



GEORG-SIMON-OHM
HOCHSCHULE NÜRNBERG

Web Application Entwicklung mit Vue.JS



Seminararbeit zum Fach Web-Apps mit HTML5 und JavaScript

Fakultät Elektrotechnik Feinwerktechnik Informationstechnik
Technische Hochschule Nürnberg

vorgelegt von: Aayush Yadav
Studiengang: Elektronische und Mechatronische Systeme
Matrikelnummer: 3549452
E-Mail: yadavaa81855@th-nuernberg.de
Prüfer: Prof. Dr. Oliver Hofmann



Inhaltsverzeichnis

1	Einleitung	3
1.1	Lösungsversprechen von Vue	3
2	Vue.js	3
2.2	Vue im Vergleich zu Angular und React	5
2.3	Struktur eines Vue.js Projekts	6
2.4	Das Vue Instanz	7
2.5	Vue Komponente	8
2.6	Instanz-Lebenszyklus Hooks	11
2.7	Vuex	12
2.8	Vue Router	12
3	Evaluation	13
4	Fazit	14
	Abkürzungsverzeichnis	15
	Literaturverzeichnis	16



1 Einleitung

Diese Seminararbeit befasst sich mit der Entwicklung von Web-Applikationen mit Hilfe eines Web-Frameworks VueJS. In der aktuellen Web-Entwicklungslandschaft stehen mehrere Web-Frontend-Frameworks zur Auswahl. In diesem Seminarbeitrag werden die Vorteile der Verwendung von Vue zur Entwicklung anspruchsvoller Web-Anwendungen erläutert.

Frameworks abstrahieren die Interaktion mit dem Browser und dem DOM. Anstatt Elemente zu manipulieren, indem im DOM auf sie verwiesen wird, werden Elemente auf einer höheren Ebene deklarativ definiert und mit diesen interaktiv interagiert. Die Verwendung eines Frameworks ist wie die Verwendung der Programmiersprache C, anstatt die Assemblersprache zum Schreiben von Systemprogrammen zu verwenden.

1.1 Lösungsversprechen von Vue

Vue wird zunächst als ein progressiver Framework bezeichnet. Das bedeutet, dass es sich an die Bedürfnisse des Entwicklers anpasst. Während andere Frameworks eine vollständige Übernahme einer Technologie durch einen Entwickler oder ein Team erfordern und oft möchten, dass eine existierende Anwendung neu geschrieben wird, weil bestimmte Konventionen erforderlich sind. Vue landet zunächst mit einem einfachen Skript-Tag in einer Anwendung und kann mit den Anforderungen wachsen, von einer kleinen Web-Komponente bis hin zur Verwaltung der gesamten Ansichtsebene. Es ist nicht notwendig, über Webpack, Babel, npm oder ähnliches Bescheid zu wissen, um mit Vue zu beginnen. HTML, CSS und Grundlagen von JavaScript reichen aus, um mit Vue anzufangen.

Der Sprung von einer reinen HTML- und CSS-basierten Website zu einer anspruchsvollen Webanwendung ist sehr einfach gehalten und kann im Laufe des Entwicklungsprozesses erlernt werden. Dies ist ein starkes Verkaufsargument, insbesondere im aktuellen Ökosystem der JavaScript-Frontend-Frameworks und -Bibliotheken, das Neueinsteigern und auch erfahrenen Entwicklern das Gefühl gibt, im Ozean der Möglichkeiten und Wahlmöglichkeiten verloren zu gehen. Das Ziel von Vue ist es, mit minimalem Vorkenntnisse vernünftige Web-Applikationen erstellen zu können.

2 Vue.js

Vue ist ein skalierbares JavaScript Framework zum Aufbau von Benutzeroberflächen. Im Gegensatz zu anderen Frameworks ist Vue von Grund auf so konzipiert, dass es schrittweise anpassbar ist. Die Kernbibliothek ist nur auf die Ansichtsebene (View) ausgerichtet und lässt sich leicht in andere Bibliotheken oder bestehende Projekte integrieren. In Kombination mit modernen Tools und zusätzlichen Bibliotheken, ist Vue in der Lage, anspruchsvolle Single-Page-Applikationen zu betreiben [vue20a].

2.1 Haupt Merkmale

Vue ist zusammen mit React und Angular eines der beliebtesten Frameworks in der Webentwicklungslandschaft. Folgende Merkmale zeichnen das Vue-Framework aus:

- Components:** Vue-Komponenten erweitern grundlegende HTML-Elemente, um wiederverwendbaren Code zu kapseln. Das Vue-Komponenten-System ist eine Abstraktion, die die Erstellung großer Anwendungen aus kleinen, in sich geschlossenen und oft wiederverwendbaren Komponenten ermöglicht.
- Templates:** Vue.js verwendet eine HTML/CSS-basierte Template-Syntax. Vue kompiliert die Vorlagen (Templates) in virtuelle DOM-Renderfunktionen. In Kombination mit dem Reaktivitätssystem ist Vue in der Lage, intelligent die minimale Anzahl von Komponenten, die aktualisiert werden müssen, zu ermitteln. Dies minimiert die Anzahl von DOM-Manipulationen bei der Zustandsänderung der Applikation.
- Reaktivität:** Vue verfügt über ein Reaktivitätssystem, welches reine JavaScript-Objekte verwendet und das Rendering optimiert. Jede Komponente behält während des Renderns den Überblick über ihre reaktiven Abhängigkeiten. Das ermöglicht dem System intelligent und optimal die Komponenten neu zu rendern.
- Integrierbarkeit:** Es ist sehr einfach, Vue in bestehende Projekte zu integrieren oder Bibliotheken von Drittanbietern zu Vue hinzuzufügen. Alle JavaScript-Bibliotheken, die ES6+ verwenden, können innerhalb von Vue verwendet werden. Zum Beispiel Bootstrap, Material Design, ThreeJS, Web Sensors API usw. Vue bietet auch die Möglichkeit, TypeScript anstelle von JavaScript zu verwenden.




[vue20d]

2.2 Vue im Vergleich zu Angular und React

Vue wurde von Evan You entwickelt, als er bei Google an AngularJS (Angular 1.0)-Anwendungen arbeitete. Vue übernahm einen Teil der Angular-Templating-Syntax aus, entfernte jedoch den komplexen Stack, den Angular erforderte. Dadurch wurde es sehr leistungsfähig. Das neue Angular (Angular 2.0) löste auch viele der AngularJS-Probleme, jedoch auf sehr unterschiedliche Weise, und erfordert TypeScript, das nicht alle Entwickler gerne verwenden (oder lernen wollen).

Vue hat viele gute Ideen von React übernommen, vor allem das virtuelle DOM. Aber Vue implementiert es mit einer Art automatischer Abhängigkeitsverwaltung, die nachverfolgt, welche Komponenten von einer Änderung des Zustands betroffen sind, so dass nur diese Komponenten neu gerendert werden, wenn sich die Zustands-Eigenschaft ändert. In React hingegen wird bei einer Änderung eines Teils des Zustandes, der Auswirkungen auf eine Komponente hat, diese Komponente neu gerendert, und damit werden auch alle zugehörigen Kinder-Komponente neu gerendert. Dies ist ein Vorteil für Vue in Bezug auf die Benutzerfreundlichkeit und die Leistungssteigerung [vue20b].

Tabelle 2.1: Vue vs React vs Angular

Kriterium	 Vue	 React	 Angular
Fokus	Nutzbarkeit	Flexibilität	TypeScript
Komplexität	Niedrig	Mittel	Hoch
Größe	80 KB	100 KB	500+ KB
Veröffentlichung	2014	2013	2010
Entwickelt von	Evan You	Facebook	Google
Sprache	JavaScript	JavaScript	TypeScript
Model	virtuelle DOM	virtuelle DOM	MVC
Unterstütz von	Open Source	Facebook	Google
Aktuellste Version	2.6.11	16.13.1	9.1.11

[Zon20]



2.3 Struktur eines Vue.js Projekts

Vue hat eine sehr intuitive Projektstruktur. Zusätzlich zum Kern des Frameworks enthält es eine Menge Utilities, die die Frontend-Entwicklung mit Vue sehr angenehm machen. Wie meisten JavaScript Projekte wird für Vue-Entwicklung NodeJS verwendet. Vue stellt die Kernbibliothek und die Zusatz Utilities über **npm** (node package manager) zur Verfügung.

NodeJS ist eine JavaScript-Laufzeitumgebung, die auf der V8-JavaScript-Engine von Chrome basiert. Sie ermöglicht die Ausführung von JavaScript-Code außerhalb eines Webbrowsers. NodeJS ermöglicht es Entwicklern, JavaScript zum Schreiben von Kommandozeilen-Tools und für die serverseitige Ausführung von Skripten zu verwenden, um dynamische Webseiteninhalte zu erzeugen, bevor die Seite an den Webbrowser des Benutzers gesendet wird. NodeJS unterstützt die Vereinheitlichung der Web-Anwendungsentwicklung um eine einzige Programmiersprache herum, anstatt verschiedene Sprachen für server- und clientseitige Skripte zu verwenden [Nod20].

npm (node package manager) ist die größte Software-Registry der Welt. Open-Source-Entwickler aus allen Kontinenten nutzen npm, um Pakete gemeinsam zu nutzen und auf sie zuzugreifen, und viele Organisationen nutzen npm auch zur Verwaltung der privaten Entwicklung [npm20].

Vue-CLI: Vue bietet eine offizielle CLI(Command Line Interface) für das schnelle Einrichten anspruchsvoller einseitiger Anwendungen. Es bietet einen umfassenden Build-Setup für einen modernen Frontend-Workflow. Es dauert nur wenige Minuten, um mit Hot-Reload, Lint-on-Save und Produktions-Builds in Betrieb zu gehen.

Installation Die Installation der erforderlichen Pakete wird mit NodeJS und npm durchgeführt. Folgende Befehle müssen ausgeführt werden um ein Basis Vue Anwendung zu erstellen:

- Die Vue-CLI global installieren:

```
1 | npm install -g @vue/cli
```

- Die Vue-CLI wird verwendet um einen neuen Vue-Projekt zu erstellen.

```
1 | vue create beispiel_vue_app
```

Dadurch wird eine CLI geöffnet, in der die gewünschte Konfiguration für ein Vue-Projekt eingestellt werden kann.

2.4 DAS VUE INSTANZ

- Um den Entwicklungsserver zu starten, wird der folgende Befehl ausgeführt:

```
1 | npm run serve
```

- Um einen Production-Build für die App zu erzeugen, wird folgender Befehl ausgeführt:

```
1 | npm run build
```

Ein typisches Vue-Projekt mit Vue-Router und Vuex hat die folgende Struktur:

Tabelle 2.2: Struktur des Projektordners

Dateien/Ordner	Beschreibung
public/index.html	Dies ist die Haupt-App-Datei, die vom Browser geladen wird. Die Datei enthält im Body nur ein einfaches HTML-Tag: <code><div id="app"> </div></code> . Dies ist das Element, mit dem die Vue-Anwendung an das DOM angehängt wird.
src/main.js	Dies ist die JavaScript-Datei, die für die Konfiguration der Vue.js Anwendung verantwortlich ist. Es wird auch verwendet, um alle Pakete von Drittanbietern zu registrieren.
src/App.vue	Dies ist die Root-Komponente, die den HTML-Inhalt enthält, der dem Benutzer angezeigt wird.
/package.json	In dieser Datei speichert npm die Namen und Versionen des Pakets, das es installiert hat.
src/components/	Dieser Ordner enthält die zusätzlichen Komponenten.
src/assets/	Dieser Ordner wird verwendet, um statische Inhalte, wie Bilder zu speichern.
src/router/	Der Ordner enthält die Implementierung von Vue-Router.
src/store/	Der Ordner enthält die Implementierung von Vuex-Store.

2.4 Das Vue Instanz

Jede Vue-Anwendung beginnt mit der Erstellung einer neuen Vue-Instanz (Root Instance) mit der Vue-Funktion. Wenn eine Vue-Instanz erstellt wird, fügt sie dem Reaktivitätssystem von Vue alle in ihrem Datenobjekt(beschrieben in Tabelle 2.3) gefundenen Eigenschaften hinzu. Alle zu verwendende Vue.js internen Module und externen Plugins werden in der Datei (`main.js`) mit ES6(ECMAScript 2015) Import Syntax ('import') importiert. Die Abhängigkeiten sind als

globale Parameter definiert und sind von allen Komponenten zugänglich. In der main.js Datei wird die Haupt-Vue-Komponente (App.vue) registriert.

```
1  import Vue from 'vue'
2  import App from './App'
3  import router from './router'
4
5  // Root Instance
6  new Vue({
7    el: '#app',
8    router,
9    template: '<App/>',
10    components: { App }
11  })
```

Listing 2.1: main.js

2.5 Vue Komponente

Das wichtigste Kriterium für die Auswahl dieses Frameworks (Vue.js) ist die Modularität der einzelnen Komponenten (beschrieben im Abschnitt 2.1). Die Komponenten sind eine der stärksten Eigenschaften von Vue.js. Sie ermöglichen, grundlegende HTML-Elemente zu erweitern, um wiederverwendbaren Code zu kapseln. Auf einer hohen Ebene sind Komponenten benutzerdefinierte Elemente, an die der Vue.js Compiler ein bestimmtes Verhalten anhängt [vue20a]. Jede Komponente wird in einer Datei mit .vue Erweiterung definiert. Die Definition enthält ein HTML basiertes Template, die die Gestaltung der Komponente festlegt und den JavaScript Code, das alle andere Funktionalitäten bzw. Verhalten der Komponente spezifiziert. Jede Komponente ist ebenfalls eine Vue-Instanz.

Üblicherweise wird eine Vue.js App in einem Baum aus verschachtelten Komponenten organisiert. Beispielsweise besteht sie aus Komponenten für einen Header, eine Seitenleiste und einen Inhaltsbereich, die jeweils andere Komponenten für Navigationslinks, Blog-Posts usw. enthalten. Die Komponenten Architektur vereinfacht solche Verschachtelung.

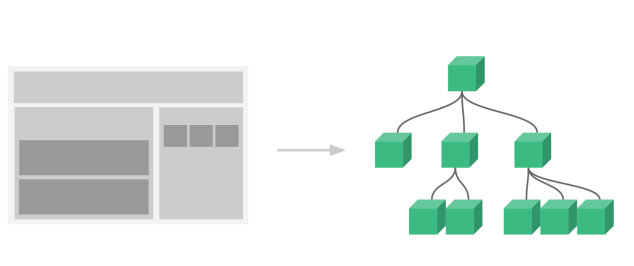


Abbildung 2.1: Organisation von Komponenten [vue20c]

2.5 VUE KOMPONENTE

Das Template-Element beinhaltet das HTML-Template und das Script-Element beinhaltet den JavaScript-Teil der Komponente.

Template-Element: Die Gestaltung bzw. das Design der Benutzeroberfläche wird in diesem Element implementiert. Dafür wird Standard HTML und CSS Technik verwendet. Es können sowohl benutzerdefinierte HTML-Elemente als auch HTML-Elemente aus externen Plugins sehr einfach integriert werden. Die HTML-basierte Template-Syntax von Vue.js ermöglicht es, das gerenderte DOM deklarativ an die Daten der zugrunde liegenden Vue-Instanz zu binden. Alle Vue.js-Templates sind gültiges HTML.

Die einfachste Form der Datenbindung ist die Textinterpolation unter Verwendung der „Mustache“-Syntax (doppelte geschweifte Klammern):

```
1      <template>
2          <div>
3              <h1>Data: {{ beispielData }}</h1>
4              <BeispielComponent/>
5          </div>
6      </template>
```

Listing 2.2: Templateelement

Script-Element: Dieses Element beinhaltet die Implementierung der gewünschten Funktionalitäten der Komponente mit VueJS spezifischem Konstrukt. Weitere Komponenten können hier importiert und registriert werden. Diese stehen dann im Template-Element zur Verfügung, welches die Verschachtelung von Komponenten realisieren lässt.

```
1      <script>
2
3          import BeispielComponent from 'components/
4              BeispielComponent'
5          export default {
6              components: {
7                  BeispielComponent
8              },
9              props : [],
10             data () {
11                 return {
12                     beispielData: 'Beispiel String'
13                 }
14             },
15             methods: {
16                 beispielFunction: function () { }
```



2.5 VUE KOMPONENTE

```
17         computed: {  
18             }  
19     }  
20 </script>
```

Listing 2.3: Skriptelement

Die wichtigsten Teile einer Komponente sind in der Tabelle 2.3 beschrieben.

Tabelle 2.3: Parameter zuständig für Datenmanipulation

Parameter	Typ	Beschreibung
components	[key: string]: Object	Dies ist eine Liste von zu verwendenden benutzerdefinierten Komponenten. Die benutzerdefinierten bzw. externen Komponenten werden mit import Befehl aus ES6 Spezifikation importiert.
data	Object Function	Dies ist ein JavaScript Objekt oder eine Funktion, die die Speicherung der notwendigen Attribute für die Komponente erlaubt. Innerhalb der Komponente kann auf das ursprüngliche Datenobjekt mit this.data Direktive zugegriffen werden.
props	Array<string> Object	Dies ist eine Liste/Hash von Attributen, die zur Aufnahme von Daten aus der übergeordneten Komponente vorgesehen sind. Es hat eine einfache Array-basierte Syntax und eine alternative Objekt-basierte Syntax, die erweiterte Konfigurationen wie Typprüfungen, benutzerdefinierte Validierung und Standardwerte ermöglicht.
methods	[key: string]: Function	Dies ist eine Liste von Methoden, die die Realisierung verschiedener Funktionalitäten der Komponente erlauben. Alle Methoden können innerhalb der Komponente als eine standard JavaScript Methode aufgerufen werden.
computed	[key: string]: Function	Dies ist eine Liste von Methoden, die automatisch bei Änderung der reaktiven Abhängigkeiten aufgerufen wird. Die Attribute innerhalb dieser Parameter werden zwischengespeichert und nur bei reaktiven Abhängigkeitsänderungen neu berechnet.

2.6 Instanz-Lebenszyklus Hooks

Bei der Erstellung durchläuft jede Vue-Instanz eine Reihe von Initialisierungsschritten, z. B. muss sie die Datenbeobachtung einrichten, die Template kompilieren, die Instanz in das DOM einbinden und das DOM aktualisieren, wenn sich die Daten ändern. Parallel dazu werden auch Funktionen ausgeführt, die als Lifecycle-Hooks bezeichnet werden und den Benutzern die Möglichkeit geben, in bestimmten Phasen des Instanz-Lebenszyklus ihren eigenen Funktionen hinzuzufügen. Dies ermöglicht eine sehr präzise Steuerung des Verhaltens einzelner Komponenten.

Der „created“ Hook kann beispielsweise zur Ausführung von Code nach der Erzeugung einer Instanz verwendet werden. Es gibt auch andere Hooks, die in verschiedenen Phasen des Lebenszyklus der Instanz aufgerufen werden, wie z.B. „mounted“, „updated“ und „destroyed“. Die Abbildung 2.2 illustriert die Details des kompletten Lifecycle-Hooks.

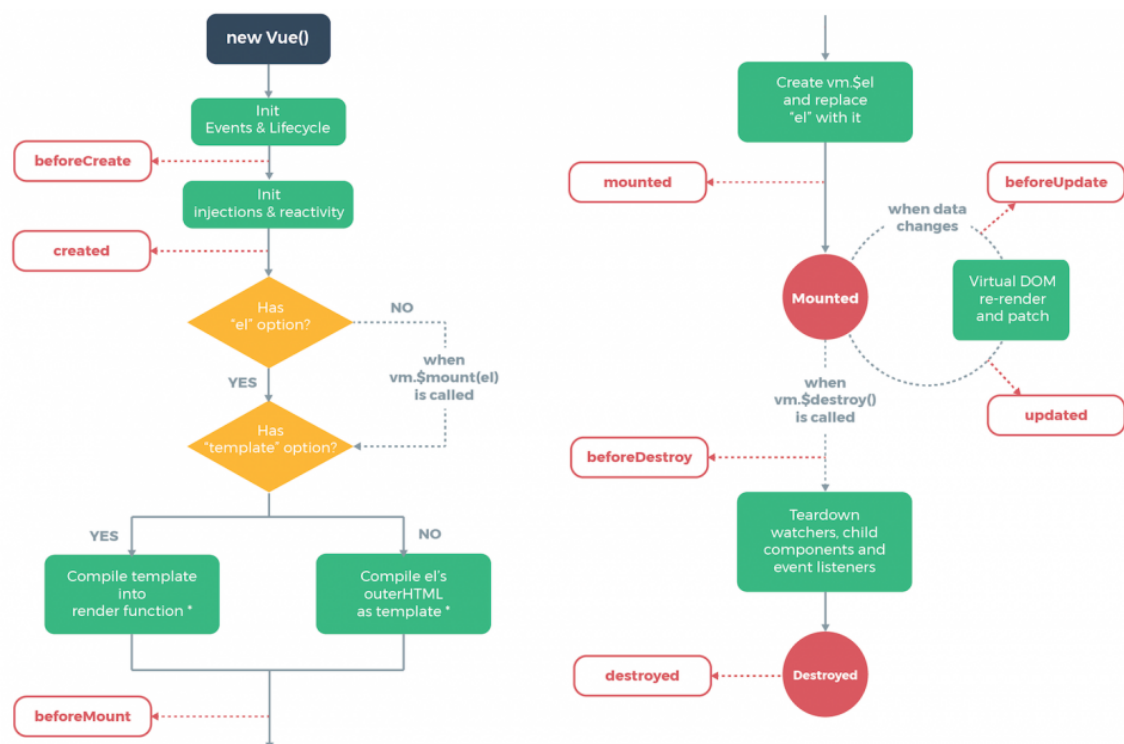


Abbildung 2.2: Vue Instanz-Lifecycle Hooks [vue20e]



2.7 Vuex

Vuex ist die offizielle Zustandsverwaltungsbibliothek für Vue. Ihre Aufgabe besteht darin, Daten zwischen den Komponenten einer Vue Anwendung auszutauschen.

Komponenten in einer Vue-Anwendung können ihren eigenen Zustand haben. Beispielsweise speichert ein Eingabefeld die darin eingegebenen Daten lokal. Komponenten können eine Eingabefeld sein oder eine ganze Seite. Es wird sehr aufwendig die Zustandsdaten zwischen verschachtelten Komponenten auszutauschen.

Vuex bietet einen zentralen Repository-Speicher(Vuex-Store) für den Zustand, und der Zustand kann durch den Aufruf von im Vuex-Store definierten Funktionen geändert werden. Jede Komponente, die von einem bestimmten Teil des Zustands abhängt, greift auf diesen über einen getter Funktion auf dem Vuex-Store zu, wodurch Synchronisierung der Daten sichergestellt wird. Vuex-Store ist eine Art temporäre Datenbank für eine bestimmte Anwendungssitzung [vue20d].

2.8 Vue Router

In einer Webanwendung ist ein Router der Teil, der die aktuell angezeigte Ansicht mit dem Inhalt der Browser-Adressleiste synchronisiert.

Ein Router wird benötigt, wenn URLs mit den Ansichten in einer Anwendung synchronisiert werden müssen. Dies ist eine sehr häufige Notwendigkeit, und alle wichtigen modernen Frameworks erlauben inzwischen die Verwaltung des Routings. Vue Router ist Teil der Vue Kern Bibliothek und übernimmt die Routenverwaltung in einer Vue-Anwendung [vue20d].

3 Evaluation

Als Prototyp zum Testen der oben genannten Technologie wurde eine Einkaufsliste App entwickelt. Das Grundlayout wurde unter Verwendung von Standard-HTML und CSS entwickelt, und der Inhalt der Einkaufsliste wurde mit der Textinterpolations-Funktion von Vue implementiert. Jedes Element in der Einkaufsliste ist eine Vue-Komponente. Die entwickelte App verwendet die von Vue angebotene verschachtelte Komponentenarchitektur. Vuex wird verwendet, um die Daten in der Einkaufsliste zu verwalten. Der Quellcode der Prototyp-Anwendung, die mit dieser Technologie entwickelt wurde, ist in diesem Git-Repository zu finden: https://git.efi.th-nuernberg.de/gitea/yadavaa81855/vue_pwa.git

Es war insgesamt eine sehr angenehme Erfahrung, die App zu entwickeln. Wirklich gute Dokumentation und einfache Anwendbarkeit des Frameworks machen die Lernkurve ziemlich flach. Die Integration von Drittbibliotheken ist nicht kompliziert. Es war nicht notwendig, alles über das Framework zu lernen, um eine einfache App zu erstellen, und neue Funktionen können der App mit Hilfe dieses Frameworks leicht hinzugefügt werden.

Tabelle 3.1: Bewertung von VueJS anhand entwickelte Prototyp

Kriterien	Bewertung
Lernkurve	In der Entwicklung überzeugt Vue.js durch die gute Erlernbarkeit der verschiedenen Konzepte, die aufeinander aufbauen.
Modularität	Das Kern-Framework ist also sehr schlank und überschaubar und hat nur wenige Grundbausteine. Zusätzliche Funktionalitäten können mit Hilfe von Zusatzmodulen hinzugefügt werden
Skalierbarkeit	Es kann verwendet werden, um kleine Websites bis hin zu anspruchsvollen Webanwendungen zu erstellen.
Ökosystem	Vue.js verfügt über ein umfassendes Ökosystem mit Unterstützung für Bibliotheken und Plugins von Drittanbietern.



4 Fazit

Vue.js ist vielleicht nicht das am weitesten entwickelte Framework, aber es erfüllt manche Anforderungen vollkommen richtig. Erstens können sehr schnell und ohne großen Aufwand schöne Widgets und Anwendungen geschrieben werden. Die Lernkurve ist niedrig, das Belohnungsgefühl ist enorm hoch. Anfänger werden ihren Spaß mit Vue.js haben, vor allem weil alles so überschaubar ist. Vue.js skaliert auch mit den Bedürfnissen des Teams und dem Wissensstand des Entwicklers und wird von der Entwickler-Community gut angenommen.

Die Komponentenarchitektur von Vue.js hilft, die Anwendung so zu strukturieren, dass es einfach ist, durch die Teile der Anwendung zu navigieren, die von anderen Entwicklern entwickelt wurden. Die robusten APIs ermöglichen es, sich auf die Implementierung der Geschäftslogik zu konzentrieren, anstatt sich mit dem Framework zu befassen.

Vue.js verfügt über eine ausgezeichnete Online-Dokumentation mit vielen Beispielen. Zusätzlich gibt es einen Vue.js Style Guide, der Best Practices zeigt, ein Vue.js-Kochbuch, das häufig verwendete Muster beschreibt, und eine Übersicht über zusätzliche Bibliotheken und Vue.js-Erweiterungen.



Abkürzungsverzeichnis

HTML	HyperText Markup Language
CSS	Cascading Style Sheets
URL	Uniform Resource Locator
MVC	Model View Controller
CLI	Command Line Interface
npm	Node Package Manager
API	Application Programming Interface
DOM	Document Object Model



Literaturverzeichnis

- [Nod20] NODEJS.ORG: *NodeJS Homepage*. <https://nodejs.org/en/>, 2020 (accessed June 24, 2020)
- [npm20] NPMJS.COM: *NPM Homepage*. <https://www.npmjs.com/>, 2020 (accessed June 24, 2020)
- [vue20a] VUEJS.ORG: *Vue.js Documentation*. <https://vuejs.org/v2/guide/index.html>, 2019 (accessed January 30, 2020)
- [vue20b] VUEJS.ORG: *Comparison with Other Frameworks*. <https://vuejs.org/v2/guide/comparison.html>, 2020 (accessed June 24, 2020)
- [vue20c] VUEJS.ORG: *Components Basics*. <https://vuejs.org/v2/guide/components.html>, 2020 (accessed June 24, 2020)
- [vue20d] VUEJS.ORG: *Vue Guide*. <https://vuejs.org/v2/guide/>, 2020 (accessed June 24, 2020)
- [vue20e] VUEJS.ORG: *The Vue Instance*. <https://vuejs.org/v2/guide/instance.html>, 2020 (accessed June 24, 2020)
- [Zon20] ZONE, Web D.: *React vs. Angular vs. Vue.js: A Complete Comparison Guide*. <https://dzone.com/articles/react-vs-angular-vs-vuejs-a-complete-comparison-gu>, 2020 (accessed June 24, 2020)