework's way of doing things, which provides high flexibility and agility.

### 5.3.3 Data and state management (Pinia)

The app heavily relies on data, and all of the visualizations are determined by the data that was computed by the offline calculations and provided by the backend. Additionally, we had to build up a storage to avoid repeatedly accessing the data in order to give customers a reliable and quick app experience. This would also enable the app to run without an internet connection.

In our implementation we choose Pinia [27] over Vuex [28] store as it needs less

[25] AngularJS is a structural framework for dynamic web apps - `https://angularjs.org/`
[26] A JavaScript library for building user interfaces - `https://reactjs.org/`
[27] Pinia is the new default store library for Vue - `https://pinia.vuejs.org/`
[28] Vuex is a state management pattern + library for Vue.js applications - `https://vuex.vuejs.org/`

bootstrapping and setup than Vuex which will save our development time. In Vuex there are four main components whereas in Pinia it has three components which is easier to understand and offer simpler integration, see figure 2. Moreover, Pinia is now also the recommended choice by Vue and is now an official part of the whole Vue ecosystem, maintained and developed by core members of the Vue team.

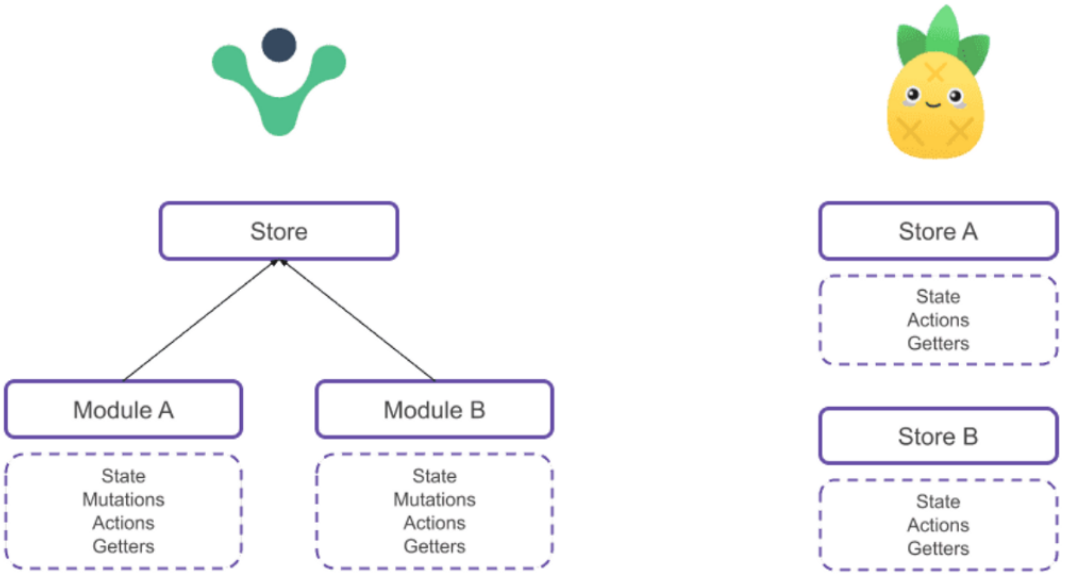


Figure 19: Shows how Vuex vs Pinia–stores works.

Pinia is a lightweight state management paradigm and official framework for Vue.js application. Pinia is reactive and propagates events when the underlying values change, while it also offers total choice in terms of what data to keep and how to store it. It serves as a central repository for all the components of an application, while rules make sure that state changes may only be made in predictable ways and works as the single origin of the application, see Figure 19. For instance, if the state information is altered, all components will be made aware of the change.

Using such library allows us to retain the state of the data and user's inputs during their usage of the app, which with the growing complexity of data-driven apps proves as a reliable method of retrieving data, correctly mutating them, and having them easily accessible from different components.

### 5.3.4 Styling libraries (Tailwind)

One integral part of eye-catching apps is styling. Modern style strategies could be an excellent option to develop contemporary designs that meet consumer expectations. To do this and increase the effectiveness of the front-end development process, CSS-style libraries have been used. These libraries provide predefined classes that are reusable and support code consistency. Additionally, they provide clear code that is understandable and scalable in the future. For our work, we used Tailwind [29], a utility-first CSS framework stuffed with classes that can be composed to build user interface design directly in the HTML files with little need to resort to CSS files.

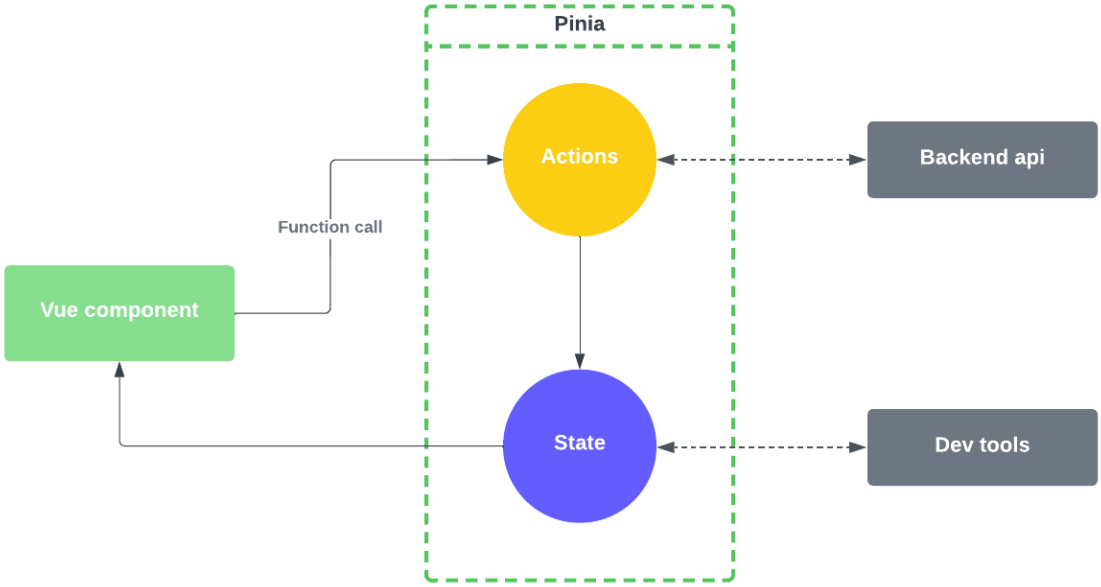


Figure 20: Pinia features integrated into an existing VueJS app with components and a backend API that provides data.

This not only results in extremely rapid progress but also a more minimal code structure. Another advantage of Tailwind is the naming conventions that are unified and pre-defined as part of the documentation which enables easy access to styles that need certain changes. One more advantage of the concept of utility classes that are being utilized by this library is that it helps to work within the constraints of a system instead of littering style sheets with arbitrary values that are hard coded. This makes it easy to be consistent with spacing, typography, shadows, and everything else that makes up a well-engineered design system.

[29] Rapidly build modern websites without ever leaving your HTML - `https://tailwindcss.com/`

For example, the shadow-lg class from tailwind defined a certain shadow styling that has been used all over the app to maintain a consistent shadow for all elements with this single keyword.Tailwind also is very high in performance where it makes use of CSS Purging to automatically remove all unused CSS when building for production, which means that the final CSS bundle is the smallest it could be. Another aspect of using styling libraries, in general, is the utilities they bring when it comes to responsive design, which in our case was a high priority since the entire single code base should be made accessible through all different types of screens. Responsive design helps enrich the user experience through creating apps that can adapt layout and content to viewing contexts across a spectrum of digital devices.

Traditionally, developers would have to wrestle with numerous complex media queries in CSS files, but with Tailwind, this can be done directly from within the HTML, which was very helpful during the development process as everything in terms of page structure was visible and helped have a better overview when making screen responsiveness decisions.

Last but not least, tailwind has a rapidly growing community that is driven by the idea of modern and flexible front-end development, which helps keep this library updated and supported. This also makes it more likely for creative UI components to emerge from open source contributors which will help accelerate development even more through usability.

# 6 App solution

The outcome of the app implementation will be presented in this section of the report, including detailed descriptions of the main pages and their functionalities, along with corresponding screenshots. Each figure displayed will consist of two versions, the mobile version and the desktop version because as mentioned before, responsiveness was put in mind from the beginning on, and we would like to show how this is reflected on the end result.

## 6.1 Initial points of interaction

The following screens are the users' first impression of the app; therefore, simplicity and minimalistic delivery of content was ensured.

### 6.1.1 Splash screen

The startup of the app is through this screen, as demonstrated in 21, is where the loading of all relevant components happens, as well as any retrieval of new updates from the backend. This is done by comparing the user's local current version with the version on the backend, and downloading only if there is an update.

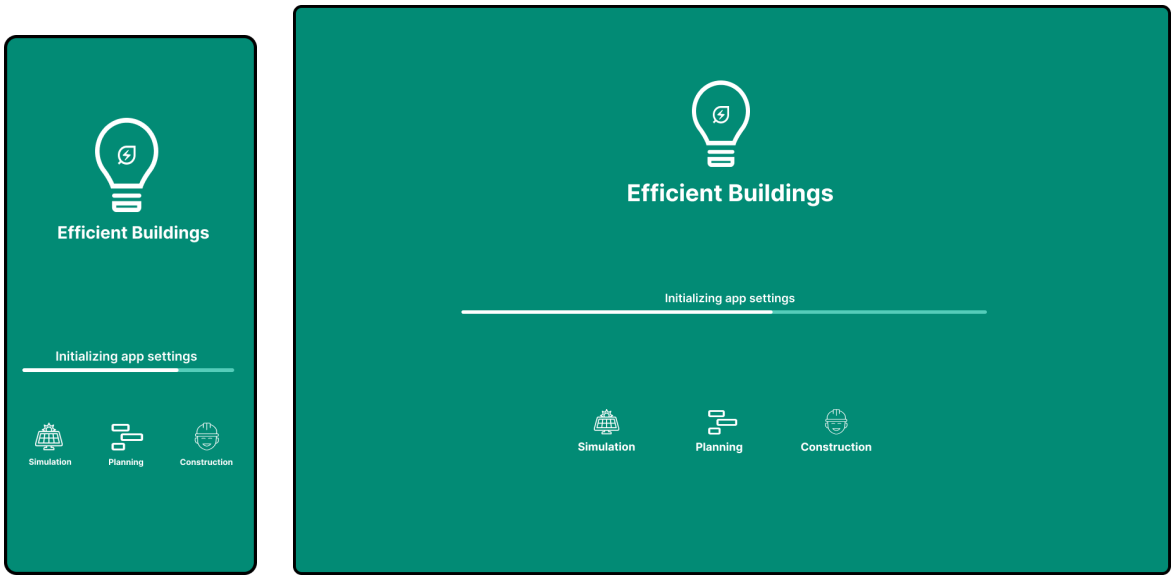


Figure 21: Splash screen with a loading progress bar.

### 6.1.2 Onboarding

This component is displayed only for the first-time users where they briefly introduced to what the app has to offer, a flag is set once they proceed so that it's

not displayed again.

### 6.1.3 User registration and login

After onboarding process, the user will be asked to setup an account by entering a unique email and password. The main objective of registration is to establish a secure account that solely the user can access, and to enable the application to retain their preferences and settings.

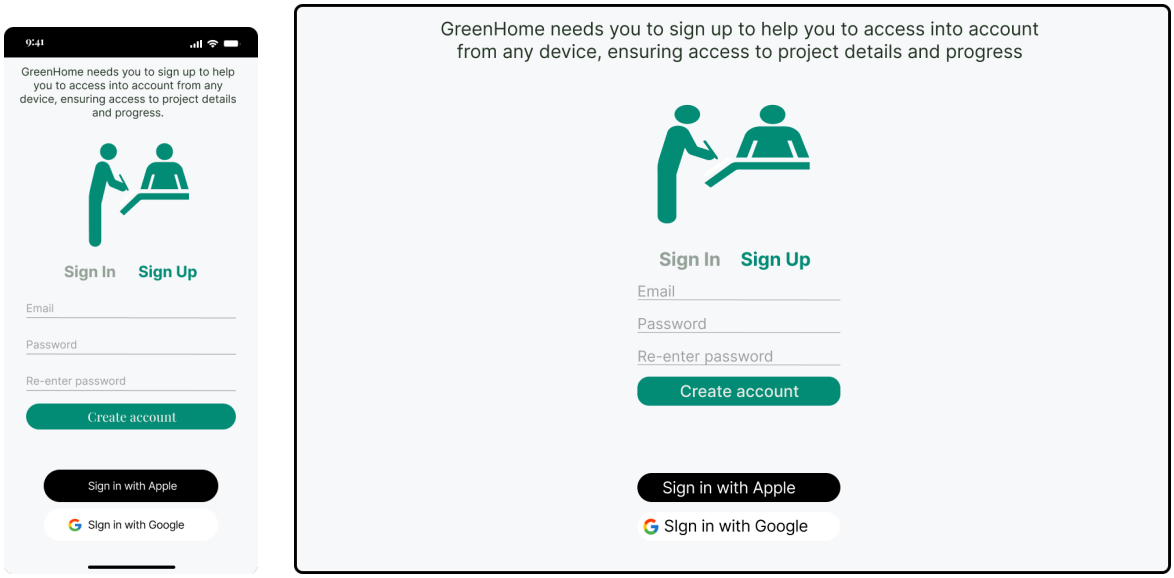


Figure 22: Registration page for the first time users.

After registering, the user can access their account from any device by entering their email address and password. This grants them access to their profile, settings, and any saved data within the app.

## 6.2 Dashboard

The Dashboard of the PV system planning, construction, and visualization app provides the user with all the essential information and services they need to plan, track, and complete their PV system installation project.

## 6.3 Photovoltaic system

The simulation of a PV system is an important step in the process of designing and constructing a photovoltaic system. To successfully simulate and construct a PV system, the user needs to complete five steps. Each step has been carefully designed to ensure a smooth user experience.

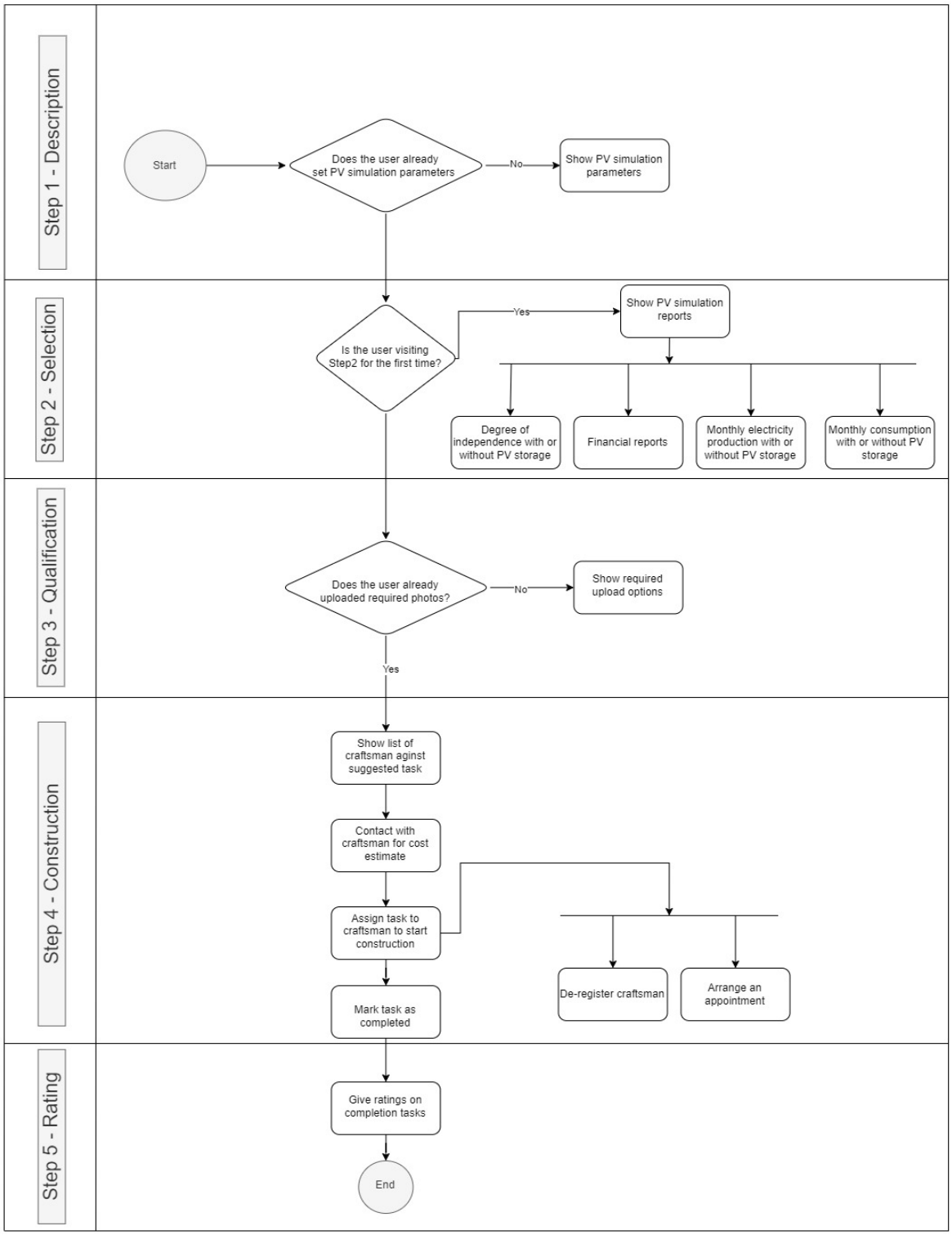


Figure 23: Screenshot describes process flow of different steps.

### 6.3.1 Step 1 - Description

The first step, Description, involves simulating the PV system by answering a series of questions to provide information about their house and electricity usage. This information is crucial for accurately simulating a PV system that fits the user's needs and the specifications of their house.

1. Is the house under monumental protection: This is important to know because if a house is under protection, the installation of PV panels may be restricted or require special permission from authorities. Knowing this beforehand can save the user time and money in obtaining the necessary permissions.

2. Type of roof: Different types of roofs have different load-bearing capacities, angles, and directions. Knowing the type of roof is important for determining the optimal placement and orientation of the PV panels. If the roof type is FLAT, is the load capacity at least 20 kg/m2 or if the roof type is gable, does the rafter have a width of at least 5.1cm? - This is important to know because PV panels are heavy and require a strong and stable roof to support them. Knowing the load-bearing capacity of a flat roof or the width of the rafters in a gable roof is important for determining if the roof is suitable for a PV system installation.

3. Slope angle and roof direction: These factors are important for determining the optimal angle and orientation of the PV panels to maximize energy production. For slope angle user can any option from the given list. However, user can also specify the precise slope angle. For roof direction, 3D models of the building will be shown. By default roof direction will be set by analyzing Cadastral data however user have the option to modify for experient with different settings.

4. Drawing roof area: Knowing the roof area and any obstacles on the roof can help determine the number of PV panels needed and their optimal placement. In germany, NRW state (LANUV) shared cadastral data which provides a detail information of the roof surface. The dataset can recognize obstacle for PV panels such as chimney, roof vents and removed from the surfaces. Shadows are also cut from the surfaces of the rooftop. However, user can also specify the available surface area and obstacle.

5. Number of residents and annual energy consumption: These factors are important for determining the size of the PV system needed to meet the energy demands of the household. There are other important factors which can also impact on annual energy consumption like Is water heated by electricity, Is the user using an electric vehicle charger, Is the user using an

electric heat pump for heating has been grouped together for better user experience.

6. Select PV panel module type: There are different types of PV panels available, each with their own efficiency and cost. Knowing the type of PV panel can help determine the overall cost and energy production of the PV system.

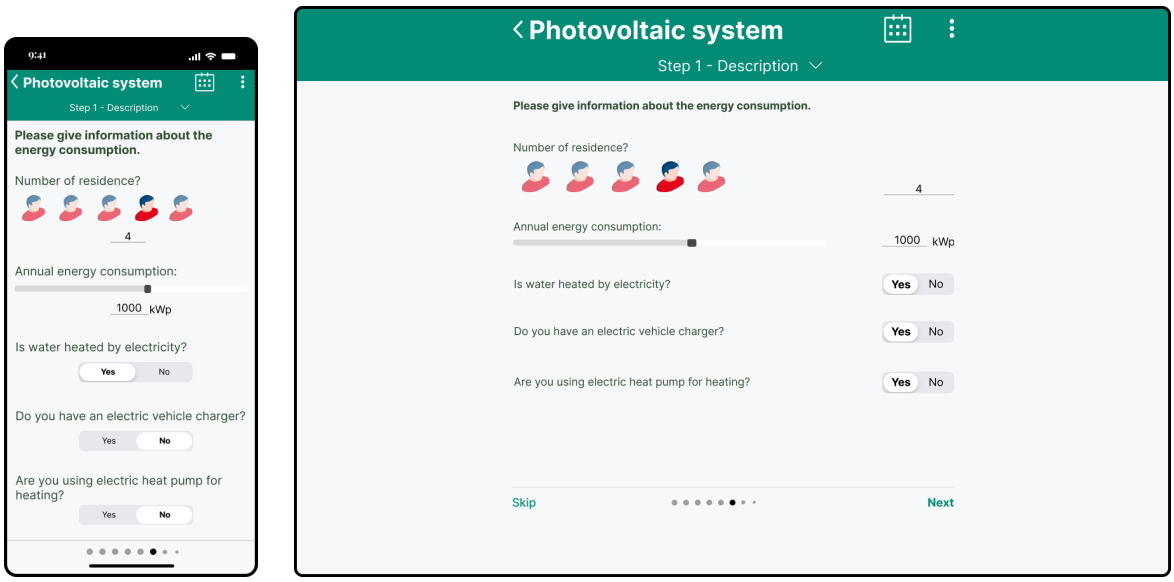


Figure 24: Example of PV simulation questions.

In summary, each question in the Description step is important for determining the feasibility and optimal design of a PV system installation for a specific house. Answering these questions can help the user make informed decisions about the size, placement, and type of PV system needed to meet their energy demands.

### 6.3.2 Step 2 - Selection

After completing the PV simulation in step 1, the app will direct the user to the selection step, where the user can see different reports for the PV system. These reports provide important information for the user to make informed decisions regarding the installation of a PV system.

The first report which will be shown by default and that is the degree of independence, which shows the percentage of electricity the PV system can provide for the household. This report is important as it shows how much the household can rely on its own generated electricity and reduce its dependence on the power grid.

The second report is the own consumption, which shows how much of the generated electricity is consumed by the household and how much is fed into the grid. This report is important as it shows how much the household can benefit from the generated electricity and optimize its usage.

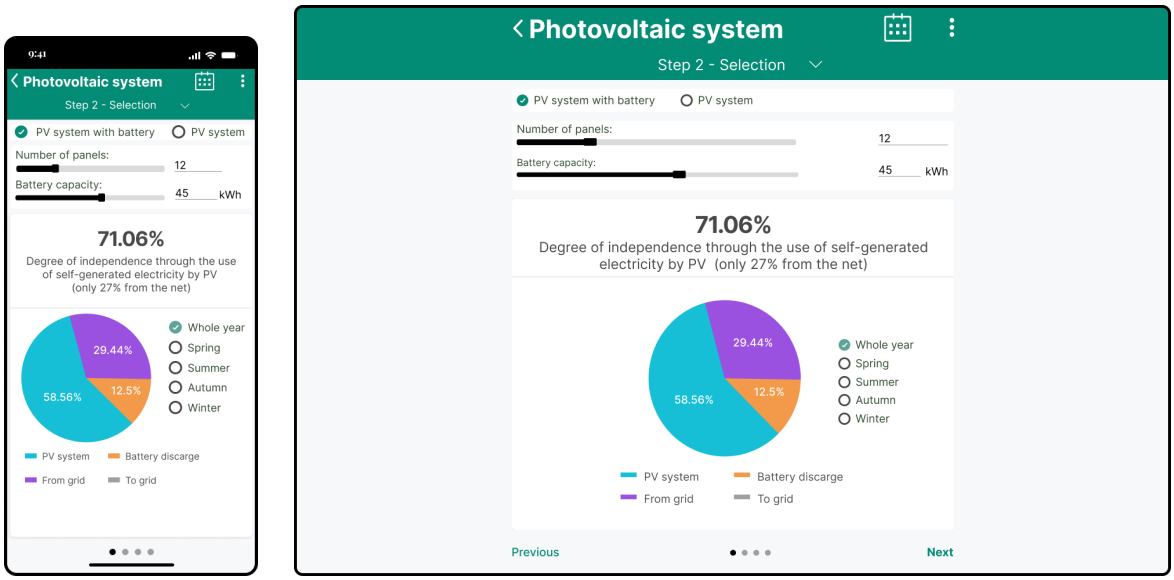


Figure 25: Step 2 degree of independence report.

The third report shows information about economics like return on investment and yearly electricity cost with PV and without PV system. Return on investment (ROI) refers to the financial benefit an house owner can gain from investing in PV systems. The ROI is calculated by comparing the total costs of installing and maintaining a PV system with the expected savings from generating and using solar electricity. In Germany, the ROI of a PV system depends on various factors, such as the size of the system, location, available subsidies, and electricity tariffs. Generally, PV systems in Germany have a payback period of 7-10 years, and investors can expect a return of 4-6 percents annually over the system's lifetime, which is typically around 20-25 years. Yearly electricity cost with PV and without PV is an important metric to consider when deciding whether to invest in a PV system. In Germany, electricity prices have been increasing steadily over the years, making PV systems more financially attractive. By installing a PV system, homeowners can generate their own electricity and reduce their reliance on grid electricity, resulting in significant cost savings in the long run. The yearly electricity cost with PV takes into account the amount of electricity generated by the PV system and the electricity consumed from the grid, while the yearly electricity cost without PV only considers grid electricity consumption. By comparing these two costs, homeowners can make an informed decision about the financial viability of installing a PV system.

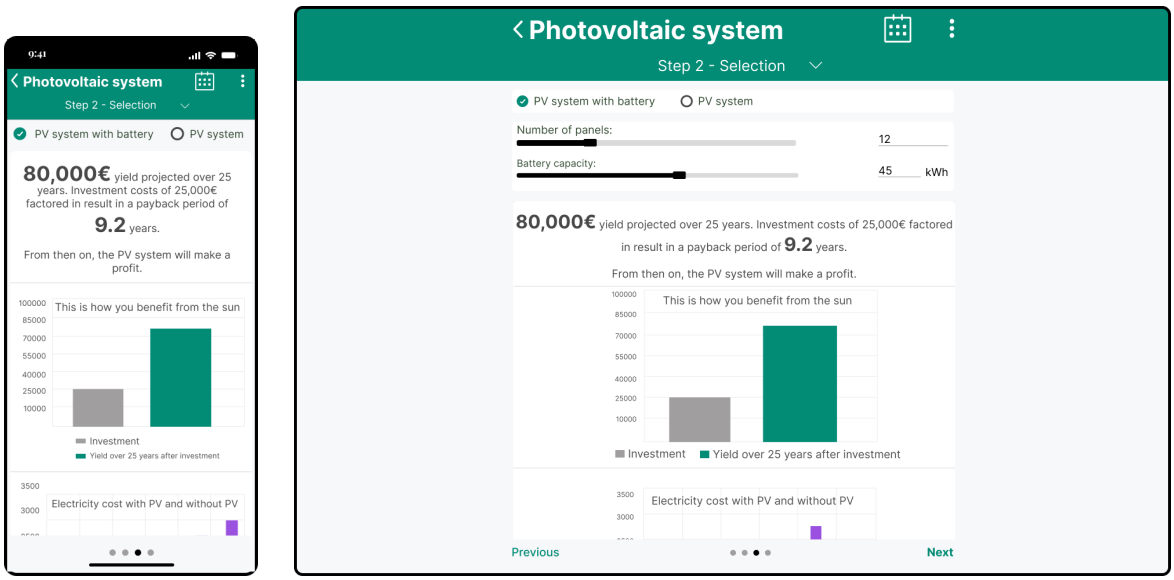


Figure 26: Step 2 return on investment report.

Lastly monthly electricity production report which shows how much energy the PV system is expected to produce each month throughout the year. This is an important report to consider because the amount of energy produced by a PV system varies depending on the time of year and the weather conditions. For example, in Germany, the amount of sunlight during the winter months is much less than during the summer months, which means that the PV system will produce less energy in the winter. By considering the monthly production over the years, users can make more informed decisions about the size and capacity of their PV system, as well as plan for potential energy shortfalls during the winter months.

### 6.3.3 Step 3 - Qualification

Step 3 - Qualification is an important step before starting the PV construction, as it provides the necessary information and images for the craftsman to start the work. In this step, the user needs to upload images of the occupied roof surface, facade of the adjacent PV roof, meter cabinet, connection box, and basement room. These images provide important information for the craftsman about the installation site and help them in planning and executing the installation process efficiently. For example, images of the occupied roof surface will help the craftsman in determining the size and placement of the PV panels. Similarly, images of the meter cabinet and connection box will help the craftsman in understanding the electrical connections and requirements for the PV system.

In summary, uploading these images in Step 3 - Qualification is necessary for

providing the necessary information and details to the craftsman to ensure a smooth and efficient PV system installation process.

### 6.3.4 Step 4 - Construction

Step 4 in the PV installation process involves the construction phase, which starts once the user has completed the qualification step and has been provided with a list of tasks based on the house conditions and simulation results.

For each task, a list of craftsmen with ratings is provided, and the user can contact them via phone, email, route, website, or chat.

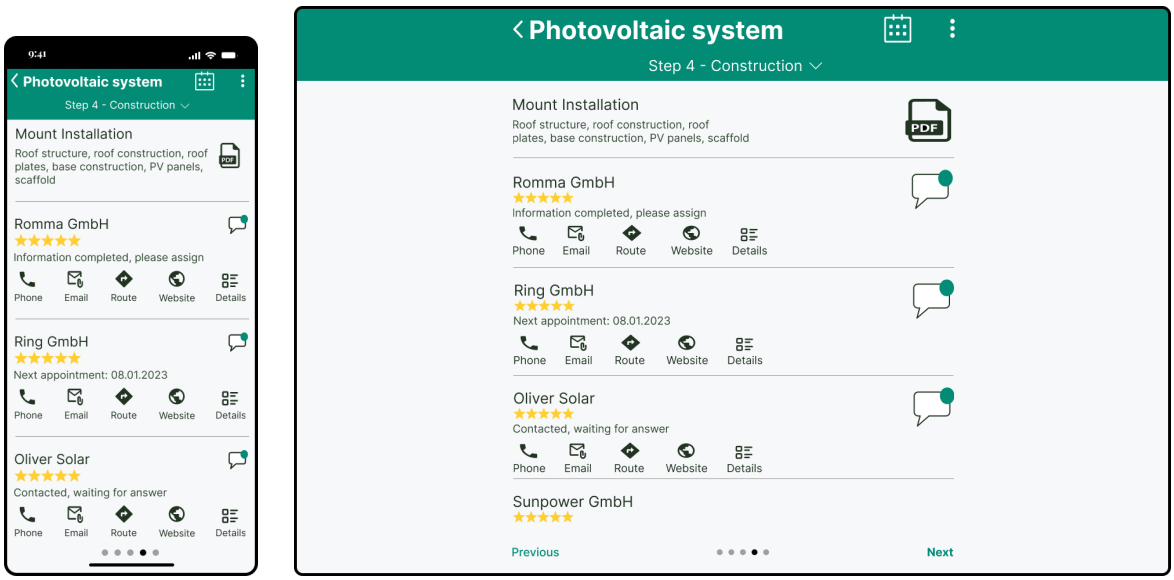


Figure 27: Showing task wise top-rated craftsman lists.

The user can send a request to the craftsman with all project details and come to an agreement before assigning the task.

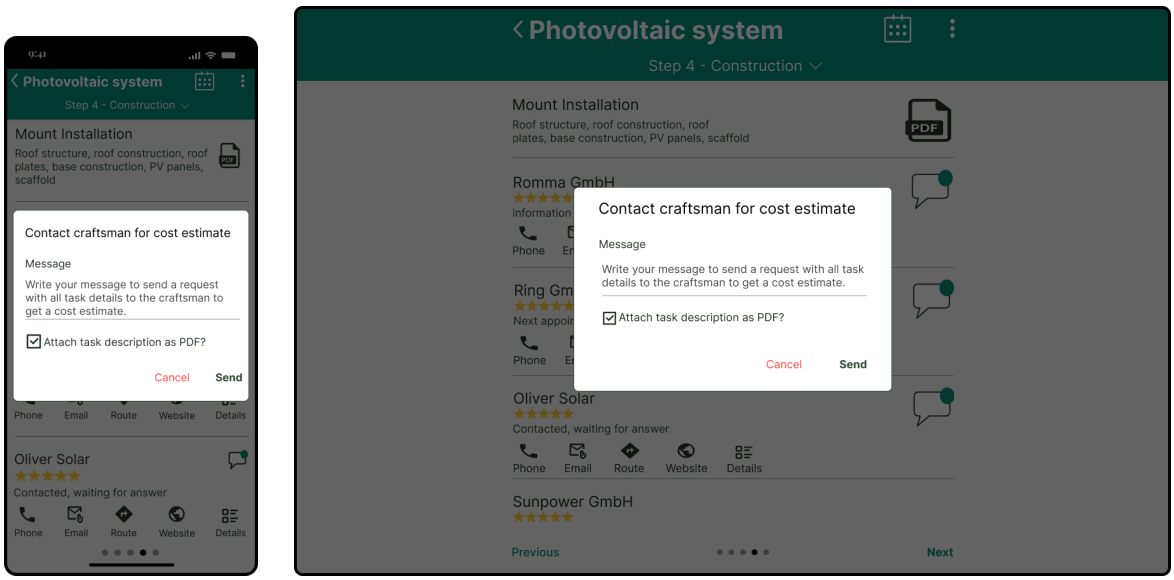


Figure 28: Sending contact request to craftsman for cost estimate.

Additionally, the user may need to schedule multiple on-site or online appointments during the construction phase, and the app provides the option to book appointments by clicking on the appointment icon. By following these steps, the user can ensure that the PV installation is carried out by qualified craftsmen and that the process is smooth and efficient.

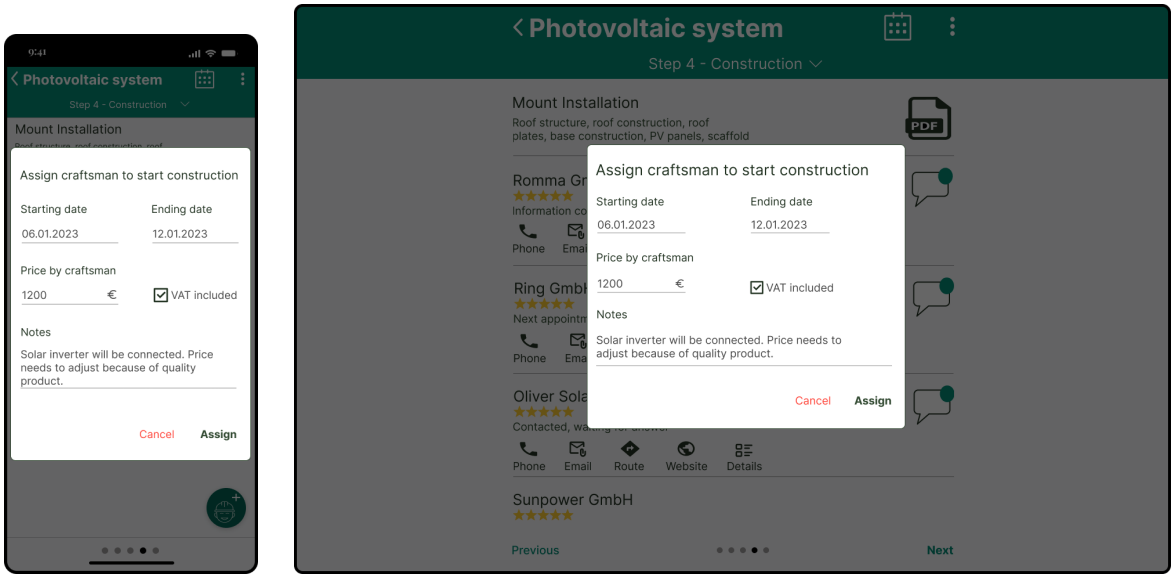


Figure 29: Assigning the task to a craftsman.

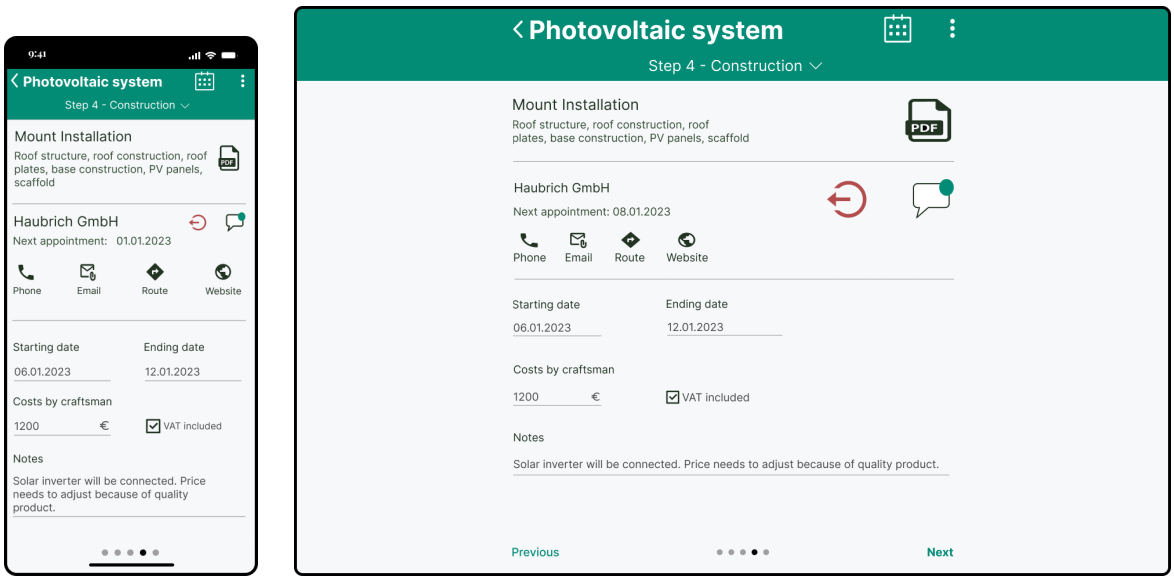


Figure 30: After assigning the task to craftsman and task details will be shown along with selected craftsman information.

### 6.3.5 Step 5 - Rating

In the final step of the PV system installation process, Step 5 - Rating, users are able to provide feedback on the craftsman's performance for each task that was assigned. The rating system is based on several factors, including the agreement amount and actual amount, task delivery time, and task quality against money.

The agreement amount and actual amount are important factors to consider because they determine how well the craftsman was able to adhere to the initial agreement that was made between the user and the craftsman. If the actual amount is significantly different from the agreement amount, user can also specify the reason like Extra work, bad time management. Task quality against price is perhaps the most important factor in the rating process as it directly reflects the level of satisfaction that the user has with the final product. If the craftsman is able to complete the task with a high level of quality and attention to detail, this will likely result in a higher rating. Conversely, if there are issues with the quality of the work or if the craftsman is not able to fully complete the task as agreed upon, this will negatively impact the rating.

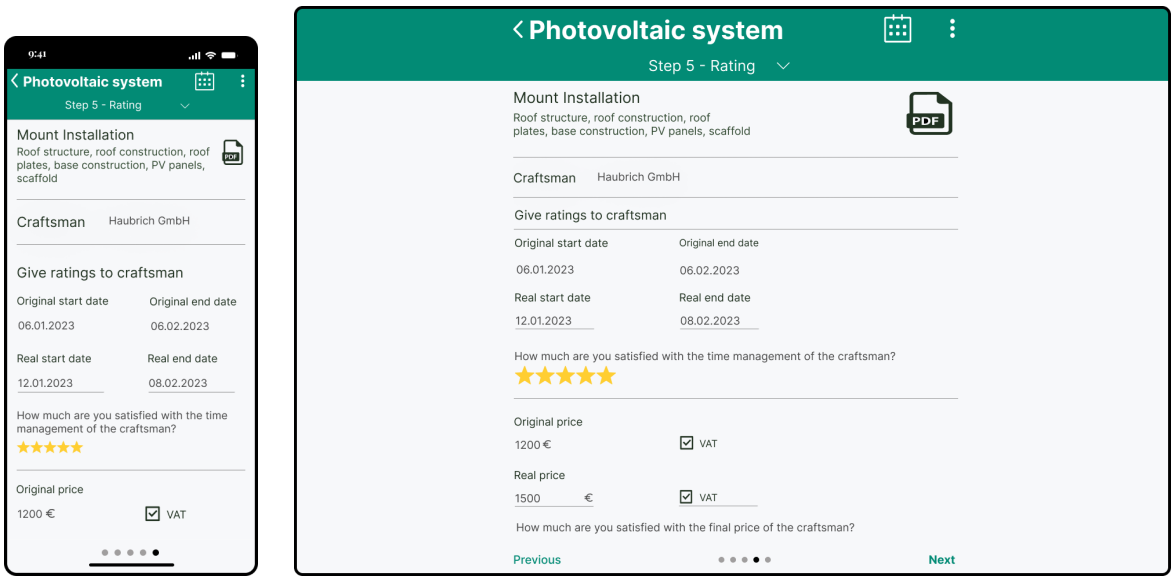


Figure 31: Default rating page.

The top-rated craftsman will be sorted on top for their respected task category. By providing constructive feedback through the rating, users can help to improve the quality of service that is provided by craftsmen in the PV system installation industry.

# 7 Conclusion

In this work, it has been shown that a photovoltaic simulation and financial model can be effectively implemented using a user-centered design approach. The process of designing and developing a prototype to create an intuitive and easy-to-use app user interface has been thoroughly explained. The selection of appropriate technologies, such as Ionic, VueJS, Capacitor, Python, and Django REST framework, has been justified with relevant arguments. The use of a local JSON file to store hard-coded content has also been highlighted as a good practice in software construction. Overall, this work demonstrates how a well planned and executed software development process can result in a high-quality and user-friendly product.

## 7.1 Future work

This work has so much potential for additional integrations and still has a lot of room for improvement. As the entire development process was done with goals like maintainability, scalability, and reusability in mind, the app can be extended easily and smoothly without the risk of breaking existing components. The following are some proposed next steps which could be implemented to refine the app's overall performance and capabilities as well as introduce new behaviors and features to it:

1. Improve overall responsiveness behavior by testing on more different devices and operating systems.

2. Refine interactiveness in sections where user engagement is desired to keep users in furthering their exploring of the app.

3. Incorporate more UX laws and ensure their overall implementation throughout the app.

4. To simulate photovoltaic systems, it is crucial to have cadastral data such as location, orientation, and roof slope of the building. Therefore, collecting cadastral data and creating a standardized format will make the process streamlined, and the data will be easily integrated into the simulation. The utilization of cadastral data in photovoltaic simulation is becoming increasingly important as more governments are promoting the use of renewable energy, and therefore, the standardization of cadastral data is essential to enable the development of accurate and reliable photovoltaic systems worldwide.

5. Add visual documentation on the input pages so that user can understand what type of data user needs to provide as the app can be used by people of different age groups.

6. Showing real time 3D visualization before and after installing photovoltaic system with highlighting it's benefits.

7. Adding more options on specific PV simulation parameters which have impact on simulation results like Battery model, Inverter model, PV panel,etc.

8. Adding more features on PV simulation like risk analysis, advance simulation optimization,etc.

9. Integrate advance AI (artificial inteligence) to generate construction tasks based on photovoltaic simulation parameters.

10. Integrate features considering different types of app user role like, house owners, craftman, admin,etc. Currently the app has been developed considering only house owners but it can be easily extendable for different roles.

11. Integrating AI bots. AI bots can enhance the user experience by providing instant and personalized assistance to users.

12. Provide app in multiple languages.

# References


[1] The design and simulation software for photovoltaic systems. https://valentin-software.com/en/products/pvsol-premium/, 2022. Accessed April 04, 2023.

[2] Pvsyst system design board. https://www.pvsyst.com/features/, 2022. Accessed April 04, 2023.

[3] Samer Alsadi and Tamer Khatib. Photovoltaic power systems optimization research status: A review of criteria, constrains, models, techniques, and software tools. *Applied Sciences*, 8(10):1761, 2018.

[4] Mermoud André and Wittmer Bruno. Pvsyst version 6 manual. Technical report, PVsyst SA, 2014.

[5] Luz Calvo, Isadora Christel, Marta Terrado, Fernando Cucchietti, and Mario Pérez-Montoro. Users' cognitive load: A key aspect to successfully communicate visual climate information. *Bulletin of the American Meteorological Society*, 103(1):E1–E16, 2022.

[6] Christopher P Cameron, Joshua Stein, and Clifford W Hansen. Evaluation of pv performance models and their impact on project risk. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2011.

[7] Mitul Ranjan Chakraborty, Subhojit Dawn, Pradip Kumar Saha, Jayanta Bhusan Basu, and Taha Selim Ustun. A comparative review on energy storage systems and their application in deregulated systems. *Batteries*, 8(9):124, 2022.

[8] European Commission, Joint Research Centre, and A Jäger-Waldau. *Pv status report 2019*. Publications Office, 2019. doi: https://op.europa.eu/en/publication-detail/-/publication/dfa5cde5-05c6-11ea-8c1f-01aa75ed71a1/language-en.

[9] David Connolly, Henrik Lund, Brian Vad Mathiesen, and Martin Leahy. A review of computer tools for analysing the integration of renewable energy into various energy systems. *Applied energy*, 87(4):1059–1082, 2010.

[10] Nathan Curtis. *Modular web design: creating reusable components for user experience design and documentation*. New Riders, 2010.

[11] Aron P Dobos. Pvwatts version 5 manual. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2014.

[12] Hua Dong, Chris McGinley, Farnaz Nickpour, Abdusselam Selami Cifter, Inclusive Design Research Group, et al. Designing for designers: Insights into the knowledge users of inclusive design. *Applied ergonomics*, 46:284–291, 2015.

[13] John A Duffie and William A Beckman. *Solar engineering of thermal processes*. Wiley New York, 1980.

[14] Diana Elliott. *PV/T Software Coherence and Applicability*. PhD thesis, Thesis. University of Strathclyde. Dept of Mechanical Engineering. http . . . , 2014.

[15] Homer energy. Homerpro version 3.7 manual. Technical report, 2016. Accessed May 7, 2023.

[16] Jose Etcheverry et al. Assessing the cost feasibility of solar projects in canada using the retscreen expert software. https://yorkspace.library.yorku.ca/xmlui/handle/10315/38602, 2021.

[17] Eurostat. Evaluation of household consumers electricity and gas price in the EU. https://ec.europa.eu/eurostat/documents/4187653/11581527/Evolution+of+household+electricity+and+gas+prices+in+the+EU.png/8e8a38f6-e791-ee36-c14c-844b91fad606?t=1634632173305, 2021. Accessed April 10, 2023.

[18] Paul Gilman, Nate Blair, Mark Mehos, Craig Christensen, Steve Janzou, and Chris Cameron. Solar advisor model user guide for version 2.0. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2008.

[19] gramwzielone. We already have 2 GW in photovoltaics, but in Europe we are still in the second ten. https://www.gramwzielone.pl/energia-sloneczna/103219/mamy-juz-2-gw-w-fotowoltaice-ale-w-europie-jestesmy-jeszcze-w-drugiej-dziesiatce, 2019. Accessed April 10, 2023.

[20] INSEL. Insel-the graphical programming language for the simulation of renewable energy systems. Technical report. Accessed May 7, 2023.

[21] Arnulf Jäger-Waldau, Ioannis Kougias, Nigel Taylor, and Christian Thiel. How photovoltaics can contribute to ghg emission reductions of 55the eu by 2030. *Renewable and Sustainable Energy Reviews*, 126: 109836, 2020. ISSN 1364-0321. doi: https://doi.org/10.1016/j.rser.2020.109836. URL https://www.sciencedirect.com/science/article/pii/S1364032120301301.

[22] S.A. Klein and W.A. Beckman. F-chart is the authoritative solar system analysis and design program. Technical report. Accessed May 7, 2023.

[23] Sari Kujala, Virpi Roto, Kaisa Väänänen-Vainio-Mattila, Evangelos Karapanos, and Arto Sinnelä. Ux curve: A method for evaluating long-term user experience. *Interacting with computers*, 23(5):473–483, 2011.

[24] A Prashant Kumar. Analysis of hybrid systems: Software tools. In *2016 2nd International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*, pages 327–330. IEEE, 2016.

[25] Chun Sing Lai, Youwei Jia, Zhao Xu, Loi Lei Lai, Xuecong Li, Jun Cao, and Malcolm D McCulloch. Levelized cost of electricity for photovoltaic/biogas power plant hybrid system with electrical energy storage degradation costs. *Energy conversion and management*, 153:34–47, 2017.

[26] Adam Lancaster. Paper prototyping: the fast and easy way to design and refine user interfaces. *IEEE Transactions on Professional Communication*, 47(4):335–336, 2004.

[27] Laplace System Co. Ltd. Solar pro-photovoltaic system simulation software. `https://www.lapsys.co.jp/english/products/pro.html`. Accessed May 7, 2023.

[28] Jian Mao, Hanjun Ma, Yue Chen, Yaoqi Jia, and Zhenkai Liang. Automatic permission inference for hybrid mobile apps. *Journal of High Speed Networks*, 22(1):55–64, 2016.

[29] B Marion, M Anderberg, and P Gray-Hann. Recent revisions to pvwatts. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2005.

[30] Adrian Mendoza. *Mobile user experience: patterns to make sense of it all*. Newnes, 2013.

[31] Jakob Nielsen. *Usability engineering*. Morgan Kaufmann, 1994.

[32] NREL. Pvwatts® calculator: India (fact sheet). Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2014.

[33] Verbraucherzentrale NRW. Mini-PPAs for residential PV in Germany. `https://www.pv-magazine.com/2022/03/09/mini-ppas-for-residential-pv-in-germany/`, 2022. Accessed April 10, 2023.

[34] University of Wisconsin System. Trnsys-a transient system simulation program. Technical report. Accessed May 7, 2023.

[35] Piotr Olczak, Małgorzata Olek, Dominika Matuszewska, Artur Dyczko, and Tomasz Mania. Monofacial and bifacial micro pv installation as element of energy transition—the case of poland. *Energies*, 14(2), 2021. ISSN 1996-1073. doi: 10.3390/en14020499. URL https://www.mdpi.com/1996-1073/14/2/499.

[36] Janosch Ondraczek. Are we there yet? improving solar pv economics and power planning in developing countries: The case of kenya. *Renewable and Sustainable Energy Reviews*, 30:604–615, 2014. URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2321127.

[37] April Oxera. Discount rates for low-carbon and renewable generation technologies. *Report. Oxera Consulting LLP*, 747, 2011. URL https://www.oxera.com/wp-content/uploads/2018/03/Oxera-report-on-low-carbon-discount-rates.pdf.

[38] Constantinos S Psomopoulos, George Ch Ioannidis, Stavros D Kaminaris, Kostas D Mardikis, and Nikolaos G Katsikas. A comparative evaluation of photovoltaic electricity production assessment software (pvgis, pvwatts and retscreen). *Environmental Processes*, 2:175–189, 2015.

[39] Ibrahim Reda and Afshin Andreas. Solar position algorithm for solar radiation applications. *Solar energy*, 76(5):577–589, 2004.

[40] Marc Rettig. Prototyping for tiny fingers. *Communications of the ACM*, 37 (4):21–27, 1994.

[41] John W. Satzinger and Lorne Olfman. User interface consistency across end-user applications: The effects on mental models. *Journal of Management Information Systems*, 14(4):167–193, 1998. doi: 10.1080/07421222.1998.11518190. URL https://doi.org/10.1080/07421222.1998.11518190.

[42] Yashwant Sawle, SC Gupta, and Aashish Kumar Bohre. Pv-wind hybrid system: A review with case study. *Cogent Engineering*, 3(1):1189305, 2016.

[43] Yashwant Sawle, SC Gupta, and Aashish Kumar Bohre. Pv-wind hybrid system: A review with case study. *Cogent Engineering*, 3(1):1189305, 2016.

[44] Dr Helen Sharp. Professor yvonne rogers, and dr jenny preece. interaction design: Beyond human-computer interaction, 2007.

[45] Walter Short, Daniel J Packey, and Thomas Holt. A manual for the economic evaluation of energy efficiency and renewable energy technologies. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 1995.

[46] Sunanda Sinha and SS Chandel. Review of software tools for hybrid renewable energy systems. *Renewable and sustainable energy reviews*, 32: 192–205, 2014.

[47] Johannes Weniger, Tjarko Tjaden, and Volker Quaschning. Sizing of residential pv battery systems. *Energy Procedia*, 46:78–87, 2014.

[48] Hui Zhang, Jin-Lan Xu, and Xin Cheng. Interactive educational software for teaching water distribution networks design. *Computer Applications in Engineering Education*, 24(4):615–628, 2016. doi: https://doi.org/10.1002/cae.21736. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/cae.21736`.

# List of Figures