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 - M

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.

1.2 Motivation

1.3 Challenges in Web Development

1.4 Goals

1.5 Structure

Chapter 2

Theoretical Background - S

2.1 Web Frameworks

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 whenever 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. When a framework serves as the foundation for web applications, it is referred to as a web application framework [2].

2.1.1 Pros and Cons of Web Frameworks

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

Advantages:

- **Reduced Development Time and Cost:** By reusing code for basic functions such as database connections, templates, caching, and security, development time is significantly reduced.
- **Open Source:** Most web frameworks are available as open-source, eliminating licensing costs.
- **Clean and Maintainable Code:** Frameworks promote the creation of clean and maintainable code, as developers can rely on proven building blocks.
- **Community Support:** Regular improvements and quick fixes for security vulnerabilities are provided by the community [2].

Disadvantages:

- **Framework-Specific Requirements:** Different frameworks have different design principles, potentially requiring compromises based on the project.
- **Code Bloat:** Developers may not use all available functions, leading to unnecessary code.

- **Dependency Risks:** Reliance on a framework and its provider can be risky if the framework is discontinued or its development slows.
- **Learning Curve:** Familiarizing oneself with a framework's structure and use takes time.
- **Security Risks:** Publicly accessible source code can pose security risks for enterprise applications [2].

2.2 Introduction to JavaScript Frameworks

JavaScript is a versatile programming language particularly suited for work in web browsers. Originally developed by Netscape, it has become one of the most widely used programming languages on the web, enabling developers to create dynamic and interactive content by manipulating the Document Object Model (DOM).

Direct manipulation of the DOM can be challenging, which is where JavaScript frameworks and libraries come into play. They provide tools to simplify these and other aspects of programming, making JavaScript frameworks especially suitable for developing complex web applications. These frameworks offer a structured approach and prebuilt components, enabling more efficient development once the concepts and guidelines of a particular framework are mastered [1].

2.3 DOM and Dynamic Web Content

The Document Object Model (DOM) is a programming interface for web documents. It represents the page so that programs can change the document structure, style, and content. The DOM is crucial for creating dynamic web content, which is essential for modern web applications that need to be interactive and responsive [4].

2.3.1 Benefits of Dynamic Web Content

1. **Personalization**: Tailoring content based on user preferences, demographics, or behavior can significantly boost engagement and conversion rates. Personalized experiences are crucial for retaining visitors and improving user satisfaction.
2. **Real-time Updates**: Providing timely information without refreshing the entire page is crucial for news websites, social media platforms, and e-commerce sites. JavaScript enables real-time updates through asynchronous requests and data manipulation.
3. **Enhanced User Engagement**: Interactive features such as live chat, dynamic forms, and multimedia sliders captivate users, extending session duration and reducing bounce rates.

2.3.2 Implementing Dynamic Web Content

To effectively implement dynamic web content using JavaScript and jQuery:

1. **Define Goals**: Clearly outline objectives, whether to enhance user experience, increase conversions, or improve engagement metrics.

2. **Utilize User Data**: Analyze user behavior with tools like Google Analytics to personalize content and tailor dynamic elements based on preferences and interactions.
3. **Choose Technologies**: Select appropriate JavaScript frameworks or libraries (e.g., React, Vue.js) and jQuery plugins to streamline development and ensure cross-browser compatibility.
4. **Test and Iterate**: Continuously test dynamic features, gather user feedback, and refine implementations to optimize performance and user satisfaction.

By leveraging JavaScript and jQuery for dynamic web content, websites can offer personalized, engaging experiences that cater to modern user expectations [6].

2.4 Comparison criteria - M

2.4.1 Metrics in General

2.4.2 Performance

2.4.3 Developer-friendliness

2.4.4 Community support

2.4.5 Ecosystem

2.5 JavaScript for Complex Web Applications

JavaScript is well-suited for complex web applications due to its versatility and the powerful features of modern JavaScript frameworks. These frameworks provide a structured environment for development, making it easier to manage large codebases and maintain code quality.

Alternatives to JavaScript:

- **TypeScript:** A superset of JavaScript that adds static types, making it easier to catch errors early.
- **Dart:** Developed by Google, Dart is used in conjunction with the Flutter framework for building web and mobile apps.
- **Elm:** A functional language that compiles to JavaScript, focused on simplicity and quality tooling [7].

2.6 Relevant JavaScript Frameworks

Several JavaScript frameworks are popular for developing web applications, each with its unique features and strengths. This section provides a brief overview of two of the most relevant frameworks, Vue.js and Angular, and explains why they are examined in detail.

2.6.1 Angular

Angular is a platform and framework for building single-page client applications using HTML and TypeScript. Developed by Google, Angular provides a comprehensive solution for complex and large-scale applications [8].

Key Features:

- Two-way data binding
- Dependency Injection
- Comprehensive CLI
- TypeScript support

2.6.2 Vue.js

Vue.js is a progressive JavaScript framework for building user interfaces. It is designed to be incrementally adoptable and can function as a library or a full-fledged framework. Vue.js is known for its simplicity and flexibility, making it suitable for both small projects and large applications [5].

Key Features:

- Virtual DOM
- Reactive data binding
- Component-based architecture
- Easy integration with other projects and libraries

2.7 Vue.js vs. Angular: A Detailed Comparison

In this section, we will compare Vue.js and Angular based on several parameters to provide a clear understanding of their respective strengths and weaknesses.

2.7.1 Performance

Vue.js:

- Uses a virtual DOM for efficient updates
- Generally faster in rendering due to its lightweight nature

Angular:

- Real DOM, which can be slower in certain scenarios
- Optimized for performance with techniques like ahead-of-time (AOT) compilation

Comparison:

- Vue.js often has better performance for simple to moderately complex applications.
- Angular provides robust performance optimizations suitable for enterprise-level applications.

2.7.2 Developer-Friendliness

Vue.js:

- Easy to learn and integrate
- Simple syntax and flexible structure
- Comprehensive documentation

Angular:

- Steeper learning curve due to its complexity
- Requires understanding of TypeScript
- Extensive documentation and resources

Comparison:

- Vue.js is generally more accessible for beginners.
- Angular is more suitable for developers with experience in TypeScript and complex frameworks.

2.7.3 Community Support

Vue.js:

- Growing community with increasing adoption
- Numerous plugins and integrations
- Active development and support

Angular:

- Large and established community
- Extensive resources and third-party libraries
- Backed by Google, ensuring long-term support

Comparison:

- Angular has a larger and more established community.
- Vue.js is rapidly growing and has a supportive community.

2.7.4 Ecosystem

Vue.js:

- Flexible and integrates well with other libraries
- Vue CLI for project scaffolding
- Vue Router and Vuex for state management

Angular:

- Comprehensive suite of built-in tools
- Angular CLI for efficient development
- Angular Material for UI components

Comparison:

- Vue.js offers flexibility in choosing tools and libraries.
- Angular provides a more integrated and consistent development experience.

2.8 Conclusion

Both Vue.js and Angular have their advantages and disadvantages, making them suitable for different types of projects. Vue.js excels in simplicity, flexibility, and ease of integration, making it a good choice for smaller projects or projects that require rapid development. Angular, on the other hand, provides a comprehensive solution for complex and large-scale applications, with a robust set of tools and a well-defined architecture.

The choice between Vue.js and Angular depends on the specific requirements and preferences of a project. Developers should consider factors such as performance needs, developer-friendliness, community support, and the overall ecosystem when making a decision [3].

Chapter 3

Case Study - C

3.1 Introduction to the Case Study

3.1.1 Objective of the Case Study

3.1.2 Application Overview

3.2 Application Description

3.2.1 User Registration and Login

3.2.2 Project Management

3.3 Technical Implementation

3.3.1 Application Architecture

3.3.2 State Management

3.3.3 Routing

3.4 Performance and efficiency

3.4.1 Loading Times and Responsiveness

3.5 Challenges and Solutions

3.5.1 Common Challenges

3.5.2 Framework-Specific Challenges

3.6 Comparison and analysis

3.6.1 Advantages and Disadvantages

3.7 Conclusion from the Case Study

3.7.1 Summary:

3.7.2 Recommendations:

3.7.3 Future Outlook:

Chapter 4

Results - S

4.1 Performance

4.2 Developer-friendliness

4.3 Community support

4.4 Ecosystem

Chapter 5

Conclusion and recommendations - M

5.1 Conclusion

5.2 Recommendations

References

Literature

[1] Ionos. "Die beliebtesten JavaScript-Frameworks und -Bibliotheken" (2023). Abgerufen am 31.05.2024. URL: https://www.ionos.at/digitalguide/websites/web-entwicklung/beliebte-javascript-frameworks-und-bibliotheken/ (cit. on p. 3).

[2] Ionos. "Webframeworks – Überblick und Klassifizierung" (2022). Abgerufen am 31.05.2024. URL: https://www.ionos.de/digitalguide/websites/web-entwicklung/webframeworks-ein-ueberblick/ (cit. on pp. 2, 3).

[3] Kinsta. "Angular vs. Vue: Ein Kopf-an-Kopf-Vergleich" (2023). Abgerufen am 31.05.2024. URL: https://kinsta.com/de/blog/angular-vs-vue/#angular-vs-vue-hnlichkeiten-und-gemeinsamkeiten (cit. on p. 7).

[4] MDN Web Docs. "Introduction to the DOM". *MDN Web Docs* (2024). URL: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction (visited on 06/25/2024) (cit. on p. 3).

Online sources

[5] Vue.js Contributors. *Vue.js*. Accessed: 2024-06-25. 2024. URL: https://vuejs.org/ (visited on 06/25/2024) (cit. on p. 5).

[6] MoldStud. *Creating Dynamic Web Content with JavaScript and jQuery*. Feb. 2024. URL: https://moldstud.com/articles/p-creating-dynamic-web-content-with-javascript-and-jquery (visited on 06/25/2024) (cit. on p. 4).

[7] Mozilla Developer Network. *JavaScript Guide*. Accessed: 2024-06-25. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide (cit. on p. 4).

[8] Angular Team. *Angular*. Accessed: 2024-06-25. 2024. URL: https://angular.io/ (visited on 06/25/2024) (cit. on p. 5).