Requirements Specification

4/10/2025

Version 1
Capstone - WillowWatt

Team Members:
Ayla Tudor
Elliott Kinsley
Luke Bowen

Sponsor - OP Ravi, Director of Energy Transformation, Willow Inc.
Mentor - Jeevana Swaroop Kalapala

Accepted as baseline requirements for the project:

For the client: _____ Date: _____

For the team: _____ Date: _____

Table of Contents:

1 Introduction:

Buildings are some of the largest consumers of energy in the world. In the United States alone, they consume approximately 40% of all energy and 74% of electricity produced annually [1]. A significant portion of that electricity is generated through the combustion of fossil fuels, meaning that inefficient building energy use directly contributes to greenhouse gas emissions and climate change. As global temperatures and energy costs rise, improving building energy usage has become a growing priority. Institutions, corporations, and universities are increasingly investing in smart, more efficient infrastructure to improve sustainability and energy costs. However, many buildings still rely on outdated systems, such as static schedules and manual adjustments, that are unable to optimize energy usage.

This project is sponsored by Willow, a company dedicated to smart building management and energy transformation. Willow uses a “digital twin” platform to monitor building systems in real time and collect data on lighting, HVAC, and other building assets. At Northern Arizona University (NAU), Willow is already actively monitoring energy usage across campus with the goal of improving energy efficiency. However, there is an opportunity to use AI to transform the raw data being collected into real-time forecasting and actionable optimization. To address this opportunity, our team is developing WillowWatt, an AI/ML-powered energy forecasting and optimization system that will integrate directly with Willow’s platform. Our goal is to help NAU reduce energy usage, lower emissions, and meet its carbon neutrality target by 2030. Our team is committed to achieving this goal without compromising building performance or comfort. By building a system that forecasts energy trends and provides optimization suggestions, we aim to help Willow enhance their platform and scale these insights to a broader set of users and applications. Our sponsor, OP Ravi, is the Director of Energy Transformation at Willow. With deep expertise in energy systems and sustainability initiatives, OP has been instrumental in guiding our team’s understanding of Willow’s platform and its goals. He has helped our team shape our approach by sharing insights into Willow’s operations and identifying opportunities.

The potential impact of our project is substantial. NAU currently spends around $15 million annually on energy at its Flagstaff Mountain Campus [2]. The university has set a goal to reduce energy and water usage by 20%, potentially saving NAU over $2.8 million per year with paybacks ranging from 2 to 9 years [3]. By supporting these efforts with energy forecasting and optimization tools, WillowWatt aims to turn data into actionable strategies that reduce emissions and cut costs. Our system not only strengthens Willow’s digital twin capabilities, but also supports broader sustainability goals. With the potential to scale beyond NAU, WillowWatt represents a meaningful step toward smarter, more sustainable buildings.

2 Problem Statement:

Willow's key business workflow revolves around improving building operations through real-time monitoring and data analysis using their digital twin platform and collecting energy data from building assets like HVAC and lighting. This data is visualized and made available to facilities staff or sustainability teams who review dashboards and metrics to identify inefficiencies and make improvements.

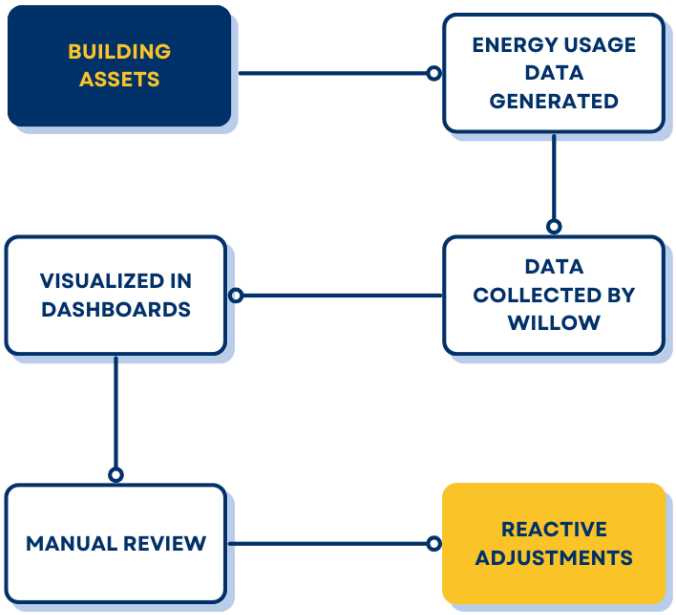


Figure 1.0: This diagram shows the current workflow for monitoring and managing building energy usage data.

While Willow's platform excels at collecting and displaying data, it currently lacks forecasting and optimization capabilities. Without these tools, decision-making is reactive and takes place after the energy inefficiencies have already happened. Building assets generate continuous energy usage data with Willow's sensors, but optimization efforts are dependent on human interpretation and limited. This workflow leads to several challenges:

- No predictive insight: There is no system in place to forecast future energy demand.
- Limited optimization: Decisions are limited to being made reactively after peak usage or inefficiencies have already occurred.
- Expertise required: Energy management decisions depend heavily on human analysis.
- Inefficient asset use: Without smart forecasting, asset scheduling does not adapt to changes in demand.
- Scalability constraints: The lack of automated forecasting and optimization makes it difficult to scale energy savings across multiple buildings.

3 Solution Vision:

Our team is proposing to build a basic machine learning model that will effectively predict a weekly forecast of energy consumption of a given building that is available on the Willow dashboard platform. This machine learning model will utilize Random Forest Regression using a well known ML tool SKlearn. Once our team has designed the model to forecast building energy consumption, we will optimize the energy system for that building by simulating reducing daily peak loads in our machine learning model. To come up with a solution for Willows problem of not knowing when peak loads will happen and how to easily inform people when they should consume less power:

- A system for acquiring data from Willow
- Splitting the data into testing and training subsections
- Train data using a machine learning algorithm (Random Forest Regression)
- Test the model continuously using training subsection
- Model packaging using ONNX
- Data visualization for the prediction model

Our team will implement a feature to pull historical data from Willows dashboard. This data will have two main columns that we will use, date and time information as well as megawatts per hour that the building is consuming. We will pull this information and use it in our python application by using the Pandas tool in the python library. Next we will use SKlearn train_test_split feature to divide our data into two subsections, which will feed our model with valuable information to train then test the model.

```
# Split data into training and testing sections
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)
```

Figure 1.1: This shows example code of how we will split our data into a training subsection and a testing subsection. X_train and Y_train will be the subsections that we pass into our training model.

Next we will want to begin training an optimized Random Forest Model by passing our training data into the model. Random Forest Regression is an ensemble machine learning method that combines multiple decision trees to make a final prediction. Typically these trees will be made up of random parameters and the trees are tested to see how accurate they are. This algorithm will then pick which tree has the most optimal forecasting outcome, the model will take an average of all individual tree predictions.

```
# Train optimized Random Forest model
model = RandomForestRegressor(n_estimators=50, max_depth=10, n_jobs=-1)
model.fit(X_train, y_train)
```

Figure 1.2: This shows how we will initialize our model with random forest regression by making 50 learning trees with the model, each tree having a maximum depth of 10. We then fit our model with both our hourly time series data and the megawatt per hour values.

Once we have our model, we can begin making predictions for the future energy consumption. Furthermore, we will be able to convert our model into an ONNX format, which will allow us to attach our model to the Willow dashboard and visualize our data.

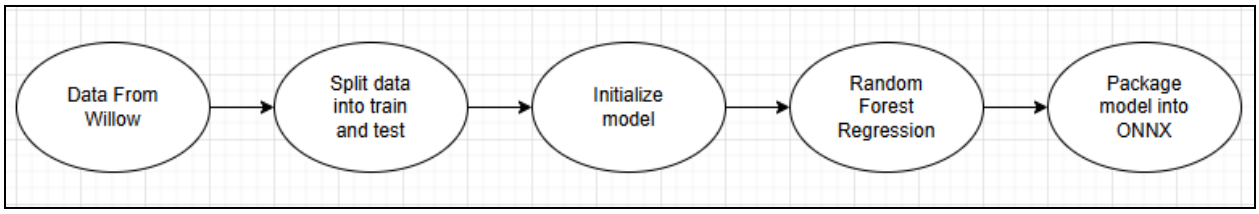


Figure 1.3: This diagram shows the basic flow in which our model will receive time series data, then train itself to forecast energy consumption to a given time interval.

Once we have our model working with the data from Willow, we will want to scale the model to only predict for the next week. Then we can begin finding the average peak load times for both the week and daily consumption averages. Now that our team has a working model to forecast NAU building energy consumption, we can now start researching average megawatt consumption values for specific equipment, such as lights, HVAC, etc. We will then reduce these loads in a simulated fashion, so that our client will have a better understanding of different ways we can optimize building consumption. Lastly, once we have an optimized forecasting machine learning model, our client has requested we have the option to feed a large language model our data and forecasting model to create a specialized client. We can then use this specialized LLM agent to be integrated into a basic user interface, potentially with the interface that is being constructed by a different computer science capstone group.

The purpose of this LLM is to have a way to communicate with the client and users of the willow platform on how we can reduce energy consumption and answer any questions our client may have about specifications of NAU's campus. Furthermore, this AI chatbot could be set up to do continuous reminders to people that use the platform if the energy consumption for their building becomes too high. If our team can find a way to segment which assets are taking the most power in that hour based on our forecasting models, we can notify people to turn off or reduce that individual asset. Doing so will improve our overall notification system substantially.

4 Project Requirements:

4.1 Functional Requirements:

To accurately forecast, display, and optimize the energy consumption of buildings across NAU's campus, the WillowWatt system must support a range of functionalities. In collaboration with our sponsor, OP Ravi, we've defined a list of the core features that we deemed necessary in order to achieve our vision and meet the needs of our client. Our system must be able to handle both real-time and historical data. This will make it possible to provide current updates and insights on energy usage. Our system must also be able to forecast future energy usage based off of the historic data passed through our model. It is crucial that our system is capable of visualizing the data of our energy usage forecasts and their insights/reports. The WillowWatt model must deploy and integrate seamlessly into Willow's existing platform, adhering to the user interaction and access and security features that Willow has established. Each of the following functional requirements contributes to the overall vision of our intelligent and actionable forecasting and optimization tool.

4.1.1 Functional Requirements - Historical Data

Our system will import historical building energy data (data/time, consumption in MWh) from Willow's existing database using a python library called pandas. This system must handle large datasets for the purpose of training our model, since forecasting models need a substantial amount of data for accurate forecasting. Our client says there is about 3 years of historical data that we will be able to pull from, which is on the shorter end of the required amount of data compared to the data we were using for our prototypes, but it is still enough to train the model. Furthermore, our historical data import functionality must validate the incoming data's quality to handle situations where we have invalid formatting, missing values, or duplicate data that will lead to imperfections in the training process. We should attempt to automate this process whenever our historic data gets refreshed to avoid conflicts with updating our model in the future. Moreover, our system should provide the functionality to create logs and reports of any data acquisition issues. We will create the ability to log missing data, overlapping data, or invalid formatting and write these messages into a log file.

4.1.2 Functional Requirements - Data Ingestion

Our system will also have the ability to connect to Willow's real-time data feed to update energy consumption metrics as they become available. We will periodically update our model based on the real-time data coming in, to keep up with any changes in the energy consumption behavior. This data should be updated in regular intervals such as every hour, to support fast responsiveness and reporting in our system. We will also need to add the functionality to store or buffer incoming data to allow our forecasting model to retrain or update the model when

necessary. We want a buffer to ensure that our model is correctly ingesting the data and not skipping over any potentially important data at a given time.

Similar to our historic data, we need our system to flag any anomalies (such as negative consumption or extreme spikes) and prevent our model from training with invalid data. This will create a safeguard to ensure our model is protected from bad data. The reason we might encounter bad data is from faulty hardware at a given building that would store the wrong information and send it to Willows database. Our system should be able to identify a rough direction where this invalid data is coming from and create a log of the incident that will notify our team. Our system should have permissible data ranges and values coming in, such as invalid date and time information, that will look for things such as mismatched data labels that are too far in the past or future or obviously corrupted timestamps. We will also want the ability to notify our team if there is a missing import or if Willows servers are down to prevent anything happening to our model pipeline.

4.1.3 Functional Requirements - Random Forest Regression Algorithm

Our system will use a Random Forest Regression algorithm to train on historical data and tune hyperparameters for optimal performance. This will provide the core function of energy usage forecasting by taking in historical data and outputting forecasted MWh at a given date-time interval. We will implement this algorithm in python using scikit-learn (SKLearn), which we chose based on our prototyping phase, in which we created multiple data forecasting models using several different algorithms and a data set we downloaded from the internet that contained date-time and MWh columns of data, which is similar to what we will be getting from the Willow database. We then compared these various models' forecasted data and found that using Random Forest Regression had fewer peak and off peak errors in comparison to algorithms like XGRegression using XGBoost. XGRegression is another popular forecasting algorithm that is known for working reliably on large datasets, however, Random Forest Regression appears to be the favorable algorithm for forecasting Willows data. Random Forest Regression works by creating a certain number of specialized decision trees (estimators), with each decision tree having a certain depth in which each branch has random weighted parameters, the model then takes the average tree and uses this tree to make accurate forecasted data. Random Forest Regression must allow setting forest parameters such as the number of estimators and the maximum depth of each tree.

4.1.4 Functional Requirements - Custom Forecasts

Once our model is trained, the system will generate short-term hourly predictions for up to one week into the future of building energy consumption. Doing so will provide actionable insights that we can use to optimize building energy systems to consume less energy. These predictions should be refreshed on a regular schedule or upon demand from authorized users, so that our model will stay up to date with energy consumption trends. Furthermore, we may possibly add the functionality to allow users to specify custom intervals (e.g. 2 weeks) for our

model to forecast, so that they might get more insight to overall energy consumption that they might not have gotten from just one week. We will also want to create the functionality to handle when the model has insufficient data to update the model (such as when a new building is added), then our model will fallback on previous data points to maintain our models operations.

4.1.5 Functional Requirements - Optimization - Peak Load Identification

The system should analyse forecast data to identify expected peak load periods (hourly or daily), this will address the need to reduce or mitigate high consumption rates. This function will need to provide a summary of peak load periods at least for the next week. Furthermore, this functionality should automatically highlight abnormal spikes beyond the given bounds. This reporting feature should contain a graph that points out peak load estimates based on the forecasted data. Our team will then need to calculate a rough estimate of each asset's power consumption using certain energy consumption estimators that we find. We will also want to determine a criteria for determining peak loads. The system should define a pearl load threshold based on historical data averages, standard deviation, or maybe user defined limits. The system should also provide not only a numeric history summary, but highlight these time intervals with an interactive time series chart. We will need to differentiate pearl loads from normal loads with clear markers and boundaries in the code. Our table should have the ability to support hover over properties that will display additional information about the data. Having all of this accessible data will hopefully give the user actionable insights into how they can reduce energy consumption.

4.1.6 Functional Requirements - Optimization - Load reduction simulation

Next, once our system has identified the expected peak load periods, it will simulate strategies to reduce energy usage during peak times (e.g. turning off assets or reducing HCAV loads). This will help us test hypothetical scenarios that will inform us what loads we can reduce to a certain limit. We also have the option to create an optimization model that uses load reductions and reduction limits to calculate the most we can reduce a certain asset load at a given time and still have the required functionality of this asset. Doing so would help our team calculate the maximum energy reduction a certain asset can have at a given time, but still have the asset provide the service that it needs to function properly. Furthermore, we can also have the functionality to have a user input of percentage reduction of certain assets and have our system calculate the overall energy reduction impact and savings that it would have on the building, along with the updated peak estimates. An input reduction system will help both the users and our team understand the overall impact that certain assets have on a building, which we can use to further strategies to reduce energy consumption in a given building. Our team has spoken with the office of energy sustainability, they gave our team insight that the Health and Learning Center (HLC) has the most available data when compared to other buildings. Our team will get as much information on this building's energy consumption as possible to begin preparing our load reduction simulation plan.

4.1.7 Functional Requirements - Optimization - Automated Recommendations

Once we have a functioning optimization recommendation system, whether it be through our own platform or integrated into the other willow platform capstone project, our system will provide the functionality to automatically send regularly scheduled updates and recommendations to our users on how they can take steps on reducing their energy consumption. This feature should also notify users if there are peak loads detected, especially if there is an irregular peak load that users should be aware of. This system will also be of great use to facility managers and Willow staff, since they will also be notified of actionable insights on how to reduce power consumption in a given building. This system should have a ranking system based on their potential impact of the system (highest to lower kWh savings). The system would also provide a rationale of confidence intervals for each recommendation, if possible. This would give users more insight to what is pulling the most power and the reasoning behind it.

4.1.8 Functional Requirements - Data Visualization - Interactive Graphs

Our system should produce easily interpreted graphs (time-series, graphs, bar charts, etc) showing both historical and forecasted data energy consumption. This will provide users with accessible reporting and insights. In the first few phases, we will be using matplotlib to display our historical and forecasted energy data. This will help our team understand the historical and forecasted energy data without needing to regularly pull real time data. However, the interactive graphs will eventually need to be embeddable into willows existing web platform or ve accessible via a standard dashboard. A third option our team has is integrating our visualization data into the platform that the other capstone group is making. Once we begin integrating our data visualizations into a platform, we will want to be able to update our real time data and model in incremented intervals. To be able to deliver a truly interactive experience for the user, these graphs must do more than just passively display a line. They should allow users to zoom, pan and scrub through any time window and reveal more data information about that given point.

4.1.9 Functional Requirements - Data Visualization - Peak Load

The dashboard that our team will be using will have overlay identified peak load periods and recommended optimizations directly on the energy consumption graphs. This will enhance the users understanding of the forecasted data and potential solutions. The graphs should use distinct markers or highlights for the peak times, so users can immediately know when the most demanding times are. We should also include an easy to understand section on forecasted load shifts and how they compare to the regular graph. Having this comparison would promote a cause and effect mental model between the regular data and how the optimization path will affect the power system. Our Peak load graphs should have the ability to save a snapshot of the current graph as a pdf. This will give users the ability to download and use our models for further use. Further into the visualization portion, our team would like to make the functionality to compare

multiple buildings of similar sizes. The ability to compare one building to another will give users insights into how much energy one building uses compared to another.

4.1.10 Functional Requirements - Model Packaging Using ONNX

A request we had from our client is that our forecasting models will be packaged into an ONNX format for seamless integration with Willows platform. ONNX (Open Neural Network Exchange) is an open-source intermediate packaging framework and tooling that lets you move a trained machine learning model from one framework to another without rewriting it. It works by using a computational graph format, a well defined operator set, as well as model and tensor metadata. This will provide our project with the ability to deploy our models into Willows dashboards. We have investigated the accuracy of the packaged model and the pre-packaged model by comparing a graph made from the prepackaged model and the packaged model, and we found no discrepancies between both the graphs. This shows that ONNX is a dependable packaging system for our project.

Our client has shown us that it is a simple process for uploading our model into the Willow dashboard, so once our team has successfully created the forecasting model, we won't have an issue implementing our model into Willow's dashboard. Once we have a functioning system for uploading our models into Willow, our team will create instructions for how Willow's platform interacts with the load model, which gives our client a better understanding of how our system interacts with theirs. Willow's platform already has a drag‑and‑drop "Add Model" panel. We can make the model upload even easier, we'll ship a small script that exports the latest model, tags it with a version number and the git commit hash, and pushes it to the upload folder automatically having no manual steps, no chance of mixing versions. Each upload will get a simple health check: the dashboard runs a quick test forecast and makes sure the numbers stay within 0.1 % of yesterday's baseline; if something is off, the model is rolled back automatically.

4.1.11 Functional Requirements - Security - Secure Login & Roles

To protect both the users and the clients data, our system will require secure login to access forecasts, data visualizations and optimization functions. Willow's data is private information, so they need to have protection for their sensitive building data and ensure only authorized personnel can access or make changes to the system. Multi-factor authentication is preferred in our login process to ensure extra security, however this may change based on the level of security our client wants in this phase. We will also use a guest feature that our users can use to have access to the system under restricted parameters. The system should go further than simple credential checks, the platform should adopt a role based access control model with predefined roles such as viewer or administrator. This will only be necessary if we don't only use the Willow platform to display our system information. This will allow our system to let the right people in, but keep the wrong people out who shouldn't have access to private information.

4.1.12 Functional Requirements - Model Performance Monitoring

Occasionally, machine-learning models regrade quietly as building-usage patterns, weather, or sensor calibrations change. Therefore, it is important for our team to implement a model performance monitoring feature for our system, to analyse the model and detect any data drift that may occur in our system. Our team will design an automatic model performance report generator that will tell us if our model is degrading in an irregular fashion. We will do this by occasionally taking a screenshot of our current model and comparing it to the rate at which our previous models have degraded. This feature will need to store the previous rate of degradation of previous models that our system can use to compare to our current model. However, since the system won't degrade very fast, we can set our model performance modeling to capture and compare the models accuracy over a long period of time (e.g. 2-3 years). Having this model performance modeling will help both our team and client understand how accurate the current mode is and if we need to make adjustments to our forecasting models.

4.2 Non-Functional Requirements

In addition to all of the core features that our system needs to deliver, there are several non-functional requirements that will help ensure that WillowWatt performs reliably and securely. These requirements define how well our system achieves the functionalities already discussed. Since our tool will be used by both Willow staff and NAU faculty, it’s critical that WillowWatt is fast, accurate, easy to use, and scalable. We’ve worked with our client to outline the key performance goals and system qualities that are needed to support the desired user experience.

4.2.1 Non-Functional Requirements - Performance

Our system should have performance regulation guidelines and parameters based on the following parameters. Our systems forecasting API shall return the next week prediction in under 5 seconds for a single building. The web dashboard in Willow shall render the main graph within under 2 seconds on a 25Mps connection. A full model retrain (every 12 months) shall finish under 30 minutes to provide fast model production.

4.2.2 Non-Functional Requirements - Accuracy

Our system should have quantifiable efficiency in both forecast accuracy and data integrity. We should have end to end checkpoints to ensure that there is no data obstruction between both ends of our pipeline. We will want to regularly test the data coming in for any glitches or bugs in our data acquisition. Furthermore, our system should check for validation rules like negative kWh, impossible timestamps, and incorrect time intervals in data coming through. Lastly, our system should ensure that the data it is reporting, along with our optimization approaches, are tested for accuracy to ensure there are no discrepancies.

4.2.3 Non-Functional Requirements - Usability

The users on the platform should not have any issues locating certain points of information or functionality. Our team should focus on making sure that all data is accessible at a first glance, and limit the amount of cognitive load our user has when accessing our information. We can achieve this by designing well structured interactive graphs and charts. Our Main goal for this project is to inform our users on what they should be doing to reduce power consumptions in buildings, they will be unable to do so if they have any confusion understanding the data that we are presenting them.

4.2.5 Non-Functional Requirements - Reliability

Building on the idea that our users should expect accurate information that they can easily understand and use, our users should also be able to rely on the data that is being presented to them. That is why our team will focus on early testing and functionality to make sure that everything in our system has a low or no likelihood of failing. We can do this by doing functionality testing using automated testing tools, to check every possible scenario to ensure nothing is faulty, especially in our data.

4.2.6 Non-Functional Requirements - Compatibility

Our models that our team designs should ensure that there is cross platform compatibility with both Willows platform, and whatever other platform our team decides to use. That is why we should make modular code that doesn't have too many dependencies for both software and hardware related requirements. Our system should also be compatible to use with handheld devices such as smartphones, but this is not guaranteed.

4.2.7 Non-Functional Requirements - Security

Adding onto the security aspects in our functional requirements, our team is aiming to go a step further than basic security steps. We want to implement a system that cannot be breached or have our sensitive data stolen. We will need to keep in mind various forms of cyber attacks when implementing our credentials system.We should have the ability to have incident responses and monitoring so our team knows if there has been any attacks.

4.2.8 Non-Functional Requirements - Maintainability

Our system should have a code base and ML pipeline that is easy to understand and troubleshoot. This will minimize time-to-fix bugs and time-to-show new features. Having a maintainable code base will also promote scalability in our project in case our system gets bigger. Our coding system will be modular and our team will focus on specific functionality in our coding to make sure that there is no overuse or god functions present in our code. Ou rteam will want to focus on continuous integration in our system, keeping up with important information and ways we can optimize.

4.3 Environmental Requirements

In addition to all the functional and nonfunctional requirements our project has, it also has multiple environmental requirements. These are a special type of nonfunctional requirement that are out of our control, rather they are imposed on the project from an outside source, such as directly from our client or the need to integrate with existing software.

4.3.1 Environmental Requirements - Platform Dependency

Our system will need to operate within Willow's existing platform in the cloud. Willow has a streamlined integration system in place within their application for AI models to be ported into, but this will also require that our model complies with the existing software environment. One requirement that comes with this from our client is using the ONNX package within our software. We have been informed that this will ease the integration and ensure that our model can successfully run within their platform.

4.3.2 Environmental Requirements - Test Environment

An additional requirement of our system is having a physical testing environment to run our optimization adjustments on. Unfortunately for this project, this might not be completely feasible due to limited access to buildings and their assets across campus. One example of this is if we were planning to adjust HVAC schedules in an attempt to reduce energy consumption, this would likely require approval from NAU administration, which might not always be available. The way to combat this is done by simulating the testing environment instead, and running tests on the emulated building or asset. Our system will need to have the environment to run tests on in order to provide an accurate assessment of the energy usage reduction based on our models insights.

4.3.3 Environmental Requirements - Legal Restrictions

The final environmental requirement that our project faces is the legal liability of gaining access to the Willow platform, and the need for NDAs. NAU faculty has been working in collaboration with Willow to have any necessary documentation signed and agreed upon to ensure we are allowed access to the Willow platform, as well as all of their data on NAU. This is completely out of our hands, as it is being handled directly by Willow and the faculty here at NAU, but it is a necessity to get finished as without access to the Willow platform and their data, we will be unable to successfully complete our project.

5 Potential Risks:

5.1 Potential Risks - Overview

There are multiple potential risks involved with this project, but thankfully most risks can be mitigated by proper planning and project management by our team. These risks could pose some negative impacts to both the company Willow as well as any users of their platform if our model is fully deployed to their application at the end of the site, making these risks critically demand our attention throughout development. Most of the main risks our project will face stem from misleading or faulty data, either being passed through our model due to the data not being properly cleaned or anomalies not being properly flagged.

5.2 Potential Risks - Faulty Chat Bot

The first main risk we are going to address is a faulty chat bot giving the wrong information to users. This risk is likely a development error, and not the fault of any data or anything on Willow's side. This could look like the chat bot simply providing incorrect information when providing the user with insights, being as simple as mixing up datasets and providing the user with insights on a different building or asset, or it could be something more intricate such as providing insights that are slightly off based on the forecast. This risk is not as dire as some other potential risks, because although this would pose a grievance to the user and Willow, the chat bot is a supplementary element to this project, as the main components are the forecasting and optimization models. Ideally, with those elements working, this risk would just pose as an annoyance forcing the user to dive deeper into the forecasting results to determine their own insights, but not a total catastrophe to the overall project. In other words, the project as a whole would not be derailed by this issue.

5.3 Potential Risks - Faulty Simulations

One potential avenue we will have to take for this project is optimizing a simulated environment if we cannot gain access to making modifications on buildings or specific assets around campus in an attempt to optimize energy usage based on our insights from our model. To accomplish this we would just emulate the building/asset we want to run a test on, and use our ML algorithm to predict how much energy usage could be reduced by performing certain actions, or lowering usage of the building/asset at certain times. The risk that comes with this however, is running faulty simulations, or getting faulty data out of these simulated environments. This is another error that would likely only be caused due to a bug in the code missed during development. This risk carries slightly more urgency than the previous risk however, as this directly affects one of the larger proponents of this project - optimization. If we are unable to gain administrative approval to make real-time adjustments to buildings or assets and have to proceed with these simulations, running a simulated adjustment to predict what the hypothetical

energy usage reduction would be is the main way our project would be attempting to optimize energy usage across campus. This means that if our simulations were to run faulty, then a large part of our project would be underperforming. This poses more of a risk to Willow than it does to any Willow user, as most users of Willow's platform would be using our model for the forecasts and insights, as opposed to the potential optimization. For Willow however, an inaccurate measurement for optimization would require them to do more work in the future for figuring out what real-time adjustments can be made to lower energy usage on campus.

5.4 Potential Risks - Project Adaptability

A final potential risk that every project must consider in the planning phase is what will happen if a better product comes along in the future and threatens to replace your product? For our project specifically, I think this is less of a critical risk, but still one worth discussing. As development and research into AI and ML becomes more advanced, it is possible that our project could be phased out and replaced by something more capable in the future. However, this does not diminish the impact of a successful completion of our project. Completing this project successfully and supplying Willow with the tools they are desiring could simply be a stepping stone going forward with more advanced AI and ML models being developed. This means that this could potentially be a positive risk for Willow, as down the line this project could end up saving them time and money, should they decide to build off of it and integrate more advanced AI and ML tools into their platform.

6 Project Plan:

6.1 Project Plan - Overview

Our project execution plan is organized into three main development phases: Energy Forecasting, Energy Optimization, and Energy Management. These phases reflect the core functionality we've defined in collaboration with our client and help us structure our project progress. We expect the first phase to move much quicker than the others, with the second and third phases taking up the majority of our development time. Each phase contains specific milestones that correspond with the functional requirements of our project and we've outlined a timeline for when each will be completed. Phase 1 represents our Minimum Viable Product (MVP) which will be a working forecasting model that integrates with Willow's platform and provides actionable short-term energy predictions. This will lay the foundation for the optimization and management tools we'll develop in Phases 2 and 3. While we've identified the key milestones and timeframes for each phase, our team recognizes that these are subject to change as the project evolves. As we receive data, feedback, and integration support, we may shift milestone timelines, add new deliverables, or revise our priorities to better meet the needs of our client and the project as a whole.

6.2 Project Plan - Phase 1: Energy Forecasting

The first phase of our project will be completing the main forecasting algorithm for our model. Based on our prototyping experience, we expect this phase to move quickly. Once completed, it will serve as our MVP. During this phase, we'll train the model and begin generating predictions for up to one week in advance. We'll also begin testing integration with the Willow platform to ensure that the model is compatible and can be visualized properly.

Phase 1 Milestones:

- M1: Finalize historical data import and validation.
- M2: Complete Random Forest Regression Model and produce weekly forecasts.
- M3: Package the model using ONNX and run deployment tests on the Willow platform.

6.3 Project Plan - Phase 2: Energy Optimization

The second phase of our project will be completing our optimization algorithms. This will likely make up the bulk of our development phase. A successful completion of this phase will allow us to accurately identify the most efficient ways to consume energy on campus and reduce costs. The goal of this phase is to identify opportunities to reduce energy consumption based on the forecasted data and simulate strategies that could help smooth out peak load periods. We'll use estimated energy usage for different building assets to simulate energy reduction and calculate potential savings. This phase will also include building a system to

generate automatic optimization recommendations and visualizing them alongside the forecasted energy usage.

Phase 2 Milestones:

- M4: Analyze forecasted data to identify peak loads.
- M5: Simulate load reduction strategies using estimated asset-level data.
- M6: Generate automated optimization recommendations.
- M7: Visualize forecast and optimization data with interactive dashboards.

6.3 Project Plan - Phase 3: Energy Management

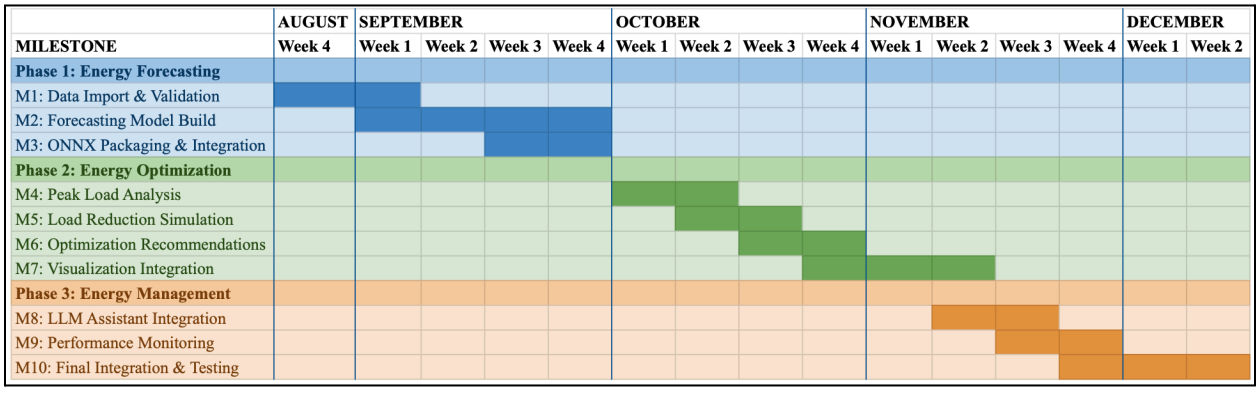
The third and final phase of our project will be integrating our own local AI Large Language Model (LLM) to provide additional insights on our forecasts and optimizations. The main goal of this local AI agent is to provide a friendly and intuitive UI that can provide a user with simplified insights and information based on the data and assumptions that our complete model makes. A successful completion of this phase will result in an interactive interface powered by a local LLM that can explain forecasts, offer optimization suggestions, and answer questions about energy usage patterns in a natural and accessible way.

Phase 3 Milestones:

- M8: Build and connect LLM assistant for interacting with forecast/optimization data.
- M9: Develop model performance monitoring and fallback logic.
- M10: Final platform integration and user testing within the Willow system.

6.5 Project Plan - Gantt Chart

The following chart provides a visual breakdown of our timeline organized by phase and milestone. A full-resolution version of this Gantt chart is included in the appendices (Section 8) and available as a Google Sheet for easier viewing:



7 Conclusion:

Reducing energy consumption in large buildings is one of the most impactful ways we can cut emissions, lower operating costs, and work toward long-term sustainability. With buildings responsible for such a large share of electricity use and emissions, there's a huge opportunity to make a difference, especially at institutions like NAU that are already investing in climate action. Our project, WillowWatt, aims to support that mission by giving stakeholders the tools they need to move from reactive decision making to proactive and data-driven energy management.

Currently, Willow's digital twin platform is great at collecting and displaying energy data, but it doesn't yet provide the kind of forecasting and optimization features that would make the system truly proactive. WillowWatt will be filling that gap by introducing an AI/ML-powered forecasting model to predict short-term energy usage. Once we have those predictions, we'll use them to identify peak load periods and simulate load reduction strategies that could lower consumption during the most energy-intensive times of day. In addition to forecasting, our team is also focused on accessibility and actionability. The insights from our system will be displayed through interactive visualizations and we plan to include a local large language model assistant that can help explain energy trends, suggest optimizations, and even send alerts when energy consumption spikes. These features are designed to make it easier for both technical users and staff to engage with the data and act on it.

This document outlines every major component of our project. We've defined the problem and why it matters, described the system we're building, and detailed the functional, non-functional, and environmental requirements we need to meet. We also included our three-phase development plan, a Gantt chart with clear project milestones, and a look at the potential risks we're preparing for. Together, these sections form the foundation of our plan to deliver a working product by the end of Fall 2025 semester. We're proud of the progress we've made so far and feel confident in the direction we're headed. Our Minimum Viable Product is already underway and we've built a strong structure for how the remaining features will come together.

This document represents the first version of our requirements specification and will continue to evolve as our project progresses. As we receive feedback, refine our approach, and gain more access to data and tools, we'll update this document to reflect the most accurate and up-to-date understanding of the system. We're continuing to work closely with Willow and NAU to make sure our project aligns with real needs and provides real value. By the end of this project, our goal is to deliver a tool that not only works, but also helps Willow enhance their platform capabilities and supports NAU in reaching its energy and sustainability goals in a meaningful way.

8 Glossaries and Appendices:

Glossary:

ONNX: Open Neural Network Exchange, a format used to export machine learning models for platform interoperability.

Random Forest Regression: An ensemble ML algorithm that uses multiple decision trees to make more accurate predictions.

MVP (Minimum Viable Product): A version of the system that includes just enough core functionality to be usable and deliver value.

LLM (Large Language Model): A type of AI that can generate text, summarize data, and assist with user communication.

Peak Load: The highest energy usage period in a given timeframe, often used to identify when to reduce consumption.

Load Shifting: The process of moving energy use to off-peak hours to reduce strain and cost.

Digital Twin: A virtual representation of a physical building and its systems, updated in real time with data from sensors and control systems.

Energy Load: The amount of electricity a building or system consumes at a specific point in time.

Appendices:

Source Links:

1. https://sftool.gov/learn/about/44/building-energy#:~:text=Building%20operations%20consume%20approximately%2040,greenhouse%20gas%20(GHG)%20emissions
2. https://in.nau.edu/green-nau/climate-action-plan/maximize-efficiency/
3. https://in.nau.edu/green-nau/nau-climate-action-plan/

Appendix A: Full-Size Gantt Chart
View the full Google Sheets version here:
https://docs.google.com/spreadsheets/d/1QoPWFDjO3MuBKFvb7_78egnNKXt68Kh-jb3K7a9vS9A/edit?usp=sharing