Many Voices Publishing Platform

Technology Review

D. Kevin McGrath & Dr. Kirsten Winters - CS461 Fall 2016

Commix

Steven Powers, Josh Matteson, Evan Tschuy

Abstract

The Many Voices Publishing Platform uses a variety of technologies to handle different aspects of the project, from the user interface to the backend database operations. These technologies enable to the Many Voices Publishing Platform to succeed in delivering a working platform for textbook collaboration.

Contents

1 Technology Review

1.1 Introduction

The Many Voices Publishing Platform is being developed for the purpose of fixing the problems currently associated with the textbook market. We will accomplish this by giving the MVP Platform an easy to use interface, a search bar with a built in results pane, source control, and many other features. Authorship is divided by subsection header.

1.2 Steven

1.2.1 User Interface

Option 1 - React
React is a JavaScript rendering engine that is developed by Facebook. Originally used with Instagram, React is often paired with Redux for added functionality. React is a popular JavaScript library meant for building user interfaces that is component based.

Option 2 - Aurelia
Aurelia is a newer JavaScript client framework for mobile, desktop, and the web, by using simplistic integration.

Option 3 - Ember
Ember uses web components and templates to increase productivity.

Option 4 - Angular2
Angular2 is a project started by Google for their internal Green Tea project. Angular2 is a widely documented JavaScript cross-platform library that is used to create native mobile and desktop web applications.

Goals
The use of this technology will aid in the development of the user interface. Having a beautiful and scalable user interface will help users interact with the platform more easily, on whatever device they choose to use it on.

Evaluation Criteria

The options are evaluated on

- Ease of Use
- File Size
- Features
- Performance
- Standards Compliance
- Non-Compliance
- Release
- License

Option Comparison

[1]	React [2]	Aurelia [3]	Ember [4]	Angular 2 [5]
Ease of Use	Substantial setup required for working system, lots of documentation and tutorials.	Simple setup using NPM and installation	Simple setup using NPM and installation	Substantial setup required for working system, lots of documentation and tutorials.
File Size	156kb to ???kb, due to added frameworks	323kb	435kb	1023kb
Features	View rendering engine with plugin frameworks	Router, Animation, HTTP Client	Router, HTTP Client	Router, HTTP Client
Performance (Paints per Second)	45-50	90-150 (Higher end with additional plugins)	60-100	80-130 (Higher end with additional plugins)
Standards Compliance	ES 2015	HTML, ES 2016, Web Components	HTML, ES 2015	ES 2016
Non-Compliance	JSX	N/A	N/A	NG2 Markup, Dart
Release	15.x	Beta	2.x	Release Candidate
License	BSD	MIT	MIT	MIT

Discussion

All of the chosen options have their pros and cons for our web application. All of them however would be a learning and research experience. Angular2 and React have the benefit of being created by large software companies, Google and Facebook respectively. This means that there will be large adoption and documentation / tutorials available. Aurelia and Ember seem to be easier to implement however, they are much newer products and they have a smaller adoption population. This could prove troublesome if we run into problems. If our implementation ends up being a fork of Ward Cunningham's Federated Wiki, then this decision will be null most likely.

Selection

Initially we were set on using Angular2 as part of the team has experience using this JavaScript library, before meeting with our client. Angular2 has a wide adoption and is used by Google for internal projects so the longevity of the framework is expected to last. With this in mind, we plan to use Angular2 if we need to use a JavaScript framework for our user interface.

1.2.2 Documentation

1.2.3 Inclusive Design

1.3 Josh

1.3.1 Testing

Option 1 - Mocha

Mocha is a JavaScript testing framework, loaded with features. It runs on Node.js and also in the browser, making asynchronous testing simple and easy to use. Mocha tests run serially, allowing for flexible and accurate reporting, while mapping uncaught exceptions to the correct test cases. [6]

Option 2 - QUnit

QUnit is a powerful, easy-to-use JavaScript unit testing framework. It's used by the jQuery, jQuery UI and jQuery Mobile projects and is capable of testing any generic JavaScript code. [7]

Option 3 - Jasmine

Jasmine is a behavior-driven development framework for testing JavaScript code. It does not depend on any other JavaScript frameworks. It does not require a DOM. And it has a clean, obvious syntax so that you can easily write tests. [8]

Goals

Using this technology will aid in proper functionality and minimize errors. Without properly testing code, a number of problems can occur that can disrupt and slow down progress in a team. In extreme cases, not properly testing could lead to failure of the application.

Evaluation Criteria

The options are evaluated on

- Ease of Use
- File Size
- Features
- Performance
- Standards Compliance
- Non-Compliance
- Release
- License

Option Comparison

	Mocha	QUnit	Jasmine

Discussion

It is well known that an application that ensures proper functionality supersedes other applications and, more importantly, competitors. Quality assurance is acknowledged as highly distinguished, and therefore, an attribute deserving of notable consideration. Having this in mind, we will be considering three different testing frameworks: Mocha, QUnit, and Jasmine.

Mocha (more info to come)

QUnit (more info to come)

The Jasmine testing framework is the most common of the three, and considerably so. (more info to come)

Selection

Mocha?

1.3.2 User Authentication

1.3.3 Database

1.4 Evan

1.4.1 Technology 1

1.4.2 Technology 2

1.4.3 Technology 3

References

[1] R. Eisenberg, "Choosing a javascript framework," https://www.youtube.com/watch?v=6I_GwgoGm1w.

[2] Facebook, "A javascript library for building user interfaces - react," https://facebook.github.io/react/index.html.

[3] Aurelia, "Aurelia," http://aurelia.io/.

[4] Ember, "A framework for creating ambitious web applications," http://emberjs.com/.

[5] Google, "Our framework," http://angular.io/.

[6] MochaJs, "Mocha, simple, flexible, fun," https://mochajs.org/.

[7] QUnit, "Qunit: A javascript unit testing framework," https://qunitjs.com/.

[8] Jasmine, "Jasmine," https://jasmine.github.io/2.0/introduction.html.