November 21, 2017
CS461 Fall 2017
Group 57

Technology Review and Implementation Plan

Prepared by

Aubrey Thenell

Abstract

This document contains various analyses of technology to be used in the Scalable Web Application Framework for Monitoring Energy Usage on Campus. The technical components will be built of different frameworks. Each one of these frameworks needs to be selected from a large pool of frameworks to ensure the best fit framework is found. There will be an analysis for three different components of this project. A framework will then be selected based on technological advantages it provides.

CONTENTS

1 Introduction

My role in this project will primarily focus on the back-end side of the web application. This means I will oversee ensuring the database does exactly what it should be doing for any given query. This also includes any security measures to protect the database from common malicious attacks. Another thing I will oversee handling JSON requests from the front-end. This means I will have to ensure that the data expected by the user is returned by the server. Finally, a task I will oversee is to ensure the general structure of the server code as well as ensure computational logic is computed correctly.

Since our application for measuring energy usage on campus is web based; we will need various tools, technologies, and frameworks to assist us in storing and displaying information correctly. For this reason, it will be of the utmost importance to ensure that the right technology is selected for our project. We want to avoid technologies that are outdated, flawed, or otherwise counterintuitive for our needs. It is thus necessary to analyze multiple different technologies for each category to guarantee we are using the best technology available.

This document will discuss both the pros and cons of different technologies that will be required for various components of our application. More specifically, this document will compare technologies for three different components needed by our application. These components are: Structural Framework, Server-side Web Application Frameworks, and Web Hosting. For the Structural Framework, comparisons will be made between Angular.js, Backbone.js, and Ember.js. For the Server-side Web Application Framework, comparisons will be made between Express.js, Koa.js, and hapi.js. For the Web Hosting, comparisons will be made between Amazon's AWS, Oregon State University's Drupal through CWS, and Heroku.

2 Structural Frameworks

Structural frameworks control how the project directory and all related files/subdirectories will be stored. They aim to automate most of the overhead that will occur with most web based applications. Each framework will usually provide built in libraries to access databases, templating frameworks, and session management. Structural frameworks will run client side.

2.1 AngularJS

Pros

- AngularJS is extremely fast and easy to set up and run on server. By simply adding a few attributes to the HTML tags, it will perform all the heavy lifting to identify controller functions [1].
- It will then create dependencies between templates and the controller, which will then deliver the data directly to the screen [2].
- AngularJS is also extremely comprehensive. It provides a completely seamless option for front-end development.
- It does not require any other framework or plugin [1].
- It also features REST actions, which provides nearly instantaneous communication between clients and the server [2]. The REST actions also assist with retrieving interactable data within the webpage.
- It also features markup for DOM, which increases simplicity, integration with existing apps, and extensibility [2].

- Finally, AngularJS boasts a high performance. From all the features AngularJS has, it still yields extremely high speeds and fast response times [1].

Cons

- AngularJS mandates support of Javascript. Some computers and networks disable JavaScript by default. This means that the user would be unable to load, see, or access our page.
- AngularJS also relies heavily on the Model-View-Controller architectural patterns [1]. This means that we might have more difficulties since we aren't completely familiar with it.
- Since AngularJS is also quite layered and hierarchically arranged, it also might be difficult to debug scopes.

2.2 Backbone.js

Pros

- Backbone.js is a highly compact and extremely versatile.
- It mainly excels in smaller projects. This is because Backbone.js itself is a very minimalistic [3]. It is often referenced as a Javascript Library rather than an entire framework because of how small it is [3].
- It only contains the bare minimums needed to structure a web app in the Model-View-Controller pattern.

Cons

- Because of the barebones nature of Backbone.js, any non-basic features will have to be pulled in from separate libraries or plugins [3]. This will require a chunk of time for researching compatible libraries.
- Backbone.js also doesn't support two-way binding, which will often result in a lot of boilerplate for updating the model [3].

2.3 Ember.js

Pros

- Out of the three options, Ember is by far the most robust of the frameworks. It comes with a plethora of features.
- By itself, it will have everything needed to construct a web application, with some exceptions [3].
- This obviously would allow us to not have to worry about external libraries, dependencies, or plug-ins to get the application running [3].

Cons

- However, the high level of control comes at a cost. Since it's such a robust framework, there always might be a concern for performance/speed issues.
- Ember also has issues with adding any extra dependencies without breaking code [3].
- Ember is notorious for having extremely outdated examples and documentation [3]. Its official tutorials are also out of date, which raises the concern of where to find the answer when a particular issue comes up.

3 Server-side Web Application Framework

Server-side web application frameworks are, as the name implies, server side. This means everything in the framework will be executed server side, with minimal to no toll on the user's local performance. These frameworks must function in a stateless fashion. These frameworks will allow server side changes without having to refresh the page.

3.1 Express.js

Pros

- Express.js excels at being both lightweight and flexible. It is simple option for single-page, multi-page, and hybrid web applications [4].
- It is a powerful tool, despite being a micro-framework.
- Due to its simplicity, testing and quality assurance are streamlined.
- Its API takes advantage of the Node.js package manager to help install and distribute any needed plug-ins [4].

Cons

- Express.js doesn't have any recommended method of organization, which can lead to an unorganized codebase [4]. This can easily lead to non-reparability for some projects.
- Express.js is extremely manual labor-intensive when it comes to tasks. This can affect memory usage and performance if not addressed and handled professionally.

3.2 Koa.js

Pros

- Koa.js is very similar to Express.js. In fact, it was developed by the same team who created Express.js [5]. The developers created Koa.js to be a more concise and vivid alternative to Express.js.
- Koa.js is focused primarily on creating web applications and API's with improved and optimized performance.
- The main advantage it has over Express.js is its usage of the ES6 Generator feature [5]. This generator avoids callbacks all together, which increases the manageability of the code. It also assists in error handling and block building.

Cons

- The documentation for Koa.js is fairly lacking. This lack of documentation could prove problematic if we were to run into an obscure issue with the framework.
- Koa.js is also said to be an unstable framework [5]. Instability is something we cannot afford to have with our project since we will have a lot of data going at any point.

3.3 Hapi.js

Pros

- Hapi.js is a great choice because it comes with built-in support for input validation, caching, and authentication.
- There is an extensive list of third-party plugins available compatible with Hapi.js that will provide Object Relation Mapper and Object Document Mapper support [6].
- Object Document Mapper support will be important to our NoSQL themed database.
- Hapi.js also grants an extensive amount of control over request handling. It is also a plugin-based architecture that would allow us to pick and choose which modules we want [6]. This would give us more control over how large the application is scaled.

Cons

- Hapi.js doesn't come with a built-in code structure, which would require us to build our own [6].
- It also would require us to use only hapi.js-specific modules that would not be compatible with other frameworks, which could cause issues down the line.
- The main drawback to Hapi.js is that refactoring is all done manually, and endpoints are both created and tested manually [6].

4 Web Hosting

Web hosting is a server that will allow us to store our files on a remote server. This server will allow internet access to our files. This is needed, simply because we must store our files somewhere. With a web host, we can ensure our website is available to the public. Without it, our website would only be accessible on a local machine/network.

4.1 Amazon Web Services

Pros

- One of the main benefits of using AWS is the uptime. With the full power of Amazon, our application is almost guaranteed to always be accessible.
- AWS will also cover all our needs within the free tier that they offer [7].
- If/when our application needs to be scaled up, AWS offers scaled pricing based on how much data is used [7].

Cons

- AWS is said to have a difficult learning curve compared to other similar web hosts [7,8]. Furthermore, it is reported as user unfriendly.
- Another downside to AWS is the sheer amount of services it offers. This may sound like a pro at first, but most of these services have their own pricing and finding exactly what you need is not easy [8].
- Finally, the pricing of AWS is just confusing in general. With how many features it offers and a scaling cost, a consultant might be required to simply figure out how much you will need to pay [8].

4.2 Drupal

Pros

- Drupal would be a good choice because of how extensible it is.
- It's a blank slate that can defined exactly to user preference [10].
- Drupal is also excellent for building rapid prototypes [10]. This would allow us to attempt to implement multiple front-end designs/mockups with little time in between.

Cons

- However, to gain access to the "High flexibility" tier, there would be a large fee attached to it [9].
- In general, Drupal can have a large learning curve [10].
- It also requires a lot of effort to push any updates to the site [10]. We believe this would be problematic while we are designing and pushing major updates and features to our application.
- Drupal is also very resource intensive, and can drastically slow down the end user's speed down [10]. This is also something we can't afford to have since one of our main priorities is a highly reactive website.

4.3 Heroku

Pros

- Heroku is an extremely easy service to get started. An app can be deployed there in only a minute or two, by doing nothing more than a git remote and pushing to it [11].
- Heroku is also extremely extensible.
- The addons Heroku provide cover almost every need an application might need [11].
- Heroku is very flexible. Since it's UNIX based, you can essentially run anything you want/need to in their manage environment, then scale it appropriately [11].

Cons

- While Heroku offers a free tier, it is very limited. It can become extremely expensive as the application scales up to higher processing needs [11].
- The deploy time is also slow compared to other options. While the first deployment from git is fast, anything after that will be extremely slow [11].
- Everything on the server needs to recompile and restart, which results in downtime for the website/application [11].

5 Conclusion

For our structural framework, we have decided to use AngularJS. This is a middle ground between lightweight/flexible and robust/rigid. It also seems to offer the most features while also working well with our Node structure. The REST functionality will also be extremely useful for responsiveness. For our Server-side Web Application Framework, we decided to use Express.js. This is mainly due to the lack of documentation/support for Koa.js and the incompatibilities with hapi.js. We may end up using hapi.js as well for additional functionality. For web hosting options, we believe that AWS will be the best option. Its free tier is perfect for our project size. It also offers the best scaling, should our project grow to need it.

6 Works Cited

[1] M. Rajput. March 18, 2016. *The pros and cons of choosing AngularJS*. jaxenter. [Online].
Available https://jaxenter.com/the-pros-and-cons-of-choosing-angularjs-124850.html

[2] M. Rajput. Feb 24, 2016. *Top Reasons Why Web Developers Choose AngularJS*. c-sharpcorner. [Online].
Available http://www.c-sharpcorner.com/article/top-reasons-why-web-developers-choose-angularjs/

[3] Y. Gidalevitz. May 7, 2015 *Javascript MVC Frameworks: Your Choice is Your Future*. codal. [Online].
Available https://www.codal.com/javascript-mvc-frameworks-your-choice-is-your-future/#

[4] T. Volodymyr. Jun 8, 2017. *Express.js Mobile App Development: Pros and Cons for Developers*. jssolutionsdev. [Online].
Available https://jssolutionsdev.com/blog/express-mobile-app-development/

[5] N. Kharchenko. N/A. *How to Choose the Best Node.js Framework: Express.js vs Koa.js vs Sails.js*. Upwork. [Online].
Available https://www.upwork.com/hiring/for-clients/choose-the-best-node-js-framework-express-js-vs-koa-js-vs-sails-js/

[6] T. Sharma. Mar 13, 2017. *Hapi JS vs LoopBack for writing Enterprise Application*. medium. [Online].
Available https://medium.com/@tkssharma/hapi-js-vs-loopback-for-writing-enterprise-application-b86c86ccfb22

[7] N/A. N/A. *Amazon AWS*. Amazon. [Online]. Available https://aws.amazon.com/free

[8] R. Goodwin. Mar 10, 2017 *Amazon Website Hosting: Complete AWS Review (Updated 2017)*. howtostartablogonline. [Online].
Available https://howtostartablogonline.net/amazon-web-hosting-review/

[9] N/A. N/A. *Drupal*. Oregon State University. [Online].
Available http://is.oregonstate.edu/service/drupal

[10] N/A. N/A. *Drupal Pros and Cons: An In Depth Look*. mcdpartners. [Online].
Available https://www.mcdpartners.com/news/drupal-pros-and-cons/

[11] J. Symonds. N/A. *My Love/Hate Relationship With Heroku*. joshsymonds. [Online].
Available http://joshsymonds.com/blog/2012/06/03/my-love-slash-hate-relationship-with-heroku/