M Ű E G Y E T E M 1 7 8 2

**Budapest University of Technology and Economics**
Faculty of Electrical Engineering and Informatics
Department of Telecommunications and Media Informatics

# Designing and Implementing an Educational Support System

Bachelor's Thesis

*Author*
Nóra Szepes

*Advisors*
Dr. Sándor Gajdos
Bence Golda

2015

# Contents

# HALLGATÓI NYILATKOZAT

Budapest, 2015. október 18.

*Szepes Nóra*
hallgató

# Kivonat

# Abstract

# Chapter 1

# Introduction

# Chapter 2

# Comparing JavaScript Frameworks

For the project I wanted to choose a JavaScript framework for faster developement than using plain JavaScript with jQuery. I chose the TodoMVC [11] website to find the most popular frameworks.

I tried these frameworks to see how fast and easily can I build a basic website, how can I access a server with AJAX requests and how routing and data binding works.

In JavaScript with AJAX requests we can send requests to a server asynchronously without reloading a page. In a single-page application we want to make the browser think it is always on the same page. When the user clicks on a new link, the browser won't reload the whole page, it will just simply load the new view into the old frame. Everything happens in the background so the application won't force the user to wait while it sends data to a server. If the application is retrieving data, then when it arrives, the application can process it and show the result to the user.

There are two types of routings. Routing can be either a way to manipulate browser's URL and the part of a web application what decides which controller will handle the requests. I was looking for a solution for URL manipulation.

The classic data binding model is when the view template and the data from the model are merged together to create the to be displayed view. Any data changes in the view won't automatically sync into the model. The developer has to write the controller what syncs the changes between the model and the view. [3]

## 2.1 React

My first choice was React [9]. It is developed by Facebook and Instagram since 2013. With the help of the official website tutorials and videos I've managed to create websites easily with JSX [4] because I can use XML-like syntax for creating views.

After reading the documentation I found out that it gave me no other option than inline jQuery for AJAX requests and routing is not part of the main core although there are many good routing libraries, e.g. React/Router [10].

## 2.2 AngularJS

AngularJS [1] is one of the most famous JavaScript frameworks nowadays. It is maintained by Google for 6 years now. It focuses mostly on dynamic views in web-applications. On the homepage they provide many examples in written tutorials and videos on YouTube.

Creating a website is done with an extended HTML vocabulary, like Android Layouts where you declare everything in XML. Building a website wasn't that hard but the data binding is different. It uses a two-way data binding template [3] which means whenever either the View or the Model is changed, it will update the other one.

Angular AJAX requests with the shortcut methods are similar to the AJAX methods in jQuery. It can also be used in unit tests with ngMock [2].

In a single-page application we want to make the browser think it is always on the same page. When the user clicks on a new link, the browser won't reload the whole page, it will just simply load the new view into the old frame. Angular binds a special listener to links. If the user clicks on the link, Angular will simply push the page to the history and replace the view with the new page. This method only works if Angular is loaded into memory otherwise the application needs a page where Angular can be loaded.

## 2.3 Mithril

My last choice was Mithril [5]. It is really similar to React. With the help of MSX [8], you can use the same HTML-like syntax to create websites. MSX uses JSX, but transforms the output to be compatible with Mithril.

It provides a simple way to work with AJAX requests [6]. A basic request returns a special *m.prop* getter-setter, what is a utility class.

Mithril provides utilites to handle routing [7]. It's organized around URL routes. For a URL we can create a new module with a controller and a view. When the user clicks on the link a new instance will be made and passed to the view.

## 2.4 Conclusion

React is basically only for building views and I should use it combined with other frameworks to have full MVC experience. I really liked the fast way of creating UIs but I was looking for a framework that supports everything what I need.

Although Angular contains everything what I need, I didn't like the extended HTML vocabulary. I liked the fast and easy way of creating websites in React. Mithril provides a similar way of doing it.

In this project I'll use Mithril. It is a simpler framework than Angular, because it doesn't implement features what make it feel like a new programming language and not a JavaScript framework.

# Chapter 3

# specification

# Chapter 4

# conceptual

# Chapter 5

# design

# Chapter 6

# implementation

# Chapter 7

# test

# Chapter 8

# deployment

# Chapter 9

# conclusion

# Acknowledgements

# List of Figures

# List of Tables

# Bibliography

[1] AngularJS. https://www.angularjs.org/.

[2] AngularJS, API Reference, http. https://docs.angularjs.org/api/ng/service/$http.

[3] AngularJS, Developer Guide, Data Binding. https://docs.angularjs.org/guide/databinding. Accessed: 2015-10-18.

[4] JSX Specification. https://facebook.github.io/jsx/.

[5] Mithril. http://mithril.js.org/.

[6] Mithril, m.request. http://mithril.js.org/mithril.request.html.

[7] Mithril, routing. http://mithril.js.org/routing.html.

[8] MSX. https://github.com/insin/msx.

[9] React. https://facebook.github.io/react/.

[10] React/router. https://github.com/rackt/react-router.

[11] TodoMVC. http://todomvc.com/. Accessed: 2015-10-18.

# Appendix