SOFA Research

Angular vs React

Loek Ehren

January 9, 2018

Contents

Chapter 1

Introduction

For our SOFA project we have to build a frontend part which interacts with a backend. Several choices of technology are available to use in making a frontend but in the end it came down to two: Angular or React. To make a proper educated choice between the two, the pros and cons of both will need to be listed and compared to eachother, as well as be related to the business needs of the project and the customer.
In this document Angular and React and their characteristics will be described and related to the requirements of the project. After that a conclusion will be drawn and a recommendation will be given on which technology to use.

Chapter 2

Requirements

Several requirements have been identified that will help choose what frontend technology to use. These will be listed here:

1. The learning curve should not be too high, since learning a new technology will also take a lot of time.
2. The technology should be able to be testable easily.
3. The technology should be able to provide a good way of having dynamic pages with lots of changing data.
4. A project using this technology should be able to be set up quickly, and be able to be extended quickly.
5. Performance is not that important.

Chapter 3

Angular

Angular, formerly called AngularJS, is a JavaScript based open source framework to develop frontend applications. It's created and maintained by Google. As a framework, Angular offers developers tools to make HTTP requests, enable page routing and do component testing among others. It's goal is to enable developing of frontends using the MVC (Model, View, Controller) pattern to separate presentation from data from logic.

Angular versions above 2 use TypeScript, which is a more enhanced language of JavaScript which enables static typing, but largely is the same as JavaScript.

Angular uses templates to render pages using components and directives. These directives come from Angular's own defined attributes on HTML tags and can manipulate data in some way. Such as 'ng-for' which enables you to do a for loop and iterate over some data, maybe to create a row of table cells or other elements in a page.

3.1 Pros and Cons

In this section and the next the characteristics of Angular will be tested against the requirements defined in chapter 2.

The learning curve should not be too high, since learning a new technology will also take a lot of time.
The learning curve will be pretty high, since Angular will require learning a whole new framework from data binding to testing, becoming familiar with using the MVC pattern as well as learning TypeScript. Angular is a huge library with a lot to learn. What is an advantage though is that every essential thing such as routing and HTTP requests is built in and plenty of documentation is available to read.
However this can also be a negative since there is so much documentation, which might not necessarily be well written, which will inevitably raise even more questions and slow down development time.

The technology should be able to be testable easily.
Along with its built in unit testing functionality, third party libraries such as Enzyme can also be used to extend and improve tests.

The technology should be able to provide a good way of having dynamic pages with lots of changing data.
Angular has one-way data binding which means the UI can only be updated from the model it represents. If something has to change in the UI, the model has to be updated first, and the UI will follow.
But Angular also enables two-way data binding which means that data in the UI and model can be updated from both sources.
Building a one way data binding can be a lot of work, since event handlers will be required to handle for instance a user typing something in the input field, and updating that value in the model. With two-way data binding the input field can update the model automatically, so time is saved implementing such a trivial feature. This proves a significant advantage over React as React enforces one way data binding.

There are counterarguments to this though. A rule in programming says that explicit is better than implicit, and that doing it in the way of React is better than cleaner than the under-the-hood implementation of Angular's two-way binding. It can also slow down your program as tons of data handling functions will be called once something changes in either the model or the view. And conflicts can arise due to that as well. Creating complex, large and rich UIs can then quickly turn into spaghetti code that slows down and where you will have no idea what is going on, as everything is abstracted behind Angular bindings and directives.

A project using this technology should be able to be set up quickly, and be able to be extended quickly.
Angular also offers a CLI (Command Line Interface) that provides a ton of functionality to quickly setup projects, run tests and builds, and add new components with a single command. It can also create a scaffold of whatever new component you need, along with its stylesheets, associated test files, and any other required files.

Chapter 4

React

React is a JavaScript library for frontend development. It is not a framework such as Angular which gives a whole MVC architecture to you, but essentially is only the View layer of MVC, allowing you to fill in the gaps yourself with other technology. It is developed and maintained by Facebook.

The thing that sets React apart from Angular is its simplicity and it being lightweight. It also uses JSX which is a syntax that combines HTML and JavaScript to provide databinding and manipulation. In essence a component is now one JSX file in React, instead of a separate HTML and TypeScript file in Angular.

4.1 Pros and Cons

In this section and the next the characteristics of React will be tested against the requirements defined in chapter 2.

The learning curve should not be too high, since learning a new technology will also take a lot of time.
Because React only defines the View part of a traditional MVC pattern the learning curve will be easier for newcomers. Users can define their own models and controllers in JavaScript and make them as simple as they want, giving users a lot of freedom. Users can also use different libraries or frameworks to do things like HTTP requests, dependency injection and data flow. The syntax for React is also fairly simple. It uses JSX which looks like a mixture between HTML and JavaScript. It does not use a lot of abstractions and creating components is easily done in a single file. React can be expanded upon by adding libraries such as React-router for routing, Flow for type checking and Redux for more complex data handling. This has the added risk of some libraries and frameworks being more complex than expected.

The technology should be able to be testable easily.
React claims to be more testable than Angular. As with Angular third party testing libraries can be used.

The technology should be able to provide a good way of having dynamic pages with lots of changing data.
As previously explained React enforces one-way data binding. There can be added complexity as React does not define controllers or any fixed way to transmit data between components. Libraries like Redux can be used to create state containers and implement more complex state changing.

A project using this technology should be able to be set up quickly, and be able to be extended quickly.
Like Angular, there is a command line tool for React to quickly setup projects. 'Create-react-app' is that tool. It offers a modern setup with no configuration. All the build options are abstracted and cannot be seen unless you explicitly remove the abstraction with a simple command. A package can be added called react-cli that allows the functionality to add more components similar to angular-cli.

Chapter 5

Conclusion

In conclusion, Angular offers a complete package with a huge amount of functionality and included packages. It has a steep learning curve but once mastered there is little need to add anything extra to the core MVC functionality.

React on the other hand has a more minimalistic approach, both in code complexity and functionality. This allows for a more modular and simple approach to coding. But will require learning extra libraries to enable routing and complex state management. This has the added risk of some libraries and frameworks being more complex than expected.

Taking the requirements defined in Chapter 2 into account, the recommended approach would be to invest time into learning the Angular framework and use this for the frontend part of the project.

Chapter 6

References

https://www.accelebrate.com/blog/two-way-data-binding-angular-2-and-react/
https://www.upwork.com/hiring/development/angularjs-vs-react/
https://react-etc.net/entry/react-gets-official-boilerplate-scaffolding-through-react-cli
https://medium.com/unicorn-supplies/angular-vs-react-vs-vue-a-2017-comparison-c5c52d620176
https://medium.com/@mnemon1ck/why-you-should-not-use-angularjs-1df5ddf6fc99
https://reactjs.org/blog/2016/07/22/create-apps-with-no-configuration.html
https://medium.com/@chriscordle/why-angular-2-4-is-too-little-too-late-ea86d7fa0bae
https://www.sitepoint.com/react-vs-angular/
https://hackernoon.com/angular-vs-react-the-deal-breaker-7d76c04496bc
https://reactjs.org/docs/introducing-jsx.html