Oregon State University
College of Engineering

CS Capstone Technology Review

November 13, 2017

A Scalable Web Application Framework for Monitoring Energy Usage on Campus

Prepared for

Oregon State Office of Sustainability

Jack Woods _____ Signature _____ Date

Prepared by

Group 57

The Dream Team

Daniel Schroeder _____ Signature _____ Date

Aubrey Thenell _____ Signature _____ Date

Parker Bruni _____ Signature _____ Date

Abstract

This document provides an analysis of different technologies that could be used to satisfy different components of our web application. The purpose of this document is to compare and contrast different technologies in respect to our project's needs and goals and choose the best choice for implementation.

CONTENTS

1 Introduction

2 Visualization Frameworks

Our web application will provide near-real time data visualizations for energy consumption on campus buildings. This application will need to dynamically create charts and graphs based on energy data from the database. A key to choosing a visualization library will be to find one that can be dynmaically created and changed as new data is received from the data acquisition servers, and the ability to create chart templates that can be reused on multiple pages with different input parameters.

2.1 D3.js

Repository Commits: 4,104

Contributors: 120

Pros

- A lightwight, versatile javascript library that creates SVG elements within web pages and appends them to DOM elements.
- Makes use of javascript functions and DOM controlling functionality to dynamically change the content of the page.
- Provides a lot of variety and ability to customize graphics.
- Widely used and there is a lot of documentation and resources available to assist the learning and development processes.

Cons

- D3 is essentially an API to to manipulate SVG, it is not a charting library in of itself.
- You cannot easily pass a dataset into a specified chart type like other libraries.
- Considered to be “code-heavy” and difficult to jump right into as a novice user.
- Angular and D3 both attempt to control the DOM and so you have to find a way to make the two work together which is counterintuitive to both framework’s APIs.

2.2 Vis.js

Repository Commits: 3,165

Contributors: 137

Pros

- Easy to use and less of a learning curve than D3.
- Allows for interaction and minpulation of data on the chart.
- Able to handle large amounts of dynamic data.
- Really clean and nice looking graphics.

Cons

- Limited amount of possible chart types.
- Does not have built in heat map.

2.3 Chart.js

Repository Commits: 2,465

Contributors: 236

Pros

- Uses HTML5 canvas element.
- Allows for easy creating based on chart type specification.
- Library provides Line Charts, Bar Charts, Radar Charts, Pie Charts, Polar Area Charts, and Doughnut Charts.
- Very responsive charts based on screen width.
- Simple API, easy to use.

Cons

- Limited amount of possible chart types.
- Does not have built in heat map.

2.4 Conclusion

In conclusion, despite the steep learning curve associated with D3.js we think it will be the best option for our web application. It has the widest range of available graphs to accomodate all the client's requirments and desired visualizations. There are also a number of wrapper libraries available for D3.js like DC.js and dimple.js to help create charts from D3. This is a great way to get around the clunkiness and downsides to D3.js and reap the benefits of all the other charting libraries. Another benefit to using D3 is the extensive amount of templates, examples, and documentation that exists to help guide the process and implmentation of our application.

3 Password Hashing Algorithms

Our web application will have an authentication layer which will allow administrative users to have access to exclusive parts of the application. Anytime user information is stored into a database, it is important to hash the passwords and keep user credentials encrypted.

3.1 PBKDF2

Pros

- PBKDF2 allows for multiple iterations, and adding salted random input on any number of iterations [**?**].
- RSA standard.

Cons

- Unsafe because PBKDF2 can be thoroughly optimized with GPU [**?**]
- Complex API and slow computationally.

3.2 Bcrypt

Pros

- Can provide multiple hash iterations to strengthen the security.
- Very secure hash that can hash the same password multiple times.
- Widely used today and remains unbroken.
- Vetted by the entire crypto community as its now 15 years old [**?**].

Cons

- Slow and computationally expensive hashing.
- Only used for password hashing, not a key-derivation function.

3.3 Scrypt

Pros

- Can provide multiple hash iterations to strengthen the security.
- Newly developed based on focusing on the issues with BCrypt and PBKDF2 involving constant memory.

Cons

- New and not widely accepted by security/cryptographic professionals.
- Uses exponential time AND exponential memory which may be overkill for our application.

3.4 Conclusion

There is a lot of discussion and documentation involving password hashing algorithms and which ones are the "safest." For our web application, we will not be handling sensitive user data like credit card information or personal information, so we do not need the most comprehansive and computationally heavy hashing algorithm to encrypt our user passwords. For this, it would be best to use BCrypt because it is so widely used, has proven to be secure, and easy to implement. BCrypt has its own npm library and can be easily configure to run on a Node.js environment. It also provides secure, salted hashed that can be iterated over a configurable amount to increase the hashes effectiveness.

4 Front-end Framework

Our web application will be a series of dashboards to display energy data based on different buildings and subsets of buildings on Oregon State University's campus. A key part to designing a clean dashboard is having a well-spaced grid-like layout with different charts and graphs to display a multitude of different datasets and trends. Rather than customizing classic html elements and using div containers to space our dashboard, we wanted to look into bootstrapped dashboard templates that allow easy customization and clean-looking results.

4.1 CSS Bootstrap

Repository Commits: 17,255

Contributors: 953

Pros

- Extensive component list, responsive design, and built-in Javascript functions.[**?**]
- EFully responsive design.
- Huge developer contrbution and maintainence.
- Used by major companies like Lyft.com, Vogue.com, Vevo.com, and Newsweek.com. [**?**]

Cons

- Unsuitable for small scale projects. [**?**]
- Not good if you want to have large control over UI.

4.2 Pure CSS

Repository Commits: 541

Contributors: 51

Pros

- Meant for small project to get up and running quickly.
- Responsive design by default.
- Pure CSS is modular so you can download only the components you need.
- The complete module is very small so it is quick loading.
- Able to be used complimentary with other frameworks.

Cons

- Not as extensive component list as Bootstrap.

4.3 Foundation

Repository Commits: 15,094

Contributors: 959

Pros

- Responsive design by default.
- Easier to customize than bootstrap.
- CSS classes are built in. [**?**]
- More unique look than the more-popular bootstrap.
- Good grid implementaions with customizable grid layouts.

Cons

- Less maintained than bootstrap.
- Lack of support.
- Higher learning curve than bootstrap.

4.4 Conclusion

A lot of online resources acknowledged the fact that it is hard to make a website not look like bootstrap when using bootstrap css. Despite this, similar to our reasoning behind choosing D3.js, we would like to use a framework that is heavily supported and well-documented as it will be easier to answer questions during development. We think that the time spent restyling Bootstrap to make it look unique will not compare to the time saved by utilizing a well-documented framework. There is also a project called UI Bootstrap that works with AngularJS to create directives for each of the bootstrap components. [**?**]