Design a “PV Living Lab” for the UCT Menzies Building



**Prepared by:**

**Tefo Motaung**

**MTNTEF001**

Department of Electrical Engineering

University of Cape Town

**Prepared for:**

**RICHARD LARMOUR**

ENGINEERING AND BUILT ENVIRONMENT

University of Cape Town

**September 2021**

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the academic requirements for a Bachelor of Science degree in Mechatronics Engineering**.**

Abstract

The Electrical Engineering Department plans to install an experimental solar photovoltaic (PV) system to generate electricity for the department as well as for research and teaching in renewable energy.

This report elaborates the process of designing a dashboard display, termed ‘living lab’, for a monitor that is planned to be deployed in the Menzies 3rd floor foyer. It would show live and historical and live PV system performance data, building demand data at the substation level, and weather data.

The method approached is to use basic web-development technologies to build a webapp that would be hosted on the cloud and displayed on the monitor. HTML5 was used as markup language, CSS3 used to style webpages, while a combination of vanilla JavaScript, Vue framework, and Django templates were used for interactivity with the website. All charts on the webapp are from JavaScript’s ChartJS library.

The backend uses the following technologies: PostgreSQL is the database; Python’s Django framework is used to administer the integration between the backend and the front-end, while several application programming interfaces (which will be referred to as APIs) were used to extract data from the server to fulfil various purposes using JavaScript’s fetch API.

The website was hosted on GitHub pages, which serves as a hosting site for static websites, and came very handy when demoing the frontend design of the webpage and its dashboards. The dashboards can be observed in the results section, as well some appendices for further demonstration

Acknowledgements

I would like to appreciate my supervisor, Richard Larmour, for his efforts, guidance and assistance. I would also like to acknowledge the support of my family throughout my university career. All of them showed immense support to me while I’m pursuing my dreams.

Plagiarism declaration

1. **I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.**
2. **I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this final year project report from the work(s) of other people, has been attributed and has been cited and referenced.**
3. **This final year project report is my own work.**
4. **I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name:** | **Tefo Motaung** |  | |
| **Signature:** |  | **Date:** | 07 November 2021 |

# Table of Contents

[Design a “PV Living Lab” for the UCT Menzies Building 1](#_Toc87176478)

[Prepared by: 1](#_Toc87176479)

[Department of Electrical Engineering 1](#_Toc87176480)

[Prepared for: 1](#_Toc87176481)

[Abstract i](#_Toc87176482)

[Acknowledgements ii](#_Toc87176483)

[Plagiarism declaration iii](#_Toc87176484)

[Table of Contents iv](#_Toc87176485)

[Table of Figures vi](#_Toc87176486)

[1. Introduction 2](#_Toc87176487)

[1.1 Introduction 2](#_Toc87176488)

[1.2 Motivation 2](#_Toc87176489)

[1.3 Problem statement 2](#_Toc87176490)

[1.4 Objectives 3](#_Toc87176491)

[1.5 Thesis contributions 3](#_Toc87176492)

[1.6 Terms of reference 3](#_Toc87176493)

[1.6.1 The user requirements 3](#_Toc87176494)

[1.6.2 List of functionalities: 3](#_Toc87176495)

[1.7 Scope and limitations 4](#_Toc87176496)

[1.9 Thesis outline 5](#_Toc87176497)

[1.9.1 Chapter 2, Background 5](#_Toc87176498)

[1.9.2 Chapter3, Methodology 5](#_Toc87176499)

[1.9.3 Chapter 4, Design 5](#_Toc87176500)

[1.9.4 Chapter 5, Results 5](#_Toc87176501)

[1.9.5 Chapter 6, Conclusions 5](#_Toc87176502)

[2. Literature Review 6](#_Toc87176503)

[2.1 Living Labs 6](#_Toc87176504)

[2.1.1 Attempt at definition 6](#_Toc87176505)

[2.1.2 Living Lab Stakeholders 6](#_Toc87176506)

[2.1.3 Campus as a Living Lab 7](#_Toc87176507)

[2.1.4 Living lab projects in other universities 8](#_Toc87176508)

[2.2 Solar Monitoring Systems 14](#_Toc87176509)

[2.2.1 Overview 14](#_Toc87176510)

[2.2.2 Taxonomy of Solar PV Monitoring System 14](#_Toc87176511)

[2.2.3 Data acquisition 14](#_Toc87176512)

[2.2.4 Data processing 17](#_Toc87176513)

[2.2.5 Data visualization 18](#_Toc87176514)

[2.2.6 Data storage 19](#_Toc87176515)

[3. Methodology 20](#_Toc87176516)

[3.1 Overview of Approach 20](#_Toc87176517)

[3.2 How to address research objective 20](#_Toc87176518)

[3.2.1 Software: Webapps and desktop applications 20](#_Toc87176519)

[3.2.2 Electronic solution 21](#_Toc87176520)

[3.2.3 Logical reason for choosing the method / approach 21](#_Toc87176521)

[3.2.4 What to prototype 23](#_Toc87176522)

[3.2.5 How to build prototype 23](#_Toc87176523)

[3.2.6 How to gather data 23](#_Toc87176524)

[4. Design 24](#_Toc87176525)

[Overview 24](#_Toc87176526)

[4.1 System design 24](#_Toc87176527)

[4.2 Software design 25](#_Toc87176528)

[4.2.1 Traceability Matrix 25](#_Toc87176529)

[4.2.2 Requirement Analysis 26](#_Toc87176530)

[4.2.3 Design Choices 28](#_Toc87176531)

[4.2.4 Figures of Merit 29](#_Toc87176532)

[4.2.5 Solution Decision 30](#_Toc87176533)

[5. Results 34](#_Toc87176534)

[5.1 Experimental setup 34](#_Toc87176535)

[5.2 Data Acquisition 34](#_Toc87176536)

[5.3 Visualization 35](#_Toc87176537)

[5.4 Simulation 38](#_Toc87176538)

[6. Conclusion 39](#_Toc87176539)

[6.1 Conclusions 39](#_Toc87176540)

[6.2 Future work 39](#_Toc87176541)

[7. List of References 40](#_Toc87176542)

[8. Appendices 42](#_Toc87176543)

[8.1 HTML 42](#_Toc87176544)

[8.2 CSS 43](#_Toc87176545)

[8.3 GitHub link 44](#_Toc87176546)

# Table of Figures

[Figure 1: Four main stakeholders of Living Labs in harmony 7](#_Toc87142272)

[Figure 2: Overlaps of primary functions of universities in the creation of living labs 8](#_Toc87142273)

[Figure 3: The Yale PAYT Logo 8](#_Toc87142274)

[Figure 4: NEST platform with several modules plugged onto it 9](#_Toc87142275)

[Figure 5: Model showing how the individual modules are placed and removed onto the platform 9](#_Toc87142276)

[Figure 6: NEST apartment for two made of sustainable resources 10](#_Toc87142277)

[Figure 7: Water Hub in NEST basement used to clean recycle water 10](#_Toc87142278)

[Figure 8: Photograph of Electric Vehicles deployed in Oxford University 11](#_Toc87142279)

[Figure 9: Total emissions produced to provide food for St. Hilda's Colleg students 11](#_Toc87142280)

[Figure 10: Graph showing nutrients found in urine in percentages. 12](#_Toc87142281)

[Figure 11: diagram of the first version of waterless urinal at UCT 13](#_Toc87142282)

[Figure 12: The UCT urine Treatment Process 13](#_Toc87142283)

[Figure 13: Example of small-scale PV system [18] 15](#_Toc87142284)

[Figure 14: South Africa's large scale PV solar plant [19] 15](#_Toc87142285)

[Figure 15: LP02-L spectrally flat Class C Pyranometer manufactured by Hukseflux[22]. 16](#_Toc87142286)

[Figure 16: I-V characteristics of four similar PV panels at different temperatures[25]. 16](#_Toc87142287)

[Figure 17: NTC Thermistor (IEC standard) [27] 16](#_Toc87142288)

[Figure 18: Circuit Symbol for Thermocouple [30] 16](#_Toc87142289)

[Figure 19: schematic of the I-V sequencer system 17](#_Toc87142290)

[Figure 20: Front panel of VI developed for displaying the IV characteristic graph and PV array chart side by side. 18](#_Toc87142291)

[Figure 21: Average energy generated per day for each month for the period shown. 19](#_Toc87142292)

[Figure 22: Monthly operational efficiencies of three different PV modules. 19](#_Toc87142293)

[Figure 23: Data flow from measurement to display 24](#_Toc87142294)

[Figure 24: webpages of software and related code files 31](#_Toc87142295)

[Figure 25: Hardware Setup of PV modules in Kraaifontein 34](#_Toc87142296)

[Figure 26: Plot of Temperature vs Wind Speed measured by SolarEdge environmental sensors 35](#_Toc87142297)

[Figure 27: SQL command which returns data from specific date 36](#_Toc87142298)

[Figure 28: SQL script showing energy and power side by side from two tables. 37](#_Toc87142299)

[Table 1: Classification of data for two types of PV systems 15](#_Toc87142303)

[Table 2: Categories used to decide between IoT and pure SW solution 21](#_Toc87142304)

[Table 3: Traceability Matrix to discuss the different requirements for the software solution 25](#_Toc87142305)

[Table 4: Figures of merit used to compare the use of a front end framework and making one's own programming architecture. 29](#_Toc87142306)

Nomenclature

PV: Photovoltaic

API: Application Programming Interface

HTML: Hypertext Markup Language

CSS: Cascading Style Sheet

SQL: Structured Query Language

GUI: Graphical User Interface

GHG: Greenhouse Gases

PCB: Printed Circuit Board

IoT: Internet of Things

IDE: Integrated Development Environment

SW: software

URL: Universal Resource Locator

DOM: Document Object Model

UCT: University of Cape Town

RAM: Random Access Memory

# Introduction

## Introduction

The University of Cape Town, like many others around the world, is looking to participate in defining the concept of ‘Living Labs’. The beginning of this long-term project starts with a commitment to transforming its campus buildings into Renewable Energy use cases.

It starts with installing PV systems to harness the energy from the sun using solar panels. This sparks a curiosity about where the installed devices will be adequately performant. Which is why an experimental PV system will require some tests to determine its performance parameters. The amount of radiation also needs to be observed to determine just how much electricity can be expected at specific times during the day.

This is where the project comes in. A performance dashboard is necessary to visualize all this information.  This study elaborates on the design and implementation of a webapp that is dedicated to plotting raw radiation, solar PV system generation, building demand and weather data, and its related technologies necessary to build the user interface and its back-end infrastructure.

## Motivation

The topic of renewable energy has been in general academics’ minds for a while. It has become increasingly evident to the public too because all over the world. In South Africa, global warming has led to climate change, and fatal flash floods and longer droughts are observed in some parts of the country [2]. This has led many to take notice of the warnings of climate change. It is necessary to take advantage of this new wave of curiosity in the public to educate and push research to its limits in order to further understand how PV systems can alleviate the situation by allowing the earth to restore some of the resources that take decades and/or millennia to replenish.

This development will contribute towards understanding the amount of energy that can be expected when PV systems are installed and utilized for electricity generation. It will attempt to formulate a comprehension of how several environmental parameters such as temperature and solar irradiation affect the performance of already existing ones.

## Problem statement

Solar monitoring systems are made to monitor how already existing PV systems are performing in order to best optimize their use within the buildings in which they’re installed. Unfortunately, not many affordable devices exist to aid customers with determining whether their intentions of acquiring PV Systems for their buildings will be worthwhile.

The limitations include lack of access to such monitoring devices and high price points for purchasing them. This has led to the need for a cheaper alternative, that’s also intuitive and does not require one to be an environmental scientist to determine how the information available on the platform can assist them in making sound decisions for their building’s energy savings plans.

## Objectives

The purpose of writing this dissertation is to discuss the design and implementation of the PV System Dashboard that is intended to be displayed in the Menzies 3rd floor foyer monitor. This small product will contribute towards establishing a university living lab.

With the assistance of a weather station and its API requesting data from a server, a PV system which will be used to measure key performance parameters for panels and building demand from the substation level. The ambition is to achieve this goal before November 2021.

The dashboard would be created using web technologies and hosted online on a hosting site or cloud service of choice so that the data is accessible.

## Thesis contributions

The main contributions of this thesis are as follows:

* A visualization of five-day weather forecast data on demand has been made available for any location in the world for the purpose of determining the trend in temperatures at peak time. This queries data from OpenWeatherMap.
* The solution also provides accurate temperature information for the University of Cape Town always displayed at the top of the dashboard.
* A plot visualizing the solar irradiance data for the University of Cape Town has also been provided. This shows just how much solar energy can be expected during the day.
* The free tier graphing libraries are not easily accessible to the average developer and have a steep learning curve and unfriendly interface, utilizing a lot of RAM memory, and do not provide consistent plots.

## Terms of reference

The objective of this product is to build the front-end and back end of the PV system dashboard display. All design choices and strategies to be elaborated on in this dissertation.

### The user requirements

The listed requirements below are from the project brief:

* Design dashboard front-end display.
* Display live PV system performance.
* Display historical PV system performance.
* Display building demand data at the substation level.
* Display weather data.
* Store weather station data in the back end
* Store power data in the back end.

### List of functionalities:

* The front-end display functions as the graphical user interface (referred to as the GUI from here onward) of the dashboard. This is the part of the webapp that encourages interaction from the user with buttons, graphics, logos, colors, and other input boxes.
* The display of live PV system performance serves to visualize the measured physical parameters that communicate to the user the amount of energy that is harnessed from the panels at that specific moment.
* The display of historical PV system performance shows how the panel has been behaving and how much energy had been created in the previous days.
* This functions to visualize the amount of energy required to meet the needs of the users in the building at specific points in time during the day.
* The performance of PV systems is greatly influenced by the climatic conditions, especially temperature [3], so this information is important to understand the behavior of panels at various weather conditions.
* Storage of weather data serves to ensure the availability of weather data in a database for further study and analysis.
* Having the electricity produced in the backend can help to track the durability of the PV system and eventually tell the expected lifespan of specific PV systems experimentally.
* One major reason why solar is so important is that it can help alleviate the financial pressure on the university. The Time of use tariff when displayed can help to tell which orientation of the panels will best contribute towards monetary savings.

## Scope and limitations

The restrictions of the project are listed in the bullet points below:

* The data is limited to being acquired from three sources. Weatherlink, Openweathermap, and SolarEdge.
* The development of the software is only restricted to working on browsers, instead of being installed on a local machine.
* Since the software solution is a webapp, one cannot access the information unless one has access to the internet.
* The screen resolution is currently restricted to 13-inch displays found on laptops, albeit this might change overtime, depending on when the viewer attempts to access the software solution developed and discussed in this paper.
* The free tier graphing libraries are not easily accessible to the average developer and have a steep learning curve and unfriendly interface, utilizing a lot of RAM memory, and do not provide consistent plots.

## Thesis outline

The remainder of this thesis is organized as follows:

### Chapter 2, Background

A literature Review is written in this chapter, in which the background of solar monitoring systems is discussed and examples of living labs at university campuses are given.

### Chapter3, Methodology

In this chapter, a comparison of methods for approaching the problem is made and one of them is chosen based on given reasons.

### Chapter 4, Design

The chosen methodology is used to create a custom solution which solves this problem and meets the requirements of the user. Simulation of the solution is also provided here.

### Chapter 5, Results

Plots from measurements, which are also used in the solution created in the design section, are discussed in this chapter. The database scripts and results are also provided.

### Chapter 6, Conclusions

This chapter concludes the project, and discusses future work which can be done, and how the developer expects to continue improving the solution.

# Literature Review

## Living Labs

The conception of the software monitoring system discussed in this report is founded on the pursuit that the university has in transforming its campus into a living lab. That is the bigger picture.

Although the term living lab wasn’t recently coined, the concept is still new to many in industry and several sources interpret it differently as they see fit. The University of Manchester describes theirs as “…a site for applied teaching and research around sustainability” [3].

However, the reference to sustainability isn’t always chained to living labs. Since each organization can practically define it to suit the needs of its stakeholders, sustainability endeavors can be but a side note in some contexts. Consider the Living Labs Methodology Handbook approach [5]. One of the key elements defined in the handbook is sustainability, in which the stakeholders ought to ensure that they take responsibility for the environmental effects of the living lab activities, not neglecting the economic nor the social.

In the context of this dissertation, sustainability will be defined as ‘meeting the needs of this generation while ensuring that the earth’s resources don’t run out so that the next generations can meet theirs’. The purpose of suitability is to maintain ecological balance [5].

### Attempt at definition

Living labs are in many contexts understood to be a research concept. They’re catered around the user and based on the ecosystem, or operating in a territorial context (e.g. campus, city, or village), which means that innovation and research are integrated in real life communities and settings [6], [7].

Living labs are at heart a learning platform, however the involvement of users is essential as well, while rapidly innovating to solve problems in their environment. They bridge the gap between the market and experimentation within traditional laboratories. They can speed up the adoption of new technologies and their enhancement [7].

There are different types of living labs. Below is a list of living lab environments as described in the living lab methodology handbook. Note that the concept of living labs is under development and several other types certainly exist.

* Research Living Lab
* Corporate Living lab
* Organization Living lab
* Intermediary Living Lab
* Time limited Living lab

There are several keys important to establish in the birth of any living lab. Namely value, influence, sustainability, openness, and realism. These define what living labs are and what counts as a living lab and provide foundation for designing it and determining its operations.

### Living Lab Stakeholders

Key stakeholders in living labs are users, for which products are designed and built, companies, which may or may not have commercial interest in projects, public organizations which hoard the interest of the general population, and finally researchers whose primary interest is to establish a comprehension of whichever topic is in question.[5]. The figure below illustrates the aim of achieving a quattro helix by harmonizing the innovation process among the aforementioned stakeholders.

Diagram

Description automatically generated

Figure 1: Four main stakeholders of Living Labs in harmony

It is without a doubt that living labs should have access and good relationship with users who are willing the partake in the innovation and development process. It is in the interest of the stakeholders to ensure that the project has access to multi-contextual environments and state of the art technology and infrastructure that can support both the processes of development and testing, and the involvement of said users [4].

### Campus as a Living Lab

As campuses begin to take the idea into perspective, the birth of another term was inevitable. Campus as a living lab is an academic approach to living labs by several universities throughout the whole world.

Within the campus, living labs are seen as problem solving technique, and type of project management technique. They use university campus and surrounding community territories as places to experiment with sustainable technologies and lifestyle[3].

The implementation of living labs on campus involves universities using their buildings and resources to investigate, test and demonstrate innovative technologies for the community. The innovations are usually concerned with sustainability. The termed coined to accommodate this is Campus as a sustainability living lab. They combine research education and campus operations together while pursuing to achieve a common goal.

The figure below shows how the three factors affecting explained previously overlap. Extending these overlaps aids in creating a successful living lab.

Diagram, venn diagram

Description automatically generated

Figure 2: Overlaps of primary functions of universities in the creation of living labs

### Living lab projects in other universities

This section will single out examples of living labs implemented in several universities around the world. Note just how different these examples are with respect to each other, while pursuing the same goal: a greener earth concerned with sustainability to ensure future generations can meet their own needs without compromising meeting the needs of today.

#### Pay as you throw at the University of Yale

This campus uses a pay as you throw system to charge entities for the amount of trash that the throw away. Much like individuals are charged for commodities such as water and electricity, the purpose is to encourage more environmentally sustainable waste habits[7].

The aim is to decrease the amount of waste that needs to be managed on campus while promoting the reuse of some materials and the recycling of others. This program was one of the goals in Yale Sustainability Plan 2025[8].

The task force implementing this strategies were hoping to

* Raise awareness of being conscious consumers
* Encourage the reduction of campus’ carbon footprint
* Encourage collaboration between operational staff, researchers and students by creating academic research opportunities.

Diagram

Description automatically generated with low confidence

Figure 3: The Yale PAYT Logo

#### NEST – The Swiss Living Lab

This project was initiated in order to bridge the gap between the technology that works well in the laboratory and the need to mature, dependable products in the market. NEST stands for Next Evolution in Sustainable Building Technologies. It is made to allow construction projects to quickly access the market and gain a foothold in the construction industry[9].

The project’s inception involved the two Swiss research institutes, Empa campus and Eawag. The figure below shows the structure they have created that operates on a plug-and-play system where research and innovative modules are plugged onto any one of the three open platforms. Once research and development work has been completed the units can be easily dismantled as seen in Figure 5 below and new modules can be installed.



Figure 4: NEST platform with several modules plugged onto it

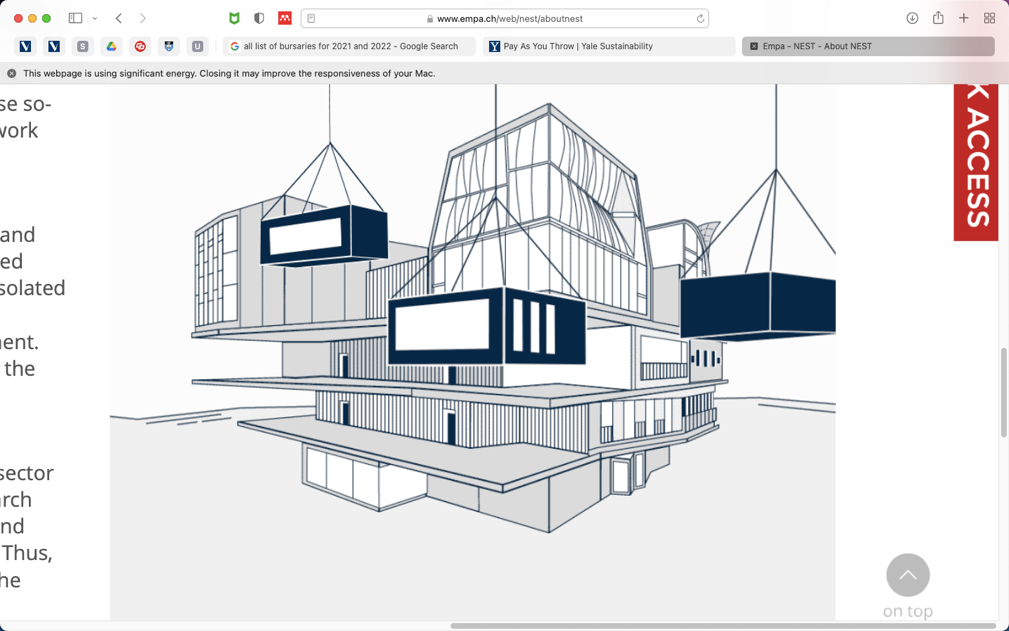


Figure 5: Model showing how the individual modules are placed and removed onto the platform

Using the NEST construction platform, the research institute can test and innovate on sustainable materials and research other lucrative developments that afford the environment ability to replenish its natural resources.

This development allows new building technology to be tested and used in real world scenario, thus accelerating market entry and saving time because it won’t be wasted on research and development of products that aren’t practical in real world contexts.

NEST prides itself in using modules that consist of materials that are fully reusable, recyclable, or compostable. It is referred to as circular construction [10]. These units act as material storage. They promote the need to think about how to deconstruct the building when we plan for it. The following image shows an excellent example of how they achieved this milestone in one of the apartments that is also a module within NEST.

A picture containing text, indoor

Description automatically generated

Figure 6: NEST apartment for two made of sustainable resources

The Empa campus has what they call Vision Wood unit and they developed using the latest technology developments in wood research and expertise in contemporary timber construction [10]. The marble looking image is what is used to build esthetic bathroom walls using recycled bottle lids. Some walls are made from drinking cardboard box, shredded and pressed and made into wall structure.

Lastly, the NEST has a water hub in its basement to collect all the water coming from showers, sinks, washing machines, etc. The water hub is shown in the building below.

A picture containing indoor, automaton

Description automatically generated

Figure 7: Water Hub in NEST basement used to clean recycle water

This facility is used for the following functions:

* Recovering grey water to be reused water on building scale,
* Recover energy from faces by using pellets that can be burned
* Recover nutrients to be used in agriculture to make fertilizer

#### Oxford University: VANS, PANS AND FANS

The University of Oxford has its own ambitions of reducing the greenhouse gas emissions of its staff and students across its colleges, thus reducing the carbon footprint of the entire university.

Two out of several attempts in reducing GHG emissions stood out. Popularly termed VANS, and PANS. FANS are not discussed here.

##### Vans

The university has purchased electric vehicles that transport staff and students around colleges and campuses and to run university errands. This was done in an effort to reduce carbon emissions caused by these vehicles throughout.

The image below shows the vehicles they bought and are using now.

A car parked next to a sign

Description automatically generated with medium confidence

Figure 8: Photograph of Electric Vehicles deployed in Oxford University

##### Pans

The university has plans to reduce the overall emissions caused by harvesting the food that is served on campus by introducing substitutions for GHG emitting foods. In a study performed by students in St Hilda’s, it was found that although beef accounted for 9% of total mains sold, they were responsible for 32% of total GHG emissions, however vegan main foods were sold over 54% but only accounted for 9% of the GHG emissions.

The total emissions caused to provide the food students are served in the dining halls is about 35000kg annually, which is almost equivalent to driving around the world 5 times!

The bar graph below shows the calculated emissions associated with main meals served in St Hilda’s college.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 9: Total emissions produced to provide food for St. Hilda's Colleg students

One of the solutions was to sell vegetable roast dinners, which is calculated to have accounted for 79% reduction in GHG emissions, 72% reduction in land use, 60% reduction in water consumption overall. Other alternatives were also introduced[11].

#### University of Pennsylvania: Living lab proposal

The University of Pennsylvania has its own version of Living lab termed ‘Research Community’. “This came after the attempt to name it living lab and majority of campus residents weren’t comfortable with the term laboratory,” says Melissa Brown Goodall, Senior Director of Environmental Innovations Initiative at University of Pennsylvania. The university overlaps into the city. There are twelve schools throughout Philadelphia, which makes living labs integration between the city and campuses inevitable[12].

An opportunity was offered for proposals to be brought forward and a seed funding was offered. Incentives were given to proposals that involved multiple faculties involved and those which had public facing outcomes. Some selected are listed below.

1. Growing together:  
   A community-based initiative for greening and nutrition
2. Laboratorio para Apreciar La VIda y El Ambiente - Mar (LAVA-Mar):

The program trains local school students to become scuba divers. They collect data when Pen researchers cannot be there to do it.

1. Public Health, Cities, and Climate Crisis  
   This is a course designed to address climate related solutions.

#### The University of Cape Town: Pee-recycling on campus

This innovative idea’s inception was in the Chemical Engineering Department in UCT. The project, led by Associate Professor Dyllon Randall, is the development of a fertilizer-producing urinal and urine bio-brick process. He won the South African Institution of Chemical Engineers (SAIChE) 2021 Innovation Award for it[13].

Urinals found in men’s bathrooms of the commercial, industrial, and institutional sectors of society use plenty of water to flush. One paper, conventional urinal use 1.5 to 5 gallons per flush (gpf) or 5.67 to 18.9 liters of water [14], many of which are inefficient and outdated. It is estimated that reducing the number of times a single urinal is flushed to fewer than three times could save about 18ML of water annually for the university[15].

The University of Cape Town is said to require 3 600 kg of fertilizer for its sports fields. This is good news considering that 6 700 kg of fertilizer can be produced hypothetically from the urine collected in waterless urinal systems [15]. The following bar graph shows the percentages of total percentage nutrients present in urine.

Graphical user interface, application

Description automatically generated

Figure 10: Graph showing nutrients found in urine in percentages.

This is convenient information because it shows just how much potential there is in making fertilizer from human urine. The diagram following this is a prototype first deployed in one of the engineering building during the pilot process in which male adults came regularly to submit their urine.

A screenshot of a toilet

Description automatically generated with medium confidence

Figure 11: diagram of the first version of waterless urinal at UCT

The proposed UCT urine treatment process is shown below, in which the urine content is separated into the nutrients in Figure 10 and used appropriately to make the fertilizer.

Graphical user interface

Description automatically generated with medium confidence

Figure 12: The UCT urine Treatment Process

Some of this fertilizer is to be used within the UCT premises/sports grounds and some commercially made available. Some niche liquid fertilizer is also produced.

This project has also been adopted in industry, first in the headquarters of Exxaro in Centurion, South Africa, developed by Growthpoint properties. The nine-story building also included a waterless urinal for women with some other low-water flush urinals[16].

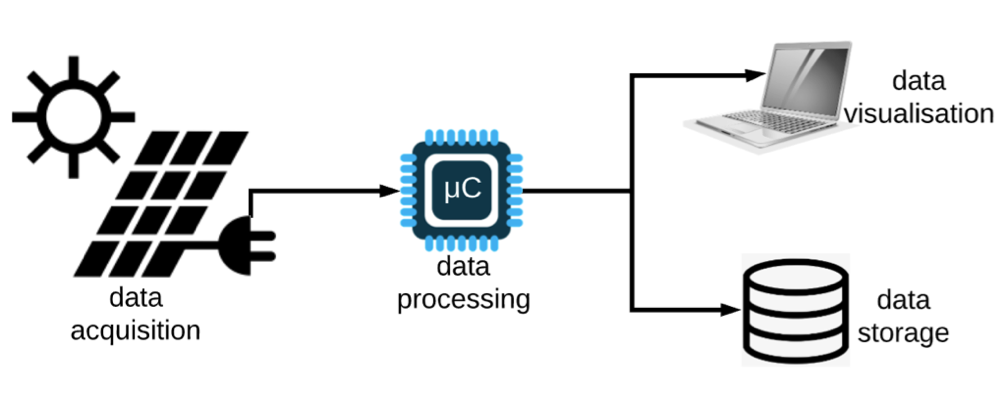
## Solar Monitoring Systems

### Overview

Solar Energy is considered by many to be the future of electricity supply for the world. Considering it’s not a firm type of energy source, it is still a reliable clean energy that can substitute fossil fuels during sunny days (and nights when stored in batteries). Since PV systems are widely adopted throughout the world, it is important to understand how much they contribute towards energy generation so that their reliability can be determined, and further studies done to improve their performance efficiency.

### Taxonomy of Solar PV Monitoring System

PV systems monitoring can be subdivided into four levels, viz. data acquisition, processing, visualization, and storage [17]. The following figure illustrates how each of the three integrate to form a complete system. The arrows show the flow of information between these subparts. Some solutions will exclude data storage depending on the needs of the client/user.



The data acquisition level involves the use of sensors to gather data which is essential for all other parts of this system. Thereafter the data is stored in auxiliary devices such as data loggers and processed through microcontrollers. The relevant information is then sent to storage facilities via wired or wireless technologies. The same data will be sent to a designed software tool for visualization on lcd screens or monitors[17].

The processing can be performed using microcontrollers or computer back-end code, depending on the information required. Examples of communication technologies appropriate for sending and receiving data include LoRa, Wi-Fi, Bluetooth or GSM, depending on the distance of the PV panels from the computer workstation[17].

### Data acquisition

Any type of monitoring requires measuring of variables by some means. In this case, sensors such as temperature sensors and pyranometer are used to capture real time meteorological data. PV panel information is necessary, too, if its performance is to be determined. PV systems come in different sizes, depending on which energy needs they’re supposed to meet. Ansari et al. 2021 [17] discusses the various electrical and environmental parameters monitored under large-scale and small-scale solar PV systems. They are summarized in the table below.

Table 1: Classification of data for two types of PV systems

|  |  |  |
| --- | --- | --- |
| **Solar PV System** | **Parameters** | |
| **Electrical** | **Environmental** |
| **Small scale** | Panel output voltage  Panel output current  Inverter output voltage  Inverter output current  Load output voltage  Load output current | Irradiance  Panel temperature  Humidity |
| **Large scale** | Array output voltage  Array output current  Grid voltage  Current to and from grid  Grid impedance | Irradiance  Array temperature  Speed of wind  Humidity  Air pressure |

For some perspective, the following two figures will show what a small-scale solution looks like versus a large-scale PV system. The size of the solar PV systems determines the variables of interest that the user is interested in.

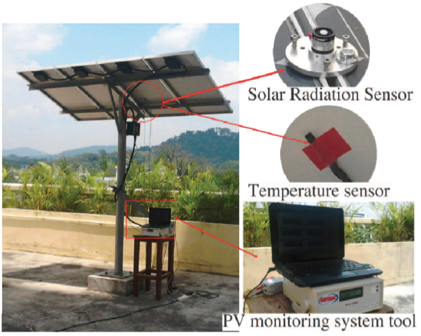


Figure 13: Example of small-scale PV system [18]



Figure 14: South Africa's large scale PV solar plant [19]

The PV plant on the right is developed by Norway’s Scatec Solar in the Northern Cape. According to the article, the country aims to produce as much as 17.8GWh of energy through solar only by 2030 [19] . This is the first project under the country’s Renewable Energy Independent Power Producer Procurement Program (REIPPPP). This program has intended to install about 4GW in its initial round of contracts.

#### Solar Irradiance

Irradiance is defined as the amount of light energy (from the sun) hitting a square meter of ground on earth per second[20]. It is a measure of how much energy can be expected to hit the PV panels installed. It is an important measure because it determines a peak of energy generation, even though the efficiency of solar panels isn’t very high.

The instrument normally used to measure solar irradiance is a pyranometer. This device only measures a portion of the solar spectrum; typically, from 0.285 to 2.8 µm. The measurement considers the angle in which the sun rays hit the pyranometer and returns its cosine value times the solar irradiance[21]. The following figure shows an example of a pyranometer.



Figure 15: LP02-L spectrally flat Class C Pyranometer manufactured by Hukseflux[22].

Other authors, for example, Anwari et al. 2011 [23], used a BPW photodiode instead of a pyranometer. The objective was to keep costs as low as possible, which is a plausible requirement for some small projects considering that the device shown in figure 15 above can range from R1000 to more than R50 000,00. [24].

#### Temperature

The temperature around the PV system is quite important for both small-scale and large-scale PV systems, as indicated in table 1 above. Especially considering the interesting relationship between the efficiency of solar panels and the ambient temperature. See figure below.

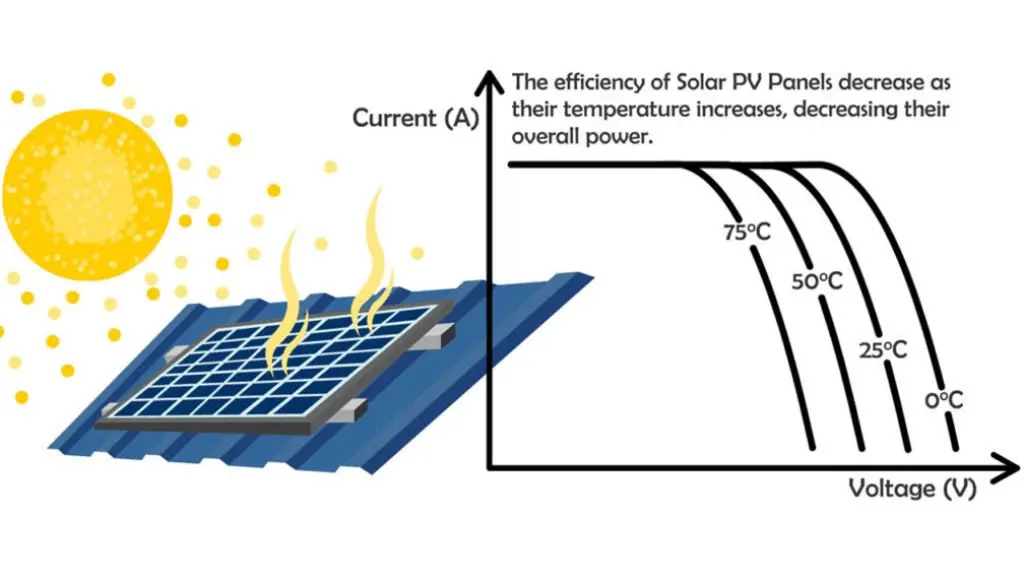


Figure 16: I-V characteristics of four similar PV panels at different temperatures[25].

It is evident from the makeshift plot shown in figure 16 above that the span of voltage is higher in lower PV panel temperatures versus higher temperature. Meaning that the efficiency of solar cell is negatively affected by increase in degrees Celsius. Forero et al. 2006 [26] used an NTC thermistor, which is a thermally sensitive resistor for which the resistance changes predictably and precisely as the core temperature of the thermistor increases over operating temperature range[27].

Other authors make use of a thermocouple for temperature measurement. For example, E.E. Van Dyke et al. 2005 [28] used a cromel-alumel (type K) thermocouple for temperature sensing measurements. This device has a much wider temperature measurement range (-200°C to 1250°C) compared to the thermistor which ranges from -50°C to 250°C[29]. The following figures show circuit diagrams for negative temperature coefficient thermistor and thermocouple:

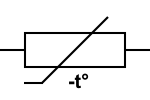


Figure 17: NTC Thermistor (IEC standard) [27]

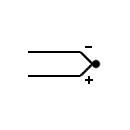


Figure 18: Circuit Symbol for Thermocouple [30]

#### Voltage, Current and Power

In order to draw I-V characteristic curve for PV panels as shown in figure 16 above, both current and voltage require to be measured as solar energy is being converted to electricity. This information is typically accessed in circuits through devices such as hall effect transducer for current, and voltage transducer for voltage measurements [28]. Power calculations only require the product of current and voltage across the device of interest. The formulae below show three different ways in which the power of the device depending on which information is available.

In the above mathematical representations, P is power calculated in Watts, I is current in Amperes, V is voltage calculated in volts.

### Data processing

The processing of data is the backend programming of the PV system monitoring solution. In this part of the solution, choices such as the measurement rate, the resolution, scale, transmission of data, etc. E.E. Van Dyke et al. 2005 [28] created a I-V sequencer system, which has several switches to select between resistors R1-Rn, which in turn allows current into the connected conductor.

Diagram

Description automatically generated

Figure 19: schematic of the I-V sequencer system

In figure 19 above, the PC is the computer, DAS stands for data acquisition system, A is the Hall effect transducer, V is the voltage transducer, the many R’s make the resistor array, PVn is thephotovoltaic array, S’n is the load relay array, and lastly, Sn is the module relay array [28].

The DAS collects data every 15 minutes, using the devices named in the above paragraph, and arranged in the circuit according to Figure 19. In this specific system, each module’s I-V data are stored in separate files for storage in text format.

For each of the data points collected by the circuit above. The irradiance is measured with each pair of current and voltage so that should the clouds cover suddenly during measurement, the irradiance can be offset. The developer used the Hewlett Packard’s Visual Engineering Environment to control the measurement of all signals and the behavior of instruments.

### Data visualization

This part of the system is usually the one that clients care most about. It is about representing the information collected from the PV system, processed through custom circuits, software or using a microcontroller, and finally displaying it on a monitor or through some other convenient method to the client.

Makbul Anwari et al. (2011) [23] used LabView to capture and analyze the data from PV monitor hardware. The analyzed data was later sent to clients via email. This is one innovative way in which the user can view the relevant information regarding their PV system.

Of course, this information needed to be presented visually on LabView as well [23], and as such the software related had to have five groups or pages, namely: main, history, e-mail, statistics and lastly the settings and calibration page. This software, however, was mostly available to the developer, and not the user.

N. Forero et al. (200) [26] also used LabView software to visually represent the data collected from the PV system, however, these charts were made available to the user’s computer screen. The following screenshot shows the IV characteristics and PV array of a typical PV solar plant.

Graphical user interface

Description automatically generated

Figure 20: Front panel of VI developed for displaying the IV characteristic graph and PV array chart side by side.

Using the IV Sequencer system shown in Figure 19, E.E. Van Dyke et al. 2005 [28] compared the performance of several PV panels and plotted several interesting facts about his findings for three different PVs. Figure 21 shows the energy that is generated by three different PV modules per day for each month and the average solar irradiance for that month while Figure 22 shows the efficiency comparison of three different modules over a period of 12 months.

A picture containing diagram

Description automatically generated

Figure 21: Average energy generated per day for each month for the period shown.

Chart, line chart

Description automatically generated

Figure 22: Monthly operational efficiencies of three different PV modules.

These two plots can be useful for users who are considering to purchase one or another PV module type and are wondering what is their money’s worth and what performance they can expect for the module they can afford.

### Data storage

There are three different data storage options for safekeeping the data is being measured and collected from PV panels. These are relational database (SQL), no database, and …

# Methodology

## Overview of Approach

This chapter of the report discusses the method followed in addressing the solution. The overall problem to address is the need for a visual representation of PV panel performance displayed on a monitor. Considering that this project was executed in the middle of the Covid-19 global pandemic, the slight modification to objective is that the solution should be available for everyone concerned on their own custom computer, anywhere in the world with internet access. Additionally, the solution will involve weather data, and cost savings as part of the dashboard. A comparison of two different methods of solving this problem is discussed in subsequent headings.

Since the solution involves capturing and interpretation of data using computer programming, the proposed storage is using relational databases systems. Alternatively, the project prototype could take advantage of cloud storage services provided by component manufacturers, such as SolarEdge and Weatherlink.

## How to address research objective

In the literature review, there exist two types of solutions to PV performance representation. The electronic type and the software type. Each one comes with its own advantages and disadvantages, and this section compares the two available (but not exhaustively) in significant depth of detail. Finally, a decision is made about which approach is best for the specific scenario on campus and the size and scale for which this project is envisioned. Following this chapter, the appropriate solution is presented in detail.

### Software: Webapps and desktop applications

The first one is software. It is purely code, taking advantage of general-purpose computers that individuals already possess. The user acquires a relevant download file for their specific machine and the application runs on their computer using its hardware resources. This type of software solution could be built using languages such as Swift, C#, or Java and a framework such as Vapor, .NET or JavaFX.

Another way to use software is to take advantage of web development technologies that run web application that are designed to execute on the user’s computer browser. The user simply clicks a URL, and it reroutes them to the hosted site showing desired results. This type of software solution could be developed using a combination of HTML5, CSS3, and JavaScript.

Regardless of which type of software is chosen as best for this type of project, data collection, processing and storage is performed in the same manner. In the context of this dissertation, a PV system is installed, and data is collected by the manufacturer through their devices into the data storage servers. Manufactures provide API keys to subscribers. Developers use APIs to access the data and use it in their software, however they built it, using methods prescribed for their chosen development languages and development environment.

By choice, developers could go the extra mile of building a back-end structure for the software and decide to store the data in a cloud storage provider of their choice. This might be done for financial reasons, or if they don’t trust the manufacturer to properly maintain the data, or if they’d like to store the data in a different format than that which the manufacturer provides.

### Electronic solution

Another solution that is popular with smaller projects is electronic. Some code is used with the integration of microcontrollers. This approach is called an IoT (Internet of Things). It would involve some small electronic components on (PCB) printed circuit board, with data collected using sensors and a microcontroller integrating the collected data with the desired display.

Popular choices of microcontrollers include Arduino, Raspberry Pi, and STM32F microcontrollers. Each of these devices have their own strengths over the other, nonetheless they fundamentally solve the same problems using similar principles, except some difference in development environments and programming languages, but those are just tools.

Data storage is very similar to that described above, except that collection is handled by the developer, since IoT systems are made up of sensors that send data to host computers, which send data to cloud storage providers of choice. This data will then be made available for visual representation on monitors or LCD screens appropriate to meet the needs of a specific client.

### Logical reason for choosing the method / approach

Since there are two different approaches to solving one problem. A comparison is made between the software solution and the IoT solution and put sided by side one can properly see which one works best for their appropriate situation. The table below summarises the categories that aided in the decision for the context and scope of this projects.

Table 2: Categories used to decide between IoT and pure SW solution

|  |  |  |  |
| --- | --- | --- | --- |
| Category | IoT / uC | software | Winner and reason |
| Access Barrier | Need to access PV system in physical location. | Needs access to login details to use data. | SW |
| Login details are available, but PV system is installed further from developer. |
| Expertise | Trained formally for the past 5 years and built some projects. | Formal learning of fundamentals + self-taught advanced software development concepts. | IoT |
| Formal training from the ground up and can almost guarantee a smoother development. |
| support | There is some support from colleagues, but small online community. | There is lots of assistance from Q&A platforms like stack overflow. | SW |
| Most available assistance online with large community means high chance of success. |
| Time barriers | There is long waiting for components after ordering. | Little to no waiting, unless need assistance from manufacturer for HTTP cycle, which might or might be short. | SW |
| Immediate availability of tools for development means smoother process. |
| Learning curve | Already know electronic development, but not  about hosting and data storage | Steeper learning curve of deep HTTP concepts and processes. | SW |
| The more knowledge the better. So, smother development process. |
| Development time | Unpredictable, but can be slow because of access barriers and travelling distance to PV system and/or unavailability of components. | Unpredictable, learning new web concepts necessary, however learning time is measurable. | SW |
| Predictable learning makes job easier. |
| Development environment | Not intuitive, lots to learn. Complex IDEs. | User friendly development environment. | SW |
| Easy to use IDEs provide better experience coding. |
| Hosting | Unfamiliar hosting services, and most are expensive. | Familiar hosting services and some are free. | SW |
| The lower the cost of development the better. |
| Maintenance | The need to travel to PV location to fix malfunctioning/broken parts | Debugging and testing are all done remotely on computer of virtual machine. | SW |
| Travelling costs hinder the progress and make budget unpredictable. |
| External assistance | Some worked examples on internet, but community is small. | Large community so plenty of examples, work done, coaching sites and large community online. | SW |
| Software offers the most hope for completion since developer is not left alone. |

As seen on the table above, the software solution wins 8-2 by counting how many times it comes up as a better choice in each category than the IoT based solution. Therefore, the decision is final. The rest of this paper will discuss the development of software-only solution.

Since there are two versions of software possible for this type of problem, the best one needs to consider which is best for the display on the monitor. Without considering metrics to decide as was done previously, it will be decided that web development technologies provide the best possible solution for the context of this project.

#### An overview of the chosen software approach is discussed below:

The development of the software will involve using the HTML5 as a markup language, CSS3 to style webpages and JavaScript to incorporate interactivity into the application. The acquisition of data and its utilization into the software will involve an HTTP request-response cycle in which application programming interfaces (API) are used to access specific information into the client side of development.

Upon the absolute necessity of custom data storage facilities, backend development of the software will use Django Framework, which is Python based. PostgreSQL is the relational database management system of choice because it is open source.

### What to prototype

For this project, it is necessary to demonstrate a working prototype of a webapp/website with using a working URL, with predictable and accurate data represented using intuitive and reliable charts representing information that PV system installers and panels users care about most.

### How to build prototype

The development process will involve writing code on an IDE such as Visual Studio Code, using HTML, CSS, and JavaScript. Hosting is currently done on GitHub because it’s free. The data will be acquired into the development using APIs and the [XMLHttpRequest method to retrieve data as a JSON file and processed using object oriented programming concepts such as the dot notation which accesses instances of classes or objects.](https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest)

### How to gather data

It is undoubtedly necessary for PV systems and weather stations to be installed. A manufacturer will be chosen for a weather station. The requirement is that the manufacturer should provide data collected in intuitive formats, which make it easy for developers to utilize the data collected for their own custom software.

PV panels will feature an inverter that converts AC voltage generated into AC low voltage. Some manufactures of such devices also gather conversion data and efficiency information and store it in their custom cloud storage facility. It is important for this project that the manufacturer provide the data to developers for use in their own software solutions.

# Design

## Overview

This chapter covers the development process of designing the dashboard to be displayed on a monitor. As discussed in the methodology, the chosen approach is a web-based software that is made to be run on the browser. This type of software is referred to as a webapp. The chapter also describes the process of deciding on the visualization parameters best fit for the product.

## System design

The prototype, although built on developer’s desktop computer, will be used to display the data measured in the PV panels and weather systems using chart libraries imported in software. The following diagram shows the position of the system’s component parts in the proposed prototype design and how they integrate to complete the flow of data measured from panels on site to the client, which in most scenarios will be a simple browser supporting front-end styling used and charts embedded in the solution.

Icon

Description automatically generated

Figure 23: Data flow from measurement to display

The detailed steps of the solution parts are as follows:

Firstly, ensuring that the data is captured by designated weather station and PV systems installed is key to the operation of the software. The information is sent by the devices in question to the manufacturer who organizes relational databases and makes the data available on their servers and grant developers custom API keys so that web-APIs can be used.

The development of the software will take place on a host computer on campus, or remotely, using JavaScript as a base front-end language because it has several key libraries and methods such as Fetch, and XMLHttpRequest, which allow the user to interact with data stored in the cloud.

The host computer will then render the software whose HTML5 markups were styled using CSS3 and to a hosting site, initially GitHub pages. Hosting involves the use of another server (computer) to make the information available to the public. GitHub pages is best used for static and dynamic sites, and it is trusted since it is run by a reputable company.

Doing this allows third party users to access the site using a provided URL. A web solution is best for ensuring that the software is most accessible by various computer owners because it means that the operating system that they’re running is irrelevant since web apps are designed to be run on a browser.

The campus display part of the project is that a computer monitor on campus student labs can be hoisted on a wall, connected to the internet, and displaying the dashboard on a screen-maximized browser to observers in the building.

## Software design

This section covers experience of the user as they operate the webapp, and the underlying backend details that allow the site to perform the way it does. Some explanations will be given for components that make up the user interface, i.e., how the user will interact with the software’s buttons to use its existing features. The front-end design communicates how the entire software is designed to take advantage of the user’s visual and sensory feedback, while the backend entails all the data related manipulations that make meaningful interactions possible.

### Traceability Matrix

Table 3: Traceability Matrix to discuss the different requirements for the software solution

|  |  |
| --- | --- |
| **USER REQUIREMENTS TRACEABILITY MATRIX** | |
| **Subsystem: Report Analytics** | |
| **User Req. ID** | **Requirement Description** |
| R.U.001 | The interface of the software should be accessible to everyone. |
| R.U.002 | The software should be responsive on all computing devices such as laptop computers, tablets, and smartphones in their browsers. |
| R.U.003 | The interactivity of the software should be quick and smooth, not sluggish. |
| R.U.004 | The different pages of the software should be intuitive when navigating. |
| R.U.005 | The appeal of the interface needs to be at attractive. |
| R.U.006 | The cost of using the software solution should be considerate to financially underprivileged users. |

### Requirement Analysis

#### Analysis of R.U.001

“The interface of the software should be accessible to everyone.”

Accessibility refers to the practice of making the software usable to a vast majority of people, including those who use mobile devices, those with slow networks and most especially disabled people. This requirement means to make the software able to accommodate those people.

|  |  |  |
| --- | --- | --- |
| **Technical Req. ID** | **Technical Requirement Description** | **Derived From** |
| R.X.001 | All users must be able to navigate and be able to tell what the content of each page entails while on the app, without any assistance from a third party. | R.U.001 |

**Verification**

To verify this requirement, the interface should be checked against guidelines described in the W3C accessibility guidelines for web content. The use of accessibility tools for color contrast is encouraged in the design of the software, including the use of validator to check for the syntactical correctness of the written code, especially in markup languages that make use of the DOM (document object model).

#### Analysis of R.U.002

“The software should be responsive on all computing devices such as laptop computers, tablets, and smartphones in their browsers.”

This requirement covers the ability to open and operate the software on several types of browsers and operating systems and the software structure itself being supported on plenty of different form factors of pre-determined operating systems.

|  |  |  |
| --- | --- | --- |
| **Technical Req. ID** | **Technical Requirement Description** | **Derived From** |
| R.X.002 | The software should be supported on several browsers on different operating systems and on at least 3 different screen sizes and dimensions. | R.U.002 |

**Verification**

All users should be able to successfully open and operate the software on their mobile and desktop devices. This requirement will be verified when all screen sizes can clearly view the charts in the dashboard without the need to scroll left or right to view some edges of certain pages.

#### Analysis of R.U.003

“The interactivity of the software should be quick and smooth, not sluggish.”

This requirement speaks about the speed of the software when the user interacts it. The performance being reasonable and how fast navigating through the webpages is important to users because it can be discouraging when every two pages need to load content before users can actually use them.

|  |  |  |
| --- | --- | --- |
| **Technical Req. ID** | **Technical Requirement Description** | **Derived From** |
| R.X.003 | The software should be fast when navigating. The user should not wait more than two seconds for the next page to appear after the click. | R.U.003 |

**Verification**

To verify this requirement the user should be able to move from one page of the software to another with only a reasonable loading time. A reasonable loading time should be about four seconds. A good approach could be to use multiple mobile devices to test the same pages on several different platforms so that responsiveness could be developed for the weakest among them.

#### Analysis of R.U.004

“The different pages of the software should be intuitive when navigating.”

Here we are dealing with how easy it is to move from the home page to other webpages essential to software solution, and how to navigate back to a previously viewed pages.

|  |  |  |
| --- | --- | --- |
| **Technical Req. ID** | **Technical Requirement Description** | **Derived From** |
| R.X.004 | It must be clear how to move from one page to another on the software, and how where to click to access specific pages. | R.U.004 |

**Verification**

To verify this requirement, several different types of users should be asked to review how the intuitive it is to use the software. This way its usability is determined. All links on the navigation bar and the body of webpages should be descriptive and obvious that they are to be clicked on, designed to look like buttons that users are familiar with.

#### Analysis of R.U.005

“The appeal of the interface needs to be at attractive.”

This is simply about the appeal of the software and its buttons and navigation bar. This addresses how aesthetically pleasing the interface is. Supposedly this should also indicate how much users want to interact with the overall software.

|  |  |  |
| --- | --- | --- |
| **Technical Req. ID** | **Technical Requirement Description** | **Derived From** |
| R.X.005 | The software visual features must be beautiful to behold, and the user should be satisfied with interacting with them. | R.U.005 |

**Verification**

This requirement's verification is not very straightforward. The opinion of users is important to determine that enough users appreciate the features made available. One verification could be using blind testing of third parties and then use surveys to determine how users see the software.

#### Analysis of R.U.006

“The cost of using the software solution should be considerate to financially underprivileged users.”

This deals with how much the average user should expect to use with respect to bandwidth while using the software. Not every user can afford to spend a lot of money/data/airtime on the browser like many do on social media platforms.

|  |  |  |
| --- | --- | --- |
| **Technical Req. ID** | **Technical Requirement Description** | **Derived From** |
| R.X.006 | While using the software, the design should be considerate of users who do not have a lot to spend on data. The average bandwidth of using viewing charts on the software for an hour should be less than 30MB. | R.U.006 |

**Verification**

This ties back to the performance of the software because when the developer should be using programming language’s in-built functions and instead of importing a library from some server or using wireframes that hog resources for features that can be handled with the traditional language. For instance, it would be bandwidth friendly to use vanilla JavaScript instead of a library such as jQuery. Therefore, it is advantageous to ensure that the developer remains mindful of access to the software as far as cost is concerned.

### Design Choices

In this section, two different solutions will be provided for how to meet the requirements for this submodule. Each of them is basically about how to code effectively so that the whole project is easier to complete, and so that the architecture is easy to understand for the developer. This will inevitably translate to coherent links and navigation of the app, making the whole experience that much smoother for the user. Design choices is about making the right thing and making sure it performs best for the end-user.

#### Discussion of two solutions to meet requirements

##### MVC architecture

Use frameworks as a boilerplate code for addressing the development process. A framework is a pre-written software structure for developers to build on top of. It is composed of files and directories in a structure that follows a set of design principles frequently encountered in the application[31].

Frameworks that use a MVC type architecture, for example Django, separate the database queries and front-end visuals using models and views respectively, while both elements are prompted by the controller element, which the user unknowingly interacts with while using the application[32].

A framework uses templates. The principle behind this is that it is used to generate logic code from one language to markup scripts recognized by the browser/emulator. The advantage of this is that the developer can control what would otherwise be several files that would have recycled code from one script to another into just one template just updated using compiled/interpreted language that iterates between queried datapoints from the database.

Since every framework is consistent with the structure and philosophy it follows, one can be certain that that the next person to take on this project and attempt to recycle the code will understand it fully. They might only ever need to learn the logic of their predecessor, but never how that predecessor made the different scripts communicate with each other for efficient flow of code. This can prove to be very valuable to the project life cycle, knowing that everyone who has learned the framework will know exactly how it operates. This means the team of developers only need to worry about the logic in the code they write in each script, and not the entire underlying structure.

The front-end operates as a standalone application, requesting data from the backend, processing and displaying it. This process described is known as separation of concerns[32]. When each part of the program operates independently, has a clear, and singular purpose, collaboration is easy, and maintenance is less painful.

##### Developer’s Custom Code Structure

Attempt to create one’s own structure of code and using front end technologies such as HTML, CSS and JavaScript. The redundancy of framework files is unnecessary in the early stages of the development processes. The developer should focus on design and implementation of features of the software instead of a structure that might not be necessary in the end. This will speed up the development process and reduce overhead of set-up and customization to the specific use case[33].

The major advantage of this approach is the flexibility of the user to create their own structure of code and philosophy that they might understand better than one that’s predetermined by developers of the framework of choice. Therefore, it gives the developer a little more flexibility when it comes to decision making with respect to the flow of content, data and how they’re presented to the user.

With that in mind, a major drawback is that anyone who takes over the project will have to learn the entire structure. The philosophy needs to be explained and the uncertainty of their understanding of how the different parts of the system fit together will determine how productive this new developer can be in improving the software.

### Figures of Merit

Table 4: Figures of merit used to compare the use of a front end framework and making one's own programming architecture.

|  |  |  |
| --- | --- | --- |
|  | Use Frameworks | Create own structure / no frameworks. |
| Cost | The learning process is quite the cost as far as this is concerned. Once that is out of way, the implementation reduces the overall cost during the process. | Each developer can wing their own philosophy with respect to structure and approach, which makes this approach costs less to the development overall. |
| Technical maturity | Since frameworks are developed by professionals of the industry, the technology that came as a result is advanced, using familiar resources and concepts and templating philosophies for others to just plug into their projects while keeping the need to create a new structure inconsequential to the project. | The team will need to decide on an architecture that best fits the project’s needs. One would argue that this is a good approach, irrespective of how imprudent this approach has proven to be in many projects before, however risk of one of the developers being underproductive because they couldn’t quite get the design structure is there. So very immature decision for large projects. |
| Ease of manufacturing | This option is almost as easy as ‘plug-and-play’. Most of the code is already in place to use and the developer’s logic only is necessary. This makes the development easy. | The sad reality of this option is that developer needs to start everything from scratch. Sometimes as scratch as the first <html> tag of the project. The project life cycle can be unpleasantly long and tedious. |
| Implementation | Frameworks require users to set-up and customize, thus increasing overhead and making the process that much longer. Could be overkill for small projects. | Implementing this option can be as easy as opening the simplest text editor, such as notepad, and typing tags. It's straight-forward when the developer is familiar with the language in used to write code. |
| Ease of testing | Since the philosophy used is familiar to most developers, anyone responsible for testing could just develop unit tests for the logic code made by others. There is certainty that the architecture works as it should. Maybe only concern could be how it was used during implementation. | Testing for this approach is hard. The user needs to test everything, from the architecture of choice and how the different scripts interact together to how the actual logic is used in the code blocks. |
| Reliability | The consistency of frameworks makes this choice undeniably reliable. Every developer knows exactly what they are doing when working with the different files within the directory tree. | There are several risks associated with building a project from the ground up, and a lot of breaking points can surface in the process. The project is just prone to misbehavior since there is plenty of opportunity for bugs. |
| Maintenance cost | Separation of concerns makes it easier to make changes that do not impact the rest of the software. One can work on only one element at a time with the confidence that they will not affect other parts of the application much. It creates accountability in the team as well since responsibility can be delegated. | As the software code grows, files will grow along with it, filling up with unmaintainable “spaghetti code”. Following the breadcrumbs of each logic becomes that much harder because there is no clear definition of where logic code or database queries should be. Each developer is just doing what they believe is best for them, thus the maintenance time is a struggle. |

### Solution Decision

#### Overview of software pages

The project is small, with the most useful webpage supposed to be only one (views.html), it is therefore imperative to simplify the code and go with the intuitive structure of setting up HTML markup files, styling them using pure CSS and then adding interactivity using vanilla JavaScript. The stylesheet and JavaScript files are named style.css and script.js respectively.

The figure below shows how the different pages of the software are connected. HTML files describe the individual webpages and how they are connected and arranged on the navigation bar. The shaded arrows indicate connection between html pages. An unshaded arrow indicates that the file from which it comes is a JS script or CSS file for the html file to which the unshaded arrow points. Only the shaded dots mean a connection between lines.

Diagram

Description automatically generated

Figure 24: Webpages of software and related code files

The software when operated, starts on the home page, named index.html. It is named this because hosting services look for and index.html or default.html and treat it as a home page. The home bar is merely a navigation bar with links to other useful pages. All other html files besides that one has a navigation bar at the top in a header tag and a footer at the bottom. As can be seen in the figure above, each html page is styled by its own CSS file.

The views.html page shows charts and that is the core to this software solution, having two scripts linked to it: one for weather data graphs and another for solar performance graphs. The entire page is styled by the view.css.

The resources.html file shows further tabulated data related to the charts plotted in the views.html. The about page shows information related to the developer and the contact page allows the user to send an email to the developer.

#### HTML discussion

HTML stands for Hypertext Markup Language, which refers to a standard language used to create content for webpages made to be displayed in web browsers. The following piece of code shows the different sections that make up the general structure of webpages in the software described in figure 24 above. Generic JS and CSS files are named script.js and style.js and within the head tags it is shown how an html file connects to these code files.

<!DOCTYPE *html*>

<html *lang*="en">

<head>

<script *src*="script.js"></script>

<link *rel*="stylesheet" *href*="style.css">

</head>

<body>

<header>

<nav>

*<!-- navigation bar linking pages of website together -->*

</nav>

</header>

<main>

*<!-- text, graphics, and charts -->*

</main>

<footer>

*<!-- some additional information -->*

</footer>

</body>

</html>

The comments, in grey font color and wrapped around “<!--" and “-->” are indicative of what type of content goes into the header, main and footer sections of the webpage. Notice how appropriate semantic elements are used instead of generic <div></div> tags which provide no obvious meaning.

Semantic tags are used to improve discoverability on the internet because when appropriate elements are used, search engines such as Bing and Google can parse content more easily and feed into their algorithm, thus helping the site surface better when searched. Some other semantic elements, such as the <button> tag, are used to aid those using accessibility technologies to better navigate the software using parsers.

#### Cascading Style Sheets

In considering the styling of the webpages, pure CSS is used as shown below. In the example, the elements being styled are the body and html tags. The font-family attribute chooses the fonts to be used within the tags. Since different browsers support some fonts and others don’t, three different fonts are selected so that the browser can at least have a font it supports. Any browser will pick a font from the left to the right, depending on which is supported first.

html, body

{

font-family: Arial, Helvetica, sans-serif;

margin: 0;

padding: 0;

min-height: 100vh;

}

Libraries such as Bootstrap or Tailwind could have been used to easily move elements in appropriate positions, and choose other features like colors, weight, padding, etc., to make the site more visually appealing. Surely this would make it easy to meet the requirement R.U.005: (“The appeal of the interface needs to be at attractive”). In contrast, doing this would compromise the requirement, R.U.003: “The interactivity of the software should be quick and smooth, not sluggish,” and R.U.006: “The cost of using the software solution should be considerate to financially underprivileged users.”

Note that drawing resources from online libraries hands over control of the interactivity of the website to the developers of that library and its developers, thus making the speed of the site unpredictable. The bandwidth used to draw those resources is beyond developer control, too. The conclusion here is therefore to use pure CSS and stick to it throughout the development cycle of this software, despite the difficulty of the language and its legacy style of compilation.

#### JavaScript

JavaScript is the brains of each webpage. It is a programing language used to manipulate the DOM (document object model). The DOM refers to the html elements used in the document to create content which becomes visible on the browser window. Each element’s attributes and properties are changed to the developer’s preference. The language has event handlers which can be used to program what happens when the specific events occur, or when the user interacts with the webpage in specific ways. This is the reason why ‘JavaScript’ and ‘interactivity’ are often found in the same sentence when JS is introduced.

# Results

## Experimental setup

The hardware of this experiment is deployed in Kraaifontein. Both the weather station and the PV modules. The PV modules setup is positioned like the diagram below on the ground. The displayed information was the energy production per module on November 6th, 2021.

Table

Description automatically generated

Figure 25: Hardware Setup of PV modules in Kraaifontein

Each of the modules have within them inverters that convert the direct current into alternating current. The setup the setup has sensors that record the temperature, wind speed and the humidity, since these meteorological variables affect the performance of the panels the significantly.

The data collected from the PV modules is pushed to database servers hosted by the companies, SolarEdge and Weatherlink. The data is stored into relational databases and information made available through the internet in several formats such as json and csv.

The two IDEs were used for the completion of the software discussed in the design section of this document. The first was Visual Studio Code. This is the environment in which most of the programming happened. All HTML, CSS, and JavaScript files were made using this IDE. The second one was pgAdmin, which handled the data storage and backend side of the software. This was used to create all tables and servers for the relational database.

## Data Acquisition

The appropriate data downloaded and used for this software is extracted from the sources shown in the discussed above. SolarEdge provided the information in csv files, which were downloaded on Google’s Chrome browser, and pushed into the database as required. For the purposes of this project, the extraction process will be repeated weekly, and database and charts updated appropriately until.

## Visualization

Several options were provided for the development of this webapp as can be read from the design section of the document. Graphing libraries such as Chart.js, ApexCharts, Canvas.js but enetually the developer, due to time constraints settled with MS Excel charts.

The setup described above measures parameters such as module current, voltage, power, and energy. Each of the 27 different modules have their own readings and stored in the appropriate columns in the relational database. One might be interested in the voltage and current through each panel. Since their product results in power, its graph might also be fascinating to a client.

The data is captured with a resolution of a single unit of measurement per hour. This is the same resolution in which the data is stored in the database and represented in the charts displayed below.

Chart, line chart

Description automatically generated

Figure 26: Plot of Temperature vs Wind Speed measured by SolarEdge environmental sensors

The plot above represents the data: wind speed, and temperatures experienced by PV cells during operation. The data points are an hour away from each other. The left y-axis represents the temperature (°C) in ranges and that on the right represents the wind speeds (km/h). The x-axis range represents the week starting from the 29 October to 5 November 2021.

The data from which the plot was represented is shown in the screenshot in figure 27 below. The simple SQL command returns only the data for the 5th of November 2021 from a table named ‘meteorological’.

Table

Description automatically generated

Figure 27: SQL command which returns data from specific date

In the chart below, the data represented is the power and energy produced from only one of the modules. Notice that the two data values were represented using two different resolutions, even though they were measured on the same day. Chart, line chart

Description automatically generated

The SQL script screenshot below shows the power and energy for module one where both variables were measured, and no empty cells are found. Notice that only 21 times were the energy and power recorded at the same time on the 6th of November 2021.

Graphical user interface

Description automatically generated with medium confidence

Figure 28: SQL script showing energy and power side by side from two tables.

If observed correctly, the shape of the graph provided in figure 27 looks like solar irradiance on any sunny day. This is because the panels will generate electricity only when the sun is shining. This makes absolute sense because the sun’s energy is the one that is converted to electrical energy, and thus without it none can be produced.

The following plot comes from the Solcast’s database for Solar Irradiance. Each peak represents a new day starting from 1st of November 2021. Notice how similar it would look if time of day was plotted plotted against power and/or energy.

Diagram

Description automatically generated with low confidence

Figure 29: Solar Irradiance from Solcast platform

## Simulation

For viewing the actual dashboard with these charts updated weekly, click on the link below.

<https://tefoam.github.io/PVdashboard/>

# Conclusion

## Conclusions

This paper discusses the development of a ‘PV Living Lab’, which refers to a dashboard displayed on a monitor in the Menzies 3rd floor foyer at the University of Cape Town. This product should be designed to show historical and live PV performance, weather conditions.

The proposed solution is a web application software designed to be operated on a browser by users. This approach is chosen so that the information is available to everyone who intends to view it without even being on campus, or even in the Menzies building at UCT.

This software was developed using web technologies including HTML5, CSS3, JavaScript. Data acquisition depends on the installed experimental PV system, whose inverter sends data to manufacturer servers and stored on their relational databases. The performance data and weather forecast information are recorded by the system, extracted and used by the developer’s software solution using web APIs and embedded into the front-end of the software. This behaves just as well as the backend design would.

Since this was a project for a student’s dissertation, the time allocated to develop this project in parallel with writing a report about was limited. The developer’s ability to recap development technologies or learn enough to build the most convenient or cost-effective software solution was thus limited.

The budget for this project was also limited, hence one could not trial with several manufacturers nor participate in the decision for which inverters and weather station would be used. The hardware installed was entirely decided for which made the steepened the learning curve associated with completing this project.

The time constraint limited the quality of work produced heftily. Most user requirements were met, and the development process was completed with all planned features executed and put into the software. The solution is appealing and shows all required charts as it anticipated.

## Future work

Such as is true for any development process of building a software solution, several versions’ releases are encouraged which incorporate performance improvements, bug fixes and introduction of new features. This software also needs further improvements on its code base.

Making the software open source would make the code base public, thus allowing for more collaborators to participate in the development which can potentially make the software even better and accommodate more browsers and screen sizes.

If the budget for this project is increased it would make the improving and maintaining this software much easier because it would allow the purchase of licenses of much better looking and performant chart libraries, and afford the yearly subscription of a domain name, and a much more reliable hosting service.

# List of References

[1] P. Mudholkar and M. Mudholkar, “A Beginner’s Guide to Git and GitHub,” *IJSRD-International Journal for Scientific Research & Development|*, vol. 5, no. 03, 2017.

[2] “Greatest South African Natural Disasters - Letsolo,” *Letsolo Water and Environmental Services CC*, 2021. https://lwes.co.za/2021/03/25/greatest-south-african-natural-disasters/ (accessed Sep. 24, 2021).

[3] J. Evans, R. Jones, A. Karvonen, L. Millard, and J. Wendler, “Living labs and co-production: University campuses as platforms for sustainability science,” *Current Opinion in Environmental Sustainability*, vol. 16. Elsevier B.V., pp. 1–6, Oct. 01, 2015. doi: 10.1016/j.cosust.2015.06.005.

[4] “The Living Lab Methodology Handbook.”

[5] Wikipedia, “Living lab,” 2021, Accessed: Oct. 08, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Living\_lab

[6] L. Verhoef and M. Bossert, *The University Campus as a Living Lab for Sustainability A Practitioner’s Guide and Handbook*. 2019.

[7] “Pay As You Throw | Yale Sustainability.” https://sustainability.yale.edu/priorities-progress/materials/pay-as-you-throw (accessed Oct. 08, 2021).

[8] “Yale Sustainability Plan 2025.pdf | Powered by Box.” https://yale.app.box.com/s/xagi53f5qvpklkv31zdebe8rilgt3b6x (accessed Oct. 08, 2021).

[9] “Empa - NEST - About NEST.” https://www.empa.ch/web/nest/aboutnest (accessed Oct. 08, 2021).

[10] “NEST EXPLORING THE FUTURE OF BUILDINGS”.

[11] N. Gray, J. Ethics, and E. Rep, “A Guide to Understanding and Reducing the Environmental Impact of Food at St Hilda’s”, Accessed: Oct. 08, 2021. [Online]. Available: www.weforum.org.

[12] “Post-summit events | UCT SDG Summit.” http://www.sdgsafricasummit.uct.ac.za/sas/post-summit-events (accessed Nov. 07, 2021).

[13] “National innovation award for UCT engineer’s urine project | UCT News.” https://www.news.uct.ac.za/article/-2021-06-03-national-innovation-award-for-uct-engineers-urine-project (accessed Oct. 12, 2021).

[14] J. Martin, J. H. Conserve, and F. W. Clearinghouse, “BMP 5: WATER USE BY URINALS,” 2008.

[15] T. Chipako and D. Randall, “Urinals for water savings and nutrient recovery: a feasibility study,” *Water SA*, vol. 45, no. 2, pp. 266–277, Apr. 2019, doi: 10.4314/WSA.V45I2.14.

[16] “Student’s urine-recovery system for R600m building | UCT News.” https://www.news.uct.ac.za/article/-2019-07-03-students-urine-recovery-system-for-r600m-building (accessed Oct. 12, 2021).

[17] S. Ansari, A. Ayob, M. S. Hossain Lipu, M. H. Md Saad, and A. Hussain, “A review of monitoring technologies for solar pv systems using data processing modules and transmission protocols: Progress, challenges and prospects,” *Sustainability (Switzerland)*, vol. 13, no. 15, Aug. 2021, doi: 10.3390/su13158120.

[18] IEEE Staff, *2019 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*. IEEE, 2019.

[19] “South Africa’s First Large-Scale PV Plant Now Online | Cleanenergyauthority.com.” https://www.cleanenergyauthority.com/solar-energy-news/south-africa-large-scale-pv-online-091713 (accessed Oct. 30, 2021).

[20] “NASA - SOLAR IRRADIANCE.” https://www.nasa.gov/mission\_pages/sdo/science/Solar%20Irradiance.html (accessed Oct. 31, 2021).

[21] “Pyranometers: What You Need to Know.” https://www.campbellsci.co.za/blog/pyranometers-need-to-know (accessed Oct. 31, 2021).

[22] “LP02-L: Pyranometer.” https://www.campbellsci.co.za/lp02-l (accessed Oct. 31, 2021).

[23] M. Anwari, M. M. Dom, and M. I. M. Rashid, “Small scale PV monitoring system software design,” in *Energy Procedia*, 2011, vol. 12, pp. 586–592. doi: 10.1016/j.egypro.2011.10.079.

[24] “pyranometer - Google Shopping.” https://www.google.com/search?q=pyranometer&client=safari&hl=en&sxsrf=AOaemvKqKktZu7UBQhZHf9-5FhoRxLsmGg:1635684364485&source=lnms&tbm=shop&sa=X&ved=2ahUKEwiKuomM1\_TzAhUXQkEAHbDjCUcQ\_AUoAnoECAEQBA&biw=1152&bih=629&dpr=1 (accessed Oct. 31, 2021).

[25] “Datasheet Values: Rating of a Solar Panel | Coulee Limited.” https://couleenergy.com/datasheet-values-rating-of-a-solar-panel/ (accessed Oct. 31, 2021).

[26] N. Forero, J. Hernández, and G. Gordillo, “Development of a monitoring system for a PV solar plant,” *Energy Conversion and Management*, vol. 47, no. 15–16, pp. 2329–2336, Sep. 2006, doi: 10.1016/j.enconman.2005.11.012.

[27] “NTC Thermistor | Resistor Types | Resistor Guide.” https://eepower.com/resistor-guide/resistor-types/ntc-thermistor/# (accessed Oct. 31, 2021).

[28] E. E. van Dyk, A. R. Gxasheka, and E. L. Meyer, “Monitoring current-voltage characteristics and energy output of silicon photovoltaic modules,” *Renewable Energy*, vol. 30, no. 3, pp. 399–411, Mar. 2005, doi: 10.1016/j.renene.2004.04.016.

[29] “Difference between Thermocouple & Thermistor (with Comparison Chart) - Circuit Globe.” https://circuitglobe.com/difference-between-thermocouple-and-thermistor.html (accessed Oct. 31, 2021).

[30] “Thermocouple Symbols.” https://www.electrical-symbols.com/electric-electronic-symbols/thermocouple-symbols.htm (accessed Oct. 31, 2021).

[31] “What is Frameworks? [Definition] Types of Frameworks.” https://hackr.io/blog/what-is-frameworks (accessed Nov. 07, 2021).

[32] “Separation of concerns - Wikipedia.” https://en.wikipedia.org/wiki/Separation\_of\_concerns (accessed Nov. 07, 2021).

[33] “Overhead (computing) - Wikipedia.” https://en.wikipedia.org/wiki/Overhead\_(computing) (accessed Nov. 07, 2021).

# 

# Appendices

## HTML

1. The following script contains the html file for making the dashboard.

<!DOCTYPE *html*>

<html *lang*="en">

<head>

<meta *charset*="UTF-8">

<meta *http-equiv*="X-UA-Compatible" *content*="IE=edge">

<meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

<link *rel*="stylesheet" *href*="stylesheets/navbar.css">

<link *rel*="stylesheet" *href*="stylesheets/views.css">

*<!-- chartjs CDN -->*

<script *src*="https://cdn.jsdelivr.net/npm/chart.js@3.5.1/dist/chart.min.js"></script>

<script *src*="scripts/charts.js"></script>

<title>Document</title>

</head>

<body>

*<!-- navtigation bar -->*

<div *id*="navbar">

<a *href*="views.html"><img *src*="images/logo.png" *alt*="uct-logo" *id*="logo"></a>

<div *class*="topnav">

<a *class*='link' *href*="about.html">About</a>

<a *class*='link' *href*="resources.html">Resources</a>

<a *class*='link' *href*="contact.html">Contact</a>

<a *id*="active" *class*='link' *href*="index.html">Views</a>

</div>

</div>

*<!---- CONTENT ---->*

*<!-- weather station -->*

<div *class*="PV\_Container">

<hr>

<a *href*="solar.html">

<h3 *class*="label">Demonstrate the Viability of a PV System</h3>

</a>

<hr>

<div *class*="figures">

<div *id*="temp-wind">

<img *src*="images/plots/Wind\_temp 5Nov.png" *alt*="wind and temperature plots">

</div>

<div *id*="irradiance">

<img *src*="images/plots/solar irradiance.png" *alt*="solar irradiance">

</div>

</div>

</div>

*<!-- PV system -->*

<div *class*="PV\_Container">

<hr>

<a *href*="weather.html" *class*="weather">

<h3 *class*="label">PV System Performance</h3>

</a>

<hr>

<div *class*="figures">

<div *id*="energy-power">

<img *src*="images/plots/power and energy.png" *alt*="power and energy">

</div>

</div>

</div>

</body>

</html>

## CSS

1. The following script shows the CSS used to style the same dashboared.

.PV\_Container{

background-color: rgb(240, 253, 255);

}

.label{

padding: 15px 0 12px 0;

margin: 0;

font-size: 2em;

width: 100%;

text-align: center;

}

.figures{

height: 20%;

padding: 5px 15px 15px 15px;

gap: 10px;

}

hr{

margin: 0;

}

#temp-wind{

display: flex;

justify-content: space-evenly;

text-align: center;

}

#irradiance{

display: flex;

justify-content: space-evenly;

text-align: center;

}

img{

width: 70%;

}

#energy-power{

display: flex;

justify-content: space-evenly;

text-align: center;

}

The rest of the code is found in the Github repository in the link below.

## GitHub link

<https://github.com/TefoAM/PVdashboard/>

EBE Faculty: Assessment of Ethics in Research Projects

Any person planning to undertake research in the Faculty of Engineering and the Built Environment at the University of Cape Town is required to complete this form before collecting or analysing data. When completed it should be submitted to the supervisor (where applicable) and from there to the Head of Department. If any of the questions below have been answered YES, and the applicant is NOT a fourth year student, the Head should forward this form for approval by the Faculty EIR committee: submit to Ms Zulpha Geyer ([Zulpha.Geyer@uct.ac.za](mailto:Zulpha.Geyer@uct.ac.za); Chem Eng Building, Ph 021 650 4791).Students must include a copy of the completed form with the final year project when it is submitted for examination.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name of Principal Researcher/Student:** | | Tefo Motaung | | **Department:** | ELECTRICAL ENGINEERING |
| **If a Student:** | YES | **Degree:** | BSc Eng Mechatronics | **Supervisor:** | Richard Larmour |
| **If a Research Contract indicate source of funding/sponsorship:** | | | |  | |
| **Research Project Title:** | | Design a “PV Living Lab” for the UCT Menzies Building | | | |

Overview of ethics issues in your research project:

|  |  |  |
| --- | --- | --- |
| **Question 1: Is there a possibility that your research could cause harm to a third party (i.e. a person not involved in your project)?** | YES | NO |
| **Question 2: Is your research making use of human subjects as sources of data?**  If your answer is YES, please complete Addendum 2. | YES | NO |
| **Question 3: Does your research involve the participation of or provision of services to communities?**  If your answer is YES, please complete Addendum 3. | YES | NO |
| **Question 4: If your research is sponsored, is there any potential for conflicts of interest?**  If your answer is YES, please complete Addendum 4. | YES | NO |

If you have answered YES to any of the above questions, please append a copy of your research proposal, as well as any interview schedules or questionnaires (Addendum 1) and please complete further addenda as appropriate.

I hereby undertake to carry out my research in such a way that

* there is no apparent legal objection to the nature or the method of research; and
* the research will not compromise staff or students or the other responsibilities of the University;
* the stated objective will be achieved, and the findings will have a high degree of validity;
* limitations and alternative interpretations will be considered;
* the findings could be subject to peer review and publicly available; and
* I will comply with the conventions of copyright and avoid any practice that would constitute plagiarism.

Signed by:

|  |  |  |
| --- | --- | --- |
|  | **Full name and signature** | **Date** |
| **Principal Researcher/Student:** | **Tefo Motaung** | 07 November 2021 |

This application is approved by:

|  |  |  |
| --- | --- | --- |
| **Supervisor (if applicable):** | **Richard Larmour** | 07 November 2021 |
| **HOD (or delegated nominee):**  Final authority for all assessments with NO to all questions and for all undergraduate research. | **Janine Buxey** | 07 November 2021 |
| **Chair : Faculty EIR Committee**  For applicants other than undergraduate students who have answered YES to any of the above questions. |  |  |

**ADDENDUM 1:**

Please append a copy of the research proposal here, as well as any interview schedules or questionnaires:

**ADDENDUM 2:** To be completed if you answered YES to Question 2:

It is assumed that you have read the UCT Code for Research involving Human Subjects (available at <http://web.uct.ac.za/depts/educate/download/uctcodeforresearchinvolvinghumansubjects.pdf>) in order to be able to answer the questions in this addendum.

|  |  |  |
| --- | --- | --- |
| 2.1 Does the research discriminate against participation by individuals, or differentiate between participants, on the grounds of gender, race or ethnic group, age range, religion, income, handicap, illness or any similar classification? | YES | NO |
| 2.2 Does the research require the participation of socially or physically vulnerable people (children, aged, disabled, etc) or legally restricted groups? | YES | NO |
| 2.3 Will you not be able to secure the informed consent of all participants in the research?  (In the case of children, will you not be able to obtain the consent of their guardians or parents?) | YES | NO |
| 2.4 Will any confidential data be collected or will identifiable records of individuals be kept? | YES | NO |
| 2.5 In reporting on this research is there any possibility that you will not be able to keep the identities of the individuals involved anonymous? | YES | NO |
| 2.6 Are there any foreseeable risks of physical, psychological or social harm to participants that might occur in the course of the research? | YES | NO |
| 2.7 Does the research include making payments or giving gifts to any participants? | YES | NO |

If you have answered YES to any of these questions, please describe below how you plan to address these issues:

**ADDENDUM 3:** To be completed if you answered YES to Question 3:

|  |  |  |
| --- | --- | --- |
| 3.1 Is the community expected to make decisions for, during or based on the research? | YES | NO |
| 3.2 At the end of the research will any economic or social process be terminated or left unsupported, or equipment or facilities used in the research be recovered from the participants or community? | YES | NO |
| 3.3 Will any service be provided at a level below the generally accepted standards? | YES | NO |

If you have answered YES to any of these questions, please describe below how you plan to address these issues:

**ADDENDUM 4:** To be completed if you answered YES to Question 4

|  |  |  |
| --- | --- | --- |
| 4.1 Is there any existing or potential conflict of interest between a research sponsor, academic supervisor, other researchers or participants? | YES | NO |
| 4.2 Will information that reveals the identity of participants be supplied to a research sponsor, other than with the permission of the individuals? | YES | NO |
| 4.3 Does the proposed research potentially conflict with the research of any other individual or group within the University? | YES | NO |

If you have answered YES to any of these questions, please describe below how you plan to address these issues: