Energy Internet of Things (E-IoT) Testbed for Smart Home Technologies and Power Grid Support

EE 496 Final Report
Fall 2021

Department of Electrical Engineering
University of Hawaii

Shanen Heu

December 5, 2021

Faculty Advisor: Yuanhang Xiao

Project Objectives and Criteria

Renewable energy has been on the rise in the past decade. From 2011-2020, renewable energy has grown 11% year-on-year in the United States. Also, in 2020 alone, renewable energy consisted for a fifth of the electricity generated in the United States as well. With a growing number of investors in renewable energy generation, growth is projected to skyrocket. From 2020-2050, renewable energy consumption is projected to rise by 8 quad. To put that into perspective, from 1950-2020, renewable energy consumption only rose by 9 quad. Looking closer into residential PV systems, it grew substantially as well. In 2005, residential PV systems generated only 27 MW of electricity. In 2020, residential PV systems generate 2,800 MW of electricity.

With the increasing amounts of renewable energy entering the power grid, it is creating a lot of instability, or randomness, in the power grid. This randomness is due to the unpredictable load demands of buildings, which can vary dramatically depending on its use. When a building is using renewable energy, it can sometimes cover the amount of energy consumed by the customer. However, renewable energy isn't always readily available and predictable as weather changes frequently week-to-week or even day-to-day. When there is not enough renewable energy being generated, the building now has to rely on the grid for energy. Thus, creating a load demand in the grid. Utility companies maintaining the grid must now accommodate for this sudden change in demand, but this accommodation can't happen so quickly and might even be too late. Now, imagine we have that same building and we have thousands of other buildings behaving in the same way. This creates a large amount of randomness in grid conditions. Randomness can cause a lot of issues including burnouts and, even worse, blackouts.

Internet of Things (IoT) devices are used to help reduce the randomness in grid conditions. Since energy companies have to find a way to keep grid conditions stable, these IoT devices help companies have access to real-time insights into the performance and behavior of the grid. By doing this, utility companies can more effectively keep a balance between supply and demand. Keeping the stability of grid conditions helps with unpredictable events and making the grid more cost effective.

There isn't much software that allows companies to test IoT devices and hinder the innovation and successful deployment of the devices. Due to the higher demand for renewable energy and given this issue, we want to design algorithms to tell how IoT devices should respond to randomness in grid conditions. Also, to develop a universal testbed to evaluate and demonstrate these algorithms.

This project collects and stores real-time data to IoT devices using a Smart Home Hub. The Smart Home Hub consists of two components; hardware and software components. For the hardware components, a RaspberryPi computer allows for WiFi and bluetooth connectivity while it also uses a USB to allow for connectivity of Zigbee and Z-Wave devices. For the software components, Home Assistant, Mosquitto MQTT, and Node-RED make up this section. Home Assistant allows for an online user interface where IoT devices are allocated and controlled

through the RapsberryPi computer. Node-RED is an automation software that, with the use of triggers, gathers the data from the IoT devices on Home Assistant using MQTT messages. From there, Node-RED stores the data collected in a database to be used for our simulation tool, EnergyPlus. EnergyPlus uses this real-time data to simulate “actual” building loads and solar outputs. EnergyPlus also uses forecasted data for the control algorithm. Controls implemented in Python, using PyTorch, makes a “control decision” and communicates this decision back to the Smart Home Hub via MQTT messages. Control algorithms are trained using data collected in the database or from other sources.

The task for this portion of the project was to supply our simulation tool, EnergyPlus, with real-time and forecasted weather data. By doing this, it will allow us to simulate building energy usage and solar production. This task is made up of six parts. The first part was to set up the Smart Home Hub and a few IoT devices. The second part was to find data sources that contain the proper field requirements for EnergyPlus. The third part was to develop code to make API calls from the chosen data sources and set them up for data extraction. The fourth part was to then extract the required fields for EnergyPlus from the data received. The fifth part was to take the extracted data and start constructing a file that EnergyPlus can read, i.e. .epw format. Lastly it was to Deploy the control algorithm on the hub using the EnergyPlus outputs.

Related Work and How it Differs

Currently, there are some devices and services that allow you to interact with your renewable energy systems. For example, some solar companies have a user interface that allows customers to see solar generation from their solar panel units. This interface also allows for users to adjust the angle of its tilt toward the sun so that their solar panel units can get optimal sunlight given the current weather conditions and the time of day. There are also energy consumption monitors that allow you to see how much power is being generated by the customer's renewable energy systems and also read the power consumption of different parts of the customer's homes. However, these devices cannot be used on a common interface. A lot of the time, devices must be of the same brand in order to be used together. On an energy company's end, this doesn't allow for clear insight on how to effectively manage grid conditions. It's hard to clearly know exactly how renewable energy is interacting with the grid. The lack of clarity in data still causes grid conditions to be unstable.

What makes the E-IoT testbed different from other related works is that instead of the customer only being able to read the power consumption of their home, they are able to help contribute, effectively, to the power grid as well. Renewable energy data will be readily available in real-time allowing energy companies to gain much more insight on how to better meet the demands of the grid. Let's say there is to be a high chance of rain and there would be no sun for the whole day. Let's also say that solar energy makes up a large part of the renewable energy homes generate in the area. It can be expected that homes like these will need to draw energy from the grid prompting more load demand for the energy company to meet to keep grid conditions stable. With a testbed like the one in this project, utility companies can simulate the conditions ahead of time allowing for better execution of meeting unexpectedness in grid conditions. Also, by having clear insights on the grid's potential activity, it makes the process of stabilizing the grid more cost effective.

The renewable energy in the grid from homes will have miniscule negative impacts on grid conditions by reducing the amount of noise coming from their renewable energy entering the grid and allowing for energy companies to utilize a customer's renewable energy to stabilize the grid. Of course, the customers of these IoT devices can opt-in or opt-out of helping with grid conditions, which adds a layer of privacy while using devices like these.

By having a testbed like this, given that there are many smart devices that aid in progression of smart buildings, it clears the way in evaluating such devices and their efficacy in certain conditions no matter the make or model. Also, it allows for innovation for IoT devices in the field of energy and allows for a more successful application of them.

Progress Toward Final Design

The setup for the physical home system consists of a RaspberryPi computer along with a USB. The purpose of the RaspberryPi is to allow for WiFi and bluetooth protocols. It is very affordable and its processing speeds are great for the circumstances required for this project compared to its price. The USB is to allow for Zigbee and Z-Wave device communications. The devices that were used for this portion of the project were the Aqara Temperature/Pressure/Humidity Reader and the Aqara Smart Plug. This was done using a software called Home Assistant, which will be discussed along with other software used, next.

Figure 1: *RaspberryPi Computer with USB*

Figure 2: *Aqara Temperature/Pressure/Humidity Reader*

Figure 3: *Aqara Smart Plug*

The setup for the software portion of the Smart Home Hub includes Home Assistant, Mosquitto MQTT, Node-RED, and Grafana. Home Assistant is an interactive user interface that displays the IoT devices in one place. Again, the RaspberryPi computer sends the data from the devices to Home Assistant using the wifi connected USB. We then use the Mosquitto MQTT software to send this data from Home Assistant to Node-RED. Node-RED is an automation software that uses triggers to enable the start different tasks. It allows us to program the computer to collect data every five seconds and have that data be stored into our database. In essence, Mosquitto MQTT is working hand-in-hand with Node-RED to gather real-time data. Mosquitto MQTT is one of the main communication protocols between smart home sites and the central computers. The purpose of Grafana gives us a visual representation of our data being fed into the database. In that way, we can identify and resolve any errors that may arise while the data collection process is taking place.

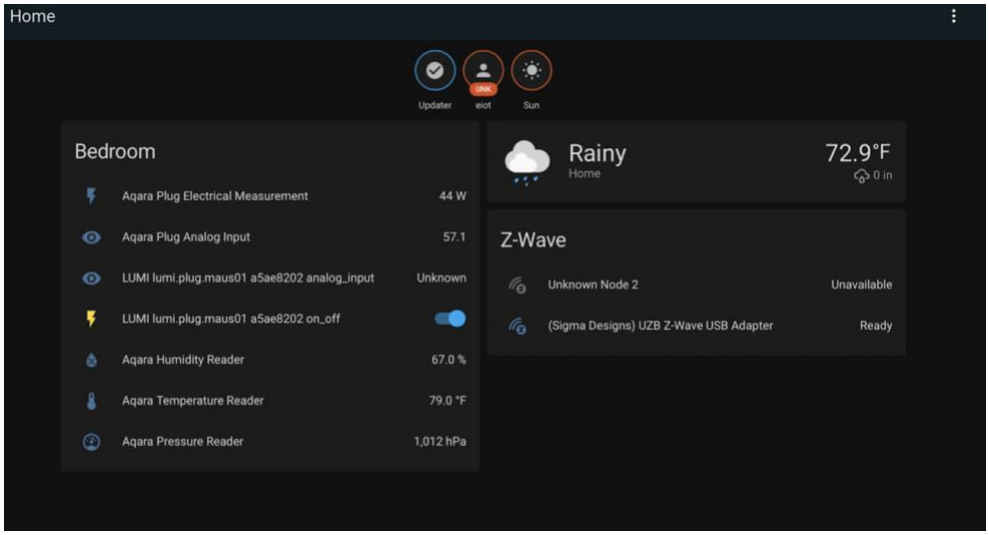

Figure 4: *Home Assistant*

Figure 5: *Node-RED*

Figure 6: *Grafana*

EnergyPlus is the simulation tool to help train and test our control algorithm. EnergyPlus uses real-time data to simulate “actual” building loads and solar outputs. It can also use forecasted data for its simulation, which we will use for the control algorithm.

With that being said, a specific part of this project that will be focused on is to identify and gather real-time and forecasted weather data, extract data and construct a file, i.e .epw, that meet the data field requirements needed by EnergyPlus.

During this process, it was found that there was no single weather data source that provided all of the fields that were needed for the .epw data file. The data sources that were chosen were OpenWeatherMaps and Solcast. The combination of these two data sources fulfilled

the requirements of the data fields required by EnergyPlus. Figure 7 shows the completed .epw file needed to run on EnergyPlus.


```
['2021', '11', '13', '16', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.49', '18.05', '73', '101400', '9999', '9999', '9999',
['2021', '11', '13', '17', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.55', '18.12', '73', '101400', '9999', '9999', '9999',
['2021', '11', '13', '18', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.17', '18.28', '71', '101500', '9999', '9999', '9999',
['2021', '11', '13', '19', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.65', '18.31', '69', '101500', '9999', '9999', '9999',
['2021', '11', '13', '20', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.1', '18.31', '67', '101500', '9999', '9999', '9999',
['2021', '11', '13', '21', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.35', '18.32', '66', '101500', '9999', '9999', '9999',
['2021', '11', '13', '22', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.5', '18.46', '66', '101400', '9999', '9999', '9999',
['2021', '11', '13', '23', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.59', '18.58', '66', '101300', '9999', '9999', '9999',
['2021', '11', '14', '0', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.54', '18.69', '67', '101200', '9999', '9999', '9999',
['2021', '11', '14', '1', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.53', '18.74', '67', '101200', '9999', '9999', '9999',
['2021', '11', '14', '2', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.37', '18.78', '68', '101200', '9999', '9999', '9999',
['2021', '11', '14', '3', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.12', '18.78', '69', '101300', '9999', '9999', '9999',
['2021', '11', '12', '5', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.1', '18.78', '68', '101500', '9999', '9999', '9999', '
['2021', '11', '12', '6', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.0', '18.92', '69', '101500', '9999', '9999', '9999', '
['2021', '11', '12', '7', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.82', '18.98', '70', '101600', '9999', '9999', '9999',
['2021', '11', '12', '8', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.59', '18.98', '71', '101600', '9999', '9999', '9999',
['2021', '11', '12', '9', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.33', '18.96', '72', '101700', '9999', '9999', '9999',
['2021', '11', '12', '10', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.98', '18.67', '73', '101600', '9999', '9999', '9999',
['2021', '11', '12', '11', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.79', '18.49', '73', '101600', '9999', '9999', '9999',
['2021', '11', '12', '12', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.6', '18.32', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '13', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.44', '18.15', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '14', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.37', '18.02', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '15', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.41', '18.05', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '16', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.52', '18.11', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '17', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.73', '18.27', '72', '101600', '9999', '9999', '9999',
['2021', '11', '12', '18', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.24', '18.32', '70', '101600', '9999', '9999', '9999',
['2021', '11', '12', '19', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.83', '18.09', '67', '101600', '9999', '9999', '9999',
['2021', '11', '12', '20', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.33', '17.87', '64', '101600', '9999', '9999', '9999',
['2021', '11', '12', '21', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.64', '17.76', '62', '101600', '9999', '9999', '9999',
['2021', '11', '12', '22', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.65', '17.94', '63', '101500', '9999', '9999', '9999',
['2021', '11', '12', '23', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.68', '17.95', '63', '101400', '9999', '9999', '9999',
['2021', '11', '13', '0', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.63', '17.86', '63', '101300', '9999', '9999', '9999',
['2021', '11', '13', '1', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.44', '17.72', '63', '101300', '9999', '9999', '9999',
['2021', '11', '13', '2', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.25', '17.68', '64', '101300', '9999', '9999', '9999',
['2021', '11', '13', '3', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.96', '17.72', '65', '101300', '9999', '9999', '9999',
['2021', '11', '13', '4', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.66', '17.73', '66', '101400', '9999', '9999', '9999',
['2021', '11', '13', '5', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.52', '17.75', '67', '101400', '9999', '9999', '9999',
['2021', '11', '13', '6', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.43', '17.66', '67', '101500', '9999', '9999', '9999',
['2021', '11', '13', '7', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.34', '17.64', '67', '101500', '9999', '9999', '9999',
['2021', '11', '13', '8', '0', '?9?9?9?9E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.19', '17.59', '67', '101500', '9999', '9999', '9999',
```


Figure 7: *Current .epw data fields implemented*

Alternate Solutions

If we look at the issue at hand, part of the reason for instability issues in grid conditions is due to the unpredictability of the load of buildings with renewables. In order to restabilize a grid system, energy companies need to produce enough energy in the grid in a short amount of time. This can be very unrealistic when it comes to modern processes of energy production and storage.

One alternative solution to this problem is to have power supplies readily available to be activated in a moment's notice. These power supplies will be high capacity batteries and will be placed strategically around a grid to achieve maximum efficiency. Due to the strategic placing and differentiating capacity sizes of the power supply, energy can be supplemented into the grid immediately without delay to achieve stability. The reasoning behind not relying on the residential/commercial production side is because of privacy. Households or businesses wouldn't want to hop on board with technology that uses their data because of the security and safety of their family or sensitive business information.

Other than that one alternative solution, there wouldn't be many others that can substitute it. This project is well thought out and is very applicable to the current issues at hand.

Previous and Concurrent Course Work

The classes at the University of Hawaii at Manoa have provided a plethora of knowledge to get people like myself acquainted with projects much like this. This project is filled with data analysis and coding requirements needed to get the tasks completed.

There are several classes offered at UH that are relevant to getting a high-level understanding of how to approach arrays and matrices. One of which is MATH 307: Linear Algebra and Differential Equations. In order to store the proper data that you would need to create a file for our simulation tool, the data points collected are needed to be stored in arrays. Knowing how to manipulate them into certain dimensions and understanding how the arrays can interact with each other would save you a lot of time and frustration. This class also helps to understand how to convert certain data points stored in the arrays into proper units that we would need for our data file.

The second class that is very relevant to a project like this would be EE 160: Programming for Engineers. This class is a fundamental piece in knowing how to create an algorithm in order to accomplish the task of retrieving the data in API calls and building the file needed for EnergyPlus. EE 160 offers fundamental knowledge on how to code. Even though it is taught in the programming language of C, it is still applicable in understanding the overall basic concept on how to code and approach problems like this. The most difficult part of EE 160 was the most applicable part of this project, which was understanding how to store arrays into objects. Also, learning how to change the dimension of arrays and calling on certain indices in an array were key parts in extracting data required for constructing our EnergyPlus data file.

The next applicable class would be EE 342: Probability and Statistics. A class like this helps you to understand the control algorithm and how it is supposed to work. Machine learning is heavily based on the probability of something happening by looking at the averages of datasets and how often they occur. By using this knowledge, you'll be able to locate a good dataset or datasets that would yield the best results for EnergyPlus. Remember, EnergyPlus is a key component to the testbed because it gives the control in PyTorch real-time building load and solar production. It also has the ability to use forecasted data to generate predicted building loads and solar production as well.

Future Work

In order to make this testbed as robust and reliable as possible, there is still a lot that can be done. There are a handful of missing fields that can contribute to the accuracy of renewable energy production. Like horizontal infrared radiation as an example. As time goes on, we are going to need to be able to include the missing fields that are directly filled with their missing value.

There is one more additional step that hasn't been completed that would be essential in knowing if the file can run on EnergyPlus. This step includes testing the file on EnergyPlus. By doing this, we know that our design of creating .epw files works and is able to take in real-time data from our home devices.

Instruments

The instruments used in this portion of the project consisted of physical and software instruments. The list of these instruments are described in the following:

Hardware:

- RaspberryPi Computer
    - Allows for WiFi and Bluetooth protocols and has ample amounts of computing power to carry out controls enabled by the machine learning control algorithms.
    - Sends and receives data through electric grid communications. Communications include grid status, carrying out utility requests, and making smart home responses.
- Nortek USB
    - This USB device allows for communication between Zigbee and Z-Wave devices. The USB is connected into the RaspberryPi computer.
- IoT Devices
    - Aqara T/H/P Sensor
        - A device that takes measurements of temperature, humidity and pressure. This device is linked to the Nortek USB.
    - Aqara Smart Plug
        - A device takes measurements of the amount of power consumed by a device connected to it and also is turned on and off online. This device is connected to the Nortek USB.

Software:

- Home Assistant
    - Provides a user interface for IoT devices. The measurements of these devices are displayed here.
    - Triggers and automated responses are coded in here and allows for communication (MQTT Messages) between the central computers and the Smart Home Hub.
- Mosquitto MQTT
    - A protocol allowing for communication between smart home sites and the central computers.
- Node-Red
    - An automation software that uses triggers to enable interaction between IoT computers, the use of control algorithms, and running IoT device simulations.
- Grafana
    - A visual representation of the data being collected and stored into the database

Data Collection and Analysis

For this portion of the project, data collection was the most important aspect. The bulk of the tasks required were to gather data from a source in order to meet the field requirements of EnergyPlus. However, this proved to be challenging on a couple of fronts. The required fields needed for EnergyPlus include the following along with the data sources that fulfilled them.

Table 1: *EnergyPlus Requirements with Data that meets the requirements*

EnergyPlus Requirements	Data
Month	OWM
Day	OWM
Hour	OWM
Minute	OWM
Data Source and Uncertainty Flags	OWM
Dry Bulb Temperature (C)	OWM, SOLCAST
Dew Point Temperature (C)	OWM
Relative Humidity (%)	OWM
Atmospheric Station Pressure (Pa)	OWM
Horizontal Infrared Radiation Intensity (Wh/m2) (If missing, calculated from *Opaque Sky Cover*)	(CAN BE CALCULATED FROM SOLCAST OPAQUE SKY COVER DATA)
Direct Normal Radiation (Wh/m2)	SOLCAST
Diffuse Horizontal Radiation (Wh/m2)	SOLCAST
Wind Direction (Degrees)	OWM
Wind Speed (m/s)	OWM
Total Sky Cover (Tenths of Coverage)	OWM
Opaque Sky Cover (Tenths of Coverage)	SOLCAST
Present Weather Observation	OWM
Snow Depth (cm)	OWM (Given in mm)
Liquid Precipitation Depth (mm)	OWM

The approach to this task was quite simple, which was to find a data source that met the requirements of EnergyPlus, as shown in Table 1. The source would have to provide free API calls and provide real-time and forecasted weather data. Also, this source had to give the data in either a .json or a .csv format so it can be easily converted to a .epw file, which is the file format that can be read by EnergyPlus. Last, but not least, I had to find the data readings for Hawaii's weather conditions.

Looking for a single data source that met all of the field requirements for EnergyPlus proved to be challenging. The first initial approach was to look at the National Oceanic and Atmospheric Administration's (NOAA) website to see if they had weather data we could retrieve that fulfilled the field requirements of EnergyPlus. The data looked promising as it provided ample amounts of different types of API calls to choose from. When doing a more thorough study of their data, it did not give us enough information as it fulfilled less than half of the requirements shown in Table 1 through a single API call. Later, it was found that the weather data wouldn't give real-time weather conditions.

The second approach was to look at blogs and forums to see what sources people recommended when stuck in the same predicament. OpenWeatherMaps, found on one of the forums, has a weather database dating back nearly 40 years and provides forecasted weather data as well, which was one of the fields needed. OpenWeatherMaps also provides current weather data as well and allows up to 60 calls a minute, which will allow for excellent real-time data. In addition, OpenWeatherMaps gave 7 days of forecasted weather conditions, which could be used for the control algorithm portion of this project. However, as you can see from Table 1, OpenWeatherMaps doesn't give the Direct Normal Radiation, Direct Horizontal Radiation, and Opaque Sky Cover required for EnergyPlus. After another week of searching, this led to the use of a second weather source.

The second weather source had to meet the same requirements as the first one. It had to give us real-time and forecasted data along with it being free. Solcast is mainly a solar weather data collection API source, which is exactly what we need. The missing fields are solar weather data requirements. However, there are some restrictions to this data. Solcast gives us only 13 API calls per day and gives only 3 days of forecasted data. This data comes in 30 minute increments. Because of the challenging nature of finding quality data sources for this project, it was decided to stick with this and try to create the file for the purpose of seeing if the control algorithms for this part of the project worked properly.

In Google Colab, a Python coding environment, API calls were made to gather the information from the sources chosen. From the data received, some of the required fields were then extracted into its respective arrays and placed in the specified order of a readable .epw file. Some of the data that was received from the data sources needed to be converted into the correct units as specified by EnergyPlus. This includes converting the units for date, time, pressure, weather ID, current weather conditions, horizontal irradiance, and direct normal irradiance. For the missing data not provided by our two sources, arrays were also needed to fill those gaps.

# File genera	by the Shiny Weather Data service: https://shinyweatherdata.com												
# Generated using Copernicus Climate Change Service information [2021]													
# Latitude (positive North): 21.25													
# Longitude (positive East): -158													
# Height above sea of the grid(s): 51													
# Aerodynamic roughness length of the grid(s) (averaged from ERA5 forecast surface roughness fsr parameter): 0.03													
# UTC offset (used for datetime_lst): -10													
#													
# long_name	Date and tim	10 metre wi	10 metre wi	100 metre w	2 metre rela	2 metre tem	Soil tempera	Soil tempera	Surface pres	Surface sola	Surface ther	Total cloud c	Total precipi
# unit: minut	minutes	m/s	deg	m/s	%	degC	degC	degC	Pa	W/m^2	W/m^2	%	mm
# source: NA	NA	ERA5	ERA5	ERA5	ERA5	ERA5	ERA5	ERA5	ERA5	ERA5	ERA5	ERA5	ERA5
datetime_lst	datetime	ws10	wdir10	ws100	r2m	t2m	stl3	stl4	sp	ssrd	strd	tcc	tp
1/1/20 0:00	##########	6.6	90	7.9000001	0.7	23.8	25.6	25.9	101200	0	354	0.09	0
1/1/20 1:00	##########	5.8	86	7	0.69	23.7	25.6	25.9	101200	0	351	0.02	0
1/1/20 2:00	##########	5.3	82	6.4000001	0.68	23.6	25.6	25.9	101200	0	349	0.02	0
1/1/20 3:00	##########	4.6	80	5.5999999	0.68	23.5	25.6	25.9	101200	0	360	0.03	0
1/1/20 4:00	##########	3.8	79	4.5999999	0.66	23.5	25.6	25.9	101100	0	360	0.02	0
1/1/20 5:00	##########	3	80	3.70000005	0.66	23.5	25.6	25.9	101100	0	355	0.03	0
1/1/20 6:00	##########	2.4	82	3	0.65	23.5	25.6	25.9	101200	0	372	0.06	0
1/1/20 7:00	##########	2	89	2.5	0.65	23.6	25.6	25.9	101200	0	373	0.07	0
1/1/20 8:00	##########	1.6	95	2	0.65	23.7	25.6	25.9	101200	38	375	0.11	0
1/1/20 9:00	##########	1.4	106	1.60000002	0.64	23.9	25.6	25.9	101300	202	372	0.09	0
##########	##########	1	126	1.10000002	0.62	24.1	25.6	25.9	101300	359	376	0.08	0
##########	##########	0.6	171	0.5	0.6	24.2	25.6	25.9	101300	553	367	0.14	0
##########	##########	0.7	186	0.69999999	0.57	24.6	25.6	25.9	101200	706	360	0.16	0
##########	##########	1.2	231	1.20000005	0.57	24.8	25.6	25.9	101100	729	354	0.1	0
##########	1/2/20 0:00	1.9	254	2	0.58	24.9	25.6	25.9	101000	691	359	0.05	0
##########	1/2/20 1:00	2.3	271	2.4000001	0.58	25	25.6	25.9	100900	602	357	0.1	0
##########	1/2/20 2:00	2.2	289	2.4000001	0.6	25	25.6	25.9	100900	458	357	0.2	0
##########	1/2/20 3:00	2	321	2.29999995	0.62	24.7	25.6	25.9	101000	268	356	0.4	0
##########	1/2/20 4:00	2.2	354	2.5999999	0.65	24.5	25.6	25.9	101000	68	359	0.42	0
##########	1/2/20 5:00	2.7	26	3.0999999	0.7	24	25.6	25.9	101100	0	359	0.16	0
##########	1/2/20 6:00	3.2	47	3.5999999	0.75	23.6	25.6	25.9	101100	0	362	0.07	0
##########	1/2/20 7:00	3.6	57	4	0.78	23.4	25.6	25.9	101100	0	369	0.04	0
##########	1/2/20 8:00	3.6	60	4.0999999	0.79	23.3	25.6	25.9	101200	0	372	0.05	0
##########	1/2/20 9:00	3.4	56	3.79999995	0.8	23.1	25.6	25.9	101200	0	371	0.04	0
1/2/20 0:00	##########	3	58	3.4000001	0.79	23.1	25.6	25.9	101200	0	375	0.41	0
1/2/20 1:00	##########	2.9	59	3.20000005	0.79	23	25.6	25.9	101100	0	369	0.25	0
1/2/20 2:00	##########	2.8	62	3.0999999	0.81	22.9	25.6	25.9	101000	0	381	0.25	0

Figure 8: *Sample .epw file (Shown in .csv format for reading purposes)*

The date was provided by the OpenWeatherMaps data source. It was given in the format of unix time, which is a series of numbers that was not easily comprehensible. This unix time provides the year, month, day, hour, and minute, which is all the information needed for the .epw file. To convert this, Python has a toolbox that contains a set of functions that allow for this conversion to be made. Once converted, we are then able to split these fields up into their respective arrays to then be placed in the .epw file.

```
#Converts unix time to readable data and time. Date and
for i in range(0,47,1):
  ts = int(hour_time_list[i])
  a = int(datetime.utcfromtimestamp(ts).strftime('%Y'))
  b = int(datetime.utcfromtimestamp(ts).strftime('%m'))
  c = int(datetime.utcfromtimestamp(ts).strftime('%d'))
  x = int(datetime.utcfromtimestamp(ts).strftime('%H'))
  y = int(datetime.utcfromtimestamp(ts).strftime('%M'))
  hour_year.append(a)
  hour_month.append(b)
  hour_day.append(c)
  hour_hour.append(x)
  hour_minute.append(y)
hour_year = np.array(hour_year)
hour_month = np.array(hour_month)
hour_day = np.array(hour_day)
hour_hour = np.array(hour_hour)
hour_minute = np.array(hour_minute)

hour_year = hour_year.reshape(-1,1)
hour_month = hour_month.reshape(-1,1)
hour_day = hour_day.reshape(-1,1)
hour_hour = hour_hour.reshape(-1,1)
hour_minute = hour_minute.reshape(-1,1)

hour_year = list(hour_year)
hour_month = list(hour_month)
hour_day = list(hour_day)
hour_hour = list(hour_hour)
hour_minute = list(hour_minute)
```

Figure 9: *Python code converting unix time to readable time and being organized into their respective arrays as needed by EnergyPlus .epw file*

Converting the pressure into its correct units was a simple task. OpenWeatherMaps provided this information as a percentage while EnergyPlus required this information to be in the thousands. To convert this, a multiplication by 100 would put that value into the thousand place.

```
#pressure list
hour_pres_list = []
for owm_pres_info in res_owm['hourly'][:47]:
 hour_pres_list.append([owm_pres_info['pressure']])
hour_pres_list = np.array(hour_pres_list)*100
```

Figure 10: *Conversion of the pressure data values from OpenWeatherMaps*

When converting the weather ID, there were a lot more steps involved then what was done when converting pressure to its correct units. OpenWeatherMaps had a different set of codes than EnergyPlus when trying to define different weather events that could occur. To tackle this issue and to avoid confusion, a comparison checklist was created in Google Sheets. It consisted of the list of EnergyPlus's weather codes and OpenWeatherMaps matching code next to it along with check boxes. These check boxes serve as a guide to make sure every code that could be converted was accounted for. However, not all of EnergyPlus's weather codes were able to be retrieved by using OpenWeatherMaps.

Weather ID Conversions

Occuances of Thunderstorm, Tornado, or Squall

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
Thunderstorm	211	Thunderstorm	0	1	☑
Heavy Thunderstorm	212	Heavy/Severe Thunderstorm	1	1	☑
		Report of a Tornado/Waterspout	2	1	☐
Squall	771	Moderate Squall	4	1	☑
		Waterspout	6	1	☐
		Funnel Cloud	7	1	☐
Tornado	781	Tornado	8	1	☑
		None	9	1	☐

Occurance of Rain, Rain Showers, or Freezing Rain

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
Light Rain	500	Light Rain	0	2	☑
Moderate Rain	501	Moderate Rain	1	2	☑
Heavy Intensity Rain	502	Heavy Rain	2	2	☑
Light Intensity Shower Rain	520	Light Rain Showers	3	2	☑
Shower Rain	521	Moderate Rain Showers	4	2	☑
Heavy Intensity Shower Rain	522	Heavy Rain Showers	5	2	☑
		Light Freezing Rain	6	2	☐
Freezing Rain	511	Moderate Freezing Rain	7	2	☑
		Heavy Freezing Rain	8	2	☐
		None	9	2	☐

Occurance of Rain Squalls, Drizzle, or Freezing Drizzle

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
		Light Rain Squalls	0	3	☐
		Moderate Rain Squalls	1	3	☐
Light Intensity Drizzle	300	Light Drizzle	3	3	☑
Drizzle	301	Moderate Drizzle	4	3	☑
Heavy Intensity Drizzle	302	Heavy Drizzle	5	3	☑
		Light Freezing Drizzle	6	3	☐
		Moderate Freezing Drizzle	7	3	☐
		Heavy Freezing Drizzle	8	3	☐
		None	9	3	☐

Occurance of Snow, Snow Pellets, or Ice Crystals

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
Light Snow	600	Light Snow	0	4	☑
Snow	601	Moderate Snow	1	4	☑
Heavy Snow	602	Heavy Snow	2	4	☑
		Light Snow Pellets	3	4	☐

Occurance of Snow Showers, Snow Squalls, or Snow Grains

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
Light Shower Snow	620	Light Snow Showers	0	5	☑
Shower Snow	621	Moderate Snow Showers	1	5	☑
Heavy Shower Snow	622	Heavy Snow Showers	2	5	☑
		Light Snow Squall	3	5	☐
		Moderate Snow Squall	4	5	☐
		Heavy Snow Squall	5	5	☐
		Light Snow Grains	6	5	☐
		Moderate Snow Grains	7	5	☐
		None	9	5	☐

Occurance of Sleet, Sleet Showers, or Hail

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
Light Shower Sleet	612	Light Ice Pellet Showers	0	6	☑
Shower Sleet	613	Moderate Ice Pellet Showers	1	6	☑
		Heavy Ice Pellet Showers	2	6	☐
		Hail	4	6	☐
		None	9	6	☐

Occurance of Fog, Blowing Dust, or Blowing Sand

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
Fog	741	Fog	0	7	☑
		Ice Fog	1	7	☐
		Ground Fog	2	7	☐
		Blowing Dust	3	7	☐
Sand	731	Blowing Sand	4	7	☑
		Heavy Fog	5	7	☐
		Glaze	6	7	☐
		Heavy Ground Fog	7	7	☐
		None	9	7	☐

Occurance of Smoke, Haze, Smoke and Haze, Blowing Snow, Blowing Spray, or Dust

OpenWeatherMaps	ID	EnergyPlus	ID	Position	Converted?
Smoke	711	Smoke	0	8	☑
Haze	721	Haze	1	8	☑
		Smoke and Haze	2	8	☐
Dust	761	Dust	3	8	☑
		Blowing Snow	4	8	☐
Mist	701	Blowing Spray	5	8	☑
		Dust Storm	6	8	☐
Volcanic Ash	762	Volcanic Ash	7	8	☑

Figure 11: *Checklist aiding in the process of weather ID conversions*

```
x = []

for a in range(len(id_list)):
  pres_weather = 999999999

  if (id_list[a] == 211 or 212 or 771 or 781): #Position 1
    if id_list[a] == 211:
      pres_weather = pres_weather - 900000000
    elif id_list[a] == 212:
      pres_weather = pres_weather - 800000000
    elif id_list[a] == 771:
      pres_weather = pres_weather - 500000000
    elif id_list[a] == 781:
      pres_weather = pres_weather - 100000000

  if (id_list[a] == 500 or 501 or 502 or 511 or 520 or 521 or 522): #Position 2
    if id_list[a] == 500:
      pres_weather = pres_weather - 90000000
    elif id_list[a] == 501:
      pres_weather = pres_weather - 80000000
    elif id_list[a] == 502:
      pres_weather = pres_weather - 70000000
    elif id_list[a] == 511:
      pres_weather = pres_weather - 20000000
    elif id_list[a] == 520:
      pres_weather = pres_weather - 60000000
    elif id_list[a] == 521:
      pres_weather = pres_weather - 50000000
    elif id_list[a] == 522:
      pres_weather = pres_weather - 40000000

  if (id_list[a] == 300 or 301 or 302): #Position 3
    if id_list[a] == 300:
      pres_weather = pres_weather - 6000000
    elif id_list[a] == 301:
      pres_weather = pres_weather - 5000000
```

Figure 12: *Python code of weather ID conversions*

Next, we needed to convert the horizontal irradiance and direct normal irradiance units that were provided by Solcast. We are given these values in every half-hour when EnergyPlus needs them in every hour. In order to convert this, the average of the values that fell in their respective hour was taken. Note that the array was divided in half to account for this leaving us with only 47 data points in the array. Due to this tactic of conversion, an adjustment to all arrays in all fields was needed when creating the .epw file.

```
hori_rad_list1 = [] # Listing all even position data (i.e. position 0,2,4,6...)
for owm_time_info in res_sol_fore['forecasts'][::2]:
 hori_rad_list1.append([owm_time_info['ghi']])
hori_rad_list2 = [] # Listing all odd position data (i.e. position 1,3,5,7...)
for owm_time_info in res_sol_fore['forecasts'][1::2]:
 hori_rad_list2.append([owm_time_info['ghi']])
hori_rad_list1 = np.array(hori_rad_list1)/2 # Divided by two
hori_rad_list2 = np.array(hori_rad_list2)/2
hori_rad_list = hori_rad_list1+hori_rad_list2 #Adding two lists together to get average per every hour
#print(hori_rad_list)

#Converting Direct Normal Irradiance to Wh/m2. Currently given in W/m2 (Follows same proceedure as global horizontal irradianc
dnorm_rad_list1 = []
for owm_time_info in res_sol_fore['forecasts'][::2]:
 dnorm_rad_list1.append([owm_time_info['dni']])
dnorm_rad_list2 = []
for owm_time_info in res_sol_fore['forecasts'][1::2]:
 dnorm_rad_list2.append([owm_time_info['dni']])
dnorm_rad_list1 = np.array(dnorm_rad_list1)/2
dnorm_rad_list2 = np.array(dnorm_rad_list2)/2
dnorm_rad_list = dnorm_rad_list1+dnorm_rad_list2
```

Figure 13: *Horizontal irradiance and direct normal irradiance conversion*

Lastly, the fields that we didn't have data for needed to have their respective "missing value" values required by EnergyPlus. These values are given to us by the EnergyPlus program. For most fields, the value to tell EnergyPlus that this particular field is missing is usually a series of nines. These nines were directly placed into the lists.

```
#Diffuse Horizontal Radiation
DifHorzRad_list = ['9999']*47 #There is no data for this from the current data we have. EnergyPlus allows ther
DifHorzRad_list = np.array(DifHorzRad_list)
DifHorzRad_list = DifHorzRad_list.reshape(-1,1)

#Global Horizontal Illuminance
GloHorzIllum_list = ['999999']*47 #There is no data for this from the current data we have. EnergyPlus allows
GloHorzIllum_list = np.array(GloHorzIllum_list)
GloHorzIllum_list = GloHorzIllum_list.reshape(-1,1)

#Direct Normal Illuminance
DirNormIllum_list = ['999999']*47 #There is no data for this from the current data we have. EnergyPlus allows
DirNormIllum_list = np.array(DirNormIllum_list)
DirNormIllum_list = DirNormIllum_list.reshape(-1,1)

#Diffuse Horizontal Illuminance
DifHorzIllum_list = ['999999']*47 #There is no data for this from the current data we have. EnergyPlus allows
DifHorzIllum_list = np.array(DifHorzIllum_list)
DifHorzIllum_list = DifHorzIllum_list.reshape(-1,1)
```

Figure 14: *Missing values being added to lists in their respective fields*

When all of the necessary conversions were done, we are not able to concatenate the lists together in EnergyPlus's specified order. As a result we get the file as shown in Figure 15.

```
['2021', '11', '13', '16', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.49', '18.05', '73', '101400', '9999', '9999', '9999',
['2021', '11', '13', '17', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.55', '18.12', '73', '101400', '9999', '9999', '9999',
['2021', '11', '13', '18', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.17', '18.28', '71', '101500', '9999', '9999', '9999',
['2021', '11', '13', '19', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.65', '18.31', '69', '101500', '9999', '9999', '9999',
['2021', '11', '13', '20', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.1', '18.31', '67', '101500', '9999', '9999', '9999',
['2021', '11', '13', '21', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.35', '18.32', '66', '101500', '9999', '9999', '9999',
['2021', '11', '13', '22', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.5', '18.46', '66', '101400', '9999', '9999', '9999',
['2021', '11', '13', '23', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.59', '18.58', '66', '101300', '9999', '9999', '9999',
['2021', '11', '14', '0', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.54', '18.69', '67', '101200', '9999', '9999', '9999',
['2021', '11', '14', '1', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.53', '18.74', '67', '101200', '9999', '9999', '9999',
['2021', '11', '14', '2', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.37', '18.78', '68', '101200', '9999', '9999', '9999',
['2021', '11', '14', '3', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.12', '18.78', '69', '101300', '9999', '9999', '9999',
['2021', '11', '12', '5', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.1', '18.78', '68', '101500', '9999', '9999', '9999', '
['2021', '11', '12', '6', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.0', '18.92', '69', '101500', '9999', '9999', '9999', '
['2021', '11', '12', '7', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.82', '18.98', '70', '101600', '9999', '9999', '9999',
['2021', '11', '12', '8', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.59', '18.98', '71', '101600', '9999', '9999', '9999',
['2021', '11', '12', '9', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.33', '18.96', '72', '101700', '9999', '9999', '9999',
['2021', '11', '12', '10', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.98', '18.67', '73', '101600', '9999', '9999', '9999',
['2021', '11', '12', '11', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.79', '18.49', '73', '101600', '9999', '9999', '9999',
['2021', '11', '12', '12', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.6', '18.32', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '13', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.44', '18.15', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '14', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.37', '18.02', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '15', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.41', '18.05', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '16', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.52', '18.11', '73', '101500', '9999', '9999', '9999',
['2021', '11', '12', '17', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '23.73', '18.27', '72', '101600', '9999', '9999', '9999',
['2021', '11', '12', '18', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.24', '18.32', '70', '101600', '9999', '9999', '9999',
['2021', '11', '12', '19', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.83', '18.09', '67', '101600', '9999', '9999', '9999',
['2021', '11', '12', '20', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.33', '17.87', '64', '101600', '9999', '9999', '9999',
['2021', '11', '12', '21', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.64', '17.76', '62', '101600', '9999', '9999', '9999',
['2021', '11', '12', '22', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.65', '17.94', '63', '101500', '9999', '9999', '9999',
['2021', '11', '12', '23', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.68', '17.95', '63', '101400', '9999', '9999', '9999',
['2021', '11', '13', '0', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.63', '17.86', '63', '101300', '9999', '9999', '9999',
['2021', '11', '13', '1', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.44', '17.72', '63', '101300', '9999', '9999', '9999',
['2021', '11', '13', '2', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '25.25', '17.68', '64', '101300', '9999', '9999', '9999',
['2021', '11', '13', '3', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.96', '17.72', '65', '101300', '9999', '9999', '9999',
['2021', '11', '13', '4', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.66', '17.73', '66', '101400', '9999', '9999', '9999',
['2021', '11', '13', '5', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.52', '17.75', '67', '101400', '9999', '9999', '9999',
['2021', '11', '13', '6', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.43', '17.66', '67', '101500', '9999', '9999', '9999',
['2021', '11', '13', '7', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.34', '17.64', '67', '101500', '9999', '9999', '9999',
['2021', '11', '13', '8', '0', '?9?9?9?E0?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9?9*9*9*9*9*9', '24.19', '17.59', '67', '101500', '9999', '9999', '9999',
```

Figure 15: *Completed .epw file for EnergyPlus*

Conclusion

This project meets the requirements of engineering standards well. Below is the requirements and a description on how it fulfills these requirements:

Economic:

- Stated previously, the E-IoT testbed for smart home technologies and power grid support will help save, firstly, consumers expenses on electricity. Having a program that can predict renewable energy production of users' devices and allowing them to turn on and off certain devices to help meet the threshold is extremely helpful. On the utility side, it allows for them to save on costs due to energy production and better stabilize the grid due to renewable energy production.

Environmental:

- The testbed is focused on the renewable side of energy generation. By having a tool like this, it will help with the implementation and innovation of IoT devices. Thus, better suiting and solving different issues that may arise. Also, it drives down the release of carbon emissions leading to global warming.

Sustainability:

- As stated in the environmental section, this is geared for renewables. The energy source for these systems are from renewable sources. Therefore, it'll help with the sustainability

Manufacturability:

- There will be little to no physical manufacturability for this technology. This testbed is a software that helps predict and optimize the use of renewables. IoT devices are readily available along with the RaspberryPi computers. The user interface will work on any smart device.

Ethical:

- Usage of this testbed is ethical. The users' part of a smart home site will be allowed to opt-in or opt-out. Meaning they can choose weather or not they want to participate in assisting grid needs. By doing this, it allows for the respect of the users privacy.

Health and Safety:

- A testbed like this helps with the stability of grid conditions due to the increased presence of renewable energy in grid conditions. Instability causes blackouts or surges that can be detrimental to homes and essential services. In some instances, a loss of electricity can be a matter of life or death. Having a system where it keeps grid conditions stable, it will minimize the risk of events like this from occurring.

Social:

- Global warming and pollution is the talk around town. People, along with businesses, are pushing to become more environmentally friendly. With this E-IoT testbed, it will help with the successful deployment of such renewables.

Political:

- This doesn't play too much into the political aspect of things. It can be said that, in politics, there is a clear division between the use of renewables and the use of fossil fuels. It will have a negative stigma around those who are against renewables, but have a very positive impact on those for them.

There is a lot to be learned from partaking in this project. As the world converts from using fossil fuels to renewable energy, our infrastructure needs to follow suit. The additional set of challenges in knowing how to work with renewables requires aid from technology, as has energy from fossil fuel. I have learned about the instability of having numerous amounts of renewables on a grid and a tool, like this testbed, that can help aid companies in stabilizing it. Without tools like this, we are unable to allow up and coming renewable energy technologies to successfully enter the market.

Change is scary because it comes with a lot of uncertainty. However, if we have the tools to mitigate this uncertainty, then the confidence of its effectiveness can soar.

References

[1] R. E. W. -, By, -, Renewable Energy World Renewable Energy World's content team members help deliver the most comprehensive news coverage of the renewable energy industries. Based in the U.S., Renewable Energy World, and Renewable Energy World's content team members help deliver the most comprehensive news coverage of the renewable energy industries. Based in the U.S., "New Sustainable Energy Factbook shows 2020 was 'blockbuster' year for renewables in America," *Renewable Energy World*, 18-Feb-2021. [Online]. Available: https://www.renewableenergyworld.com/storage/new-sustainable-energy-factbook-shows-2020-was-blockbuster-year-for-renewables-in-america/. [Accessed: 01-May-2021].

[2] "U.S. Renewable Energy Factsheet," *U.S. Renewable Energy Factsheet | Center for Sustainable Systems*, 2020. [Online]. Available: http://css.umich.edu/factsheets/us-renewable-energy-factsheet. [Accessed: 02-May-2021].

[3] M. Jaganmohan, "PV capacity installations: U.S. residential sector 2019," *Statista*, 27-Jan-2021. [Online]. Available: https://www.statista.com/statistics/185694/us-residential-annual-pv-installed-capacity-since-2005/. [Accessed: 02-May-2021].

[4] S. X. staff, "Fast mitigation of power grid instability risks," *Tech Xplore - Technology and Engineering news*, 21-Apr-2021. [Online]. Available: https://techxplore.com/news/2021-04-fast-mitigation-power-grid-instability.html. [Accessed: 03-May-2021].

[5] Douglas Ellman, Pratiksha Shukla, Yuanzhang Xiao, Magdy Iskander, and Kevin Davies, "Integrated Energy Monitoring and Control IoT System and Validation Results From Neural Network Control Demonstration," accepted by 2021 World AI IoT Congress (AIIoT).