Documentation:
On-Premises Energy Management Dashboard

Nicole Hessner
IP05, CS504 – Software Engineering, Masters of Science in Computer Science, School of Computing and Technology, City University of Seattle
hessnernicole@cityuniversity.edu

About the On-Premises Energy Management Dashboard

In today's business environment, sustainability is not just a buzzword or a lofty, moralistic goal for the betterment of our children and grandchildren. Typically, the goals of sustainability align with the goals of Lean, because reducing waste is such a core aspect of both philosophies. The goal of this software is to enable organizations to pair their existing utility data, whether from their monthly utility bill or from continuous monitoring on-site, with their existing maintenance database to help objectively prioritize maintenance jobs by cost and utility waste. This documentation is a shell or outline of a hypothetical software system, and may not represent everything that is required for a functional piece of software. The sections of this documentation are based loosely on the documentation for the Keras Python library.

Keywords: Energy Management, Lean, Sustainability, Software Engineering

1. GETTING STARTED

This section would contain quick start guides for different users reflecting their common use cases. This is the manual for people who don't like to read manuals and probably don't need to know *how* the software works as much as they need to know how to make it do what they bought it for.

For Maintenance Professionals
VIEWING PRIORITIES

FLAGGING PRODUCTION CRITICAL MAINTENANCE JOBS

For Energy Management Professionals
UPDATING PARAMETERS

VIEWING SAVINGS METRICS

For Business Managers
SAVINGS REPORTS

For Database Administrators
PROTECTING YOUR DATA

2. DEVELOPER GUIDES

This section would contain references for internal developers who may maintain the software or be interested in developing their own custom modules and features.

Use Cases

Architecture

Testing Procedure Recommendations

3. API REFERENCE

As the software is developed, this section would include references for all of the Namespaces and Classes, and their associated parameters and attributes. Since this software has a lot of elements and I did not actually develop it over the course of this class, this is a very rough list. The hierarchy is:

Namespace

- Class
- Function(Parameters)
    - Attributes

With further development, these different aspects of the software would have functional descriptions included, as well.

User Interface

- HomeScreen
- PriorityScreen
- ExportSavings
- UpdateUtilityCosts
    - ElectricityCost
    - NaturalGasCost
    - SteamPlantEfficiency
    - WaterCost
    - SewerCost
    - EstLightWattage
    - AvgHVACMotorSize
- JobRank
    - FlagProdCritical
        - Removes job from the ranking queue due to opportunity cost from stopped production being out of scope for the prioritization program.

Utility Savings Module

- MAINTENANCE DATABASE CONNECTIONS
    - GetOpenJobs
        - JobPriority
        - JobType
        - JobStatus
        - OpenDate

Cost Savings Module

- CalculateCost
    - LightingCost(OnTime, OffTime, EstLightWattage, ElectricityCost)
    - HVACCost(StartTime, StopTime, AvgHVACMotorSize, ElectricityCost)
    - WaterCost(lossRate, WaterCost)

Database Connections

Because of the wide variety of proprietary databases and industry-specific data communication protocols, it may be necessary to work with our team to build a custom solution to get the type of connectivity you would like for your custom energy management dashboard solution. If this applies to you, please reach out!

- MAINTENANCE DATABASE CONNECTIONS
    - GetOpenJobs
        - JobPriority
        - JobType
        - JobStatus
        - OpenDate
    - GetJobHistory
        - JobType
        - ActiveWorkTime
        - MaterialsCost
- HVAC CONTROL SYSTEM CONNECTIONS
    - GetHVACSchedules
        - StartTime
        - StopTime
    - GetHVACSetpoints
        - CoolingTemp
        - HeatingTemp
    - GetHVACRunTime
        - RunTime
    - GetOutsideAirTemp
        - OATemp
    - GetInsideAirTemp
        - IATemp
- LIGHTING CONTROL SYSTEMS
    - GetLightingSchedules
        - OnTime
        - OffTime

- LOCAL UTILITY MONITORING
This will be specific to local metering capabilities, so the following are examples.

    - InstantaneousFlow
    - TotalFlow
    - kWDemand

4. CODE EXAMPLES

Coming Soon

This section would be filled with code examples in an actual development project.

5. WHY CHOOSE US

Your energy management dashboard will be customized to your specific data management needs and utility metering capabilities. Additionally, we pride ourselves on extensive testing to ensure that we are meeting our high benchmarks for reliability, availability, and user experience.

Metrics

Metric	Value
API Availability	99%
API Error Code	All error codes captured and logged correctly
API Response Time	≥200ms
API Latency	≥30ms
Maintainability Index	≥20

Cyclomatic Complexity	≤4
Depth of Inheritance	≤5
Class Coupling	≤9
Lines of Source Code	Minimize
Lines of Executable Code	Minimize

6. CHANGE MANAGEMENT

Change Management will need to be managed per user contract, due to the high level of customization of the software. However, it would include a rough schedule for high level version releases of core features that are universal across contracts.

7. AFTER ACTION REVIEW

Over the course of CS504 – Software Engineering, I learned a lot very rapidly. Since this was my first quarter after taking my Bridge courses, this course was a very welcome introduction to more of what the field of Software Engineering entails. I was pleasantly surprised to find that computer scientists and software engineers follow a lot of the same processes and procedures that I learned as a mechanical engineer, but applied to software, especially with respect to the Requirements Engineering assignments.

Because of how new the field is to me, there were several moments during the course where I took on more than I was necessarily ready for, and my ability to adhere to deadlines suffered. This is not necessarily a new phenomenon for me – I have a tendency to struggle with just-in-time learning, because I naturally tend to seek depth of understanding. I also am more of a big picture thinker, so I tend to grasp concepts very easily and don't always successfully extrapolate the amount of effort of execution related to those concepts. And, when I don't grasp concepts quickly, I tend to take a "not all who wander are lost" approach to the material until I find a path out of the metaphorical woods. However, this course was a good reminder of my resolve, perseverance, and commitment to the process of learning.

Having done all of the projects, I probably would have chosen a simpler example with fewer moving parts and less requisite customization. I chose an energy management portal as my example, because it is the only software project I have really been a part of in the real world, even though I was on the user/customer side of the development process. I often felt like I did not have a great understanding of exactly how to undertake the projects until I was about halfway through, but found a lot of web resources that I will trust as references in the future.

If I were to develop this software into a real product, I would move this documentation onto the web and make it interactive, possibly using Blazor.