Green Lean Electrics

M7011E, Design of Dynamic Web Systems

Github repo:
https://github.com/PandaFood/M7011E_EnergySystem

Jonathan Brorsson, *jonbro-6@student.ltu.se*

Gustav Hansson, *gushan-6@student.ltu.se*

LP2 2019-2020

1 Introduction

The system is divided into 3 mayor parts. The Front End, the Authenticator and the Simulator.

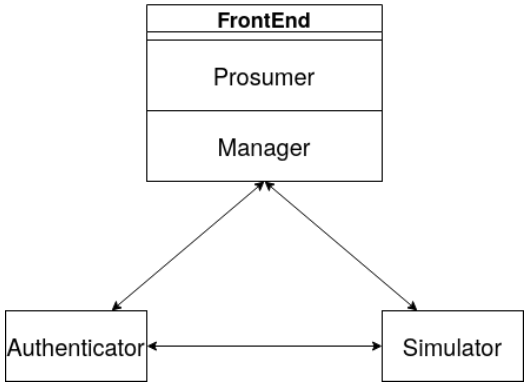

Figure 1: System Overview

1.1 Simulator

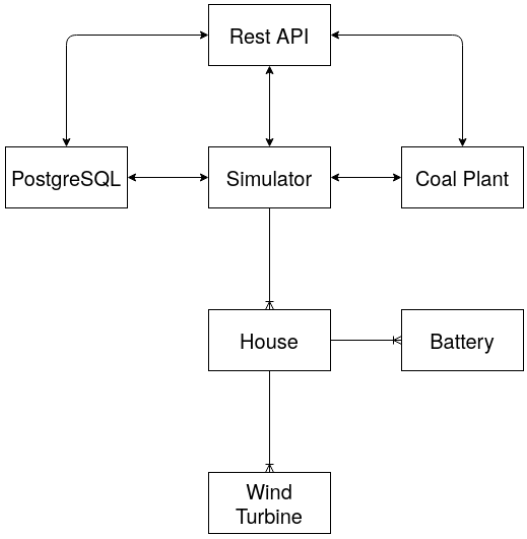

Figure 2: Simulator Structure

The simulation is constructed with a main Simulator Object whose responsibilities is to initializing and run the simulation. This object creates the houses with

a One-to-Many relation. Each house contains a list of batteries and a list of wind turbines.

The simulation is polled using a poll time that for the moment is set to one second. During each poll each house calculates how much power is generated by its turbines, how much should be stored in the batteries and how much power is consumed. Any excess power is sold to the market and if the production of power does not meet the demand power will be bought from the market. If the market can not supply enough the house will experience a blackout. The current state of the house is stored in the database and an event for each turbine and battery is stored as well.

During each poll the Coal Plant's status is checked, if it is running the plant will generate power and store it in its battery as well as sell a part to the market. How much is sent to the battery can be set by the manager via the API. A recommended price for power is calculated each poll as well. This price can be seen by the manager and a call to the API can set the current price to either a manually chosen value or to follow the recommended calculated price. The calculation on the price follows a supply and demand model where higher demand from the houses increases the price and more power stored in the market decreases the price.

New houses, turbines and batteries can be created via the API. These new entities are loaded into the simulation during run time ensuring a smooth simulation without to many hiccups.

The Coal Plant can be turned on and off by the Manager via the API. Startup takes 30 seconds before the plant starts generating power and shutdowns takes 20 seconds before it stops generating power.

1.1.1 Wind and Power

The wind speed is based on Simplex Noice. When the simulation starts a base map of 200 by 200 pixels of simplex noise is generated. To simulate the wind a new noise map calculated and added on top of the base map for each interval. On top of this the value of four added sinusoids are added as well. This gives a nice semi random moving noise map where positions close to each other have similar wind speeds. It also gives areas with a higher probability of higher speeds simulating differences in terrain.

The wind turbines uses a pair of coordinates between 0 and 200 to get the position in the noise map. The noise map then gives a value between 0 and 1 and is sent into a function that calculates the actual wind speed. The function looks as follows with w being the wind speed and n being the value from the noise map.

$$w = 7 \cdot (n^5 + \frac{n^3}{3} + 2 * n) + 3.5 \tag{1}$$

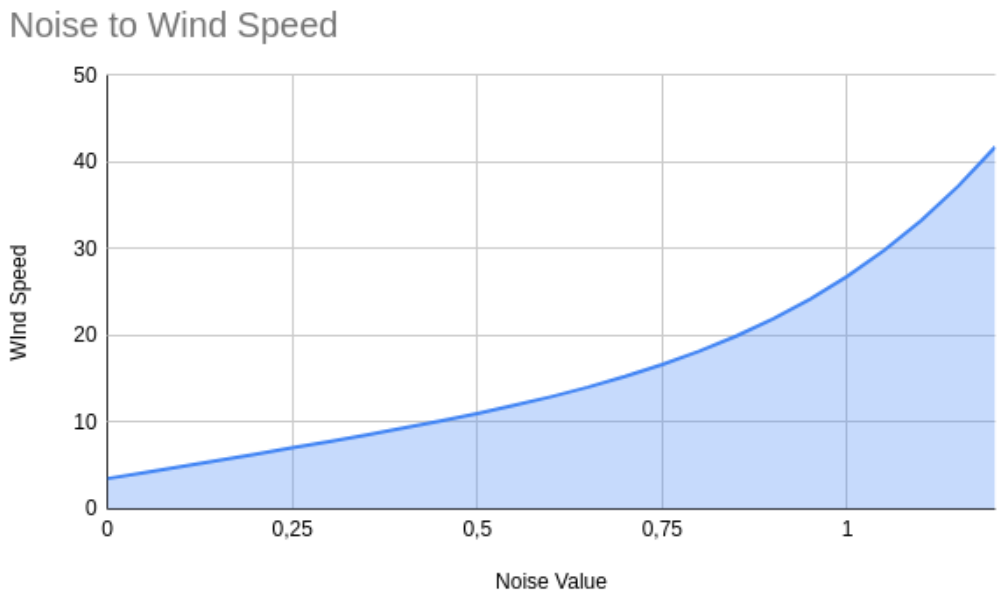

Figure 3: Noise to Wind Speed

The actual power is then calculated with a sigmoid function that makes speeds higher than $14m/s$ generating the same power and speeds less then around $3m/s$ gives 0 power. When the wind speed goes higher than $25m/s$ the generation of power is stopped until the speed lessens.

The Wind Turbines can also break down with a random chance. This chance gets higher the higher the wind speed and when the turbine breaks it is broken for around 40 seconds.

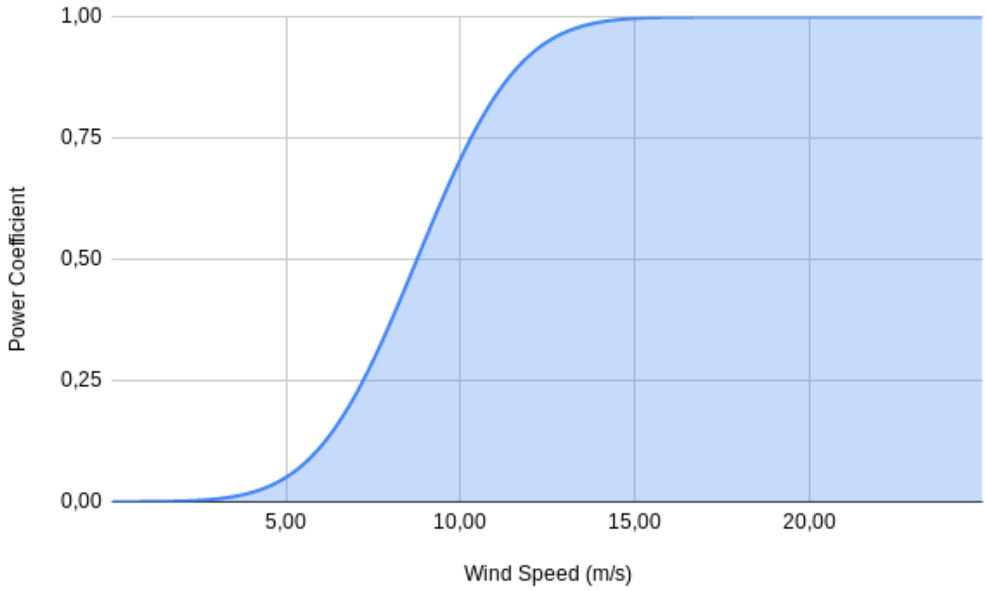

Figure 4: Power Coefficient

1.2 Front End

1.2.1 Prosumer

When a user logs into the website the dashboard is shown. The dashboard gives the user a quick and clear picture of the state of the turbines and batteries that the user have. A field also shows information like the users total production and consumption as well as the current price the market demands.

Two buttons on the dashboard opens up a modal where the user can create a new turbine or a new battery respectively.

The turbines and batteries are each shown in a table where each row represents one turbine/battery. The row shows the current state of that specific entity and when clicked a new tab opens with more information. This new tab shows a graph with the historical data collected during the simulation.

1.2.2 Manager

When the manager logs in a dashboard displaying each user is shown. A button to a control panel opens up a new tab with information about the system and control over the Coal Plant and the price.

2 Design choices

2.1 Simulator

We chose to run the simulation on its own node server and thus split it from the actual website. This was done to simulate a more realistic scenario where the system monitoring the power grid runs on a system independent from the website.

We also chose to give a user the ability to have as many wind turbines and batteries as the user wants. This was done because we though that it would be more of a realistic simulation if it could handle a non static amount of turbines and batteries per user.

2.2 Front End

For the front end the framework Vue.js was chosen because of its component based design and how it handles real time updates of these components. This made it easy to apply the dynamic aspect of the website without the hassle of using native Javascript or JQuery.

To minimize the amount of code needed and time spent the manager site uses the prosumer part to show the prosumers system. This means that time spent on the prosumer part also gives a big part of the manager page without any extra work.

3 Scalability analysis

The current system is build for easy scaling. Except for key management in the authentication server, everything is built microservice-like, which easily can be scaled up and down according to demands.

4 Security analysis

With the purpose of being a power-management system, security was deemed to be of high priority.

4.1 Authentication

For the authentication method between the website and the API, a simplified version of Oauth 2 was chosen. Due to time constraints, some features of Oauth 2 was left out, such as refresh tokens and different authentication methods.

JSON Web Tokens [2] are used along with JSON Web Signature [1] to verify the identity of a user. A basic overview of how it works is shown in figure 5.

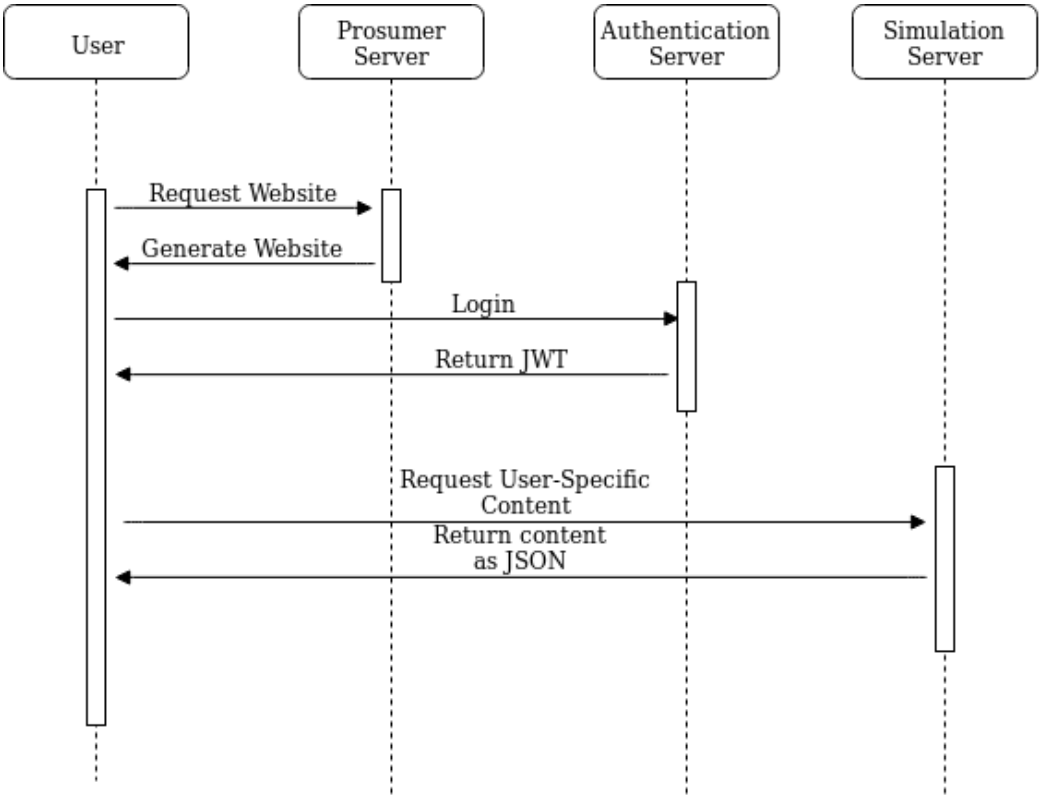

Figure 5: Website flow

Currently, the website doesn't have a certificate, due to a lack of a domain name. When deploying, this should be the first thing to do, which should stop data and passwords from being transmitted in clear text.

4.2 Attack Vectors

The project mostly uses resilient frameworks which mitigates common attack vectors such as SQL-injections, Cross-side scripting and XSRF. There are however much larger and more likely attack vectors which are left open. The project has no defence against malicious NPM dependencies.

The databases, although behind an access filter, has easily guessed passwords and usernames to access. This should also be considered a security issue, in case a malicious actor gets access to the server or a coding error is introduced.

5 Advanced features

5.1 Simulator

5.1.1 Breakdown

The implementation of the Wind Turbines supports breakdown of the turbine. The breakdown uses the function:

$$t = 0.25 \cdot w \tag{2}$$

where t is the threshold and w is the wind speed. Then a random number between 0 and 1 is generated and if this number is less then the threshold a breakdown occurs. Then a breakdown time is generated with a guassian distribution with a mean of 40 seconds and a standard deviation of 5 seconds. When this time has passed the turbine is up and running again.

5.1.2 Locational Data

The locational data is implemented using the noise function explained in the introduction.

5.1.3 Historical Data

Each interval the simulator does stores the generated data of each turbine and each battery in the simulation. Thus historical data of the simulation is stored and can be accessed by the website to be displayed.

5.1.4 API Documentation

Every API needs some kind of documentation, without it a developer would have more challenges than necessary when using it. Thus we use Swagger to document all the functionality our REST API provides.

6 Challenges

As both members of the group have worked a lot with developing website no mayor challenges was faced during this course.

7 Future work

7.1 Performance

As the site works at the moment performance can be an issue. The front end does a lot of requests to the Simulator API each second. This can make the website perform poorly when it is open in the browser a longer period of time. Time could be spent looking into using web sockets to stream data instead of polling. Another possible way to increase performance is to fetch more data with fewer get requests.

As of now the graphs showing historical data of the wind turbines and the batteries are capped on 5000 elements. This is because the graph performs very poorly with a lot of inputted elements. This is something that should be solved with a better solution.

7.2 House Power Consumption

As of now the user can manually set how much power the house consumes. To improve the simulation this value should be calculated with time of day and day in year in mind. This could be calculated by using a sinusoid for

the year, winter gives a higher consumption and summer gives a lower. This sinusoid could be added with another simulating the time of day where power use decreases during night and increases during the day. One could also add a random term to this to increase the variance in the simulation.

7.3 Subproject separation

Currently the dockerfiles are not up to date and everything has been deployed in a single docker environment using docker-compose. To better deploy and scale, the dockerfiles needs to be updated.

7.4 Key/Authenticaiton management

Asymmetric keys are currently generated on and stored in an environment variable on startup of the authentication server. They keys need to be stored in a safe location aswell as not being generated on startup.

Authentication of databases are also using a default username and password for access, which should be managed better.

7.5 User Control

The user settings page needs to be updated with functionality to edit the user information. This should be accessible by the manager as well.

Functionality to reset passwords are also missing and should be a high priority fix in future updates.

7.6 Manager User View

As of now the manager can view the system of a user. However it is not very clear which user is selected at the moment as well. To increase the usability of the manager system this should be fixed.

The manager should also be able to see which user is currently logged in. This is not implemented at the moment.

7.7 Buy percentage from market

When a prosumer can't meet its own demand for power the needed power will first be drained from the batteries. If these batteries are empty the prosumer will buy from the market. This should be changed so that a percentage of the needed power is taken from the batteries and the rest bought from the market.

7.8 Profile Picture

As of now a user can not upload any profile picture. This should be fixed by adding a option to upload a picture to the auth server and store a path to this image in the user table. If no image is uploaded a default image should be shown.

7.9 OAuth

Currently, while working, the authentication is not fully implemented and therefore not as secure as it can be. Implementing refresh tokens, shorter timed access tokens could lead to a more secure system.

7.10 Logging

Logging of events in the system is currently close to none, leading to hard to trace issues and in the event of a data breach, the perpetrator would most likely not be found. Expanding on the logging and mining that information would therefore be very useful.

7.11 Testing

The simulation have a test suite that covers some of the calculations in the simulation. However this suite should be further developed to ensure the correctness of the simulation.

The Front End lacks tests as well which creates a high risk of regression.

A Time Report Analysis

A.1 Gustav

During the month of November I spent around 32 hours working on the simulation. After November some time here and there was spent to further develop and improve the simulation when I noticed problems and missing features. In the beginning of January I spent around 10-12 hours reworking some parts of the simulation with focus on loading new entities without a need to restart and I remade how the Coal Plant works with its battery.

During the month of December I began work on the prosumer page. First I spent a few hours working on how to display simulation data over time with a graph. Then I spent around 10 hours working on the prosumer dashboard using EJS. This was however scrapped when we decided to move to Vue.js. The rest of December was spent on redoing the dashboard in Vue.

In January I have spent around 36 hours working on both the prosumer and manager page as well as doing some updates to the API and simulation when needed.

A.2 Jonathan

Due to our experience with web systems, the first week was spend on mostly designing and selecting which technologies we would use. I took this opportunity to check out some that i hadn't tried before. I also wrote part of the API. I spent about 15 hours on writing the API and database connection, with a couple hours additionally designing the system together with Gustav.

After that I set out to build the authenticator, spending around 27 hours on that. During that, some time was also spent evaluating new technologies, and setting up NGINX for the project to get a better and more professional website under one domain.

Some time was also spend on building the frontend, around 20h. At the end of the project, we set aside time to finalizing and writing the report.

B Contributions

The work was initially divided into Gustav working on the simulation and Jonathan working on the authentication with OAuth and the dockerization of the project. Further on the focus for Jonathan was the login and authentication of prosumers and manager as well as the checks for authentication on the websites as well as the API. Gustav focused on the user interface for the prosumer and the display of real time data.

When these things was done we went over what issues we still had to solve and where each member just took a issue and began to solve it.

We agree that a 4 for each member would be a fair grade on this project.

C Links

Github repo: https://github.com/PandaFood/M7011E_EnergySystem

Github release: https://github.com/PandaFood/M7011E_EnergySystem/releases

References

[1] *RFC 7515 - JSON Web Signature*. URL: https://tools.ietf.org/html/rfc7515.

[2] *RFC 7519 - JSON Web Token*. URL: https://tools.ietf.org/html/rfc7519.