Angular.js vs Vue.js

Cornelia Langsenlehner, Sandra Hartlauer, Michael Nußbaumer

BACHELORARBEIT

eingereicht am

Fachhochschul-Bachelorstudiengang

Mobile Computing

in Hagenberg

im Mai 2024

Advisor:

FH-Prof. Dr-Ing. Krösche Jens

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere. This printed copy is identical to the submitted electronic version.

Hagenberg, May 22, 2024

Cornelia Langsenlehner, Sandra Hartlauer, Michael Nußbaumer

Contents

Preface

This is version **2023/11/06** of the LaTeX document template for various theses at the School of Informatics, Communication and Media at the University of Applied Sciences Upper Austria in Hagenberg. We are pleased to learn that this document collection is meanwhile also used at various other institutions in Austria and abroad.

The document was initially created in response to requests from students after the 2000/01 academic year when an official LaTeX introductory course was offered in Hagenberg for the first time. The fundamental idea was to "simply" convert the already existing *Microsoft Word* template for diploma theses to LaTeX and possibly to add some unique features. This quickly turned out to be not very useful since LaTeX, especially concerning the handling of literature and graphics, requires a substantially different way of working. The result is— rewritten from scratch and much more extensive than the previous document—a manual for writing with LaTeX, supplemented with additional (meanwhile removed) hints for *Word* users. Technical details of the current version can be found in Appendix.

While this document was initially intended exclusively for the preparation of diploma theses, it now also covers *master theses*, *bachelor theses*, and *internship reports*. The differences between these documents have been deliberately kept small.

When creating this template, an attempt was made to work with the basic functionality of LaTeX and—as far as possible—to achieve this without additional packages. This was only partially successful; several supplementary "packages" are necessary, but only common extensions have been used. Of course, there is a large number of additional packages which can be helpful for further improvements and refinements. Everyone is encouraged to experiment with these as soon as they have the necessary self-confidence and sufficient time to experiment. Many details and tricks are not explicitly mentioned in this document but can be explored in the underlying source code at any time.

Numerous colleagues have provided valuable support through careful proofreading and constructive suggestions for improvement. We thank Heinz Dobler for consistently improving our "computer slang" and Elisabeth Mitterbauer for her proven "orthographic eye".

Usage of this template is free without any restrictions and not bound to any mention. However, when used as a basis for one's work, one should not simply start working on it, but at least *read* the essential parts of the document and, if possible, take them to heart. Experience has shown that this improves the quality of the results significantly.

This document and the associated LaTeX classes have been available since November

2017 on CTAN[1] as package `hagenberg-thesis`,

https://ctan.org/pkg/hagenberg-thesis.

The current source code, as well as additional materials—such as a wiki with instructions for the integration of often requested functionalities and extensions—can be found at

https://github.com/Digital-Media/HagenbergThesis.[2]

Despite great efforts, a document like this always contains errors and shortcomings. Comments, suggestions, and helpful additions are welcome. Ideally, as comments or issues on GitHub.

By the way, here, in the preface (which is common in diploma and master theses but dispensable for bachelor's theses), you may briefly describe the genesis of the document. This is also the place for any acknowledgments (e.g., to the supervisor, the examiner, the family, the dog, etc.) as well as dedications and philosophical remarks. These should be balanced and limited to a maximum of two pages.

W. Burger (em.) and W. Hochleitner

University of Applied Sciences Upper Austria
Department of Digital Media, Hagenberg
https://www.fh-ooe.at/campus-hagenberg/

[1]Comprehensive TeX Archive Network

[2]https://github.com/Digital-Media/HagenbergThesis/blob/main/CHANGELOG.md contains a list of chronological changes (formerly included in the appendix of this document).

Abstract

Here goes an abstract of the work, with a maximum of 1 page. Unlike other chapters, the abstract is usually not divided into sections and subsections. Footnotes are also not used here.

By the way, abstracts are often included in literature databases with the author and title of the work. It is, therefore, essential to ensure that the information in the abstract is coherent and complete in itself (i.e., without other parts of the work). In particular, *no literature references* are typically used at this point (as is the case also in the *title* of the thesis and the German *Kurzfassung*)! If such is needed—for example, because the paper is a further development of a particular, earlier publication—then *full* references are necessary for the abstract itself, e.g., [ZOBEL J.: *Writing for Computer Science – The Art of Effective Communication.* Springer, Singapore, 1997].

It should also be noted that special characters or list items are usually lost when records are added to a database. The same applies, of course, to the German *Kurzfassung.*

In terms of content, the abstract should not be a list of the individual chapters (the introduction chapter is intended for this purpose). However, it should provide the reader with a concise summary of your thesis. Therefore, the structure used here is necessarily different from that used in the introduction.

Kurzfassung

Dies sollte eine maximal 1-seitige Zusammenfassung Ihrer Arbeit in deutscher Sprache sein.

The German “Kurzfassung” should contain the same content as the English abstract. Therefore, try to translate the abstract precisely but not word for word. When translating, remember that certain idioms from English have no counterpart in German or must be formulated differently. Also, word order in German is very different from English. Without knowledge of the German language, it is acceptable to resort to translators. Nevertheless, hiring a skillful person for proofreading is recommended even with the highest confidence in one’s German knowledge.

The correct translation for “diploma thesis” is *Diplomarbeit*, a “master thesis” is called *Masterarbeit*. For “bachelor’s thesis”, *Bachelorarbeit* is the appropriate translation.

By the way, for this section, the *language setting* in LaTeX should be switched from English to German to get the correct form of hyphenation. However, the correct quotation marks must be set manually.

Chapter 1

Introduction - C

1.1 WebApps and JavaScript Frameworks

Web apps are fundamentally important in today's digital world as they provide a flexible and efficient way to access content and services across various platforms. The platform independence of web apps allows developers to reach a broad user base without the need to create separate applications for each operating system or device. By implementing and comparing an application developed with both the Vue and Angular frameworks, relevant metrics such as load times, performance, and developer-friendliness can be effectively evaluated. This analysis aims to provide a comprehensive overview of both frameworks and serve as a decision-making guide, offering valuable insights into the strengths and weaknesses of both technologies. These insights aid development teams in making informed decisions about the most suitable technologies for specific application requirements, and facilitate the identification of optimization opportunities to continuously improve the end-user experience and enhance development efficiency.

Chapter 2

Theoretical background - S

2.1 What are WebFrameworks

A framework is a program skeleton that serves as a foundation for software development. Frameworks already contain numerous functions in software development as proposed solutions for individual problems faced by various developers. This way, they do not have to start from scratch when they want to write new software.

A framework describes a collection of interacting classes and thus also fixes the design structure for software developed based on the framework.

If a framework serves as the foundation for web applications, it is referred to as a web application framework.

2.2 WebFrameworks - Pros and Cons

Web frameworks offer many advantages, but they also hide significant disadvantages.

2.2.1 Advantages

The use of web frameworks aims to reduce time and cost in software development. The focus is on reusing code for basic functions such as database connections, templates, caching, and security, which are provided as pre-built modules.

This allows development to concentrate on the specific code of the new application. Since most web frameworks are available as open-source, there are generally no licensing costs.

Frameworks also promote the creation of clean and maintainable code, as developers can rely on proven building blocks. These are regularly improved by the community, and security vulnerabilities are quickly addressed.

2.2.2 Disadvantages

On the internet, there are numerous frameworks available for web development, which differ in their design principles and functionality. Depending on the project, a specific framework may be required, which can involve compromises.

Although frameworks are intended as general solutions, developers often do not use all the available functions, leading to unnecessary code, known as bloat.

Another disadvantage is the dependency on the framework and its provider, as well as potential licensing restrictions. Problems can also arise if the development of the framework is discontinued. Developers must also familiarize themselves with the structure and use of the framework, which requires time, although this is compensated by pre-built functions and code modules.

Since the source code of many web frameworks is publicly accessible, anyone can view it. This can lead to security risks, especially if enterprise applications are based on public code.

2.3 Introduction to JavaScript Frameworks

JavaScript is a straightforward programming language particularly suited for working in web browsers. However, the interface to the website, the DOM (Document Object Model), can pose a challenge for many programmers. This is where JavaScript frameworks and libraries come into play: they provide developers with tools to simplify this and other aspects of programming.

JavaScript frameworks are especially suitable for developing complex web applications. When developers become familiar with the concepts and guidelines of a particular framework, they can work very effectively with it.

2.4 Short description of Angular.js

AngularJS, developed by Google, is a JavaScript framework for web development that emphasizes structure and quality. It was the first framework suitable for large enterprise applications, addressing architecture, testability, and isolated components. Techniques like dependency injection enable efficient and maintainable software development based on JavaScript.

Angular is an evolution of AngularJS, with a completely rewritten codebase now using TypeScript as its foundation. This allows for migration or even hybrid use of the versions. The project has evolved from the development of a pure framework to a comprehensive platform for web applications.

2.5 Short description of Vue.js

Vue.js (pronounced /vju/, like "view") is a JavaScript framework for designing user interfaces. It is based on standard HTML, CSS, and JavaScript and offers a declarative, component-based programming model that allows for the efficient development of user interfaces of any complexity.

2.6 What are the differences between Angular.js and Vue.js

Angular and Vue.js are both JavaScript frameworks that can be used to develop modern web applications. A key difference is that Angular uses an MVC (Model-View-

Controller) architecture, while Vue.js is a more progressive framework. Angular offers a robust and well-established development environment with features like efficient two-way data binding, a comprehensive CLI, and a well-defined architecture. Vue.js, on the other hand, is more flexible and lightweight, with features like the virtual DOM, CSS transitions, and simpler computed properties. Both frameworks have their advantages and disadvantages, and the choice between them depends on the specific requirements and preferences of a project.

Chapter 3

Comparison criteria - M

3.1 Performance

3.2 Developer-friendliness

3.3 Community support

3.4 Ecosystem

Chapter 4

Case Study - C

4.1 Case Study

In this academic paper, a detailed case study of “ProTrack,” an innovative web application engineered for tracking, sharing, managing, and collaboratively editing project work is presented. ProTrack is an invaluable resource for project management teams, providing tools that enhance administrative efficiency and promote robust teamwork and collaboration.

ProTrack has been developed in two separate versions, one using Angular.js and the other with Vue.js. This dual-framework approach allows for a comprehensive comparison, providing clear insights into the unique benefits and potential limitations that each framework offers when applied to the same functional parameters.

The case study section delves into several critical areas of ProTrack:

- **Functionality**: The paper thoroughly explores ProTrack’s essential features, including project tracking, document sharing, task management, and collaborative capabilities. It examines how each framework supports these features and assesses their impact on the user experience.
- **Development Experience**: The section offers insights from the development process with Angular.js and Vue.js, focusing on tooling support and the ease of integration with other technologies.
- **Performance Metrics**: Empirical data regarding performance metrics such as load times, runtime efficiency, and resource utilization for each framework are presented. This quantitative analysis enhances the qualitative evaluations provided throughout the case study.

Through this examination, the case study section aims to provide a detailed assessment of both frameworks as applied to a real-world application, thereby offering essential insights that assist developers and researchers in selecting between Angular.js and Vue.js for similar projects.

Case Study 1: Angular.js Application

- **Functionality**:
- **Development Experience**:
- **Performance Metrics**:

Case Study 2: Vue.js Application

- **Functionality**:
- **Development Experience**:
- **Performance Metrics**:

Chapter 5

Results - S

5.1 Performance

5.2 Developer-friendliness

5.3 Community support

5.4 Ecosystem

Chapter 6

Conclusion and recommendations - M

6.1 Conclusion

6.2 Recommendations

References