Software Design Document

Version 2.1
9/26/2025

Capstone - WillowWatt

Team Members:
Ayla Tudor
Elliott Kinsley
Luke Bowen

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

Table of Contents

1 Introduction:

Buildings are surprisingly one of the largest consumers of energy worldwide. Buildings in the United States make up almost 40% of the total energy consumption within the country annually, while NAU alone spends over $15 million per year on energy to power its campus. A large portion of these costs is coming from the combustion of fossil fuels, which means that inefficient use and management of building energy usage is directly impacting the ongoing climate crisis and growing greenhouse gas emissions. Fortunately, many corporations around the world are turning to smarter energy systems to combat this, but many buildings around the world, but especially here at NAU rely on archaic systems to manage their energy output, such as static schedules or manual adjustments, and completely lack any forecasting or optimization capabilities.

This project is sponsored by Willow, a company committed to improving the management of buildings and energy transformation. Willow uses their own digital twin platform to monitor building systems in real-time through energy meters, collecting data on various energy assets (HVAC, lighting, etc..). Here at NAU, Willow is already actively tracking energy usage across campus in an attempt to improve energy efficiency. However, there is a clear opportunity within the Willow platform to integrate a smart AI model to transform the real-time raw data energy into real-time forecasting and optimization insights. To take action on this opportunity, our capstone team is launching WillowWatt, an AI/ML energy forecasting and optimization system that will seamlessly integrate within the already existing Willow platform. Through the use of converting telemetry into real-time forecasts and actionable setpoint recommendations, we hope to help NAU reduce overall energy usage, lower total emissions, as well as meet its carbon neutrality goals by 2030. As a team, we have committed to this goal without having to compromise building comfort or performance. Through the use of our finished product, we aim to help Willow further enhance their platform, and scale these insights to a broader audience. Our individual sponsor within Willow, OP Ravi, is the Director of Energy Transformation at Willow. With a deep expertise in energy systems as well as various sustainability initiatives, OP has been a crucial part in aiding our team's understanding of Willow's platform.

Some of our key user-level requirements include the following; users viewing hourly/daily forecasts per building/meter; receiving optimization suggestions; integrating with Willow's API for data I/O. With these key requirements in mind, it is clear that the potential impact of successful completion of our project would provide a substantial impact here on campus. With NAU spending over $15 million annually on energy, we could aid in NAU's goal to reduce energy and water consumption by 20%, potentially saving NAU over $2.8 million per year. Our system will not only strengthen Willow's current digital twin platform, but adds potential to scale beyond NAU, taking a step towards smarter, more sustainable buildings.

2 Implementation Overview:

To reiterate the purpose of our system, WillowWatt will be expanding on an already capable energy management system in Willow's digital twin platform, to integrate an AI/ML model that will accurately forecast energy usage across buildings at NAU, as well as optimization insights by simulating reducing daily peak loads. Our main tool we will be using to complete this is a well-known machine learning tool called Sci-Kit Learn, and we will be using a method known as Random Forest Regression. Packaging our model will be relatively straightforward, as Willow's platform already has porting capabilities for AI models using the ONNX package.

2.1 Energy Data Access:

One of the key factors in how successful our project will be is our access to building energy data. Willow's platform has some data within their digital twin platform, but accessing historical data from NAU's energy meters across campus would provide us with much more in depth data, in turn leading to more rigorous training of our ML model, which ideally leads to more accurate predictions as well as more valuable insights. We are currently working with both OP at Willow as well as the energy team here at NAU to get as much data as possible from both ends, in hopes that we will not run into any shortages of data for our model.

2.2 Visualization:

In terms of visualization and displaying our results, it should be relatively easy to communicate all meaningful forecasts and insights within the Willow platform. Because Willow's digital twin platform already has a professional dashboard and display with many customization options, our model will be able to overlay on their existing graphs. Partly due to the ease of porting with the ONNX model packaging, our model will smoothly integrate with the existing displays in the digital twin platform.

2.3 Time Data Intervals:

Some known limitations and assumptions that we've made are also playing a pretty crucial role in our plan. One of the biggest challenges is going to be the time intervals of energy usage data that we can get. We have received campus-wide energy data from NAU spanning back five years, where we can set the time intervals at ranges from 15 minutes all the way up to quarterly/yearly. For our model to accurately predict real-time forecasts within a 24-48 hour timeframe ahead down to every 15-30 minutes, it is important that we have access to years worth of data down to the 15 minute time interval increments, to ensure accuracy of our predictions. As of right now we are hopeful that we will get the more specific data as we have been told from multiple sources that this data is available, so we are progressing with this in mind.

All three of these factors play crucial roles in the implementation of our project, and all work together cohesively. Each one of these factors flows smoothly into the next one; we first

need to gain access to the Willow platform (which we have done) as well as collect as much energy data as possible. Once we have done this and fed the data into our model, we will need to visualize what our model is predicting so provide the information in a user-friendly way. Then on top of all of this, we know that the intervals of the data we collect will impact the performance of our model, so we are prepared to build out the model accordingly.

3 Architectural Overview:

3.1 Introduction

Our system will have two different sources of data pulling, this includes the live data feed we get from the Willow Platform, and the historical data that we have received from NAU's Energy Team. We plan on using libraries such as pandas, to create a dataframe of the date/time and a data frame for the energy added up for the entirety of North Campus. This will be used primarily from the live data that we are getting from the Willow platform. More on the live data from the Willow Platform, we have no way to access the API from the Willow platform.

3.2 Initial Problems

Our team is currently trying to solve this issue by using some data scraping techniques, however, if we are unable to get a reliable solution to this, we may have to require the user to manually save new data and train the short range model. More on the data collection process, our team is still working on the best way to pull from the historical data CSV that we have received, nonetheless, once we have a certain window and rate of data defined, will will then split our system into three primary sections: Long_Range, Short_Range, Gather.

3.3 Long Range Protocol

The long range protocol is supposed to handle multi parameter, subtle changes throughout the year in temperature, energy usage, and any other effects that might have impact on energy usage of a given building, or all of north campus (we will just focus on north campus at first). We will plan on using a bare bones deep learning algorithm to try to pick up on any long term trends that our data makes present. The purpose of this model is to look at the bigger picture when it comes to overall energy consumption, which is something that Random Forest Regression is not very good at. The input of this model can be various things, but our client is mostly interested in energy forecasting by using multi atmospheric parameters. While the long range design specifications are not fully designed, our team does anticipate using this feature since we have so much data we can use. This long range detection system aims to combat seasonal patterns and other things that can cause drastic short term changes that our short term protocol might not be able to pick up on.

3.4 Short Range Protocol

The next protocol will be the Short_Range, which will be responsible for predicting the next three days of energy consumption for all of the buildings summed up for NAU. Starting at the first billing period, our model will forecast our incremental 15 minute interval data points within willow (This may look as far back as the entire month), then it will analyze this prediction to find the highest peak within this interval. Furthermore, our system will continuously track the peak load that has been reported for the billing period, if our projection peak load is higher than the current highest peak, then this protocol will report back to our gathering stage that there is a projected peak at a given time with a specific number of Watts that are estimated to be lost. This is the model that we plan on using Random Forest Regression for, since this particular machine learning model is really good at short term predictions that our long range prediction will not be able to pick up. This is really important for Willow and Nau, as it lets people know that they are wasting energy and money, before it ever happens. Our team predicts that we will have some trouble with the accuracy of this model at first, and that we will have to do a little more than the split test train method to be able to get highly accurate predictions that we can give to our gathering model.

3.5 Gather Protocol

The last protocol that puts all of this together is the Gather protocol. This will be responsible for receiving the inputs of our short and long term forecasting models and spitting out actionable insights on what is happening currently with the building's energy consumption. The input and output for this is still unknown, since we have to look further into what parameters we can pass into this for it to gather relevant information, while leaving out any unnecessary data that might do more damage than good for the models. The gather protocol and the Long_Range protocol are both new additions to the system, which we still need to test whether or not we can have multi model communication within the willow platform. The short term protocol will work within the implementation of the Willow platform, since we know this works and we have a clear understanding of the short term goals, this will be the priority for our MVP of the product.

3. 6 Mid Level Overview - First Design

Lastly, we will give an overall mid level overview of this system design and what it aims to do. First and foremost, we will design a short term prediction model using random forest regression on the multiple years of five minute data that we have procured. Next we will visualize and clean the data as necessary, we will want to really analyze certain spikes to see if they are artificial or real natural numbers. Once we have nicely cleaned data, and we have used the pandas library to pull the information over, we will split this data into two sections down the middle, which becomes the test/train split that we can use for verification. Next we will feed the training portion of the data into the Random Forest Regression model (Prebuilt by SKlearn for simplicity).

3.7 Diagram of Project Architecture:

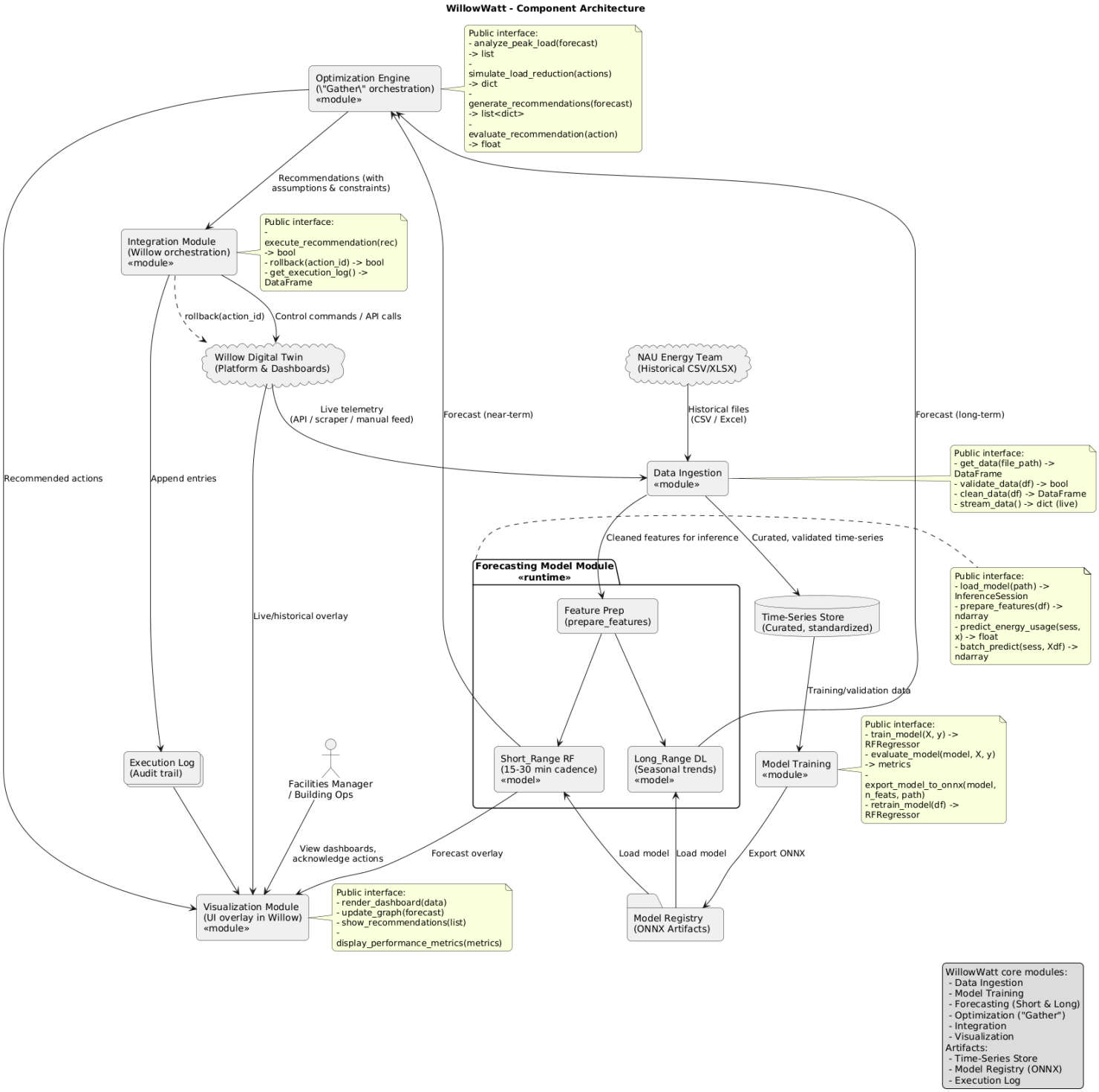

Figure 3.7.1: This diagram illustrates the high-level system architecture of WillowWatt, showing how data flows from ingestion through model training, forecasting, optimization, and visualization.

4 Module and Interface Descriptions:

4.1 Data Ingestion Module:

The data ingestion module is responsible for gathering all the data to feed into our forecasting and optimization system. This will be handled by collecting historical and live data from energy meters, and by pulling data provided by Willow and the NAU energy team. Another crucial component of the data ingestion module is responsibly cleaning the data before passing it to our model. This will include validating datapoints, aligning timestamps, and handling missing and/or outlier data. This ensures that our model is running off of correct data, and cuts out a major potential avenue for incorrect forecasts or optimization insights. A final important piece of the data ingestion model is to standardize schemas, and store curated, queryable time-series data with lineage. In the architecture, this module is responsible for feeding the forecasting model with reliable, normalized data.

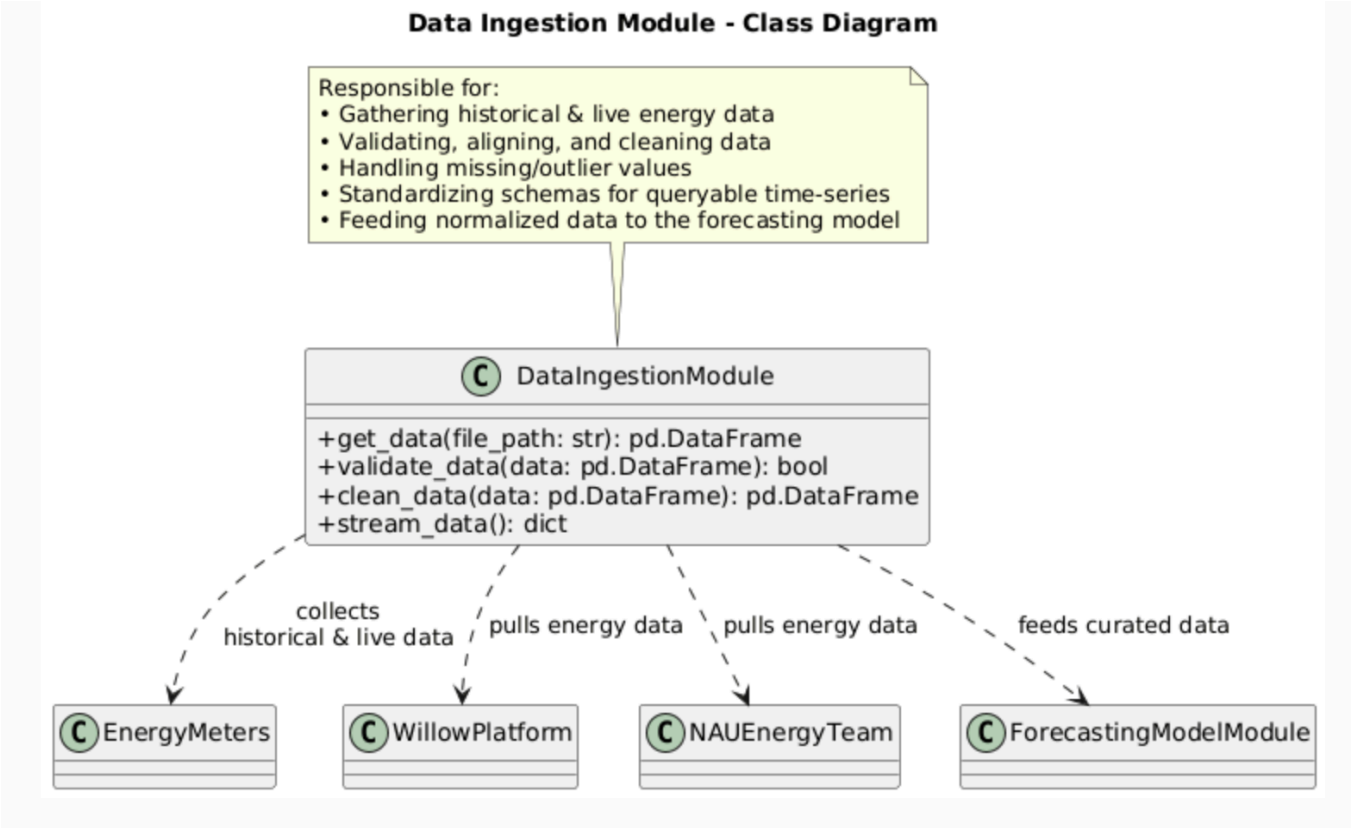

Figure 4.1.1: This diagram outlines the Data Ingestion Module, which collects, validates, and standardizes building energy data from Willow's digital twin platform and other sources.

The public interface for the data ingestion module lists the functions used to load, validate, clean, and stream energy data. These services make sure the rest of the system always has accurate and well-formatted data to work with:

- **get_data(file_path: str) -> pd.DataFrame**
  Loads raw building energy data from the specified file path and returns it as a DataFrame.
- **validate_data(data: pd.DataFrame) -> bool**
  Checks for missing values, duplicates, and invalid data points. Returns True if data passes validation, otherwise raises an error.
- **clean_data(data: pd.DataFrame) -> pd.DataFrame**
  Cleans and formats the dataset (timestamps, units, and column names) and returns a standardized DataFrame ready for feature engineering.
- **stream_data() -> dict**
  Retrieves live building sensor data in a structured dictionary format for real-time processing.

4.2 Model Training Module:

The model training module is responsible for building and maintaining the machine learning models that power WillowWatt. This module ingests historical building energy data, engineers predictive features, and trains our Random Forest Regression algorithm. It also supports retraining on updated datasets to ensure that the system remains accurate. Once a model has been trained and validated, this module exports it into ONNX format so that it can be deployed by the forecasting model module for real-time predictions. In the overall project architecture, this module ensures that the forecasting engine is always backed by a reliable, up-to-date model.

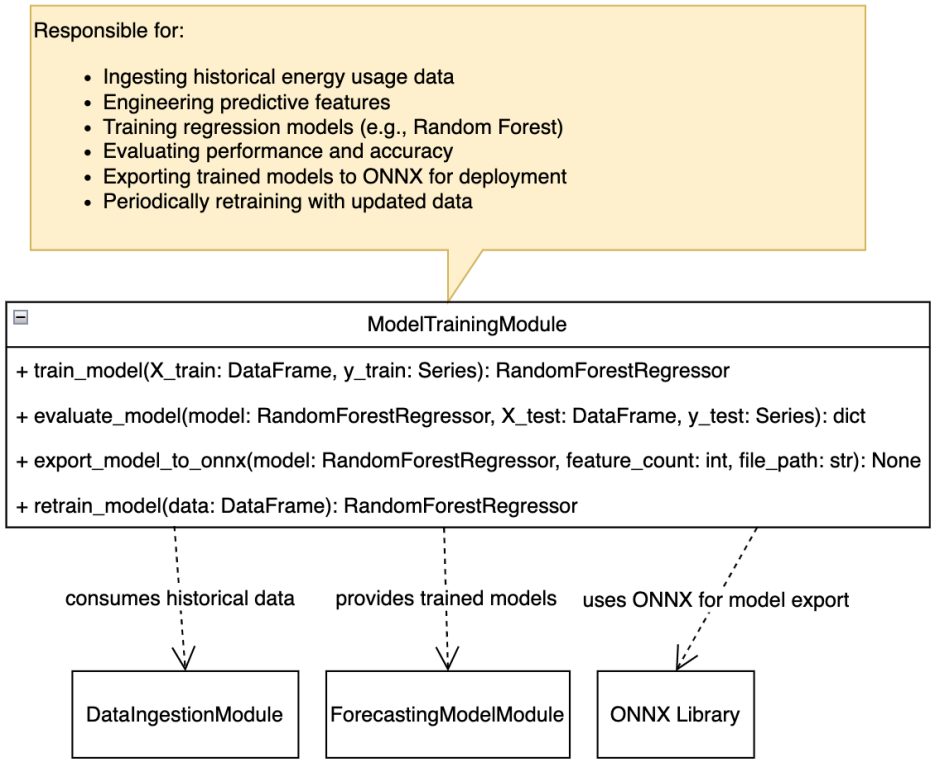

Figure 4.2.1: This diagram shows the Model Training Module, which is responsible for ingesting historical building data, training forecasting models, and exporting them into ONNX format for deployment.

The public interface for the model training module defines the core functions used to build and maintain forecasting models. These methods cover training, evaluation, exporting to ONNX format, and retraining to ensure that the system always has an accurate model ready for deployment:

- **train_model(X_train: pd.DataFrame, y_train: pd.Series) -> RandomForestRegressor**
  Trains a regression model using historical building energy data.
- **evaluate_model(model: RandomForestRegressor, X_test: pd.DataFrame, y_test: pd.Series) -> dict**
  Evaluates model performance and returns metrics such as mean squared error.
- **export_model_to_onnx(model: RandomForestRegressor, feature_count: int, file_path: str) -> None**
  Converts the trained model into ONNX format and saves it for later use.
- **retrain_model(data: pd.DataFrame) -> RandomForestRegressor**
  Updates the model with new historical data to improve accuracy and adaptability.

4.3 Forecasting Model Module:

The forecasting model module is responsible for the main driver of our project: predicting future energy consumption. It does not perform training itself, but instead consumes models that have been trained and exported by the model training module. The module loads the ONNX model, applies it to pre-processed input features such as time of day or day of week and outputs forecasts. This makes up the bulk of our deliverable for this project, so it is one of the more crucial modules to accurately integrate. This module will also support periodic retraining by analyzing past predictions and its overall accuracy. These predictions are then used by the optimization engine to recommend energy efficiency strategies and load balancing measures.

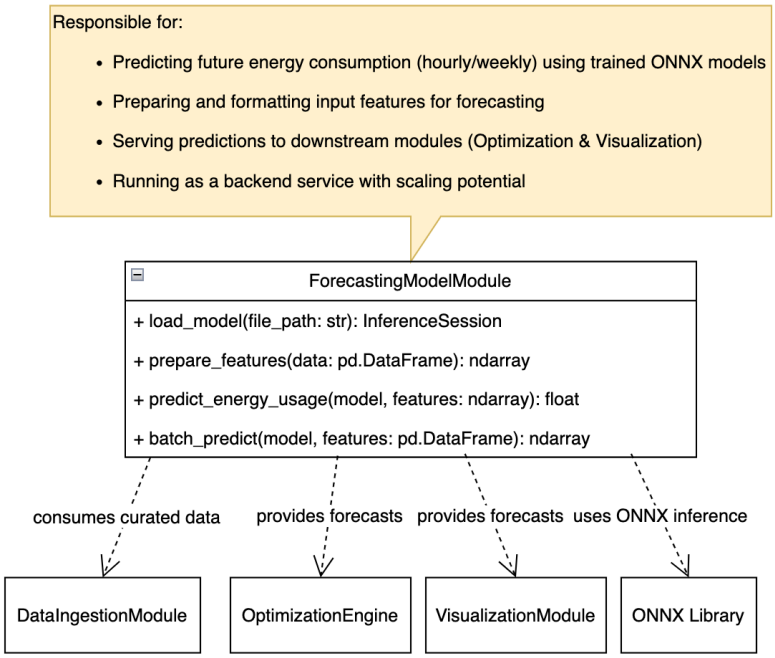

Figure 4.3.1: This diagram highlights the Forecasting Model Module, which loads trained ONNX models, prepares input features, and generates predictions for energy usage.

The public interface for the forecasting model module defines the functions responsible for loading trained models and generating predictions. These methods handle preparing input features, running the ONNX model, and returning energy usage forecasts that the other modules can act upon:

- **load_model(file_path: str) -> onnxruntime.InferenceSession**
  Loads a trained ONNX model for use in making predictions.
- **prepare_features(data: pd.DataFrame) -> np.ndarray**
  Transforms incoming raw building data into the appropriate feature format expected by the trained model.
- **predict_energy_usage(model: onnxruntime.InferenceSession, features: np.ndarray) -> float**
  Uses the trained model to forecast energy consumption for the next interval.
- **batch_predict(model: onnxruntime.InferenceSession, features: pd.DataFrame) -> np.ndarray**
  Performs predictions over a series of future time periods for scenario analysis.

4.4 Optimization Engine Module:

The optimization engine module is responsible for the second main part of our project: finding both problems and solutions for energy consumption on campus. This module will solve

control/dispatch problems to minimize the cost and consumption of energy while adhering to equipment constraints (e.g. demand charges). Another step of the optimization engine module is to produce actionable schedules and potential avenues to be taken on campus through various simulations of peak shaving, load reductions, and more. This module will leave a paper trail of its findings by logging all of its assumptions, constraints, and outcomes to ensure proper auditability. In the architecture, the optimization engine module will consume the output of the forecasting model for actuation, while still supporting human approval before taking any action.

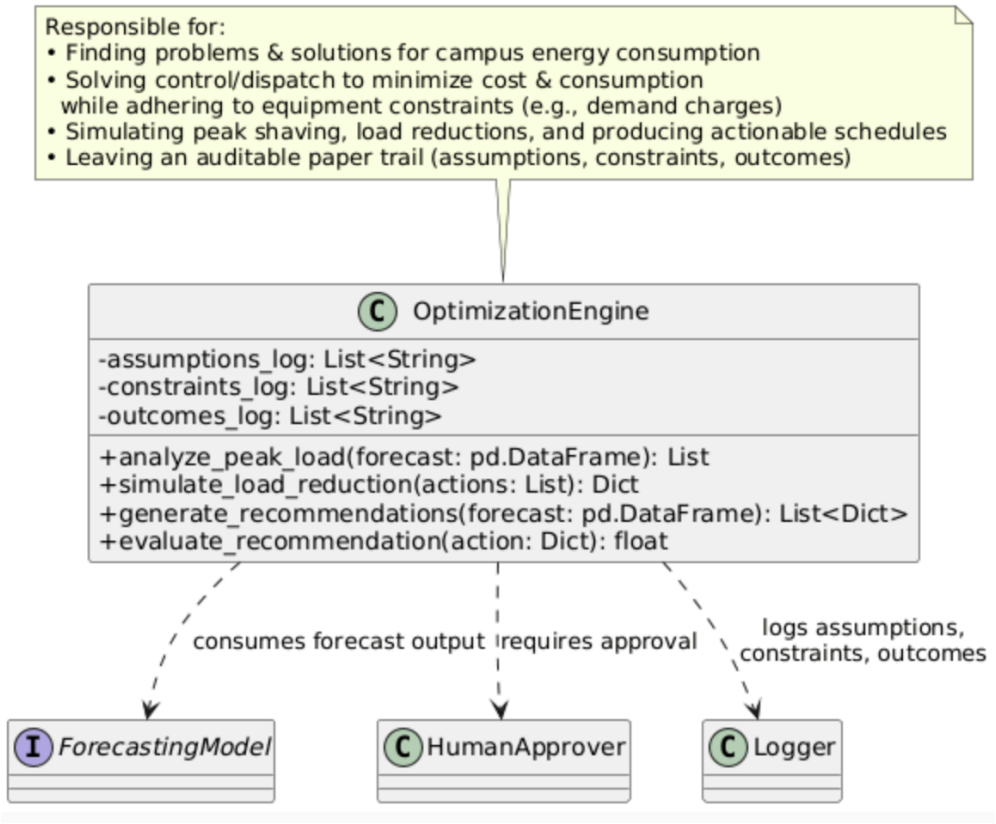

Figure 4.4.1: This diagram presents the Optimization Engine Module, which uses forecasts to simulate strategies such as load shifting and scheduling to improve building energy efficiency.

The public interface for the optimization engine module describes the methods used to analyze peak load periods, simulate potential energy-saving strategies, and generate recommendations for load shifting and scheduling:

- **analyze_peak_load(forecast: pd.DataFrame) -> list**
  Identifies time periods with the highest predicted energy usage and returns them as a list.
- **simulate_load_reduction(actions: list) -> dict**
  Runs simulations to estimate the impact of potential load-shifting or curtailment strategies and returns expected savings.

- **generate_recommendations(forecast: pd.DataFrame) -> list[dict]**
  Produces a list of recommended optimization actions (e.g., pre-cooling, schedule shifts) based on forecasted data.
- **evaluate_recommendation(action: dict) -> float**
  Returns an estimated energy or cost savings value for a specific optimization action.

4.5 Integration Module:

The integration module is responsible for making sure our overall system is easily integrated within the existing Willow digital twin platform. This module will orchestrate the flow of data between the start of our model and the final visualization on the digital twin platform. Because of the existing portability within the Willow platform, our system is set to easily be ported and integrated by using the ONNX packaging library. By request of our client OP Ravi, we will be building out our project using the ONNX library which will greatly simplify the final porting process which we have already tested on various prototypes to this point. In the architecture, the integration module is smaller on paper than the other modules, but just as crucial as any of them to ensure our final result can actually benefit the Willow platform.

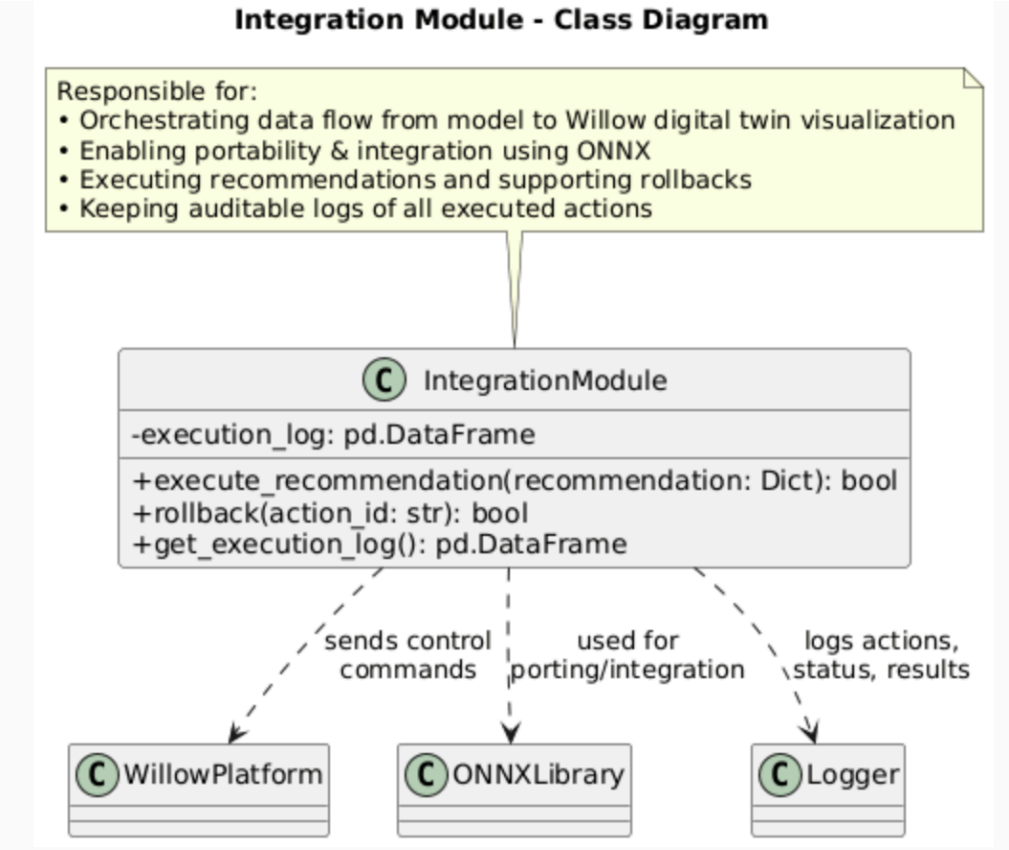

Figure 4.5.1: This diagram shows the Integration Module, which ensures WillowWatt connects seamlessly with Willow's digital twin platform by orchestrating data flow, executing recommendations, and supporting ONNX-based deployment.

The public interface for the integration module defines the functions that turn recommendations into real actions by communicating with Willow's system, as well as logging results and allowing rollbacks if needed:

- **execute_recommendation(recommendation: dict) -> bool**
  Sends a control command through Willow's platform to implement the recommended action and returns True if successful.
- **rollback(action_id: str) -> bool**
  Reverts a previously executed action if necessary and returns True if rollback succeeds.
- **get_execution_log() -> pd.DataFrame**
  Returns a log of all executed actions, including timestamps, status, and results for auditing.

4.6 Visualization Module:

The visualization module is responsible for providing dashboards for all components of our system (forecasts, optimizations, etc.). Similar to the simplicity of our integration module, much of this is already handled by the existing Willow platform. Willow's digital twin platform already has professional-grade visualization of all of the tracked energy on NAU's campus. Because of this, our model will be able to easily build out on top of the existing graphs and dashboards, simply adding a layer to show the forecast beyond the historical/live data. For optimization insights, our model can provide simple plain text messages to users through a simple terminal/chat window on the Willow platform. In the architecture, this is the final step for rounding out our project, and ensures everything that we create for Willow is easily accessible by users on the Willow platform.

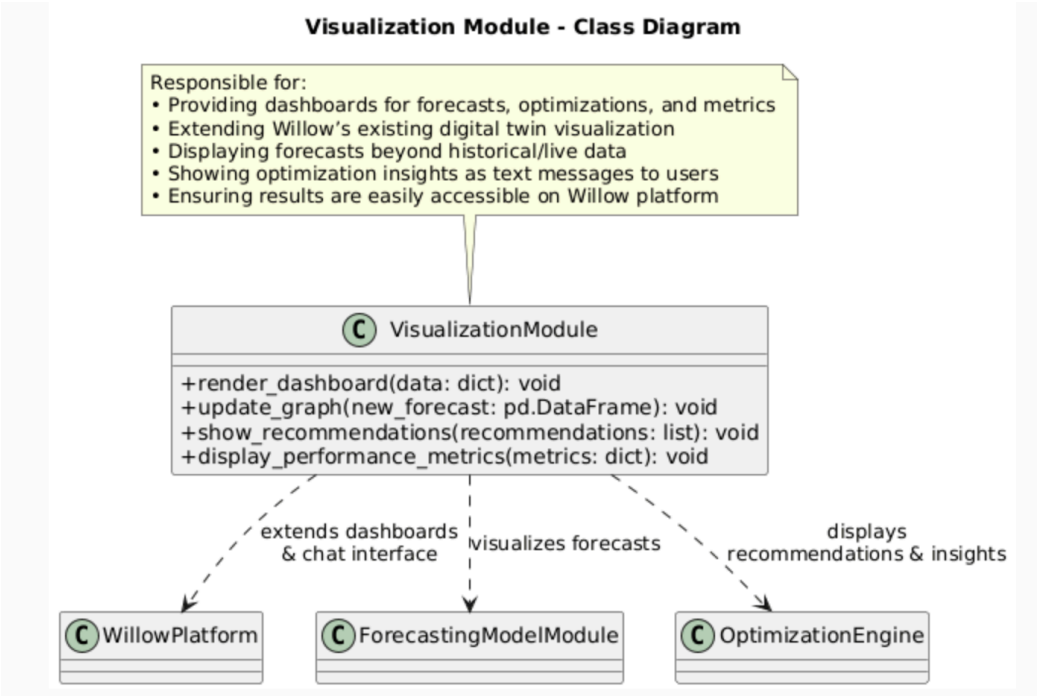

Figure 4.6.1: This diagram reflects the Visualization Module, which provides dashboards and reporting tools to display forecasts, optimization recommendations, and performance metrics to facility managers.

The public interface for the visualization module covers the functions that update the dashboard with forecasts, display optimization recommendations, and show performance metrics to end users:

- **render_dashboard(data: dict) -> None**
  Displays the main dashboard with energy forecasts, current recommendations, and performance metrics.
- **update_graph(new_forecast: pd.DataFrame) -> None**
  Refreshes forecast visualizations in real time using new data.
- **show_recommendations(recommendations: list) -> None**
  Displays a list of active recommendations and their expected impacts for user review.
- **display_performance_metrics(metrics: dict) -> None**
  Shows tracked savings, accuracy statistics, and carbon reduction metrics in the user interface.

5 Implementation Plan:

Our team has broken the project into three main phases: Energy Forecasting, Energy Optimization, and Energy Management. Each phase builds on the last, but we're also planning to work in parallel as much as possible so that we can keep making progress on multiple parts of the project at the same time. The figure below shows our updated timeline which outlines when we expect to complete each milestone, including time for testing and integration at the end. This plan is flexible and may be adjusted as we gain access to data, receive feedback from Willow, and learn more about system requirements during development.

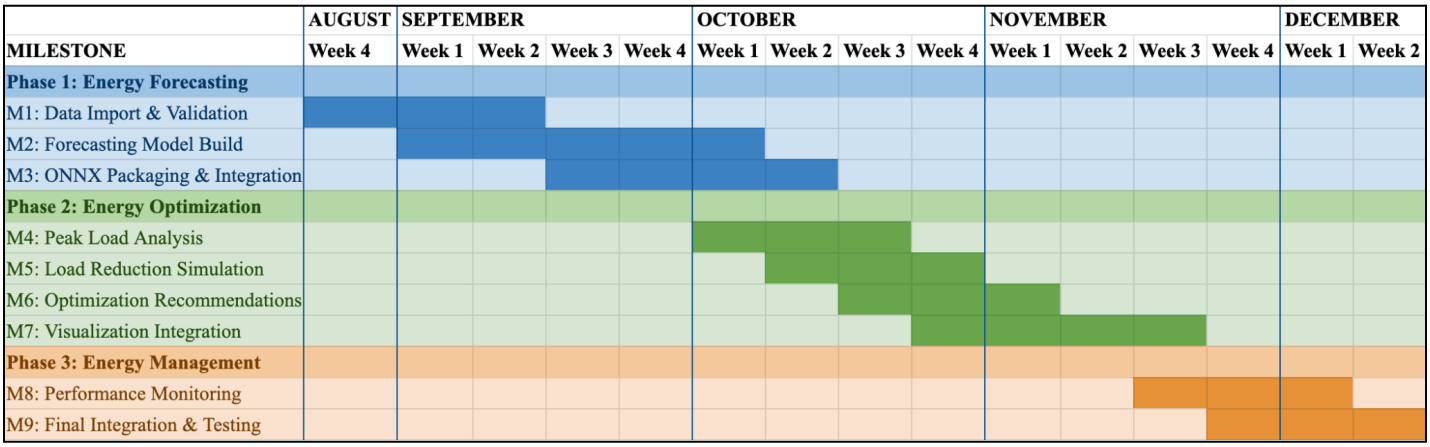

Figure 5.1: This Gantt chart outlines the implementation timeline for WillowWatt, showing the development, testing, and integration phases of each module.

5.1 Phase 1: Energy Forecasting

Our first step is to import and validate NAU's building energy data to make sure we're working with accurate and consistent information. This is crucial because the forecasting model will only be as good as the data it learns from. Once we have clean data, we'll build the forecasting model itself and test it with historical usage data to see how well it can predict energy demand. We'll keep refining the model until it gives reliable results. When we're happy with its accuracy, we'll package it using ONNX so it can connect to Willow's platform. This phase is the foundation for everything that comes next. Once we have solid energy forecasts, we'll be able to make strong optimization decisions.

5.2 Phase 2: Energy Optimization

While we're building the forecasting model, we'll also start working on peak load analysis. This will help us figure out when NAU uses the most energy and where we should focus our efforts to get the biggest impact. Once we know when the peaks happen, we'll run simulations to test potential strategies for reducing load. For example, we'll shift things like HVAC schedules or using battery storage differently. Next, we'll generate optimization recommendations and connect them to our visualization interface so they're easy for building managers to understand and act upon based on what we find. The goal for this phase is to turn forecasts into actionable suggestions that actually save energy and money.

5.3 Phase 3: Energy Management

Our last phase focuses on making sure the system runs smoothly and is ready for real-world use. We'll set up performance monitoring so that we can track how accurate our model is over time and how much energy we're actually saving with our recommendations. This will give NAU a clear picture of the system's impact. Finally, we'll integrate the forecasting model, optimization engine, and visualization dashboard into one complete solution and run full testing to make sure all of the parts work together as expected.

5.4 Team Responsibilities

Since this project has a lot of moving pieces, we've divided up the work based on each team member's strengths. Here's how we're planning to handle the main milestones:

Milestone/Module	**Primary Lead**
M1-M3: Data Import & Forecasting	Elliott
M4-M5: Peak Load Analysis & Simulation	Ayla
M6-M7: Optimization & Visualization	Luke
M8-M9: Final Integration & Testing	All

Even though each module has a designated lead, we'll all be working together closely and regularly reviewing each other's work to keep the project moving forward and ensure every part of the system integrates smoothly. We've also built in buffer time towards the end of the semester for debugging, testing, and making improvements based on sponsor feedback. We'll meet regularly with Willow to make sure our progress is on track and that our system is meeting their expectations. This way, by the time we finish, we'll have a well-tested solution that NAU can start using to save energy and move closer to its 2030 carbon neutrality goal.

6 Conclusion:

Our project, WillowWatt, was designed with one goal in mind: to help NAU use energy more efficiently, cut costs, and move closer to its 2030 carbon neutrality goal. Buildings consume an enormous share of global energy, and by combining Willow's digital twin platform with our AI/ML forecasting and optimization system, we have an opportunity to make a real impact. This document outlined our system's design, starting with our architectural overview and breaking it down into individual modules (data ingestion, forecasting, optimization, integration, and visualization) that each play a key role in turning raw building data into actionable insights. We also presented our implementation plan, which lays out how we will build, test, and integrate each piece of the system while working in parallel as a team.

Beyond addressing NAU's immediate needs, our design is intentionally modular and scalable, meaning it could easily be extended to other buildings and campuses in the future. We're building a solution that can grow with Willow's platform and support a much broader range of energy optimization efforts over time by focusing on flexibility and clear interfaces between components. By completing this design, we now have a clear roadmap to move from concept to prototype. The next steps will involve building and testing each module, connecting them together, and validating our results with real NAU data. Our team is confident that this plan positions us to deliver a solution that will not only strengthen Willow's platform but also create measurable energy savings for NAU.