Requirements Document for: Scalable Web Application Framework for Monitoring Energy Usage on Campus

Daniel Schroeder, Aubrey Thenell, Parker Bruni

✦

Abstract

Data visualization is an important technique for analyzing data trends and identifying problems that may not have been apparent. This document outlines our plan to create a web application which will gather energy data from data aquisition servers and create information management dashboards. Our application will allow users to generate charts and graphs to see energy use over time, compare energy usage across multiple buildings, and create collections of different visualizations.

CONTENTS

1 INTRODUCTION

1.1 Purpose

The purpose of this document is to outline the project and all associated information. Furthermore, it will outline how the end product will be used, how it was developed, and all resources and documentation referenced and used during development. This document will also describe the applications target audience, user interface, and any hardware/software requirements. Finally, it will define how our client, team, and users will see the final product and related functionality.

1.2 Scope

This project will provide a web application that displays different collections of energy use reports for campus buildings. Each of these collections will be customizable, with the ability to add, remove, and filter building data according to user preference. The web application will allow users to visually comprehend energy data in a way that can raise awareness about consumption and reduce the campus-wide carbon footprint. The web application will also have three different permission levels: Guest, Registered Users, and Admin Users. Each one will provide permissions to what can be viewed and edited.

1.3 Definitions, acronyms, and abbreviations

Term	**Definition**
OSU	Oregon State University
MEAN stack	MongoDB, Express, AngularJS, Node.js
JSON	JavaScript Object Notation
AcquiSuite	Data Acquisition Servers responsible for metering energy usage
Collections	User Interface dashboard that displays assortment of blocks
Blocks	Charts/graphs of building data
Stories	Collections of dashboards

1.4 References

The references are:

[1] “AcquiSuite-A8812”, AcquiSuite — English, 2017. [Online]. Available: http://www.obvius.com/Products/A8812. [Accessed: 02Nov2017].

[2] “Data-Driven Documents”, D3 — English, 2017. [Online]. Available: https://d3js.org. [Accessed: 02Nov2017].

[3] “Vis.js”,Vis.js — English, 2017. [Online]. Available: http://visjs.org/. [Accessed: 02- Nov- 2017].

[4] “MongoDB, Express, AngularJS, Node.js”, MEAN.IO — English, 2017. [Online]. Available: http://mean.io. [Accessed: 02- Nov- 2017].

[5] “Introducing JSON”, JSON. — English, 2017. [Online]. Available: http://json.org [Accessed: 02- Nov- 2017].

[6] “BuildingOS”, BuildingOS — English, 2017. [Online]. Available: https://buildingos.com/. [Accessed: 02- Nov- 2017].

[7] “Modbus How Does Modbus work Modbus Tutorial”, Learning Engineering — English, 2016. [Online]. Available: https://www.youtube.com/watch?v=jfKY928Ozrw/. [Accessed: 02- Nov- 2017].

[8] “AcquiSuite EMB”, AcquiSuite — English, 2017. [Online]. Available: http://www.obvius.com/Products/A8810-0/. [Accessed: 02- Nov- 2017].

[9] “Modbus Addressing & Wiring Best Practices”, obviusenergy — English, 2017. [Online]. Available: https://www.youtube.co
[Accessed: 02- Nov- 2017].

[10] “Mocha”, Mocha — English, 2017. [Online]. Available: https://mochajs.org/. [Accessed: 02- Nov- 2017].

1.5 Overview

The remaining sections of this document provide a more elaborate description of product functionality, assumptions, and specific requirements. Section 2 provides information about functional requirements, data requirements, and assumptions made for designing the *Scalable Web Application Framework for Monitoring Energy Usage on Campus*. Section 3 outlines the specific requirements for the final product, the external interfaces that communicate with the software, and the functional requirements of the system.

2 OVERALL DESCRIPTION

This section will provide an overview of the web application as a whole, including:

1) Details about the user interface and expected user interactions.
2) An outline of specific constraints and dependences included in development.
3) Background information about the specific software requirements.

2.1 Product perspective

This product will replace the current web application that is used by The Oregon State Sustainability Office. The product will display energy usage information about Oregon State University buildings through an intuitive user interface. The application will gather data from energy meters by connecting to AcquiSuite^{TM}data acquisition servers. The energy data from the meters will be transferred to a database that the product will access directly. Users will interact with the application interface from an internet browser application.

2.2 Product functions

The product will allow users to create accounts that may be given either administrative or user permissions. Users have the ability to personalize their accounts. Administrators accounts will have special capabilities that user accounts will not.
The application will allow users to create customizable dashboards that will contain easily adjustable blocks of campus building data that will contain the campus building energy usage data. These blocks of data will be the basic building blocks for the dashboard and will provide an intuitive view of the data. They will feature various graph types, building energy efficiency rankings, and data trends.
Each OSU building that contains the energy monitoring meter(s) will have a specific, non-customizable page that will display general information. The product will also have a public interface and a private administrative interface.
Administrators of the application will have the ability to add, remove, or edit entire buildings profiles, building subspaces, or individual meters.

2.3 User characteristics

A user that will be using the general public UI will not need to know any specific information about the application to navigate the various energy data presentations. A public user will be able to intuitively navigate the UI at their discretion.
An administrative level user will need a basic understanding of the tools of the application because they will be allowed the freedom to control parts of the website as well as have access to more specific energy data within the application. An administrator will likely not need extensive training to use this application for more specific purposes as the administrator UI will be designed to be intuitive to navigate.

2.4 Constraints

Data updates will be limited to a granularity of 15 minute intervals. The data acquisition server is capable of providing a granularity of up to 15 second intervals but it is not necessary for the purposes of the application. Any other constraints on the product are subject to the internet browsing interface that is accessing our product. The UI will be intuitive and able to navigate data presentations, but will not allow manipulation of the data or underlying structure of the application.

2.5 Assumptions and Dependencies

The application will be dependant on the data acquisition server. If the data acquisition server is drastically changed or removed, the application will not be functional. Meter data within the application will also be limited by the functionality of the meters themselves. Should a meter malfunction, the energy data will not be gathered which may cause some of the data presentations to deviate from expected data. The use of the application will require a compatible web browsing interface, which may be limited to browsers such as Firefox, Chrome, and Internet Explorer. The functionality of the product will depend on those internet browsers performing as expected with regards to their intended functionality and internet access.

2.6 Apportioning of Requirements

Future versions of the application may include features such as cost tables, automated electronic invoice generation, energy billing analysis capabilities, budget analysis capabilities, and mobile energy data entry.

3 SPECIFIC REQUIREMENTS

3.1 External Interfaces

AcquiSuite data acquisition servers [meters] made by Obvious

- Used for collecting electric, water, gas, steam, and other energy parameters over the web.
- Data is received through IP-based connection.
- Data can be reached anywhere with an internet connection, as long as the AcquiSuite is online.
- Data will be collected every 15 minutes.
- Data from the AcquiSuite servers will be stored into a database.

3.2 Functions

This section defines how the software system should behave with regards to input and output.

- The system shall receive and store data from AcquiSuite data acquisition servers into the database.
- The system shall have input validation measures in place to monitor incoming data and protect from malicious injections.
- The system shall retrieve data from the database and populate webpages with filtered datasets when prompted.
- The system shall have permission based restrictions for accessing certain data.
- The system shall generate alerts for offline buildings and high energy usage.
- The system shall generate emails for users to access a sign-up form and create an account.
- The system shall be able to create arbitrary combinations of datasets given user filters.
- The system shall be able to calculate rankings for all the buildings based on certain metrics like “kilowatt-hour consumption”.

3.3 Performance Requirements

This section describes the functionality requirements for the software as well as what a user should be able to accomplish when using the web application.

3.3.1 Users

The system will have 3 types of users: admin level users, registered users, and generic users. An administrative user will be able to add new meters, buildings, and other objects into the database. A registered user, for example the head of a department, can create their own stories with information and collections that are unique to their own interests or departments. Lastly, a generic user can simply view the publically facing web application and see public dashboards and browse public stories.

- A registered user should be able to customize dashboard layouts in a grid-based orientation. A *story* page should have a customizable layout where a user can add different *blocks* with information relevant to their personal needs.
- An administrative user should be able to add buildings to the database through a web form.
- An administrative user should be able to add data aquisitions servers.
- An administrative user should be able to download specific datasets as a .csv file.
- An administrative user should be able to customize public stories.

3.3.2 Data Management

- The web application should update data blocks every 15 minutes as new data is received into the database.
- The web application should allow users to create which are collections of dashboards. These *stories* are meant to bring related buildings and datasets together into intuitive groups, for instance “Residence Halls” or “Engineering Buildings.”
- The application should be able to filter building data by date range specifications.
- The web application should be able to scale up to as many buildings are on campus.
- The web application should properly create and store new entities (buildings, users, meters) into the database.

3.3.3 Visualization

- The web application should be able to generate different types of graphs for different *blocks*. For example load-profile charts, comparative line charts, and heat maps.
- The web application should have generic pages for each building in the database. This page will display a series of graphs and charts that outline energy usage for a particular building.
- The *blocks* on each page should automatically update as new data is received by the database.

3.3.4 Objects

The system should have three main entities:

- **Building** - A building on campus that is connected to a data acquisition server.
- **Meter** - A specific data acquisition server.
- **User** - A user with a specific role.

3.4 Logical Database Requirements

Most of the calls to the database will be requests for meter data from a specific building or group of buildings. Access to the database should be hidden from the user and only accessed from the back-end of the application itself. Any input into the database should be validated and encrypted, if applicable.

3.5 Design Constraints

Design constraints may include server availability which could harm scalability.

3.5.1 Standards Compliance

There may be standards when storing energy data based on The Office of Sustainability Standards. [Check with Client]

3.6 Software System Attributes

3.6.1 Reliability

The system will be reliable at the time of deployment if all data displayed in graphs and charts is correct.

3.6.2 Availability

The system should be consistently available as long as the servers are up and running. As a web application, the system will be available via URL.

3.6.3 Security

The system will validate input for duplicate and malicious content. The system will also have back-end functionality to protect user passwords through proper hashing. The web application will restrict access to certain pages and content based on user roles and permissions.

3.6.4 Maintainability

Maintainability for this application should be simple for anyone with web development experience. The application code will be organized into logical directories and sub directories that should mimic a full stack web application. There will be documentation that explains how to connect to the AcquiSuite servers and how to use our API for data collection.

3.6.5 Portability

Our web application should be visible and accessible from most web browsers. There are not any host-dependent constraints since it will be hosted from a central server and available to the public via the internet. The source code should stay on Github so modifications can be made easily.

Gantt Chart (Months)